



ICONICS
100 Foxborough Blvd.
Foxborough, MA 02035
Tel: 508-543-8600
Fax: 508-543-1503
E-Mail: support@iconics.com
Web: www.iconics.com

© ICONICS, Inc. All rights reserved.

Содержание

Глава 1: Начальные сведения
Глава 2: Настройка системы
Глава 3: Управление файлами экранных форм
Глава 4: Создание и редактирование объектов
Глава 5: Функции редактирования экранной формы
Глава 6: Просмотр экранных форм
Глава 7: Установление соединений с источниками данных
Глава 8: Динамизация объектов экранных форм
Глава 9: Сервис
Глава 10: Среда исполнения
Глава 11: Мастера VBA
Глава 12: GraphWorX32 ActiveX
Глава 13: Справочник по OLE Automation
Глава 14: Проект VBA в GraphWorX32

Глава 1 Начальные сведения.

1.1 Введение в GraphWorX32
1.2 Начало работы с GraphWorX32
1.3 Экран GraphWorX32
1.4 Использование мыши и клавиатуры.

1.1 Введение в GraphWorX32

Назначение GraphWorX32

GraphWorX32 является программным пакетом, который предназначен для представления параметров технологических процессов на графических мнемосхемах. GraphWorX32 представляет собой полностью соответствующее спецификации OPC клиентское приложение, основанное на технологиях ActiveX и OLE. Характерные особенности приложения:

32-разрядная архитектура

GraphWorX32 является 32-разрядным многопоточным приложением, которое может исполняться под управлением операционных систем Windows 95/98, Windows NT и Windows 2000 на платформах Intel и Digital Alpha.

Универсальный интерфейс OPC

GraphWorX32 является клиентским приложением, соответствующим спецификации OLE for Process Control (интерфейс связывания и внедрения объектов для систем сбора данных и управления).

Мощные средства разработки экранных форм визуализации параметров технологических процессов

GraphWorX32 обеспечивает полный набор инструментов для создания объектно-ориентированной динамической графики с возможностью масштабирования экранных форм. При изменении размеров окна экранной формы происходит автоматический подбор размеров элементов отображения и графических объектов.

Контейнер для управляющих элементов ActiveX и OLE

Имеется возможность вставки в экранные формы GraphWorX32 управляющих элементов ActiveX и объектов OLE различных производителей.

Возможность Сервер ActiveX

Экранные формы могут просматриваться в контейнерах ActiveX, таких как Microsoft Internet Explorer.

публикации экранных форм в Интернет/Интранет

Экранные формы GraphWorX32 могут публиковаться в Интернет (глобальной) и Интранет (локальной) сетях и просматриваться при помощи MS Internet Explorer 4 и 5

Развитый интерфейс OLE автоматизации

Экранные формы и графические объекты экранных форм GraphWorX32 могут управляться из других приложений посредством набора методов и свойств OLE автоматизации.

Высокая скорость обновления графики

Время обновления графической информации GraphWorX32 составляет 50 мс.

Обширная библиотека символов технологической графики

GraphWorX32 включает в себя программу для работы со встроенной библиотекой символов объектно-ориентированной технологической графики, которая может быть расширена пользователем.

Поддержка переключения языков.

Изменение языка интерфейса оператора без корректировки проекта.

Загрузка ActiveSync в Windows CE и Pocket PC.

Режим разработки для Pocket PC; поддержка Compaq, Casio, and HP

Функции установки приоритета потоков данных и таймеров.
Возможность назначения функциональных клавиш и запрещение выбранных клавиш с помощью ICONICS Secure Desktop Configurator. Защита экранных форм и символов, разработанных пользователями, паролем.
Пользовательская настройка меню и панелей инструментов.
Поддержка цифровой клавиатуры.
Панель инструментов для удобного доступа к командам масштабирования и управления слоями.
Удобные средства работы со слоями, сходные с AutoCAD.
Работа с псевдонимами.
Утилита TraceWorX32 для отладки и диагностики.
Встроенная Библиотека символов с поддержкой OLE Automation.
Встроенные редакторы VBScript и Jscript
Мастер Web-публикаций для экспорта экранных форм в HTML и их публикации.
Обратная совместимость с V7.
Обработка событий от мыши и клавиатуры в VBA.
Прозрачность и полупрозрачность.
Пределы масштабирования.
Совместимость с OLExpress 6.0.
Свободное вращение объектов.
Функции редактирования источников данных.

Новые возможности и функции:

1. Максимальный размер экрана 30,000 x 30,000 пикселей
2. Интеграция с VCRWorx – воспроизведение исторических данных.
3. Новая технология Доступа к Данным.
   Встроенные подключения к базам данным- как к OPC-тегам.
   Поддержка соединений с открытыми базами данных (MS Access, Ms SQL Server, Oracle 8.x и 9.x, MSDE, а также ODBC и OLEDB совместимые базы данных)
4. Добавлена новая функция «Формат по образцу» (как в MS Office)
5. Новые функции Конфигуратора подтверждающих сообщений, в т.ч. возможность изменения сообщений при редакции свойств объекта.
6. Интеграция с Универсальным Навигатором Данных
7. Возможность FTF-загрузки при публикации проекта
8. Оптимизация работы системы глобальных псевдонимов
9. Опция отключения сообщений сервера безопасности и диалога входа в систему
10. Использование в выражениях имени пользователя, находящегося в системе, и группы
11. Перепривязка псевдонимов при обновлении связанных объектов
12. Новые операции кнопок: исполнение отчета ReportWorX, исполнение транзакции BridgeWorX, обновление баз данных с помощью элемента управления Data Mining, загрузка VCRWorX32 для клиента WebHMI.
13. Отображение текущего масштабирования как встроенной локальной переменной или свойства объекта
14. Новые возможности создания пользовательских подсказок для объектов GraphWorX
15. Новые возможности клавиатуры в режиме исполнения (запоминание позиции курсора)
16. Новые возможности в разработке сценариев, в т.ч. отключение сценариев при старте/загрузке
17. Новые возможности интерфейса OLE Automation

18. Возможность прописки относительного пути к файлу
19. Возможность экспорта диалога редактирования соединений
20. Быстрая привязка псевдонимов при обновлении связанных объектов
21. Возможность сохранять шаблоны без VBA
22. Существенно улучшена функция привязки глобальных псевдонимов благодаря возможности кэширования данных.
23. Увеличение скорости открытия всплывающих окон
24. Новый инструмент «Вставить последний помещенный на экране объект».
25. Добавлен метод OLE Automation для установки размера и типа шрифта существующих текстовых объектов
26. Добавлены новые функции OLE Automation для получения локальных переменных и псевдонимов из экранной формы (GetLocalVariables, GetLocalAliases)
27. Новый улучшенный объект Дата/Время для отображения времени при воспроизведении архивных данных через VCRWorX

Мастера VBA.

- **Помощь.** Подробная информация об использовании мастеров VBA.
- **Мастер OPC Data Access.** Опрашивает и устанавливает данные OPC.
- **Мастер Microsoft Excel**. Заносит и извлекает данные из Excel.
- **Мастер отчетов.** Создает пользовательские отчеты.
- **Мастер рецептов.** Создает рецепты.
- **Мастер Microsoft Word.** Вносит данные в MS Word.
- **Мастер E-mail и Outlook.** Посылает сообщения в зависимости от событий VBA.
- **Мастер OPC Calculator.** Подключает к тегу калькулятор с четырьмя действиями.
- **Мастер OPC KeyPad.** Подключает к тегу клавиатуру QUERTY.
- **Мастер настройки псевдонимов.** Изменяет настройки псевдонимов в режиме исполнения.
- **Мастер аналоговых диаграмм.** Создает экран с аналоговыми OPC-тегами.
- **Мастер дискретных диаграмм.** Создает экран с дискретными OPC-тегами.

1.2 Начало работы с GraphWorX32

Запуск GraphWorX32 выполняется, как показано на рис. 1-1.

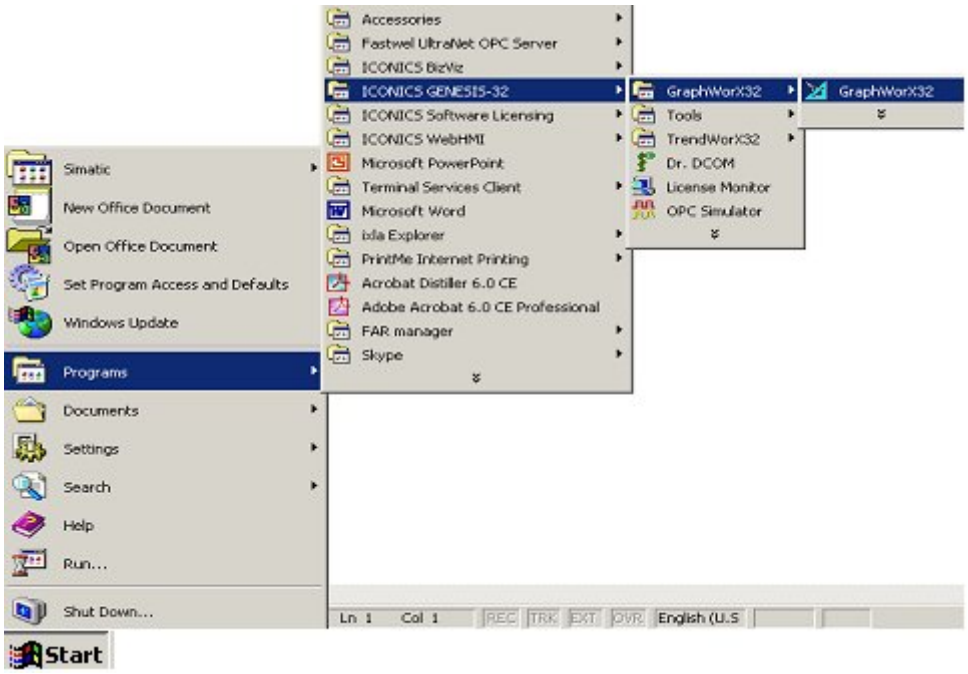


Рис. 1-1. Запуск GraphWorX32

1.3 Экран GraphWorX32

1.3.1 Главное окно GraphWorX32

Внешний вид главного окна GraphWorX32 с его основными компонентами, включая панель цветовой палитры, главную панель инструментов, набор инструментов рисования, упорядчивания, шрифтов и динамики, показан на рис. 1-2.

Настоящий раздел содержит описание основных элементов главного окна GraphWorX32, инструментов для создания экранных форм и установления динамических связей со значениями контролируемых параметров. Для получения информации о правилах работы с основными элементами интерфейса пользователя операционной системы Windows обратитесь к руководству пользователя на применяемую операционную систему или справочной литературе.

1.3.2 Область отображения GraphWorX32

Рабочей областью является внутренняя область главного окна GraphWorX32, которая предназначена для создания и просмотра экранных форм GraphWorX32. *Область отображения* содержит только видимую часть всей рабочей области. Просмотр остальных частей рабочей области выполняется при помощи горизонтальной и вертикальной полос прокрутки. Кроме того, имеется возможность просмотра различных частей рабочей области путем использования команд **Масштаб** или **Показать весь экран** меню **Вид**.

После сохранения экранной формы, информация о размерах области отображения сохраняется вместе с экранной формой. В результате при повторной загрузке данной экранной формы размеры области отображения будут восстановлены.

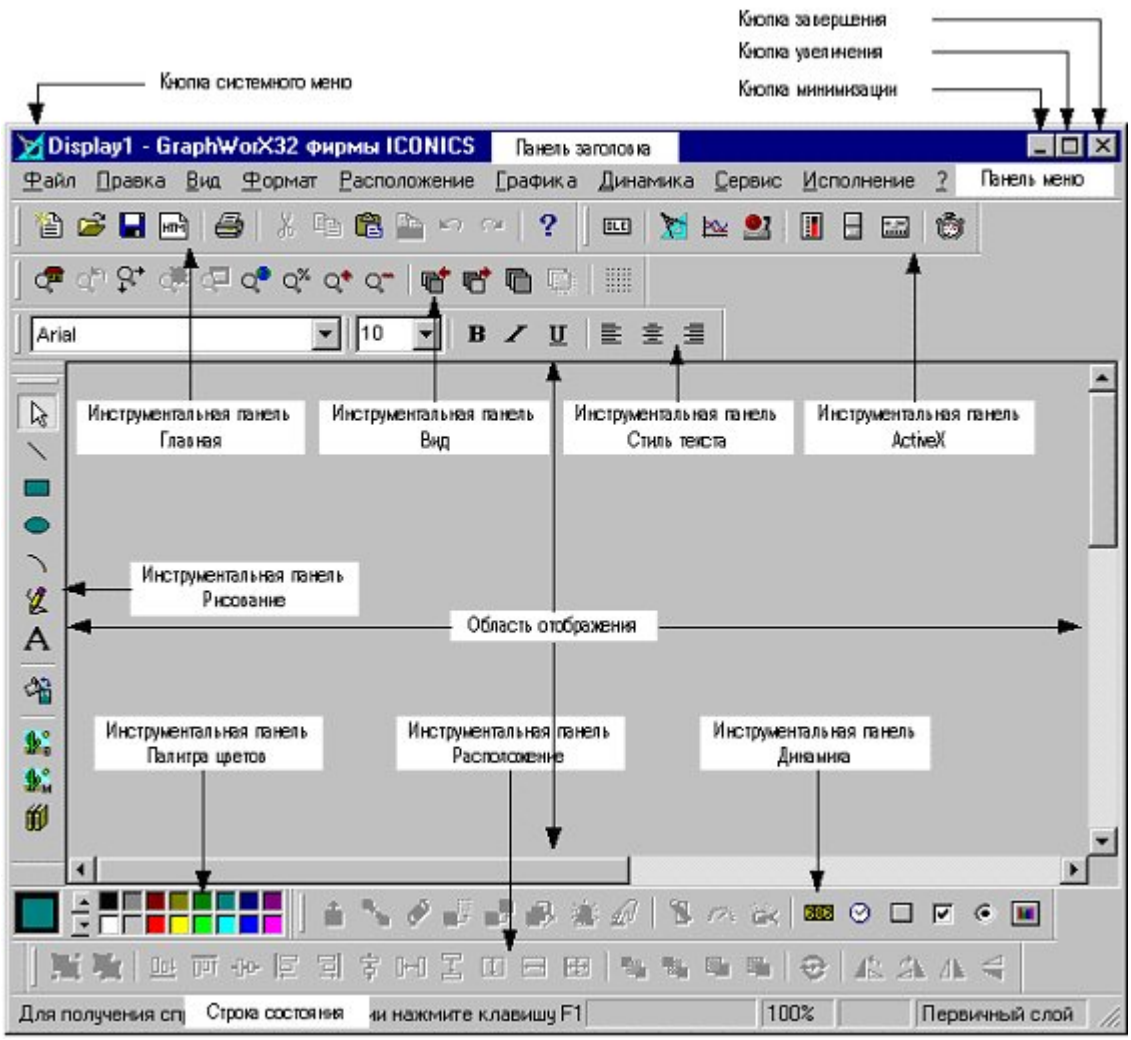


Рис. 1-2. Внешний вид главного окна GraphWorX32

1.3.3 Основные понятия и определения

Размер экрана определяет границы экранной формы в абсолютных координатных единицах. Размер экрана может быть установлен пользователем.

Размеры окна состоят из совокупности параметров, определяющих высоту, ширину и положение главного окна GraphWorX32 на экране монитора.

Масштабируемость экранной формы означает, что при изменении размеров главного окна GraphWorX32 происходит автоматическое изменение размеров графических объектов внутри области отображения, в результате чего все графические объекты области отображения остаются видимыми.

Сохранение пропорций означает, что соотношение геометрических размеров графических объектов экранной формы остаются неизменными при изменении размеров главного окна GraphWorX32.

Фиксированный масштаб означает, что размеры области отображения остаются неизменными при изменении размеров главного окна GraphWorX32. После уменьшения или увеличения окна происходит соответственно усечение или раскрытие графических объектов.

Областью отображения является видимая часть рабочей области экранной формы.

Основной вид – область отображения, размеры которой были сохранены при последнем выполнении операции сохранения экранной формы на диске.

Масштаб 100% – масштаб, при котором логическая единица координат экранной формы равна одному пикселу. Если масштаб равен 100%, то указанное правило не зависит от

того, выбран ли режим сохранения пропорций или фиксированный масштаб для экранной формы.

1.3.4 Строка состояния

Строка состояния находится в нижней части главного окна GraphWorX32. Для скрытия или отображения строки состояния воспользуйтесь командой **Строка состояния** меню **Вид**.

Левая область строки состояния содержит краткую информацию о текущем выполняемом действии. Для вывода краткой информации о назначении команды меню или кнопки в какой-либо панели инструментов следует выбрать объект, о котором требуется получить краткую информацию, нажать и не отпускать левую клавишу мыши. Если после просмотра информации о команде меню или кнопке в панели инструментов не требуется выполнять соответствующее действие, следует переместить курсор мыши в поле окна вне команды меню или кнопки и отпустить левую клавишу мыши.

В четырех полях правой области строки состояния выводится следующая информация:

- Текущие координаты курсора мыши
- Текущий масштаб
- Включена ли привязка графических объектов экранной формы к координатной сетке
- Название текущего отображаемого слоя экранной формы.

1.3.5 Панели инструментов

Инструменты GraphWorX32, используемые для создания экранных форм, объединены в функциональные группы и доступны в соответствующих меню главного окна, а также в панелях инструментов, описание которых приведено ниже.

1.3.5.1 Главная панель инструментов (**Main**)

Внешний вид главной панели инструментов показан на рис. 1-3.

Данная панель инструментов по умолчанию располагается в верхней части окна GraphWorX32 под заголовками меню и обеспечивает возможность быстрого доступа к основным операциям над объектами экранной формы с помощью мыши. К указанным операциям относятся: создание нового файла экранной формы, открытие файла существующей экранной формы, сохранение экранной формы в файле, экспорт экранной формы в HTML-страницу, печать файла экранной формы, операции удаления объектов с помещением в универсальный буфер обмена (операция Вырезать), вставки, дублирования объектов, а также новая функция **«Формат по образцу».** Кроме того, в данной панели инструментов доступны операции отмены последнего действия по редактированию графических объектов экранной формы, повторения последней отмененной операции, выполнение сценариев, а также вызов справочной системы GraphWorX32.

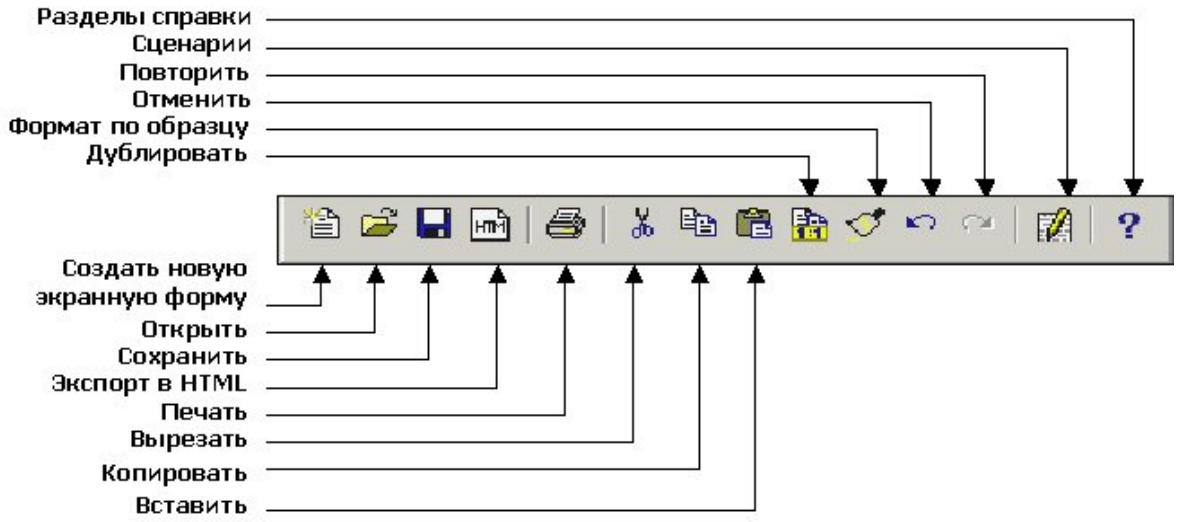


Рис. 1-3. Внешний вид панели инструментов Главная (Main)

1.3.5.2 Панель инструментов *Динамика* (**Dynamics**)

Внешний вид панели инструментов Динамика показан на рис. 1-4.

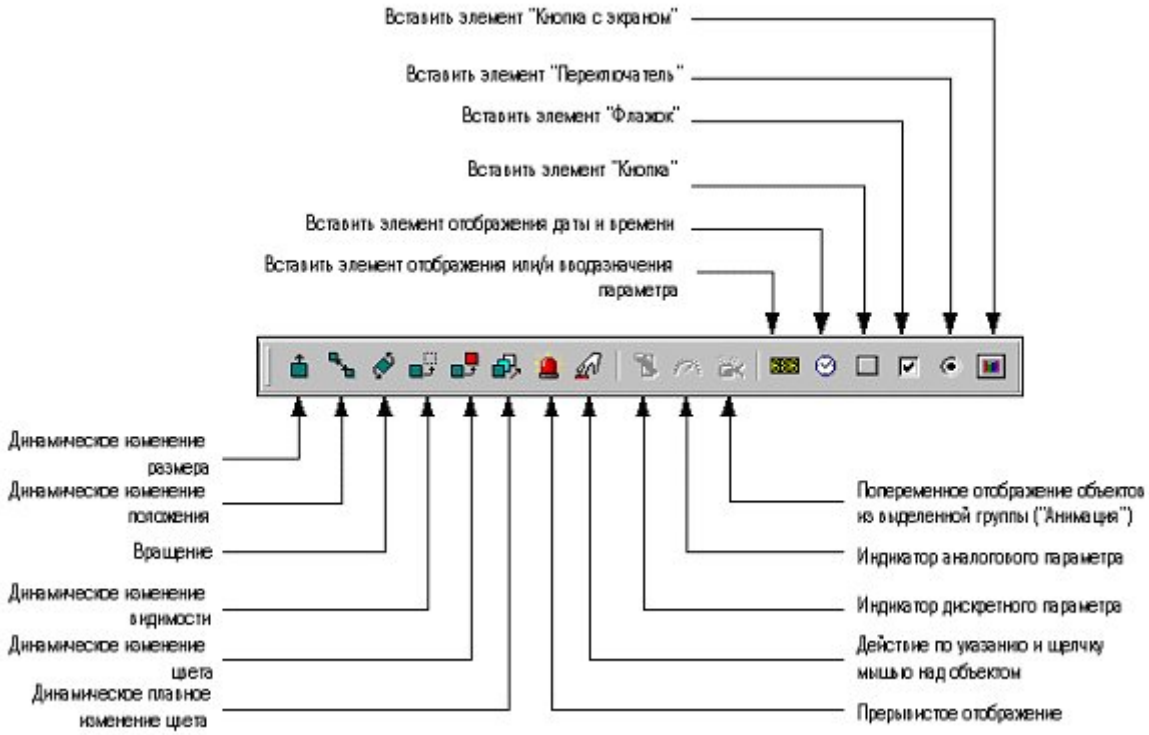


Рис. 1-4. Внешний вид панели инструментов Динамика (Dynamics)

Панель инструментов *Динамика* предназначена для установления динамических связей между графическими объектами экранной формы и переменными в серверах OPC, представляющими контролируемые технологические параметры. Для получения более подробной информации о командных кнопках панели инструментов Динамика обратитесь к разделу 9 “Динамические соединения” настоящего руководства.

1.3.5.3 Панель инструментов *ActiveX*

Внешний вид панели инструментов *ActiveX* показан на рис. 1-5.

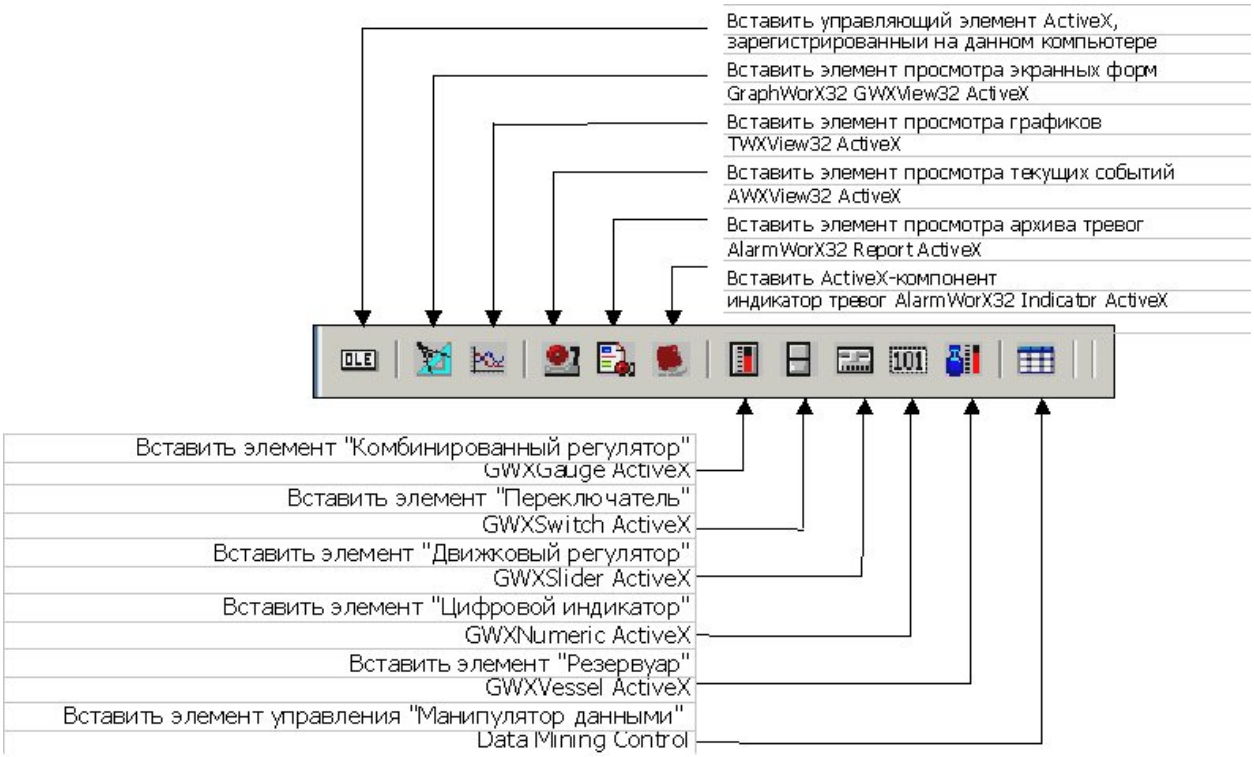


Рис. 1-5. Внешний вид панели инструментов ActiveX

Данная панель инструментов предназначена для вставки в экранную форму управляющих элементов ActiveX фирмы ICONICS, а также любых управляющих элементов ActiveX и объектов OLE других производителей.

1.3.5.4 Панель инструментов *Рисование* (**Draw**)

Внешний вид панели инструментов *Рисование* показан на рис. 1-6.

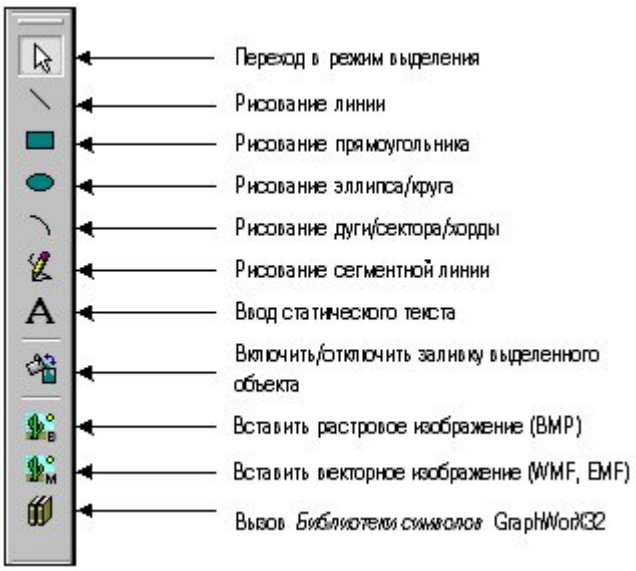


Рис. 1-6. Внешний вид панели инструментов Рисование

Данная панель инструментов предназначена для рисования статических графических объектов, выбора (выделения) ранее нарисованных объектов, заливки графических

объектов и текстовых строк, для вставки в экранную форму векторных изображений в формате WMF/EMF и растровых изображений (BMP), а также для библиотечных символов GraphWorX32.

1.3.5.5 Панель инструментов *Расположение* (**Arrange**)

Внешний вид панели инструментов *Расположение* показан на рис. 1-7.
Данная панель инструментов используется для объединения нескольких выделенных графических объектов в один символ, а также для разбиения символов на отдельные графические объекты, для выравнивания и вращения объектов, а также для изменения их взаимного расположения.

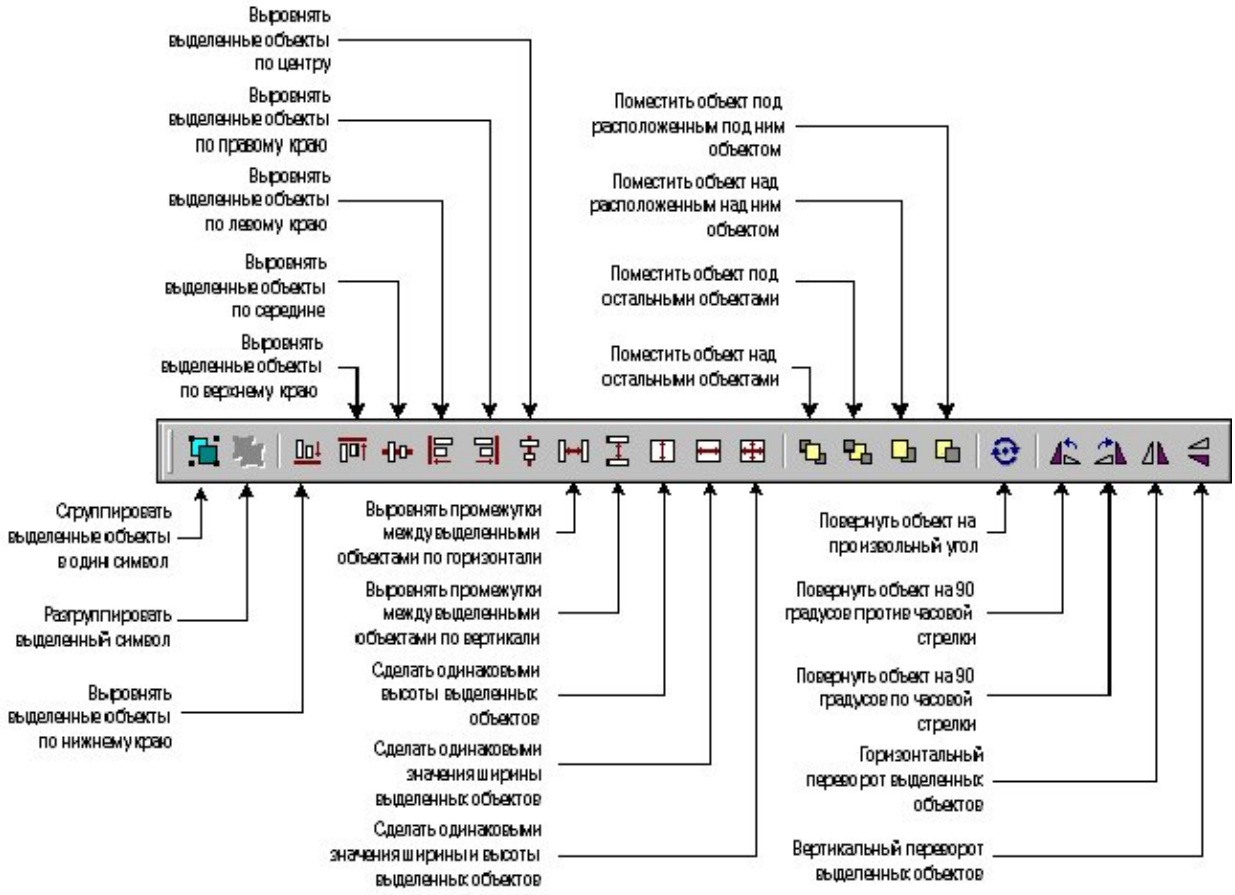


Рис. 1-7. Внешний вид панели инструментов Расположение

1.3.5.6 Панель инструментов *Стиль текста* (**Text Style**)

Внешний вид панели инструментов *Стиль текста* показан на рис. 1-8.



Рис. 1-8. Внешний вид панели инструментов Стиль текста (Text Style)

Данная панель инструментов предназначена для изменения атрибутов отображения текстовых объектов экранных форм GraphWorX32.

1.3.5.7 Панель инструментов *Вид* (**View**)

Внешний вид панели инструментов *Вид* показан на рис. 1-9.
Данная панель инструментов предназначена для изменения масштаба отображения, переключения слоев экранной формы, а также для включения/отключения сетки в области отображения.

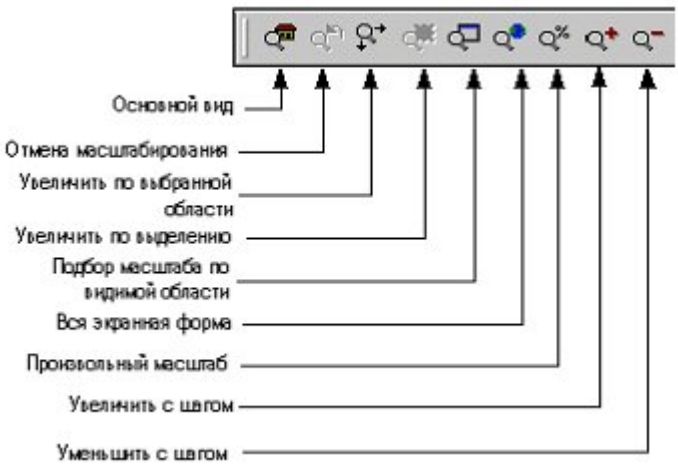


Рис. 1-9. Внешний вид панели инструментов Вид

1.3.5.8 Панель ***Слои***

Панель **Слои** предназначена для добавления, удаления и дублирования слоев. Также можно изменять свойства слоя, устанавливать активный слой, переключать на следующий и предыдущий слой, скрывать слои над и под текущим слоем.



1.3.5 Встроенная панель символов.

Библиотека Символов используется в GraphWorX32 в виде встроенной фиксированной панели, расположенной слева, справа, вверху, внизу, или в виде «плавающего» в экранной форме окна. Функции Библиотеки Символов полностью реализованы во **Встроенной панели символов**. Можно создавать, переименовывать, удалять как файлы категорий(.sdf), так и символы, а также получать доступ к категориям с паролем и менять отображение символов на панели просмотра. Формат файлов категорий (.sdf) полностью совместим с предыдущими версиями **Библиотеки Символов** и обратно.
В GraphWorX32 используются два вида просмотра Библиотеки Символов:

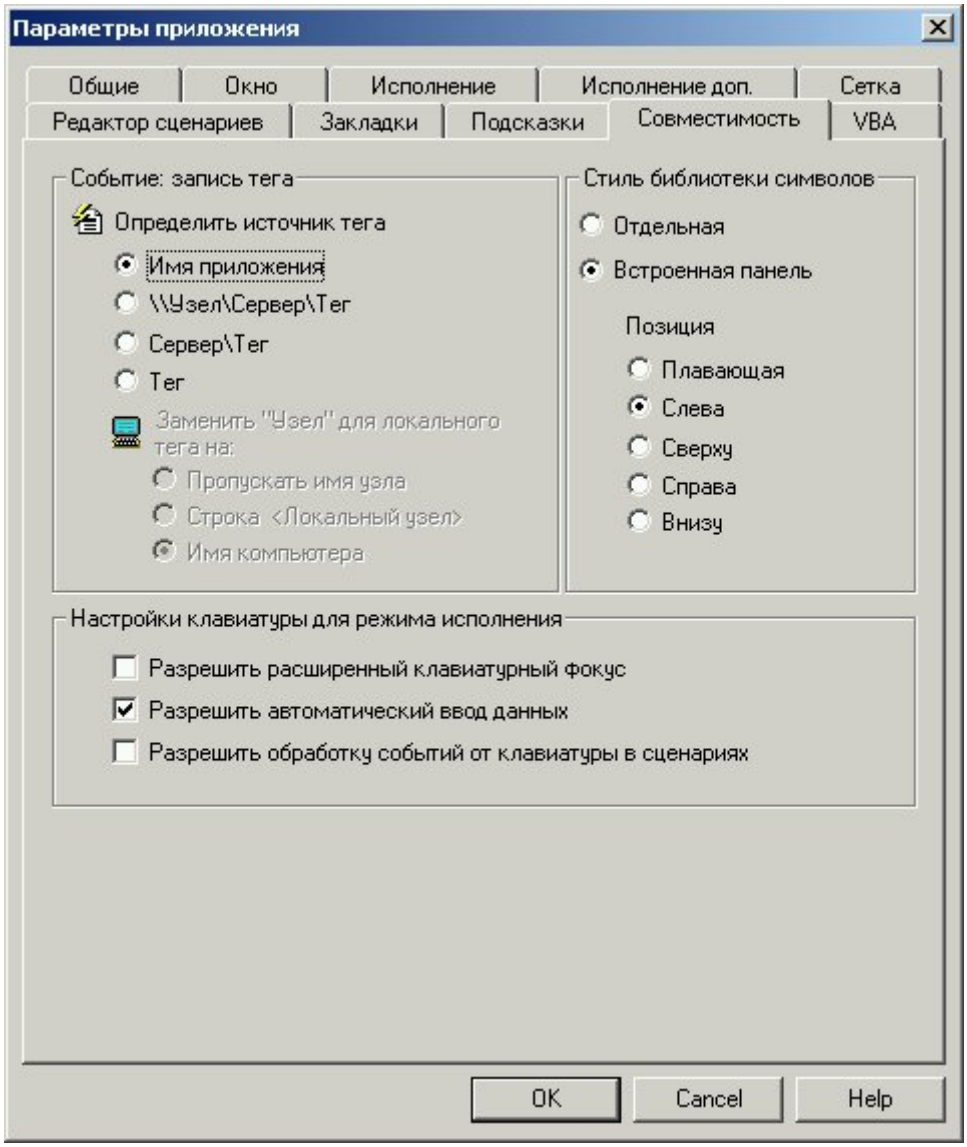
- Стандартная отдельная Библиотека Символов
- Встроенная панель символов

Для переключения между режимами:

1. Выбрать **Параметры приложения** из меню **Формат**.
2. Появляется диалоговое окно **Параметры Приложения**. Выберите закладку **Совместимость**. Под полем

Стиль Библиотеки Символов можно выбрать режим **Отдельная или Встроенная панель символов**.

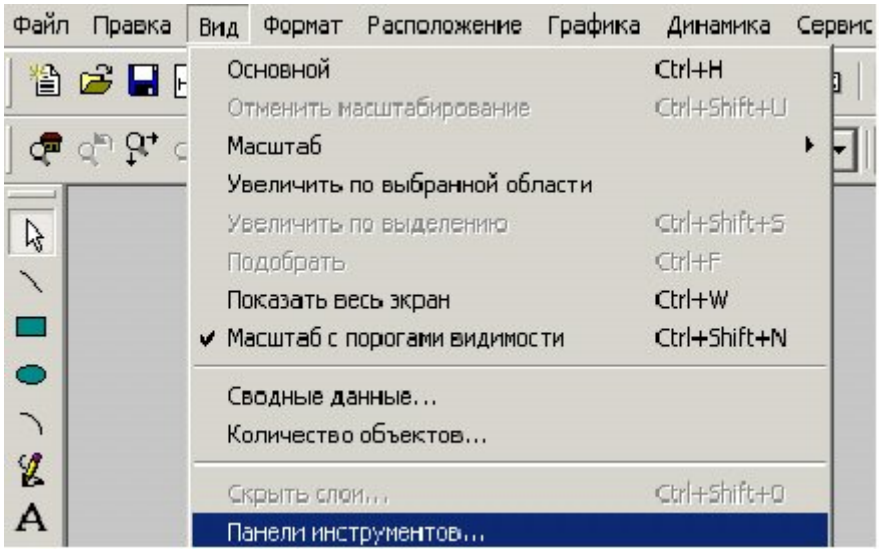
- Если установлен режим **Отдельная**, нажатие на кнопке **Символы** на панели инструментов **Рисование** запускает **SymbolLibrary.exe**. Также можно выбрать Импорт **> Символ** из меню **Рисование**.
- Если установлен режим **Встроенная панель символов**, Библиотека Символов располагается вместе с Панелью Символов в экранной форме. Можно выбрать расположение Панели Символов – справа, слева, вверху, внизу или в плавающем окне. Кнопка **Символы** на панели инструментов **Рисование** управляет видимостью Панели символов (вместо запуска Библиотеки Символов). Команда **Скрыть Панель Символов** из меню **Вид** в этом случае доступна, и диалог **Показать панели инструментов** также содержит флажок **Символы** для управления видимостью **Панели Символов.** Команда **Импорт > Символ** удалена из меню **Графика**.



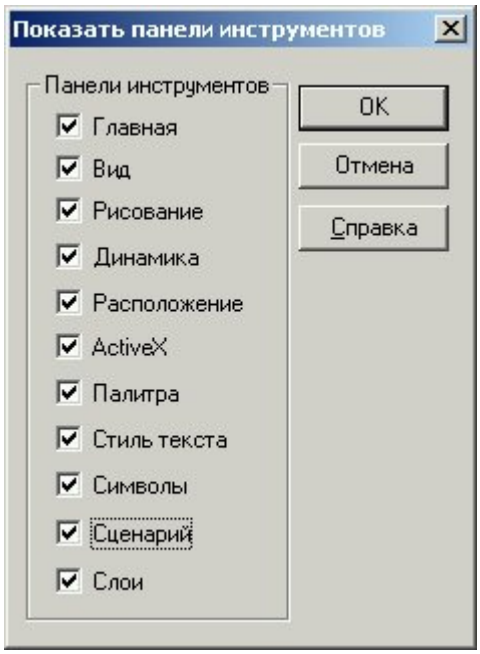
Просмотр Встроенной Панели Символов в GraphWorX32.

Для просмотра Встроенной панели Символов в экране GraphWorX32:

1. Выбрать **Панели инструментов** из меню **Вид**.

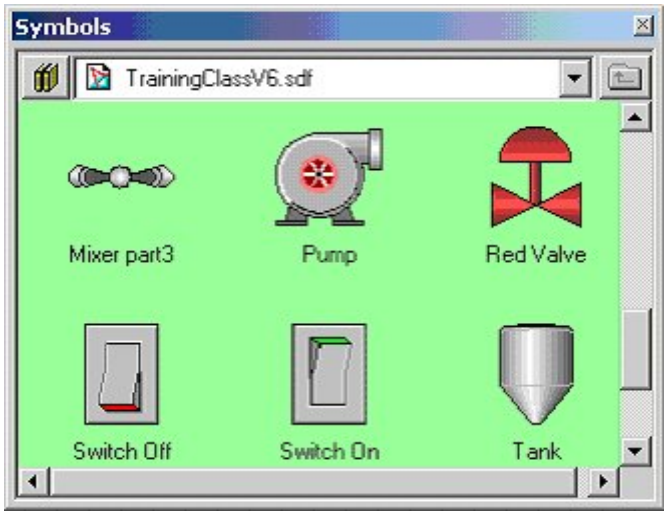


2. В диалоговом окне **Панели инструментов** установить флажок **Символы** и нажать **ОК**.



3. Встроенная Панель Символов появится в экранной форме.

ФУНКЦИИ ВСТРОЕННОЙ ПАНЕЛИ СИМВОЛОВ.

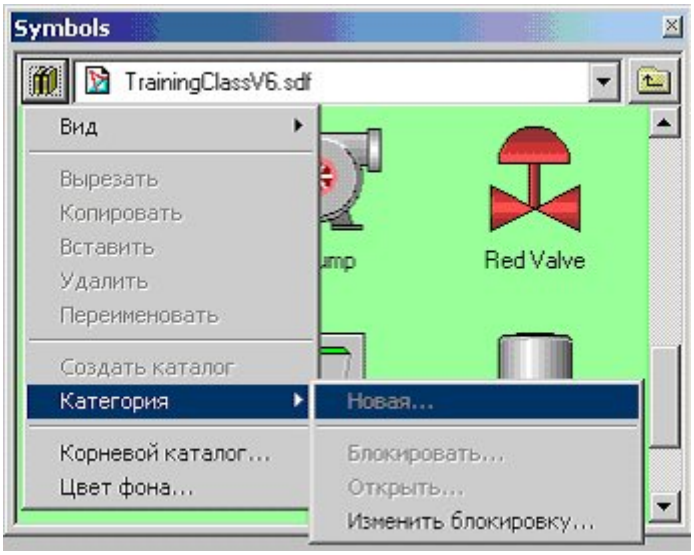


На рисунке показана **Встроенная Панель Символов** в плавающем окне. Раскрывающийся список вверху окна позволяет выбирать символы из всех доступных категорий. Символы, содержащиеся в категории, отображаются на контрольной панели. Операций с символами и категориями выполняются из двух выпадающих меню.

Меню Панели Символов.

Основное меню Встроенной панели Символов отображается при нажатии верхней левой клавиши, как показано на рисунке. Также можно щелкнуть правой клавишей в любом месте окна. С помощью этого меню можно добавлять, переименовывать, удалять, блокировать и разблокировать файлы категорий, изменять вид просмотра символов на панели. Также можно вырезать, копировать, вставлять, удалять и переименовывать символы. Кроме того, из основного меню можно определить корневой каталог для файлов .sdf. Предыдущая версия Библиотеки Символов поддерживала иерархическую структуру файлов категорий; другими словами, файлы категорий могли располагаться в различных подкаталогах. Встроенная Библиотека символов располагает все файлы категорий на верхнем уровне (даже включенные в подкаталоги) и показывает путь к каждой категории, так что файлы категорий не теряются.

Примечание: полная информация о функциях меню содержится в документации Библиотеки Символов.

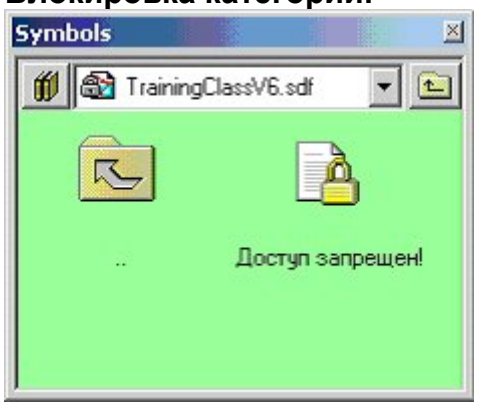


Команды основного меню.

Command	Function
Вид	Определяет размер иконок символов в выбранной категории.
Вырезать	Удаляет выделенный символ из списка просмотра.
Копировать	Копирует выделенный символ.
Вставить	Вставляет символ в список просмотра.
Удалить	Удаляет выделенный символ.
Переименовать	Переименовывает выделенный символ.
Корневой каталог	Определяет каталог для файлов категорий (.sdf).
Создать каталог	Создает папку в корневом каталоге для файлов категорий (.sdf).
Категория	Добавляет, переименовывает, удаляет, блокирует и разблокирует файлы категорий.
Новая	Создает новый файл категории

Блокировать	Устанавливает пароль режима Чтения и Записи в диалоговом окне Установить пароль режима Записи для категории. Пароль должен быть подтвержден. Нельзя устанавливать пароль для режима Только Чтение без установки пароля на Чтение и Запись; это может привести к блокировке и невозможности удаления файла категории.
Открыть	Открывает заблокированную категорию. В заблокированном файле всегда закрыт доступ к режиму Записи. Для открытия файла нужно ввести пароль на Чтение и Запись (или Только Чтение). По введенному паролю распознается соответствующий режим.
Изменить блокировку	Изменяет пароль для файла категории. Необходимо иметь права для открытия категории в режиме Чтение и Запись.
Переименовать	Переименовывает выбранный файл категории.
Удалить	Удаляет выбранный файл категории.
Цвет фона	Изменяет цвет фона панели просмотра.

Блокировка категории.



Есть возможность выполнять блокировку записи и чтения любого файла категории, включенных в выпадающий список. Заблокированная категория показывается в списке с помощью другого значка. Кроме того, если категория заблокирована как от чтения, так и от записи, на Панели Символов появляется сообщение, извещающее пользователя об этом.

Особенности.

Настроечная информация Встроенной Панели Символов (цвет фона, текущая категория, корневой каталог) сохраняется в системном реестре в разделе:

HKCU\Software\ICONICS\SymbolLibrary\Settings.

Остальные данные панели инструментов сохраняются в системном реестре в разделе:

HKCU\Software\ICONICS\Gwx32\.

Важное замечание: из-за сохранения в реестре дополнительной информации Встроенной Панели Символов, GraphWorX32 Version 6.1 не может быть запущен на том компьютере, где до этого

запускался GraphWorX32 Version 7.0, если не удалена ветвь HKCU\Software\ICONICS\Gwx32\General-Summary\Bars. После этого GraphWorX32 Version 6.1 запускается с исходными настройками.

1.3.6 Панель цветовой палитры.

Панель цветовой палитры предназначена для выбора цветов, применяемых к графическим объектам и примитивам GraphWorX32 в процессе создания редактирования экранных форм. Возможно применение цветов из палитры к следующим атрибутам графических объектов и примитивов: *заливка*, *рамка*, *фон* и *текст*.
Палитра цветов может быть закреплена в области инструментальных панелей окна GraphWorX32 либо выведена на экран монитора в виде диалоговой панели.



Рис. 1-10. Внешний вид инструментальной панели Палитра цветов

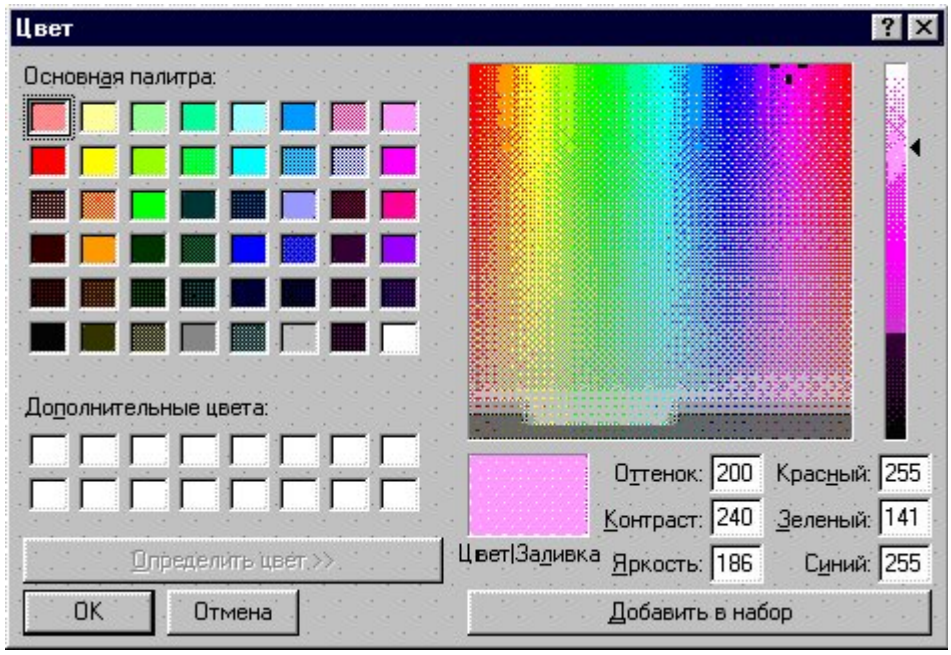


Рис. 1-11. Внешний вид диалоговой панели Палитра цветов

Инструментальная панель палитры цветов состоит из двух или более рядов выбираемых цветов, кнопок прокрутки остальных доступных цветов, а также индикатора текущего выбранного цвета.
Для изменения *цвета заливки* следует выполнить щелчок левой клавишей мыши на одном из прямоугольников палитры, которыми отображаются доступные цвета.
Для изменения *цвета линии* или *цвета текста* следует выполнить щелчок правой клавишей мыши на одном из прямоугольников палитры.
Двойной щелчок левой клавишей мыши на одном из прямоугольников цветовой палитры позволяет выбрать произвольный цвет для данного прямоугольника с помощью системной диалоговой панели Windows **Палитра цветов**.
Если два или более графических объектов сгруппированы в один объект, имеется возможность изменения цветов заливки и рамки для всех объектов.
Инструментальная панель палитры цветов может быть закреплена горизонтально или вертикально в областях панелей инструментов окна GraphWorX32 либо выведена из

указанных областей и размещена во внутренней области окна GraphWorX32 над редактируемыми графическими объектами экранной формы.

1.3.7 Дополнительные возможности инструментальной панели палитры цветов

Щелчок правой клавишей мыши при помещении курсора в область вокруг кнопок прокрутки палитры цветов приводит к появлению контекстного (всплывающего) меню, внешний вид которого показан на рис. 1-12.

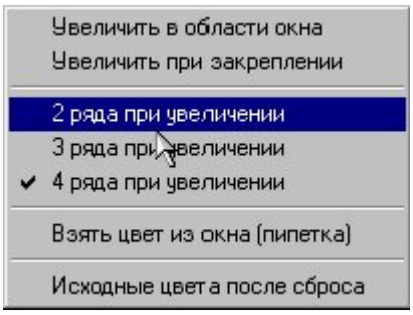


Рис. 1-12. Внешний вид меню палитры цветов

Команда меню **Увеличить в области окна** позволяет увеличить размеры инструментальной панели палитры цветов, когда она расположена во внутренней области окна GraphWorX32.
При отметке строки **Увеличить при закреплении** размеры инструментальной панели увеличиваются, когда она прикреплена к какой-либо из областей панелей инструментов окна GraphWorX32.
Команды **2, 3, 4ряда при увеличении** устанавливают количество видимых рядов цветов в панели палитры.
Команда **Исходные цвета после сброса** предназначена для установки состава палитры цветов по умолчанию.
Команда **Взять цвет из окна (Пипетка)** может быть использована для выбора цвета из текущей открытой экранной формы и сохранения в панели палитры цветов. Для этого следует выбрать указанную команду меню палитры цветов, после чего курсор мыши примет вид пипетки. Поместите курсор на элемент отображения или графический объект, чей цвет должен быть добавлен в палитру цветов, и щелкните левой клавишей мыши. Расположите курсор над одним из прямоугольников палитры и щелкните левой клавишей мыши. Выбранный цвет будет добавлен в палитру цветов GraphWorX32.

1.4 Использование мыши и клавиатуры.

Для работы с GraphWorX32 следует пользоваться клавиатурой и манипулятором типа “мышь” (далее - мышь). Настоящий подраздел содержит описание функций клавиатуры и мыши, используемых в GraphWorX32.

Функции мыши

В Windows мышь используется для перемещения курсора (как правило, в виде стрелки) по экрану, выбора тех или иных объектов, а также для основных действий над объектами (запуск приложений и т.п.).

Функция	Описание
Щелчок левой клавишей мыши	Выбор команд меню или панелей инструментов или выделение объекта, такого как прямоугольник, эллипс и т.п.
Двойной щелчок клавишей мыши	Вызов **Инспектора свойств**, содержащего параметры выбранного объекта и позволяющего изменять значения параметров.

SHIFT+ Щелчок левой клавишей мыши (щелчок левой клавишей мыши при нажатии и удержании клавиши SHIFT на клавиатуре)	Выделение или отмена выделения одного и более графических объектов.
Щелчок правой клавишей мыши	Вызов меню **Формат** по отношению к текущему выделенному объекту. Щелчок правой клавишей мыши во внутренней области экранной формы GraphWorX32, не содержащей ни одного графического объекта приведет к вызову команд меню **Формат**.
Перемещение курсора с удержанием левой клавишей мыши в нажатом состоянии	Перемещение, растяжка и изменение размеров выделенного графического объекта (объектов), а также прикрепление и откреплление панелей инструментов.
Удержание клавиши SHIFT в нажатом состоянии при изменении размеров объекта **Примечание.** Изменение размеров возможно выполнять клавишами "↑", "↓", "→", "←"	Изменение размеров объекта происходит с сохранением пропорций.
Удержание клавиши CTRL в нажатом состоянии при изменении размеров объекта	Изменение размеров происходит относительно центра объекта.
Удержание клавиш SHIFT и CTRL в нажатом состоянии при изменении размеров объекта **Примечание.** Изменение размеров возможно выполнять клавишами "↑", "↓", "→", "←"	Изменение размеров происходит с сохранением пропорций размеров относительно центра объекта.
Удержание клавиши ALT в нажатом состоянии при включенной сетке	Временное отключение привязки объекта к сетке.
Удержание клавиши SHIFT в нажатом состоянии при перемещении объекта	Объект при перемещении будет оставаться в пределах вертикальной и горизонтальной областей, ограниченных воображаемыми линиями, которые проходят горизонтально и вертикально в первоначальном месте нахождения объекта, касаясь наиболее удаленных его точек по горизонтальной и вертикальной осям.
Перенести и отпустить (Drag and Drop)	Объект может быть перенесен в любое окно GraphWorX32, а также в библиотеку символов.
Перенести объект из экранной формы GraphWorX32 и отпустить на Рабочем столе Windows	На *Рабочем столе* Windows создается пиктограмма объекта, позволяющая в любой момент перенести объект обратно в экранную форму.
Перенос объекта с удержанием клавиши CTRL в нажатом состоянии	Создание копии объекта в пределах экранной формы.
Перенос объекта с удержанием клавиш CTRL и SHIFT в нажатом состоянии	Создание копии объекта в пределах экранной формы с возможностью перемещения ее по горизонтали или вертикали.
Удержание клавиши SHIFT в нажатом состоянии во время рисования прямоугольника или эллипса	Позволяет рисовать квадрат или круг.
Удержание клавиши SHIFT в нажатом состоянии во время рисования линии	Сегменты (отрезки) линии могут быть нарисованы под углом, пропорциональным 45°.
SHIFT+ Щелчок правой клавишей мыши	Позволяет перейти в режим редактирования элементов сгруппированного объекта или символа, а также строк текста, дуг и линий.
Удержание клавиши ALT в нажатом состоянии во время выделения нескольких объектов	Щелчки левой клавишей мыши не приводят к выделению отдельных объектов. Разрешено только выделение **группы** объектов.

Операция Перенести и Отпустить

GraphWorX32 поддерживает операцию "Перенести и Отпустить" для собственных (встроенных) объектов, а также для внедренных объектов OLE. Статические и динамические объекты GraphWorX32 могут быть перенесены из одного экземпляра GraphWorX32 в другой. Кроме того, графические объекты GraphWorX32 могут быть перенесены на *Рабочий стол* Windows с возможностью последующего возвращения в экранную форму GraphWorX32. Объекты, не являющиеся собственными объектами GraphWorX32 и отпускаемые после переноса во внутренней области экранной формы или вставляемые из универсального буфера обмена Clipboard, становятся встроенными объектами OLE. В число поддерживаемых форматов при вставке из буфера Clipboard входят: текст, растровые изображения и метафайлы Windows.

Комбинации клавиш

Операции с файлами

CTRL+N	Создать новый файл
CTRL+O	Открыть существующий файл
CTRL+S	Сохранить текущий файл
CTRL+P	Распечатать экранную форму

Операции редактирования

CTRL+Z	Отменить последнюю операцию
CTRL+X	Вырезать выделенный объект и поместить в буфер обмена
CTRL+C	Поместить в буфер обмена копию выделенного объекта
CTRL+V	Вставить объект из буфера обмена
CTRL+D	Дублировать объект

Операции просмотра

CTRL+H	Основной вид
CTRL+SHIFT+U	Отменить масштабирование
CTRL+SHIFT+S	Увеличить по выделению
CTRL+F	Подобрать размер до включения всех объектов в видимую часть окна
CTRL+W	Показать весь экран
CTRL+T	Показать/скрыть панели инструментов
CTRL+0	Произвольный масштаб
CTRL+1	Масштаб 50%
CTRL+2	Масштаб 75%
CTRL+3	Масштаб 100%
CTRL+4	Масштаб 150%
CTRL+5	Масштаб 250%
CTRL+M	Перейти в режим *Разработка*/*Исполнение*

Операции управления расположением

CTRL+G	Сгруппировать в символ
CTRL+U	Разгруппировать символ
CTRL+<Page Down>	Выдвинуть на задний план
CTRL+<Page Up>	Поместить на передний план
<Page Down>	Переместить объект назад
<Page Up>	Переместить объект вперед
Нажатие клавиши TAB после выделения какого-либо объекта	Приводит к снятию выделения с текущего выделенного объекта и выделению следующего по порядку объекту

Операции выравнивания

CTRL+SHIFT+T	Выровнять по верхнему краю
CTRL+SHIFT+B	Выровнять по нижнему краю
CTRL+SHIFT+M	Выровнять по середине

CTRL+SHIFT+L	Выровнять по левому краю
CTRL+SHIFT+R	Выровнять по правому краю
CTRL+SHIFT+C	Выровнять по центру

Операции распределения

CTRL+SHIFT+A	Распределить по горизонтали
CTRL+SHIFT+D	Распределить по вертикали

Операции выравнивания размеров

CTRL+SHIFT+H	Одинаковый размер по высоте
CTRL+SHIFT+W	Одинаковый размер по ширине
CTRL+SHIFT+e	Одинаковый размер по высоте и ширине

Глава 2 Настройка системы.

2.1 Введение
2.2 Параметры командной строки
2.3 Настройка параметров приложения
2.4 Копирование и сброс свойств и параметров
2.5 Поддержка переключения языков.
2.6 Поддержка глобальных псевдонимов.

2.1 Введение

Настоящий раздел содержит описание различных способов настройки параметров системы. Настройка GraphWorX32 может выполняться с использованием:

- Параметров командной строки
- Диалоговой панели **Параметры приложения**
- Диалоговой панели **Свойства экранной формы**

2.2 Параметры командной строки

Имеется возможность настройки режима запуска программы GraphWorX32 в Windows 95/98, Windows NT и Windows 2000 путем использования параметров командной строки. В частности, параметры командной строки позволяют запустить GraphWorX32 с автоматической загрузкой требуемой экранной формы в режиме *Исполнение*. Возможные параметры командной строки приведены в таблице:

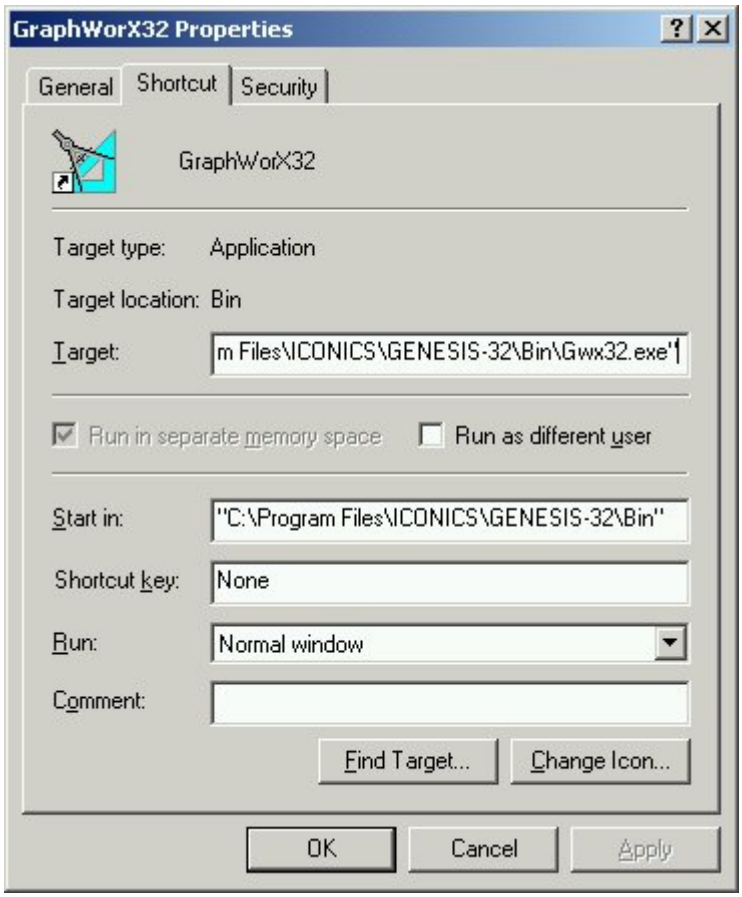
Параметр	Описание
-Runtime	Автоматический запуск GraphWorX32 в режиме *Исполнение*. При использовании данного параметра командной строки заставка приложения перед отображением его главного окна не выводится.
-NoSplash	Запуск GraphWorX32 без вывода на экран заставки приложения
-RegServer	Регистрация приложения в реестре Windows
-UnregServer	Удаление регистрационных данных о приложении из реестра Windows

Установка режима запуска GraphWorX32 в среде Windows

Для установки режима запуска GraphWorX32 в Windows:

1. Выполните щелчок правой клавишей мыши на кнопке **Пуск** (Start) и выберите строку **Открыть** (Open) появившегося контекстного меню. На экран монитора будет выведено окно *Проводника* Windows с выбранной папкой *Главное меню* в древовидном списке.
2. Откройте группу *Программы* (Programs).
3. В группе *Программы* последовательно откройте группу *ICONICS GENESIS-32*, после чего - группу *GraphWorX32*.
4. Выполните щелчок правой клавишей мыши на пиктограмме *GraphWorX32* и в появившемся контекстном меню выберите строку **Свойства** (Properties). На экран монитора будет выведена диалоговая панель, содержащая страницы свойств приложения GraphWorX32.
5. Щелкните на закладке **Ярлык** (Shortcut) диалоговой панели и в поле **Объект** (Target) после названия исполняемого файла *GraphWorX32* (Gwx32.exe) введите путь и название файла экранной формы, которую требуется загрузить автоматически при запуске GraphWorX32, а также

требуемый параметр командной строки. Примеры командных строк приведены в п. 2.2.2.



*Установка режима запуска GraphWorX32 на странице **Ярлык** диалоговой панели **Свойства приложения***

2.2.2 Примеры командных строк

В следующем примере командной строки GraphWorX32 запускается с автоматической загрузкой экранной формы Tanks.gdf:

C:\ICONICS\GENESIS-32\BIN\GWX32.EXE C:\ICONICS\GENESIS-32\PROJECT1\TANKS.GDF

Допускается совместное использование нескольких параметров командной строки, как в приведенном ниже примере:

C:\ICONICS\GENESIS-32\BIN\GWX32.EXE C:\ICONICS\GENESIS-32\PROJECT1\TANKS.GDF –Runtime

2.3 Настройка параметров приложения

Функция настройки параметров приложения позволяет установить параметры среды разработки экранных форм GraphWorX32. Доступ к указанной функции осуществляется в меню **Формат**. Установленные параметры приложения будут применяться ко всем экранным формам, создаваемым в GraphWorX32. Выбор строки **Параметры приложения** меню **Формат** приводит к появлению на экране монитора диалоговой панели **Параметры приложения**, содержащей параметры, применяемые ко всем экранным формам GraphWorX32 по умолчанию. Поскольку параметры приложения после установки их значений применяются только ко вновь создаваемым экранным формам, до выбора строки **Создать** меню **Файл** изменения, внесенные в параметры приложения, не вступят в силу и не изменят внешний вид окна GraphWorX32 и текущей открытой экранной формы.

Общие параметры приложения

Внешний вид страницы свойств **Общие** диалоговой панели **Параметры** приложения показан на рис. 2-2.

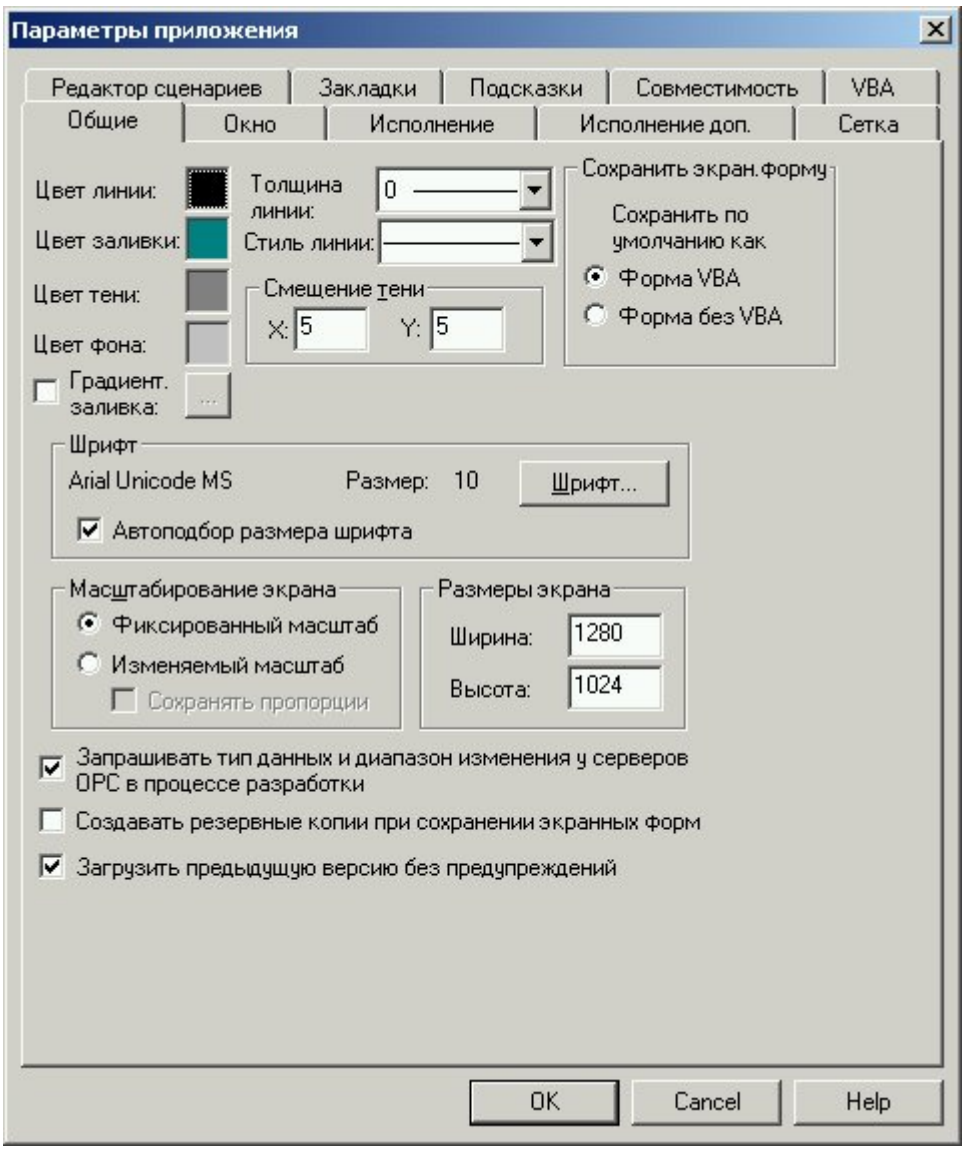


*Рис. 2-2. Страница свойств **Общие** диалоговой панели **Параметры приложения***

Поля **Цвет линии** и **Цвет заливки** позволяют установить цвета, которые применяются по умолчанию к линиям и заливке графических объектов экранных форм.

Поле **Цвет фона** предназначено для выбора цвета фона экранной формы по умолчанию.

Поле **Цвет тени** позволяет выбрать цвет тени по умолчанию для графических объектов, в статических свойствах которых установлен флажок **Цвет тени**.

Поля **Смещение тени -- X:**, **Смещение тени -- Y:** предназначены для установки сдвига тени относительно контура графического объекта в пикселах.

Выпадающие списки **Толщина линии** и **Стиль линии** позволяют выбрать толщину линии в пунктах, а также стиль начертания линии.

Группа параметров **Шрифт** позволяет выбрать тип шрифта и размер шрифта по умолчанию. Если требуется вводить текст символами кириллицы, нажмите кнопку **Шрифт** и в появившейся диалоговой панели **Выбор шрифта** установите *Кириллица* в выпадающем списке **Набор символов**. После сохранения текущей открытой экранной формы и последующего запуска GraphWorX32 выбранный параметр шрифта вступит в силу.

Установка флажка **Автоподбор размера шрифта** приводит к тому, что при изменении размеров текстовых полей экранной формы по умолчанию будет происходить автоматическое пропорциональное изменение размеров шрифта.
Группа параметров **Масштабирование экрана** позволяет выбрать фиксированный или изменяемый масштаб для графических объектов при изменении размеров главного окна приложения. Отметка флажка **Сохранять пропорции** приводит к сохранению пропорций размеров графических объектов экранной формы при изменении размеров окна приложения в случае, если установлен переключатель **Изменяемый масштаб**.
Группа параметров **Размер экрана** позволяет установить высоту и ширину рабочей области экранной формы в пикселах.
Флажок **Запрашивать тип данных и диапазон изменения у серверов OPC в процессе разработки** при установке предписывает приложению запрашивать у серверов OPC, с которыми устанавливаются динамические связи в процессе разработки экранных форм, диапазоны изменения значений и тип отображаемых параметров.
Флажок **Создавать резервные копии при сохранении экранных форм** предназначен для резервного копирования файлов экранных форм в процессе работы с ними. При первом сохранении экранной формы будет создаваться файл, имя которого совпадает с именем сохраняемой экранной формы, а расширение изменено на *BAK*. Обновление резервной копии происходит каждый раз, когда пользователь сохраняет изменения. При этом все изменения, внесенные в экранную форму между предыдущей и текущей операциями сохранения, будут помещены в основной файл экранной формы с расширением *GDF*, а состояние экранной формы на момент предыдущей операции сохранения будет сохранено в файле с расширением *BAK*. Для того чтобы использовать резервную копию экранной формы, следует изменить расширение файла резервной копии с *BAK* на *GDF*, после чего открыть полученный файл командой **Открыть** меню **Файл** GraphWorX32.

Функция **Сохранить в предыдущей версии** позволяет сохранять файлы экранных форм, созданные в более поздней версии GraphWorX32, в варианте для более ранних версий, так что файл экранной формы может быть беспрепятственно загружен в предыдущую версию GraphWorX32. По умолчанию, GraphWorX32 выдает предупреждающие сообщения о совместимости версий. Для того чтобы сообщения не выдавались, следует установить флажок **Загрузить предыдущую версию без предупреждений** (только в диалоге Параметры приложения). Для получения доп. информации см. раздел **Сохранить в предыдущей** версии.
Для публикации экранной формы GraphWorX32 на Web-сервере, файл должен быть сохранен как “Форма без VBA(*.gdf)”, т.к. Мастер Web-публикаций не поддерживает VBA-скрипты. Поэтому в диалоге **Параметры приложения** есть возможность выбрать вариант сохранения экранной формы по умолчанию - как «Форма VBA» или «Форма без VBA». Такая же возможность предусмотрена в диалоге **Свойства экранной формы (глава 3, п.3)**.

Параметры окна

Страница свойств **Окно** диалоговой панели **Параметры приложения** GraphWorX32, внешний вид которой показан на рис. 2-3, позволяет выполнить настройку параметров главного окна GraphWorX32.
Примечание. Ряд параметров, устанавливаемых на странице свойств **Окно**, действуют только в режиме *Исполнение*.

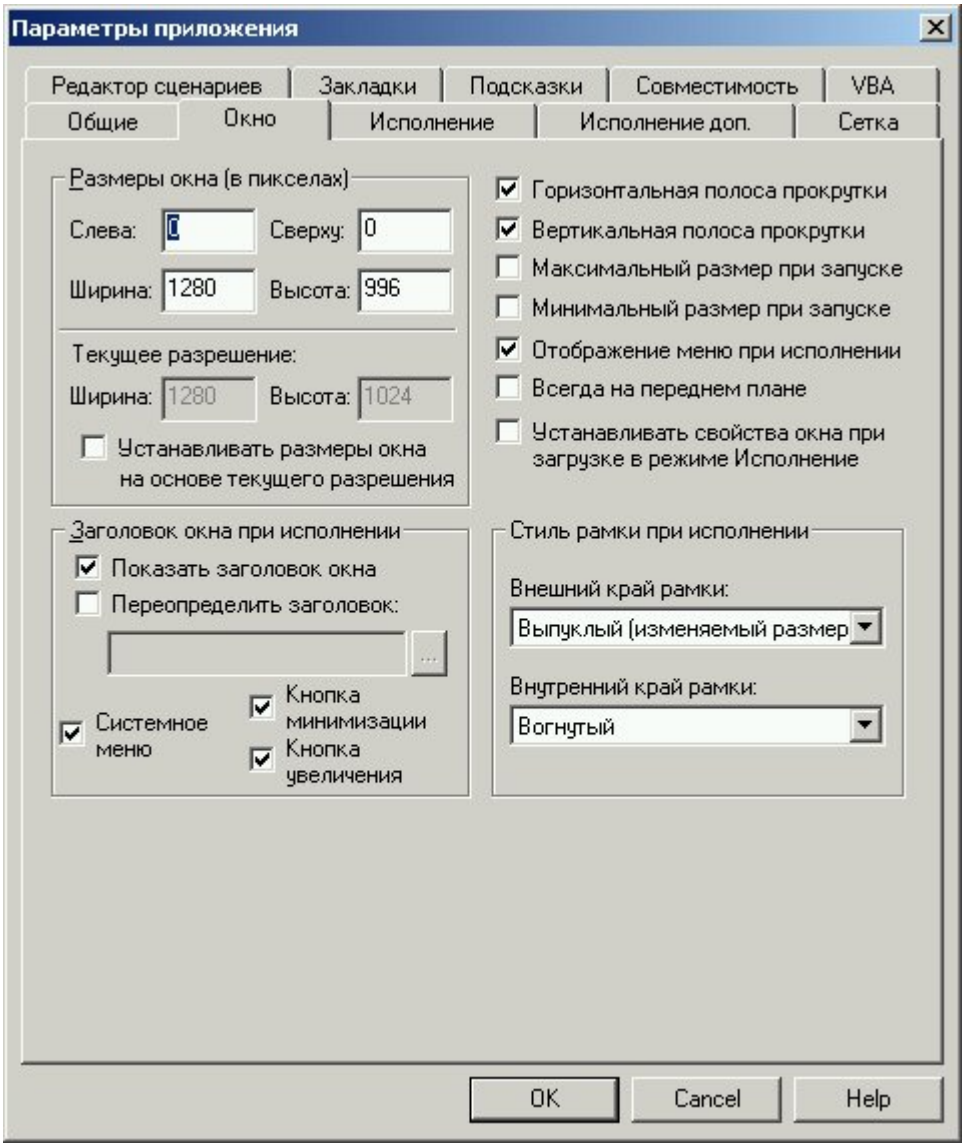
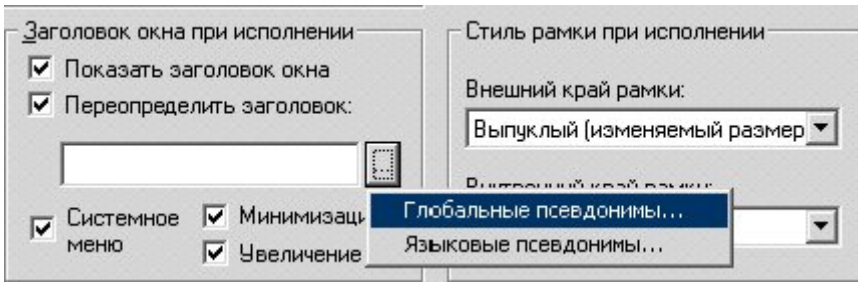


*Рис. 2-3. Внешний вид страницы свойств **Окно** диалоговой панели **Параметры приложения***

Группа параметров **Размеры окна (в пикселах)** позволяет установить требуемый размер и положение относительно левой верхней точки экрана для главного окна приложения. При установке флажка **Устанавливать размеры окна на основе текущего разрешения** размеры главного окна приложения автоматически выбираются на основе текущего разрешения экрана. Например, если текущее разрешение экрана составляет 640x480 и размеры окна GraphWorX32 равны 640x480, то при последующем изменении разрешения экрана до 800x600 происходит автоматическое изменение размеров главного окна GraphWorX32 до 800x600. Если требуется всегда сохранять размеры главного окна GraphWorX32 равными 640x480, следует оставить указанный флажок неотмеченным.
Примечание. После изменения разрешения экрана, главное окно будет иметь размеры, которые были заданы до изменения разрешения.
Группа параметров **Заголовок окна при исполнении** позволяет отключить отображение области заголовка главного окна приложения в режиме *Исполнение*, переопределить заголовок окна, отключить системное меню, вызываемое щелчком левой клавишей мыши на пиктограмме, расположенной в области заголовка приложения слева, а также управлять отображением кнопок увеличения и минимизации главного окна приложения.
Кроме того, имеется возможность показать или скрыть горизонтальную и вертикальную полосы прокрутки окна, заголовки меню в режиме *Исполнение*, а также всегда отображать главное окно GraphWorX32 поверх окон других запущенных приложений Windows в

режиме *Исполнение*. Указанные установки задаются при помощи отметки соответствующих флажков, расположенных на странице свойств справа.
Для переопределения заголовка окна следует установить флажок **Переопределить заголовок** и ввести новый в текстовом окне. Для выбора в качестве заголовка псевдонима нажать кнопку **...** справа от текстового окна и выбрать **Глобальные псевдонимы** или **Языковые псевдонимы** языков из выпадающего меню. Будет открыт соответствующий навигатор псевдонимов, позволяющий выбирать из списка доступных псевдонимов.



При установке флажка **Устанавливать свойства окна при загрузке в режиме Исполнение** динамическая загрузка экранных форм в режиме *Исполнение* будет приводить к изменению текущих параметров главного окна GraphWorX32 на установленные для загружаемой экранной формы. В противном случае, при динамической загрузке экранных форм первоначальные параметры окна, установленные для первой загруженной экранной формы, будут применены ко всем последующим динамически загружаемым экранным формам.
Группа параметров **Стиль рамки при исполнении** позволяет задать внешний вид внутреннего и внешнего краев рамки главного окна приложения.

Исполнение

Параметры, используемые GraphWorX32 в режиме *Исполнение*, устанавливаются на странице свойств **Исполнение**, внешний вид которой показан на рисунке

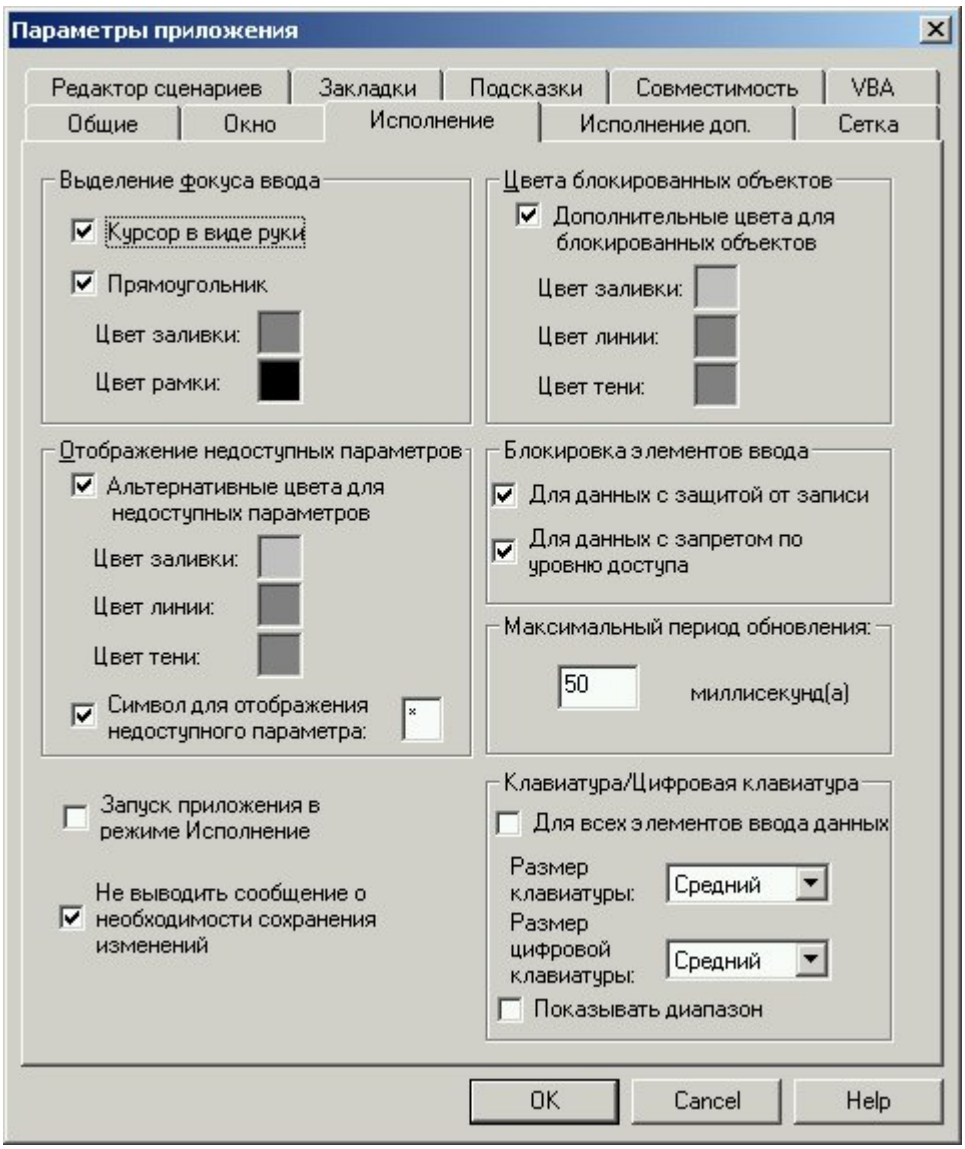


*Рис. 2-4. Внешний вид страницы свойств **Исполнение** диалоговой панели **Параметры приложения***

Группа параметров **Выделение фокуса ввода** предназначена для выбора способа отображения полей экранной формы, в которые предполагается вводить числовые значения, производить операции типа **Указание и щелчок** и т.п., в момент, когда пользователь помещает курсор мыши в область отображения указанного поля. При этом возможно отображать курсор в виде руки, а также заключать объект в прямоугольную рамку. Поле **Цвет заливки** служит для выбора цвета, который будет применен к внутренней области рамки, выделяющей фокус ввода. Поле **Цвет рамки** определяет цвет линий, ограничивающих рамку фокуса ввода внутри и снаружи.

Группа **Отображение недоступных параметров** позволяет выбрать режим отображения для параметров, чьи значения по какой-либо причине стали недоступными приложению во время исполнения. По умолчанию недоступные параметры отображаются символами “*”, как показано на рис. 2-4.

Группа **Цвета блокированных объектов** позволяет выбрать дополнительные цвета, которые будут применены к объектам, находящимся в блокированном состоянии. Состояние блокировки графического объекта экранной формы GraphWorX32, для которого установлено действие **Указание и щелчок**, характеризуется тем, что данное действие не может быть произведено, что весьма похоже на поведение ряда органов управления диалоговых панелей и команд меню Windows, когда связанные с ними

операции не могут быть выполнены при тех или иных условиях. Поля отображения параметров, которые могут быть введены пользователям и переданы серверам OPC (поля ввода), соединенные с тегами OPC, доступными только для чтения, автоматически блокируются.
Поле **Период обновления** содержит значение интервала времени, в течение которого происходит обновление графических объектов и элементов отображения экранной формы GraphWorX32 (смена видеокадра). Минимальный интервал обновления составляет 50 мс.
Группа параметров **Клавиатура/Цифровая клавиатура** позволяет установить размеры отображения цифровой и алфавитно-цифровой клавишных панелей, используемых для ввода числовых и строковых значений в полях ввода в режиме *Исполнение*. Кроме того алфавитно-цифровая клавиатура может использоваться для ввода имени пользователя и пароля при регистрации на **Сервере администрирования**. Если установлен флажок **Для всех элементов ввода данных**, то ввод числовых и строковых значений для всех элементов ввода данных в режиме *Исполнение* будет выполняться только из цифровых и алфавитно-цифровых клавишных панелей, показанных на рис. 2-5. Применение клавишных панелей в GraphWorX32 весьма удобно при использовании компьютеров без устройств ввода, но оснащенных сенсорными экранами.

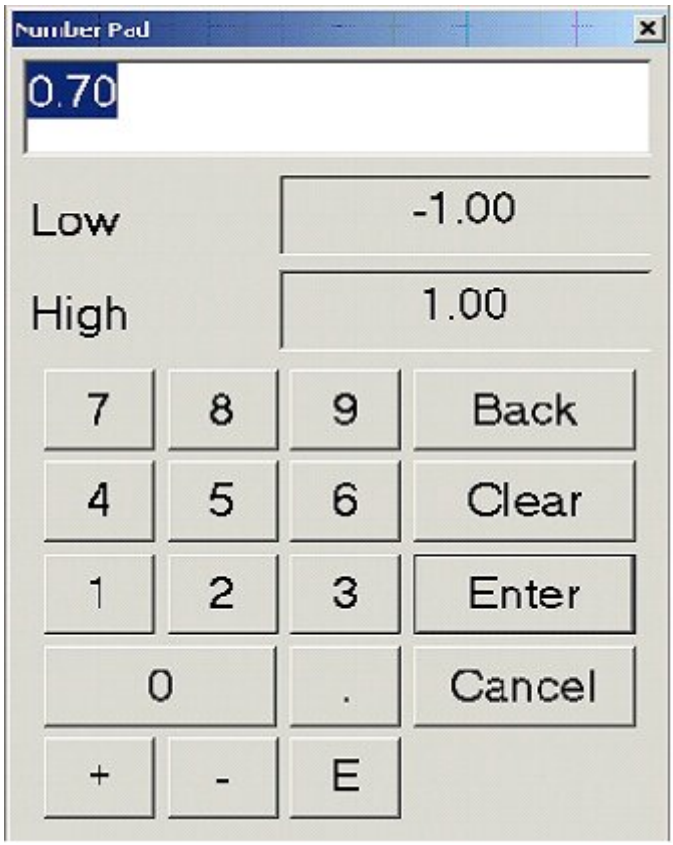




Рис. 2-5 Клавишные панели режима Исполнение GraphWorX32. Алфавитно-цифровая панель может использоваться только для ввода строк символами латинского алфавита

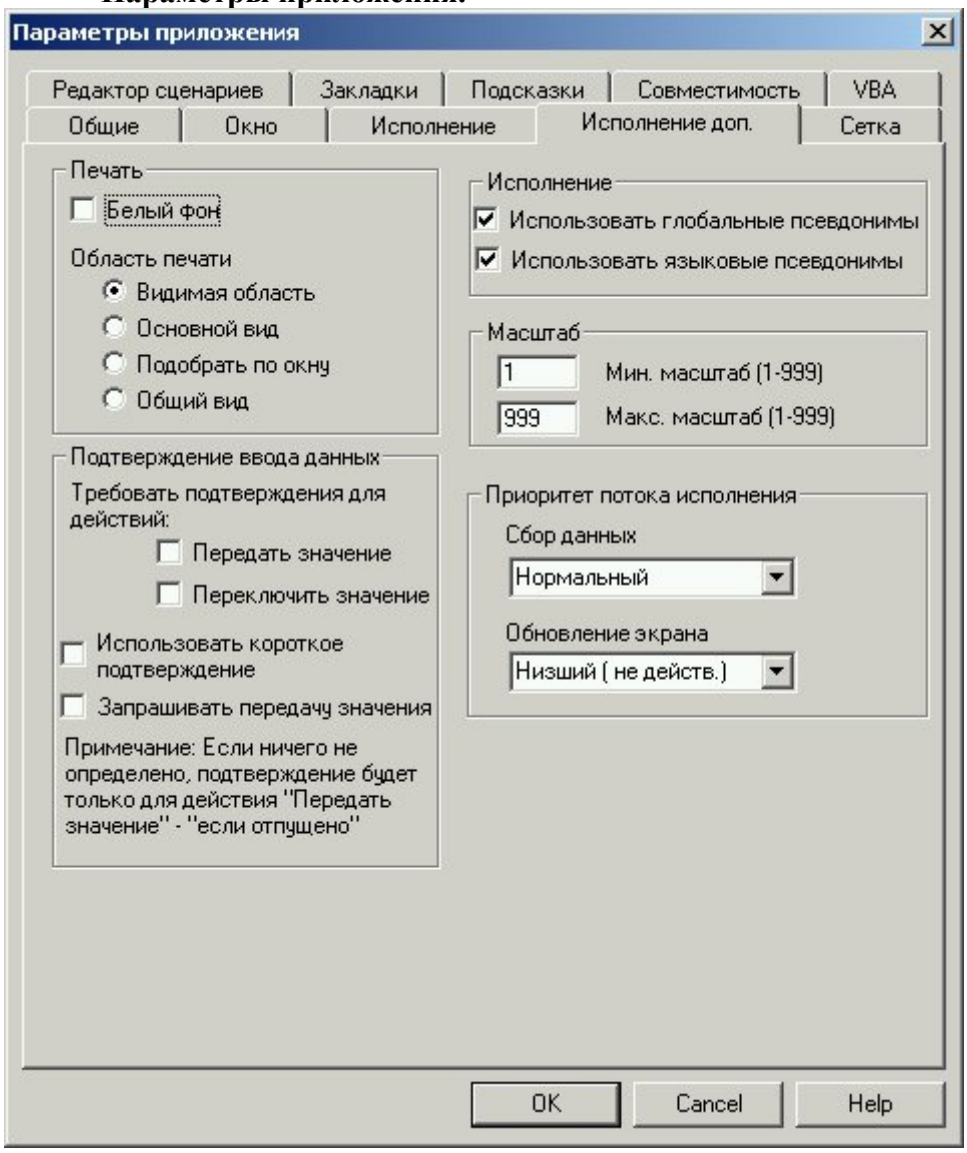
Установка или снятие флажка **Показывать диапазон** показывает или скрывает отображение пределов значения на цифровой клавиатуре.
Цифровая клавиатура также может быть показана в функции **Ввод данных** в диалоговом окне
Инспектор свойств /Параметр.
Переключение CapsLock изменяет символы клавиатуры на верхний регистр. Клавиша **Международная** переключает отображение дополнительных символов. Если она не нажата (исходное состояние), на клавиатуре не показываются символы, которые не содержатся в исходной Английской раскладке клавиатуры. Если клавиша Международная нажата, отображается полная раскладка клавиатуры со специальными символами для соответствующего языка.
Разделители десятичной части и разрядов на цифровой клавиатуре изменяются в соответствии с переключением языков.
Можно также настраивать клавиатуру с помощью редактора ресурсов Microsoft Visual Studio. Нажатие CapsLock будет переключать клавишу между этими двумя символами (они могут и не соответствовать, как, например, 2 и @). Если для клавиши не определено второго символа, будет использоваться соответствующий символ нижнего регистра.

При установке флажка **Не выводить сообщение о необходимости сохранения изменений** системный запрос пользователю на сохранение выполненных изменений не выводится на экран монитора в режиме *Исполнение*. Установка флажка **Запуск приложения в режиме Исполнение** позволяет запускать GraphWorX32 с автоматическим переходом в режим *Исполнение*.

Свойства окна в режиме Исполнение.
Выбор пункта **Отобразить как в режиме Исполнение** из меню **Вид** дает возможность просмотреть, как будет выглядеть экранная форма в режиме исполнения. Это позволяет правильно подобрать размеры экрана и масштаб. Если скрыты панели инструментов, доступ к функциям конфигурации можно получить из основного меню или из меню, открывающегося при щелчке правой кнопки мыши.
Примечание: Нажатие **CTRL+M** переключает экран между режимами разработки и исполнения.

Дополнительные настройки исполнения.

Параметры приложения.



Закладка **Исполнение доп**. в диалоговом окне **Параметры приложения** определяет следующие параметры для режима исполнения.

Настройки печати и предварительного просмотра. Установка флажка **Белый фон** заменяет исходный цвет фона при печати на белый. **Область печати** определяет часть экрана, которая будет напечатана.

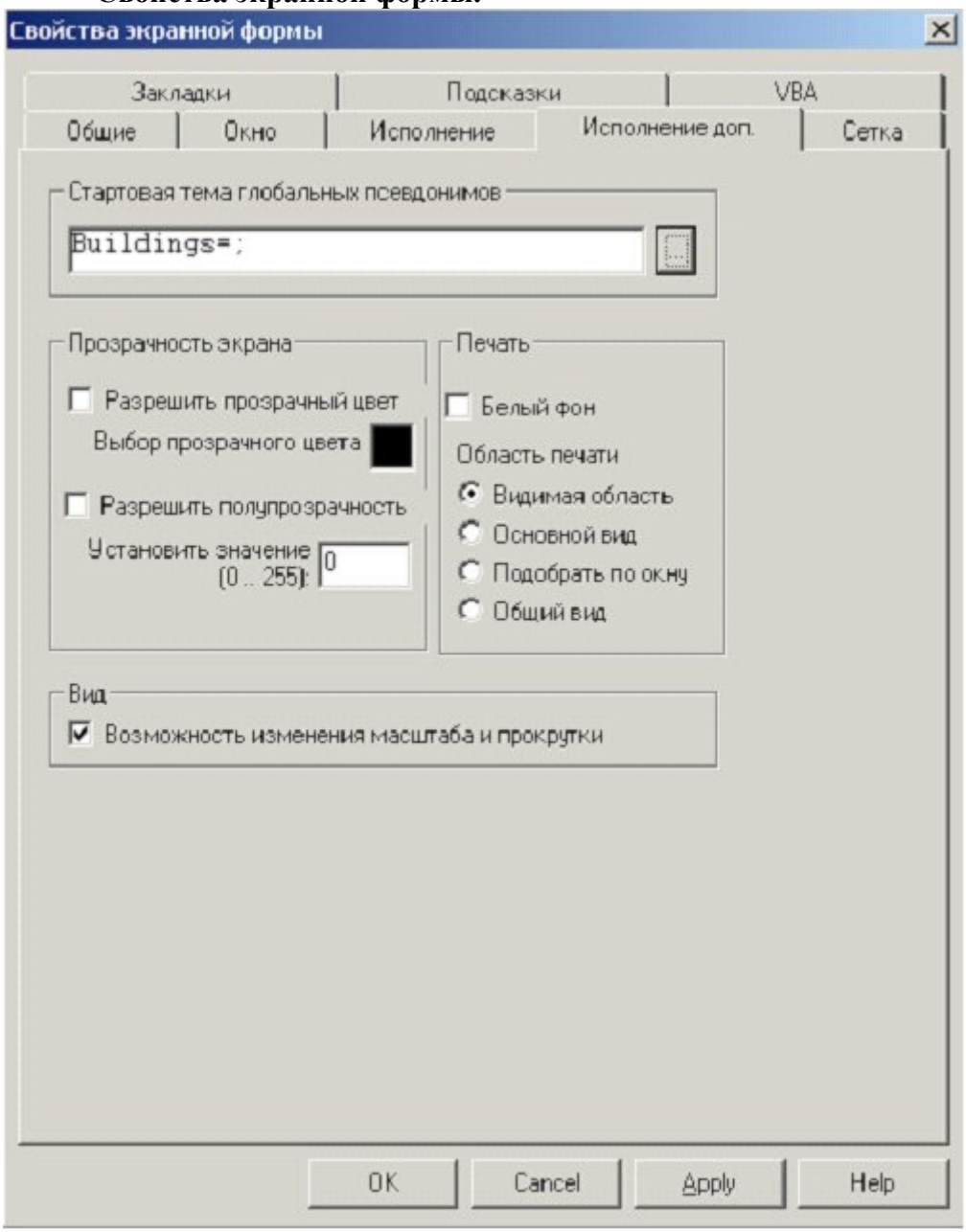
Подтверждение ввода данных позволяет настроить требование подтверждения для функций **Передать значение** и **Переключить значение**. Можно минимизировать длину сообщения, установив флажок **Использовать короткое подтверждение**.
Примечание: Если для события **Передать значение – если отпущено** не определено никакого значения, при действии **Передать значение** в режиме исполнения будет отображаться **Клавиатура/Цифровая клавиатура** для ввода значения. Эта функция включается установкой флажка **Запрашивать передачу значения**.

Исполнение. Флажки **Использовать глобальные псевдонимы** и **Использовать языковые псевдонимы** позволяют включить или отключить поддержку соответствующих псевдонимов.

Панель **Масштаб** позволяет определить Минимальный и Максимальный уровни масштабирования. Если ввести значение минимума больше, чем максимума, GraphWorX32 автоматически поменяет эти два значения, т.к. максимум всегда должен быть больше минимума.

Приоритет потока исполнения позволяет настроить уровни приоритета для сбора данных и обновления экрана во время режима исполнения.

Свойства экранной формы.



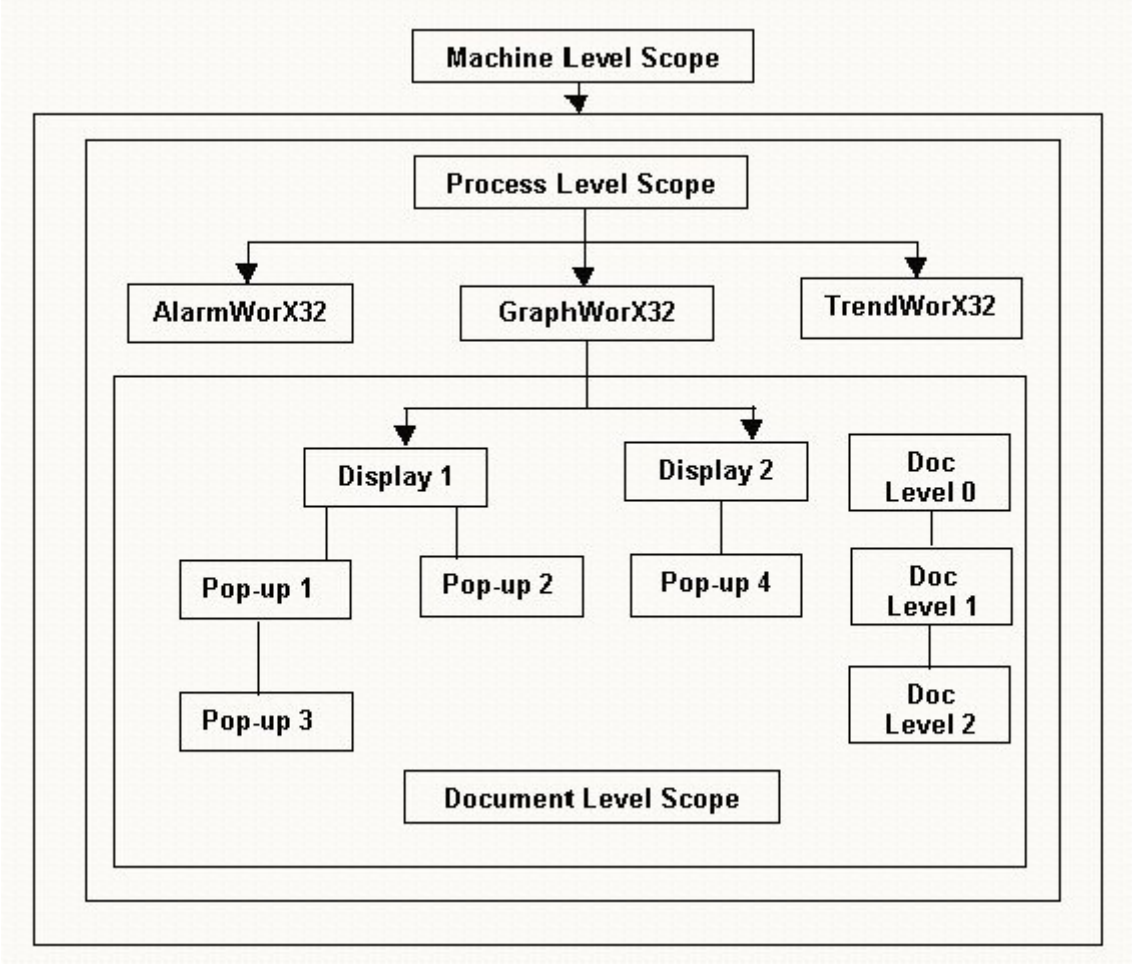
Закладка **Исполнение доп**. в диалоговом окне **Свойства экранной формы** определяет следующие параметры для режима исполнения.

Область тем глобальных псевдонимов.
Привязка псевдонимов контролируется с помощью **Области псевдонимов** (т.е. определяется, какой уровень тем применен к Вашему приложению). Система Глобальных псевдонимов поддерживает три различных уровня псевдонимов:
Компьютер – применяется ко всему компьютеру.

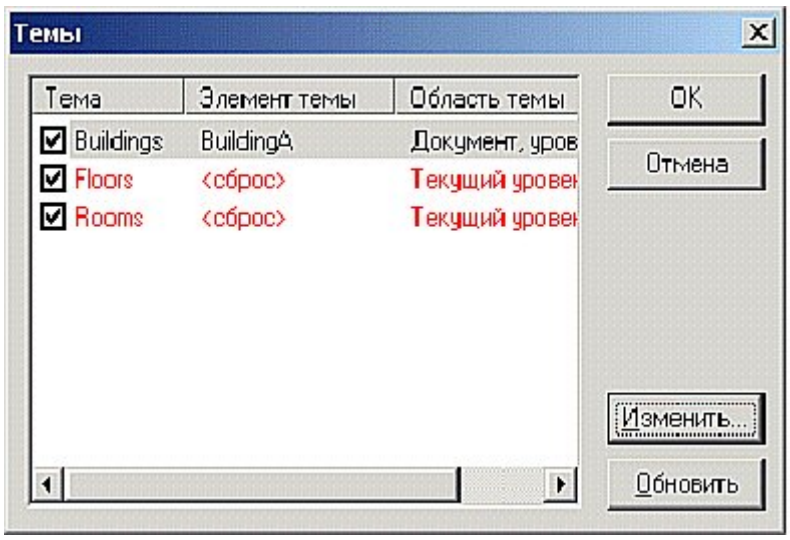
Процесс – применяется к текущему процессу (напр., GraphWorX32 с рабочими окнами).
Документ – применяется только к текущему документу/экранной форме и встроенным в него элементам управления (напр., окно GraphWorX32).
Псевдонимы, принадлежащие к определенной группе, выделяются следующими разделителями:

- **<!MACHINE!>**
- **<@PROCESS@>**
- **<#DOCUMENT#>**

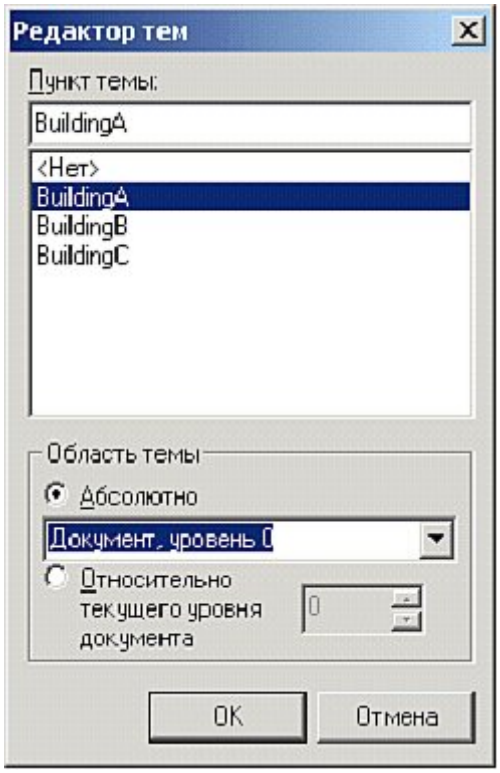
В GraphWorX32 может быть несколько уровней документов или рабочих окон:



Уровень документа для Области темы задается в **Свойствах экранной формы** GraphWorX32. Для определения исходного области тем:
1. Введите имя темы в поле **Стартовая тема глобальных псевдонимов** или нажмите кнопку **...** и выберите имя темы в диалоговом окне **Темы**:

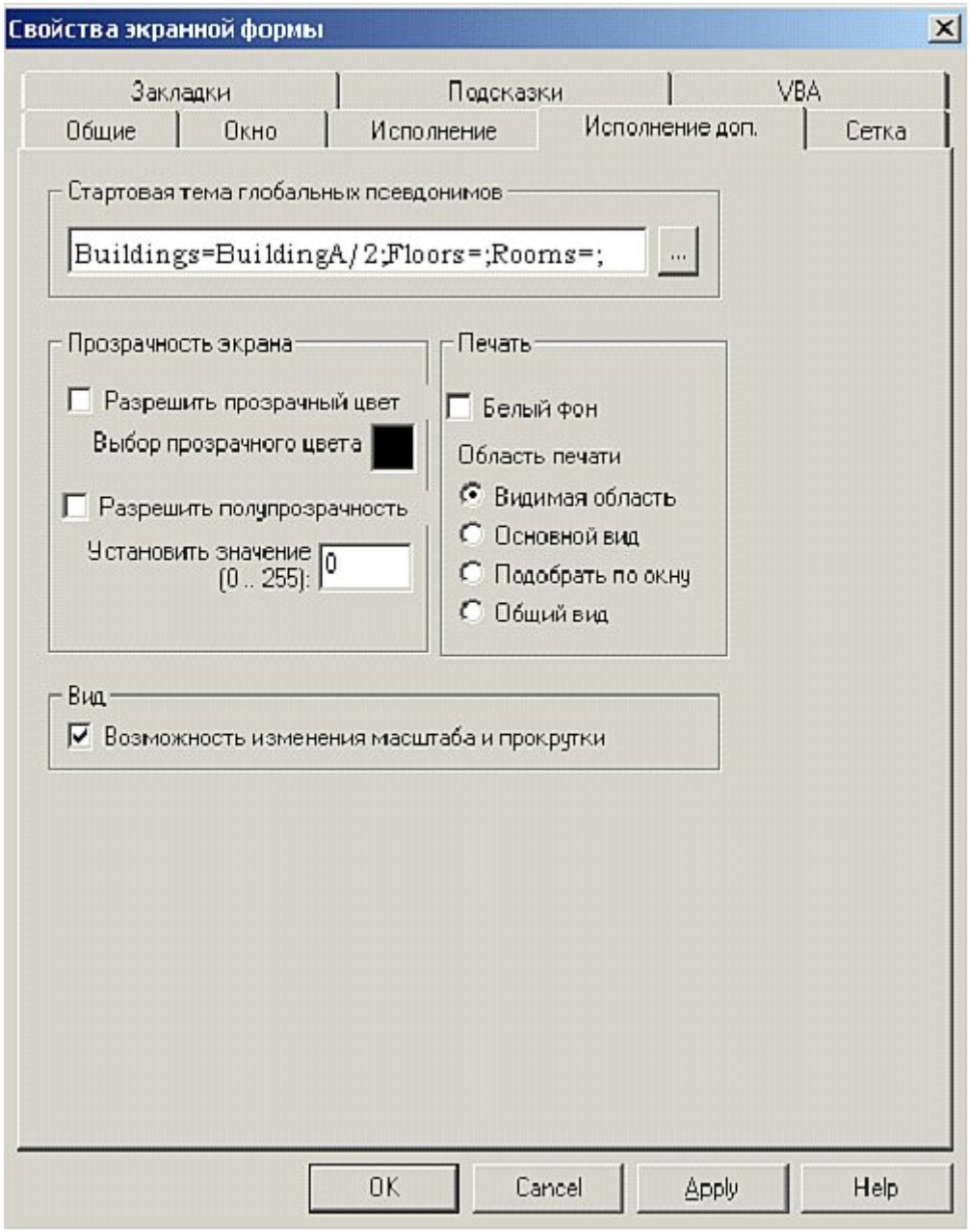


2. Для того чтобы определить область темы, отметить тему и нажать кнопку **Изменить**. Будет открыт **Редактор тем**. Выберите пункт темы(напр., «BuildingA») для связи со стартовой темой. В области **Область темы** можно выбрать **Абсолютно** (напр., уровень компьютера, уровень процесса или уровень документа). Можно выбрать область **Относительно текущего уровня документа** (напр., основной экран, окно 1, окно 2 ит.д.). Нажмите **ОК**.



3. Выбранный пункт темы и область темы отобразятся в диалоге **Темы**. Нажмите **ОК**.

Выбранные темы появятся в диалоге **Старт. тема Глоб. псевдонимов** в **Свойствах экранной формы.**



Прозрачность и полупрозрачность.
Функция GraphWorX32 (поддерживается в Windows 2000 и более поздних) позволяет задавать для экрана **Прозрачность и полупрозрачность.**
Для включения прозрачности установите флажок **Разрешить прозрачный цвет** и выберите цвет, который во время исполнения будет прозрачным.
Для включения полупрозрачности (называемой также альфа-сопряжением) установите флажок **Разрешить полупрозрачность**. В окне **Установить значение** определяется значение полупрозрачности, которое может быть задано в диапазоне 0-255. Если значение равно 0, окно полностью прозрачно. Если значение равно 255 – полностью непрозрачно.
Кроме настроек в **Свойствах экранной формы** существуют следующие методы в объекте **Gwx32.GwxDisplay,** с помощью которых можно устанавливать полупрозрачность в режиме исполнения:
Функция **SetTransparency**(EnableTransparency As Boolean, transparentColor As OLE_COLOR, EnableAlpha As Boolean, AlphaBlending As Integer) As Boolean

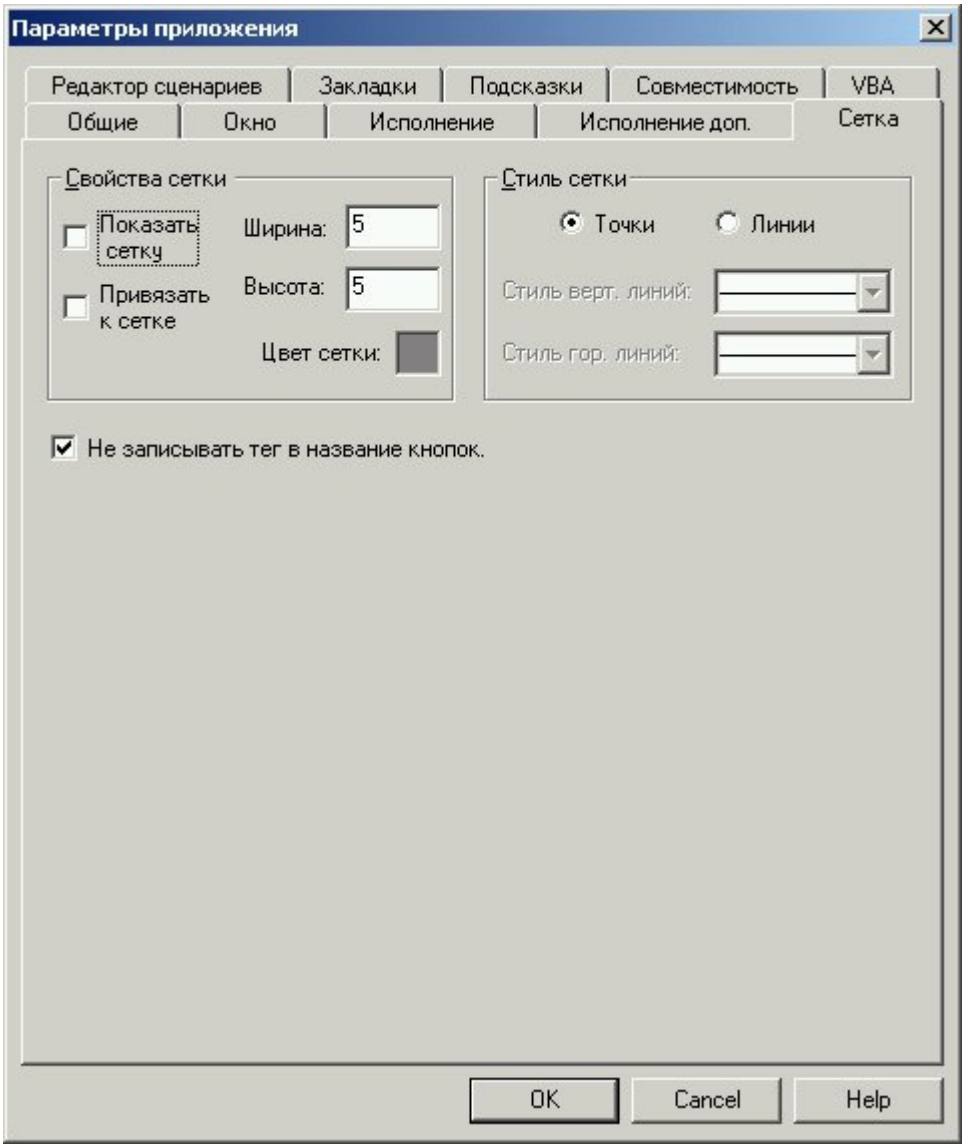
Функция **GetTransparency**(TransparencyEnabled As Boolean, transparentColor As <Unsupported variant type>, AlphaEnabled As Boolean, AlphaBlending As Integer) As Boolean

Печать и область печати
Установка флажка **Белый фон** заменяет исходный цвет фона при печати на белый. **Область печати** определяет часть экрана, которая будет напечатана.

Вид
Можно использовать возможность изменения масштаба экранной формы и прокрутки.

Сетка

GraphWorX32 для удобства рисования графических объектов позволяет использовать сетку, настройка параметров которой осуществляется на странице свойств **Сетка** диалоговой панели **Параметры приложения и Свойства экранной формы**. Внешний вид страницы свойств показан на рисунке.



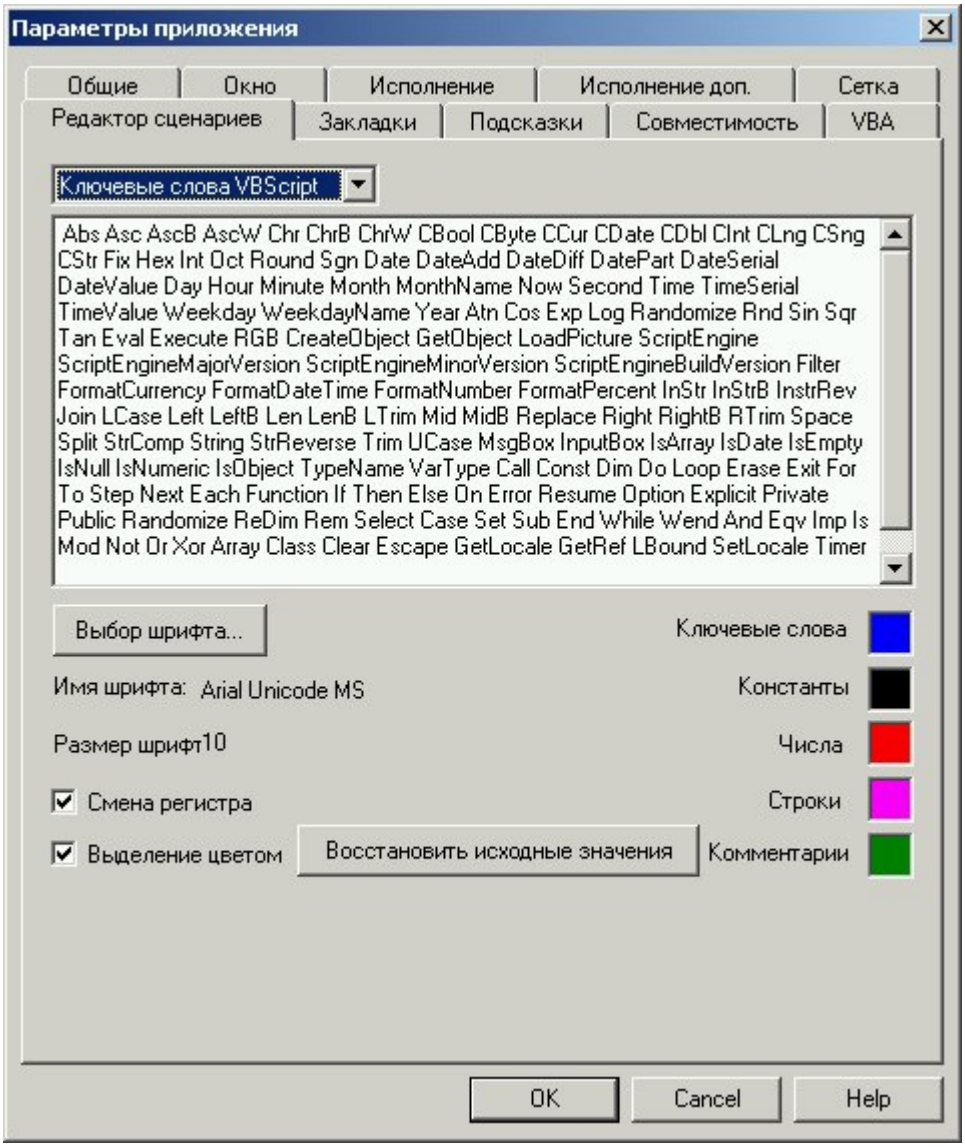
*Внешний вид страницы свойств **Сетка** диалоговой панели **Свойства экранной формы***

Группа **Размеры сетки** позволяет выбрать горизонтальный и вертикальный шаг координатной сетки в базовых единицах (в зависимости от текущего режима масштабирования). Когда выбран масштаб 100%, размер единицы шага сетки равен одному пикселу.

Поле **Цвет сетки** предназначено для выбора цвета, который будет применен к линиям сетки.
Группа параметров **Стиль сетки** позволяет установить внешний вид сетки: в виде точек или линий, расположенных в видимой области экранной формы.
Для привязки создаваемых графических объектов к сетке следует установить флажок **Привязать к сетке**.
Флажок **Показать сетку** позволяет включить или отключить отображение линий сетки.

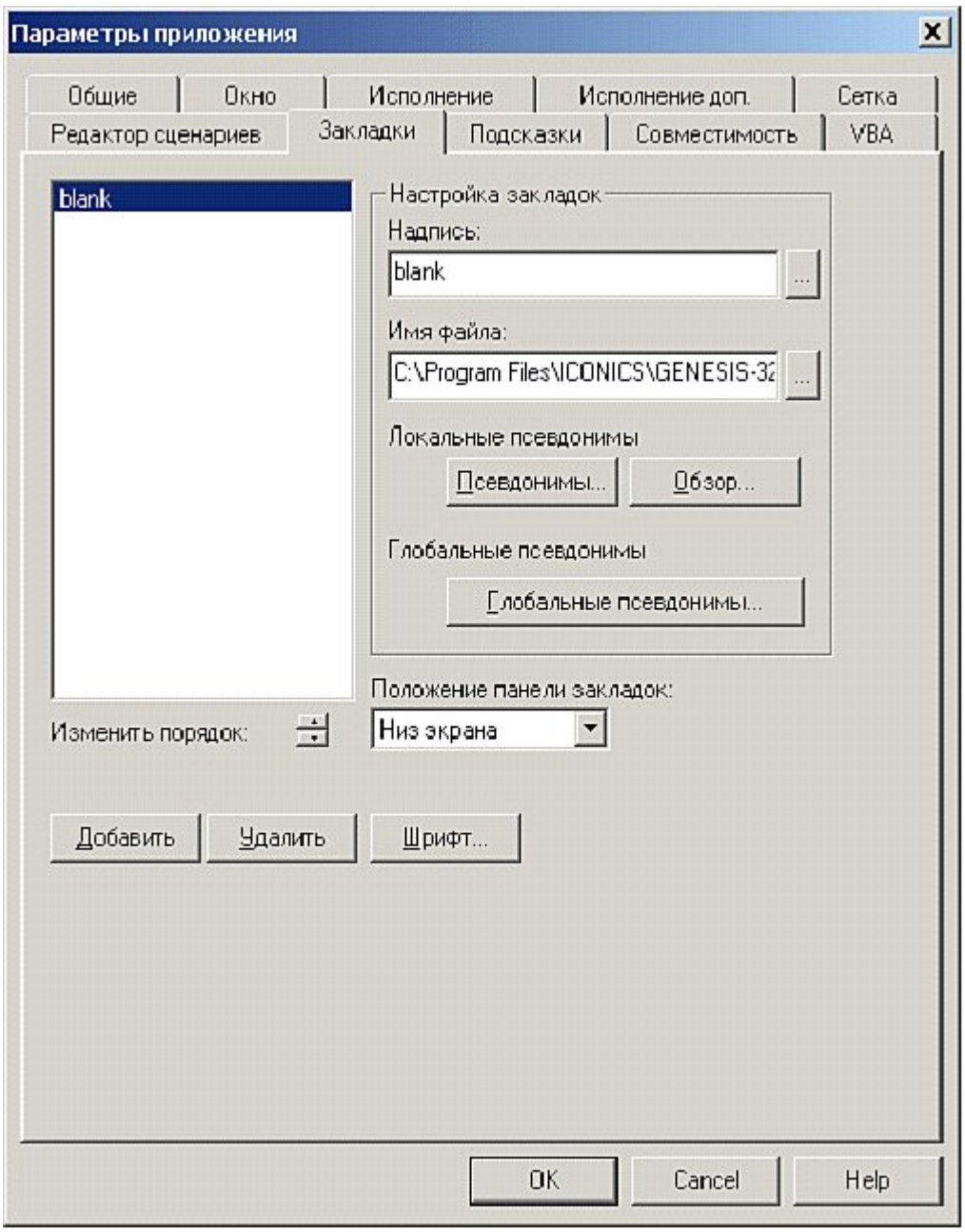
Редактор сценариев.

Закладка **Редактор сценариев** диалога **Параметры приложения** позволяет выполнять пользовательские настройки для редактора сценариев.



Закладки

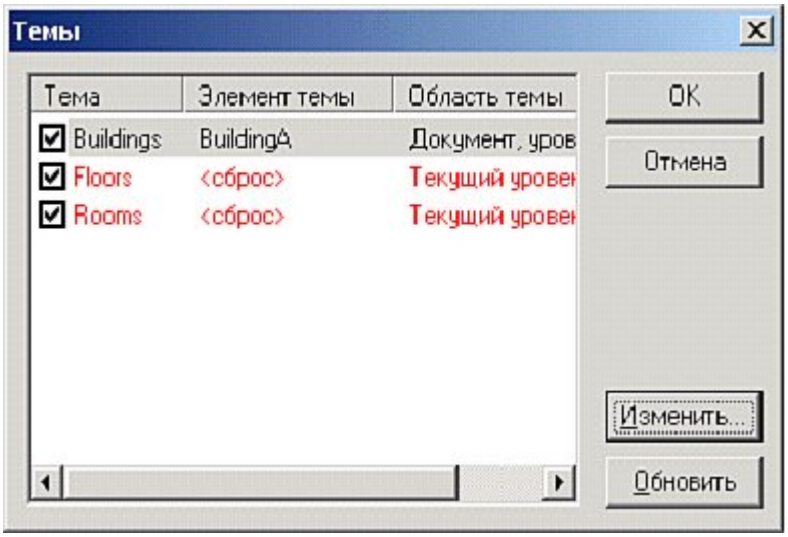
Страница свойств **Закладки** предназначена для построения ряда закладок, которые располагаются в верхней или нижней части главного окна GraphWorX32 и позволяют выполнять загрузку связанных с ними файлов экранных форм. Внешний вид закладок показан на рисунке. Закладки загрузки экранных форм обеспечивают простой доступ к множеству экранных форм в режиме *Исполнение*.



*Внешний вид страницы свойств **Закладки** диалоговой панели **Параметры приложения***

Для создания закладки следует нажать кнопку **Добавить** страницы свойств **Закладки**. На экран монитора будет выведена диалоговая панель **Связать файл с новой закладкой**, в которой выбирается имя файла экранной формы, который будет загружаться после щелчка левой клавишей мыши на данной закладке. Надпись на закладке вводится в поле **Надпись**. Для выбора шрифта, которым будет отображаться надпись, следует нажать кнопку **Шрифт.** Для удаления закладки нажмите кнопку **Удалить**. Имя файла и путь к файлу, связанному с закладкой, выбранной в списке закладок, который расположен в левой части страницы свойств, отображается в поле **Имя файла** страницы свойств. Для выбора другого файла, связанного с существующей закладкой, выполните двойной щелчок левой клавишей мыши на названии закладки в списке.
Местоположение закладок в окне устанавливается в выпадающем списке **Положение панели закладок**.
Для изменения положения закладки в списке закладок, а значит и на панели закладок, следует выбрать имя закладки и нажать кнопки прокрутки **Изменить порядок**.

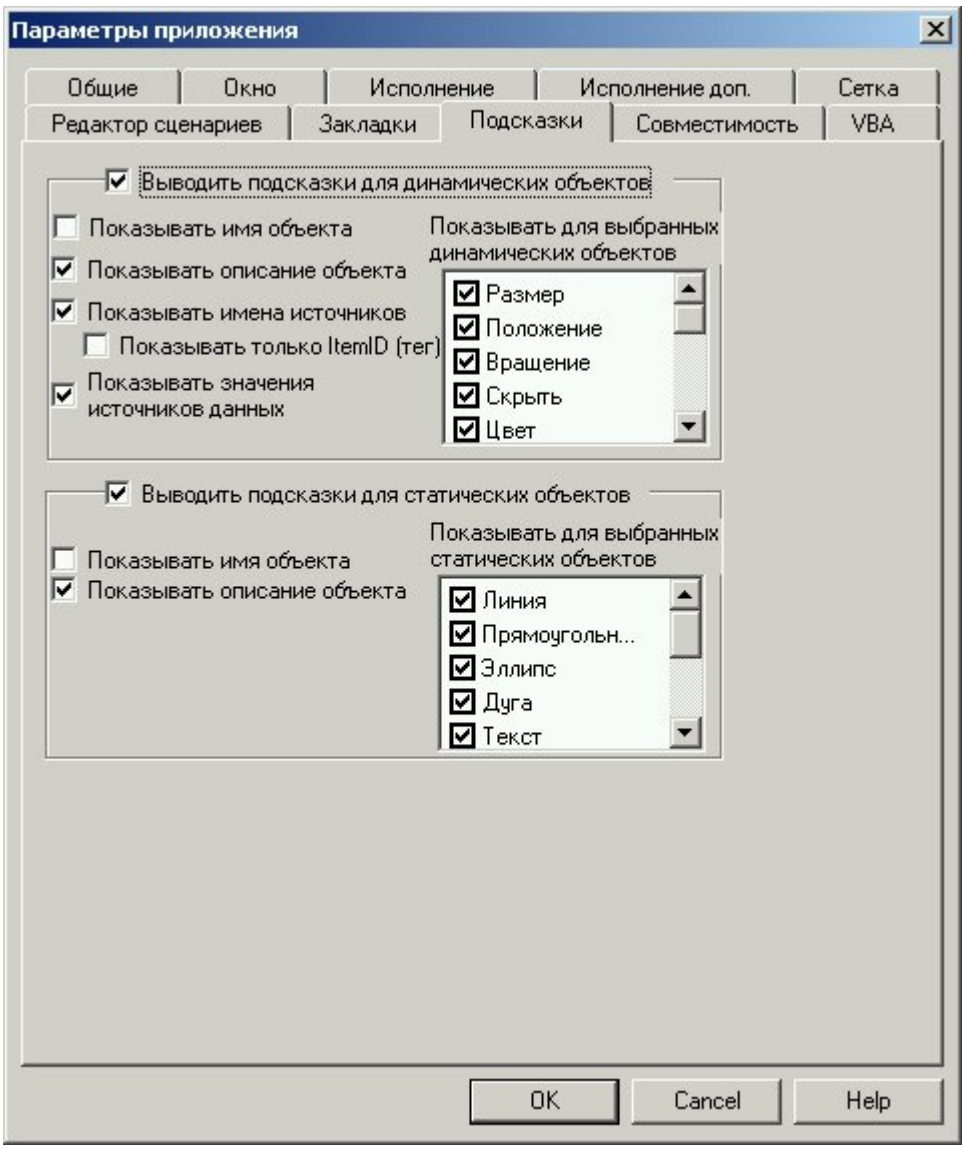
Здесь же можно определить локальные и глобальные псевдонимы. Локальные псевдонимы: нажать кнопку **Псевдонимы**. Откроется диалоговое окно **Настройка конфигурации для псевдонимов**, где определяются параметры псевдонимов.
Глобальные псевдонимы: нажать кнопку **Глобальные псевдонимы** для выбора темы в диалоговом окне **Темы**. Нажать **ОК**.



Примечание: для редактирования свойств темы следует выбрать тему и нажать кнопку **Изменить**. Откроется **Редактор тем**, где можно будет выбрать пункт темы и область темы

Подсказки

Подсказками являются краткие текстовые сообщения, выводимые на экран монитора при помещении курсора в область отображения какого-либо графического объекта. Подсказки выводятся вблизи объектов, над которыми располагается курсор мыши.
Внешний вид страницы свойств **Подсказки** показан на рисунке.



*Внешний вид страницы свойств **Подсказки** диалоговой панели **Параметры приложения***

Флажок **Выводить подсказки для динамических объектов** при установке позволяет выводить краткую информация о графических объектах, связанных с тегами серверов OPC, локальными переменными или псевдонимами. Содержание выводимой информации соответствующим образом определяется флажками **Показывать имя объекта**, **Показывать описание объекта**, **Показывать имена источников данных** (или Показывать только имя тега (ItemID), **Показывать значения источников данных**.
Для статических объектов экранной формы, которые не связаны с данными серверов OPC и другими переменными, возможно отображение имен и описаний.

Совместимость.
Расширение возможностей Отчета событий (Event Log) включает в себя дополнительные события GraphWorX32, которые могут быть занесены в отчет:

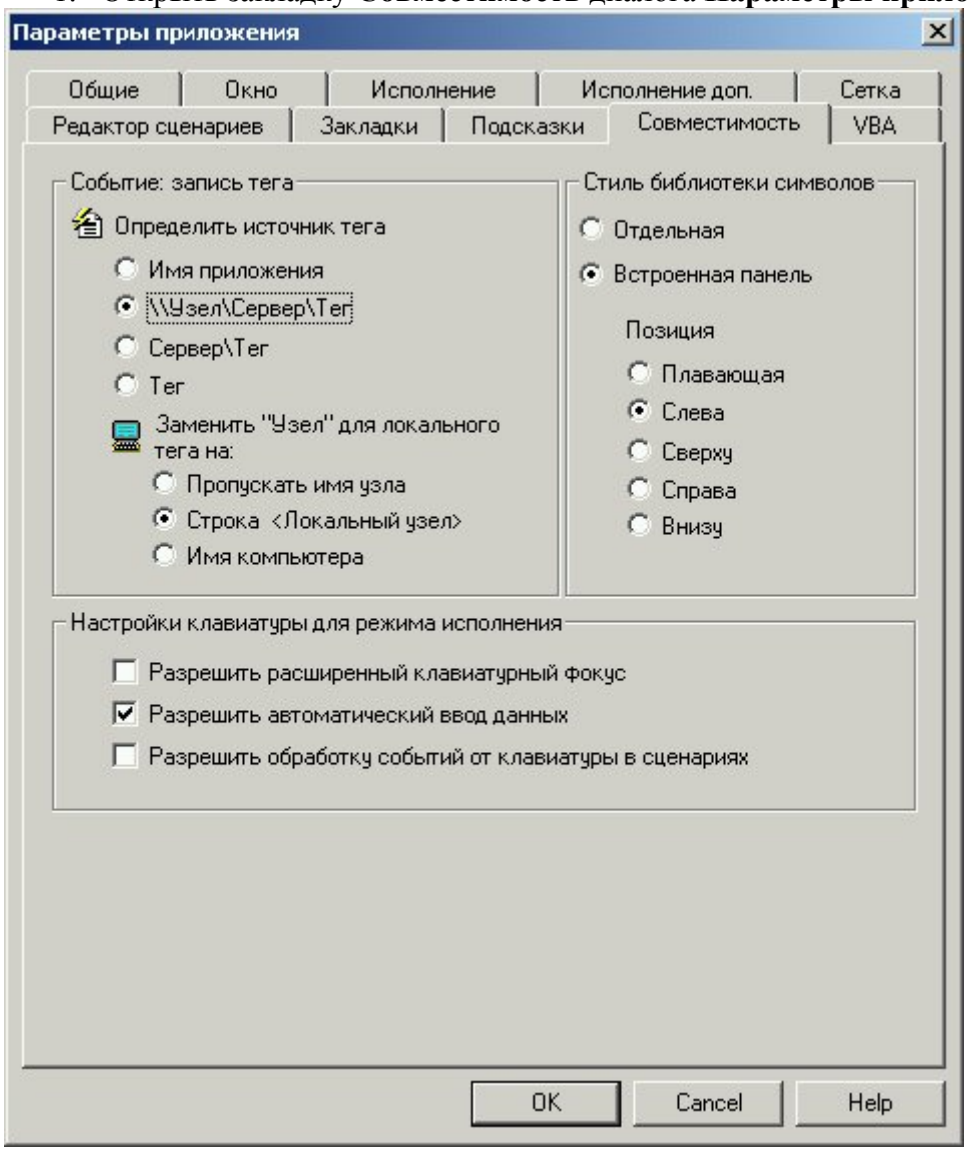
Запуск GraphWorX32;
Закрытие GraphWorX32;
Сохранение пользователем файла GraphWorX32.

Когда пользователь сохраняет файл GraphWorX32, Серверу Безопасности передается сообщение, содержащее полный путь к измененному файлу и имя пользователя.

При записи данных OPC из экрана GraphWorX32 в поле **variant prevVal** Отчета событий (Event Log) можно определить «Источник» (в предыдущих версиях GraphWorX32 там всегда было имя приложения).

Для конфигурирования типа источника:

1. Открыть закладку **Совместимость** диалога **Параметры приложения**.



2. Выберите имя источника, которое может быть одним из следующих:
имя приложения (как в предыдущих версиях)
\\Узел\Сервер\Тег (если выбрана эта опция, можно заменить имя узла для локального тега на одну из строк, задаваемых опцией Заменить «Узел» для локального тега на)
Сервер\Тег
Тег

3. Нажать **ОК**. Текущая конфигурация сохранится в системном реестре.

Стиль библиотеки символов

В GraphWorX32 используются два режима Библиотеки Символов:
Отдельная или **Встроенная панель символов**.

Если установлен режим **Отдельная**, нажатие на кнопке **Символы** на панели инструментов **Рисование** запускает **SymbolLibrary.exe**. Также можно выбрать Импорт > **Символ** из меню **Рисование**.

Если установлен режим **Встроенная панель символов**, Библиотека Символов располагается вместе с Панелью Символов в экранной форме. Можно выбрать расположение Панели Символов – справа, слева, вверху, внизу или в плавающем окне.

Настройки клавиатуры для режима исполнения

- Разрешить расширенный клавиатурый фокус – фокус, подобный Internet Explorer и другим приложениям. «Клавиатурный» фокус остается виден на последнем выделенном управляющем элементе, пока указатель мыши перемещается и располагается над другим объектом.
- Разрешить автоматический ввод данных – автоматически переключает изменение параметра в режим ввода, позволяя вводить любое значение. Если этот режим не включен, ввод данных возможен по нажатию клавиши Enter.
- Разрешить обработку событий от клавиатуры в сценариях – GraphWorX32 инициирует события, которые могут быть обработаны в сценариях VBA или VBScript/Jscript.

В дополнениее к вышеперечисленным возможностям, добавлены следующие методы автоматического контроля фокуса в режиме исполнения и упрощения навигации:

GwxDisplay

SetRuntimeFocus (ObjectName As String, bDataEntry As Boolean)
Установить фокус в режиме исполнения для видимых объектов ObjectName, содержащих свойства **Указание и щелчок, Параметр, Ползунок, Круговая шкала.**

SetRuntimeFocusNext (bWrap As Boolean, bDataEntry As Boolean)
Переход к следующему объекту (действие аналогично нажатию клавиши Tab).

SetRuntimeFocusPrev(bWrap As Boolean, bDataEntry As Boolean)
Переход к предыдущему объекту (действие аналогично нажатию клавиш Shift+Tab).

Object GetRuntimeFocus()
Функция возвращает фокусированный объект

CancelRuntimeFocus ()
Отменить фокус в режиме исполнения

GwxProcessPoint

SeRuntimetFocus (bDataEntry As Boolean)
Установить фокус на конкретном объекте Параметр. Функция возможна только на полях **DE.**

GwxPick, GwxLocation, GwxRotation

SetRuntimeFocus ()
Установить фокус на конкретных объектах **Указание и щелчок, Ползунок, Круговая шкала.**

Параметры в этих функциях определяются следующим образом:

ObjectName
Имя видимого объекта, который содержит данные динамические свойства, позволяющие вводить данные.

BWrap
Булевое значение, которое позволяет по нажатию табуляции переключаться между объектами, от первого до последнего и обратно.

BDataEntry
Булевое значение, атоматически открывающее ввод данных для редактирования; этот параметр невозможен при активном поле Указание и щелчок.

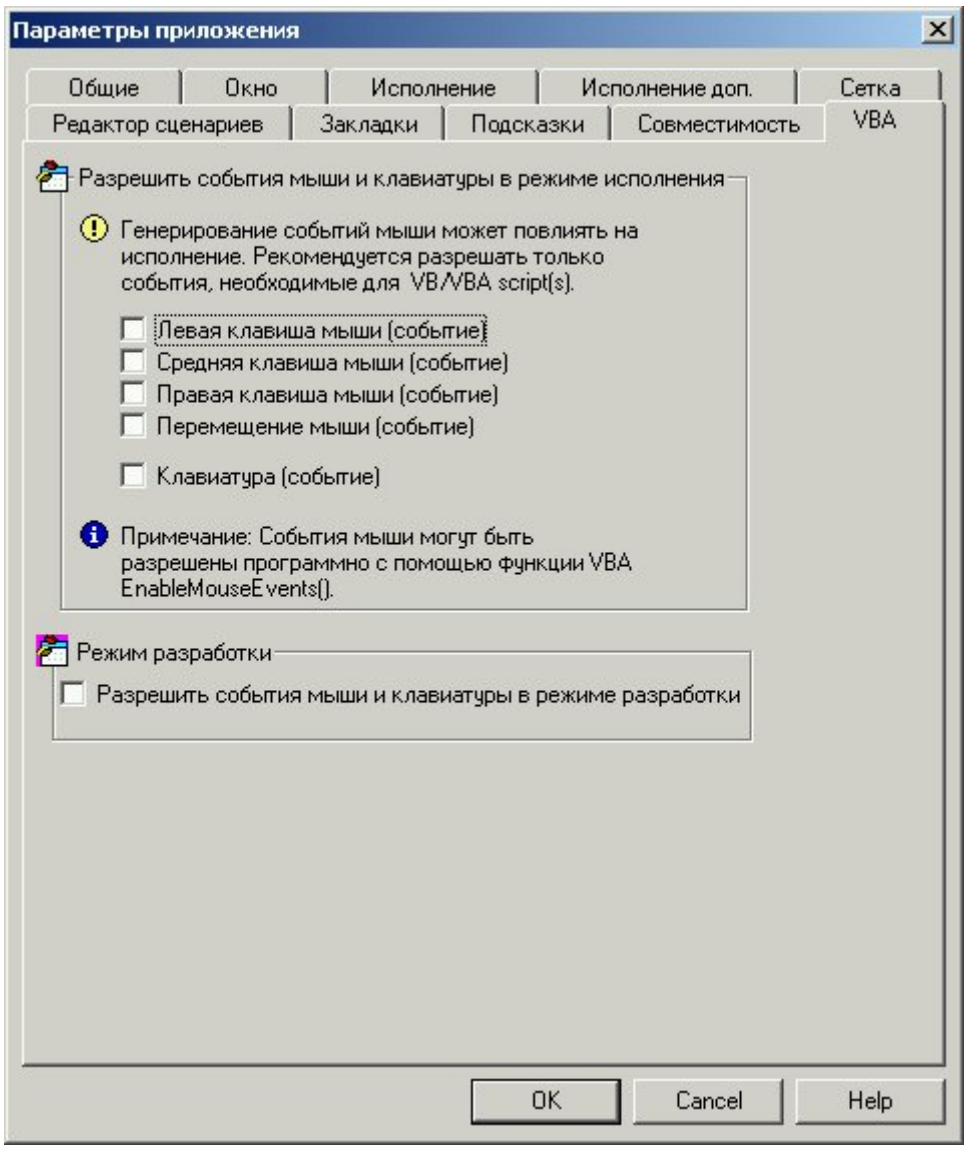
VBA. События мыши и клавиатуры.

Ранние версии GraphWorX32 (до 5.0) позволяли обрабатывать события мыши и клавиатуры в VBA и в контейнере. Однако, для улучшения параметров исполнения, из последующих версий эти события были удалены.

Они были включены в версию 7.0 снова, и сохранились в версии 8.0, но для исключения проблем с исполнением, включение этих событий может производиться как глобально, так и для каждого экрана в отдельности.

Глобальная настройка событий (для всех экземпляров GraphWorX32 на компьютере):

1. Выберите закладку **VBA** диалога **Параметры приложения.**
2. Выберите события мыши/клавиатуры, которые будут затем обрабатываться VBA или контейнером. Нажмите **ОК**. Настройки сохранятся в системном реестре после закрытия GraphWorX32, и восстановятся из него при запуске GraphWorX32.



Локальная настройка событий (для одного экрана)
Настройку событий мыши и клавиатуры можно выполнить программно с помощью функции

Sub EnableRuntimeEvents(_
LeftButton As Boolean, MiddleButton As Boolean, RightButton As Boolean, _
MouseMove as Boolean, Keyboard as Boolean)

Пример использования метода:

```
Private Sub GwxDisplay_PreRuntimeStart()
    ' Enable right click events only
    Call ThisDisplay.EnableRuntimeEvents(False, False, True,
False, False)
End Sub
```

Настройка событий мыши в режиме разработки (для одного экрана) .
Включить/отключить события мыши для режима разработки можно в закладке VBA диалогов **Параметры приложения** и **Свойства экранной формы**.
Эту настройку можно выполнить программно с помощью функции:

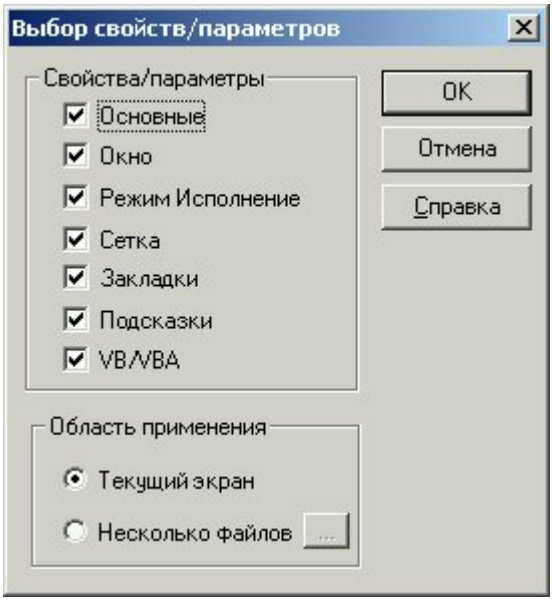
```
Sub SetDesignModeEvents(Enable As Integer)
Function GetDesignModeEvents() As Integer
```

2.4 Копирование и сброс свойств и параметров

Имеется возможность присвоения параметрам GraphWorX32 и свойствам каждой экранной формы значений по умолчанию, которые они имели до внесения изменений пользователем (восстановить параметры/свойства).

Для восстановления свойств экранной формы следует выбрать **строку Восстановить свойства экранной формы** меню **Формат**. Для восстановления параметров приложения следует выбрать строку **Восстановить параметры приложения**.

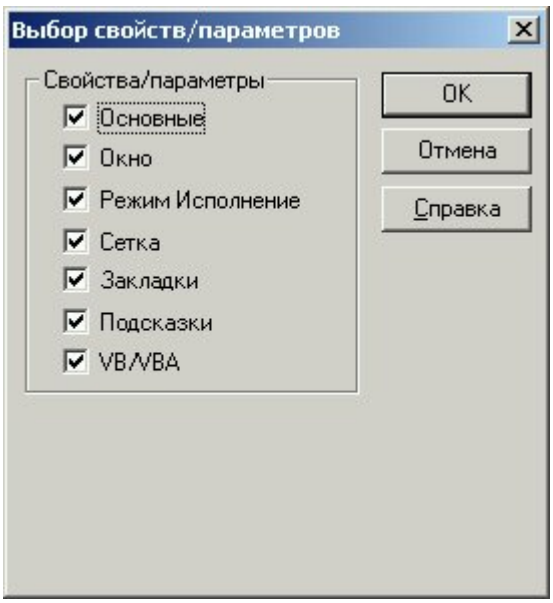
Кроме того, имеется возможность скопировать все или часть значений параметров приложения в свойства текущей загруженной экранной формы и наоборот, для чего служат команды **Сохранить свойства как параметры** и **Применить параметры к свойствам** меню **Формат**. При этом возможно указать, копирование каких значений требуется выполнить. При выборе строки **Применить параметры к свойствам** на экран монитора выводится диалоговая панель **Выбор свойств/параметров**, внешний вид которой показан на рисунке:



*Внешний вид диалоговой панели **Выбор свойств/параметров** при копировании параметров приложения в свойства экранной формы*

Имеется возможность применить параметры приложения к одной или нескольким экранным формам, для чего используется группа **Область применения**.

При выборе строки **Сохранить свойства как параметры** на экран монитора выводится диалоговая панель **Выбор свойств/параметров**, показанная на рисунке, которая позволяет указать, значения каких страниц свойств текущей экранной формы следует применить к параметрам GraphWorX32.

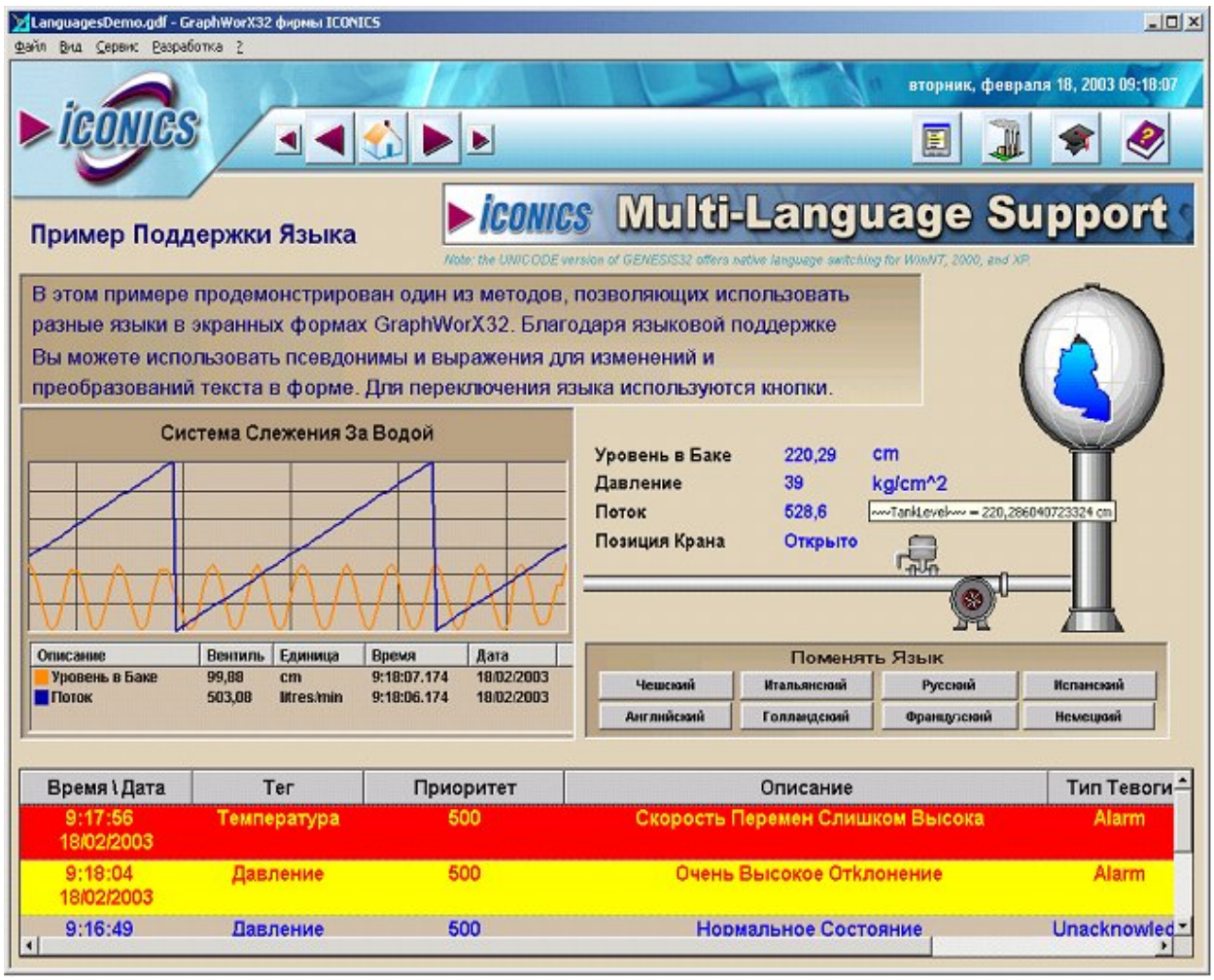


*Внешний вид диалоговой панели **Выбор свойств/параметров** при копировании свойств экранной формы в параметры приложения*

2.5 Поддержка переключения языков.

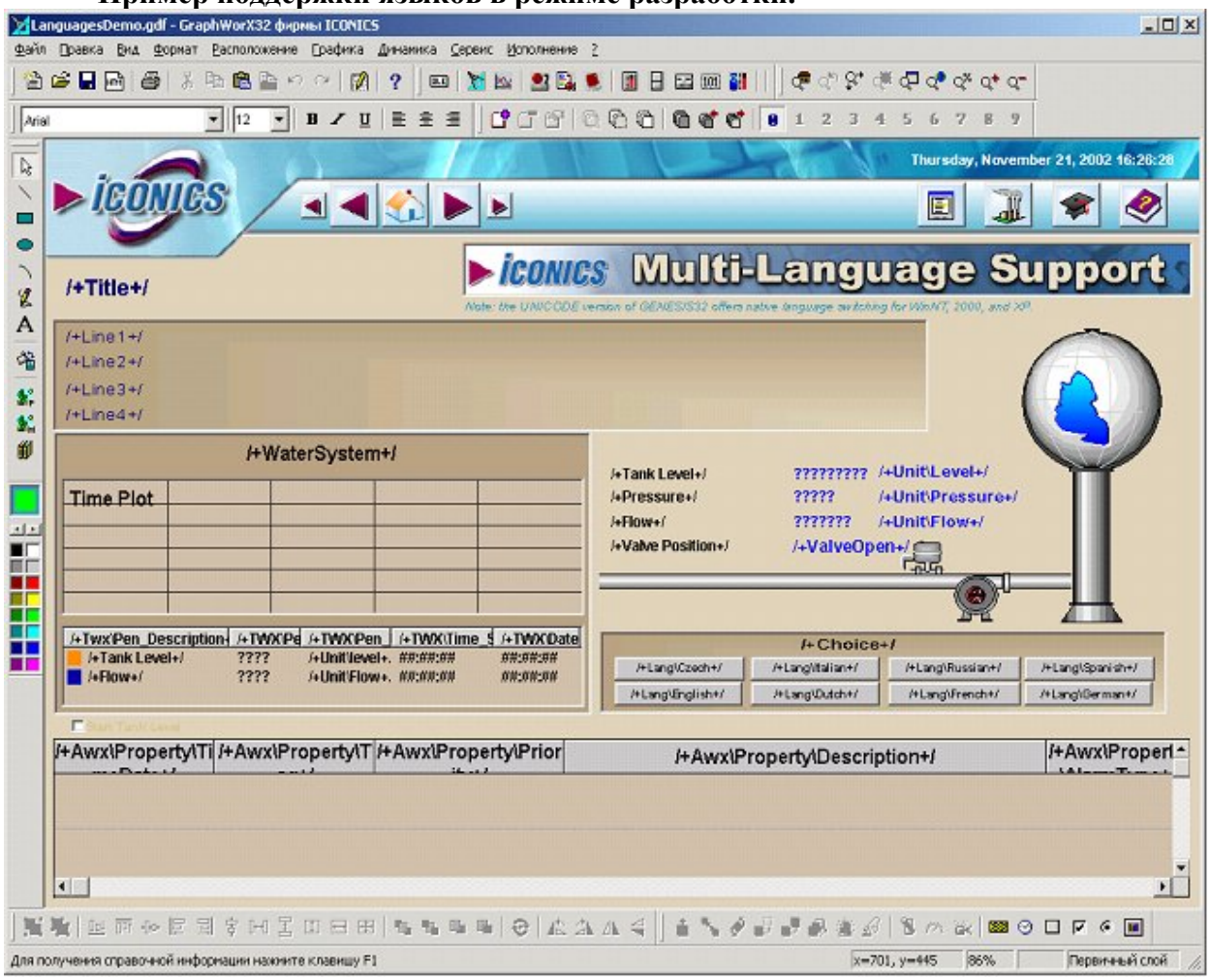
GraphWorX32 сохраняет строки в файле ресурсов, и их можно модифицировать в соответствии с пользовательскими настройками. Кроме того, для поддержки языков библиотека dll, содержащая ресурсы, может быть загружена в реальном времени. Эта библиотека компилируется как стандартный файл Win32 .dll. Каждый раз, когда в приложении ICONICS переключается язык, GraphWorX32 загружает соответствующую библиотеку ресурсов.
Unicode - версия GraphWorX32 также поддерживает псевдонимы языков с автоматическим масштабированием диапазона и значений, преобразованием единиц измерения, форматов и выбором шрифта. Дополнительная информация о поддержке переключения языков в версии Unicode содержится в документации на **Конфигуратор Языков** (**ICONICS Language Configurator).**
Примечание: при переключении языков наиболее целесообразно выбирать шрифт **Microsoft Arial Unicode**, который содержит все символы Unicode. Для этого до начала конфигурирования следует выбрать пункт **Параметры приложения** из меню **Формат** и на странице **Общие** установить шрифт Arial Unicode MS. Шрифт может быть установлен отдельно в диалоге **Свойства** каждого ActiveX компонента.

Пример переключения языков.
При инсталляции Unicode - версии GENESIS32 устанавливается образец конфигурации Сервера Языков в папке Languages в каталоге установки GENESIS32. В этой же папке располагается пример экранной формы "languagesDemo.gdf" , которая включает образец конфигурации Просмотра. На рисунке изображена эта экранная форма в режиме исполнения.

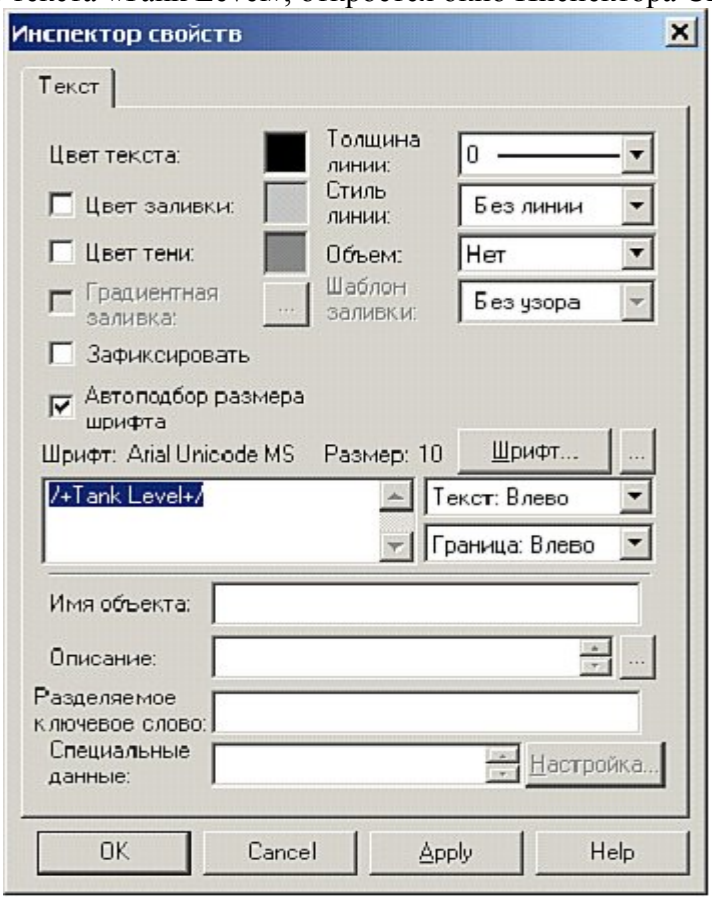


Пример поддержки языков в режиме исполнения

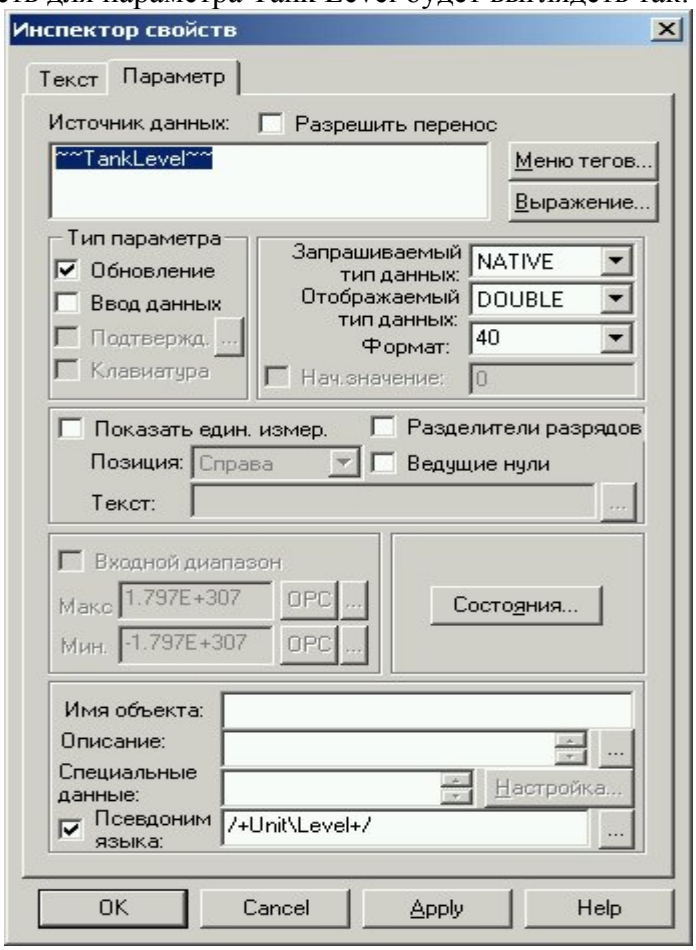
Пример поддержки языков в режиме разработки.



Если щелкнуть на окне текста «Tank Level», откроется окно Инспектора Свойств:



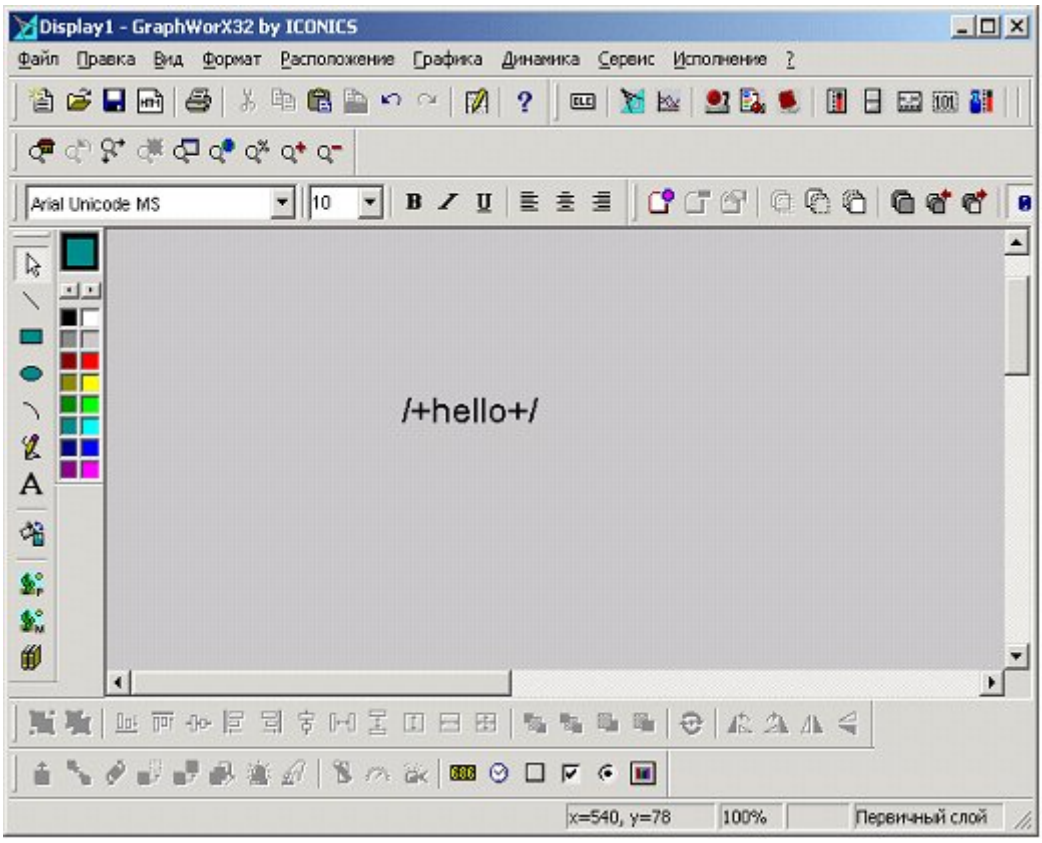
Окно Инспектора Свойств для параметра Tank Level будет выглядеть так:



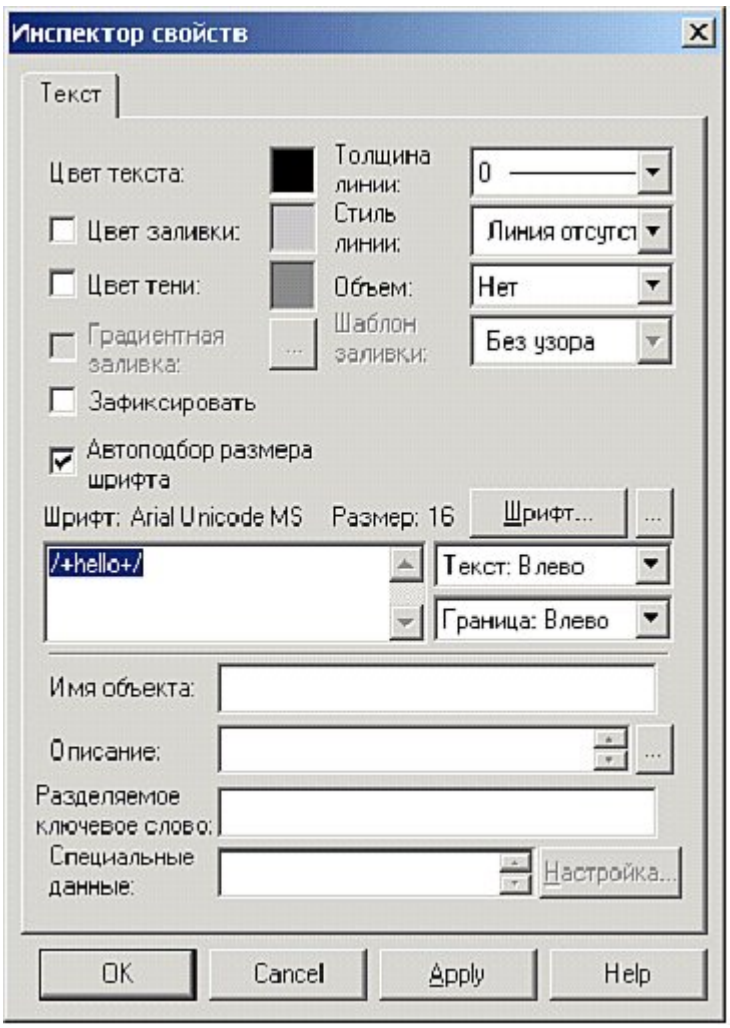
Внизу закладки появляется окно, в котором в разделителях /+ и +/ находится строка, определяющая псевдоним языка.
Примечание: переключение языков в версии Unicode не зависит от выбора библиотеки ресурсов. Библиотека ресурсов применяется в основном для текста в диалоговых окнах и других элементах пользовательского интерфейса.

Создание строк с переключением языков.
Псевдонимы могут использоваться в Инспекторе Свойств для преобразования единиц, даты/времени, выбора шрифта. Во время исполнения GraphWorX32 обращается к **Серверу Языков** (ICONICS Language Server) для привязки строк с переключением языков.
В следующем примере показано, как сконфигурировать строки с переключением языков в GraphWorX32.

Вообще, можно использовать псевдонимы языков в GraphWorX32 почти везде, где встречается текст. Для создания строки с переключением языков необходимо:
1. В режиме разработки GraphWorX32 выбрать кнопку **Текст** на панели инструментов **Рисование** и создать окно текста на экране.
2. Введите в окне нужный текст в разделителях «/+» и «+/» (например, “hello”). По разделителям **Серверу Языков** определит, что слово hello является строкой с переключением языков.



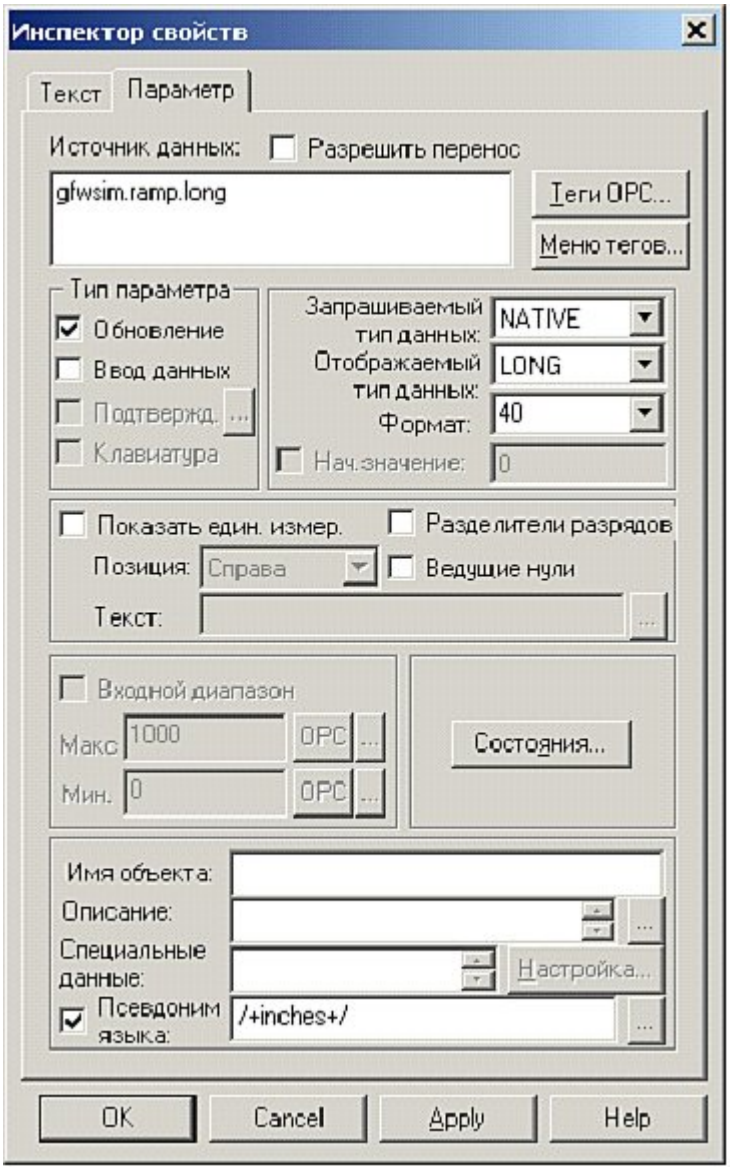
3. Сделать двойной щелчок на окне текста для открытия Инспектора Свойств.



4.Открыть Конфигуратор Языков (ICONICS Language Configurator), выбрав **Пуск – Программы - ICONICS GENESIS32 - Languages - Language Configurator.**
Настроить новый псевдоним языка в базе данных языков для текстовой строки, которая только что была создана, причем имя создаваемого псевдонима в Конфигураторе Языков должно точно соответствовать этой текстовой строке.
Если этого не сделать, Сервер Языков не сможет определить, как перевести текст.
Далее следует установить языки, между которыми будет переключать текст псевдоним, и ввести для каждого языка текст перевода. Например, текст будет "Hello, how are you?" для английского языка, "Guten tag" – для немецкого и «Как дела?» - для русского. Для получения дополнительной информации см. справку по Конфигуратору Языков.
5. Перевести проект в режим исполнения. Для переключения языка следует выбрать пункт **Выбор языка** из меню **Вид**. В зависимости от выбранного языка в строке будет отображаться тот текст, который был настроен в Конфигураторе Языков.

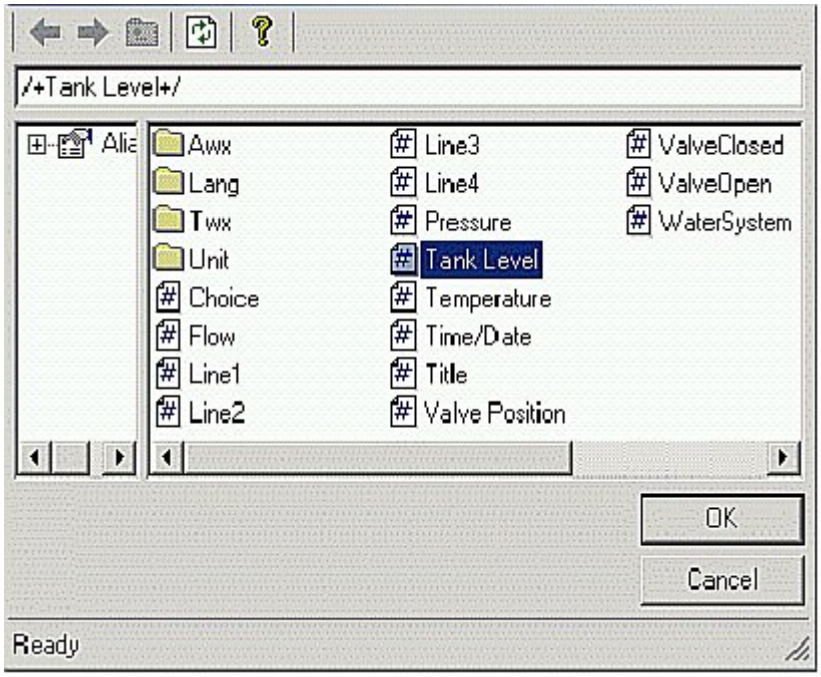
Другие строки с переключением языков.
Псевдонимы языков могут использоваться в **Инспекторе Свойств** для преобразования единиц, даты/времени, выбора шрифта. Допустим, что нужно создать строку с переключением языков для текста "/+inches+/" (дюймы). С помощью Конфигуратора Языков можно создать псевдоним **inches** для текстовой строки. Предположим, Вы хотите ввести текст «**inches**» для английского языка и «**см**» (сантиметры) – для русского.
1. Если необходимо связать псевдоним **inches** с динамикой, то следует указать Конфигуратору Языков, как преобразовать дюймы в сантиметры. Для этого создается новое выражение и связывается с псевдонимом.

2. После добавления выражения, переводящего дюймы в сантиметры, надо связать псевдоним с точкой ввода/вывода. Для этого выбрать меню **Динамика** – **Встроенные объекты** – **Значение параметра**.



3. Открыть окно Инспектор Свойств параметра. Выбрать источник данных (например, имитационная переменная "gfwsim.ramp.long").
4. Установить флажок **Псевдоним языка** внизу окна **Инспектора Свойств**.
5. Ввести имя псевдонима языка в соответствующем окне вручную, или нажать кнопку **...** и выбрать его с помощью Навигатора псевдонимов языков.
6. Перевести проект в режим исполнения и проверить изменения значения при переключении языков через пункт **Выбор языка** из меню **Вид**.

Выбор псевдонимов языков с помощью Навигатора псевдонимов языков.
1. В окне **Инспектора Свойств** установить флажок **Псевдоним языка** внизу окна и нажать кнопку **...** для выбора его с помощью Навигатора псевдонимов языков.
2. Откроется окно Навигатора псевдонимов языков. Навигатор включает все псевдонимы, содержащиеся в базе данных языков. Выбор псевдонима осуществляется двойным щелчком мыши, после чего выбранный псевдоним появляется в верхней строке и дополняется разделителями.
3. Нажать ОК.



2.6 Поддержка глобальных псевдонимов.

В GraphWorX32 поддерживаются различные уровни глобальных псевдонимов – уровень документа, процесса, компьютера.
Глобальный псевдоним – это строка, позволяющая ссылаться на различные источники данных, используя одно уникальное имя. Это позволяет существенно уменьшить количество необходимых экранных форм. Система Глобальных псевдонимов включена в следующие функции GraphWorX32:

Видимые объекты (прямоугольники, эллипсы, линии и т.д.)
- Описание
- Команда пользователя
- Навигатор глобальных псевдонимов на общей странице.

Динамика (размер, вращение и т.д.)
- Теги OPC, Верхний и нижний пределы
- Описание
- Команда пользователя
- Навигатор глобальных псевдонимов для выражений и соединений

Текст (текст, кнопки и т.д.)
- Надпись
- Навигатор глобальных псевдонимов для надписей и описаний

Кнопки экрана
- Файл экранной формы (поддерживается автоматически – наследование от PICK)
- Надпись (поддерживается автоматически – наследование от Текста)
- Навигатор глобальных псевдонимов для надписей и описаний.

Постоянные поля
- Постоянное значение
- Постоянная строка
- Исходное значение

Доступ к Навигатору Глобальных псевдонимов из полей Описание и Псевдоним языка для Аналогового Индикатора, Аниматора, Цвета, Аналогового цвета,

Цифрового индикатора, Прерывистое отображение, Скрыть, Положение, Указание и Щелчок, Параметр, Вращение, Размер, Время/Дата.
Кнопки просмотра для локальных переменных – начальное значение, верхний предел, нижний предел, закладки экранной формы, свойства окна
Свойства экрана
Заголовок
Загрузка значения
Переключение значения
Новая функция кнопки – Выбор Темы глоб. псевдонимов

Командная строка GraphWorX32 позволяет задать Стартовую тему Глобальных псевдонимов в качестве аргумента в командной строке:
-Themes="MACHINE=<Buildings=Building1"

Дополнительная информация о глобальных псевдонимах содержится в документации на **Конфигуратор глобальных псевдонимов**.

Глава 3. Управление файлами экранных форм.

3.1 Введение
3.2 Расширения файлов GraphWorX32
3.3 Работа с файлами экранных форм GraphWorX32
3.4 Завершение работы GraphWorX32
3.5 Шаблоны
3.6 Слои

3.1 Введение

Настоящий раздел содержит указания по работе с файлами экранных форм GraphWorX32. Функции создания, открытия, сохранения и печати файлов экранных форм реализованы в соответствующих командах меню **Файл**. Кроме того, меню **Формат** содержит команду **Пароль для режима Разработка**, позволяющий ограничить доступ к файлам экранных форм в режиме *Разработка*.

Команды меню Файл

Command	Shortcut Keys	Function
Создать	**CTRL +N**	**Создать новую экр. форму**
Открыть	**CTRL +O**	**Открыть экр. форму.**
Сохранить	**CTRL +S**	**Сохранить экр. форму.**
Сохранить как		**Сохранить экр. форму с определенным типом, именем файла или версией.**
Печать		**Печать текущей экр. формы.**
Предварительный просмотр		**Показать вид напечатанного экрана перед печатью**
Настройка печати		**Задать настройки печати**
Выход		**Закрыть приложение**

3.2 Расширения файлов GraphWorX32

В GraphWorX32 файлы могут иметь следующие расширения:

- Файлы экранных форм GraphWorX32 – *.gdf
- Файлы шаблонов GraphWorX32 – *.tdf
- Файлы экранных форм GraphWorX32 для Windows CE – *.gdc
- Файлы шаблонов GraphWorX32 для Windows CE – *.tdc
- Файлы категорий символов – *.sdf

3.3 Работа с файлами экранных форм GraphWorX32

Создание

Для создания нового файла экранной формы GraphWorX32 выберите строку **Создать** меню **Файл**.



Загрузка

Для загрузки (открытия) существующего файла экранной формы GraphWorX32 выберите строку **Открыть** меню **Файл**.



Сохранение под другим именем

Для сохранения текущей экранной формы в файле выберите команду **Сохранить как…** меню **Файл**. На экран монитора будет выведена диалоговая панель, показанная на рис. 0-1.

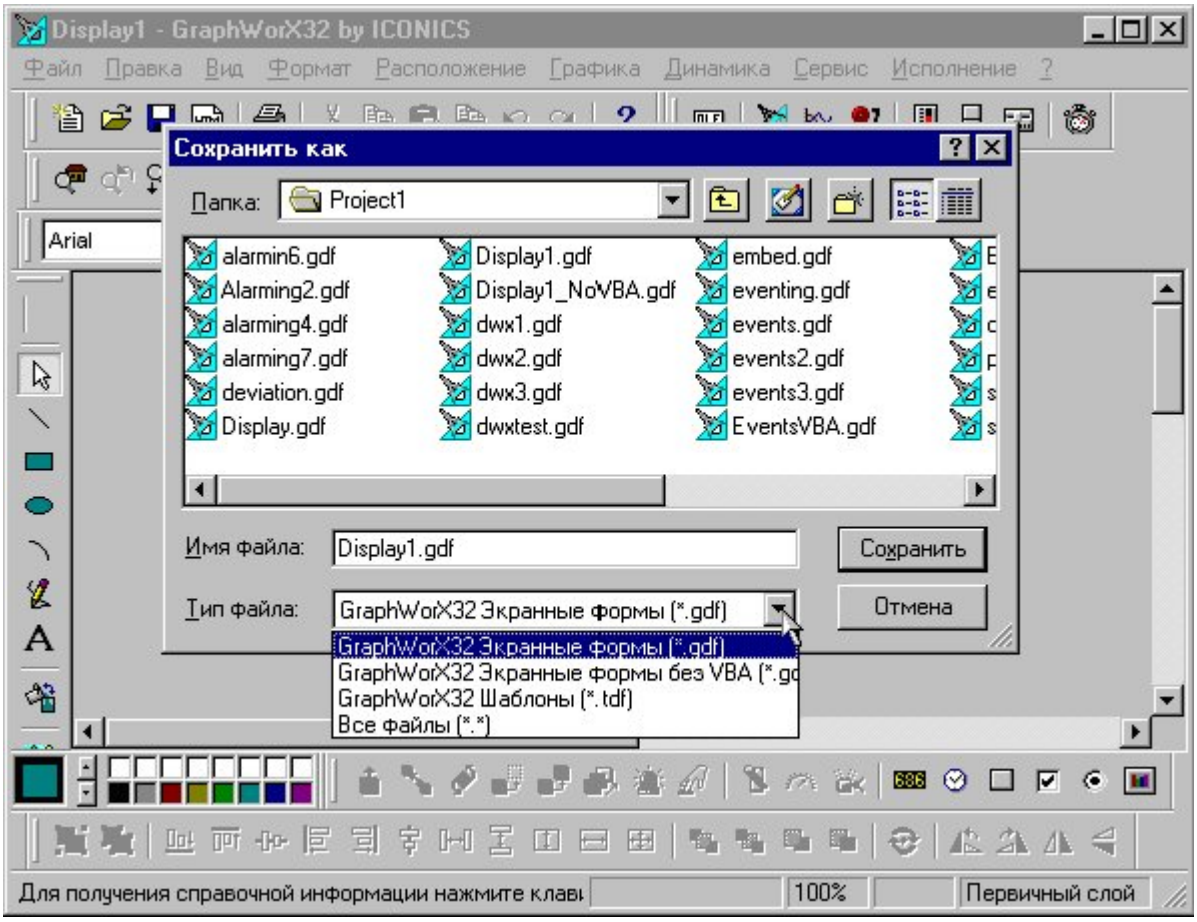


Рис. 0-1. Внешний вид диалоговой панели ***Сохранить как***

Поле **Имя файла** диалоговой панели позволяет ввести имя файла. При этом расширение файла, выбранное в поле **Тип файла**, будет автоматически добавлено к имени файла при сохранении.

Выпадающий список **Тип файла** предназначен для выбора типа файла: стандартной экранной формы со сценарными процедурами VBA, файл экранной формы без VBA или шаблон GraphWorX32 (*.tdf).

Примечание. Каждая экранная форма по умолчанию содержит код VBA, даже если в нее не включено ни одной сценарной процедуры пользователя. Если экранную форму предполагается просматривать элементом просмотра экранных форм GWXView32 ActiveX или публиковать на Web-сервере, следует сохранить ее с использованием опции *Экранные формы без VBA*.

Сохранение

Для сохранения изменений, внесенных в текущую открытую экранную форму, следует выбрать команду **Сохранить** меню **Файл** или нажать кнопку *Сохранить* в главной панели инструментов GraphWorX32.



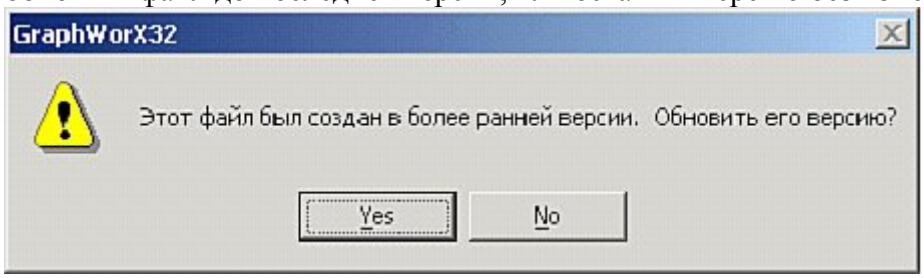
Сохранение в предыдущей версии

Функция **Сохранить в предыдущей версии** обеспечивает совместимость текущих экранных форм с предыдущими версиями GraphWorX32. GraphWorX32 поддерживает совместимость «снизу вверх», т.е. старые файлы экранных форм могут быть загружены в новую версию GraphWorX32.

Функция **Сохранить в предыдущей версии** позволяет сохранять файлы экранных форм, созданные в более поздней версии GraphWorX32, в варианте для более ранних версий, так что файл экранной формы может быть беспрепятственно загружен в предыдущую версию GraphWorX32.

Открыть экранную форму.

При открытии файла экрана GraphWorX32 определяет его версию. Если файл создан в более ранней версии, GraphWorX32 выдает сообщение, предоставляющее возможность обновить файл до последней версии, или оставить версию без изменений.



Для того, чтобы сообщения не выдавались, следует установить флажок **Загрузить предыдущую версию без предупреждений** (только в диалоге Параметры приложения).

Для обновления экранной формы следует нажать кнопку Да(Yes). Экран будет сохранен со всеми новыми функциями последней версии. В дальнейшем такой экран не сможет быть загружен в GraphWorX32 более старой версии.

Для сохранения исходной версии файла следует нажать кнопку Нет (No).

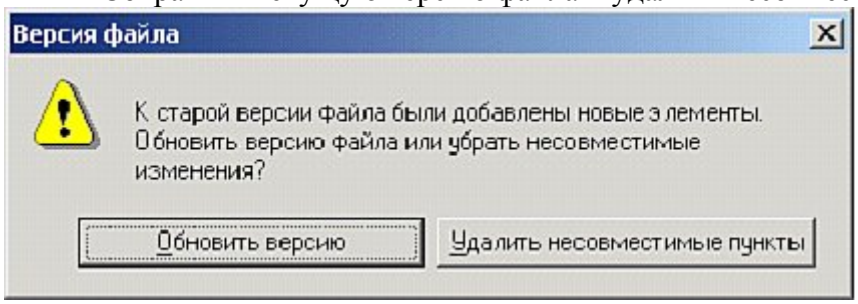
Сохранить экранную форму.

При сохранении файла экрана GraphWorX32 определяет, какие функции были добавлены в экранную форму и поддерживаются ли они выбранной версией. Версия может быть изменена в пункте меню **Файл – Сохранить как.**

Если версия самая новая, или в файл не было добавлено несовместимых функций, GraphWorX32 сохраняет изменения без предупреждающих сообщений. Если были добавлены несовместимые функции, GraphWorX32 информирует об этом и предоставляет две возможности:

Обновить версию файла и сохранить все новые изменения. В дальнейшем такой экран не сможет быть загружен в GraphWorX32 более старой версии.

Сохранить текущую версию файла и удалить несовместимые функции.



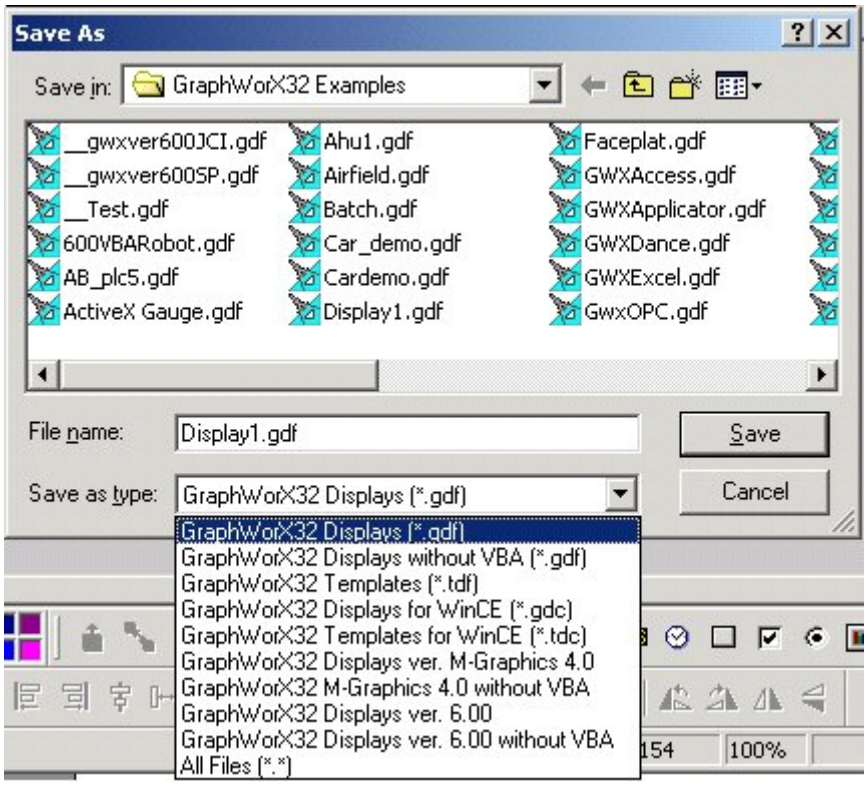
Для того, чтобы сообщения не выдавались, следует установить флажок **Загрузить предыдущую версию без предупреждений** (только в диалоге Параметры приложения): Меню **Формат – Параметры приложения – Общие**.

Сохранение в определенной версии

Для сохранения экранной формы в одной из предыдущих версий:

1. Выбрать пункт **Сохранить как** из меню **Файл.**

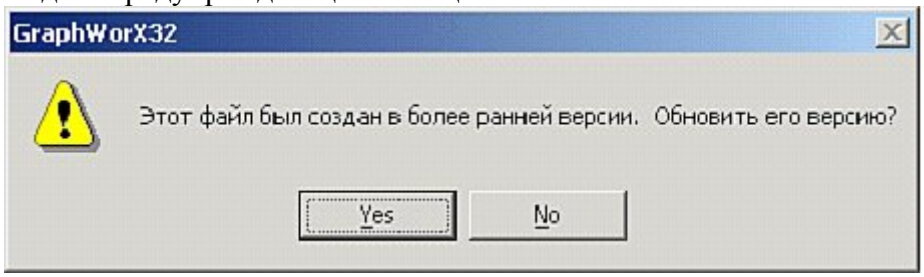
2. В диалоговом окне **Сохранить как** выбрать версию GraphWorX32 из выпадающего списка **Тип файла** (Save as type).



3. Нажать кнопку **Сохранить** (Save). Файл будет сохранен в формате выбранной версии. Информация о версии хранится внутри файла, поэтому при следующем сохранении командой **Сохранить** из меню **Файл** экранная форма будет сохранена уже в выбранной версии.
Примечание: экранная форма может быть сохранена как с VBA, так и без VBA. Версия без VBA требуется для GraphWorX32 Viewer ActiveX и для GraphWorX32 без VBA.

Загрузка экранной формы в режиме исполнения.

Если экран загружается в режиме исполнения, информационные сообщения о версии подавляются, но процесс исполнения не останавливается. При переводе в режим разработки GraphWorX32 проверяет версию экранной формы и, в случае несоответствия, выдает предупреждающее сообщение:



Для того, чтобы сообщения не выдавались, следует установить флажок **Загрузить предыдущую версию без предупреждений** (только в диалоге Параметры приложения): Меню **Формат – Параметры приложения – Общие**.

Возможности OLE Automation.

Если экранная форма загружается или сохраняется с помощью методов **OLE Automation,**

предупреждающие сообщения подавляются автоматически, и таким образом, не прерывают исполнение сценария. Три новых метода позволяют контролировать соответствие версий в OLE Automation:

- `Функция GetFileVersion() As long`
- `Функция GetFileContainsVBA() As Boolean`
- `Функция SetFileVersion(NewFileVersion As long) As long`

Функция GetFileVersion() **возвращает следующие значения:**

- `-1    .. OpenFileError (&HFFFFFFFF)`
- `0     .. Ver520`
- `1     .. Ver600`
- `2     .. Ver600SP`
- `3     .. VerMGraphics40`
- `4     .. Ver610`
- `9999  .. VerLatest (&H270F)`

Функция **GetFileContainsVBA()** возвращает значение True, если экран содержит код VBA и False, если не содержит. Наличие кода VBA контролируется при загрузке экрана без VBA и сохранении экрана без VBA, или при использовании методов **FileSaveAsType(1)** (с VBA) и **FileSaveAsType(1)** (без VBA).

Функция **SetFileVersion()**устанавливает версию экранной формы для следующего сохранения. Обратите внимание, что количество различных версий для сохранения, меньше, чем количество версий, определяемых при чтении файла. Другими словами, GraphWorX32 определяет большое количество версий, но позволяет сохранять в формате недавних версий; сохранение в версиях 5.20 и более старых не поддерживается. При попытке сохранения файла в одной из старых неподдерживаемых версий, он будет сохранен в версии SetVer600.

Входные значения SetFileVersion()

- `2     .. SetVer600SP`
- `3     .. SetVerMGraphics40`
- `4     .. SetVer610`
- `9999  .. SetVerLatest (&H270F)`

предоставляет большие возможности, и может использоваться для автоматизированной работы с экранными формами (например, если необходимо обновить версию или изменить её на предыдущую, или внести определенные изменения в экранные формы, сохранив существующую версию).

Visual Basic Code Sample

```
Private Sub DisplayVersion()
    ' Specify the path and filename of an existing display
    Const strPath As String = "C:\PathToYourDisplay\"
    Const strFilename As String = strPath & "YourDisplay.gdf"
    Const SetVer600SP as Long = 2
    Const VerLatest as Long = 9999

    ' Create GWX instance
    Dim g As New Gwx32.GwxDisplay

    ' Check if GWX was created
    If g Is Nothing Then
        MsgBox "Failed to create GWX"
        Exit Sub
    End If

    ' Show GWX in the front and load the requested display
```

```
    Call g.ShowWindow
    Call g.BringWindowToTop
    Call g.FileNew
    Call g.FileOpen(strFilename)

    ' Check the file version
    Dim ver As long
    ver = GetFileVersion(g)

    ' If this is the latest version, save in the
    ' version 6.00
    If (ver = VerLatest) Then
        g.SetFileVersion(SetVer600SP)
        g.FileSave
    End If

    ' Exit GWX, the FileNew suppresses the message box
    ' if to save changes.
    g.FileNew
    g.ExitApplication
End Sub

' Get file version and trace the current version in the
' Immediate window for debugging purposes.
Private Function GetFileVersion(g As GwxDisplay) As Long
    Const OpenFileError as Long = -1
    Const Ver520 as Long = 0
    Const Ver600 as Long = 2
    Const Ver600SP as Long = 2
    Const VerMGraphics40 as Long = 2
    Const Ver610 as Long = 2
    Const VerLatest as Long = 9999

    Dim ver As long

    ' Check the version
    ver = g.GetFileVersion
    Select Case (ver)
        Case OpenFileError:
            Debug.Print "Version: FileOpenError"
        Case Ver520:
            Debug.Print "Version: Ver520"
        Case Ver600:
            Debug.Print "Version: Ver600"
        Case Ver600SP:
            Debug.Print "Version: Ver600SP"
        Case VerMGraphics40:
            Debug.Print "Version: VerMGraphics40"
        Case Ver610:
            Debug.Print "Version: Ver610"
        Case VerLatest:
            Debug.Print "Version: VerLatest"
    End Select

    ' Check if the display contains VBA
    If g.GetFileContainsVBA Then
        Debug.Print "VBA: Yes"
    Else
        Debug.Print "VBA: No"
    End If

    GetFileVersion = ver
End Function
```

Известные ограничения.

Определение версий.

Ограничение касается определения версий для экранных форм, которые не содержат динамики, а только статические элементы и ActiveX-компоненты. GraphWorX32 определяет версию экранной формы по версии наиболее распространенных динамических объектов, используемых в файле. Если экранная форма содержит только статические объекты и ActiveX-компоненты, GraphWorX32 не может определить версию и считает, что файл имеет самую новую версию.

Статические объекты и ActiveX-компоненты.
Статические объекты не изменялись между версиями GraphWorX32 6.0 и 6.10, поэтому такие экраны могут быть открыты в предыдущей версии GraphWorX32.
ActiveX-компоненты хранят номер версии внутри себя, и GraphWorX32 не может его контролировать. Как следствие, экран, содержащий ActiveX-компоненты TrendWorX32 and AlarmWorX32 не может быть загружен в версию 6.00 SP, если он был сохранен в формате более новой версии GWX.

Объяснение проблемы.
В случае с ActiveX-компонентами GraphWorX32 запрашивает у каждого компонента перевод его содержимого в текущий экран, но GraphWorX32 не может контролировать, что и каким компонентом было переведено. Если установлена новая версия компонента, скорее всего, он будет сохранен в самой новой версии.

Ограничение доступа к экранным формам в режиме *Разработка*

Для защиты экранной формы паролем выберите строку **Пароль для режима Разработка** меню **Формат**. На экран монитора будет выведена диалоговая панель, показанная на рис. 3-2.

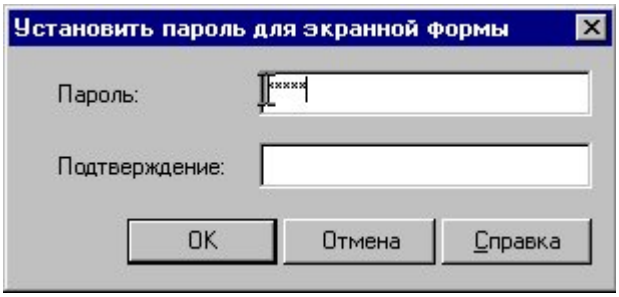
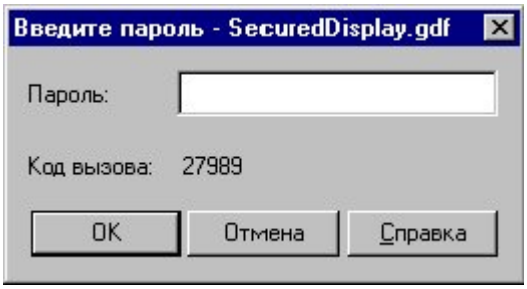


Рис. 3-2 Ввод пароля, ограничивающего доступ к экранной форме в режиме Разработка

Введите пароль в поля **Пароль** и **Подтверждение**. Следите за тем, чтобы выбранные регистр и раскладка клавиатуры совпадали при вводе как в первое, так и во второе поле. Нажмите кнопку **OK**. Если строки пароля, введенные в поля **Пароль** и **Подтверждение**, не совпадают, на экран монитора будет выведено сообщение "*Пароли не совпадают*".
После ввода пароля и закрытия диалоговой панели **Установить пароль для экранной формы**, сохраните экранную форму командой **Сохранить** меню **Файл**.
Последующие попытки открыть данную экранную форму для редактирования будут сопровождаться появлением диалоговой панели, показанной на рисунке. При вводе неправильного пароля в поле **Пароль** диалоговой панели **Введите пароль - <имя_файла>** на экран монитора будет выведено сообщение "*Неправильный пароль*" и попытка загрузки экранной формы для редактирования будет отвергнута.

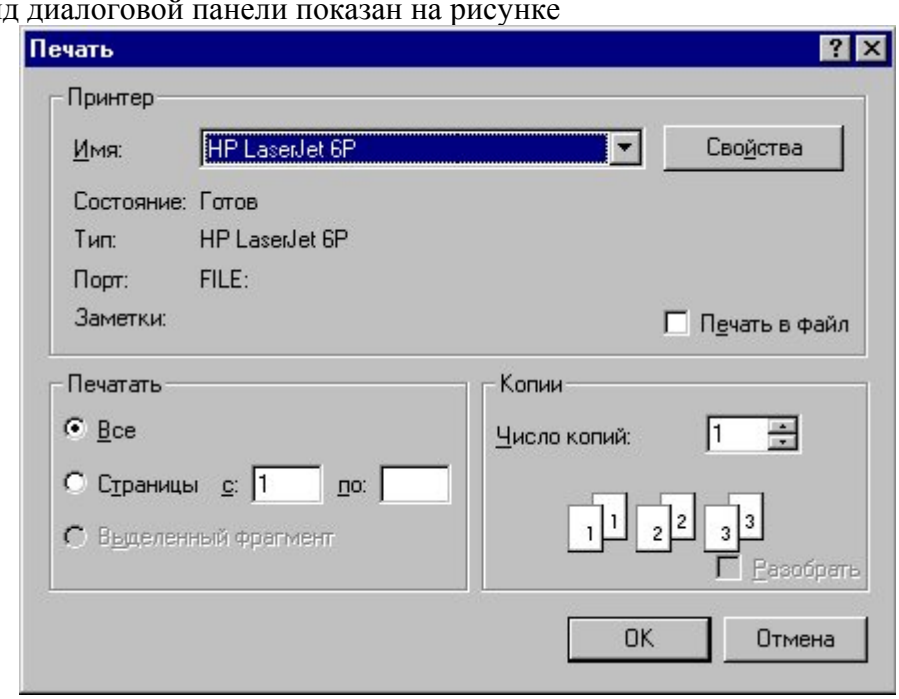


Ввод пароля при загрузке экранной формы для редактирования

Печать экранных форм GraphWorX32



Для печати текущей экранной формы выберите строку **Печать** меню **Файл** или нажмите кнопку *Печать* главной панели инструментов GraphWorX32. При этом на экран монитора будет выведена диалоговая панель, которая позволяет установить диапазон страниц, количество копий и другие параметры печати.
Функция печати может быть активизирована нажатием комбинации клавиш Ctrl+P.
Внешний вид диалоговой панели показан на рисунке

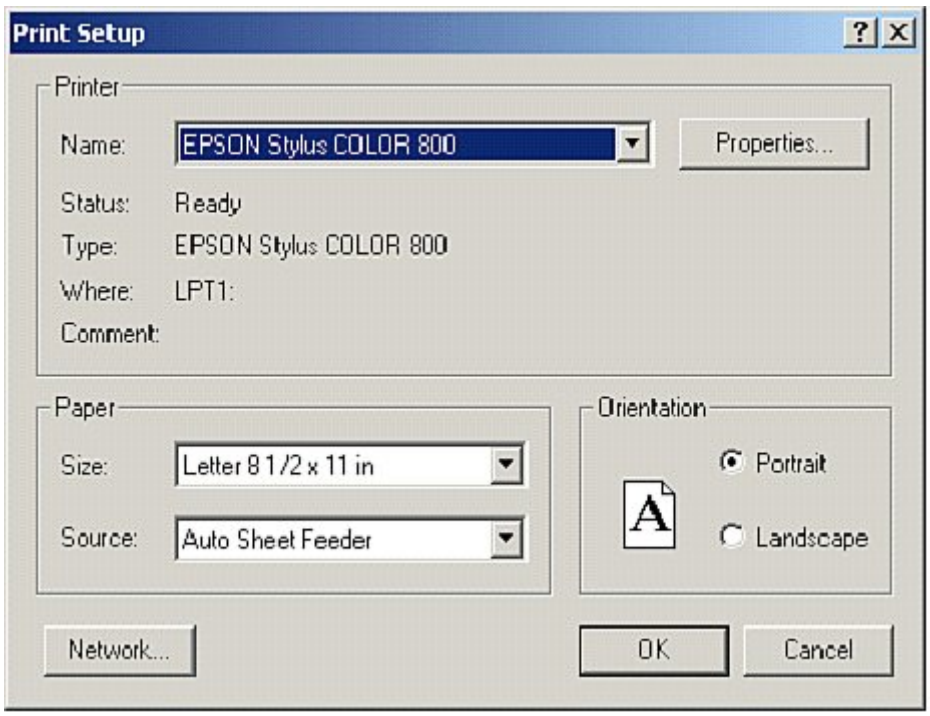


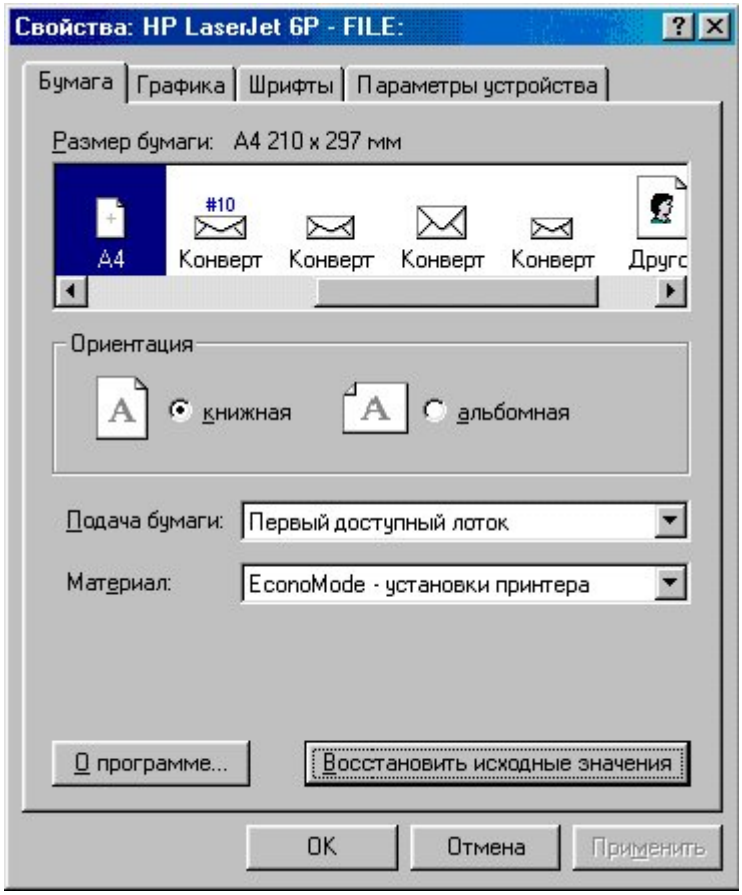
*Диалоговая панель **Печать***

Группа параметров **Принтер** содержит имя, состояние, тип и местонахождение принтера, а также кнопку вызова диалоговой панели настройки параметров принтера **Свойства**.
Группа параметров **Печатать** позволяет установить количество страниц документа, которые будут выведены на печать.
Группа параметров **Копии** позволяет установить количество копий документа, которое требуется напечатать.

Настройка печати

Команда **Настройка печати** меню **Файл** предназначена для вызова диалоговой панели настройки параметров принтера, установленного по умолчанию в окне **Принтеры** (Printers) операционной системы.
Внешний вид диалоговой панели показан на рисунке.
Описание полей диалоговой панели настройки параметров принтера приведено в руководстве пользователя на операционную систему или в эксплуатационной документации, поставляемой с принтером.





Диалоговая панель настройки параметров принтера.

Предварительный просмотр экранной формы как в режиме *Исполнение*

Имеется возможность проверить, как экранная форма будет выглядеть в режиме *Исполнение*, не переводя GraphWorX32 в режим *Исполнение*.

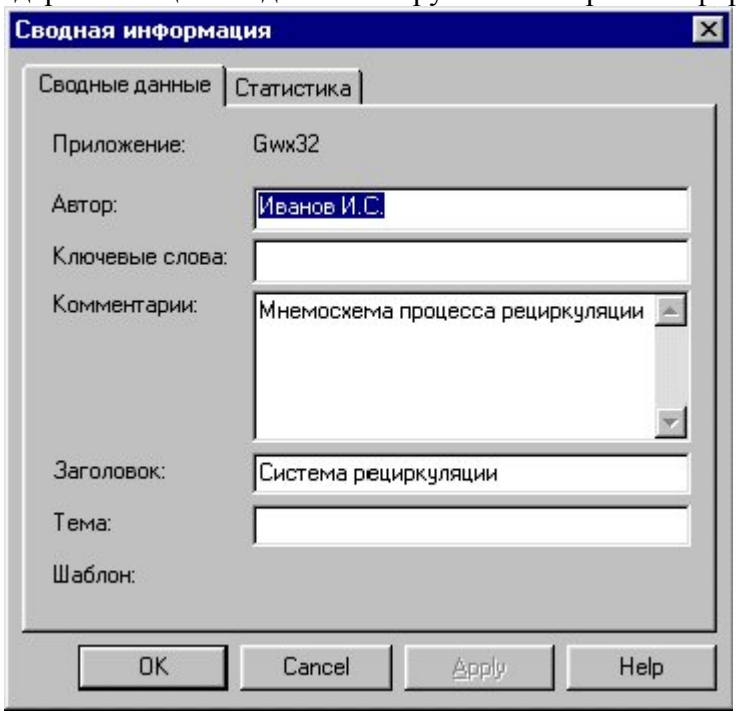
Для этого следует выбрать команду **Отобразить как в режиме Исполнение** меню **Вид**.

Сводные данные об экранной форме

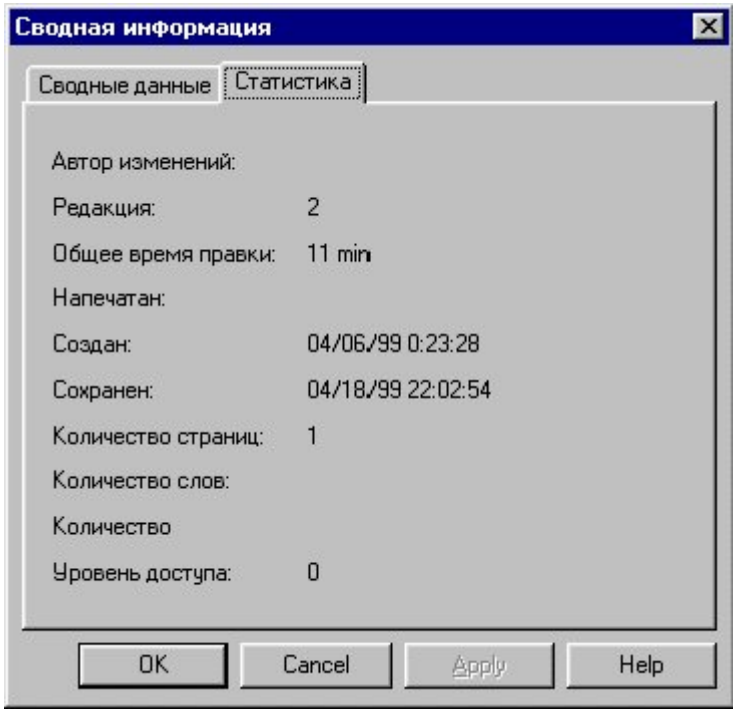
Сводная информация об открытой экранной форме может быть получена путем выбора строки **Сводные данные** меню **Вид**. На экран монитора будет выведена диалоговая панель **Сводная информация**, внешний вид которой показан на рисунке.

Страница **Сводные данные** диалоговой панели позволяет ввести фамилию автора экранной формы, ключевые слова, примененные при разработке экранной формы, краткий комментарий, заголовок и тему. Поле **Шаблон** будет содержать имя файла шаблона, который применен к экранной форме.

Страница **Статистика** диалоговой панели **Сводная информация** показана на рисунке. Данная страница содержит общие сведения о загруженной экранной форме.



*Диалоговая панель **Сводная информация***



*Страница **Статистика** диалоговой панели **Сводная информация***

Примечание. Просмотр сводной информации об экранной форме может быть осуществлен в *Проводнике* Windows. Для этого следует выбрать имя файла экранной формы, нажать правую клавишу мыши и в появившемся контекстном меню выбрать строку **Свойства** (Properties).

3.4 Завершение работы GraphWorX32

Для завершения работы GraphWorX32 следует выбрать строку **Выход** меню **Файл**. При этом на экран монитора будет выведено предложение сохранить изменения, внесенные в экранную форму. Завершение работы GraphWorX32 также осуществляется по нажатию комбинации клавиш Alt+F4.
Для запрещения ряду пользователей завершать работу GraphWorX32, находящегося в режиме Исполнение, в том числе комбинацией клавиш Alt-F4 или из *Панели задачи*, вызываемой по Alt-Ctrl-Del, следует выполнить настройку подсистемы администрирования и управления правами доступа. Для получения более подробной информации обратитесь к разделу 12 настоящего руководства.
Примечание. Если изменения не были сохранены в режиме *Разработка*, а завершение работы GraphWorX32 произведено в режиме *Исполнение*, то указанные изменения не будут сохранены.

3.5 Шаблоны.

Шаблоном является набор графических объектов с заранее установленными свойствами, который может быть применен к экранной форме в любой момент в режиме *Разработка*. Использование шаблонов наиболее удобно при создании нескольких экранных форм в одинаковом формате.
GraphWorX32 поддерживает два типа файлов: .GDF (graphic display files) – файлы графических экранных форм; .TDF (template display files) – файлы шаблонов экранных форм.
В настоящем подразделе приведено описание операций над шаблонами GraphWorX32, которые выполняются с помощью подменю **Шаблон** в меню **Формат**.

Применение шаблона

Файл шаблона может быть применен к текущей экранной форме в режиме *Разработка*, для чего следует выбрать строку **Применить шаблон** подменю **Шаблон** меню **Формат**. Применение шаблона означает, что набор графических объектов выбранного шаблона вместе со всеми их свойствами будет загружен в экранную форму. При этом ни один объект в экранной форме не будет подвергнут каким-либо изменениям. Кроме того, сводная информация об экранной форме также останется неизменной. Если ранее к экранной форме был применен другой шаблон, то при последующем применении другого шаблона произойдет замена старого шаблона новым. Объекты примененного шаблона не могут быть изменены, передвинуты или удалены до выполнения команды **Редактировать примененный шаблон** подменю **Шаблон**.

Удаление примененного шаблона

Примененный шаблон может быть удален из экранной форме в режиме *Разработка*, для чего следует выбрать строку **Удалить примененный шаблон** подменю **Шаблон** меню **Формат**.

Изменение примененного шаблона

Графические объекты шаблона и их свойства могут быть изменены, для чего следует выбрать строку **Редактировать примененный шаблон** подменю **Шаблон** меню **Формат**.

После выполнения указанной команды видимыми остаются только объекты примененного шаблона (остальные графические объекты экранной формы будут оставаться невидимыми до завершения редактирования шаблона). Любые изменения, внесенные в шаблон, вступают в силу только в текущей загруженной экранной форме. Для того, чтобы выполненные изменения в шаблоне вступили в силу во всех экранных формах, к которым применен данный шаблон, необходимо выбрать строку **Обновить экраны с данным шаблоном** в подменю **Шаблон**.

Завершение редактирования шаблона

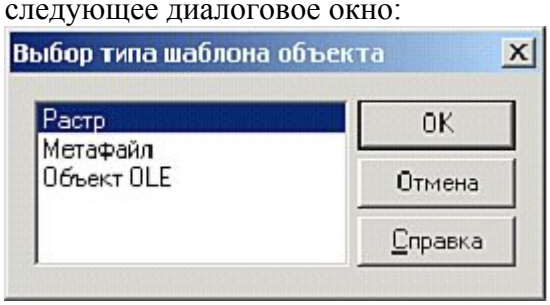
Для завершения изменения примененного шаблона следует выбрать строку **Завершить редактирование шаблона** в подменю **Шаблон** либо нажать клавишу Esc.

Обновление экранных форм, использующих один шаблон

Данная функция доступна только при редактировании файла .TDF, содержащего шаблон, который применен к нескольким экранным формам GraphWorX32. Для того, чтобы изменения в шаблоне вступили в силу во всех экранных формах, к которым применен данный шаблон, необходимо выбрать строку **Обновить экраны с данным шаблоном** в подменю **Шаблон** меню **Формат**.

Вставка шаблона объекта

При выборе пункта **Шаблон - Вставить шаблон объекта** из меню **Формат** открывается следующее диалоговое окно:



Шаблоном объекта является рамка, которая в дальнейшем может быть заменена требуемым объектом. В текущей версии пакета имеются три типа шаблонов объектов: растровое изображение (bitmap, jpeg и т.д.), метафайл (wmf) и объект OLE. После выполнения команды **Вставить шаблон объекта** подменю **Шаблон** меню **Формат**, в видимой области экранной формы появляется рамка с надписью "*Двойной щелчок Место для добавления Объект OLE*" (содержание надписи зависит от выбранного типа шаблона объекта). Двойной щелчок левой клавишей мыши на рамке выводит на экран монитора соответствующую диалоговую панель, которая позволяет заменить рамку шаблона объекта на реальный объект. Данная функция может использоваться при создании графических основ экранных форм, когда часть графики в виде растровых изображений, метафайлов или объектов OLE добавляется на завершающих этапах разработки проекта.

3.6 Слои.

Слои в экранных формах GraphWorX32 является весьма удобным средством объединения наборов графических объектов, когда объекты, входящие в данный набор, должны отображаться только при определенных условиях. В оставшееся время объекты, помещенные в определенный слой, должны быть скрыты. Подменю Слои меню Формат позволяет добавлять, удалять, дублировать, редактировать слои, а также устанавливать текущий активный слой, делать текущий активный слой следующим или предыдущим в последовательности слоев, скрывать слои, находящиеся над или под текущим активным слоем.

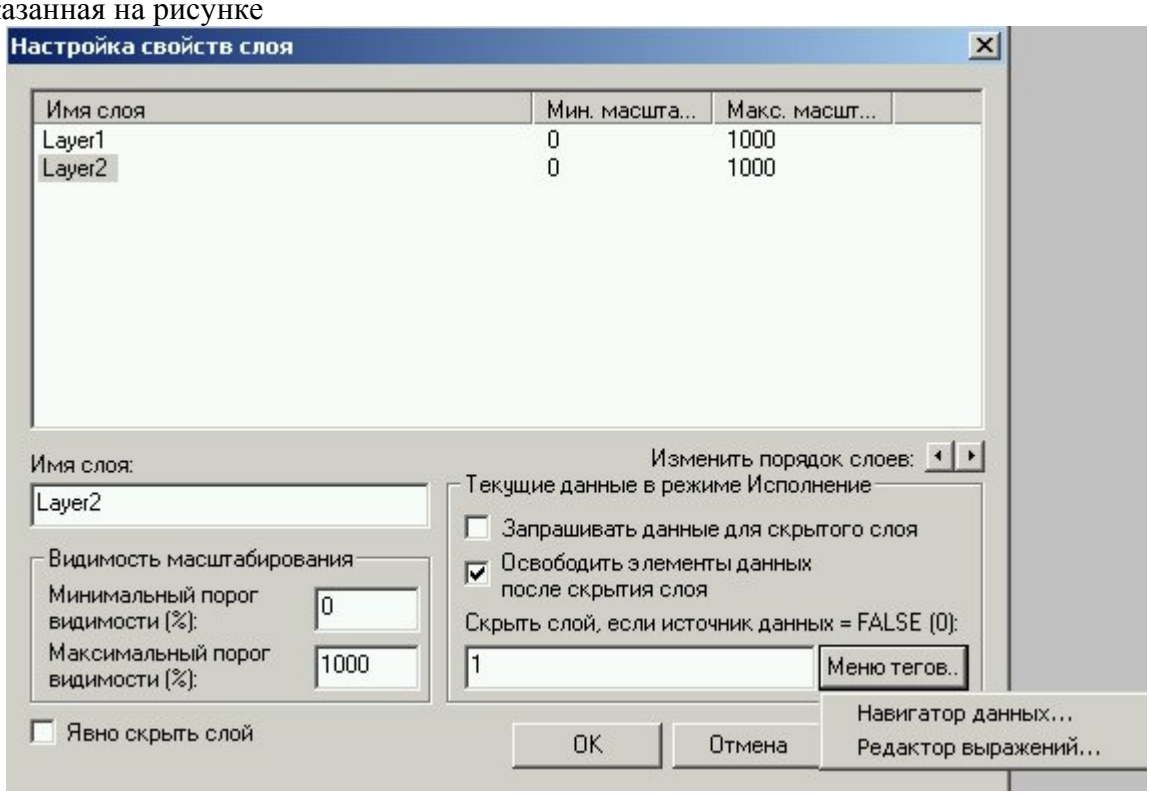
Конфигурирование слоев

Каждая экранная форма GraphWorX32 изначально содержит один слой, называемый *Первичным* или *Системным*.

Примечание. Шаблон экранной формы может быть применен только к первичному слою экранной формы.

Весьма важно понимать назначение первичного слоя. При работе с экранными формами, содержащими несколько слоев, любой объект, добавленный или нарисованный в первичном слое, будет находиться и, в случае, если они является видимым, отображаться во всех остальных слоях экранной формы. Однако объекты, помещенные в первичный слой, не могут быть изменены в других слоях экранной формы.

Для добавления еще одного слоя в экранную форму выберите команду **Добавить слой** подменю **Слои** меню **Формат**. На экран монитора будет выведена диалоговая панель, показанная на рисунке



Диалоговая панель ***Настройка свойств слоя***

В диалоговой панели **Настройка свойств слоя** возможно задать имя слоя, которое будет отображаться в крайнем правом поле строки состояния окна GraphWorX32, установить диапазон масштаба экранной формы, в котором графическая информация, находящаяся в данном слое, будет видима, а также определить правила взаимодействия с источниками данных (с элементами данных в серверах OPC), когда указанный слой скрыт. Кроме того, диалоговая панель позволяет задать условие, при выполнении которого данный слой будет скрыт.

Примечание. Для изменения параметров уже существующего слоя следует выбрать команду **Изменить свойства слоя** подменю **Слои** меню **Формат**.

Группа параметров **Видимость масштабирования**

Поля **Минимальный порог видимости** и **Максимальный порог видимости** определяют диапазон масштаба экранной формы в процентах, при котором графические объекты, находящиеся в выбранном слое, станут видимыми в режиме *Исполнение*. По умолчанию диапазон видимости всех слоев составляет от 0 до 1000 процентов. Это значит, что слой является видимым при любом масштабе экранной формы.

Для установки порогов видимости слоя:

1. Выберите имя слоя в столбце **Имя слоя**, щелкнув на имени левой клавишей мыши.
2. В поле **Минимальный порог видимости** введите минимальное значение масштаба в процентах, при котором слой должен быть видимым.
3. В поле **Максимальный порог видимости** введите максимальное значение масштаба в процентах, при котором слой должен быть видимым.
4. Если слой должен быть скрыт независимо от текущего масштаба экранной формы и условия, заданного в поле **Скрыть слой, если источник данных = FALSE (0)**, отметьте флажок **Явно скрыть слой**. В режиме *Исполнение* слой останется скрытым до тех пор, пока в режиме *Разработка* не будет снята отметка флажка **Явно скрыть слой**.

Группа параметров **Текущие данные в режиме Исполнение**

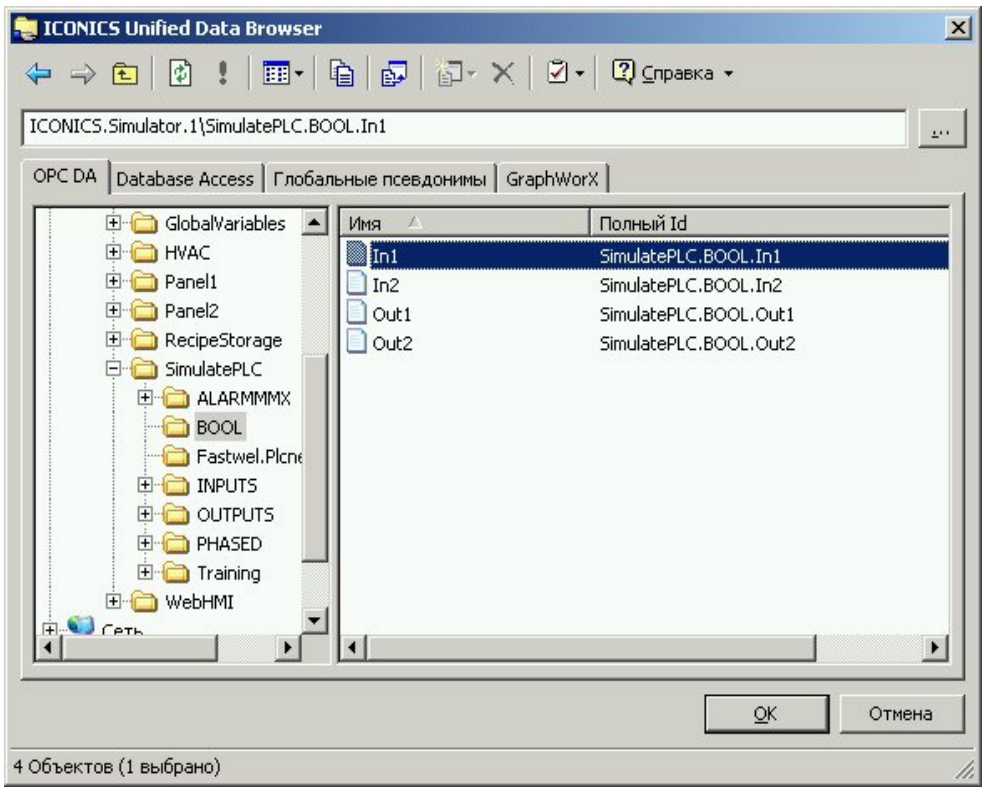
Указанная группа параметров позволяет определить способ взаимодействия графических объектов, находящихся в редактируемом слое, с элементами данных в серверах OPC, с которыми они были связаны в режиме *Разработка*.

Если слой скрыт при переходе в режим *Исполнение*, то в случае отметки флажка **Запрашивать данные для скрытого слоя** данные от связанных с находящимися в нем графическими объектами серверов OPC будут поступать в GraphWorX32 несмотря на то, что слой скрыт. Это позволяет незамедлительно получить отображение действительных данных, как только слой станет видимым. Если флажок **Запрашивать данные для скрытого слоя** не отмечен, данные от связанных с указанным слоем серверов OPC поступать не будут, пока слой не станет видимым.

Если отмечен флажок **Освободить элементы данных после скрытия слоя**, то при каждом скрытии текущего редактируемого слоя в режиме *Исполнение* будет происходить отключение от связанных с ним элементов данных в серверах OPC. Это позволяет сократить количество используемых точек ввода-вывода для невидимых слоев одной и той же экранной формы.

Поле **Скрыть слой, если источник данных = FALSE (0)** позволяет определить условие, при котором текущий редактируемый слой будет скрыт в режиме *Исполнение*. Кнопка **Меню тегов**, расположенная справа от поля **Скрыть слой, если источник данных = FALSE (0)**, предназначена для определения указанного условия.

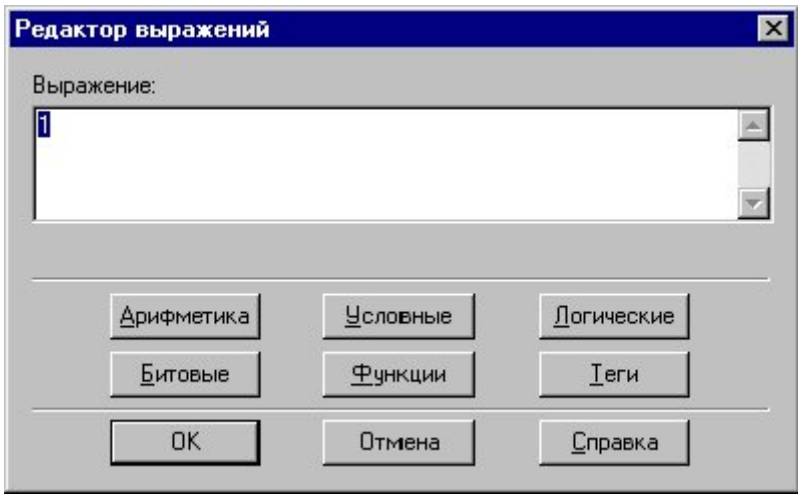
Если в меню, появляющемся при нажатии кнопки **Меню тегов**, выбрана строка **Навигатор данных**, на экран монитора будет выведено окно **Универсального навигатора данных**, показанное на рисунке.



Универсальный навигатор тегов OPC

Универсальный навигатор данных позволяет выбрать теги OPC, псевдонимы или переменные. Навигатор показывает все доступные данные вашей машины и сети. Выбрав в навигаторе нужные данные, нажмите OK, чтобы вернуться в диалоговое окно **Настройка свойств слоя.**

Если в меню выбрана строка **Редактор выражений**, на экран монитора будет выведена диалоговая панель **Редактор выражений**, показанная на рисунке.



Редактор выражений

Редактор выражений позволяет ввести простое арифметическое, функциональное, логическое или условное выражение, нулевой результат которого (или результат, равный FALSE), приведет к скрытию текущего редактируемого слоя. Для получения более подробной информации о **Редакторе выражений** обратитесь к п. 7.5 настоящего руководства.

Управление слоями во время редактирования экранных форм

Имеются следующие основные средства управления слоями во время редактирования экранных форм:

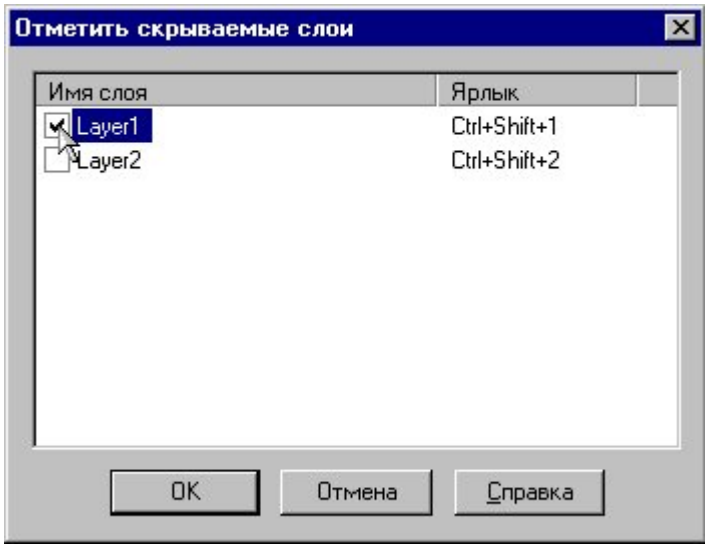
- Команда **Скрыть слои** меню **Вид** и соответствующая кнопка  в инструментальной панели *Вид*
- Команды подменю **Слои** меню **Формат**, дублированные в инструментальной панели *Слои*.

Внешний вид и назначение кнопок инструментальной панели *Слои* показаны на рисунке



Инструментальная панель Слои.

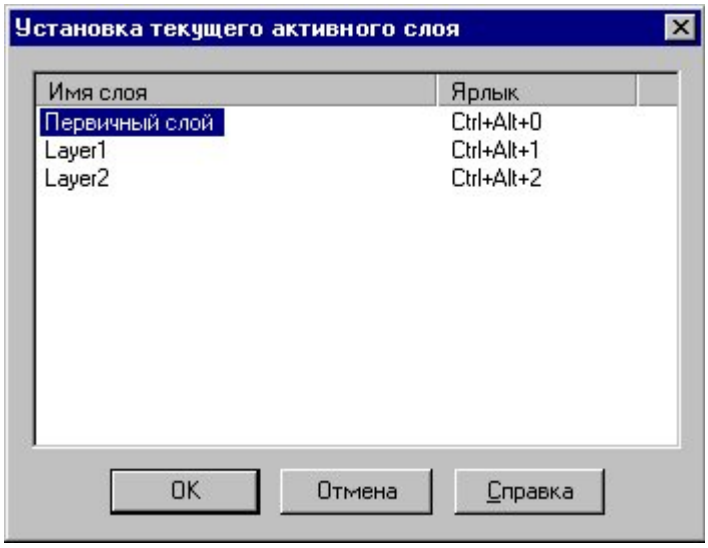
Команда **Добавить слой** позволяет создать новый слой в дополнение к первичному.
Команда **Удалить слой** позволяет открыть диалоговое окно для выбора удаляемого слоя.
Команда **Скрыть слои** меню **Вид** соответствующая кнопка в инструментальной панели *Слои*, а также комбинация клавиш Ctrl+Shift+0 позволяют включить или отключить отображения слоев, входящих в экранную форму. При выборе данной команды на экран монитора будет выведена диалоговая панель **Отметить скрываемые слои**, показанная на рисунке



Диалоговая панель Скрыть слои

Для скрытия слоев следует отметить флажки, расположенные слева от имен слоев, и нажать кнопку **OK**.
Включение и отключение отображения отдельных слоев может выполняться нажатием комбинаций клавиш Ctrl+Shift+<номер_слоя>, где <номер_слоя> -- порядковый номер (сверху вниз) слоя в списке слоев диалоговой панели **Изменить свойства слоя**.
Функциональность операций управления слоями по команде **Скрыть слои** меню **Слои** сохраняется в режиме *Исполнение*.
Примечание. Для изменения порядкового номера слоя следует выбрать имя слоя в списке диалоговой панели **Изменить свойства слоя** и нажать кнопки **Изменить порядок слоев** требуемое количество раз.
Команда **Выбрать текущий активный слой** подменю **Слои** меню **Формат** или кнопка  позволяет выбрать слой, графические объекты которого должны быть подвергнуты редактированию. После выбора данной команды на экран монитора будет выведена диалоговая панель **Установка текущего активного слоя**, показанная на рисунке.
Выберите имя слоя в списке и нажмите кнопку **OK**. Выбранный слой станет активным и доступным для редактирования.

Выбор текущего активного слоя может быть осуществлен нажатием комбинации клавиш Ctrl+Alt+<номер_слоя>, где <номер_слоя> -- порядковый номер (сверху вниз) слоя в списке слоев диалоговой панели **Настройка свойств слоя**.



*Диалоговая панель **Установка текущего активного слоя***

Команда **Сделать активный слой следующим** подменю **Слои** меню **Формат** или кнопка  позволяет сделать активным слой, следующий за текущим редактируемым в списке слоев диалоговой панели **Изменить свойства слоя**.

Команда **Сделать активный слой предыдущим** подменю **Слои** меню **Формат** или кнопка  позволяет сделать активным слой, находящийся перед текущим редактируемым в списке слоев диалоговой панели **Настройка свойств слоя**.

Настройки Системы Безопасности для слоев.

Возможно установить настройки безопасности для определенного слоя во всех экранах, для всех слоев в определенном экране, или для всех слоев во всех экранах. Для установки безопасности для слоя следует выбрать **Настройка прав доступа** из меню **Сервис**. Будет открыто диалоговое окно Сервера Безопасности. Формат для настройки безопасности для слоев: **имя файла**| **имя слоя**. При настройках безопасности можно использовать групповые символы.

Использование слоев в режиме Исполнение

Управление слоями в режиме *Исполнение* осуществляется следующими способами:

- Выбором масштаба отображения экранной формы. При этом становятся видимыми те слои, чьи диапазоны видимости при масштабировании включают в себя текущее значение масштаба.
- Нажатием комбинаций клавиш Ctrl+Shift+<номер_слоя>, <номер_слоя> -- номер слоя по порядку в списке слоев диалоговой панели **Настройка свойств слоя**.
- Вызовом диалоговой панели **Отметить скрываемые слои**. Для вызова указанной диалоговой панели следует нажать комбинацию клавиш Ctrl+Shift+0.
- Изменением условия, заданного в поле **Скрыть слой, если источник данных = FALSE (0)** диалоговой панели **Настройка свойств слоя**.

События VBA для слоев.

PreAnimateLayer(BSTR layerName)	**Отрабатывает перед запросом данных для слоя.**
PostAnimateLayer(BSTR layerName)	**Отрабатывает после запроса данных для слоя.**
PreDeanimateLayer(BSTR layerName)	**Отрабатывает перед загрузкой данных для слоя.**
PostDeanimateLayer(BSTR layerName)	**Отрабатывает после запроса данных для слоя.**

Глава 4. Создание и редактирование объектов

4.1 Введение
4.2 Режим Разработка
4.3 Инспектор свойств
4.4 Функции рисования
4.5 Функции импорта
4.6 Функции экспорта
4.7 Символы GraphWorX32
4.8 Форматирование объектов
4.9 Настройка палитры цветов
4.10 Изменение расположения объектов
4.11 Вращение графических объектов

4.1 Введение

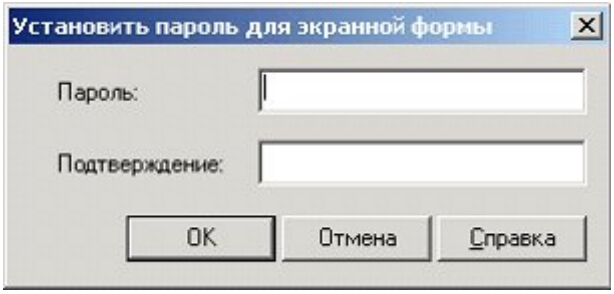
Настоящий раздел содержит указания по созданию и редактированию графических примитивов в экранных формах GraphWorX32. Рисование графических примитивов, таких как линии, круги, эллипсы, прямоугольники и т.д., выполняется при помощи панели инструментов *Рисование* или команд меню **Графика**. Имеется возможность выбора различных свойств и параметров графических примитивов, таких как стиль и толщина линий, цвета тени, заливки и фона, а также тип шрифта, которым отображается текстовая информация, для чего используется меню **Формат**.

4.2 Режим Разработка

Режим *Разработка* предназначен для создания экранных форм. В данном режиме пользователь может создавать статические и динамические графические объекты, устанавливать параметры приложения и экранных форм и т.д. Статические объекты обычно рисуются в рабочей области экранной формы, после чего их параметры (свойства) настраиваются в **Инспекторе свойств**. Динамические объекты также настраиваются при помощи **Инспектора свойств**.

Пароль на режим разработки.

В GraphWorX32 есть возможность защитить паролем вход в режим разработки экранной формы, выбрав пункт **Пароль для режима Разработки**… из меню **Формат**. При этом открывается диалоговое окно **Установить пароль текущей экранной формы**. Введите пароль в поле **Пароль** и в поле **Подтверждение**.



В дальнейшем при попытке открыть экранную форму, защищенную паролем, будет выводиться следующее диалоговое окно:



При введении правильного пароля экранная форма открывается в режиме разработки. Если пароль неверный, выдается предупреждающее сообщение, и доступ в разработку будет запрещен до тех пор, пока не будет введен правильный пароль. Этот пароль не применяется при открытии экранной формы в режиме исполнения.

Изменение размеров с помощью клавиатуры.

В режиме разработки GraphWorX32 есть возможность изменять размер и форму объектов при помощи клавиш со стрелками совместно с клавишами **SHIFT** или **CTRL+SHIFT.** Удерживая нажатой клавишу **SHIFT**,с помощью стрелок настройте верх и правую стороны объекта. Удерживая нажатой комбинацию клавиш **CTRL+SHIFT**,с помощью стрелок также настройте низ и левую стороны объекта.

Выделение объектов с помощью клавиши Tab.

С помощью клавиши Tab можно перемещать выделение с одного объекта на другой. Выделение перемещается по экрану в том порядке, в каком создавались объекты, или, если объекты были расположены в определенном порядке, например, перемещались вперед или назад, выделение перемещается от наиболее дальнего объекта к наиболее ближнему.

Выделении группы с помощью клавиши Alt.

Клавиша Alt служит для выделения группы объектов. Удерживая ее нажатой, нужно обвести контурным прямоугольником область, объекты которой необходимо выделить. Объект будет выделен, если выделяющий прямоугольник охватит все его границы.

Перемещение при помощи клавиш CTRL+SHIFT.

Если перемещать объект, удерживая нажатой комбинацию клавиш CTRL+SHIFT, то он не только копируется, но и смещается относительно исходного только в горизонтальном или вертикальном направлениях. Это удобно применять при создании «линии» объектов.

Функции поиска и замены.

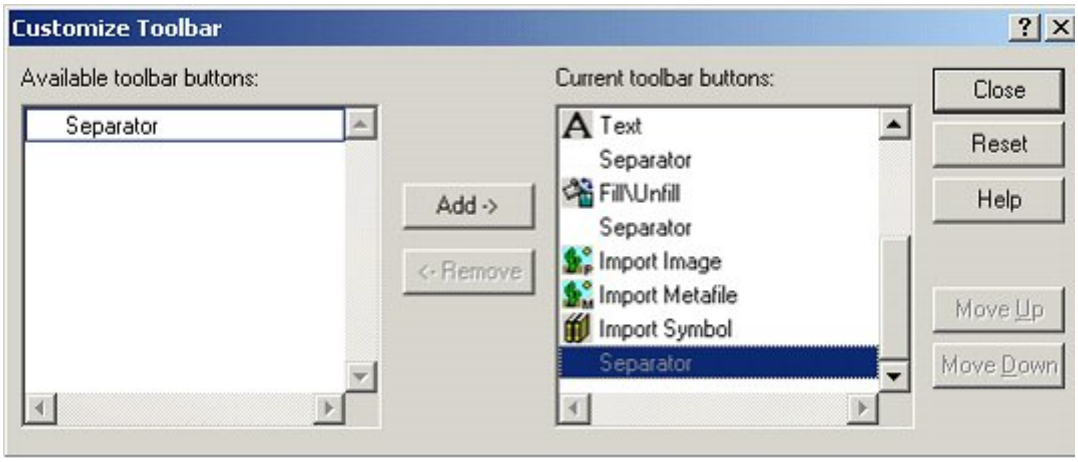
Функции поиска и замены в GraphWorX32 поддерживают поиск по Специальным данным, Описанию и Имени файла (для кнопок).

Применение шаблонов.

Когда шаблон изменен и сохранен, система автоматически выдает сообщение «Желаете ли Вы, чтобы все экранные формы, использующие этот шаблон, были обновлены?». Это позволяет проследить за тем, каким образом изменение шаблона может повлиять на другие экранные формы.

Настройка панелей инструментов.

Для настройки панелей инструментов следует щелкнуть правой клавишей на панели, которую нужно редактировать. Будет открыто диалоговое окно:

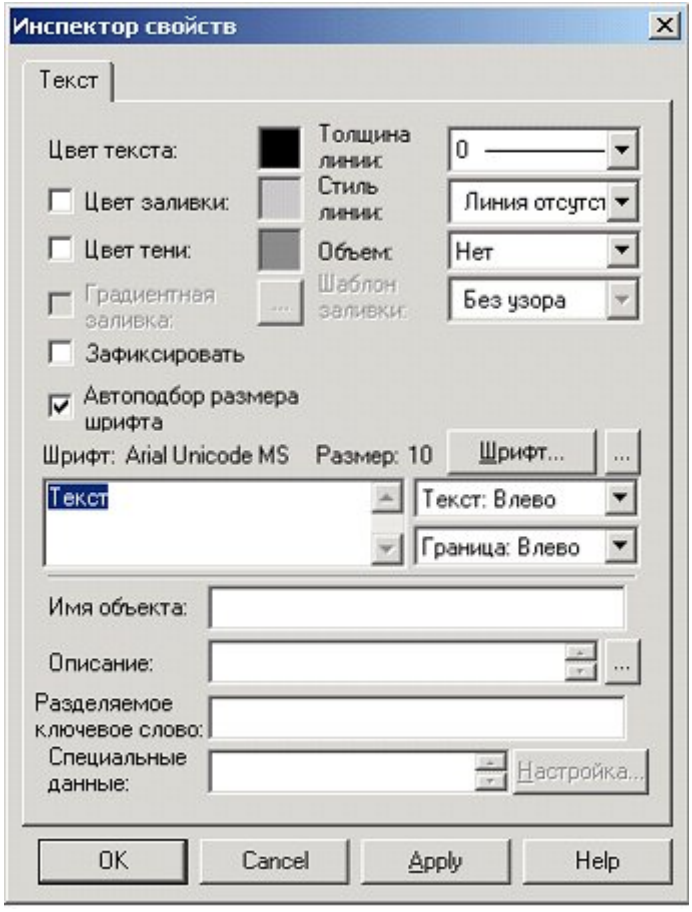


Кнопки **Добавить** и **Удалить** перемещают инструменты из списка доступных на панель инструментов и обратно.

4.3 Инспектор свойств.

Двойной щелчок левой клавишей мыши на выделенном графическом объекте приводит к появлению **Инспектора свойств**.

Инспектор свойств является диалоговой панелью, которая содержит набор страниц свойств объекта. Основная страница представляет свойства выбранного статического объекта (на рисунке показана страница свойств графического примитива *Текст*).

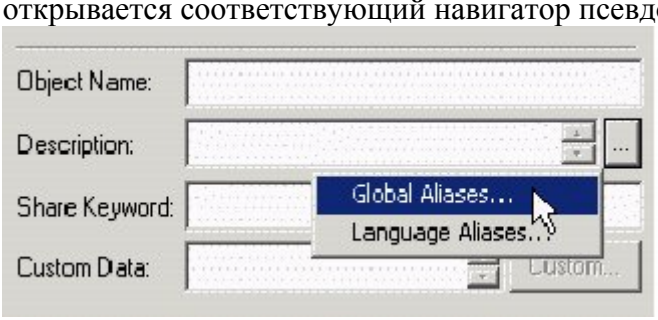


*Диалоговая панель **Инспектор свойств***

Инспектор свойств также может содержать дополнительные страницы, представляющие параметры динамических действий, примененных к статическому объекту (к объекту *Текст*, **Инспектор свойств** которого показан на рис. 4-1, применены действия

Скрыть/Блокировать и *Указание и щелчок*). Данные страницы свойств открываются путем щелчка левой клавишей мыши на соответствующих закладках диалоговой панели. В нижней части диалоговой панели **Инспектора свойств** расположены четыре кнопки: **OK**, **Отмена** (Cancel), **Применить** (Apply) и **Справка** (Help). Нажатие кнопки **OK** приводит к закрытию диалоговой панели с одновременным вступлением в силу всех изменений, внесенных в параметры объекта. Кнопка **Отмена** (Cancel) закрывает диалоговую панель с отменой всех внесенных изменений. Кнопка **Применить** (Apply) служит для вступления в силу внесенных изменений без закрытия диалоговой панели (последующее нажатие кнопки **Отмена** не приведет к отмене внесенных изменений). Кнопка **Справка** (Help) предназначена для вызова раздела справочной системы, связанного с текущей страницей **Инспектора свойств**.

В поле **Описание** можно ввести псевдоним или нажать кнопку **…** и выбрать из выпадающего списка **Глобальные псевдонимы** или **Псевдонимы языков**. При этом открывается соответствующий навигатор псевдонимов.



Общие параметры основной страницы Инспектора свойств

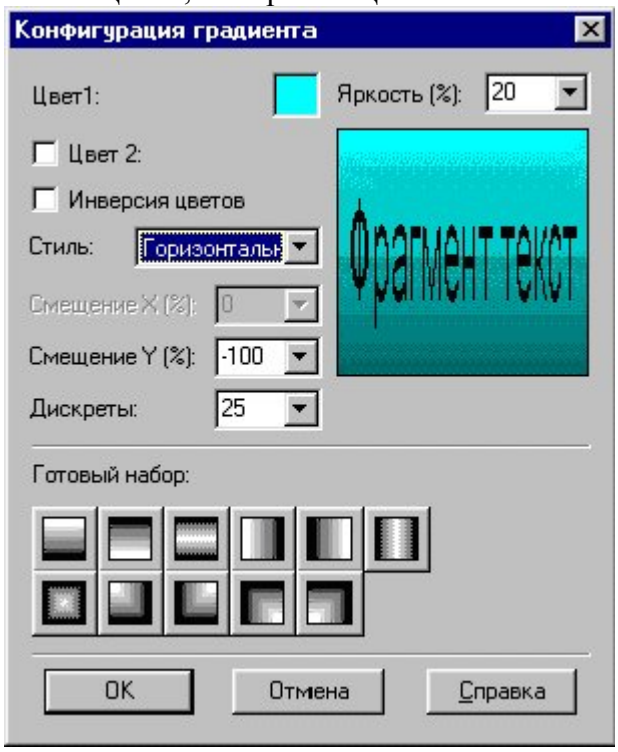
Таблица 4-1

Параметр	Описание
Цвет линии	Указывает цвет линии для объекта. Для изменения цвета линии щелкните левой клавишей мыши на прямоугольнике справа от надписи **Цвет линии**, который отражает текущий установленный цвет, и в появившейся палитре цветов_ выберите требуемый.
Цвет заливки	Указывает цвет заливки (заполнения) объекта. Для изменения цвета заливки щелкните левой клавишей мыши на прямоугольнике справа от надписи **Цвет заливки**, который отражает текущий установленный цвет, и в появившейся палитре цветов выберите требуемый.
Цвет тени	Указывает цвет тени объекта. Для изменения цвета тени щелкните левой клавишей мыши на прямоугольнике справа от надписи **Цвет тени**, который отражает текущий установленный цвет, и в появившейся палитре цветов выберите требуемый.
Градиентная заливка	Данный флажок становится доступным только после отметки флажка **Цвет заливки**. После отметки флажка **Градиентная заливка** становится доступной кнопка …, предназначенная для вызова диалоговой панели настройки параметров градиентной заливки графического объекта. Более подробная информация о градиентной заливке приведена в п. 4.3.2.
Зафиксировать	Если данный флажок установлен, то перемещение и изменение размеров зафиксированного объекта становятся невозможными.
Толщина линии	Указывает текущую установленную толщину линии. Выбор другой толщины линии выполняется в выпадающем списке, расположенном справа от надписи **Толщина линии**.
Стиль линии	Указывает стиль отображения линии. Выбор другого стиля отображения выполняется в выпадающем списке, расположенном справа от надписи **Стиль линии**.
Объем	Указывает и позволяет выбрать 3-мерные эффекты для отображения рамки объекта.
Шаблон заливки	Позволяет выбрать повторяющийся узор для установленного цвета заливки.

Примечание. Состав полей основной страницы **Инспектора свойств** может незначительно отличаться от приведенного на рисунке в зависимости от типа графического объекта.

Градиентная заливка графических примитивов и фона экранной формы

Градиентная заливка может быть применена для фона экранной формы, прямоугольников, эллипсов, сегментов и хорд, линий и текстовых фрагментов, включая элементы отображения даты и времени, параметров процесса, а также для кнопок.
Для вызова диалоговой панели настройки градиентной заливки отметьте флажок **Градиентная заливка** на основной странице свойств графического объекта в диалоговой панели **Инспектора свойств** и нажмите кнопку ... . Внешний вид диалоговой панели **Конфигурация градиента** показан на рисунке.
Диалоговая панель содержит поле предварительного просмотра заливки графического объекта, чьи параметры в настоящий момент настраиваются в **Инспекторе свойств**.
Поле **Цвет1** позволяет установить основной цвет заливки графического объекта.
Выпадающий список **Яркость (%)** доступен только в том случае, если не отмечен флажок **Цвет 2**. Данный список позволяет выбрать или ввести с клавиатуры яркость воображаемого источника света выбранного цвета, освещающего настраиваемый графический объект.
После отметки флажка **Цвет 2**, справа от него появится поле квадратной формы, позволяющее установить второй цвет для градиентной заливки. При этом заливка объекта будет выглядеть таким образом, будто бы объект освещается двумя источниками света, один из которых задан полем **Цвет1**, а второй -- **Цвет 2**.



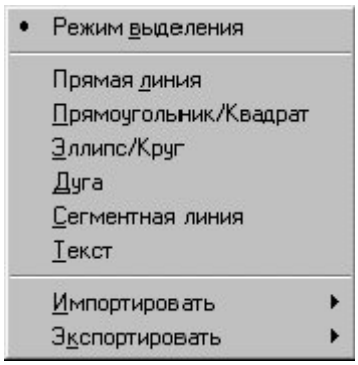
*Диалоговая панель **Конфигурация градиента***

Флажок **Инверсия цветов** позволяет поменять местами воображаемые источники света.
Выпадающий список **Стиль** позволяет выбрать стиль градиента: *Горизонтально*, *Вертикально* или *Квадрат*. Если выбран стиль *Горизонтально*, то воображаемый источника света будет освещать объект от вершины до основания. Если выбран стиль *Вертикально*, то воображаемый источник света будет находиться под углом к объекту. Если выбран стиль *Квадрат*, то воображаемый источник света будет освещать объект с левой или правой стороны.

Выпадающие списки **Смещение X** и **Смещение Y** позволяют сдвигать воображаемый источник света горизонтально и вертикально.
Выпадающий список **Дискреты** предназначен для выбора или ввода с клавиатуры значения, определяющего плавность перехода фрагментов градиентной заливки.
В нижней части диалоговой панели находится группа кнопок **Готовый набор**. Нажатием кнопки можно выбрать один из предопределенных способов градиентной заливки объекта без кропотливого подбора.
Примечание. Когда графическому примитиву, для которого настроена градиентная заливка, назначается динамическое действие *Вращение*, местоположение градаций заливки не будет успевать отображаться синхронно с новым положением объекта.

4. 4 Функции рисования

Функции рисования позволяют создавать графические объекты с помощью инструментов рисования. Имеется возможность создания сложных графических объектов путем группировки (объединения) более простых. Функции рисования находятся в меню **Графика** и в панели инструментов *Рисование*, внешний вид которых показан на рисунках.
Панель инструментов *Рисование* как правило расположена вертикально в левой области рамки окна GraphWorX32.



*Меню **Графика***



Панель инструментов Рисование

Инструмент выделения графических объектов

Данный инструмент после выбора позволяет выделять графические объекты для последующего изменения их внешнего вида, положения и других свойств.
Для выделения одного объекта:

1. Нажмите кнопку режима выделения графических объектов в панели инструментов *Рисование* или отметьте строку **Режим выделения** меню **Графика**. Курсор мыши примет форму стрелки.
2. Поместите курсор мыши в область отображения объекта, который необходимо выделить, и щелкните левой клавишей мыши. Объект будет выделен небольшими символами квадратной формы, расположенными вдоль воображаемого прямоугольника, охватывающего объект.

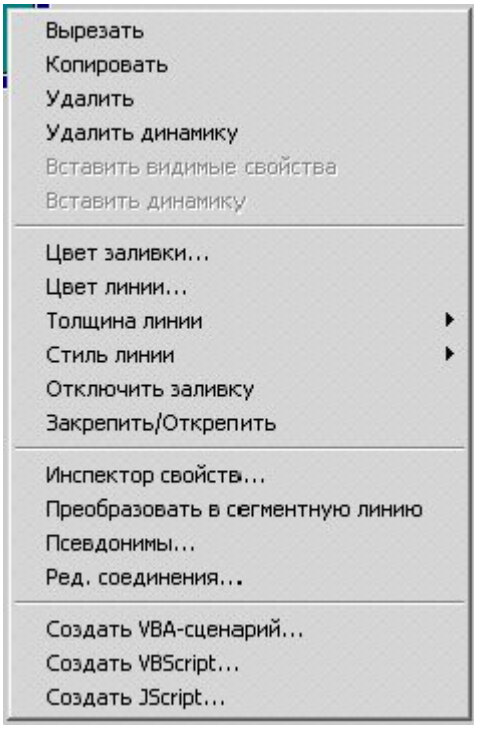
Для выделения нескольких объектов:

1. Нажмите кнопку режима выделения графических объектов в панели инструментов *Рисование* или отметьте строку **Режим выделения** меню **Графика**. Курсор мыши примет форму стрелки.
2. Поместите курсор мыши вблизи области отображения нескольких графических объектов, которые требуется выделить, и, удерживая левую клавишу мыши в нажатом состоянии, переместите курсор по диагонали воображаемого прямоугольника, охватывающего выделяемые объекты. Указанный прямоугольник в процессе перемещения курсора будет отображаться тонкими прерывистыми линиями.
3. Отпустите левую клавишу мыши. Все графические объекты, которые попали внутрь прямоугольной области выделения, будут выделены небольшими символами квадратной формы, расположенными вдоль воображаемых прямоугольников, охватывающих каждый объект.

Операция поочередного выделения нескольких объектов может осуществляться в соответствии с указаниями по выделению одного объекта, но с одновременным удержанием клавиши SHIFT в нажатом состоянии.

Примечание. По умолчанию всегда устанавливается режим выделения.

Щелчок правой клавишей мыши на выделенном объекте приводит к появлению специальной версии контекстного меню **Формат**, внешний вид которого показан на рисунке



*Контекстное меню **Формат***

Инструмент *Линия*

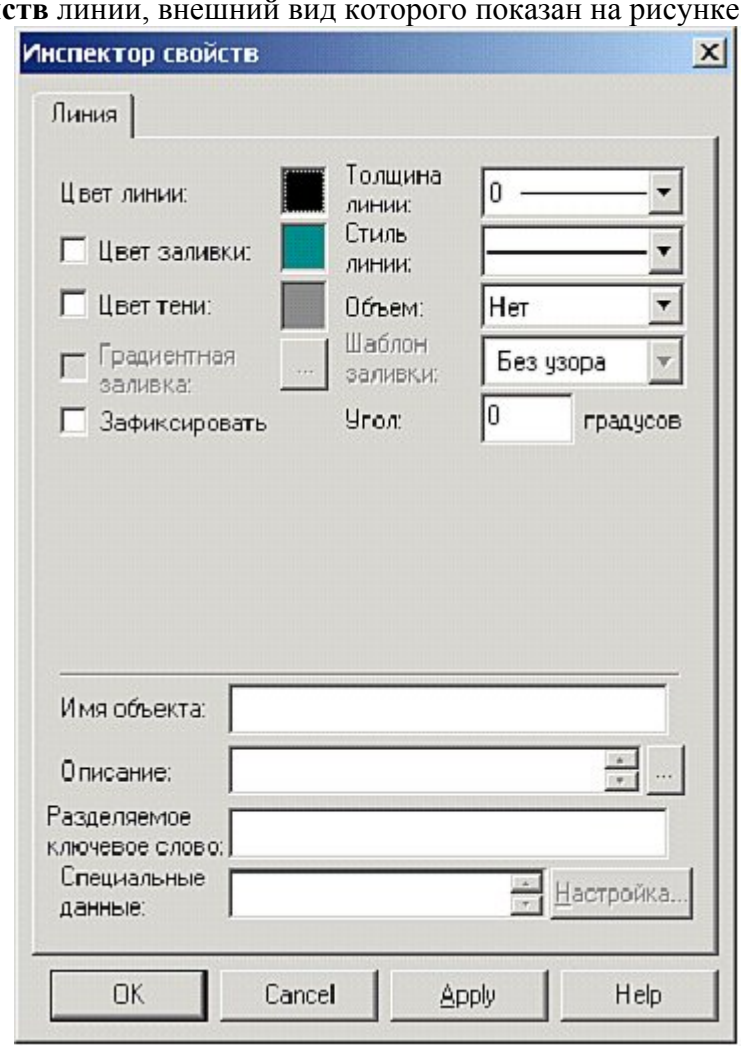
Данный инструмент предназначен для рисования прямых линий в рабочей области экранной формы. Имеется возможность изменения толщины линии и ее внешнего вида путем выбора соответствующих строк подменю **Толщина линии** и **Стиль линии** меню

Формат, а также с помощью соответствующих параметров основной страницы диалоговой панели **Инспектора свойств**.

Для рисования линии:

1. Выберите строку **Прямая линия** меню **Графика** или инструмент *Линия* панели инструментов *Рисование*. Курсор мыши примет вид карандаша.
2. Нажмите левую клавишу мыши в точке, которая будет являться началом линии, и, удерживая левую клавишу мыши в нажатом состоянии, переместите курсор в точку, в которой должна оканчиваться линия, после чего отпустите левую клавишу мыши. Для рисования линий под углом, кратным 45-ти градусам, выполняйте указанные действия при одновременном удержании клавиши SHIFT в нажатом состоянии.

Двойной щелчок левой клавишей мыши на изображении линии приведет к появлению **Инспектора свойств** линии, внешний вид которого показан на рисунке



***Инспектор свойств** объекта Линия*

Для изменения длины линии:

1. Выделите линию.
2. Нажмите левую клавишу мыши, поместив курсор на изображение небольшого символа квадратной формы, расположенного на одном из концов линии, и, удерживая левую клавишу мыши в нажатом состоянии, переместите курсор мыши в направлении увеличения или уменьшения длины линии, после чего отпустите левую клавишу мыши.

Для изменения положения линии:

1. Щелкните левой клавишей мыши на изображении линии.
2. Удерживая левую клавишу мыши в нажатом состоянии, переместите курсор в новую позицию рабочей области экранной формы.
3. Отпустите левую клавишу мыши.

Для добавления угловых изгибов к прямой линии:

1. Щелкните левой клавишей мыши на изображении линии.
2. Удерживая клавишу SHIFT в нажатом состоянии, поместите курсор в точку на линии, в которой необходимо добавить угловой изгиб. Курсор мыши примет форму креста.
3. Удерживая левую клавишу мыши в нажатом состоянии и не отпуская клавишу SHIFT, переместите курсор.
4. Отпустите левую клавишу мыши.

Для удаления угловых изгибов прямой линии:

1. Щелкните левой клавишей мыши на изображении линии.
2. Удерживая клавишу SHIFT в нажатом состоянии, поместите курсор на угловой изгиб линии. Курсор мыши примет форму креста.
3. Не отпуская клавишу SHIFT, произведите щелчок правой клавишей мыши. Угловой изгиб будет удален.

Щелчок правой клавишей мыши на изображении линии приведет к появлению контекстного меню **Формат**.

Инструмент *Сегментная линия*

Данный инструмент предназначен для рисования линий произвольной конфигурации, а также замкнутых полигонов, что обеспечивает возможность создания графических изображений треугольников и других многоугольных геометрических фигур.
Для рисования линии произвольной (свободной) конфигурации выберите инструмент *Сегментная линия* или соответствующую строку меню **Графика**, после чего перемещайте курсор мыши, удерживая левую клавишу мыши в нажатом состоянии.
Для рисования многоугольника:

1. Выберите инструмент *Сегментная линия* или соответствующую строку меню **Графика**, после чего щелкните левой клавишей мыши в первой вершине многоугольника.
2. Переместите курсор мыши, предварительно отпустив левую клавишу мыши, к точке, в которой должна располагаться вторая вершина многоугольника, и щелкните левой клавишей мыши.
3. Повторяйте действия п. 2. При завершении рисования многоугольника следует щелкнуть правой клавишей мыши.

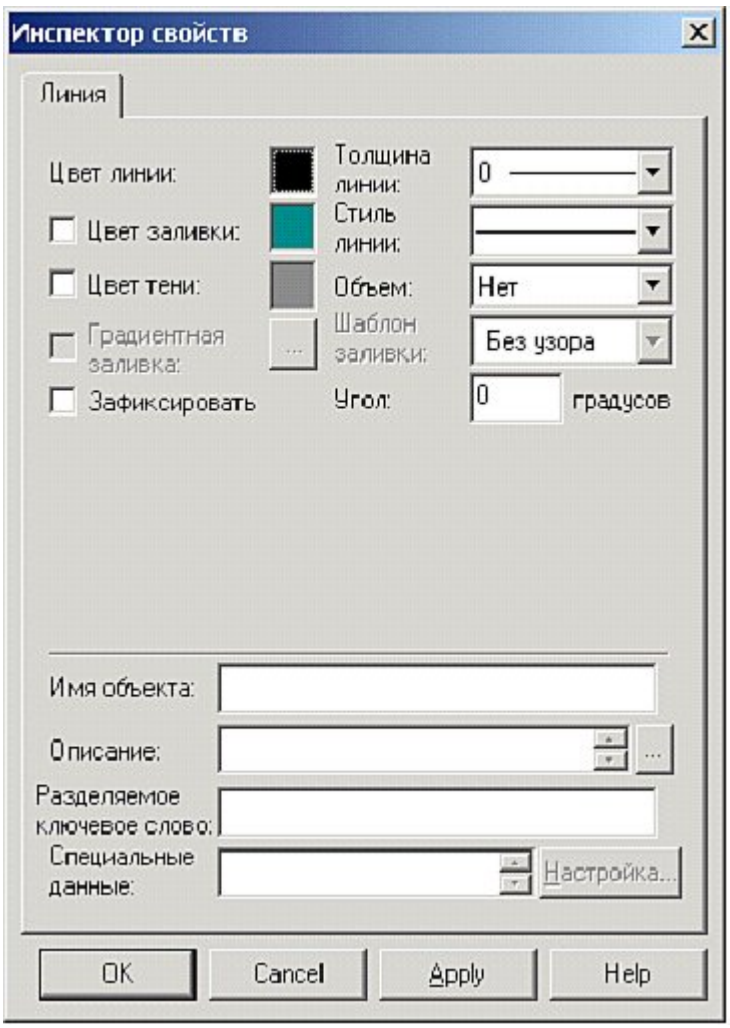
Для нарисованного многоугольника помимо основных свойств, совпадающих со свойствами объекта *Прямая линия*, могут быть использованы такие свойства, как заливка и др.

Инструмент *Прямоугольник/Квадрат*

Данный инструмент предназначен для рисования прямоугольников с заполнением и без заполнения (с заливкой и без заливки) в рабочей области экранной формы. Цвет нарисованного прямоугольника будет совпадать с выбранным в **Палитре цветов**.
Для рисования прямоугольника:

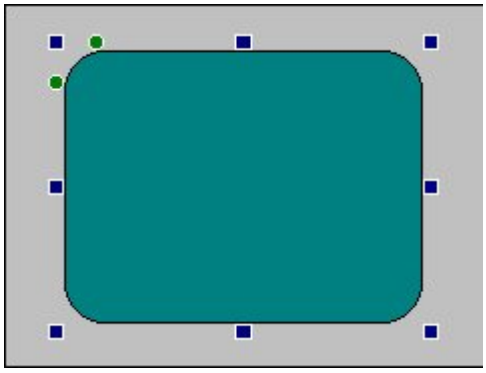
1. Выберите строку **Прямоугольник/Квадрат** в меню **Графика** или соответствующий инструмент в панели инструментов *Рисование*. Курсор мыши примет вид символа квадратной формы, на который наложен крест.
2. Щелкните левой клавишей мыши в рабочей области экранной формы и, удерживая левую клавишу мыши в нажатом состоянии, переместите курсор, после чего отпустите левую клавишу мыши. Для рисования квадрата в процессе перемещения курсора следует удерживать в нажатом состоянии клавишу SHIFT.

Для просмотра и редактирования свойств прямоугольника следует поместить курсор на его изображение и выполнить двойной щелчок левой клавиши мыши. На экран монитора будет выведена диалоговая панель свойств, внешний вид которой показан на рисунке.



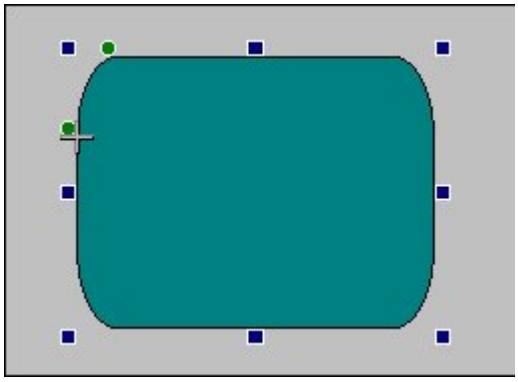
***Инспектор свойств** объекта Прямоугольник/Квадрат*

Кроме параметров **Инспектора свойств**, характерных для всех графических объектов, на странице свойств объекта *Прямоугольник* имеется флажок **Скругленные углы**, установка которого позволяет установить для объекта скругленные углы вместо прямых. После отметки данного флажка прямоугольник примет вид, показанный на рисунке.



Прямоугольник, имеющий скругленные углы

Для изменения радиуса скругления поместите курсор на один из символов круглой формы красного цвета, расположенный вблизи левого верхнего угла прямоугольника. Курсор мыши примет вид, показанный на рисунке. Удерживая левую клавишу в нажатом состоянии, перемещайте курсор. Эта операция приведет к изменению радиуса скругления углов прямоугольника.



Изменение радиуса скругления углов прямоугольника

Поле ввода **Угол** диалоговой панели **Инспектора свойств** прямоугольника предназначено для поворота объекта на заданный угол.

Таблица 4-2

Назначение других параметров Инспектора свойств

Параметр	Описание
Имя объекта	Предназначено для идентификации объекта при программном доступе к его свойствам через OLE Automation (из сценарных процедур VBA или из внешних приложений)
Описание	Может содержать краткое описание объекта, которое после ввода будет выводиться в виде подсказки (Tooltip)
Разделяемое ключевое слово	Предназначено для ввода идентификатора, который используется при выполнении команды **Обновить объекты**. Более подробная информация приведена в п. 5.6
Специальные данные	Используется для ввода произвольных данных пользователя. Для получения более подробной информации обратитесь к разделу 17 настоящего руководства

Щелчок правой клавишей мыши на объекте *Прямоугольник* приводит к появлению контекстного меню **Формат**, команды которого позволяют выполнять различные операции редактирования объекта, описание которых приведено выше.

Примечание. После выбора инструмента *Прямоугольник/Квадрат* курсор будет сохранять вид символа квадратной формы, на который наложен крест, до тех пор, пока пользователь не произведет двойной щелчок левой клавишей мыши, однократный щелчок правой клавишей мыши или не нажмет клавишу Esc.

Для изменения размеров объекта:

1. Выделите прямоугольник, щелкните левой клавишей мыши на одном из небольших символов квадратной формы, расположенных вдоль контура прямоугольника, и, удерживая левую клавишу мыши в нажатом состоянии, перемещайте курсор.
2. Для сохранения пропорций прямоугольника при изменении его размеров следует удерживать клавишу SHIFT в нажатом состоянии.

Применение и отмена заливки

Для включения или отключения заливки объекта воспользуйтесь **Палитрой цветов** или полем **Цвет заливки Инспектора свойств**.

Использование градиентной заливки

Для применения к объекту градиентной заливки отметьте флажок **Градиентная заливка Инспектора свойств** и нажмите кнопку "**...**", расположенную справа от флажка **Градиентная заливка**. Выполните настройку градиентной заливки или выберите один из вариантов градиентной заливки из готового набора в диалоговой панели **Настройка градиента**.

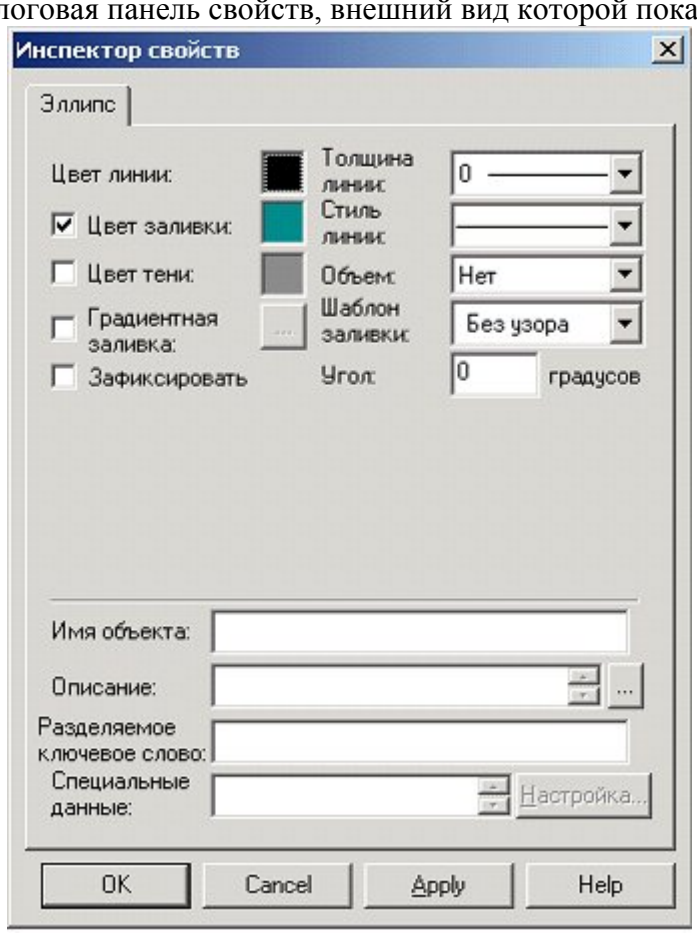
Инструмент *Эллипс/Круг*

Данный инструмент предназначен для рисования соответствующих графических объектов в рабочей области экранной формы.

Для рисования эллипса

1. Выберите строку **Эллипс/Круг** меню **Графика** или соответствующий инструмент в панели инструментов *Рисование*. Курсор мыши примет вид символа круглой формы, на который наложен крест.
2. Щелкните левой клавишей мыши в рабочей области экранной формы и, удерживая левую клавишу мыши в нажатом состоянии, переместите курсор, после чего отпустите левую клавишу мыши. Для рисования круга в процессе перемещения курсора следует удерживать в нажатом состоянии клавишу SHIFT.

Для просмотра и редактирования свойств эллипса следует поместить курсор на его изображение и выполнить двойной щелчок левой клавишей мыши. На экран монитора будет выведена диалоговая панель свойств, внешний вид которой показан на рисунке.



***Инспектор свойств** объекта Эллипс/Круг*

Для изменения размеров объекта

1. Выделите эллипс, щелкните левой клавишей мыши на одном из небольших символов квадратной формы, расположенных вдоль контура прямоугольника, в который вписан эллипс, и, удерживая левую клавишу мыши в нажатом состоянии, перемещайте курсор.
2. Для сохранения пропорций эллипса при изменении его размеров следует удерживать клавишу SHIFT в нажатом состоянии.

Применение и отмена заливки

Для включения или отключения заливки объекта воспользуйтесь **Палитрой цветов** или полем **Цвет заливки Инспектора свойств**.

Использование градиентной заливки

Для применения к объекту градиентной заливки отметьте флажок **Градиентная заливка Инспектора свойств** и нажмите кнопку "**...**", расположенную справа от флажка **Градиентная заливка**. Выполните настройку градиентной заливки или выберите один из вариантов градиентной заливки из готового набора в диалоговой панели **Настройка градиента**.

Инструмент *Текст*

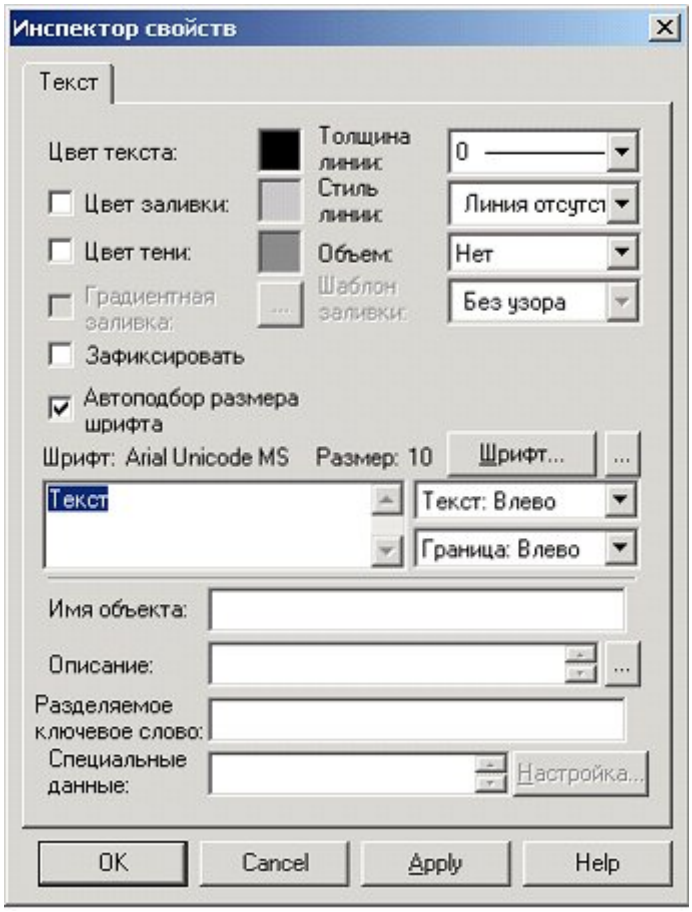
Данный инструмент предназначен для ввода фрагментов текста в рабочей области экранной формы.

Для вставки фрагмента текста

1. Выберите строку **Текст** меню **Графика** или соответствующий инструмент в панели инструментов *Рисование*.
2. Щелкните левой клавишей мыши в точке рабочей области, в которой требуется ввести фрагмент текста.
3. С помощью клавиатуры введите текст, который будет распространяться влево от места начала ввода. Если требуется начать новую строку, нажмите клавишу Enter.

Щелчок правой клавишей мыши на объекте *Текст* приводит к появлению контекстного меню **Формат**, команды которого позволяют выполнять различные операции редактирования объекта, описание которых приведено выше. Щелчок правой клавишей мыши с одновременным удержанием клавиши SHIFT в нажатом состоянии позволяет редактировать текстовый фрагмент непосредственно.

Для просмотра и редактирования свойств текстового фрагмента следует поместить курсор на его изображение и выполнить двойной щелчок левой клавиши мыши. На экран монитора будет выведена диалоговая панель свойств, внешний вид которой показан на рисунке



***Инспектор свойств** объекта Текст*

Описание свойств объекта *Текст* приведено в таблице
Параметры Инспектора свойств объекта Текст

Параметр	Описание
Цвет текста	Определяет цвет текстового фрагмента
Цвет заливки	Если флажок **Цвет заливки отмечен**, данный параметр позволяет установить цвет заливки прямоугольной области, в которую вписан текстовый фрагмент.
Градиентная заливка	Данный флажок становится доступным только после отметки флажка **Цвет заливки**. После отметки флажка **Градиентная заливка** становится доступной кнопка  , предназначенная для вызова диалоговой панели настройки параметров градиентной заливки фрагмента текста.
Цвет тени	Если флажок **Цвет тени** отмечен, данный параметр позволяет установить цвет тени каждой буквы текстового фрагмента.
Зафиксировать	Установка данного флажка позволяет зафиксировать положение текстового фрагмента в рабочей области экранной формы.
Шаблон заливки	Определяет узор заливки
Область ввода текста	Данная область позволяет редактировать текст. Текст, введенный в данную область **Инспектора свойств**, будет отображен внутри прямоугольника, в который вписан текстовый фрагмент.
Шрифт	Данная кнопка позволяет выбрать тип и набор шрифта для текстового фрагмента.
Размер	Данное поле содержит размер шрифта для текстового фрагмента.
Выравнивание границ	Выпадающий список позволяет установить стиль выравнивания границ текстового фрагмента.
Выравнивание текста	Выпадающий список позволяет установить стиль выравнивания текстового фрагмента внутри прямоугольника, в который вписан текстовый фрагмент.
Автоподбор размера шрифта	При установке данного флажка изменение размеров прямоугольника, в который вписан текст, будет приводить к автоматическому изменению размера шрифта.

На рисунках приведены примеры различных способов форматирования текстового фрагмента.

Резервуар 1

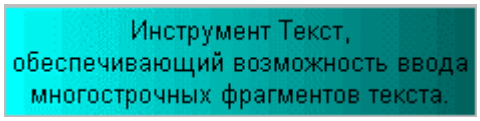
Текстовый фрагмент без атрибутов



Текстовый фрагмент с применением заливки и трехмерного эффекта



Текстовый фрагмент с применением заливки, тени и трехмерного эффекта



Текстовый фрагмент с применением градиентной заливки

Добавление атрибутов к фрагменту текста

Начиная с версии 6.0, GraphWorX32 позволяет вращать фрагменты текста.

Для поворота фрагмента текста на 90°, выберите фрагмент текста и нажмите кнопку  или  панели инструментов *Расположение*.

В поле **Описание** можно ввести псевдоним или нажать кнопку **…** и выбрать из выпадающего списка **Глобальные псевдонимы** или **Псевдонимы языков**. При этом открывается соответствующий навигатор псевдонимов.

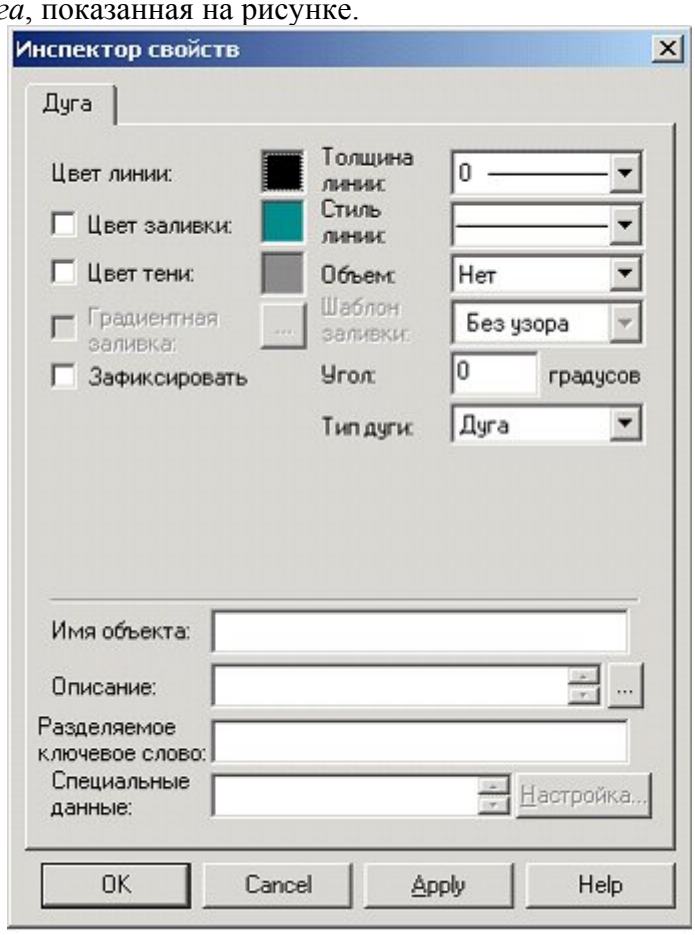
Инструмент *Дуга*

Данный инструмент предназначен для рисования дуг, секторов и хорд в рабочей области экранной формы.

Для рисования дуги

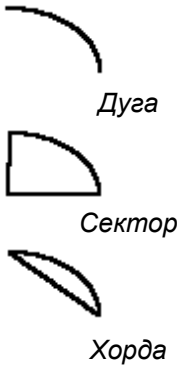
1. Выберите строку **Дуга** меню **Графика** или соответствующий инструмент в панели инструментов *Рисование*.
2. Поместите курсор в рабочую область экранной формы и нажмите левую клавишу мыши. Удерживая левую клавишу мыши в нажатом состоянии, переместите курсор, после чего отпустите левую клавишу мыши. В рабочей области экранной формы появится дуга.

Для просмотра свойства объекта *Дуга* поместите на нее курсор мыши и выполните двойной щелчок. На экран монитора будет выведена диалоговая панель **Инспектора свойств** объекта *Дуга*, показанная на рисунке.



***Инспектор свойств** объекта Дуга*

В дополнение к стандартным свойствам графических объектов, **Инспектор свойств** объекта *Дуга* позволяет установить тип *Дуги*, для чего служит выпадающий список **Тип дуги**. На рисунке показаны различные типы объекта *Дуга*.



Различные типы представления объекта Дуга

Для изменения размеров дуги

1. Выделите дугу, щелкните левой клавишей мыши на одном из небольших символов квадратной формы, расположенных вдоль контура прямоугольника, в который вписана дуга, и, удерживая левую клавишу мыши в нажатом состоянии, перемещайте курсор.
2. Для сохранения пропорций дуги при изменении ее размеров следует удерживать клавишу SHIFT в нажатом состоянии.

Для редактирования изображения дуги

1. Выделите дугу, после чего выполните щелчок правой клавиши мыши, удерживая клавишу SHIFT в нажатом состоянии, или щелкните правой клавишей мыши, поместив курсор на изображение дуги, и в появившемся контекстном меню выберите строку **Изменить дугу**.
2. Курсор мыши примет вид креста, а на концах дуги появятся небольшие символы прямоугольной формы. Поместите курсор на один из символов, нажмите левую клавишу мыши и, удерживая ее в нажатом состоянии, перемещайте курсор для редактирования изображения дуги.

Объединение дуги с другими линейными объектами

Появилась новая возможность объединения дуги с другими линейными объектами. Эта возможность позволяет применять свойства (например, заливка) ко всему (объединенному) объекту. Чтобы объединить два объекта, выберите эти объекты, затем в меню **Расположение** нажмите функцию **Сгруппировать.**

4.5 Функции импорта

Вставка (импорт) растрового изображения

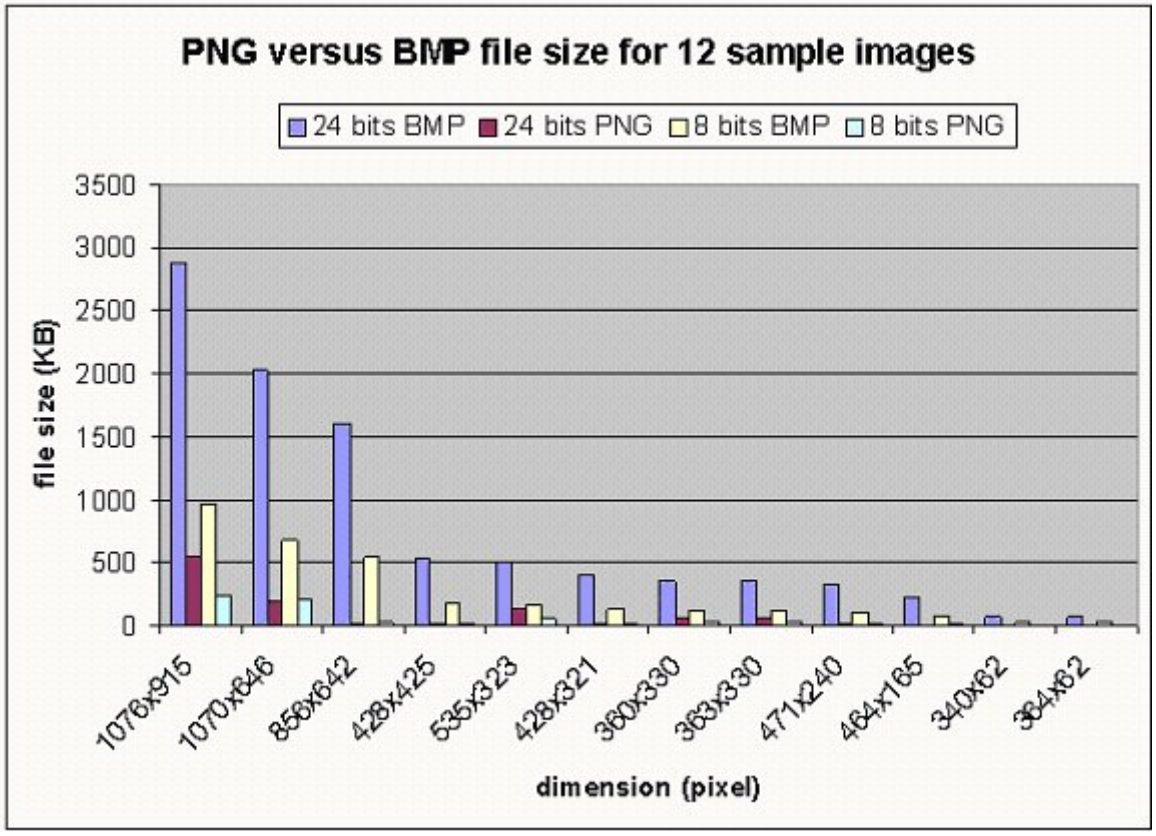
Выбор инструмента *Импортировать растр* или команда **Импортировать -- Растр** меню **Графика** позволяет вставить в рабочую область экранной формы растровое изображение в формате bmp. Имеется возможность изменения размеров и перемещения импортированных растровых изображений.

В новой версии GraphWorX32 поддерживает импорт графических файлов следующих форматов:

- **Adobe Photoshop Files (*.PSD)**
- **C64 Koala Graphics (*.KOA)**
- **CompuServe GIF (*.GIF)**
- **Dr. Halo (*.CUT)**
- **Icon (*.ICO)**
- **IFF Interleaved Bitmap (*.IFF)**
- **JPEG (*.JPG;*.JPEG)**
- **Kodak PhotoCD (*.PCD)**
- **Liner Bitmap Graphics (*.LBM)**
- **Multiple Network Graphics (*.MNG)**
- **Zsoft Paintbrush (*.PCX)**
- **Portable Network Graphics (*.PNG)**
- **Portable Network Media (*.PBM;*.PGM;*.PPM)**
- **Sun Raster Images (*.RAS)**
- **Truevision Targa (*.TGA;*.TARGA)**
- **Tag Image File Format (*.TIF;*.TIFF)**
- **Windows or OS/2 Bitmap (*.BMP)**
- **Wireless Bitmap (*.WBMP)**

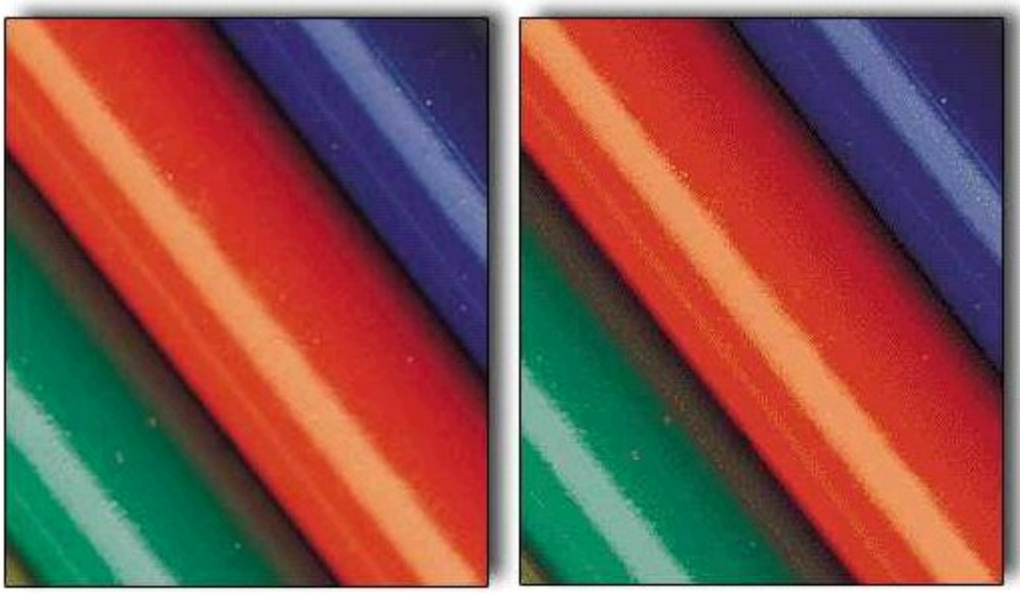
Нейронная сеть/сжатие PNG.

GraphWorX позволяет импортировать различные графические форматы. Некоторые из них применяют сжатие (например, JPEG), другие – нет (например, BMP). Файлы растров могут занимать большое количество пространства на диске. Для уменьшения размеров дискового пространства, необходимого для экранных форм GraphWorX, добавлен новый алгоритм сжатия. Этот алгоритм позволяет уменьшать необходимое пространство на диске и оптимизировать исполнение при работе с растровыми файлами или передаче экранных форм в Интернет с помощью WebHMI. Алгоритм сжатия базируется на технологии нейронных сетей и сжатии PNG. Нейронные сети помогают уменьшить количество цветов, необходимых для представления растра, и таким образом, уменьшить окончательный размер файла. Алгоритм сжатия PNG позволяет сохранить доп. место на диске. Следующая диаграмма демонстрирует различие в размере файлов между обычным растровым файлом BMP и в случае применения алгоритма сжатия PNG.



Растровые файлы на следующих рисунках показывают уменьшение цветов с помощью алгоритма нейронных сетей.

Слева расположен исходный рисунок, исп. 16 млн. цветов, справа – он же после первой обработки алгоритмом нейронных сетей для оптимизации цвета. Оба рисунка выглядят практически одинаково, и заметить разницу можно только при очень внимательном изучении.



Original Image (16 million colors) Color Reduced Image (256 colors)

Вставка (импорт) метафайла

Выбор инструмента *Импортировать метафайл* или команды **Импортировать -- Метафайл** меню **Графика** позволяет вставить в рабочую область экранной формы векторное изображение в форматах wmf и emf (расширенный метафайл). Имеется возможность изменения размеров и перемещения импортированных векторных изображений.

Примечание. Для сохранения приемлемой производительности системы в режиме *Исполнение* не рекомендуется импортировать растровые и векторные изображения с количеством цветов, превышающим 256.

4.6 Функции экспорта

Выбор команды **Экспортировать -- Метафайл** меню **Графика** позволяет сохранить рабочую область экранной формы в файле векторного изображения в следующих форматах:

- **Bitmap (*.bmp)**
- **Jpeg (*.jpg)**
- **Png (*.png)**
- **Tiff (*.tif)**

Exporting Custom Colors

Выбрав пункт **Импорт > Настр. цвета** из меню **Графика**, можно импортировать текстовый файл, содержащий параметры красного, зеленого и синего цвета для выбранного объекта на экране.

4.7 Символы GraphWorX32

Библиотека символов предназначена для хранения, предварительного просмотра и повторного использования графических объектов, созданных и объединенных в *символы GraphWorX32*. Дополнительные функции приложения позволяют управлять содержимым библиотеки , создавать новые символы, назначать им имена, объединять в категории и т.д. *Библиотека символов* поддерживает операцию "Перенести-и-Отпустить" (Drag-and-Drop) для обмена данными между экранными формами GraphWorX32 и своим содержимым.

В GraphWorX32 используются два вида просмотра Библиотеки Символов:

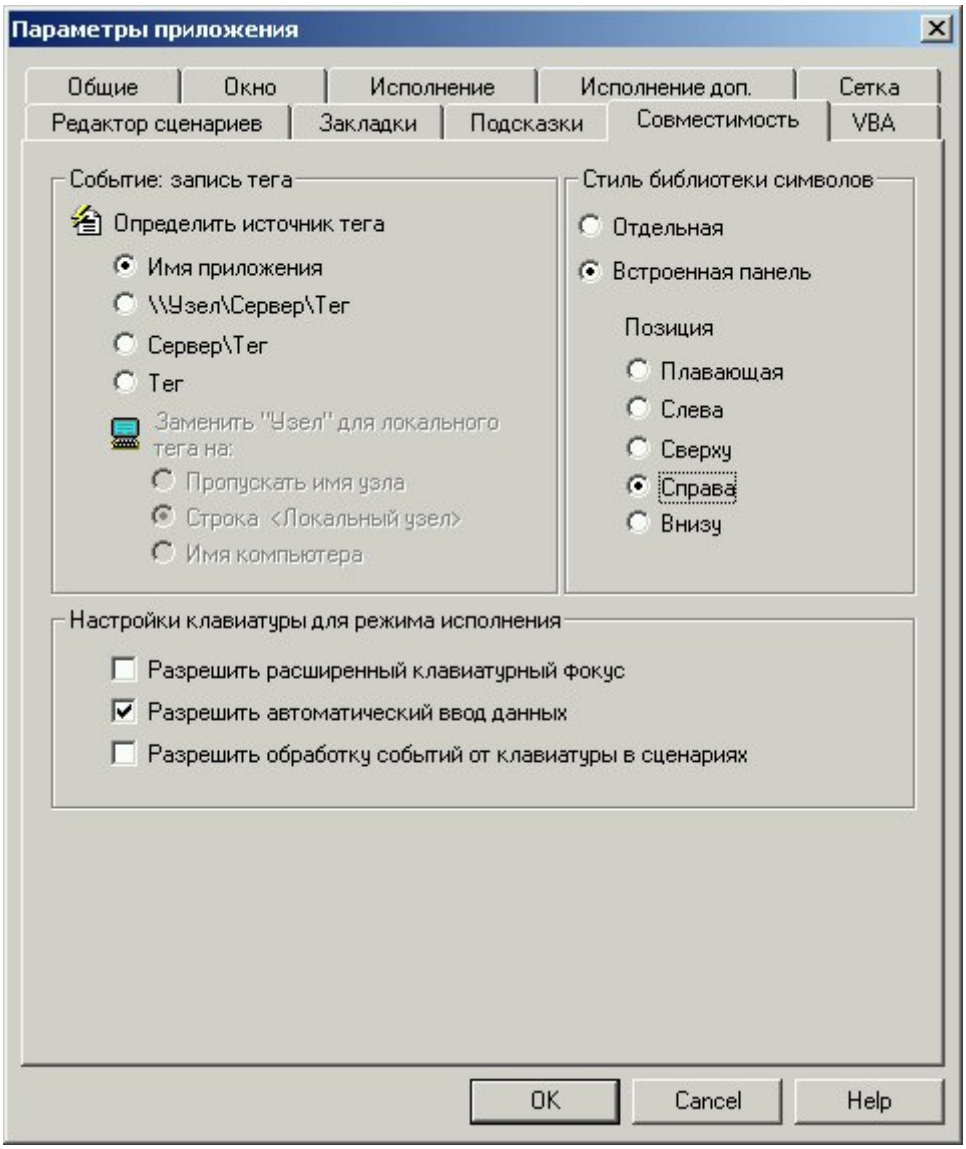
- Стандартная отдельная Библиотека Символов
- Встроенная панель символов

Для переключения между режимами:

1. Выбрать **Параметры приложения** из меню **Формат**.

2. Появляется диалоговое окно **Параметры Приложения**. Выберите закладку **Совместимость**. Под полем **Стиль Библиотеки Символов** можно выбрать режим **Отдельная или Встроенная панель символов**.

- Если установлен режим **Отдельная**, нажатие на кнопке **Импортировать символ** на панели инструментов **Рисование** запускает **SymbolLibrary.exe**. Также можно выбрать Импортировать **> Символ** из меню **Графика**.
- Если установлен режим **Встроенная панель символов**, Библиотека Символов располагается вместе с Панелью Символов в экранной форме. Можно выбрать расположение Панели Символов – справа, слева, вверху, внизу или в плавающем окне. Кнопка **Импортировать символ** на панели инструментов **Рисование** управляет видимостью Панели символов (вместо запуска Библиотеки Символов). Команда **Скрыть Панель Символов** из меню **Вид** в этом случае доступна и диалог **Показать панели инструментов** также содержит флажок **Символы** для управления видимостью **Панели Символов.** Команда **Импортировать > Символ** удалена из меню **Графика**.

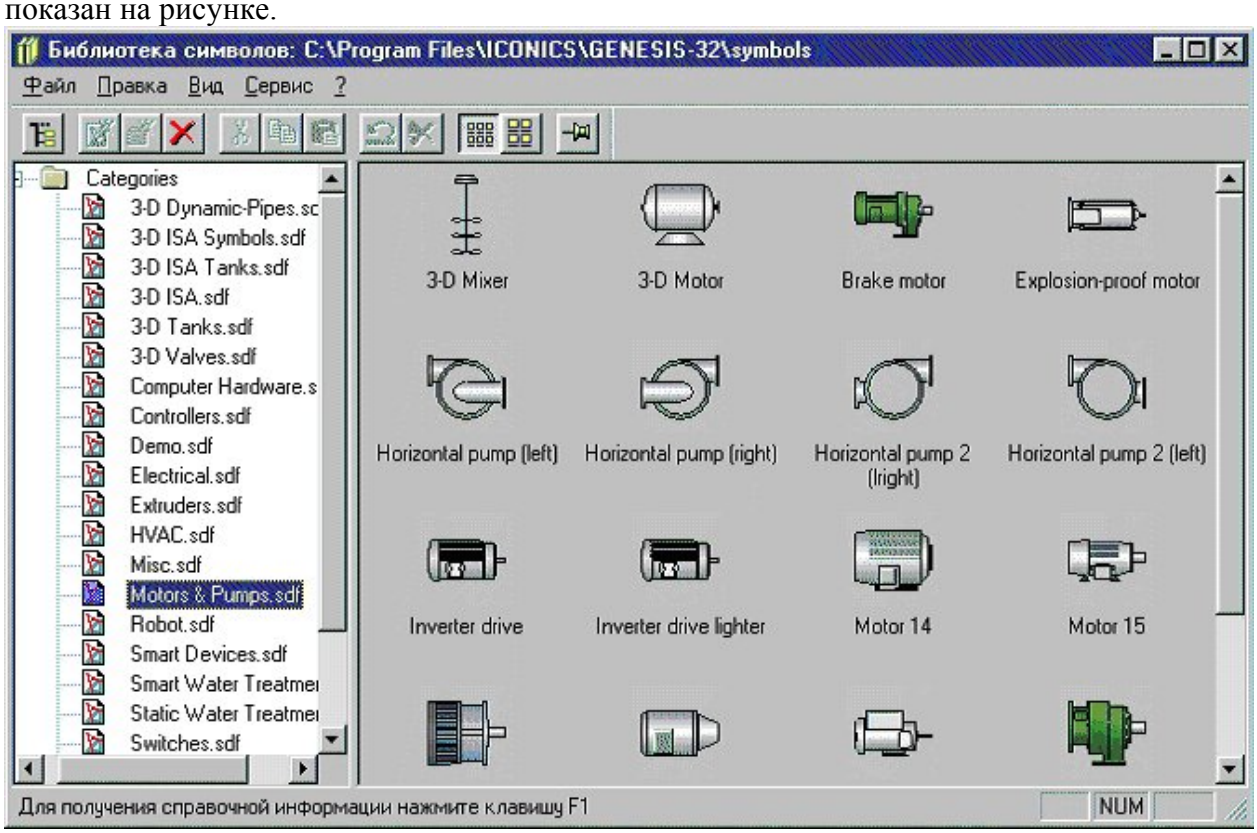


Меню Панели Символов.

Основное меню Встроенной панели Символов отображается при нажатии верхней левой клавиши, как показано на рисунке. Также можно щелкнуть правой клавишей в любом месте окна. С помощью этого меню можно добавлять, переименовывать, удалять, блокировать и разблокировать файлы категорий, изменять вид просмотра символов на панели. Также можно вырезать, копировать, вставлять, удалять и переименовывать символы. Кроме того, из основного меню можно определить корневой каталог для файлов .sdf. Предыдущая версия Библиотеки Символов поддерживала иерархическую структуру файлов категорий; другими словами,файлы категорий могли располагаться в различных подкаталогах. Встроенная Библиотека символов располагает все файлы категорий на верхнем уровне (даже включенные в подкаталоги) и показывает путь к каждой категории, так что файлы категорий не теряются.

Примечание**: полная информация о функциях меню содержится в документации Библиотеки**

Инструмент *Импортировать символ* и команда **Импортировать -- Символ** меню **Графика** предназначены для запуска приложения *Библиотека символов*, входящего в поставку GraphWorX32. Внешний вид главного окна приложения *Библиотека символов* показан на рисунке.



Библиотека символов

Настройки реестра.

Вид Библиотеки Символов определяется во время инсталляции при создании параметров реестра в разделе:

[HKEY_LOCAL_MACHINE\SOFTWARE\ICONICS\Gwx32\Compatibility Properties]:

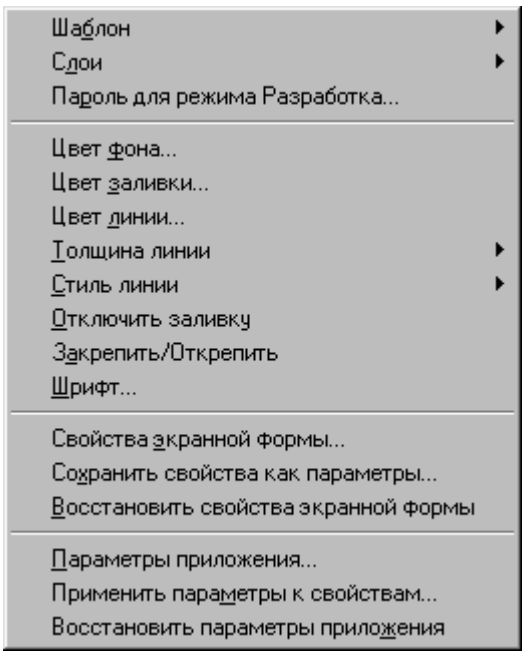
режим панели символов: отдельная или встроенная панель символов.

встроенная панель: плавающее окно, слава, справа, вверху или внизу.

Настройки сохраняются в этом разделе отдельно для каждого пользователя.

4.8 Форматирование объектов

Настоящий подраздел содержит указания по форматированию графических объектов и описание соответствующих команд меню Формат. Внешний вид меню Формат показан на рисунке.



*Команды меню **Формат***

Команда Цвет фона

Данная команда позволяет изменить цвет фона выделенного графического объекта. После выбора строки **Цвет фона** меню **Формат** на экран монитора выводится **Палитра цветов,** внешний вид которой показан на рис. 4-17.

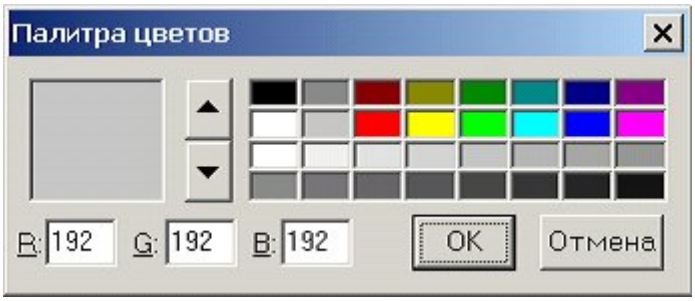


Рис. 4-17 Палитра цветов

Команда Цвет заливки

Данная команда позволяет выбрать цвет заливки для выделенного объекта из *Палитры цветов*.

Команда Цвет линии

Данная команда позволяет выбрать цвет линии для выделенного объекта из *Палитры цветов*.

Команда Толщина линии

Данная команда позволяет установить толщину линии для выделенного объекта.

Для установки толщины линии

1. Выделите объект
2. Выберите строку **Толщина линии** меню **Формат**.
3. В подменю, появившемся справа от строки **Толщина линии**, выберите требуемую толщину линии.

Команда Стиль линии

Данная команда позволяет выбрать внешний вид (стиль отображения) линии для выделенного объекта.

Для выбора стиля линии

1. Выделите объект
2. Выберите строку **Стиль линии** меню **Формат**.
3. В подменю, появившемся справа от строки **Стиль линии**, выберите требуемый вид отображения линии.

Примечание. Стиль может быть применен только к линии, для которой установлена толщина 0.

Команда Отключить заливку

Данная команда позволяет включить/отключить заливку внутренней области выделенного объекта цветом, установленным в *Палитре цветов* в качестве текущего.

Для включения/отключения заливки внутренней области графического объекта

1. Выделите объект
2. Выберите строку **Отключить заливку** меню **Формат** или инструмент *Включить заливку/Отключить заливку* в панели инструментов *Рисование*.

Примечание. Для графических объектов *Линия*, *Растровое изображение* и *Метафайл* применение заливки цветом невозможно.

Команда Закрепить/Открепить

Данная команда позволяет зафиксировать/освободить графический объект. После фиксации графического объекта невозможно изменить его размеры и положение в рабочей области экранной формы.

Для закрепления графического объекта

1. Выделите объект
2. Выберите команду **Закрепить/Открепить** меню **Формат**. В дальнейшем при помещении курсора мыши в область отображения закрепленного объекта справа от курсора будет появляться изображение замка, указывающее на то, что объект зафиксирован.

Команда Шрифт

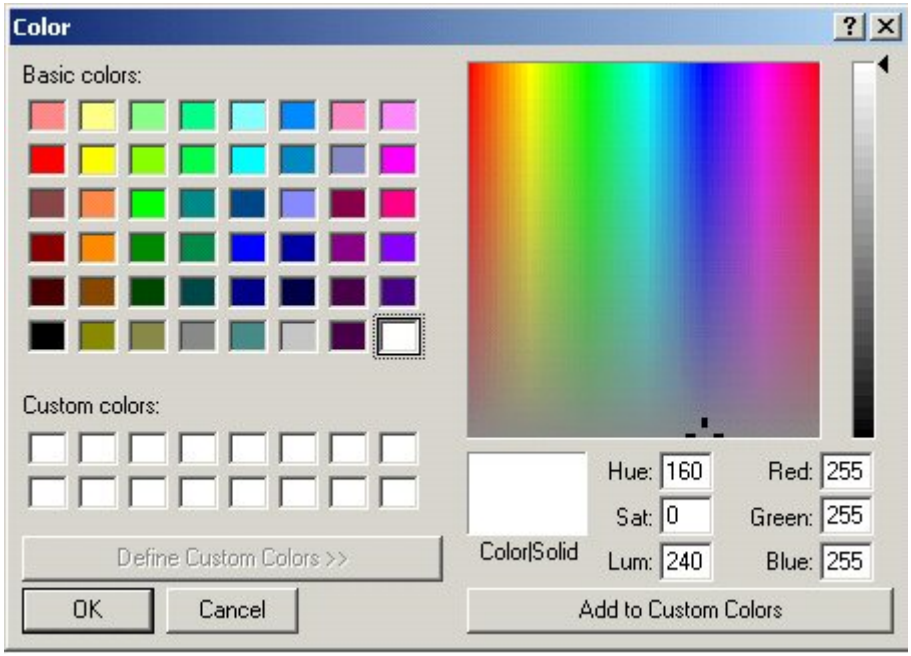
Данная команда позволяет установить тип, стиль, размер и набор шрифта. Установка параметров шрифта может быть произведена до или после создания объекта Текст.

Для установки параметров шрифта

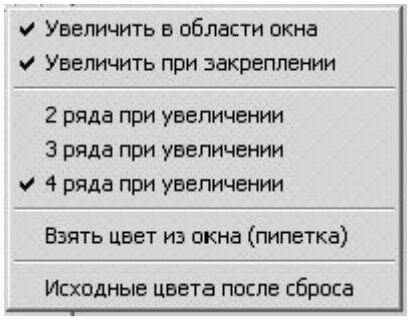
1. Выделите объект *Текст*
2. Выберите строку **Шрифт** меню **Формат** и в появившейся системной диалоговой панели установите требуемые параметры шрифтового оформления текста.

4.9 Настройка палитры цветов

Двойной щелчок по окну **Палитра цветов** открывает диалоговое окно **Цвет**, показанное на рисунке:

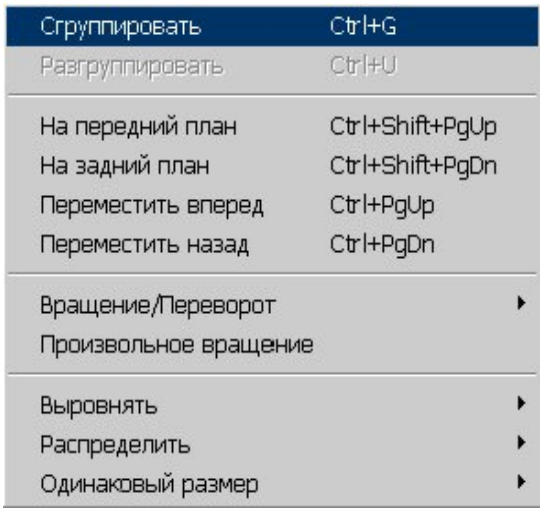


Можно использовать ползунок для настройки цвета и его оттенков. Выбранный цвет отображается в диалоговом окне **Color/Solid.** Можно добавлять дополнительные цвета в палитру с помощью кнопки **Add to Custom Colors.**
По щелчку правой клавиши мыши на окну **Палитра цветов** будет выведено меню, показанное на рисунке ниже, где можно настроить параметры палитры цветов.

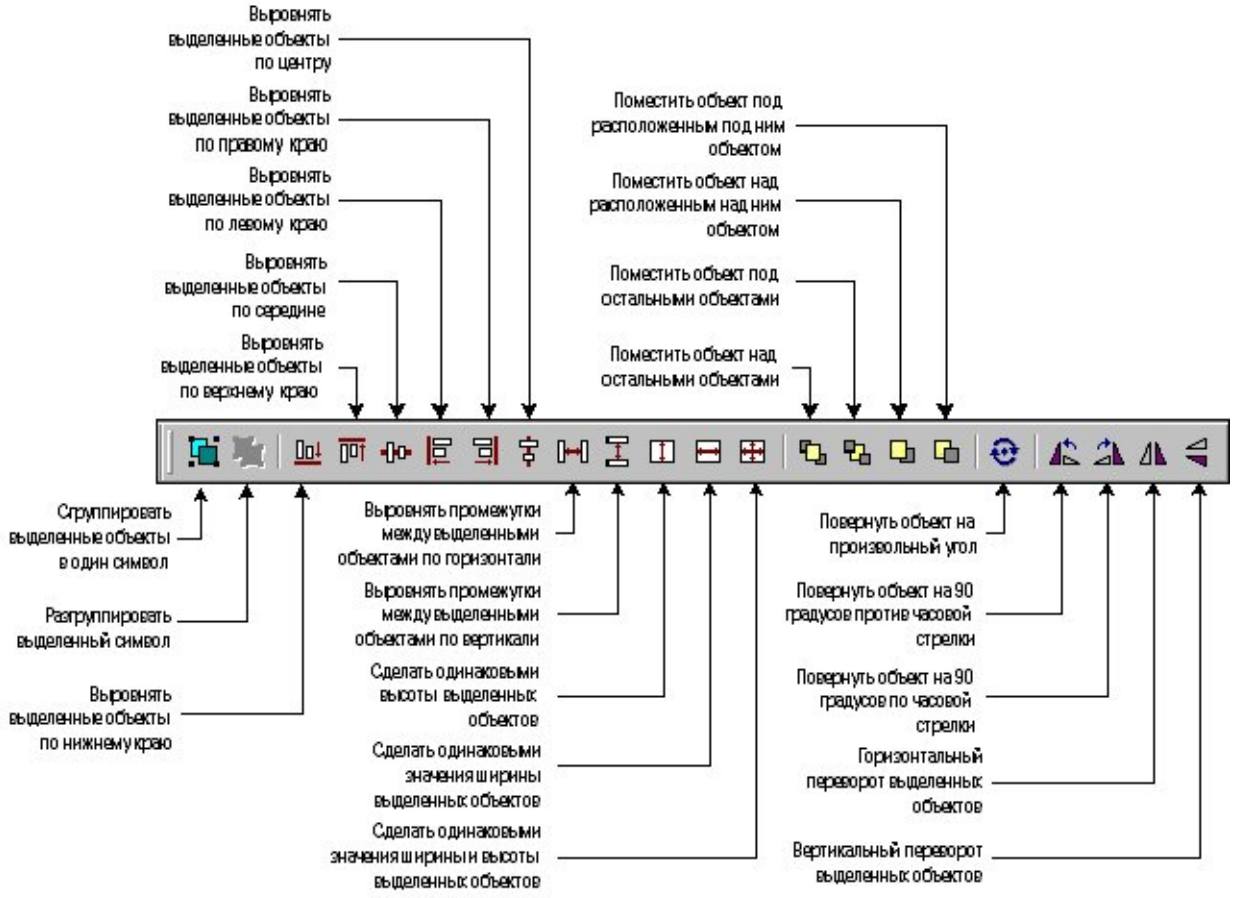


4.10 Изменение расположения объектов

Меню **Расположение** и панель инструментов *Расположение* позволяют объединять графические объекты в символы, разгруппировывать символы, устанавливать одинаковые промежутки между объектами, выравнивать объекты и выполнять другие операции над графическими объектами.



*Меню **Расположение***



*Панель инструментов **Расположение***

Команда Сгруппировать

Данная команда позволяет объединить несколько графических объектов в один символ, обладающий общими свойствами примитивных графических объектов GraphWorX32.

Команда Разгруппировать

Данная команда позволяет разбить символ на составляющие графические объекты.

Выравнивание объектов

Подменю **Выровнять** меню **Расположение** и соответствующая группа инструментов в панели инструментов *Расположение* позволяют выполнять операции выравнивания нескольких выделенных графических объектов по верхнему и нижнему краям, по середине, по левому и правому краям, по центру, а также по центру и по середине.

Для выравнивания объектов

1. Выделите несколько объектов
2. Выберите требуемую строку подменю **Выровнять** меню **Расположение** или щелкните мышью на соответствующем инструменте панели инструментов *Расположение*.

По нижнему краю		По верхнему краю	
По левому краю		По правому краю	
По центру		По середине	

Имеется возможность выравнивания размеров нескольких графических объектов (высота, ширина), для чего служит подменю **Одинаковый размер** меню **Расположение**. Данная функция применима к таким графическим объектам, как прямоугольники, круги, эллипсы, линии, кнопки, элементы отображения значений параметров процесса

Для выравнивания размеров нескольких объектов

Выделите несколько объектов

Выберите требуемую строку подменю **Одинаковый размер** меню **Расположение** или щелкните мышью на соответствующем инструменте панели инструментов *Расположение*.

Одинаковый размер по высоте	
Одинаковый размер по ширине	
Одинаковый размер по высоте и ширине	

Команда Выдвинуть на передний план

Данная команда позволяет поместить выделенный графический объект поверх остальных, перекрывающих его или находящихся рядом с ним в пределах текущей рабочей области, графических объектов.

Команда Переместить на задний план

Данная команда позволяет поместить выделенный графический объект под остальными, перекрывающими его или находящихся рядом с ним в пределах текущей рабочей области, графических объектов.

Команда Переместить вперед

Данная команда позволяет поместить выделенный графический объект на один уровень выше, чем остальные, перекрывающие его или находящиеся рядом с ним, графические объекты.

Команда Переместить назад

Данная команда позволяет поместить выделенный графический объект на один уровень ниже, чем остальные, перекрывающие его или находящиеся рядом с ним, графические объекты.

4.11 Вращение графических объектов

Данная функция предназначена для поворота выделенных графических объектов на произвольный или заданный угол.

Произвольное вращение статических объектов

Произвольное вращение означает, что графический объект поворачивается и, в дальнейшем, отображается под произвольным углом относительно своего первоначального положения. Функция произвольного вращения может быть применена к таким объектам, как *Линия*, *Сегментная линия*, *Прямоугольник*, *Эллипс*, *Дуга, С*имволы и Растр. Объекты типа *Текст* и *Метафайл* не могут быт подвергнуты вращению на произвольный угол.
Поворот объектов может быть выполнен с помощью **Инспектора свойств** путем ввода значения угла в градусах, на который требуется повернуть объект, в поле **Угол** диалоговой панели **Инспектора свойств**.
Вращение графических объектов производится путем выбора команды **Произвольное вращение** меню **Расположение** или щелчком на кнопке *Произвольное вращение* в панели инструментов *Расположение*.
Примечание. Графические объекты, повернутые на произвольный угол, могут быть перевернуты или повернуты на угол 90° при помощи соответствующих команд.

Команды Левый/Правый поворот
Команды **Левый поворот/Правый поворот** подменю **Вращение/Переворот** и соответствующие кнопки панели инструментов *Расположение* позволяют выполнить поворот выделенного графического объекта на 90° против или по часовой стрелки относительно первоначального положения. Функции левого и правого поворота могут быть применены к таким объектам, как *Сегментная линия*, *Текст*, *Прямоугольник*, *Эллипс*, *Дуга*, *Растр*, а также к символам.

Команды Вертикальный/Горизонтальный переворот
Команды **Вертикальный/Горизонтальный переворот** подменю **Вращение/Переворот** и соответствующие кнопки панели инструментов *Расположение* позволяют выполнить переворот выделенного графического объекта относительно вертикальной/горизонтальной оси его первоначального положения. Данные функции могут быть применены к таким объектам, как *Сегментная линия*, *Прямоугольник*, *Эллипс*, *Дуга*, *Растр*, а также к символам. Функция неприменима к объектам типа *Текст* и *Метафайл*.

Глава 5. Функции редактирования экранной формы

5.1 Введение
5.2 Операции Отменить/Повторить
5.3 Основные операции редактирования
5.4 Операции поиска, замены и формирования отчета содержимого экранной формы
5.5 Вставка объектов
5.6 Обновление совместно используемых объектов

5.1 Введение

Настоящий раздел содержит указания по редактированию графических объектов и экранных форм GraphWorX32. Данные операции включают в себя перенос графических объектов в универсальный буфер обмена Clipboard (операция **Вырезать**), копирование, дублирование, замена, обновление объектов, входящих в несколько экранных форм, а также вставку новых объектов. Доступ к функциям редактирования может осуществляться через меню **Правка**, внешний вид которого показан на рисунке 5-1.

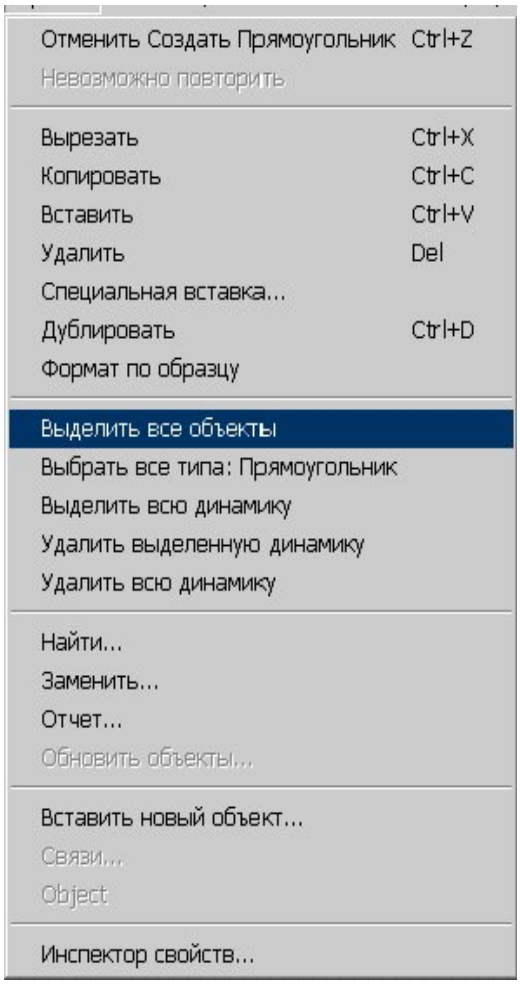


*Рис. 5-1 Внешний вид меню **Правка***

5.2 Операции Отменить/Повторить

Для отмены последней выполненной операции редактирования следует воспользоваться командой **Отменить** меню **Правка**. Для повтора последней отмененной операции

редактирования следует выбрать команду **Повторить** меню **Правка**. Название операции, которая может быть повторена или отменена, отображается справа от названия соответствующей команды в меню **Правка**. Операции отмены и повтора могут выполняться при помощи соответствующих кнопок главной панели инструментов.
Если вместо строки **Отменить/Повторить** в меню **Правка** отображаются строки **Невозможно отменить/Невозможно повторить**, то выполнение соответствующих операций невозможно.
Примечание. Для отмены нескольких операций редактирования выполните несколько щелчков левой клавишей мыши на кнопке **Отменить**. Аналогичные действия относятся к операции **Повторить**.
Быстрый доступ к операциям:

Панель инструментов:



Комбинации клавиш:

CTRL+Z – отменить
CTRL+Y – повторить

5.3 Основные операции редактирования

Имеются следующие основные операции редактирования: **Вырезать**, **Копировать**, **Вставить**, **Специальная вставка** и **Дублировать, Формат по образцу**

Операция Копировать

Данная операция предназначена для помещения копии выделенного объекта в универсальный буфер обмена Clipboard.
Для копирования объекта:

1. Выделите объект. Вдоль границ прямоугольной области, в которой находится объект, появятся небольшие символы квадратной формы.
2. Выберите строку **Копировать** меню **Правка** или произведите щелчок левой клавишей мыши на кнопке *Копировать* главной панели инструментов.
3. Копия выделенного объекта будет помещена в универсальный буфер обмена Clipboard.

Быстрый доступ к операции:

Панель инструментов:



Комбинация клавиш:

CTRL+C – копировать.

Операция Вырезать

Данная операция предназначена для удаления выделенного объекта из рабочей области экранной формы и помещения в универсальный буфер обмена Clipboard
Для вырезания объекта:

1. Выделите объект. Вдоль границ прямоугольной области, в которой находится объект, появятся небольшие символы квадратной формы.
2. Выберите строку **Вырезать** меню **Правка** или произведите щелчок левой клавишей мыши на кнопке *Копировать* главной панели инструментов.

Выделенный объект будет перенесен в универсальный буфер обмена Clipboard и исчезнет с экрана. Вырезанный объект будет оставаться в буфере обмена до выполнения следующей операции копирования или вырезания.
Примечание. Операция удаления без помещения в универсальный буфер обмена может быть выполнена нажатием клавиши Delete.

Быстрый доступ к операции:

Панель инструментов:



Комбинация клавиш:

CTRL+X – вырезать.

Операция Вставить

Данная операция предназначена для вставки объекта, находящегося в универсальном буфере обмена Clipboard, в рабочую область экранной формы. Объект находится в буфере обмена до очередного выполнения операции **Вырезать** или **Копировать** и может быть вставлен в рабочую область экранной формы. Операция вставки не может быть выполнена, если универсальный буфер обмена пуст.

Для вставки объекта из универсального буфера обмена Clipboard:

1. Выберите строку **Вставить** меню **Правка** или щелкните левой клавишей мыши на кнопке *Вставить* главной панели инструментов.
2. Объект, который в текущий момент находится в универсальном буфере обмена, будет вставлен в рабочую область экранной формы GraphWorX32 вблизи центра. После выполнения данной операции объект будет выделен, что обеспечивает возможность изменения его свойств и положения.

Быстрый доступ к операции:

Панель инструментов:



Комбинация клавиш:

CTRL+V – вставить.

Специальная вставка

Данная команда предназначена для вставки в экранную форму объектов выбранного формата. Например, некоторые приложения могут одновременно создавать в универсальном буфере обмена Clipboard копии графического изображения в форматах BMP (растр) и WMF (метафайл). Таким образом, команда **Специальная вставки** меню **Правка** позволяет выбрать формат вставляемого объекта, – растровое изображение (BMP), метафайл (WMF) или аппаратно-независимое растровое изображения (DIB).
Если в диалоговой панели **Специальная вставка**, внешний вид которой показан на рисунке, отметить флажок **В виде значка**, то содержимое универсального буфера обмена будет вставлено в экранную форму в виде пиктограммы, связанной в Windows с объектами данного формата.
Имеется возможность специальной вставки как самого объекта, так и ссылки на него (установлен переключатель **Вставить связь**). В последнем случае объект будет храниться не в файле экранной формы GraphWorX32, а в файле приложения, с помощью которого была создана копия указанного объекта в универсальном буфере обмена

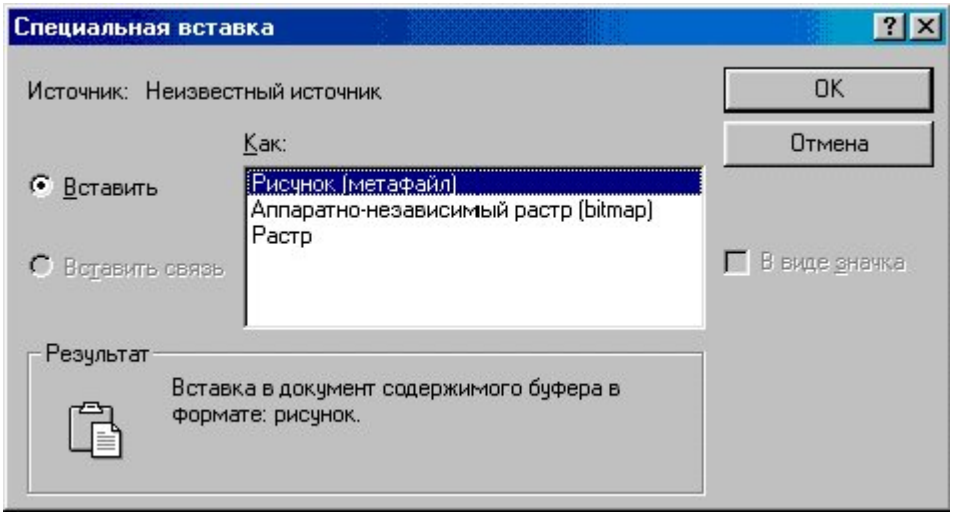


*Рис. 5-2. Диалоговая панель **Специальная вставка**.*

Операция Дублировать

Данная операция предназначена для создания копий выделенного объекта в рабочей области экранной формы с одновременным помещением копии указанного объекта в универсальном буфере обмена Clipboard.

Для дублирования объекта

1. Выделите объект. Вдоль границ прямоугольной области, в которой находится объект, появятся небольшие символы квадратной формы.
2. Выберите строку **Дублировать** меню **Правка** или произведите щелчок левой клавишей мыши на кнопке *Дублировать* главной панели инструментов.

Копия выделенного объекта будет перенесена в универсальный буфер обмена Clipboard, а также появится в рабочей области экранной формы. Копия объекта будет оставаться в буфере обмена до выполнения следующей операции копирования или вырезания.

Быстрый доступ к операции:

Панель инструментов:



Комбинация клавиш:

CTRL+D – дублировать.

Формат по образцу

Эта операция позволяет осуществлять копирование форматирования выделенного объекта.

Чтобы скопировать формат объекта:

4. Выделите объект, формат которого необходимо скопировать. Вдоль границ прямоугольной области, в которой находится объект, появятся небольшие символы квадратной формы.
3. Выберите строку **Формат по образцу** меню **Правка** или произведите щелчок левой клавишей мыши на кнопке *Формат по образцу* главной панели инструментов.
4. Выберите объект, на который требуется скопировать форматирование.

Быстрый доступ к операции:

Панель инструментов:



5.4 Операции поиска, замены и формирования отчета содержимого экранной формы

Операция поиска с дополнительной возможностью формирования текстового отчета позволяет находить фрагменты текста в экранных формах GraphWorX32, включая названия статических объектов, динамических связей и ключевые слова. Выполнение операции поиска, доступ к которой осуществляется с помощью команды **Найти** меню **Правка**, приводит к выделению найденных объектов в рабочей области экранной формы и к выводу списка указанных объектов в специальной области диалоговой панели **Найти**, представленного в виде иерархического дерева, как показано на рисунке 5-4.
Операция **Заменить** выполняется аналогично операции **Найти**, за исключением того, что в диалоговую панель **Замена**, внешний вид которой аналогичен диалоговой панели **Найти**, добавляется поле **Заменить на:**, предназначенное для ввода текстовой строки, на которую должна быть заменена найденная строка.
Имеются две различные диалоговые панели для выполнения операций поиска и замены. Внешний вид основной диалоговой панели **Найти** показан на рис 5-3. На рис.5- 4 показана диалоговая панель **Найти**, расширенная полем отображения результатов поиска и кнопками, позволяющими задать параметры отображения результатов поиска и вывести результаты поиска в текстовый файл. Для расширения основной диалоговой панели **Найти** следует нажать кнопку **Дополнительно**. Описание операций поиска и замены приведено в *Таблице 5-0*.

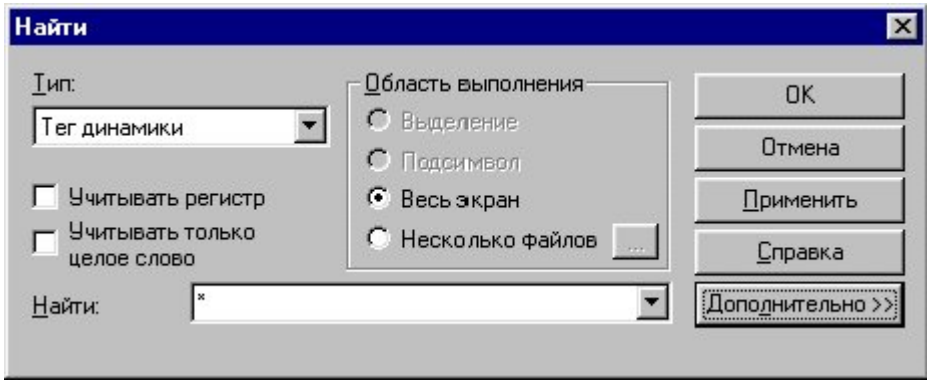


*Рис.5-3. Диалоговая панель **Найти***

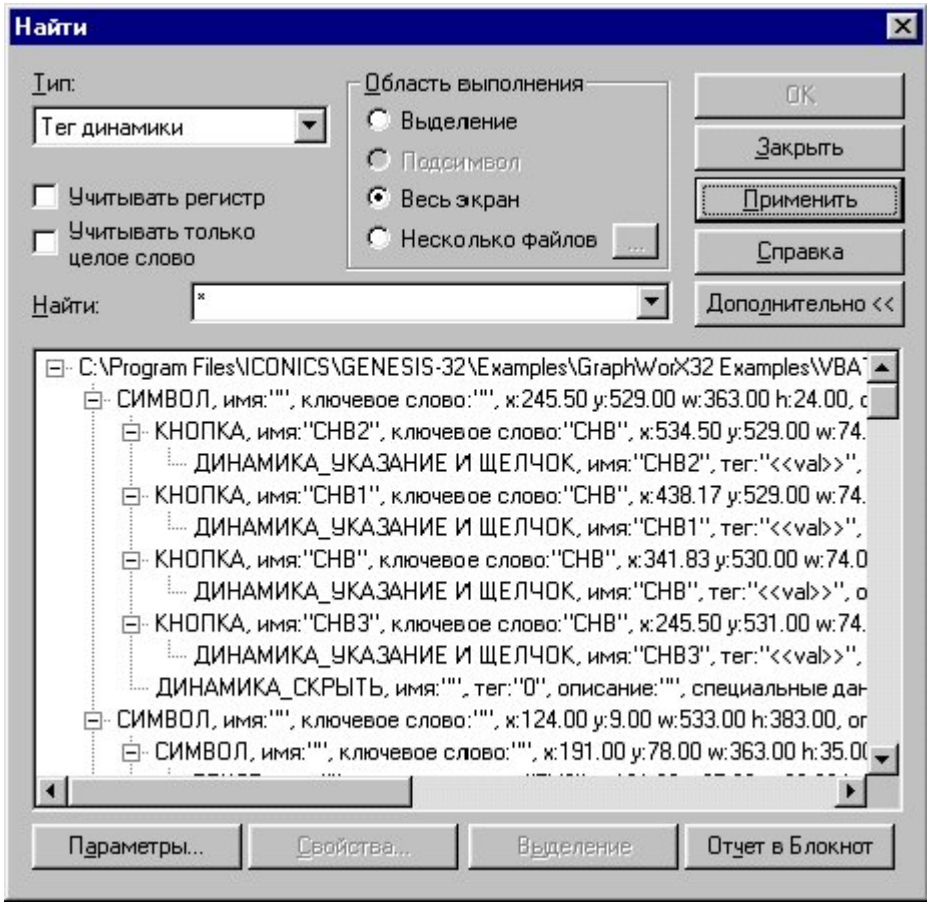


*Рис. 5-4. Расширенная диалоговая панель **Найти***

Кнопка **Параметры** расширенной диалоговой панели позволяет установить параметры отображения результатов поиска в древовидном списке. Нажатие кнопки **Параметры** приводит к выводу на экран диалоговую панель **Параметры отчета**, внешний вид которой показан на рис. 5-5

Кнопка **Отчет в Блокнот** позволяет сохранить результат поиска в текстовом файле редактора *Блокнот* (Wordpad).

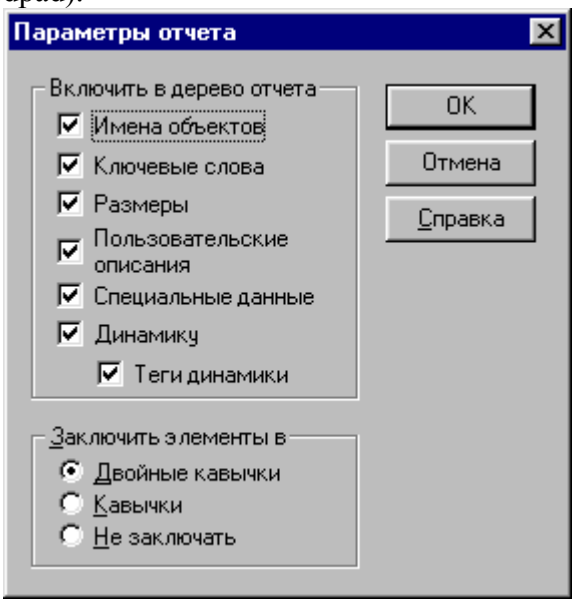


Рис. 5-5. Диалоговая панель настройки параметров вывода отчета

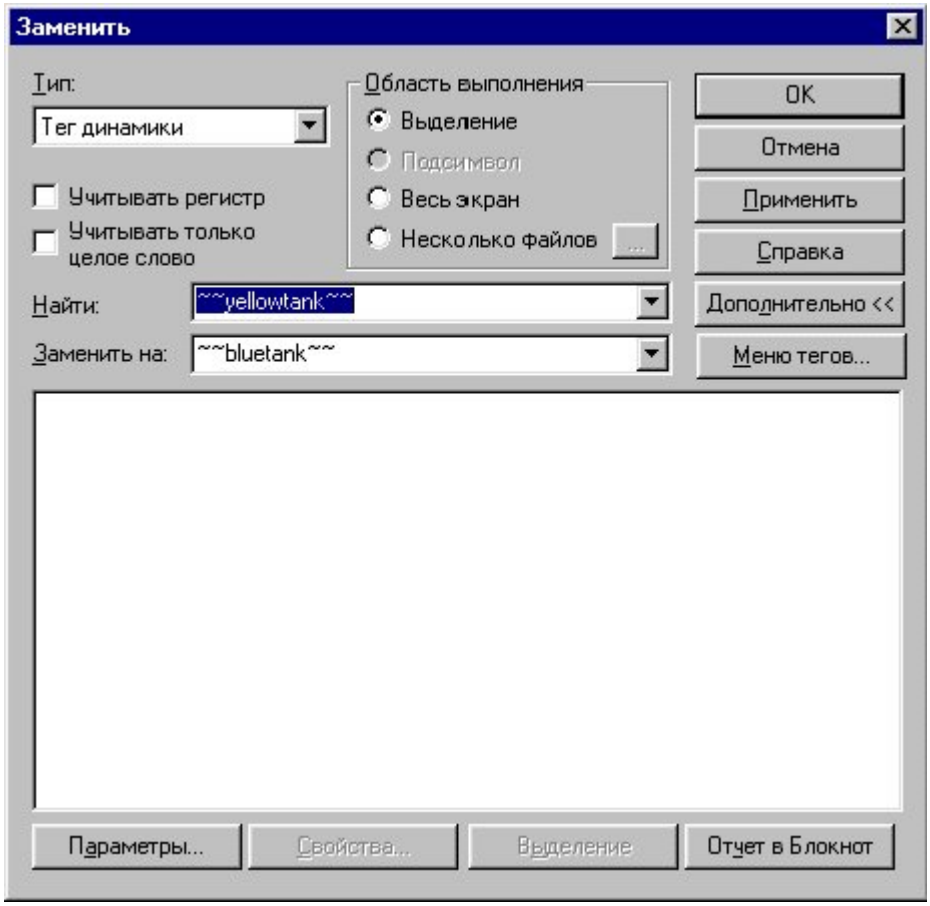


*Рис. 5-6. Диалоговая панель **Замена***

Описание назначения полей диалоговой панели Поиск/Замена

Таблица 5-0

Поле	Описание
Тип	Выпадающий список позволяет указать характеристику типа текстовой строки для поиска и/или замены. Операции поиска и замены будут выполняться среди текстовых строк заданного типа, относящихся к графическим объектам одной или нескольких экранных форм. Под текстовыми строками подразумеваются названия и значения параметров графических объектов, устанавливаемые в диалоговой панели **Инспектора свойств**.
Тег динамики	Поиск и замена выполняются по отношению к идентификаторам динамических связей графических объектов с контролируемыми параметрами технологического процесса.
Элемент текста	Поиск и замена текста в графических объектах *Фрагмент текста*.
Имя объекта	Поиск и замена имен статических объектов, а также имен объектов для динамических связей, заданных в полях **Имя объекта** в **Инспекторе свойств**
Ключевое слово	Поиск и замена глобальных ключевых слов, идентифицирующих разделяемые объекты.
Описание	Поиск и замена текстовых описаний, заданных в полях **Имя объекта** в **Инспекторе свойств**
Специальные данные	Поиск и замена специальных данных, заданных в полях **Специальные данные** в **Инспекторе свойств**
Имя файла	Поиск замена имен файлов, заданных в полях **Имя файла** в **Инспекторе свойств** объектов, к которым применено действие *Указание и щелчок*.
Область выполнения	Группа параметров, определяющих область, на которую будет распространяться выполнение операций поиска и замены.

Выделение	Поиск и замена будут выполняться только среди выделенных в настоящий момент объектов.
Подсимвол	Поиск и замена будут выполняться среди объектов, из которых был сгруппирован выделенный символ.
Весь экран	Поиск и замена будут выполняться среди всех объектов текущей экранной формы.
Несколько файлов	Поиск и замена будут выполняться в группе файлов экранных форм GraphWorX32. Если в какую-либо из обрабатываемых экранных форм вносятся изменения в процессе выполнения операций поиска и замены, перед обработкой другой экранной формы на экран монитора выводится предложение сохранить внесенные изменения.
Учитывать регистр	При установке данного флажка поиск и замена выполняются в заданной области с проверкой соответствия регистра букв (строчные, заглавные) в строках для поиска и замены.
Учитывать слово целиком	Если данный флажок установлен, то при выполнении операций поиска и замены будут учитываться только целые слова или строки текста для поиска и замены.
Найти	Данное поле, представленное выпадающим списком, предназначено для ввода текста, поиск и/или замена которого должны быть выполнены в процессе обработки области применения операции. Доступ к ранее введенным строкам текста осуществляется раскрыванием выпадающего списка. Выпадающий список содержит набор ранее введенных строк для поиска и замены, состав которого зависит от текущей выбранной области поиска, а также от типа объектов, среди которых выполняется операция. Имеется возможность ввода шаблона искомых текстовых фрагментов путем использования символа `*`. При этом результат поиска будет содержать все строки, в которых имеются фрагменты, совпадающие с введенным шаблоном. Пример шаблона поиска: *random При установке в качестве области поиска текущей экранной формы, а типа объекта, – динамической связи, результат поиска будет содержать имена тегов OPC, внутренних переменных имитации сигналов и локальных переменных, содержащие слово random.
Заменить на	Данное поле, представленное выпадающим списком, предназначено для ввода текста, который должен быть вставлен вместо найденного в процессе обработки области применения операции. Доступ к ранее введенным строкам текста осуществляется раскрыванием выпадающего списка. Выпадающий список содержит набор ранее введенных строк для замены, состав которого зависит от текущей выбранной области поиска/замены, а также от типа объектов, среди которых выполняется операция. Использование шаблонов в данном поле невозможно.
Область отображения результата выполнения операции	Древовидный список, выводимый в данную область, содержит результат выполнения операции поиска/замены. Иерархия объектов, выводимая в списке, соответствует области применения и типу, установленным в соответствующих группах параметров диалоговой панели. При выборе ветви списка (объекта или символа) и последующем нажатии кнопки **Выделение** или двойном щелчке левой клавишей мыши на ветви списка в рабочей области экранной формы произойдет выделение соответствующего графического объекта. Обратите внимание, что после поиска и замены в нескольких файлах, при выборе объекта или символа в списке и нажатии кнопки **Выделение** произойдет загрузка соответствующей экранной формы, содержащей выбранный объект. При выборе ветви списка и нажатии кнопки **Свойства** на экран будет выведена диалоговая панель **Инспектора свойств**, открытая на странице, соответствующей выделенному объекту. Если в поле **Найти** ввести шаблон '*', нажать кнопку **Применить**, то в области отображения результата поиска будет выведен аналог "дерева проекта". Ускоренный

	доступ к отдельным объектам выполняется путем выбора в древовидном списке и нажатия кнопки **Выделение** (для визуального редактирования, в том числе без разгруппировки) или **Свойства** – для вывода **Инспектора свойств**.
Кнопка **OK**	Данная кнопка предназначена для запуска выполнения операции поиска/замены с закрытием диалоговой панели **Найти/Заменить**.
Кнопка **Закрыть/Отмена**	Данная кнопка предназначена для закрытия диалоговой панели **Найти/Заменить**. Если в процессе работы с диалоговой панелью **Найти/Заменить** была нажата кнопка **Применить**, надпись на кнопке **Отмена** изменится на **Закрыть** с сохранением состояния диалоговой панели.
Кнопка **Применить**	Данная кнопка предназначена для запуска выполнения операции поиска/замены без закрытия диалоговой панели **Найти/Заменить**. Таким образом, возможен просмотр результата выполнения операции поиска/замены в соответствующей области диалоговой панели и выделение найденных объектов в рабочей области экранной формы. Если в процессе работы с диалоговой панелью **Найти/Заменить** была нажата кнопка **Применить**, надпись на кнопке **Отмена** изменится на **Закрыть** с сохранением состояния диалоговой панели.
Кнопка **Справка**	Данная кнопка предназначена для вывода на экран монитора справочной информации об операции поиска и замены.
Кнопка **Дополнительно**	Данная кнопка предназначена для отображения/скрытия расширенной части диалоговой панели **Найти/Заменить**, которая состоит из области отображения результата операции, кнопки **Параметры**, с помощью которой выводится диалоговая панель, позволяющая выбрать степень детализации отображения результата выполнения операции поиска/замены. Кроме того, в расширенной части диалоговой панели **Найти/Заменить** находится кнопка **Свойства**, предназначенная для вывода на экран монитора **Инспектора свойств** для объекта, выбранного в древовидном списке результата операции. Кнопка **Отчет в Блокнот** позволяет сохранить результат выполнения операции в текстовом файле редактора *Блокнот* (*Wordpad*).
Параметры	Открывается диалоговое окно **Параметры Отчета**, в котором выполняются настройки просмотра.
Свойства	Открывается Инспектор Свойств для выбранного пункта.
Показать	В просмотре показывается выделенный пункт.
Отчет в Блокнот	Экспорт результатов поиска в текстовый файл.

Шаблоны поиска

В поле **Найти** могут быть использованы следующие символы ввода шаблона поиска: “*”, “?”. Символ “*” применяется вместо группы символов в искомой строке. Символ “?” используется вместо одного символа. Для нахождения всех объектов выбранного типа в установленной области выполнения операции поиска/замены следует ввести в поле **Найти** символ “*”.

5.5 Вставка объектов

Данная функция предназначена для вставки объектов OLE, таких, как диаграммы *Microsoft Graph* или выражения *Microsoft Equation*, в экранную форму GraphWorX32. Приложение, с помощью которого был создан внедренный объект, будет активизироваться в пределах экранной формы. Кроме того, данная функция позволяет вставлять в экранную форму управляющие элементы ActiveX различных производителей. Страницы свойств вставленного элемента управления ActiveX будет автоматически выводиться на экран монитора после двойного щелчка левой клавишей мыши на его изображении.

При выборе команды **Вставить новый объект** меню **Правка** на экран монитора выводится системная диалоговая панель **Вставка объекта**, внешний вид которой показан на рис.5-7.

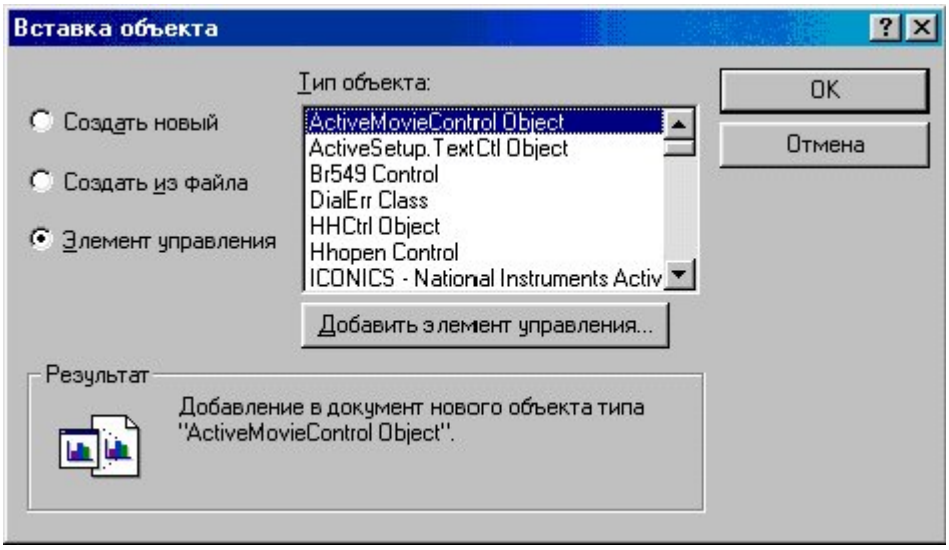


*Рис. 5-7 Диалоговая панель **Вставка объекта**.*

Для создания нового объекта

1. Установите переключатель диалоговой панели **Вставка объекта** в положение **Создать новый**, **Создать из файла** или **Элемент управления**. Описание положений переключателя приведено в табл. 5-2.

Таблица 5-2

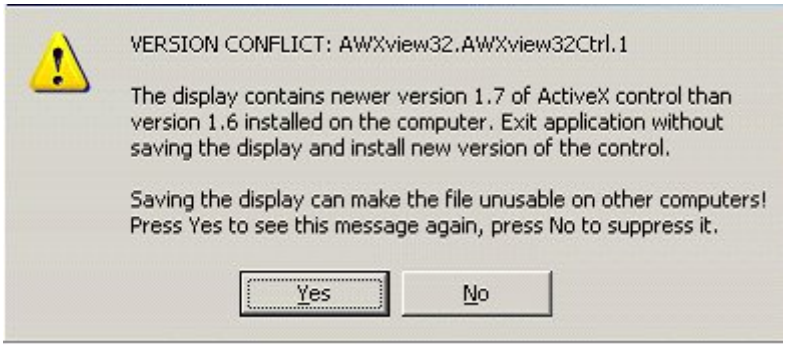
Создать новый	Позволяет внедрить в экранную форму экземпляр объекта OLE из списка **Тип объекта** диалоговой панели **Вставка объекта**. Некоторые типы объектов имеют исходные установки свойств, изменение которых производится с использованием страниц свойств объекта, вызываемых двойным щелчком левой клавишей мыши на изображении экземпляра объекта.
Создать из файла	Если переключатель установлен в данное положение, то список типов внедряемых объектов будет заменен на поле **Файл** с кнопкой **Обзор**, которое позволяет ввести имя файла, в котором содержится экземпляр внедряемого объекта.
Элемент управления	Позволяет вставить в рабочую область экранной формы экземпляр управляющего элемента ActiveX, выбранного в списке **Тип объекта** диалоговой панели **Вставка объекта**.

2. Выберите тип внедряемого объекта.

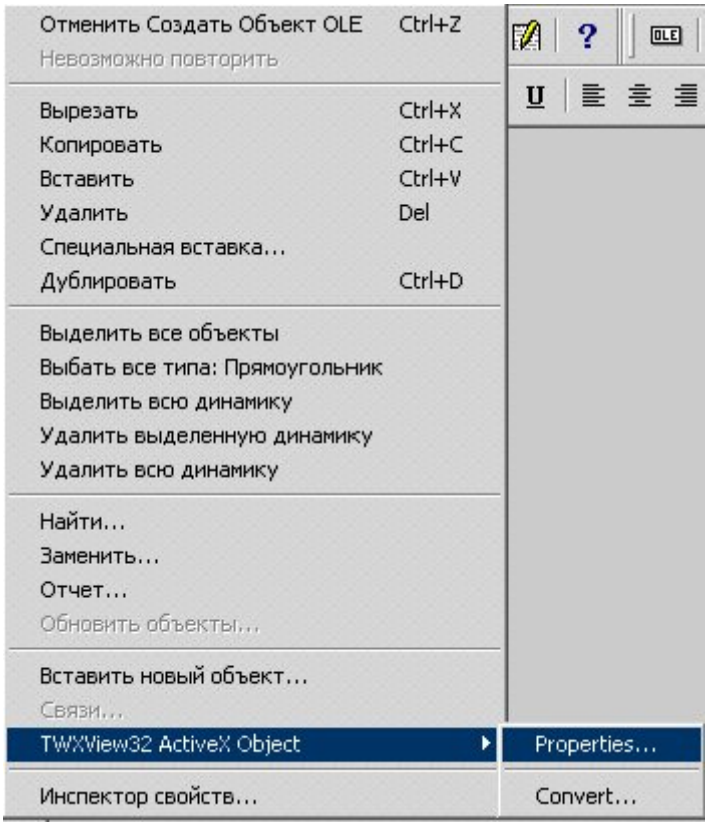
Примечание. Кнопка **Добавить элемент управления** диалоговой панели **Вставка объекта** позволяет включить в список **Тип объекта** дополнительные элементы управления из файлов.

3. Нажмите кнопку **OK**, после чего экземпляр объекта будет вставлен в рабочую область экранной формы.

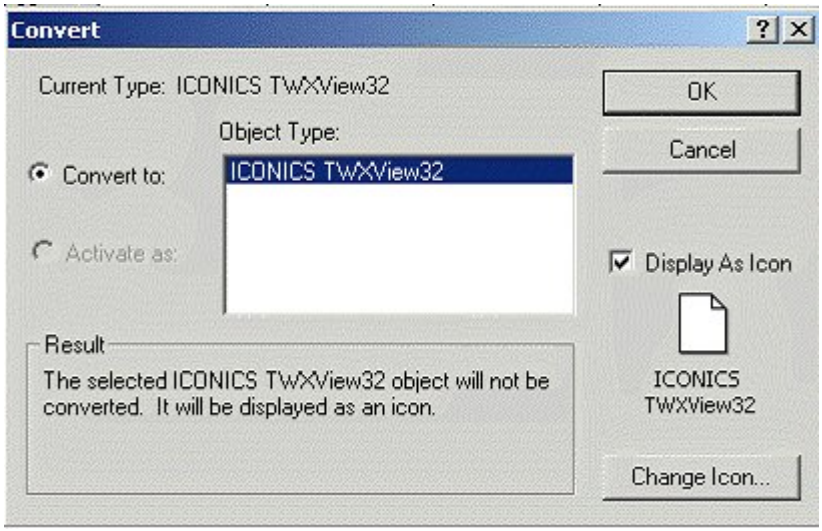
Примечание. Для вызова методов, активизации и изменения свойств управляющих элементов ActiveX в режиме *Исполнение* (за исключением поставляемых фирмой Iconics) следует использовать сценарные процедуры VBA.
GraphWorX32 сравнивает версии ActiveX –компонента, вставленного в экранную форму и установленного на компьютере, и предупреждает о возможных конфликтах.



После того, как объект вставлен в экранную форму, его свойства становятся доступны в меню **Правка**. При выборе **Объект > Свойства** из меню **Правка** открывается диалоговое окно **Свойств** объекта.



При выборе **Объект > Преобразование** из меню **Правка** открывается диалоговое окно **Преобразование**. (Язык диалогового окна зависит от операционной системы). Для преобразования объекта выберите **Тип объекта** и установите флажок **Активировать как**. Если объект большой, и требуется уменьшить количество пространства, занимаемого на экране, установите флажок **В виде значка**. (Display As Icon). Для выбора иконки, отличной от стандартной, нажмите кнопку **Изменить значок** (Change Icon).



5.6 Обновление совместно используемых объектов

Функция обновления *совместно используемых объектов* реализована в виде команды **Обновить объекты** меню **Правка**.

Совместно используемые объекты могут быть реализованы на основе обычных графических объектов и их динамических связей с контролируемыми параметрами путем ввода так называемых *разделяемых ключевых слов* в соответствующие поля **Инспектора свойств**, как показано на рис.5-8.

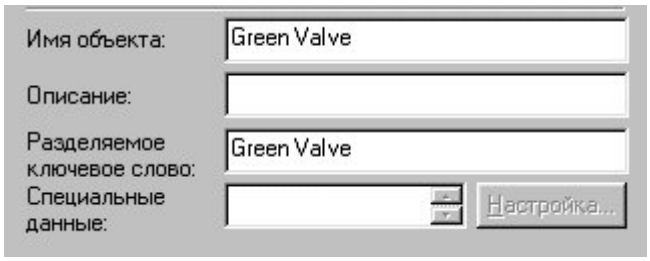


Рис. 5-8 Создание библиотечного совместно используемого символа

В поле **Имя объекта Инспектора свойств** содержится имя объекта для осуществления доступа к его свойствам через набор методов OLE Automation, а также для уникальной идентификации библиотечных символов одного типа при работе с псевдонимами, а также при выполнении операций поиска и замены.

В поле **Разделяемое ключевое слово** содержится строка, обеспечивающая возможность одновременного обновления внешнего вида и других свойств библиотечного символа, входящего во множество экранных форм проекта, в случае, если в какой-либо экранной форме пользователь изменил указанные свойства. На рис.5-9 показано выполнение операции обновления нескольких экземпляров библиотечного символа типа *Задвижка*.

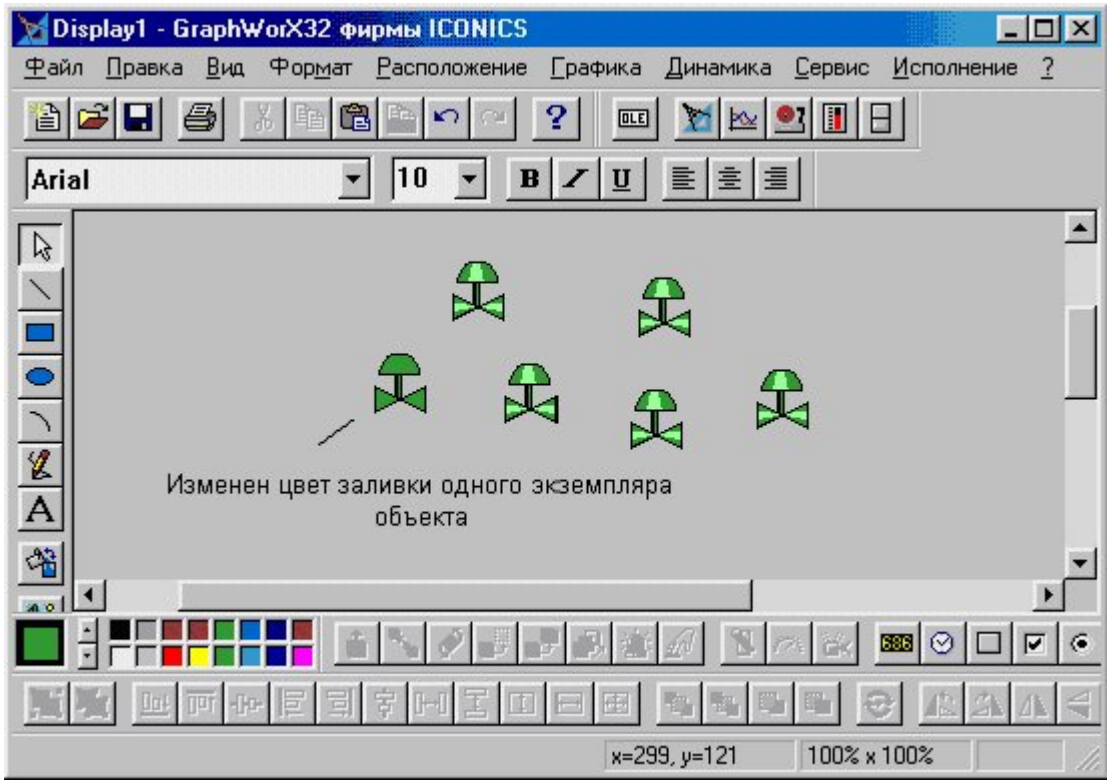


Рис. 5-9. Обновление всех экземпляров библиотечного символа

После выбора строки **Обновить объекты** меню **Правка** на экран монитора выводится диалоговая панель **Обновить совместно используемые объекты**, внешний вид которой показан на рис.5-10.

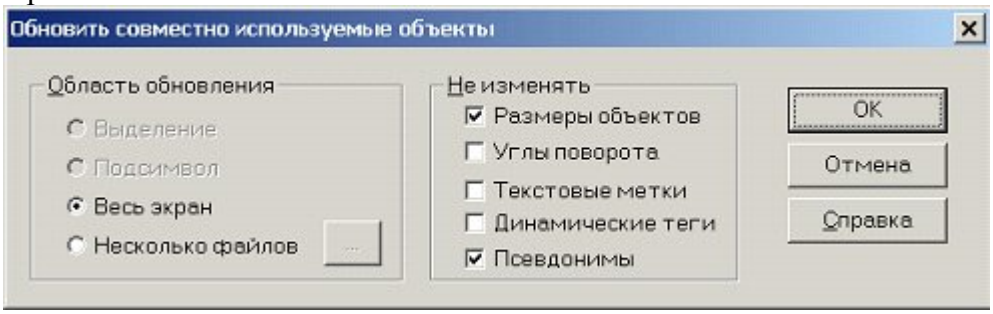


Рис. 5-10 Диалоговая панель выбора области применения и параметров обновления совместно используемых объектов

Описание полей диалоговой панели Обновить совместно используемые объекты

Таблица 5-3

Параметр	Описание
Область обновления	Группа параметров, определяющих область, на которую будет распространяться выполнение операций обновления объектов с разделяемым ключевым словом.
Выделение	Обновление будет выполняться только для выделенных в настоящий момент объектов.
Подсимвол	Обновление будет выполняться в объектах, из которых был сгруппирован выделенный символ.
Весь экран	Обновление будет выполняться для всех объектов текущей экранной формы.
Несколько файлов	Обновление будет выполняться в группе файлов экранных форм GraphWorX32. Если в какую-либо из обрабатываемых экранных форм вносятся изменения в процессе выполнения операции обновления совместно используемых объектов, перед обработкой каждой следующей экранной формы на экран монитора выводится предложение сохранить внесенные изменения.

Не изменять	Данная группа параметров позволяет выбрать ряд свойств обновляемых объектов, которые не требуется подвергать изменениям при обновлении совместно используемых объектов. Неизменными могут оставаться размеры объектов, углы их поворота, текстовые надписи и динамические связи.

Примечание: При обновлении объектов Вы можете изменять или оставлять неизменными группы *источник данных* и *псевдонимы* независимо друг от друга. Однако, при выборе в диалоговой панели **Обновить совместно используемые объекты** пункта **не изменять Динамические теги**, группа **Псевдонимы** станет недоступна для выбора. Если группа **не изменять Динамические теги** не отмечена, то есть возможность изменять или оставлять неизменными **Псевдонимы** обновляемых объектов.

Например:

1. Создайте прямоугольник.
2. Пропишите RECT как разделяемое ключевое слово в Инспекторе свойств.
3. Добавьте свойство Размер и установите источник данных <<tag>>.
4. Продублируйте выбранный прямоугольник **(CTRL+D).**
5. Привяжите псевдоним <<tag>> первого прямоугольника к ICONICS.Simulator.1\SimulatePLC.Sine
6. Привяжите псевдоним <<tag>> второго прямоугольника к ICONICS.Simulator.1\SimulatePLC.Ramp

Задача состоит в том, чтобы изменить источник этих объединенных прямоугольников с <<tag>> на <\\<<machine>>\<<tag>>>, пытаясь сохранить привязку псевдонимов к sine и ramp.
Для решения этой задачи нужно изменить источник данных в одном из прямоугольников и затем, оставляя этот прямоугольник выбранным, обновить другой прямоугольник, выбрав пункт **Обновить объекты** в меню **Правка**.
После этих изменений, источники данных этих прямоугольников будут обновлены с <<tag>> на <\\<<machine>>\<<tag>>>, но превдонимы останутся привязаны к sine и ramp.

Заметьте, что если псевдоним привязать к <<machine>> перед обновлением другого объединенного с ним объекта, обновление произойдёт как обычно.

Рекомендации по использованию объектов с разделяемыми ключевыми словами

Предположим, что экранная форма GraphWorX32 содержит большое количество графических объектов, отображающих состояния и значения входных сигналов, а также представляющих оператору возможность устанавливать значения выходных сигналов. Если все текстовые надписи для входных сигналов объединены ключевым словом *INPUT*, а для выходных сигналов, – *OUTPUT*, то появляется возможность быстрого изменения свойств текстовых надписей для входов и выходов, включая цвет заливки, текста, трехмерные эффекты и другие. Очевидно, что перед обновлением следует отметить флажок **Текстовые метки** в группе **Не изменять**.
Предположим, что при создании экранных форм проекта было создано множество графических изображений насосов, которым в процессе разработки было назначено разделяемое ключевое слово *Pump*. Предположим также, что все указанные графические изображения представляют в нескольких экранных формах состояние одного и того же реального насоса. Если по какой-либо причине произошло изменение назначения одного или нескольких входных и выходных сигналов, характеризующих состояние насоса, то имеется возможность быстрого переназначения источников динамических связей изображений насоса с входными сигналами во всех экранных формах проекта. При этом следует учитывать, что в описанном случае обновлению не должны подвергаться размеры изображений насоса.

Глава 6. Просмотр экранных форм

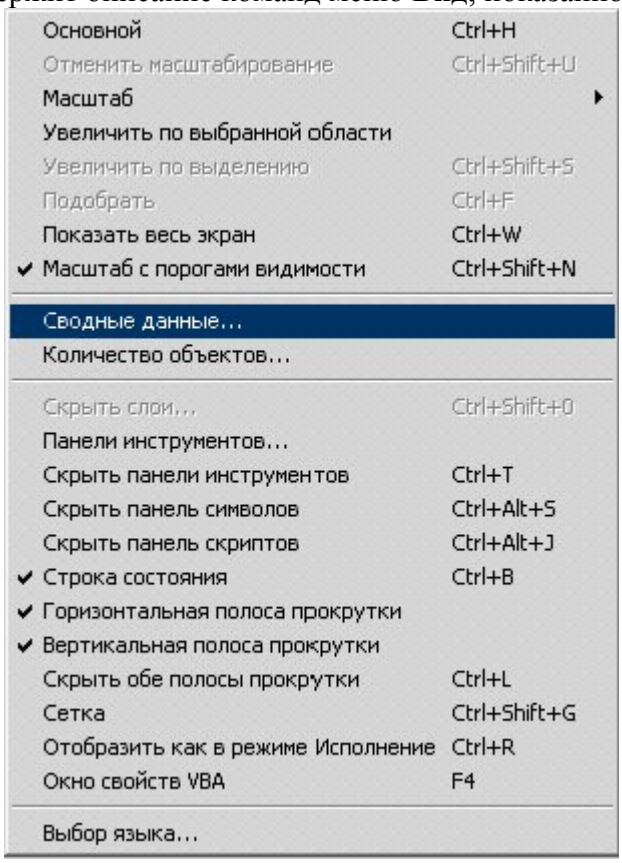
6.1 Введение
6.2 Меню Вид
6.3 Команда Основной
6.4 Функции масштабирования
6.5 Сводная информация
6.6 Вывод сводной информации об объектах экранной формы
6.7 Скрыть слои
6.8 Команда Панели инструментов
6.9 Отображение текущих координат курсора в строке состояния
6.10 Прокрутка изображения
6.11 Сетка
6.12 Свойства окна в режиме Исполнение
6.13 Окно свойств VBA
6.14 Выбор языка

6.1 Введение

Меню **Вид** GraphWorX32 представляет различные способы просмотра файлов экранных форм. Имеется возможность изменения масштаба экранной формы, использования сетки с различным шагом для привязки графических объектов.

6.2 Меню Вид

Настоящий раздел содержит описание команд меню **Вид**, показанного на рисунке.



*Меню **Вид***

6.3 Команда Основной

Команда **Основной** меню **Вид** позволяет установить масштаб экранной формы, при котором была выполнена последняя операция сохранения экранной формы.

Для восстановления основного вида экранной формы

Выберите команду **Основной** меню **Вид**. Масштаб экранной формы станет таким, каким он был при последнем ее сохранении. Значение масштаба выводится в правой части строки состояния.

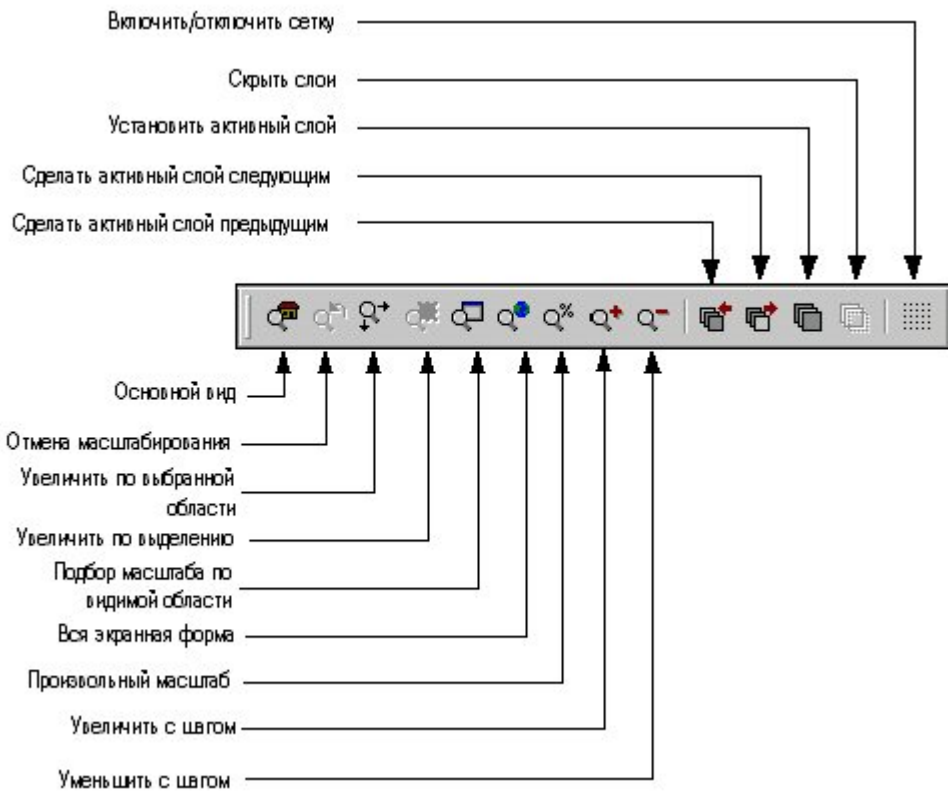
6.4 Функции масштабирования.

Значение масштаба в процентах отображается в правой части строки состояния. По умолчанию масштаб экранной формы равен 100%, а одна логическая единица координат равна одному пикселу.

Подменю **Масштаб**

Для увеличения или уменьшения размеров экранной формы в логических единицах выберите требуемое значение масштаба в подменю **Масштаб** меню **Вид**. Для установки произвольного масштаба выберите строку **Произвольный** подменю **Масштаб** и в диалоговой панели **Произвольный масштаб** введите требуемое значение масштаба в процентах.

Операции подменю **Масштаб** дублированы в панели инструментов *Вид*, внешний вид и назначение кнопок которой показаны на рисунке.



*Инструментальная панель **Вид**. Снизу расположены описания кнопок, дублирующих действия подменю **Масштаб** меню **Вид***

Отмена масштабирования

Имеется возможность отмены последовательно выполненных операций изменения масштаба путем выбора строки **Отменить масштабирование** меню **Вид** или нажатием кнопки  панели инструментов *Вид* требуемое количество раз.

Увеличение по выбранной области

Для увеличения фрагмента выделенной области экранной формы

1. Выберите строку **Увеличить по выделенной области** меню **Вид** или нажмите кнопку  панели инструментов *Вид*. Курсор мыши примет вид лупы.
2. Поместите курсор в точку, начиная от которой требуется начать выделение области, подлежащей увеличению, после чего, удерживая левую клавишу мыши в нажатом состоянии, переместите курсор в противоположную по диагонали от начальной точку выделяемой прямоугольной области увеличения. Отпустите левую клавишу мыши. Масштаб экранной формы будет увеличен по выделенной области.
3. Иным способом изменения масштаба экранной формы в данном случае является последовательные однократные щелчки правой клавишей мыши для последовательного уменьшения масштаба с шагом 50%. Последовательные щелчки левой клавишей мыши приводят к увеличению масштаба с шагом 50%.
4. По завершении операций редактирования графических объектов в требуемой области, по которой было выполнено увеличение масштаба, выберите строку **Основной** меню **Вид** или нажмите кнопку  панели инструментов *Вид* для возврата к первоначальному масштабу экранной формы.

Увеличение по выделению



Данная команда позволяет увеличить масштаб экранной формы по области, которая охватывает все выделенные графические объекты.

Подбор масштаба по области, охватывающей все объекты экранной формы

Для установки масштаба экранной формы таким образом, чтобы в ее видимую область поместились все графические объекты, выберите строку **Подобрать** меню **Вид**. Или нажмите кнопку  панели инструментов *Вид.*

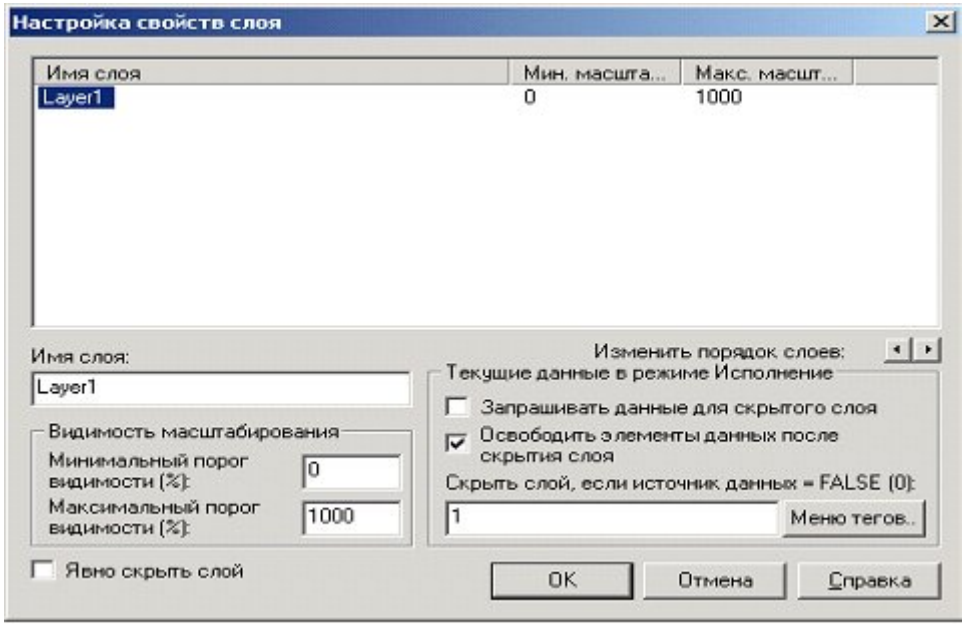
Подбор масштаба по размерам экранной формы



Для установки масштаба экранной формы таким образом, чтобы размеры видимой области совпадали с размерами экранной формы, выберите строку **Показать весь экран** меню **Вид**.

Масштаб с порогами видимости.

Пункт **Масштаб с порогами видимости** из меню **Вид** дает возможность скрывать слои, когда установленный масштаб выходит за границы, установленные для масштаба данного слоя. Для использования этой функции следует выбрать **Формат – Слои – Изменить свойства слоя** и в открывшемся диалоговом окне определить пороги видимости масштабирования:

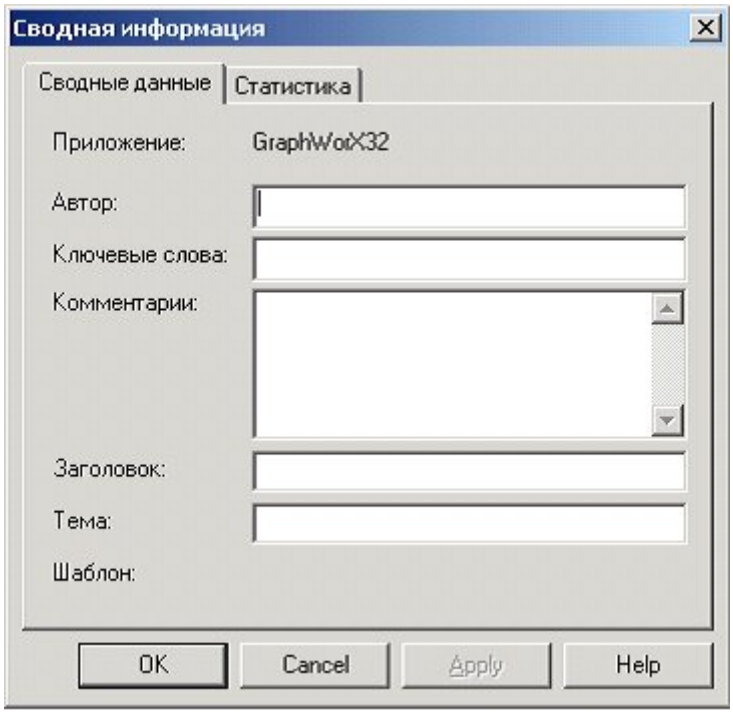


Подбор масштаба с помощью мыши.

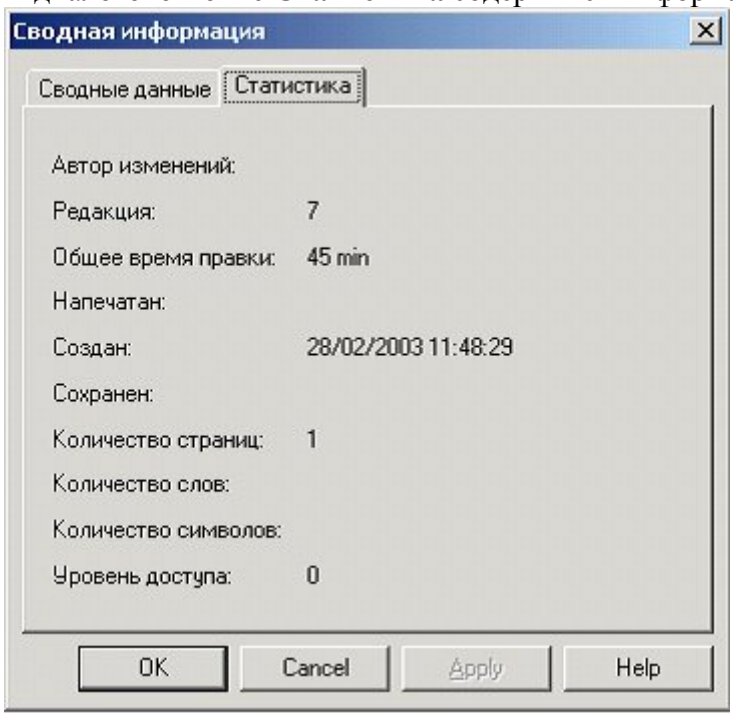
Увеличение изображения – удерживая клавишу ALT, вращать колесико мыши вперед.
Уменьшение изображения – удерживая клавишу ALT, вращать колесико мыши назад.
Прокрутка изображения - удерживая клавишу ALT, нажать левую клавишу мыши и перемещать мышь в нужном направлении.
Масштабирование зависит от выделения. Если объект выделен, при масштабировании изображение на экране будет автоматически смещаться так, чтобы объект располагался в центре. Это правило применимо и при выделении нескольких объектов.

6.5 Сводная информация.

Диалоговое окно **Сводной информации** открывается при выборе пункта **Сводные данные** из меню **Вид**.



Статистика.
В диалоговом окне **Статистика** содержится информация о текущей экранной форме.

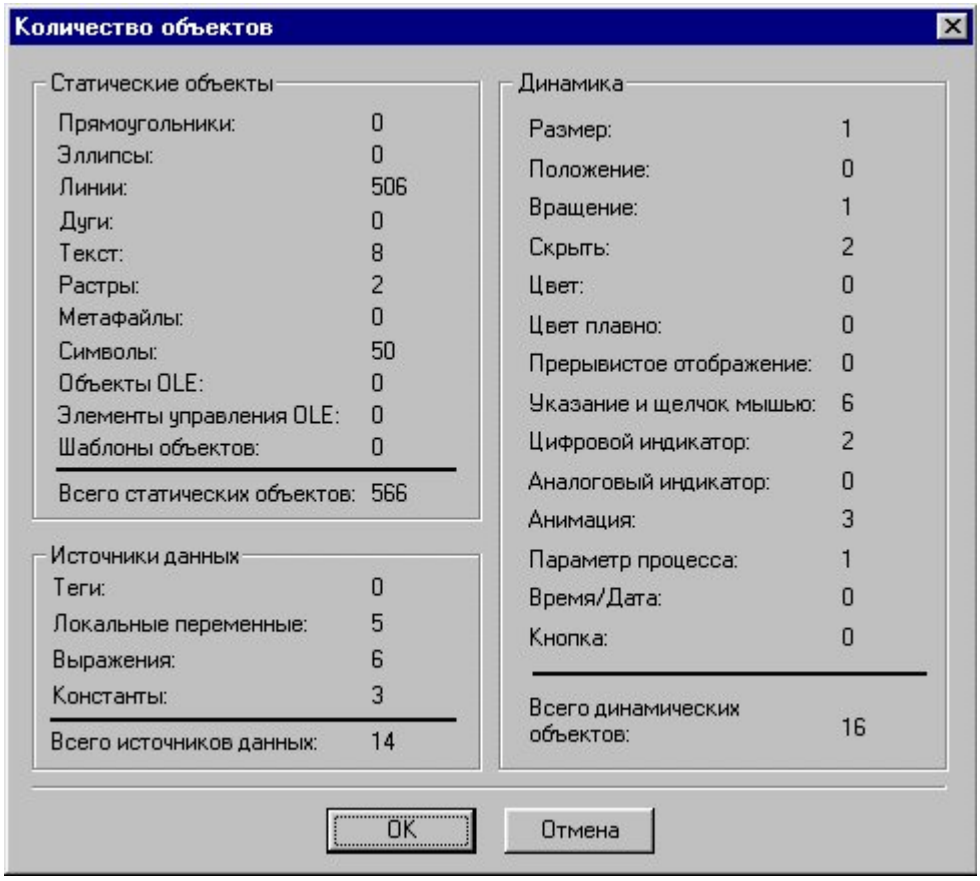


6.6 Вывод сводной информации об объектах экранной формы

Функция отображения сводных данных позволяет вывести на экран количество статических и динамических объектов, размеры и другую статистическую информацию об экранной форме.

Для отображения сводной информации

1. Выберите строку **Количество объектов** меню **Вид**. На экран монитора будет выведена диалоговая панель, содержащая статистику текущей экранной формы. Внешний вид диалоговой панели показан на рисунке.



*Диалоговая панель **Количество объектов***

2. Описание полей диалоговой панели приведено в табл. 6-1.

Таблица 6-1

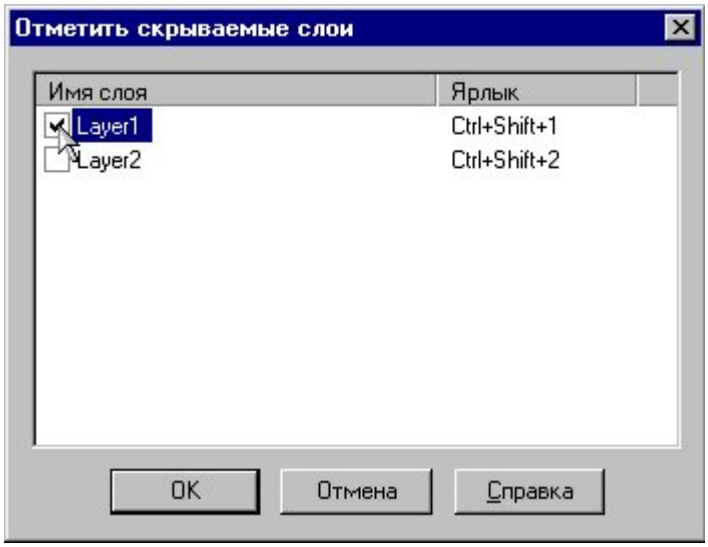
Поле	Описание
Статические объекты	
Прямоугольники	Количество графических объектов типа *Прямоугольник*.
Эллипсы	Количество графических объектов типа *Эллипс*.
Линии	Количество графических объектов типа *Линия*.
Дуги	Количество графических объектов типа *Дуга*.
Текст	Количество графических объектов типа *Текст*.
Растры	Количество импортированных растровых изображений.
Метафайлы	Количество импортированных векторных рисунков.
Символы	Количество графических объектов, сгруппированных из более простых графических примитивов или импортированных из библиотеки символов.
Объекты OLE	Количество внедренных объектов OLE.
Элементы управления OLE	Количество вставленных управляющих элементов ActiveX.
Шаблоны объектов	Количество шаблонных объектов.
Всего статических	Суммарное количество статических объектов.
Источники данных	
Теги	Количество тегов OPC, связанных с графическими объектами экранной формы.
Локальные переменные	Количество локальных переменных в экранной форме.
Выражения	Количество простых выражений, в качестве операндов которых используются другие источники данных, связанных с графическими объектами экранной формы.

Константы	Количество констант (неизменяемых величин), связанных с графическими объектами экранной формы.
Всего источников данных	Суммарное количество источников данных, связанных с экранной формой.
Динамика	
Размер	Количество действий по динамическому изменению размеров графических объектов в зависимости от значений связанных с ними источников данных.
Положение	Количество действий по динамическому изменению положения графических объектов в зависимости от значений связанных с ними источников данных.
Вращение	Количество действий по динамическому вращению графических объектов в зависимости от значений связанных с ними источников данных.
Скрыть	Количество действий по динамическому включению или выключению отображения графических объектов в зависимости от значений связанных с ними источников данных.
Цвет	Количество действий по динамическому дискретному изменению цвета графических объектов в зависимости от значений связанных с ними источников данных.
Цвет плавно	Количество действий по динамическому плавному изменению цвета графических объектов в зависимости от значений связанных с ними источников данных.
Прерывистое отображение	Количество действий по динамическому прерывистому отображению графических объектов в зависимости от значений связанных с ними источников данных.
Указание и щелчок мышью	Количество операций указания с последующим щелчком мышью, связанных с графическими объектами.
Цифровой индикатор	Количество динамических действий типа *Цифровой индикатор*, связанных с группами графических объектов.
Аналоговый индикатор	Количество динамических действий типа *Аналоговый индикатор*, связанных с группами графических объектов.
Анимация	Количество динамических действий типа *Анимация*, связанных с группами графических объектов.
Параметр процесса	Количество элементов отображения параметров контролируемого процесса в числовой форме.
Время и дата	Количество элементов отображения времени и даты.
Кнопка	Количество графических объектов типа *Кнопка*, *Флажок*, *Переключатель* и *Кнопка с экраном*.
Всего динамических объектов	Суммарное количество динамических объектов.

3. По окончании просмотра сводной информации об объектах экранной формы, нажмите кнопку **OK** диалоговой панели.

6.7 Скрыть слои.

Команда **Скрыть слои** меню **Вид** и соответствующая кнопка в инструментальной панели *Слои*, а также комбинация клавиш Ctrl+Shift+0 позволяют включить или отключить отображения слоев, входящих в экранную форму. При выборе данной команды на экран монитора будет выведена диалоговая панель **Отметить скрываемые слои**, показанная на рисунке



Диалоговая панель Скрыть слои

Для скрытия слоев следует отметить флажки, расположенные слева от имен слоев, и нажать кнопку **OK**.

Включение и отключение отображения отдельных слоев может выполняться нажатием комбинаций клавиш Ctrl+Shift+<номер_слоя>, где <номер_слоя> - порядковый номер (сверху вниз) слоя в списке слоев диалоговой панели **Изменить свойства слоя**.

Функциональность операций управления слоями по команде **Скрыть слои** меню **Вид** сохраняется в режиме *Исполнение*.

Кроме этого, скрыть слои можно с помощью настроек в диалоге **Настройка свойств слоя** (Формат – Слои – Изменить свойства слоя).

Поле **Скрыть слой, если источник данных = FALSE (0)** позволяет определить условие, при котором текущий редактируемый слой будет скрыт в режиме *Исполнение*. Кнопка **Меню тегов**, расположенная справа от поля **Скрыть слой, если источник данных = FALSE (0)**, предназначена для определения указанного условия.

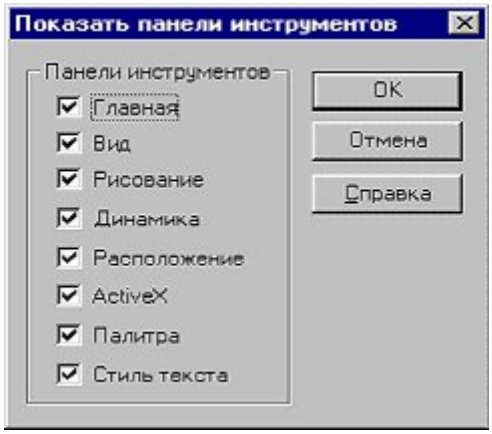
Если в меню, появляющемся при нажатии кнопки **Меню тегов**, выбрана строка **Теги OPC**, на экран монитора будет выведено окно **Универсального навигатора тегов.**

Если слой должен быть скрыт независимо от текущего масштаба экранной формы и условия, заданного в поле **Скрыть слой, если источник данных = FALSE (0)**, отметьте флажок **Явно скрыть слой**. В режиме *Исполнение* слой останется скрытым до тех пор, пока в режиме *Разработка* не будет снята отметка флажка **Явно скрыть слой.**

6.8 Команда Панели инструментов

Данная команда меню **Вид** позволяет включить или выключить отображение различных панелей инструментов GraphWorX32.

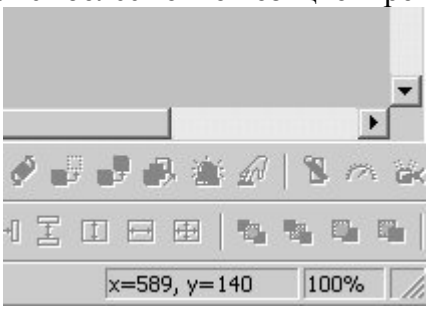
1. Выберите строку **Панели инструментов** меню **Вид**. На экран монитора будет выведена диалоговая панель **Показать панели инструментов**, внешний вид которой показан на рисунке.
2. Установите флажки слева от названий панелей инструментов, которые требуется вывести в служебных областях главного окна GraphWorX32. Для того, чтобы скрыть какие-либо панели инструментов, следует снять флажки слева от их названий в диалоговой панели **Показать панели инструментов**.
3. Нажмите кнопку **OK** диалоговой панели.



*Диалоговая панель **Показать панели инструментов***

6.9 Отображение текущих координат курсора в строке состояния

В правом нижнем углу строки состояния отображаются текущие координаты курсора мыши. Данные координаты представляют положение курсора по горизонтальной и вертикальной осям, что позволяет более точно позиционировать курсор.

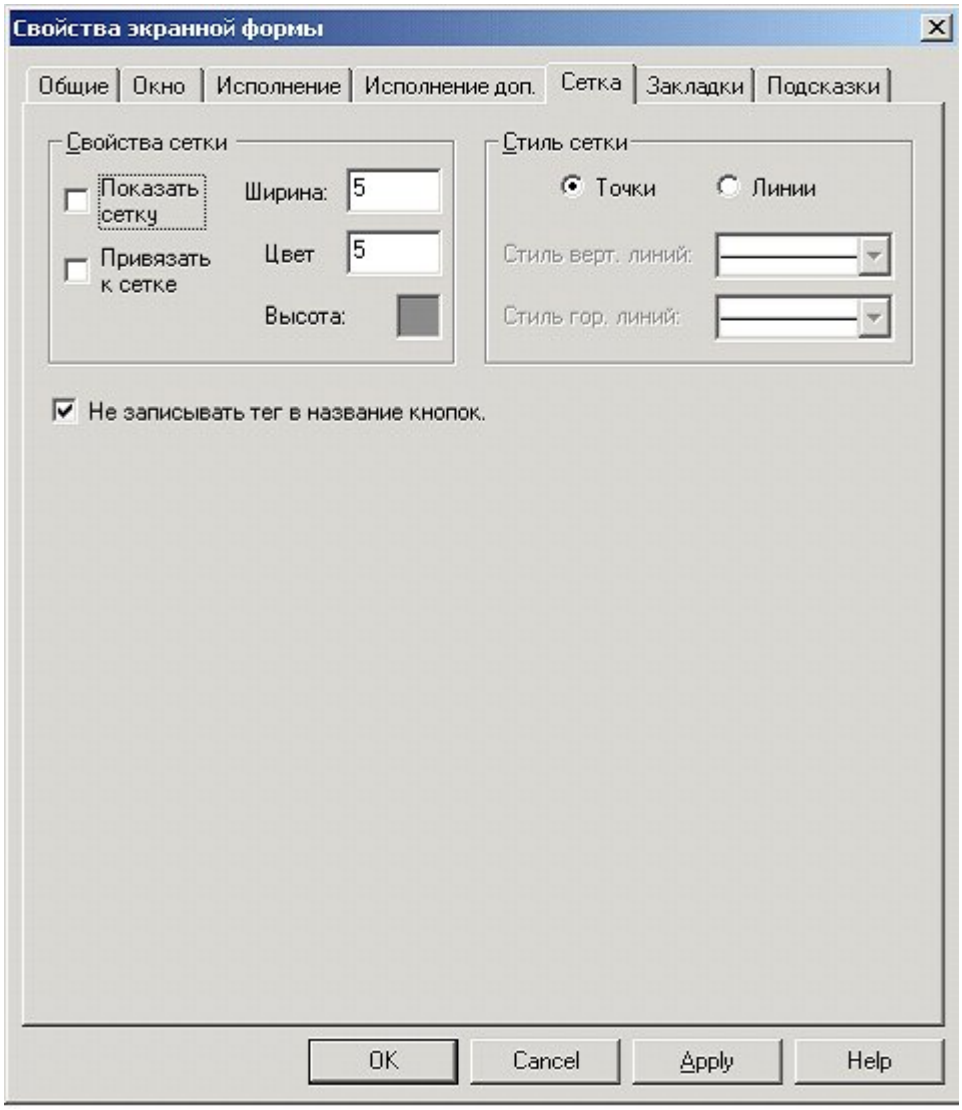


6.10 Прокрутка изображения

Данная функция позволяет перемещать текущую видимую область экранной формы в горизонтальном и вертикальном направлениях при помощи горизонтальной и вертикальной полос прокрутки, которые отображаются соответственно в нижней и в правой частях рамки окна GraphWorX32. Строки **Горизонтальная полоса прокрутки** и **Вертикальная полоса прокрутки** предназначены для управления отображением соответствующих полос прокрутки.

6.11 Сетка

GraphWorX32 для удобства рисования графических объектов позволяет использовать сетку, настройка параметров которой осуществляется на странице свойств **Сетка** диалоговой панели **Параметры приложения и Свойства экранной формы**. Внешний вид страницы свойств показан на рисунке.



*Внешний вид страницы свойств **Сетка** диалоговой панели **Свойства экранной формы***

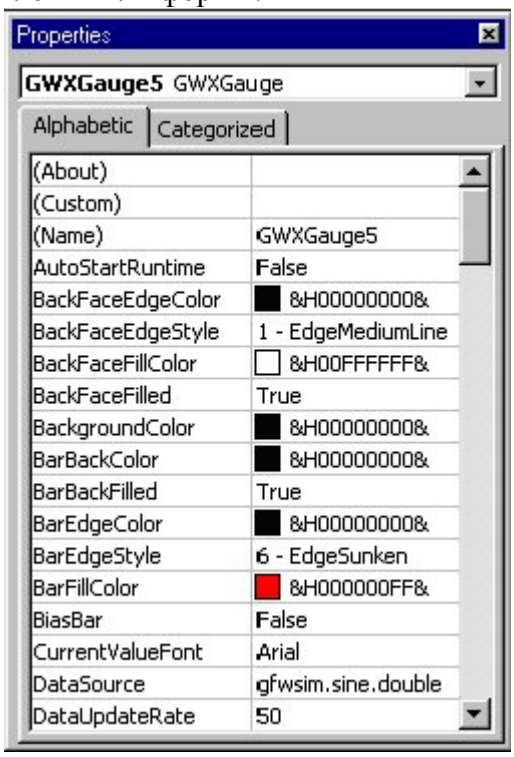
Группа **Размеры сетки** позволяет выбрать горизонтальный и вертикальный шаг координатной сетки в базовых единицах (в зависимости от текущего режима масштабирования). Когда выбран масштаб 100%, размер единицы шага сетки равен одному пикселу.
Поле **Цвет сетки** предназначено для выбора цвета, который будет применен к линиям сетки.
Группа параметров **Стиль сетки** позволяет установить внешний вид сетки: в виде точек или линий, расположенных в видимой области экранной формы.
Для привязки создаваемых графических объектов к сетке следует установить флажок **Привязать к сетке**.
Флажок **Показать сетку** позволяет включить или отключить отображение линий сетки

6.12 Свойства окна в режиме Исполнение.

Выбор пункта **Отобразить как в режиме Исполнение** из меню **Вид** дает возможность просмотреть, как будет выглядеть экранная форма в режиме исполнения. Это позволяет правильно подобрать размеры экрана и масштаб. Если скрыты панели инструментов, доступ к функциям конфигурации можно получить из основного меню или из меню, открывающегося при щелчке правой кнопки мыши.
Примечание: Нажатие **CTRL+M** переключает экран между режимами разработки и исполнения.

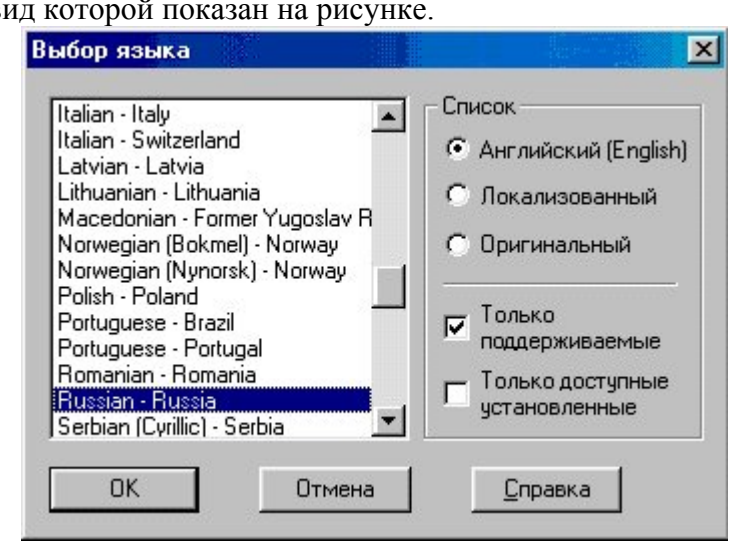
6.13 Окно свойств VBA.

Выбор пункта **Окно свойств VBA** из меню **Вид** открывает список свойств для выбранного элемента управления или формы.



6.14 Выбор языка

Данная функция обеспечивает возможность выбора языка интерфейса пользователя GraphWorX32, включая меню, диалоговые панели и сообщения. Для выбора языка интерфейса пользователя, отличного от текущего используемого, используйте команду **Выбор языка** меню **Вид**. На экран монитора будет выведена диалоговая панель **Выбор языка**, внешний вид которой показан на рисунке.



*Диалоговая панель **Выбор языка***

Выберите в списке название языка, который требуется установить для интерфейса пользователя GraphWorX32 и нажмите кнопку **OK** диалоговой панели. В зависимости от установок флажков и переключателей группы параметров **Список** диалоговой панели **Выбор языка** будет изменяться содержимое списка поддерживаемых языков.
Если установлен переключатель **Английский (English)**, то названия всех поддерживаемых языков в списке будут написаны по-английски.

При установке переключателя **Локализованный** названия всех языков будут написаны на языке, выбранном в качестве основного для Windows (в разделе **Язык и стандарты** *Панели управления* Windows).
Если установлен переключатель **Оригинальный**, то название каждого языка будет представлено на соответствующем языке (English – United Kingdom, Русский – Россия, italiano – Italia и т.д.).
Если установлен флажок **Только поддерживаемые**, то список будет содержать названия языков интерфейса пользователя, которые поддерживает GraphWorX32 (но не действительно установленные).
Если установлен флажок **Только доступные установленные**, то список будет содержать названия языков интерфейса пользователя, которые установлены в системе, а значит, могут выбраны.

Глава 7. Установление соединений с источниками данных

7.1 Введение
7.2 Теги
7.3 Универсальный навигатор
7.4 Редактор выражений
7.5 Локальные переменные GraphWorX32
7.6 Константы
7.7 Альтернативные имена (псевдонимы) связей с источниками данных

7.1 Введение

Настоящий раздел содержит информацию о способах установления связи между графическими объектами GraphWorX32 и источниками данных. Под источниками данных понимаются входные и выходные сигналы, передаваемые между GraphWorX32 и устройствами ввода-вывода автоматизированной системы управления технологическим процессом. Ниже приведено описание следующих основных понятий и операций:

- Установление соединений с источниками данных с помощью **Универсального навигатора данных**.
- Использование выражений
- Использование Констант и локальных переменных
- Использование технологии альтернативных имен связей с источниками данных (технологии псевдонимов).

Динамизация графических объектов основана на изменении их визуальных свойств в зависимости от значений связанных с этими свойствами источников данных. Каждому графическому объекту может быть поставлен в соответствие ряд зависимостей его визуальных свойств от значений нескольких технологических параметров или источников данных. При этом для некоторых зависимостей могут быть переопределены диапазоны изменения визуальных свойств, причем границы диапазонов изменения в свою очередь могут быть связаны с источниками данных. Источниками данных в GraphWorX32 могут быть, как указано выше, входные и выходные сигналы устройств ввода-вывода (*элементы данных в серверах OPC*), *локальные переменные* GraphWorX32, *константы*, *выражения*, операндами в которых могут быть другие источники данных, а также альтернативные (переопределяемые) имена источников данных, далее называемые *псевдонимами*.
Имена связей с источниками данных могут быть непосредственно введены или перенесены в поле **Источник данных** в страницах динамических свойств графических объектов, которые представлены в **Инспекторе свойств**, как показано на рисунке 7-1.

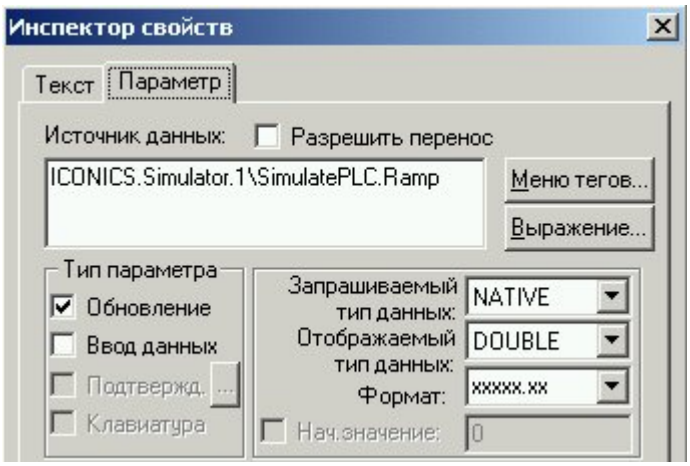


*Рис. 7-1. Поле **Источник данных** в **Инспекторе свойств***

Справа от поля **Источник данных** расположены кнопки **Меню тегов** и **Выражение**. Первая кнопка предназначена для вызова **Универсального навигатора данных**. Вторая кнопка предназначена для вызова **Редактора выражений**.
Для более подробной информации, просьба обратиться к документации Unified_Data_Browser и Expression_Editor.

7.2 Теги

Тегом называется *текстовая строка* (описатель), уникальным образом идентифицирующая структуру данных, которая связана с одним, входным, выходным или внутренним, сигналом в системе сбора данных и управления.
Любая текстовая строка, определяющая источник данных и не являющаяся числовым значением, а также не обозначенная специальными символами идентификации, описание которых приведено ниже, интерпретируется в GraphWorX32 в качестве тега.
Если требуется использовать имена тегов, целиком состоящие из цифр, следует применять следующий синтаксис:
{{имя_тега}}
Все теги GraphWorX32, для которых не определены или отсутствуют реальные источники данных, в режиме *Исполнение* отмечаются как недействительные.

7.3 Навигатор тегов

Щелчок левой клавишей мыши на кнопке **Меню тегов** в Инспекторе свойств открывает Универсальный навигатор данных, внешний вид которого показан на рис.7-3.
Универсальный навигатор данных позволяет выбрать теги OPC и связи с базами данных. (Для более подробной информации, просьба обратиться к документации Unified_Data_Browser).

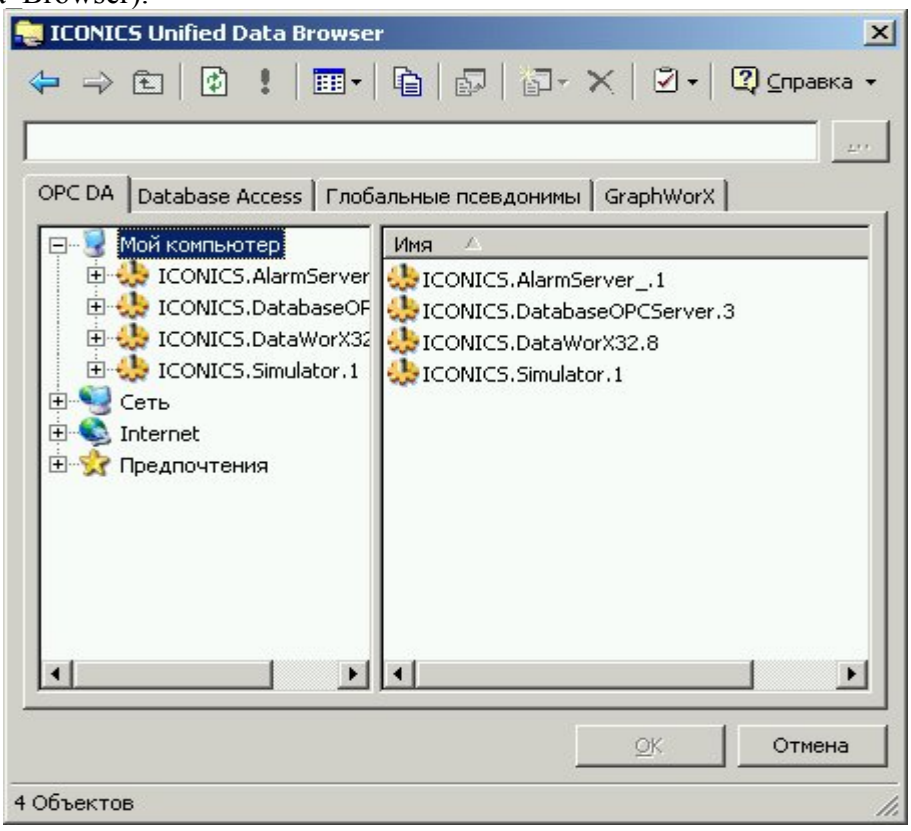
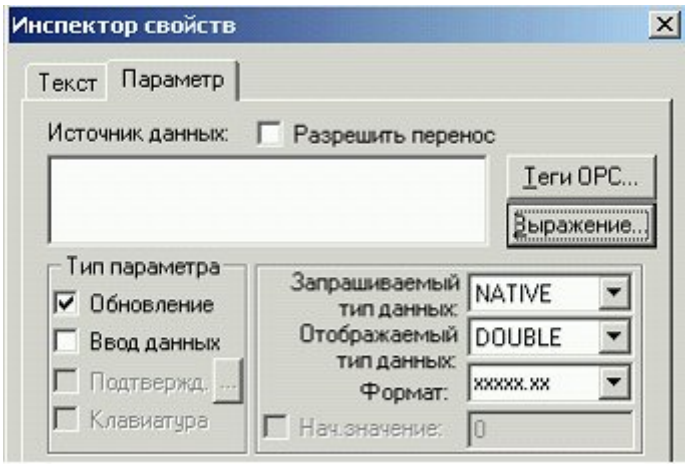


Рис. 7-3. ***Универсальный навигатор данных***

7.4 Редактор выражений

Доступ к категориям выражений может быть получен в **Редакторе выражений**, внешний вид которого показан на рис. 7-4. Для вызова **Редактора выражений** нажмите кнопку **Выражение** в Инспекторе свойств.



Благодаря возможности изменения размера окна Редактора Выражений, можно выбрать наиболее удобный для пользователя вид просмотра выражений. Выпадающий список для поля Выражение позволяет выбрать одно из 50 последних введенных выражений. Первым в выпадающем списке будет наиболее часто употребляемое выражение.

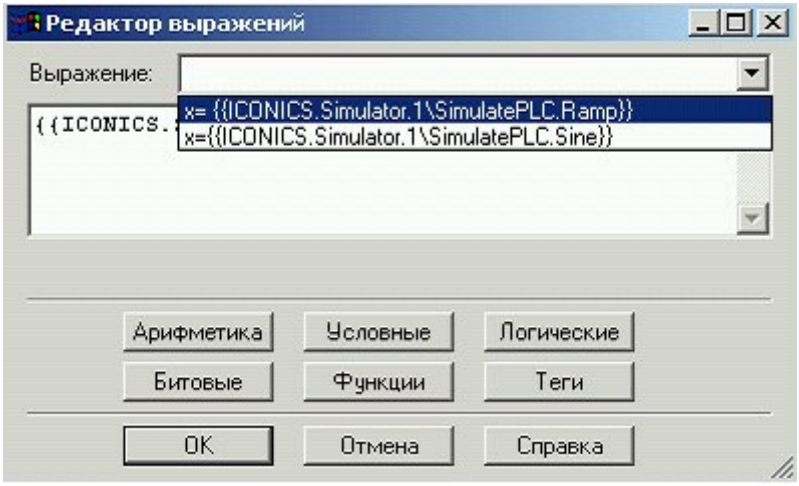


Рис. 7-2. Редактор выражений

Примечание: Для более подробной информации о Редакторе выражений, обратитесь к Справочной документации Expression_Editor.

7.5 Локальные переменные GraphWorX32

Локальной называется такая переменная GraphWorX32, непосредственный доступ к значению которой возможен только в пределах экранной формы, в которой она объявлена. Значения локальных переменных GraphWorX32 не могут передаваться в другие экранные формы или в другие приложения GENESIS32, включая управляющий элемент ActiveX просмотра графиков, без принятия специальных мер. Наиболее простой способ передачи значений локальных переменных другим приложениям GENESIS32 состоит в

использовании *Сервера вторичной обработки данных DataWorX32*. Более подробная информация приведена в руководстве пользователя DataWorX32.
Локальные переменные GraphWorX32 обозначаются следующим образом:
~~имя_локальной_переменной~~
Как правило, локальные переменные используются в качестве источников данных для различных анимационных эффектов в экранных формах GraphWorX32.
Редактирование локальных переменных выполняется в соответствующей диалоговой панели, для вызова которой следует выбрать строку **Локальные переменные** меню **Динамика**. Диалоговая панель Редактирования локальных переменных показана на рис. 7-10.

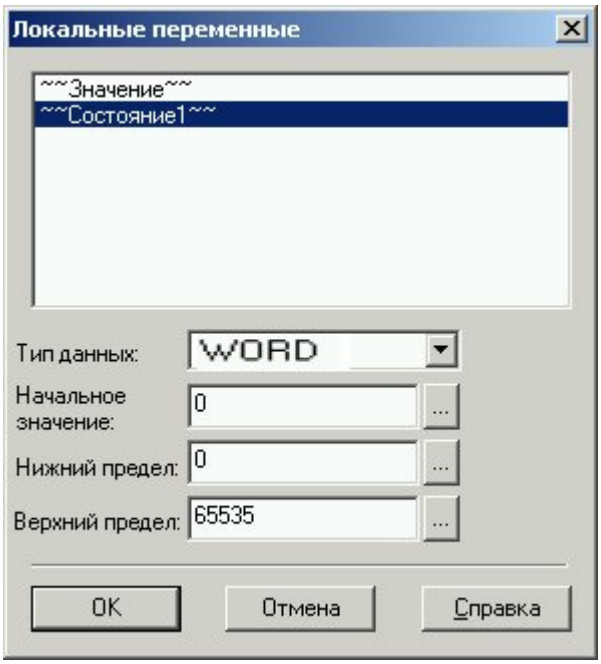


Рис.7-10. Диалоговая панель Редактирования локальных переменных

Поле **Тип данных** диалоговой панели позволяет установить тип значения выделенной в списке локальной переменной.
Поле **Начальное значение** предназначено для ввода начального значения выбранной в списке локальной переменной.
Поля **Нижний** и **Верхний предел** предназначены для ввода нижнего и верхнего пределов диапазона изменения выбранной локальной переменной.

GraphWorX32 содержит ряд встроенных локальных переменных для моделирования. Список этих переменных можно вызвать по кнопке **Меню тегов**... (открытие Универсального навигатора данных) в Инспекторе свойств. В закладке GraphWorX нужно выбрать **Переменные для моделирования**. Список возможных переменных, имитирующих сигналы, показан на рис. 7-11. Переменные для моделирования могут быть очень полезны при тестировании экранной анимации в случае, когда OPC-сервер не доступен.

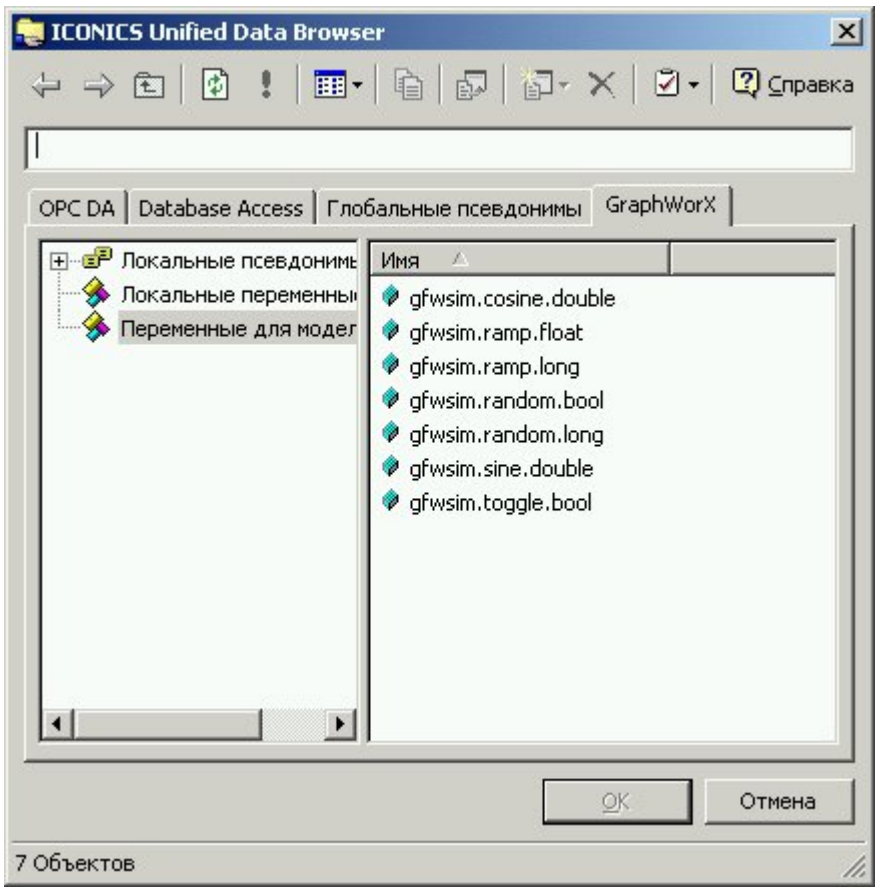


Рис. 7-11. Переменные для моделирования

Переменные для моделирования имитируют сигналы, изменяющиеся по следующим законам:

Gfwsim.cosine.double	Сигнал косинуса, изменяющийся в диапазоне от –1 до 1. Тип значения переменной – *длинное целое*.
Gfwsim.ramp.float	Сигнал пилообразной формы, изменяющийся в диапазоне от 0 до 1000. Тип значения переменной – *действительное* (float).
Gfwsim.ramp.long	Сигнал пилообразной формы, изменяющийся в диапазоне от 0 до 1000. Тип значения переменной – *длинное целое*.
Gfwsim.random.bool	Случайный сигнал, принимающий значения 0 и 1. Тип значения переменной – *булево*.
Gfwsim.random.long	Случайный сигнал, изменяющийся в диапазоне от 0 до 1000. Тип значения переменной – *длинное целое*.
Gfwsim.sine.double	Синусоидальный сигнал, изменяющийся в диапазоне от –1 до 1. Тип значения переменной – *длинное целое*.
Gfwsim.toggle.bool	Периодический сигнал, принимающий значения 0 и 1 с периодом около 3 с. Тип значения переменной – *булево*.

7.6 Константы

Константой в GraphWorX32 является число или строка, которые остаются неизменными в режиме *Исполнение*.

GraphWorX32 автоматически определяет, что введенная строка, идентифицирующая источник данных, является числовым значением, и интерпретирует такую строку, как константу. При необходимости ввода строковой константы следует использовать следующий синтаксис:

$”строковая_константа”$

Применение данного синтаксиса приводит к тому, что GraphWorX32 не воспринимает введенную строку как имя тега.

7.7 Альтернативные имена (псевдонимы) связей с источниками данных

Общие сведения

Технология *псевдонимов* основывается на том, что для представления одной строки, являющейся частью или целым именем тега, используется другая, более короткая строка. Применение псевдонимов позволяет изменять связи объектов с различными физическими источниками данных в режимах *Разработка* и *Исполнение*.
Команда **Псевдонимы**, вызывающая редактор альтернативных имен, находится в меню **Динамика** GraphWorX32. Вызов редактора альтернативных имен для выделенного объекта может также осуществляться выбором команды **Псевдонимы** контекстного меню, которое появляется после щелчка правой клавишей мыши на графическом объекте.
При вводе альтернативного имени в поле **Источник данных** в **Инспекторе свойств** для какого-либо динамического действия с графическим объектом, следует использовать специальные символы обозначения псевдонима:
<<имя_псевдонима>>
Символы ‘<<’ и ‘>>’ служат для идентификации псевдонима. Например:
[opcserver].<<tank>>.out
В данном случае ‘tank’ является альтернативным именем.

Псевдонимы уровня объектов

Каждый объект, который связан с источниками данных, может иметь собственную таблицу псевдонимов. Указанная таблица создается автоматически, как только в поля **Источник данных** в **Инспекторе свойств** объекта вводятся альтернативные имена. Записи в таблице содержат все альтернативные имена источников данных, связанных с объектом. По умолчанию *определения* (то, вместо чего используется псевдонимы) всех альтернативных имен совпадают с самими альтернативными именами.
Подстановка определений для псевдонимов не производится до тех пор, пока определения совпадают с альтернативными именами (псевдоним указывает сам на себя).
Изменение определений псевдонимов уровня объектов (подстановка строк, соответствующих именам или частям имен реальных источников данных) выполняется в соответствующей таблице с помощью редактора псевдонимов. Для редактирования псевдонимов уровня объектов в GraphWorX32 следует выделить один или несколько видимых объектов перед вызовом редактора псевдонимов. При выделении нескольких объектов все связанные с ними альтернативные имена объединяются в одну таблицу. Для идентификации принадлежности различных псевдонимов различным объектам следует использовать имена объектов, которые вводятся в основной странице **Инспектора свойств** в поле **Имя объекта**. Если в экранной форме присутствует множество графических объектов, с которыми связаны псевдонимы, и ни один из объектов не выделен, при вызове редактора псевдонимов в таблицу будут помещены все псевдонимы всех объектов экранной формы.

Интерфейс пользователя таблицы редактирования псевдонимов

Таблица псевдонимов уровня объектов, позволяющая переопределять определения альтернативных имен, показана на рис 7-12.

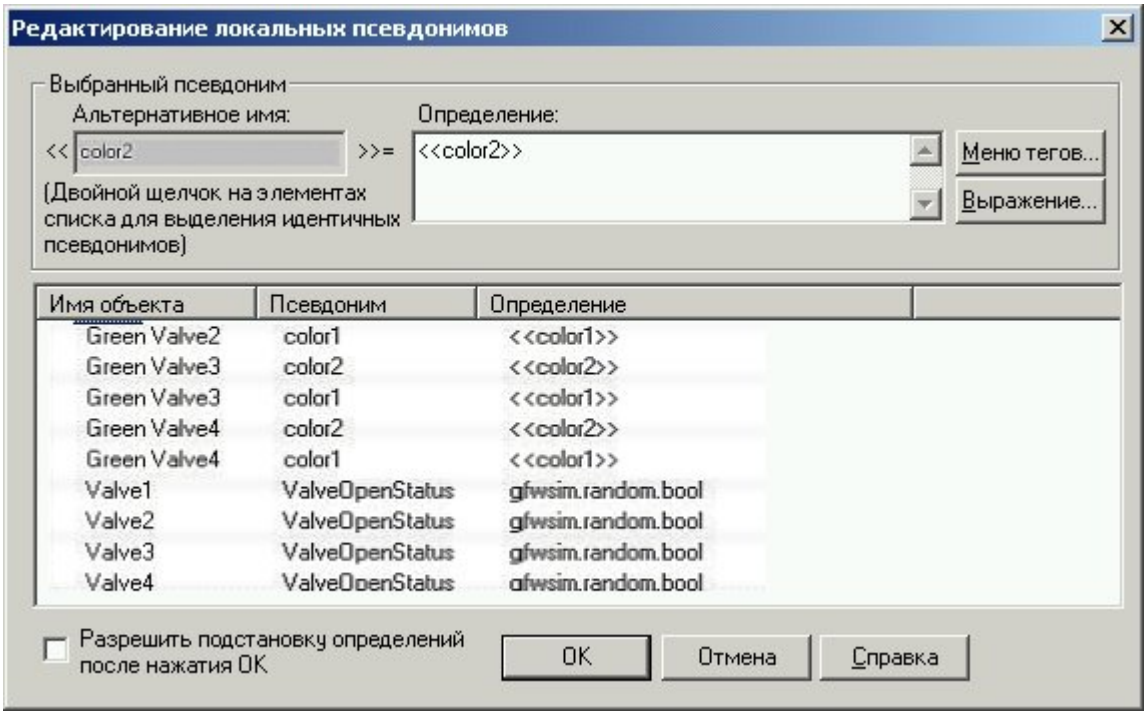


Рис. 7-12. Таблица псевдонимов

Для редактирования псевдонима:

1. Щелкните левой клавишей мыши на имени объекта, расположенного слева в строке альтернативного имени, которое требуется переопределить. Выбранный псевдоним появится в поле **Альтернативное имя**.
2. Введите строку определения псевдонима в поле **Определение**. Привязка к локальным переменным, результатам вычисления выражений или тегам OPC выполняется с помощью кнопок **Меню тегов** и **Выражение**.
3. Для того, чтобы подстановка определений была выполнена сразу после закрытия диалоговой панели **Редактирование локальных псевдонимов** нажатием кнопки **OK**, отметьте флажок **Разрешить подстановку определений после нажатия OK**. В противном случае подстановка будет выполнена после перевода GraphWorX32 в режим *Исполнение*.

Имеется возможность выделения множества элементов списка и одновременного определения для всех выделенных псевдонимов одного источника данных. Для выделения нескольких одинаковых альтернативных имен следует выполнить двойной щелчок левой клавишей мыши на имени объекта, связанного с одним из них.
Описание полей диалоговой панели **Редактирование локальных псевдонимов** приведено в табл. 7-1.

Таблица 7-1

Поле	Описание
Определение	Содержит строку, идентифицирующую имя реального источника данных, на которую указывает псевдоним. Начиная с версии 6.0, данное поле может содержать другой псевдоним, тем самым обеспечивая возможность использования псевдонимов второго уровня.
Кнопка **Меню тегов**	Предназначена для вызова **Универсального навигатора данных.**
Кнопка **Выражение**	Предназначена для вызова **Редактора выражений.**
Флажок **Разрешить подстановку определений после нажатия OK**	Отмеченное состояние приводит к тому, что подстановка определений псевдонимов выполняется сразу после нажатия кнопки **OK** диалоговой панели. В противном случае подстановка выполняется только после перевода GraphWorX32 в режим *Исполнение*.

Примечание. Строки, идентифицирующие имена реальных источников данных, на которые указывают псевдонимы, могут заменяться в режиме *Исполнение* из других

приложений Windows, а также из сценарных процедур VBA через соответствующие методы OLE Automation.

Замена определений псевдонимов в режиме *Исполнение*

Имеется несколько способов переопределения псевдонимов экранной формы в режиме *Исполнение*. Заданный набор псевдонимов может быть переопределен при выполнении одного действия типа *Указание и щелчок*. Например, при использовании данного действия для загрузки экранной формы, в момент загрузки указанной экранной формы может быть произведено переопределение заданного набора псевдонимов. Причем переопределение будет выполнено до того, как экранная форма обратится за значениями к реальным источникам данных. Переопределение псевдонимов может выполняться непосредственно в GraphWorX32 или из текстового файла, в котором альтернативные имена и их определения расположены в два столбца и разделены символами табуляции.
Действия, с которыми может быть связана операция переопределения псевдонимов, включают:

- *Указание и щелчок – Загрузка экранной формы:* переопределение псевдонимов во вновь загружаемой экранной форме.
- *Указание и щелчок – Загрузка экранной формы в стиле Перенести и Оставить*: переопределение псевдонимов в переносимой экранной форме.
- *Указание и щелчок – Всплывающее окно:* переопределение псевдонимов во всплывающем окне экранной формы.
- *Указание и щелчок – Встроенное окно:* переопределение псевдонимов во встроенном окне экранной формы.
- *Указание и щелчок – Настройка псевдонимов:* переопределение псевдонимов для отдельных объектов или всей текущей экранной формы.
- *Указание и щелчок – Диалог настройки псевдонимов:* переопределение псевдонимов оператором в диалоговой панели, содержащей списки возможных определений псевдонимов для отдельных объектов или всей текущей экранной формы.
- Загрузка экранной формы по щелчку на экранных закладках, устанавливаемых в диалоговых панелях **Свойства экранной формы** и **Параметры приложения**: переопределение псевдонимов во вновь загружаемой экранной форме.
- Запуск GraphWorX32 из командной строки: текстовый файл, содержащий определения псевдонимов может быть использован в качестве одного из параметров запуска.

Для настройки переопределения псевдонимов в режиме *Исполнение* одним из перечисленных выше способов следует выполнить типовые операции, которые будут рассмотрены на примере действия *Указание и щелчок – Загрузка экранной формы*. Страница **Инспектора свойств**, относящаяся к действию *Указание и щелчок – Загрузка экранной формы*, показана на рис. 7-13.

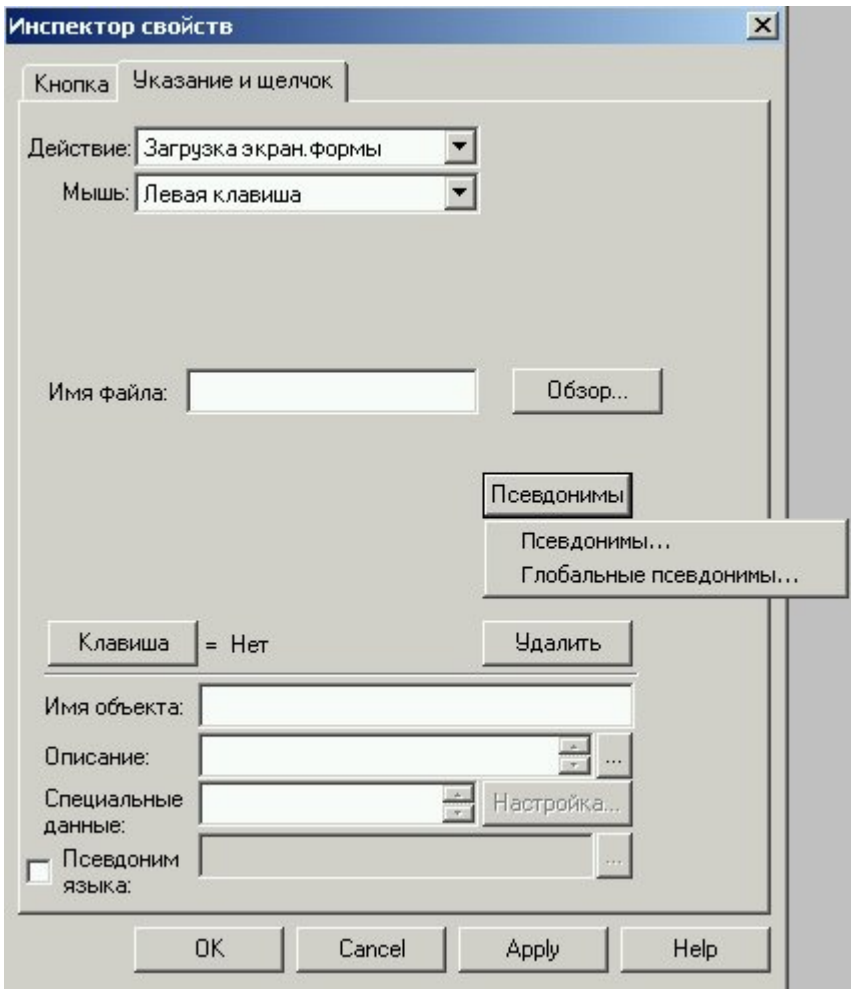


Рис. 7-13. Страница свойств ***Указание и щелчок*** *– Загрузка экранной формы*

Нажмите кнопку **Псевдонимы**. На экран монитора будет выведена диалоговая панель **Настройка конфигурации псевдонимов**, внешний вид которой показан на рис. 7-14.

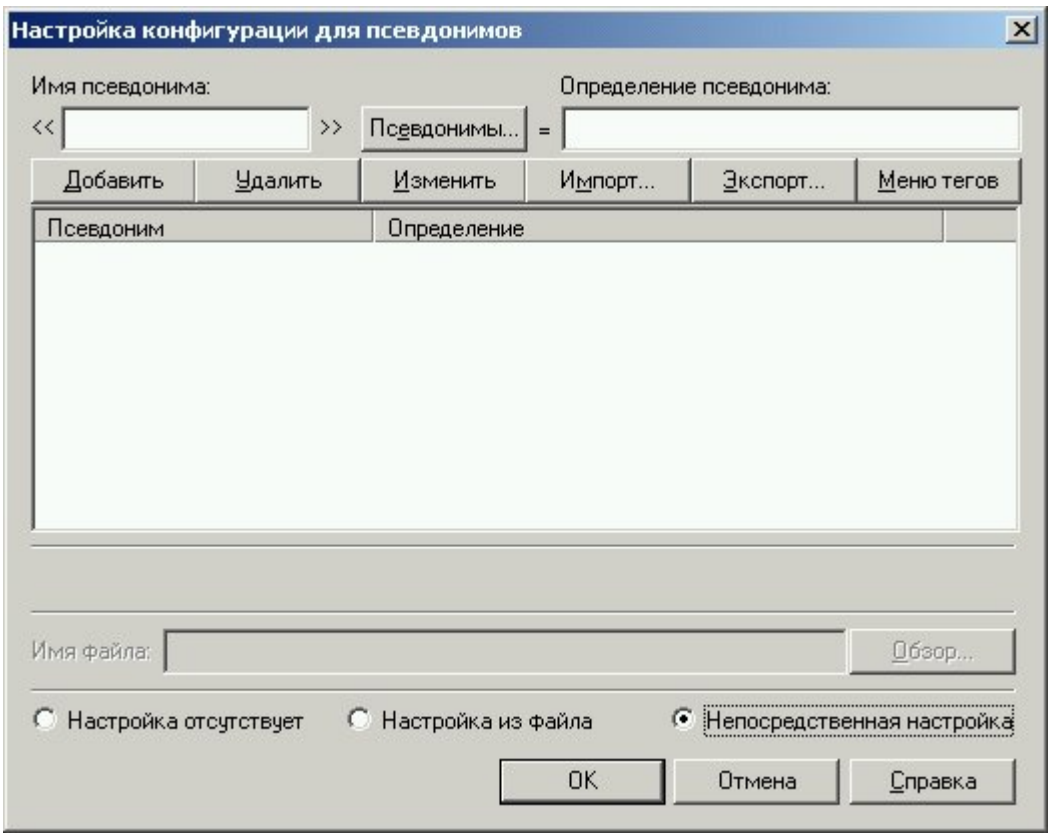


Рис. 7-14. Диалоговая панель ***Настройка конфигурации для псевдонимов***

Описание полей диалоговой панели приведено в табл. 7-2.

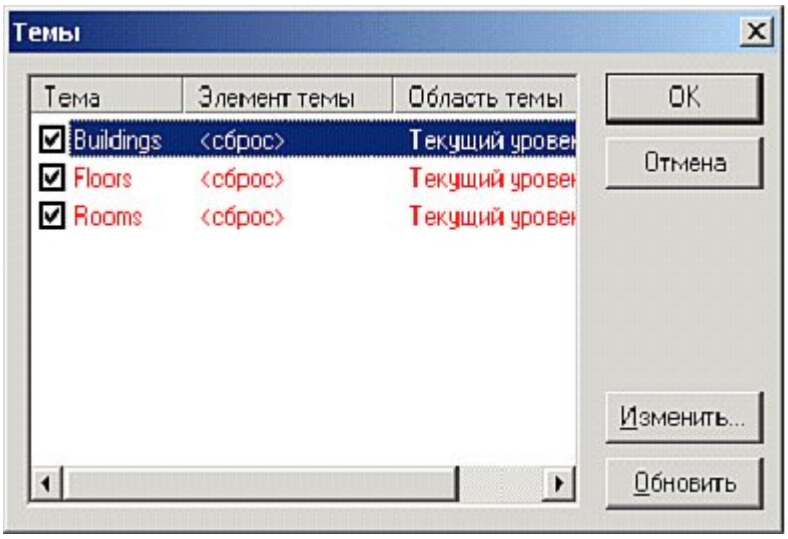
Таблица 7-2

Поле	Описание
Определение	Содержит строку, идентифицирующую имя реального источника данных, на которую указывает псевдоним. Начиная с версии 6.0, данное поле может содержать другой псевдоним, тем самым обеспечивая возможность использования псевдонимов второго уровня.
Кнопка **Меню тегов**	Предназначена для вызова **Универсального навигатора данных и Редактора выражений.**
Кнопка **Псевдонимы**	Предназначена для вызова диалоговой панели **Псевдонимы**, содержащей полный список псевдонимов текущей экранной формы, а также обеспечивающей доступ к псевдонимам в других экранных формах.
Кнопка **Экспорт**	Предназначена для сохранения созданной конфигурации псевдонимов в текстовом файле.
Кнопка **Импорт**	Предназначена для загрузки конфигурации псевдонимов из текстового файла.
Кнопка **Изменить**	Предназначена для изменения определения текущего псевдонима, выбранного в списке.
Кнопка **Удалить**	Предназначена для удаления псевдонима, выбранного в списке.
Кнопка **Добавить**	Предназначена для добавления псевдонима, выбранного в диалоговой панели **Псевдонимы**, в текущую редактируемую конфигурацию.
Список	Содержит альтернативные имена и определения текущей конфигурации псевдонимов.
Поле **Имя файла**	Содержит путь и имя текстового файла, в котором содержится информация о привязке псевдонимов.
Переключатель с положениями:	
Настройка отсутствует	Положение по умолчанию.

Настройка из файла	Настройка псевдонимов при выполнении действия *Указание и щелчок* в режиме *Исполнение* осуществляется из файла, указанного в поле **Имя файла**.
Непосредственная настройка	Настройка псевдонимов при выполнении действия *Указание и щелчок* в режиме *Исполнение* осуществляется с использованием конфигурации псевдонимов, содержащейся в списке.

Примечание. Строки, идентифицирующие имена реальных источников данных, на которые указывают псевдонимы, могут заменяться в режиме *Исполнение* из других приложений Windows, а также из сценарных процедур VBA через соответствующие методы OLE Automation.

Можно также определить глобальные псевдонимы, нажав кнопку **Псевдонимы** в окне **Инспектор Свойств** и выбрать **Глобальные псевдонимы** из выпадающего меню.
В появившемся диалоговом окне **Темы** выбрать **тему глоб. псевдонимов**.

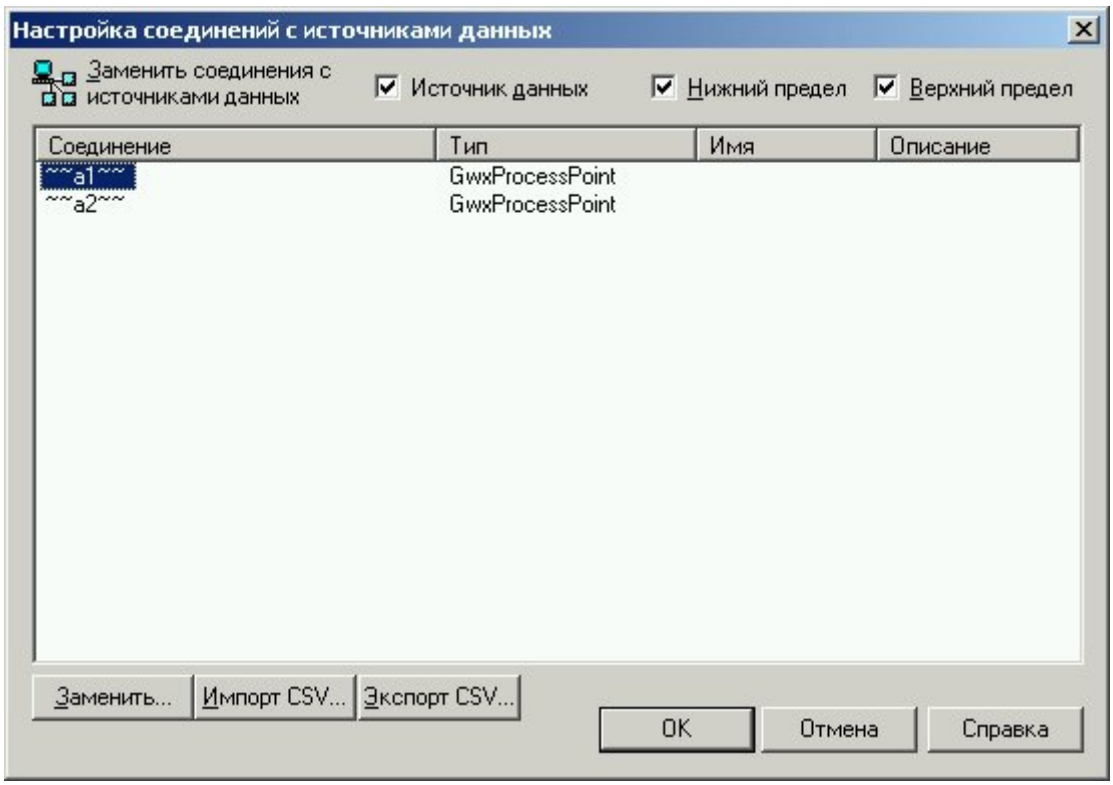


Примечание. Для редактирования свойств темы следует выделить её и нажать кнопку **Изменить.** Будет открыт **Редактор Тем**, где можно определить элементы и область темы. Доп. информация о редактировании тем гл.псевдонимов содержится в описании диалога **Параметры приложения/Исполнение доп.**

Псевдонимы второго уровня.

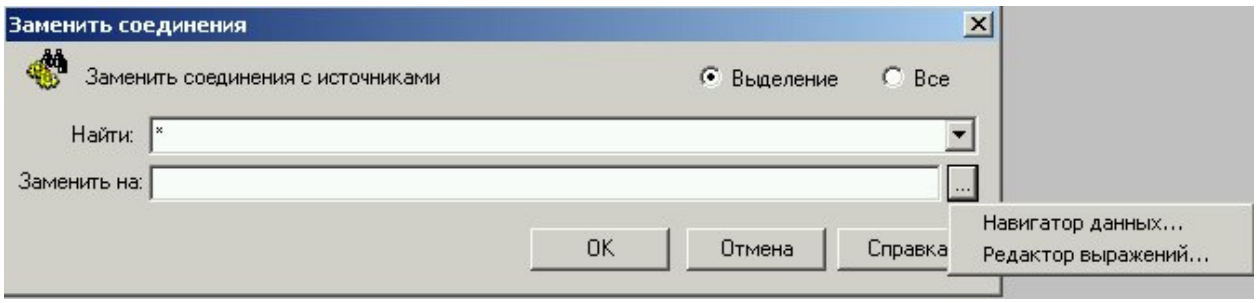
Допускаются «двухуровневые псевдонимы», т.е. в качестве определения псевдонима указывается другой псевдоним, который должен быть определен раньше выполнения команды установки первого псевдонима.

Настройка соединений с источниками данных.
При выборе **Динамика – Соединения** будет открыто диалоговое окно **Настройка соединений с источниками данных**.



Замена соединений.

Нажатие на кнопке **Заменить** открывает диалоговое окно, позволяющее запустить операцию поиска и замены для соединений и символов в текущей экранной форме. Так же, как для функции Поиска и Замены, здесь поддерживается использование групповых символов.



Импорт и экспорт.

Соединение элементов управления и отображения с тегами может быть выполнено автоматически путем импорта файла *.csv. Щелчок на кнопке **Импорт CSV** открывает обзор файлов для поиска файла *.csv. Существующие соединения могут быть экспортированы в файл *.csv по нажатию кнопки **Экспорт CSV**.

Глава 8. Динамизация объектов экранных форм.

8.1 Введение
8.2 Источник данных
8.3 Описание подменю Действия меню Динамика
8.4 Индикаторы
8.5 Встроенные динамические объекты
8.6 Копирование динамических действий, примененных к графическим объектам
8.7 Использование специальных данных
8.8 Конфигуратор подтверждающих сообщений
8.9 Доступ к данным и их визуализация

8.1 Введение

Настоящий раздел содержит указания по установлению одного или нескольких динамических соединений между графическими объектами и значениями тегов в серверах OPC. Визуальные свойства объектов после установления динамических соединений в режиме *Исполнение* будут изменяться в зависимости от значений связанных с ними тегов OPC. К одному графическому объекту может быть применено несколько *динамических действий*, которые являются потребителем информации от динамического соединения. Например, можно создать объект, цвет, размер и положение которого изменяются в зависимости от значения одного или нескольких тегов OPC.
Создание динамических соединений производится с помощью команд меню **Динамика**. Доступны следующие типы динамики:
Размер, *Положение/Движок*, *Вращение/Круговая шкала*, *Скрыть/Блокировать*, *Цвет*, *Плавное изменение цвета*, *Прерывистое отображение*, *Указание и щелчок мышью*, *Цифровой индикатор*, *Аналоговый индикатор*, *Анимация*, *Значение параметра*.

8.2 Источник данных

Страница **Инспектора свойств**, соответствующая динамическому соединению, создаваемому для графического объекта, имеет поле **Источник данных**, которое предназначено для ввода имени тега, значение которого будет определять характер динамического изменения визуальных свойств объекта. Описание поля **Источник данных** приведено в разделе 7 настоящего руководства.

8.3 Описание подменю Действия меню Динамика

Как указывалось выше, динамические действия позволяют изменять визуальные свойства графического объекта, к которому они применены, в зависимости от значений тегов OPC, имена которых содержатся в полях **Источник данных** соответствующих страниц **Инспектора свойств** объекта. Внешний вид подменю **Действия** меню **Динамика** показан на рис. 8-1.

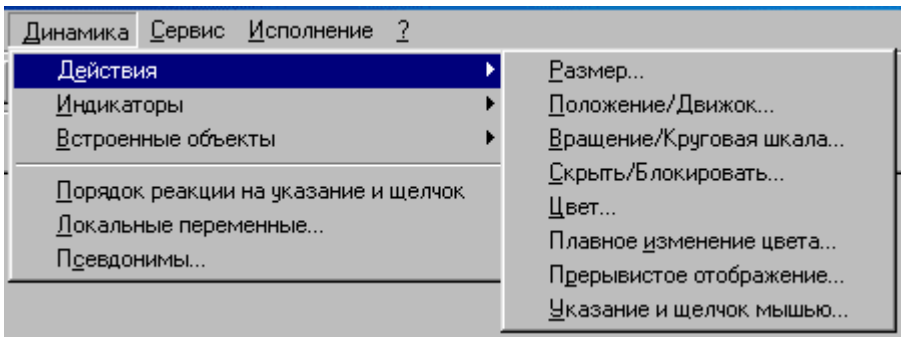


*Рис. 8-1. Подменю **Действия** меню **Динамика***

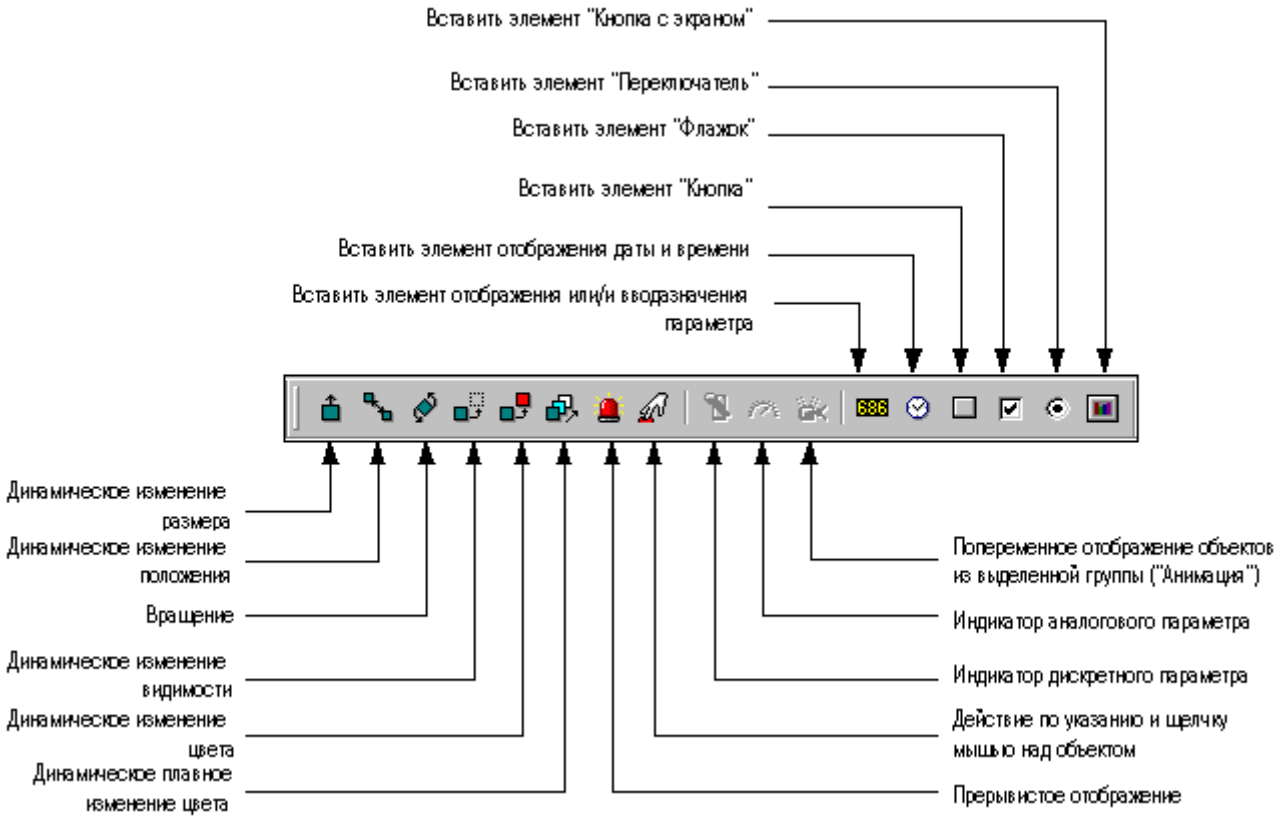


Рис. 8-2. Панель инструментов Динамика

Динамическое изменение размера



Инструмент *Динамическое изменение размера* (*Размер*) позволяет установить динамическое соединение объекта с источником данных, при котором один или несколько геометрических размеров графического объекта будут динамически изменяться в режиме *Исполнение* пропорционально значению связанного с ним источника данных. Имеется возможность переопределения диапазона изменения размеров объекта путем ввода соответствующих значений в процентах в поля **Начальный размер** и **Конечный размер** страницы **Размер** в **Инспекторе свойств**, показанной на рис. 8-3.
Кроме того, диапазон изменения значения источника данных может быть переопределен в соответствующих полях группы параметров **Переопределить диапазон** страницы свойств.
Для установления динамического соединения *Размер*:

1. Выделите графический объект
2. Выберите строку **Размер** подменю **Действия** меню **Динамика** или нажмите кнопку *Размер* набора инструментов *Динамика*. На экран монитора будет выведена

диалоговая панель **Инспектор свойств** с выбранной страницей **Размер**, внешний вид которой показан на рис. 8-3.

3. Введите имя тега, локальной переменной или псевдонима в поле **Источник данных** или воспользуйтесь Универсальным навигатором данных, нажав на кнопку **Меню тегов**.
4. Выберите тип изменения размера нажатием кнопки с изображением способа изменения размера.

Описание полей страницы **Размер** в **Инспекторе свойств** приведено в табл. 8-1.

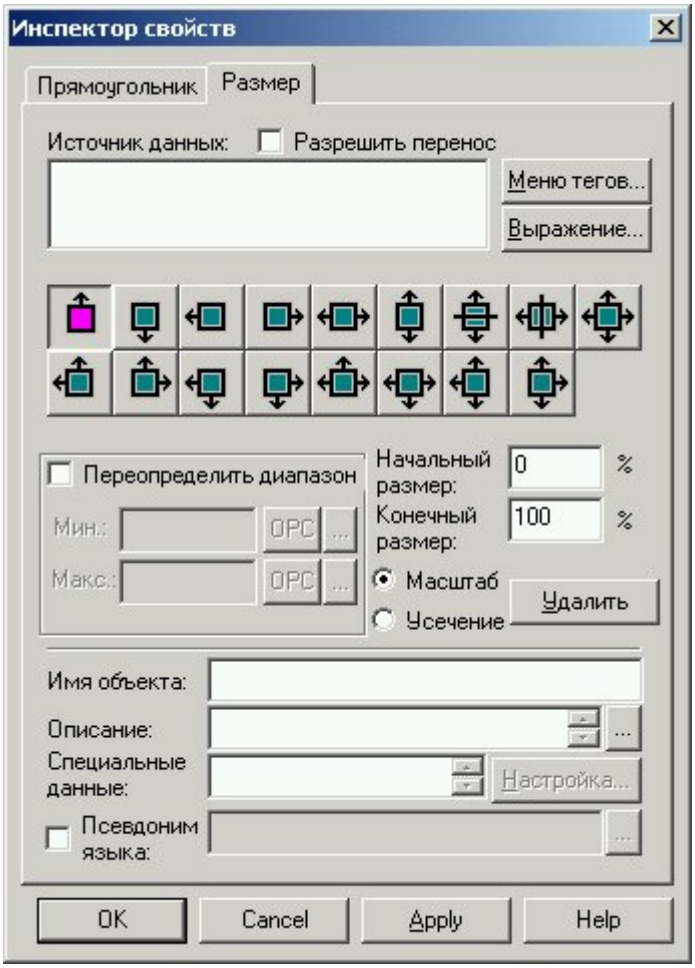


*Рис. 8-3. **Инспектор свойств** – страница **Размер***

Таблица 8-1

Поле	Описание
Источник данных	Данное поле предназначено для ввода имени тега или выражения, результат выполнения которого будет определять пропорциональное изменение одного или нескольких геометрических размеров графического объекта.
Меню тегов	Данная кнопка открывает Универсальный навигатор данных, позволяющий выбрать соединение с тегами OPC, переменными и псевдонимами. Для более подробной информации, обратитесь к Справочной документации Unified Data Browser.
Выражение	Данная кнопка открывает Редактор выражений. Для более подробной информации, обратитесь к Справочной документации Expression_Editor.
Набор кнопок выбора способа изменения размера	Данные кнопки предназначены для выбора типа изменения одного или нескольких геометрических размеров объекта.
	Изменение размера вдоль вертикальной оси. Начальному значению размера соответствует самая нижняя точка объекта.

Поле	Описание
	Изменение размера вдоль вертикальной оси. Начальному значению размера соответствует самая верхняя точка объекта.
	Изменение размера вдоль горизонтальной оси. Начальному значению размера соответствует точка объекта, расположенная правее остальных.
	Изменение размера вдоль горизонтальной оси. Начальному значению размера соответствует точка объекта, расположенная левее остальных.
	Изменение размера вдоль горизонтальной оси. Начальному значению размера соответствует горизонтальная координата экранной формы, расположенная в центре воображаемого прямоугольника, в который вписан объект.
	Изменение размера вдоль вертикальной оси. Начальному значению размера соответствует вертикальная координата экранной формы, расположенная в центре воображаемого прямоугольника, в который вписывается объект.
	Изменение размера вдоль вертикальной оси. При выполнении указанного динамического действия объект делится на две части воображаемой прямой линией, проведенной параллельно горизонтальной оси экранной формы через точку, расположенную в центре воображаемого прямоугольника, в который вписан объект. Увеличение значения связанного источника данных сначала приводит к уменьшению вертикального размера (высоты) объекта от самой нижней точки до координаты, расположенной в центре воображаемого прямоугольника, а затем, к увеличению вертикального размера от центра до самой верхней точки.
	Изменение размера вдоль горизонтальной оси. При выполнении указанного динамического действия объект делится на две части воображаемой прямой линией, проведенной параллельно вертикальной оси экранной формы через точку, расположенную в центре воображаемого прямоугольника, в который вписан объект. Увеличение значения связанного источника данных сначала приводит к уменьшению горизонтального размера (ширины) объекта слева до координаты, расположенной в центре воображаемого прямоугольника, а затем, к увеличению вертикального размера от центра вправо.
	Одновременное изменение высоты и ширины объекта в двух направлениях соответственно.
	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины и высоты справа налево и снизу вверх соответственно.
	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины и высоты слева направо и снизу вверх соответственно.
	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины и высоты справа налево и сверху вниз соответственно.
	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины и высоты слева направо и сверху вниз соответственно.
	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины в обоих направлениях и высоты снизу вверх соответственно.
	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины в обоих направлениях и высоты сверху вниз соответственно.
	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины справа налево и высоты сверху в обоих направлениях соответственно.
	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины слева направо и высоты сверху в обоих направлениях соответственно.

Поле	Описание
Переопределить диапазон	Установка данного флажка позволяет установить отличный от используемого по умолчанию диапазон значений связанного источника данных.
Мин.	После установки флажка **Переопределить диапазон** данное поле позволяет установить нижнюю границу значений связанного источника данных.
Макс.	После установки флажка **Переопределить диапазон** данное поле позволяет установить верхнюю границу значений связанного источника данных.
OPC	Кнопки, расположенные справа от полей **Мин**. и **Макс**., позволяют установить в качестве нижней и верхней границ источника данных значения тегов путем вызова их из **Универсального навигатора данных**.
…	Вызов Редактора выражений для полей **Мин**. и **Макс**.
Начальный размер	Данное поле предназначено для ввода начального значения (в процентах от текущих размеров), с которого начинается увеличение размера объекта при увеличении значения связанного источника данных в процессе выполнения назначенного действия *Размер*.
Конечный размер	Данное поле предназначено для ввода конечного значения (в процентах от текущих размеров), с которого начинается уменьшение размера объекта при уменьшении значения связанного источника данных в процессе выполнения назначенного действия *Размер*.
Масштаб	При выборе данной опции изменение размеров выполняется прямо пропорционально изменению значения связанного источника данных.
Усечение	При выборе данной опции изменение размеров имеет вид “раскрытия”/”усечения”.
Имя объекта	Предназначено для идентификации объекта при программном доступе к его свойствам через OLE Automation
Описание	Может содержать краткое описание объекта, которое после ввода будет выводиться в виде подсказки (Tooltip)
Специальные данные	Используется для ввода произвольных данных пользователя.
Псевдоним языка	Отображение псевдонима языка. Если установлен флажок, нажатие на кнопку **...** открывает Навигатор Языковых псевдонимов, в котором можно выбрать псевдоним языка из базы данных.

Динамическое изменение положения



Инструмент *Динамическое изменение положения* (*Положение*) позволяет установить динамическое соединение объекта с источником данных, при котором графический объект в режиме *Исполнение* будет динамически перемещаться по траектории, мгновенные координаты которой пропорциональны значению источника данных. Диапазон изменения значения источника данных может быть переопределен в соответствующих полях группы параметров **Переопределить диапазон** страницы свойств.

Имеется возможность использования графического объекта, к которому применено данное динамическое действие, в качестве движкового регулятора (ползунка), когда положение объекта, изменяемое пользователем в пределах траектории, определяет значение связанного источника данных. Перемещение созданного ползунка выполняется путем помещения курсора на объект с последующим перемещением при одновременном удержании левой клавиши мыши в нажатом состоянии.

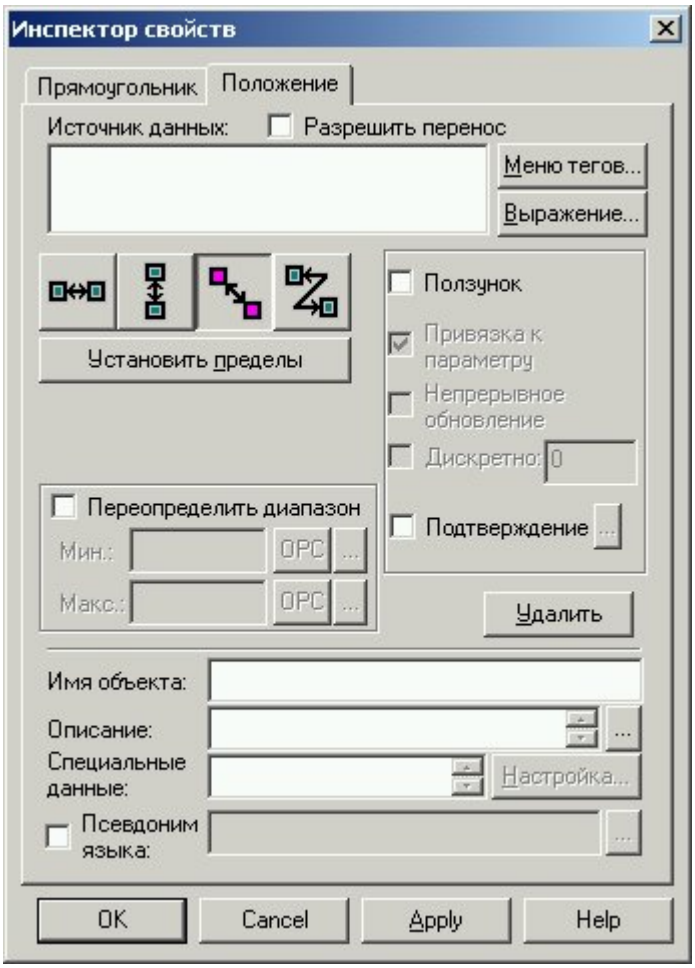


Рис. 8-4. ***Инспектор свойств*** *– страница* ***Положение***

Для установления динамического соединения *Положение*

1. Выделите графический объект
2. Выберите строку **Положение** подменю **Действия** меню **Динамика** или нажмите кнопку *Положение* набора инструментов *Динамика*. На экран монитора будет выведена диалоговая панель **Инспектор свойств** с выбранной страницей **Положение**, внешний вид которой показан на рис. 8-4.
3. Введите имя тега, переменной или псевдонима в поле **Источник данных** или воспользуйтесь кнопкой **Меню тегов**.

4. Выберите тип траектории нажатием кнопки с изображением способа перемещения.
5. Нажмите кнопку **Установить пределы**. Объект и его траектория будут выделены пунктирными линиями.
6. С помощью мыши переместите объект из начальной точки траектории в требуемую конечную, после чего нажмите клавишу Esc или выполните двойной щелчок левой клавишей мыши.
7. При необходимости использования объекта в качестве движкового регулятора отметьте флажок **Ползунок** и настройте параметры создаваемого регулятора, отметив флажки группы параметров **Ползунок** в соответствии с указаниями табл. 8-2.
8. Закройте диалоговую панель **Инспектора свойств**.

Описание полей страницы **Вращение** в **Инспекторе свойств** приведено в табл. 8-2.

Таблица 8-2

Поле	Описание
Источник данных	Данное поле предназначено для ввода имени тега или выражения, результат выполнения которого будет определять пропорциональное изменение одной или нескольких координат графического объекта.
Меню тегов	Данная кнопка открывает Универсальный навигатор данных, позволяющий выбрать соединение с тегами OPC, переменными и псевдонимами. Для более подробной информации, обратитесь к Справочной документации Unified Data Browser.
Выражение	Данная кнопка открывает Редактор выражений. Для более подробной информации, обратитесь к Справочной документации Expression_Editor.
Набор инструментов (кнопок) выбора типа траектории	Данные кнопки предназначены для выбора типа траектории, по которой будет перемещаться объект.
	Горизонтальная прямая траектория.
	Вертикальная прямая траектория.
	Прямая траектория с произвольным углом наклона.
	Произвольная траектория в виде ломаной. Для ввода вершин ломаной: 1. Нажмите данную кнопку, задайте произвольную прямую траекторию, для чего нажмите кнопку **Установить пределы** и переместите объект в конечную точку траектории с помощью мыши. 2. Поместите курсор на траекторию, отображаемую прямой пунктирной линией, соединяющей центры начального и конечного положений объекта. Курсор примет форму креста. 3. Щелкните левой клавишей мыши в точке траектории, в которой требуется создать вершину ломаной, после чего перемещайте курсор, удерживая левую клавишу мыши в нажатом состоянии.
Переопределить диапазон	Установка данного флажка позволяет установить отличный от используемого по умолчанию диапазон значений связанного источника данных.
Мин.	После установки флажка **Переопределить диапазон** данное поле позволяет установить нижнюю границу значений связанного источника данных.
Макс.	После установки флажка **Переопределить диапазон** данное поле позволяет установить верхнюю границу значений связанного источника данных.
OPC	Кнопки, расположенные справа от полей **Мин**. и **Макс**., позволяют установить в качестве нижней и верхней границ источника данных значения тегов путем вызова их из **Универсального навигатора данных**.

Поле	Описание
…	Вызов Редактора выражений для полей **Мин**. и **Макс**.
Ползунок	Установка данного флажка позволяет реализовать на базе объекта движковый регулятор, перемещаемый пользователем по выбранной траектории. Для перемещения движка в режиме *Исполнение* щелкните на нем мышью и перемещайте курсор, удерживая левую клавишу мыши в нажатом состоянии. Значение связанного источника данных будет изменяться в пределах установленного диапазона прямо пропорционально длине отрезка траектории, по которой передвинут ползунок.
Привязка к параметру	При установке данного флажка положение ползунка будет не только зависеть от действий пользователя, но и от значения связанного с ним источника данных.
Непрерывное обновление	При установке данного флажка значение связанного с ползунком источника данных будет обновляться с периодом обновления экранной формы.
Дискретно	При установке данного флажка диапазон значений, передаваемых от движкового регулятора связанному источнику данных, будет разбит на дискретные уровни, количество которых определяется целым числом в поле ввода **Дискретно**. Например, при установке **Мин**.=0, **Макс**.=10, **Дискретно**=5, ползунок будет иметь 5 фиксированных положений в пределах траектории: 0,0; 2,5; 5,0; 7,5; 10,0.
Подтверждение	Определяет подтверждающее сообщение, которое будет вызываться GraphWorX32 при вводе пользователем значений. После установки флажка, можно воспользоваться Конфигуратором подтверждающих сообщений. Для этого нужно нажать на кнопку … Для более подробной информации, обратитесь к пункту Конфигуратор подтверждающих сообщений (глава 8)
Установить пределы	Данная кнопка предназначена для задания траектории объекта. По завершении задания траектории следует выполнить двойной щелчок левой клавишей мыши вне прямоугольной области, охватывающей траекторию, или нажать клавишу Esc.
Удалить	Кнопка предназначена для удаления динамического соединения *Положение*.
Имя объекта	Предназначено для идентификации объекта при программном доступе к его свойствам через OLE Automation
Описание	Может содержать краткое описание объекта, которое после ввода будет выводиться в виде подсказки (Tooltip)
Специальные данные	Используется для ввода произвольных данных пользователя.
Псевдоним языка	Отображение псевдонима языка. Если установлен флажок, нажатие на кнопке **...** открывает Навигатор Языковых псевдонимов, в котором можно выбрать псевдоним языка из базы данных.

Динамическое вращение



Инструмент *Динамическое вращение* (*Вращение*) позволяет установить динамическое соединение объекта с источником данных, при котором графический объект в режиме *Исполнение* будет вращаться вокруг заданной оси. Мгновенное значение угла поворота объекта относительно своего первоначального положения будет пропорционально значению источника данных. Диапазон изменения значений источника данных может быть переопределен в соответствующих полях группы параметров **Переопределить диапазон** страницы свойств.

Имеется возможность использования графического объекта, к которому применено данное динамическое действие, в качестве круговой шкалы, когда угол поворота объекта, изменяемый пользователем в пределах заданного диапазона углов, определяет значение связанного источника данных. Перемещение регулятора созданной круговой шкалы выполняется путем помещения курсора на объект с последующим перемещением при одновременном удержании левой клавиши мыши в нажатом состоянии.

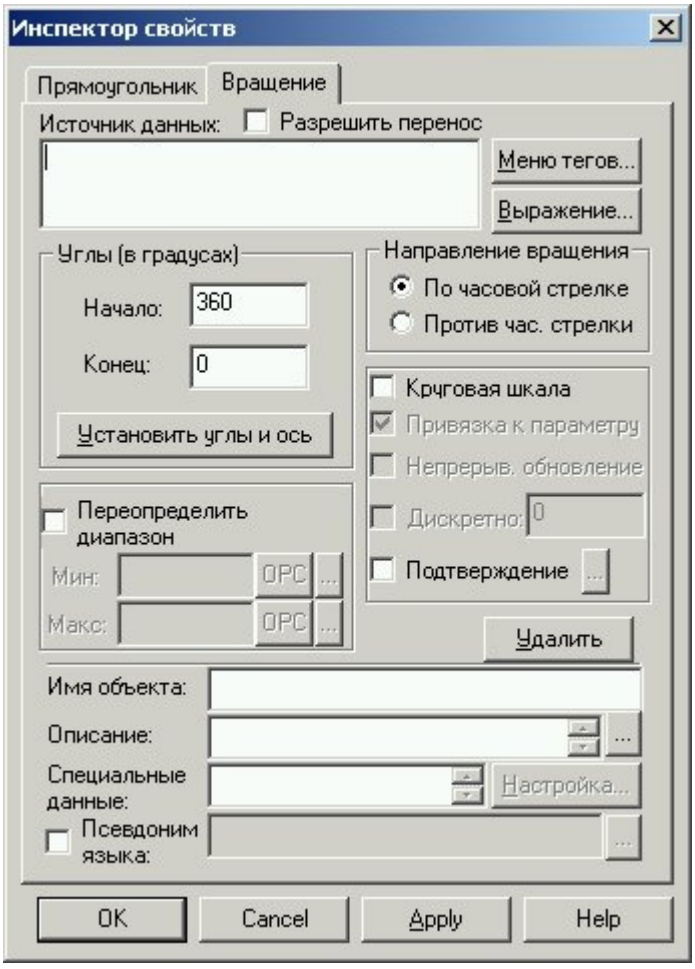


Рис. 8-5. ***Инспектор свойств*** *– страница* ***Вращение***

Для установления динамического соединения *Вращение*

1. Выделите графический объект
2. Выберите строку **Вращение** подменю **Действия** меню **Динамика** или нажмите кнопку *Вращение* набора инструментов *Динамика*. На экран монитора будет выведена диалоговая панель **Инспектор свойств** с выбранной страницей **Вращение**, внешний вид которой показан на рис. 8-5.

3. Введите имя тега, локальной переменной или псевдонима в поле **Источник данных** или воспользуйтесь кнопкой **Меню тегов**.
4. Нажмите кнопку **Установить углы и ось**. Ось вращения будет выделена символом в виде креста, заключенного в круг. Начальное положение объекта будет выделено пунктирными линиями.
5. С помощью мыши переместите ось вращения в требуемую точку экранной формы.
6. Выберите направление вращения, установив соответствующий переключатель в положение **По часовой стрелке** или **Против часовой стрелки**.
7. С помощью мыши переместите объект в точку, соответствующую верхнему предельному углу поворота, после чего нажмите клавишу Esc или выполните двойной щелчок левой клавишей мыши. Установка диапазона углов поворота объекта относительно выбранного положения оси вращения может быть произведена вводом соответствующих значений в поля **Углы (в градусах):** -- **Начало**, **Конец**.
8. При необходимости использования объекта в качестве круговой шкалы отметьте флажок **Круговая шкала** и настройте параметры создаваемой шкалы-регулятора, отметив флажки группы параметров **Круговая шкала** в соответствии с указаниями табл. 8-3.
9. Закройте диалоговую панель **Инспектора свойств**.

Описание полей страницы свойств **Вращение** приведено в табл. 8-3.

Таблица 8-3

Поле	Описание
Источник данных	Данное поле предназначено для ввода имени тега или выражения, результат выполнения которого будет определять пропорциональное изменение угла поворота графического объекта относительно выбранной оси.
Меню тегов	Данная кнопка открывает Универсальный навигатор данных, позволяющий выбрать соединение с тегами OPC, переменными и псевдонимами. Для более подробной информации, обратитесь к Справочной документации Unified Data Browser.
Выражение	Данная кнопка открывает Редактор выражений. Для более подробной информации, обратитесь к Справочной документации Expression_Editor
Углы (в градусах)	
Начало	Начальное значение угла поворота вокруг выбранной оси вращения.
Конец	Конечное значение конечного угла поворота вокруг выбранной оси вращения.
Установить углы и ось	Данная кнопка позволяет задать с помощью мыши положение оси вращения в пределах экранной формы, а также начальное и конечное значения углов поворота объекта.
Направление вращения	Переключатель определяет направление вращения объекта (по или против часовой стрелки).
Переопределить диапазон	Установка данного флажка позволяет установить отличный от используемого по умолчанию диапазон значений связанного источника данных.
Мин.	После установки флажка **Переопределить диапазон** данное поле позволяет установить нижнюю границу значений связанного источника данных.
Макс.	После установки флажка **Переопределить диапазон** данное поле позволяет установить верхнюю границу значений связанного источника данных.
OPC	Кнопки, расположенные справа от полей **Мин**. и **Макс**., позволяют установить в качестве нижней и верхней границ источника данных значения тегов путем вызова их из **Универсального навигатора данных**.
…	Вызов Редактора выражений для полей **Мин**. и **Макс**.
Круговая шкала	Установка данного флажка позволяет реализовать на базе объекта круговую шкалу, нониус которой может перемещаться пользователем в пределах выбранных углов, а также отображать значение связанного источника данных в виде стрелочного индикатора. Для перемещения

Поле	Описание
	нониуса в режиме *Исполнение* щелкните на нем мышью и перемещайте курсор, удерживая левую клавишу мыши в нажатом состоянии. Значение связанного источника данных будет изменяться в пределах установленного диапазона прямо пропорционально углу поворота нониуса.
Привязка к параметру	При установке данного флажка положение нониуса круговой шкалы будет не только зависеть от действий пользователя, но и от значения связанного с ним источника данных.
Непрерывное обновление	При установке данного флажка значение связанного с круговой шкалой источника данных будет обновляться с периодом обновления экранной формы.
Дискретно	При установке данного флажка диапазон значений, передаваемых от нониуса шкалы связанному источнику данных, будет разбит на дискретные уровни, количество которых определяется целым числом в поле ввода **Дискретно**. Например, при установке **Мин**.=0, **Макс**.=10, **Дискретно**=5, нониус будет иметь 5 фиксированных положений в пределах траектории: 0,0; 2,5; 5,0; 7,5; 10,0.
Подтверждение	Определяет подтверждающее сообщение, которое будет вызываться GraphWorX32 при вводе пользователем значений. После установки флажка, можно воспользоваться Конфигуратором подтверждающих сообщений. Для этого нужно нажать на кнопку ... Для более подробной информации, обратитесь к пункту Конфигуратор подтверждающих сообщений (глава 8).
Удалить	Кнопка предназначена для удаления динамического соединения *Вращение*.
Имя объекта	Предназначено для идентификации объекта при программном доступе к его свойствам через OLE Automation
Описание	Может содержать краткое описание объекта, которое после ввода будет выводиться в виде подсказки (Tooltip)
Специальные данные	Используется для ввода произвольных данных пользователя.
Псевдоним языка	Отображение псевдонима языка. Если установлен флажок, нажатие на кнопке **...** открывает Навигатор Языковых псевдонимов, в котором можно выбрать псевдоним языка из базы данных.

Динамическое соединение *Скрыть/Блокировать*



Инструмент *Скрыть/Блокировать* позволяет установить динамическое соединение объекта с булевым значением источника данных, при котором графический объект в режиме *Исполнение* будет исчезать с экрана, если связанное булево значение равно *ИСТИНА* (*True*) или *ЛОЖЬ* (*False*). Если переключатель **Скрыть или Блокировать** страницы **Скрыть** в **Инспекторе свойств**, установлен в положение **Блокировать**, то при выполнении условия **Скрыть/Блокировать, если False/True** на экране останется только контур объекта, отображаемый тонкой линией серого цвета, и пользователь не будет иметь возможности выполнения всех операций, связанных с вводом значений с помощью объекта (например, если объектом является кнопка, то при блокировке пользователь не сможет ее нажать).

Для установления динамического соединения *Скрыть/Блокировать*

1. Выделите объект.
2. Выберите строку **Скрыть/Блокировать** подменю **Действия** меню **Динамика** или нажмите кнопку *Скрыть/Блокировать* набора инструментов *Динамика*. На экран монитора будет выведена диалоговая панель **Инспектор свойств** с выбранной страницей **Скрыть**, внешний вид которой показан на рис. 8-6.
3. Введите имя тега, переменной, псевдонима в поле **Источник** или воспользуйтесь кнопкой **Меню тегов**. Следует использовать источники данных с булевыми значениями.
4. Установите в требуемое положение переключатель **Скрыть или блокировать** в зависимости от того, какое действие должно выполняться в режиме *Исполнение*.
5. С помощью переключателя **Скрыть/Блокировать, если False/True** выберите булево значение источника данных, при котором будет выполняться целевое действие.
6. Для завершения работы с **Инспектором свойств** нажмите кнопку **OK**.

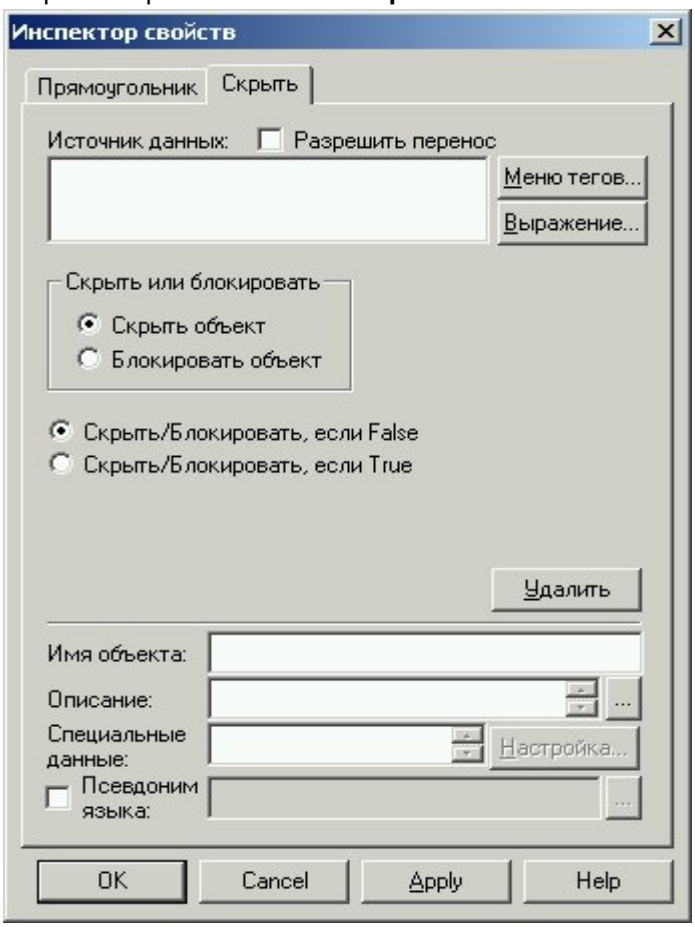


*Рис. 8-6. **Инспектор свойств** – страница **Скрыть/Блокировать***

Дискретное изменение цвета



Инструмент *Цвет* позволяет установить динамическое соединение объекта с источником данных, при котором цвет графического объекта в режиме *Исполнение* будет динамически изменяться. Изменение цвета происходит дискретно в зависимости от событий, определяемых в поле **Источник данных**. Имеется возможность назначения нескольких событий, возникновение которых будет приводить к соответствующему изменению цвета объекта. События имеют приоритет, который означает, что при одновременном возникновении нескольких событий цвет объекта будет изменен на заданный для события, находящегося ниже остальных событий в списке **Источник данных**. Изменение приоритета события производится в режиме *Разработка* с помощью кнопок **Изменить порядок** страницы **Цвет** в **Инспекторе свойств**, показанной на рис. 8-7.

Для установления динамического соединения *Цвет:*

1. Выделите объект.
2. Выберите строку **Цвет** подменю **Действия** меню **Динамика** или нажмите кнопку *Цвет* набора инструментов *Динамика*. На экран монитора будет выведена диалоговая панель **Инспектор свойств** с выбранной страницей **Цвет**, внешний вид которой показан на рис. 8-7.
3. Введите имя тега, переменной или псевдоним в поле **Источник данных** или воспользуйтесь кнопкой **Меню тегов**. Следует использовать источники данных с булевыми значениями.
4. По завершении ввода нажмите кнопку **Добавить**. Имя источника данных или выражение будет помещено в начало списка событий, по которым выполняется динамическое действие.
5. В группе параметров **Изменение цвета** установите флажки для цветов графического объекта, которые должны изменяться.
6. Выберите условие, по которому будет происходить изменение цветов объекта, установив в соответствующее положение переключатель **Изменить цвет, если True/False**.
7. В группе параметров **Изменение цвета** щелкните левой клавишей мыши на символе квадратной формы, расположенном справа от флажка **Цвет заливки**. В появившейся палитре цветов выберите цвет, который будет применен к цвету заливки объекта при возникновении события с параметрами, определенными в п.п. 3–6.
8. При необходимости выполните действие п. 7 для цветов линии и тени.
9. При необходимости добавьте требуемое количество событий и цветов согласно указаниям п.п. 3–8.
10. Назначьте приоритеты событиям, выделяя каждое из них в списке и нажимая кнопки **Изменить порядок**. Наивысший приоритет имеет событие, расположенное ниже остальных в списке.
11. Закройте **Инспектор свойств** нажатием кнопки **OK**.

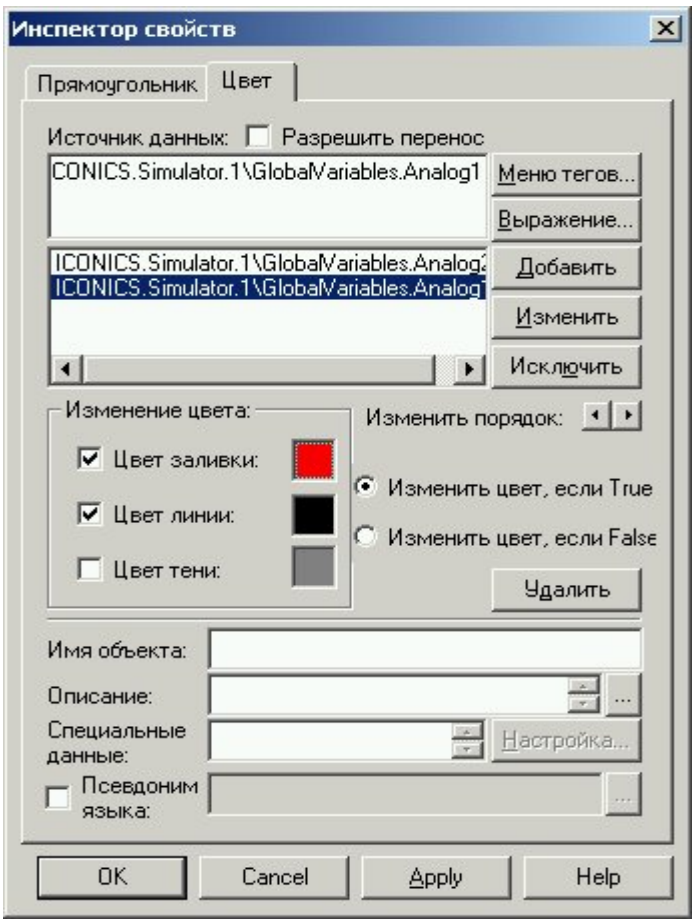


*Рис. 8-7. **Инспектор свойств** – страница **Цвет***

Таблица 8-4

Поле	Описание
Источник данных	Данное поле предназначено для ввода имени тега или выражения, булев результат которого будет определять установленные для данного события цвета графического объекта. Верхняя область поля содержит вводимое или текущее редактируемое имя источника данных или выражение. Нижняя область содержит список установленных событий.
Меню тегов	Данная кнопка открывает Универсальный навигатор данных, позволяющий выбрать соединение с тегами OPC, переменными и псевдонимами. Для более подробной информации, обратитесь к Справочной документации Unified Data Browser.
Выражение	Данная кнопка открывает Редактор выражений. Для более подробной информации, обратитесь к Справочной документации Expression_Editor
Изменение цвета	Данная группа параметров позволяет выбрать свойства объекта, к которым будет применяться действие в режиме *Исполнение*.
Цвет заливки	При установке данного флажка возникновение события приведет к дискретному изменению начального цвета заливки объекта на цвет, установленный правее флажка.
Цвет линии	При установке данного флажка возникновение события приведет к дискретному изменению начального цвета линии, изображающей контур объекта, на цвет, установленный правее флажка.
Цвет тени	При установке данного флажка возникновение события приведет к дискретному изменению начального цвета тени объекта на цвет, установленный правее флажка.
Добавить	Данная кнопка предназначена для добавления условия в список существующих из верхней части поля **Источник данных**
Изменить	Данная кнопка позволяет отредактировать ранее введенное условие изменения цвета.

Поле	Описание
Исключить	Данная кнопка предназначена для удаления одного из имеющихся условий изменения цвета.
Изменить порядок	Данные кнопки служат для изменения взаимного расположения установленных событий в списке **Источник данных**.
Удалить	Кнопка предназначена для удаления динамического соединения *Цвет*.
Имя объекта	Предназначено для идентификации объекта при программном доступе к его свойствам через OLE Automation
Описание	Может содержать краткое описание объекта, которое после ввода будет выводиться в виде подсказки (Tooltip)
Специальные данные	Используется для ввода произвольных данных пользователя.
Псевдоним языка	Отображение псевдонима языка. Если установлен флажок, нажатие на кнопке **...** открывает Навигатор Языковых псевдонимов, в котором можно выбрать псевдоним языка из базы данных.

Плавное изменение цвета



Инструмент *Плавное изменение цвета* позволяет установить динамическое соединение объекта с источником данных, при котором цвет графического объекта в режиме *Исполнение* будет динамически изменяться. Изменение цвета происходит плавно в заданном диапазоне цветов в зависимости от значения, определяемого полем **Источник данных**.

Для установления динамического соединения *Плавное изменение цвета*.

1. Выделите объект.
2. Выберите строку **Плавное изменение цвета** подменю **Действия** меню **Динамика** или нажмите кнопку *Плавное изменение цвета* набора инструментов *Динамика*. На экран монитора будет выведена диалоговая панель **Инспектор свойств** с выбранной страницей **Цвет плавно**, внешний вид которой показан на рис. 8-8.
3. Введите имя тега, переменной или псевдоним в поле **Источник данных** или воспользуйтесь кнопкой **Меню тегов**.
4. Установите флажки для цветов графического объекта, которые должны изменяться.
5. При необходимости переопределите диапазон значений **Источника данных**.
6. Выберите начальный и конечный цвета для заливки, линии и тени объекта.
7. При необходимости установите флажки **Выше/Ниже диапазона** в группе **Цвет по умолчанию**. Цвет по умолчанию может быть изменен на основной странице свойств статического объекта. Таким образом, если значение связанного источника данных будет выходить за пределы диапазона, установленного в группе параметров **Переопределить диапазон**, к объекту будут применены цвета по умолчанию.
8. Закройте **Инспектор свойств** нажатием кнопки **OK**.

Описание всех полей страницы свойств **Цвет плавно** приведено в табл. 8-5.

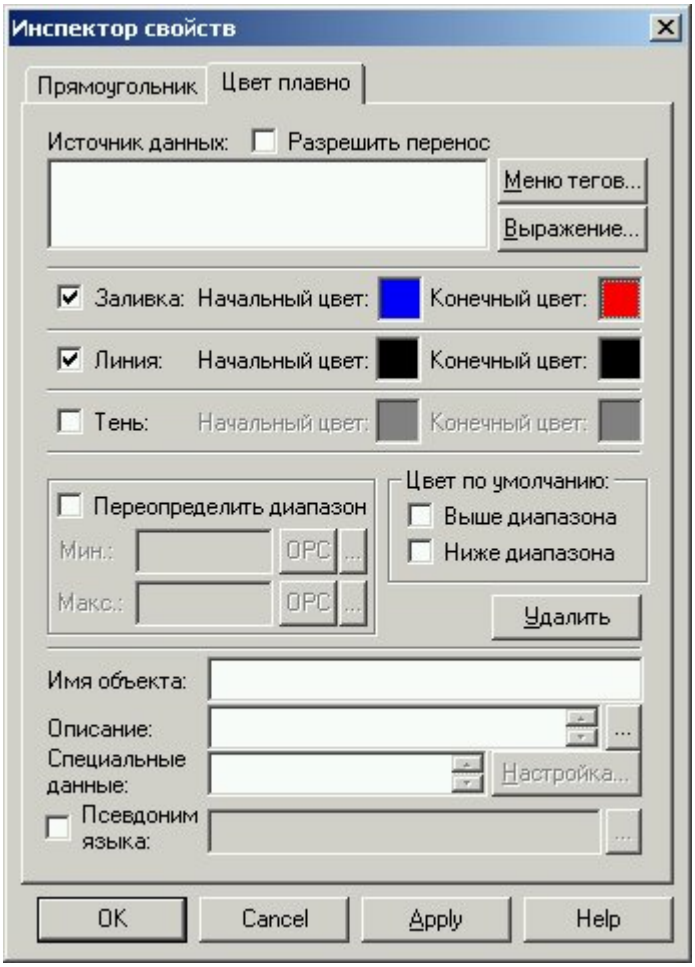


*Рис. 8-8. **Инспектор свойств** – страница **Цвет плавно***

Таблица 8-5

Поле	Описание
Источник данных	Данное поле предназначено для ввода имени тега или выражения, значения которого будет определять установленные для данного события цвета графического объекта.
Меню тегов	Данная кнопка открывает Универсальный навигатор данных, позволяющий выбрать соединение с тегами OPC, переменными и псевдонимами. Для более подробной информации, обратитесь к Справочной документации Unified Data Browser.
Выражение	Данная кнопка открывает Редактор выражений. Для более подробной информации, обратитесь к Справочной документации Expression_Editor
Цвет заливки	При установке данного флажка попадание значений источника данных в диапазон, установленный в группе параметров **Переопределить диапазон**, приведет к плавному изменению цвета заливки от установленного в поле **Начальный цвет** до цвета в поле **Конечный цвет**.
Цвет линии	При установке данного флажка попадание значений источника данных в диапазон, установленный в группе параметров **Переопределить диапазон**, приведет к плавному изменению цвета линии контура объекта от установленного в поле **Начальный цвет** до цвета в поле **Конечный цвет**.
Цвет тени	При установке данного флажка попадание значений источника данных в диапазон, установленный в группе параметров **Переопределить диапазон**, приведет к плавному изменению цвета тени объекта от установленного в поле **Начальный цвет** до цвета в поле **Конечный цвет**.
Цвет по умолчанию	Флажки данной группы позволяют указать, что при выходе значения источника данных за установленные пределы, объект будет иметь цвета по умолчанию.

Поле	Описание
Удалить	Кнопка предназначена для удаления динамического соединения *Цвет*.
Переопределить диапазон	Установка данного флажка позволяет установить отличный от используемого по умолчанию диапазон значений связанного источника данных.
Мин.	После установки флажка **Переопределить диапазон** данное поле позволяет установить нижнюю границу значений связанного источника данных.
Макс.	После установки флажка **Переопределить диапазон** данное поле позволяет установить верхнюю границу значений связанного источника данных.
OPC	Кнопки, расположенные справа от полей **Мин**. и **Макс**., позволяют установить в качестве нижней и верхней границ источника данных значения тегов путем вызова их из **Универсального навигатора данных.**
…	Вызов Редактора выражений для полей **Мин**. и **Макс**.
Имя объекта	Предназначено для идентификации объекта при программном доступе к его свойствам через OLE Automation
Описание	Может содержать краткое описание объекта, которое после ввода будет выводиться в виде подсказки (Tooltip)
Специальные данные	Используется для ввода произвольных данных пользователя.
Псевдоним языка	Отображение псевдонима языка. Если установлен флажок, нажатие на кнопке **...** открывает Навигатор Языковых псевдонимов, в котором можно выбрать псевдоним языка из базы данных.

Прерывистое отображение



Графический объект, для которого установлено динамическое соединение *Прерывистое отображение*, в режиме *Исполнение* отображается прерывисто (“мигает”) с установленным периодом в зависимости от логического состояния связанного источника данных или имеет альтернативные цвета заливки, линии и тени (находится в альтернативном состоянии).

По умолчанию при выполнении заданного условия **Прерывисто, если True/False** объект исчезает и вновь появляется с периодом, установленным в поле **Период (мс)**. Например, установка переключателя **Прерывисто, если False** инициирует в режиме *Исполнение* прерывистое отображение объекта, если значение связанного с ним источником данных равно *ЛОЖЬ* (*False*).

Если в группе параметров **Альтернативное состояние** установлен переключатель **Изменить цвет**, то в при выполнении заданного условия будет происходить смена цветов заливки, линии и тени объекта. Например, если исходным цветом объекта является зеленый и установлен переключатель **Изменить цвет**, а также флажки **Цвет заливки** (красный) и **Цвет линии** (красный), то при выполнении заданного условия будет происходить периодическое изменение цвета заливки и линии контура объекта с зеленого на красный.

Как указывалось выше, период прерывистого отображения устанавливается в поле **Период (мс)**.

Если установлен переключатель **Альтернативное состояние, пока не прерывисто**, то при выполнении условия, противоположного заданному для прерывистого отображения, объект будет переходить в альтернативное состояние. Например, если исходным цветом объекта является зеленый и установлен переключатель **Изменить цвет**, а также флажки **Цвет заливки** (красный), **Цвет линии** (красный) и **Альтернативное состояние, пока не прерывисто**, то при выполнении заданного условия будет происходить периодическое изменение цвета заливки и линии контура объекта с зеленого на красный. При выполнении условия, противоположного заданному для прерывистого отображения, заливка и контурная линия объекта будут непрерывно отображаться красным цветом.

Для установления динамического соединения *Прерывистое отображение*

1. Выделите объект.
2. Выберите строку **Прерывистое отображение** подменю **Действия** меню **Динамика** или нажмите кнопку *Прерывистое отображение* набора инструментов *Динамика*. На экран монитора будет выведена диалоговая панель **Инспектор свойств** с выбранной страницей **Прерывистое отображение**, внешний вид которой показан на рис. 8-9.
3. Введите имя тега, переменной или псевдоним в поле **Источник данных** или воспользуйтесь кнопкой **Меню тегов**. Источник данных должен представлять в качестве значения одно из двух логических состояний *ИСТИНА* (*True*) или *ЛОЖЬ* (*False*).
4. Задайте условие, при выполнении которого в режиме *Исполнение* объект будет отображаться прерывисто, для чего установите в требуемое состояние переключатель **Прерывисто, если True/False**.
5. Выберите альтернативное состояние объекта при прерывистом отображении в группе параметров **Альтернативное состояние**.
6. При необходимости установите флажок **Альтернативное состояние, если не прерывисто**.
7. Для закрытия **Инспектора свойств** щелкните на кнопке **OK**.

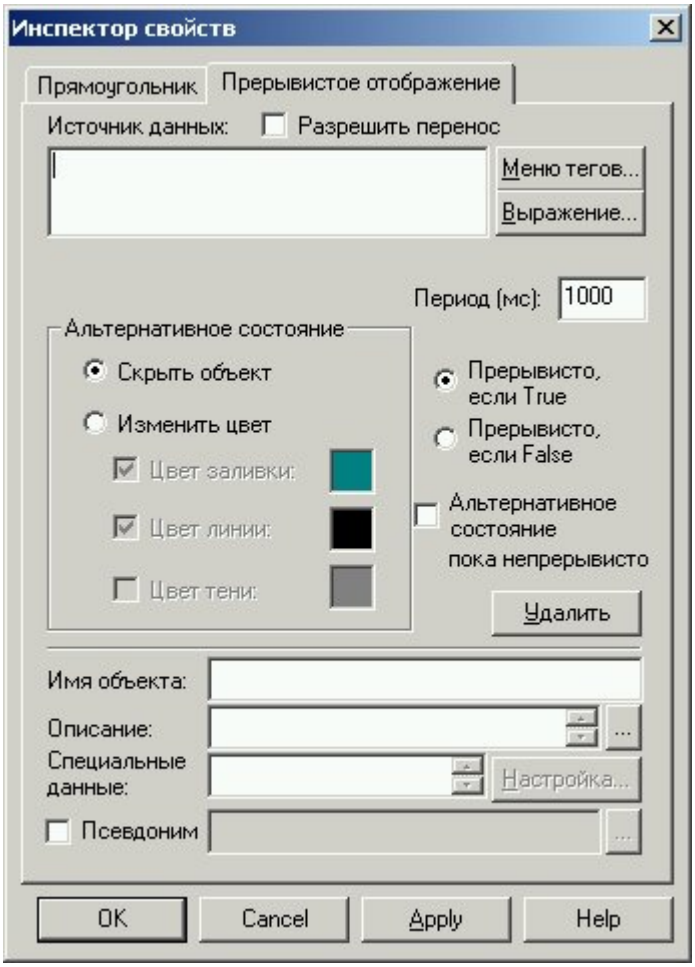


Рис. 8-9. ***Инспектор свойств*** *– страница* ***Прерывистое отображение***

Указание и щелчок



Динамическое соединение *Указание и щелчок* позволяет выполнять заданную операцию в случае, если пользователь помещает курсор мыши на объект и выполняет однократный щелчок левой, правой или средней клавишей мыши. Имеется возможность использования следующих операций: Загрузка экранной формы, Загрузка в стиле Перенести/Оставить, Загрузка предыдущей или следующей экранной формы, Загрузка экранной формы во всплывающем или встроенном окне, Закрыть окно экранной формы, Запустить приложение, Передать значение, Переключить значение, Настройка псевдонимов, Диалог настройки псевдонимов, Скрыть слой, Показать слой, переключить отображение слоя, Выполнить сценарий VBA, Специальная команда и Всплывающее меню.

Для установления динамического соединения *Указание и щелчок*:

1. Выделите объект.
2. Выберите строку **Указание и щелчок мышью** подменю **Действия** меню **Динамика** или нажмите кнопку *Указание и щелчок* набора инструментов *Динамика*. На экран монитора будет выведена диалоговая панель **Инспектор свойств** с выбранной страницей **Указание и щелчок**, внешний вид которой показан на рис. 8-10.
3. Выберите клавишу мыши в выпадающем списке **Мышь**, щелчок которой будет активизировать операцию.
4. Выберите операцию, которая будет выполняться при указании и щелчке мышью, в выпадающем списке **Действие**.
5. Установите дополнительные параметры выполнения операций, количество и содержание которых зависят от типа операции.

6. Закройте диалоговую панель **Инспектора свойств**, для чего нажмите кнопку **OK**.

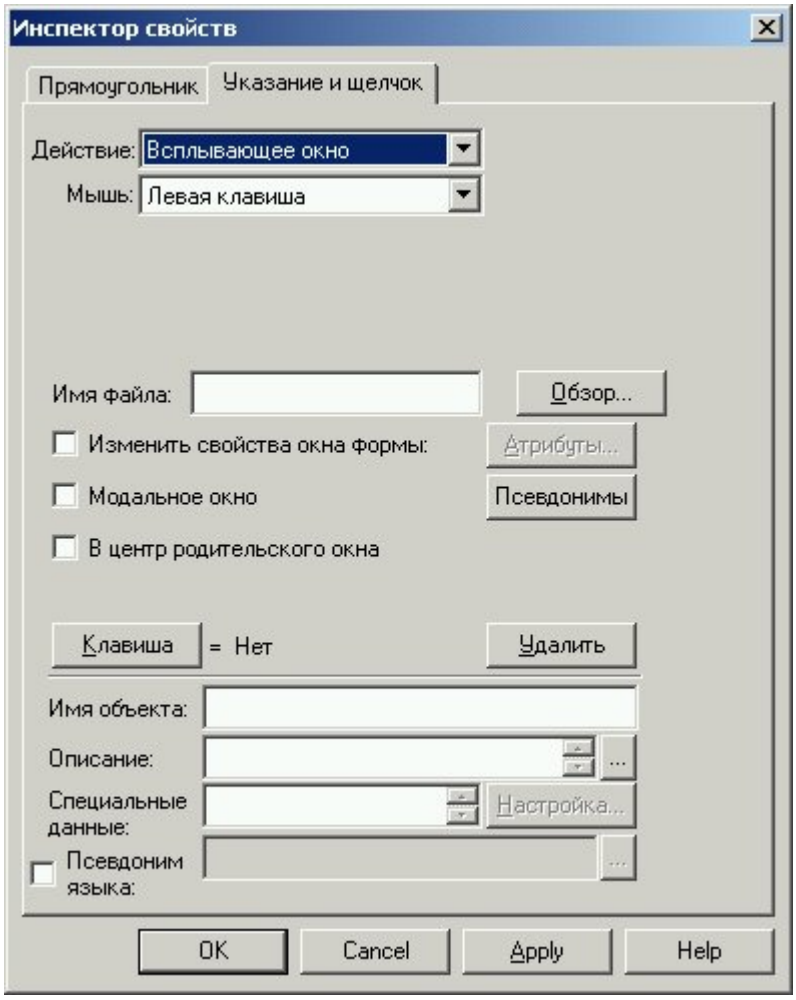


Рис. 8-10. ***Инспектор свойств*** *– страница* ***Указание и щелчок***

Нажатие на кнопке **Клавиша** открывает окно, позволяющее определить сочетание клавиш, с помощью которого можно будет выполнять аналогичное действие.

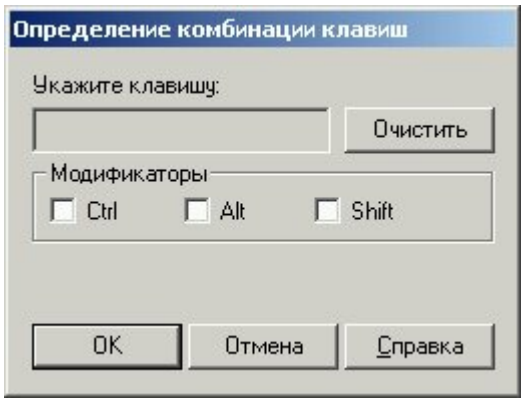


Рис. 8-11. Определение комбинации клавишей

Параметры страницы свойств динамического соединения *Указание и щелчок*

Описание общих параметров страницы свойств динамического соединения *Указание и щелчок* приведено в табл. 8-6.

Таблица 8-6

Поле	Описание
Мышь	Выпадающий список позволяет выбрать клавишу мыши, однократный щелчок которой при помещении курсора на объект будет приводить к выполнению заданной операции.
Удалить	Данная кнопка предназначена для удаления динамического соединения *Указание и щелчок* из числа примененных к графическому объекту.
Тип	Имеются три варианта действия динамического соединения *Указание и щелчок*: *Без фиксации* – указание и щелчок считаются выполненными сразу после щелчка мышью на объекте. Объект можно рассматривать в качестве кнопки без фиксации нажатого состояния. *С фиксацией* – объект имеет два фиксированных состояния: *Нажато* и *Отпущено*. Первое указание и щелчок приводят к переводу объекта в состояние *Нажато*. Последующий второй щелчок – в состояние *Отпущено*. Объект можно рассматривать в качестве кнопки с фиксацией нажатого состояния. *Зависимая* – позволяет создать группу графических объектов, порядок работы которых аналогичен принятому в группах зависимых кнопок. Указание и щелчок на одном из объектов приводят к тому, что действие предыдущих указаний и щелчков на других объектах группы отменяется
Начальное состояние	Выпадающий список позволяет выбрать состояние (*Нажато* или *Отпущено*) объекта сразу после перевода экранной формы в режим *Исполнение*.
Группа	Данное поле доступно только в случае, если в выпадающем списке **Тип** выбрана строка *Зависимая*. Поле предназначено для ввода имени группы, которой принадлежат объекты, выполняющие функцию группы зависимых кнопок.
Порядок выполнения	Данная группа параметров задает последовательность выполнения операции, установленной в поле **Действие**. Последовательность выполнения определяется тремя положениями выбранной клавиши мыши: **Если нажато**, **Пока нажато** и **Если отпущено**.
Если нажато	В случае установки данного флажка выбранная операция выполняется при переходе выбранной клавиши мыши из отпущенного в нажатое положение (состояние). Если объект, к которому применено динамическое соединение *Указание и щелчок*, сравнивать с кнопкой, то выполнение выбранного действия происходит в момент перехода кнопки из отпущенного в нажатое состояние.
Пока нажато	В случае установки данного флажка выбранная операция выполняется в течение заданного (в миллисекундах) интервала времени при нахождении выбранной клавиши мыши в нажатом состоянии. Если объект, к которому применено динамическое соединение *Указание и щелчок*, сравнивать с кнопкой, то выполнение выбранного действия происходит в течение установленного в поле **Интервал (мс)** времени, пока кнопка находится в нажатом состоянии.
Если отпущено	В случае установки данного флажка выбранная операция выполняется при переходе выбранной клавиши мыши из нажатого в отпущенное состояние. Если объект, к которому применено динамическое соединение *Указание и щелчок*, сравнивать с кнопкой, то выполнение выбранного действия происходит в момент перехода кнопки из нажатого в отпущенное состояние.

1. Действие *Загрузка экранной формы*

Данная операция в режиме *Исполнение* состоит в загрузке указанной в поле **Имя файла** экранной формы GraphWorX32. Обратите внимание на тот факт, что работа текущей экранной формы будет завершена.

Примечание. Если в диалоговой панели **Параметры приложения** на странице свойств **Исполнение** установлен флажок **Не выводить сообщение о необходимости сохранения изменений**, то все изменения, внесенные в экранную форму с момента ее последнего сохранения перед переводом в режим *Исполнение*, будут аннулированы (отменены).

Действие ***Загрузка в стиле Перенести/Оставить***

Данная операция в режиме *Исполнение* состоит в загрузке указанной в поле **Имя файла** экранной формы в другое окно GraphWorX32 методом "Перенести-и-Оставить " (Drag-and-Drop).

Операция выполняется следующим образом:

1. Запустите вторую копию GraphWorX32 не останавливая исполнение экранной формы, в которой находится графический объект, к которому применено действие *Указание и щелчок* с операцией *Загрузка в стиле Перенести и оставить*.
2. Поместите курсор на графический объект в исполняющейся экранной форме, нажмите выбранную клавишу и мыши и, не отпуская, переместите курсор в область меню или панели инструментов окна второй копии GraphWorX32, после чего отпустите клавишу мыши.
3. Экранная форма, указанная в поле **Имя файла**, будет загружена в окно второй копии GraphWorX32.

2. Действие *Предыдущая экранная форма*

Выполнение данной операции в режиме *Исполнение* состоит в загрузке экранной формы, которая исполнялась перед загрузкой текущей экранной формы. GraphWorX32 сохраняет список последних 50-ти загруженных экранных форм. Таким образом, данная операция аналогична выполняемой в *MS Internet Explorer* операции *Назад*.

Примечание. Сохраняемый список последних 50-ти экранных форм содержит исходные настройки псевдонимов для каждой предыдущей экранной формы. Таким образом, при переходах к предыдущим экранным формам настройки всех псевдонимов будут сброшены в соответствующие начальные состояния. Для получения более подробной информации обратитесь к разделу 7 настоящего руководства.

3. Действие *Всплывающее окно*

Данная операция в режиме *Исполнение* состоит в загрузке указанной в поле **Имя файла** экранной формы GraphWorX32 во всплывающем независимом окне.

Параметры страницы **Указание и щелчок** в **Инспекторе свойств** для операции *Всплывающее окно* показаны на рис. 8-10.

Описание параметров приведено в табл. 8-7.

Таблица 8-7

Поле	Описание
Имя файла	Предназначено для ввода пути и имени файла экранной формы GraphWorX32, которая будет загружена в дочернем окне.
Обзор	Позволяет выбрать имя файла экранной формы.
Изменить свойства экранной формы	После установки данного флажка кнопка **Атрибуты**, позволяющая задать параметры окна загружаемой экранной формы в режиме *Исполнение*, становится доступной.
Модальное окно	Если установлен данный флажок, то после загрузки экранной формы в дочернее окно, все события, связанные с вводом информации пользователем, включая щелчки мышью и нажатия клавиш клавиатуры, поступают и обрабатываются только дочерним окном. Таким образом, родительское окно становится недоступным для ввода информации.
В центр родительского окна	Если установлен данный флажок, то дочернее окно загруженной экранной формы будет автоматически размещено в центре родительского окна.
Атрибуты	Данная кнопка доступна после установки флажка **Изменить свойства экранной формы**. Нажатие кнопки выводит на экран монитора диалоговую панель **Свойства дочернего окна**, которая позволяет настроить параметры отображения экранной формы в режиме *Исполнение* после загрузки. Следует заметить, что параметры экранной формы, указанной в поле **Имя файла**, остаются неизменными при ее автономной работе (не в дочернем окне).

4. Действие *Встроенное окно*

Встроенное окно аналогично всплывающему, за исключением того, что его границы всегда располагаются в пределах окна экранной формы, из которой произведен его вызов. Отличие всплывающего окна от встроенного иллюстрируется рис. 8-12.

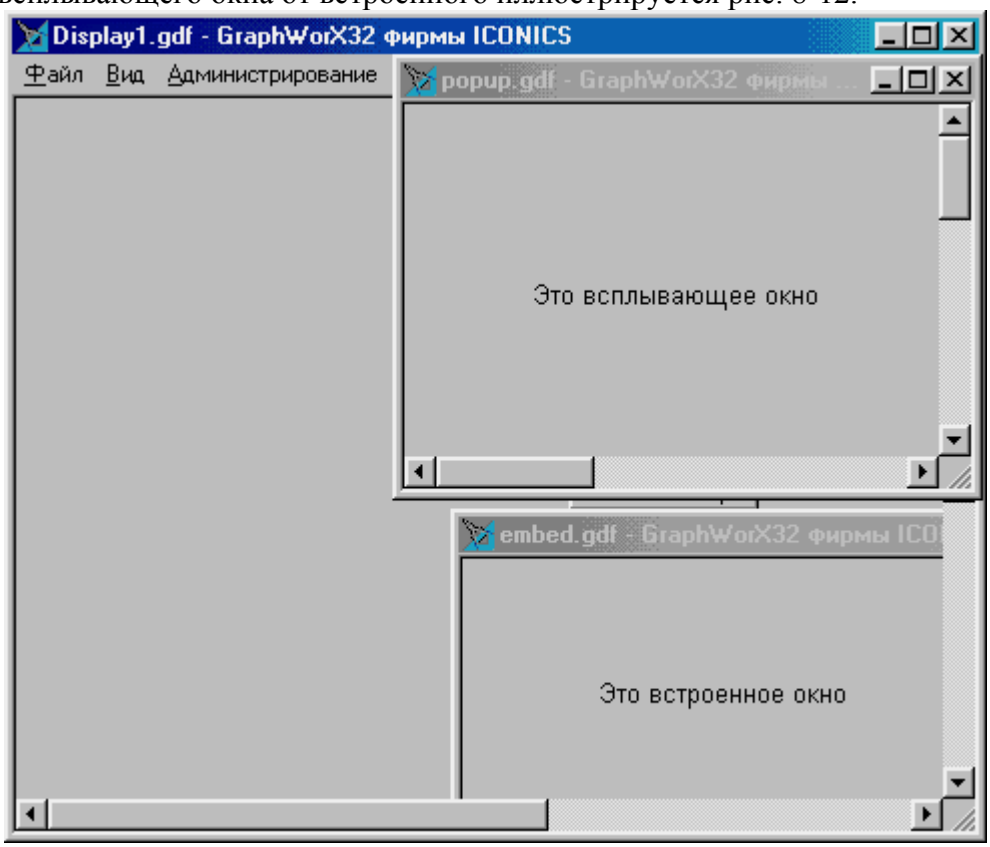


Рис. 8-12. Отличие встроенного окна от всплывающего

Для встроенного окна могут выполняться стандартные операции перемещения, прокрутки, изменения размеров, сворачивания в пиктограмму и увеличения в пределах родительского окна.

Параметры страницы **Указание и щелчок** в **Инспекторе свойств** для операции *Встроенное окно*, внешний вид которой показан на рис. 8-13, аналогичны используемым для всплывающего окна, за исключением отсутствия параметра **Модальное окно**.

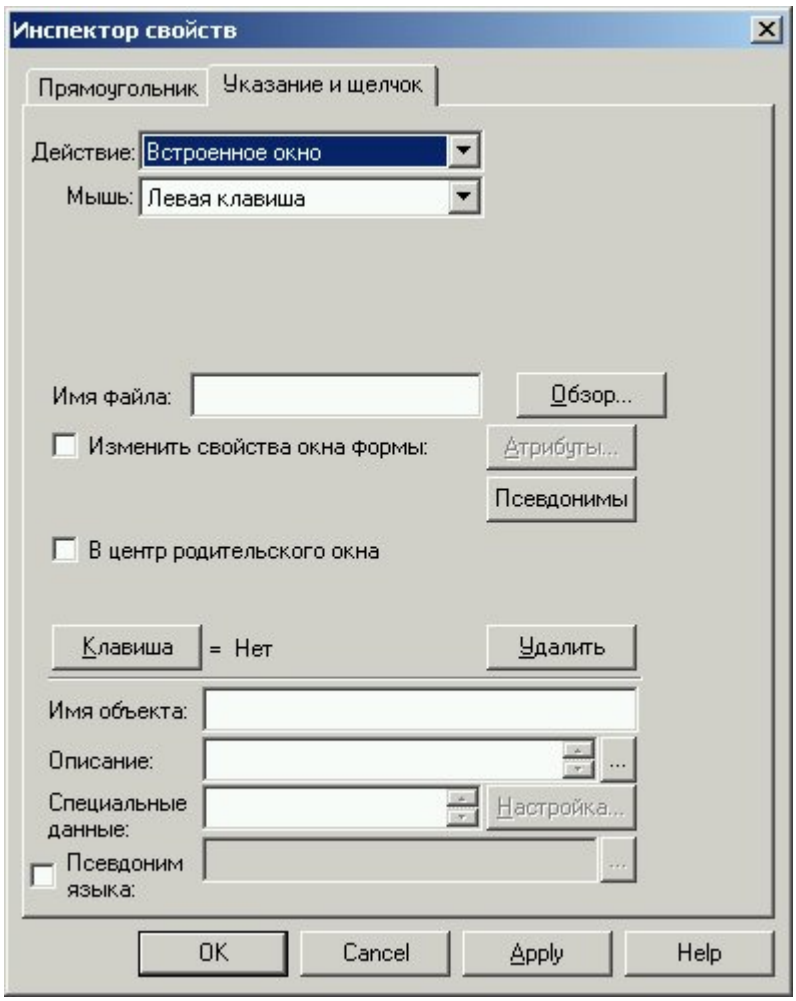
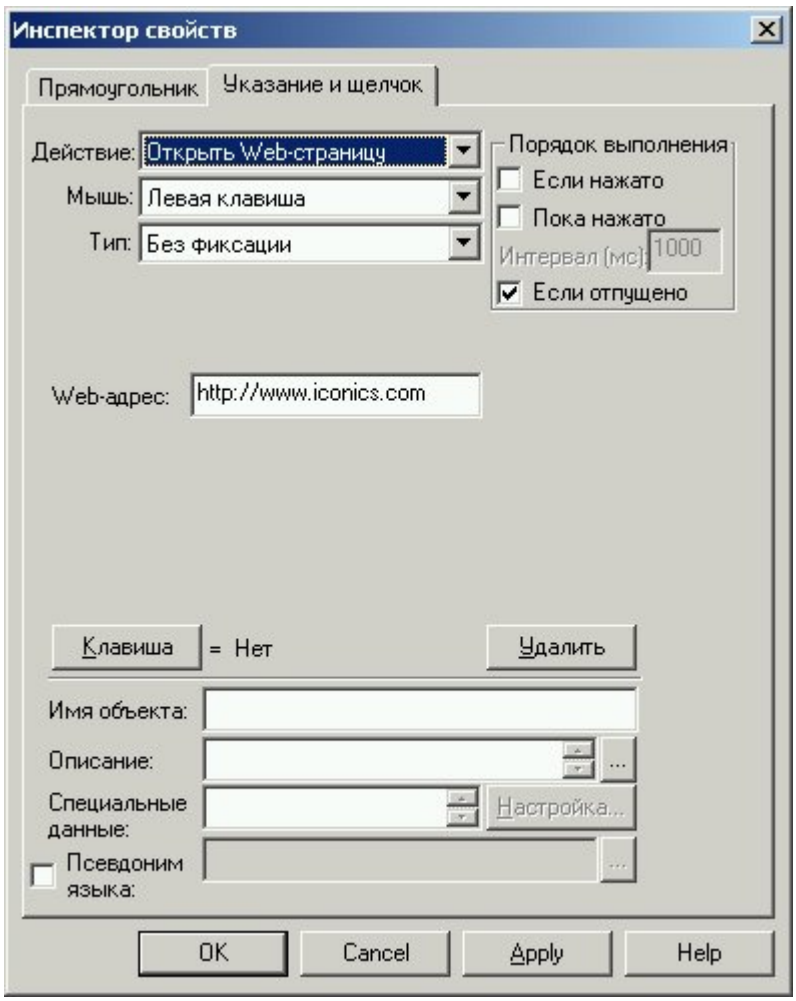


*Рис. 8-13. Параметры страницы **Указание и щелчок** в **Инспекторе свойств** при выборе операции Встроенное окно*

5. Действие *Запустить приложение*

Данная операция в режиме *Исполнение* приводит к запуску приложения, имя исполняемого файла которого указано в поле **Имя файла** страницы **Указание и щелчок** в **Инспекторе свойств**.

6. Действие *Открыть Web – страницу*



Данная операция в режиме *Исполнение* открывает Web – сайт или URL – адрес, введенный в поле **Web-адрес**. Web-страница будет открыта в браузере, используемом в системе по умолчанию.

7. Действие *Закрыть окно*

Данная операция в режиме *Исполнение* приводит к тому, что окно экранной формы GraphWorX32, в котором выполнено действие *Указание и щелчок*, будет закрыто.

8. Действие *Переключить значение*

Данная операция в режиме *Исполнение* предназначена для переключения значений источника данных, указанного в поле **Источник данных** страницы **Указание и щелчок** в **Инспекторе свойств**, показанной на рис. 8-14.

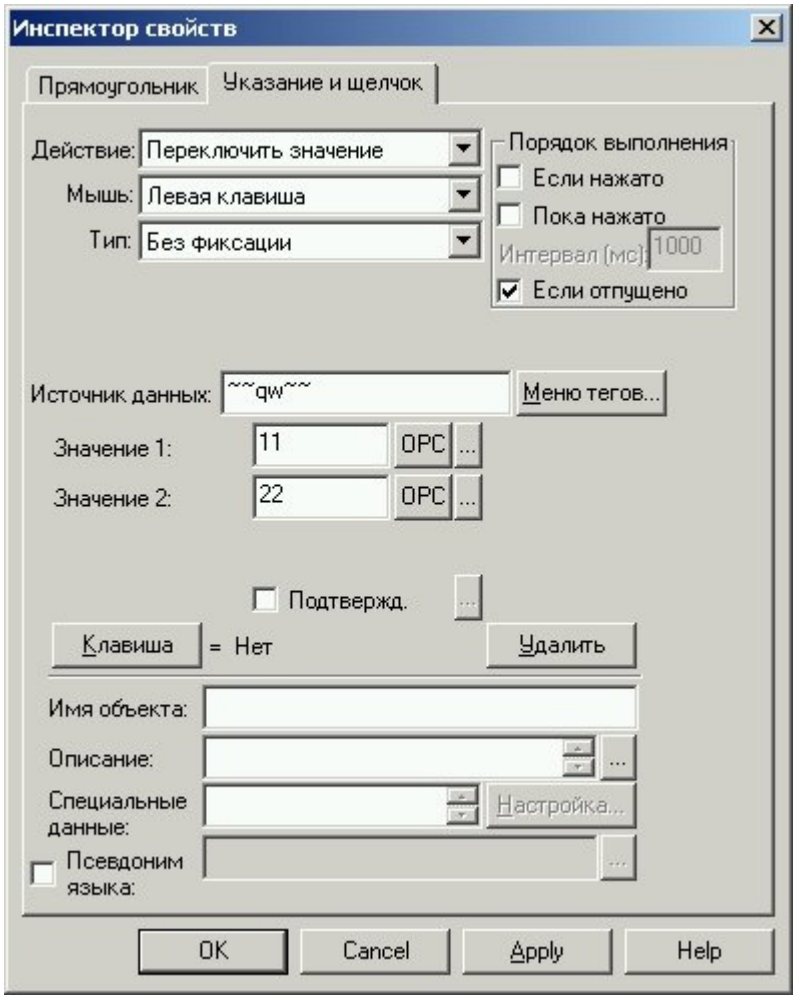


*Рис. 8-14. Параметры страницы **Указание и щелчок** в **Инспекторе свойств** при выборе операции Переключить значение*

Как указывалось выше, поле **Источник данных** служит для ввода имени тега, локальной переменной, псевдонима или выражения. Выбранный источник данных при выполнении первой части операции *Указание и щелчок* будет принимать значение, заданное в поле **Значение 1**, а при выполнении второй части – значение, заданное в поле **Значение 2**.
Для операций *Указание и щелчок* типа *Зависимая* и *С фиксацией* становится доступным флажок **Принимать начальное состояние Нажато**. Если данный флажок отмечен, при переходе в режим *Исполнение* "кнопка", реализуемая графическим объектом, к которому применено данное действие, будет находиться в нажатом состоянии.
Если отмечен флажок **Следить за значением**, то состояние графического объекта, к которому применено действие *Указание и щелчок -- Переключить значение*, будет соответствовать текущему значению источника данных. Например, если значение переменной ~~qw~~ переключается щелчком на объекте *Кнопка* путем использования действия *Указание и щелчок -- Переключить значение*, и указанная переменная была изменена щелчком на другом графическом объекте, к которому применено аналогичное действие, то объект *Кнопка* перейдет в состояние, соответствующее новому значению ~~qw~~.
Если отмечен флажок **Подтвержд.**, то будут использоваться подтверждающие сообщения при вводе пользователем значений. Чтобы воспользоваться Конфигуратором подтверждающих сообщений, нажмите кнопку **...**. Для более подробной информации, обратитесь к пункту Конфигуратор подтверждающих сообщений.

8. Действие *Передать значение*

Данная операция в режиме *Исполнение* позволяет передавать значения в источник данных, указанный в поле **Источник данных** страницы **Указание и щелчок** в **Инспекторе**

свойств, показанной на рис. 8-15. Количество и состав параметров страницы свойств зависит от выбранной в группе параметров **Порядок исполнения** последовательности выполнения операции *Указание и щелчок*.

Таблица 8-8

Параметр	Описание
Источник данных	Служит для ввода имени тега, локальной переменной, псевдонима или выражения, которому будут передаваться значения в процессе выполнения операции *Указание и щелчок*.
Значение (если нажато, пока нажато, отпущено)	Указанные поля предназначены для ввода значений, передаваемых источнику данных на каждой стадии выполнения операции *Указание и щелчок*.
Подтверждение	Определяет подтверждающее сообщение, которое будет вызываться GraphWorX32 при вводе пользователем значений. После установки флажка, можно воспользоваться Конфигуратором подтверждающих сообщений. Для этого нужно нажать на кнопку ... Для более подробной информации, обратитесь к пункту Конфигуратор подтверждающих сообщений.
Клавиша	Данная кнопка позволяет задать “горячую” клавишу на клавиатуре, нажатие которой активизирует операцию *Указание и щелчок*.

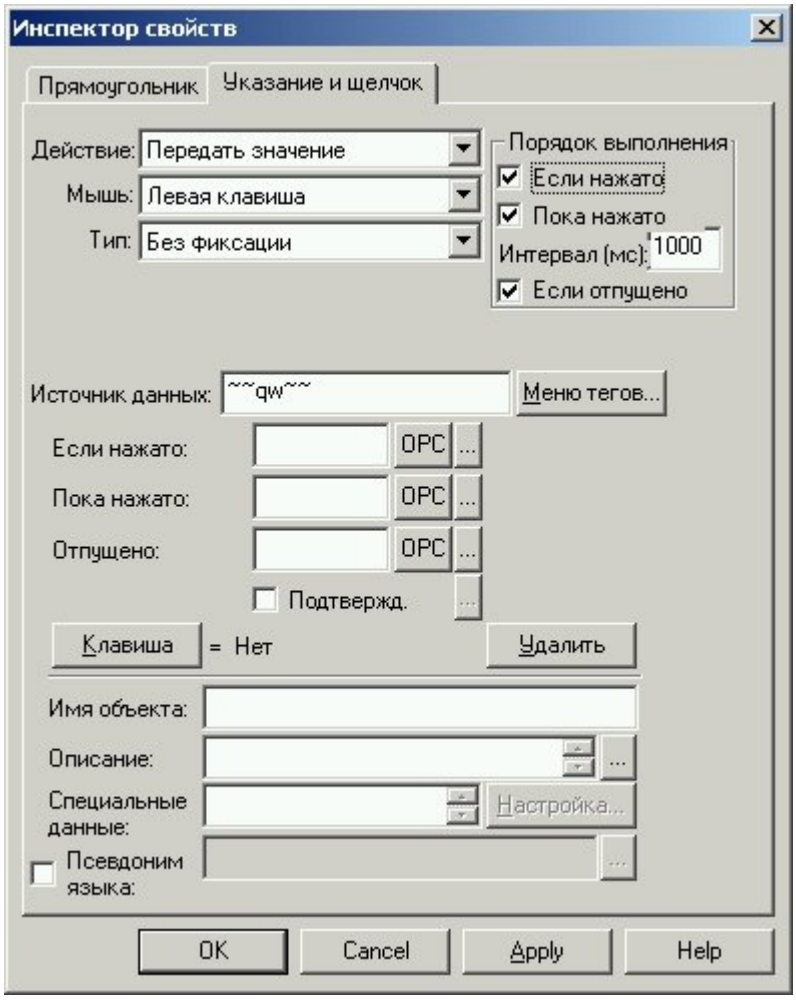


*Рис.8-15. Параметры страницы **Указание и щелчок** в **Инспекторе свойств** при выборе операции Передать значение*

9. Действие *Настройка псевдонимов*

Данная операция в режиме *Исполнение* позволяет произвести переопределение псевдонимов. Нажатие кнопки **Псевдонимы** страницы **Указание и щелчок** в **Инспекторе свойств** приведет к появлению диалоговой панели **Настройка конфигурации псевдонимов**, внешний вид которой показан на рис. 7-14. Описание полей диалоговой панели приведено в разделе 7 настоящего руководства.

10. Действие *Диалог настройки псевдонимов*

Данная операция в режиме *Исполнение* позволяет произвести переопределение псевдонимов в диалоговой панели, внешний вид которой показан на рис. 8-16.

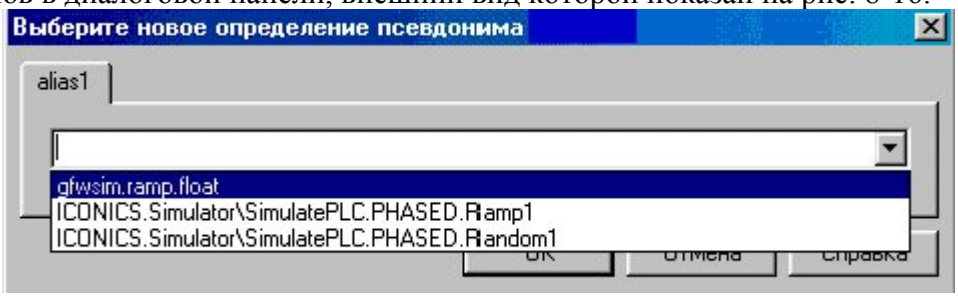


Рис. 8-16. Диалог настройки псевдонимов в режиме Исполнение

Нажатие кнопки **Псевдонимы** страницы **Указание и щелчок** в **Инспекторе свойств** приведет к появлению диалоговой панели **Настройка конфигурации псевдонимов**, внешний вид которой показан на рис. 7-14. Описание полей диалоговой панели приведено в разделе 7 настоящего руководства.

11. Действие *Скрыть слой*

Данная операция в режиме *Исполнение* позволяет выключить отображение слоя, имя которого введено в поле **Имя слоя** страницы **Указание и щелчок** в **Инспекторе свойств**. Страница **Указание и щелчок** в **Инспекторе свойств** при выборе операции *Скрыть слой* показана на рис. 8-17.

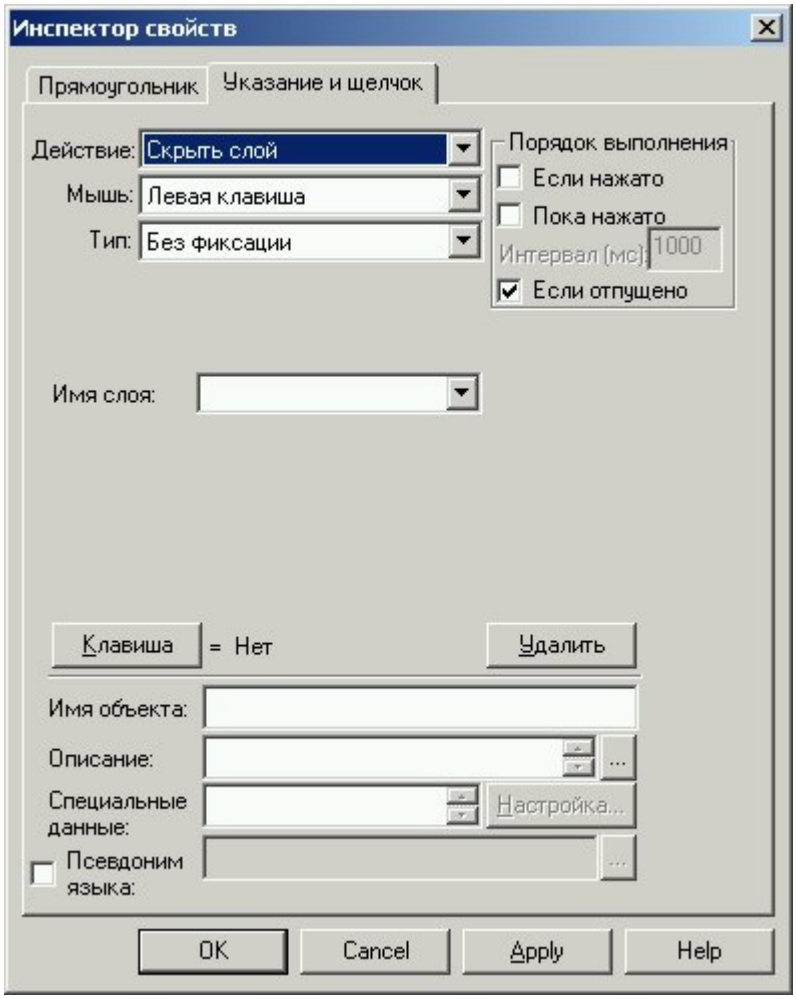


*Рис. 8-17. **Инспектор свойств -- Страница Указание** и щелчок при выборе операции Скрыть слой*

12. Действие *Показать слой*

Данная операция в режиме *Исполнение* позволяет включить отображение слоя, имя которого введено в поле **Имя слоя** страницы **Указание и щелчок** в **Инспекторе свойств**.

13. Действие *Переключить отображение слоя*

Данная операция в режиме *Исполнение* позволяет включить или отключить отображение слоя, имя которого введено в поле **Имя слоя** страницы **Указание и щелчок** в **Инспекторе свойств**. Если перед первым выполнением действия *Указание и щелчок -- Переключить* отображение слоя, слой отображался, то после выполнения указанного действия слой будет скрыт. Следующий щелчок на объекте приведет к включению отображения слоя.

14. Действие *Выполнить сценарий VBA*

Данная операция в режиме *Исполнение* позволяет выполнить три различных типа сценарных процедур: VBA, VBScript and JScript. Имя процедуры указывается в поле **Сценарий** страницы **Указание и щелчок** в **Инспекторе свойств**, внешний вид которой показан на рис. 8-18.

Если имя сценария вводится вручную, необходимо убедиться в том, что процедура с вводимым именем присутствует в среде разработки процедур VBA GraphWorX32. Имя модуля формируется с использованием следующего соглашения:

GwxИмя_процедуры_Main

Если процедура не существует в момент настройки действия, следует нажать кнопку **Создать** на странице свойств **Указание и щелчок**. На экран монитора будет выведена

диалоговая панель **Мастер VBA-сценариев**, позволяющая автоматически включить новую процедуру, связанную с настраиваемым действием,. Для получения более подробной информации обратитесь к разделу 9 настоящего руководства.

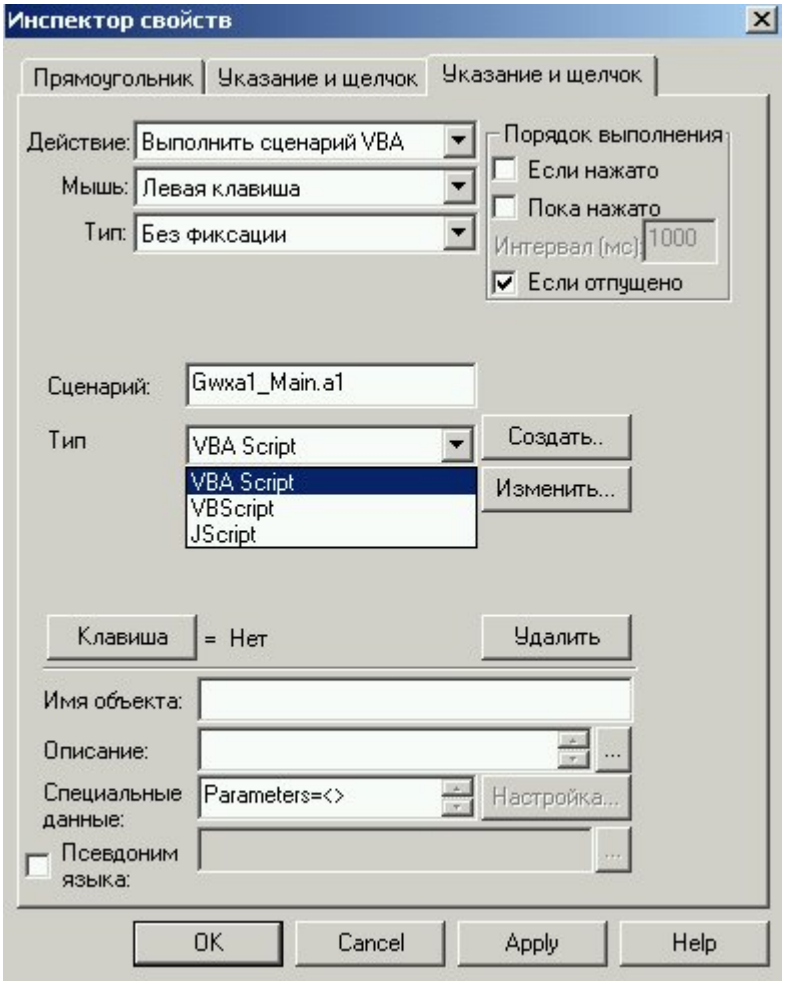


*Рис. 8-18. Параметры страницы **Указание и щелчок** в **Инспекторе свойств** при выборе операции Выполнить сценарий VBA*

После создания сценарной процедуры в *Мастере VBA-сценариев*, нажмите кнопку **Изменить** для ввода исходного текста процедуры в *Редакторе VBA*.

15. Действие *Специальная команда*

Данная операция позволяет выполнить специальную пользовательскую функцию. Функция может быть реализована в виде исполняемого файла (.EXE) или библиотеки динамической компоновки (.DLL).

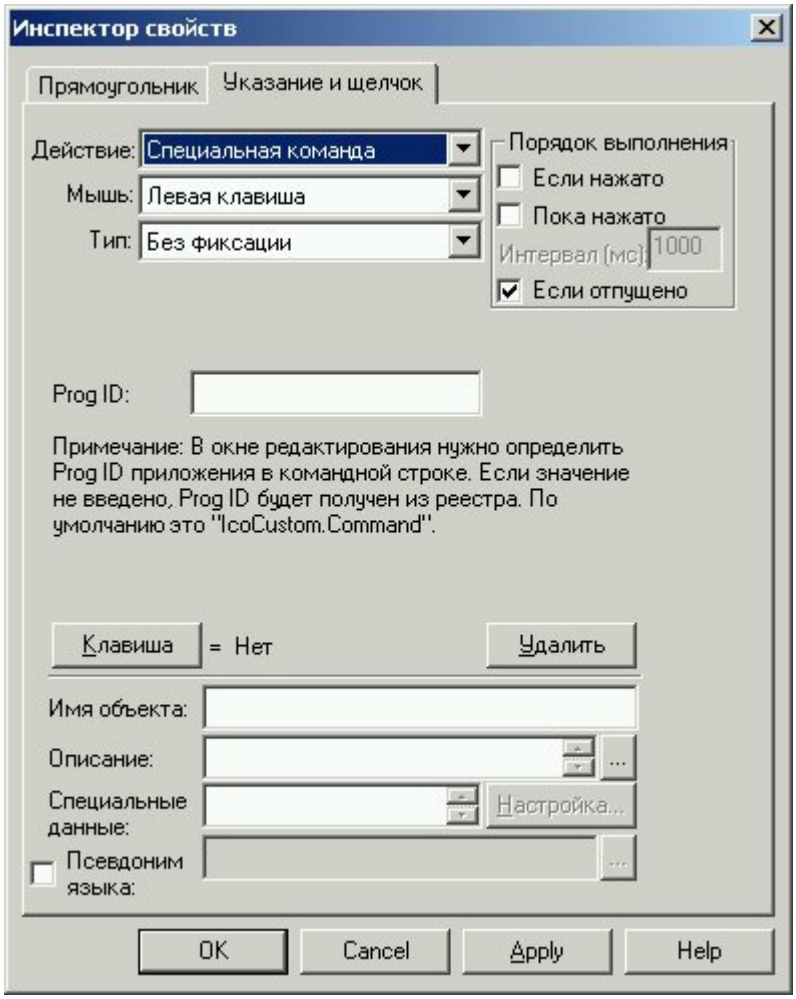


*Рис. 8-19. Параметры страницы **Указание и щелчок** в **Инспекторе свойств** при выборе операции Специальная команда*

16. Действие *Всплывающее меню*

Данная операция позволяет конструировать всплывающие (контекстные) многоуровневые меню для графических объектов экранной формы.

Диалоговая панель **Инспектор свойств** при выборе операции *Всплывающее меню* показана на рис. 8-20.

Для создания всплывающего меню:

1. Выделите графический объект, к которому должно быть привязано всплывающее меню
2. В инструментальной панели *Динамика* выберите действие *Указание и щелчок* или выберите команду **Указание и щелчок мышью** подменю **Действия** меню **Динамика** GraphWorX32.
3. В появившейся диалоговой панели **Инспектор свойств**, открытой на странице **Указание и щелчок**, выберите операцию *Всплывающее меню* в выпадающем списке **Действие**.
4. Нажмите кнопку **Меню**. На экран монитора будет выведена диалоговая панель **Редактор меню**, показанная на рис. 8-21.
5. Для создания строки меню нажмите кнопку **Добавить строку** диалоговой панели **Редактор меню**. Если предполагается сформировать многоуровневое меню, нажмите кнопку **Добавить подменю**, после чего нажмите кнопку **Дублировать** для размножения команд подменю. Далее поочередно выделите строки созданного меню, нажмите кнопку **Переименовать** и введите желаемые имена команд меню.
6. Для назначения действия каждой команде меню, выделите ее имя и нажмите кнопку **Изменить** диалоговой панели **Редактор меню**. На экран монитора будет

выведена диалоговая панель **Указание и щелчок**, показанная на рис. 8-22, которая позволяет определить действие по каждой команде меню.

Для упорядочивания команд в созданном меню пользуйтесь операцией "*Перенести-и-Оставить*" в **Редакторе меню**.

Для того, чтобы всплывающее меню появлялось в позиции курсора, отметьте флажок **Появление в позиции курсора** на странице **Указание и щелчок** в **Инспекторе свойств**.

Пример использования всплывающего меню в режиме *Исполнение* показан на рис. 8-23.

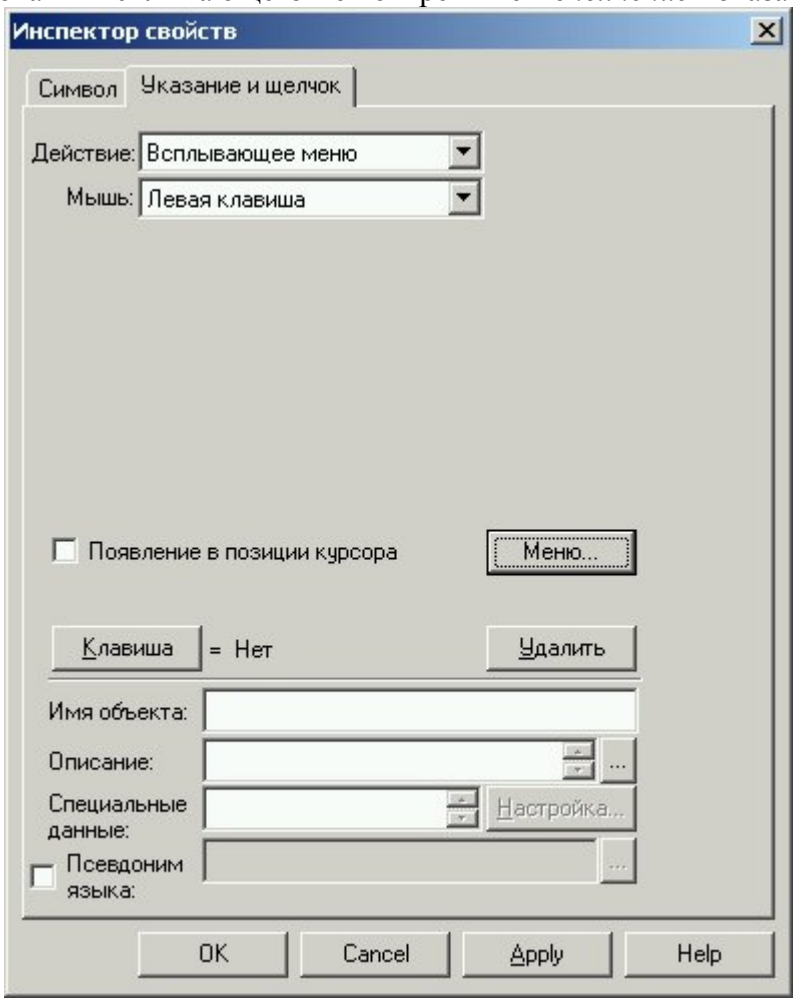


*Рис. 8-20. **Инспектор свойств** объекта при выборе операции Указание и щелчок -- Всплывающее меню*

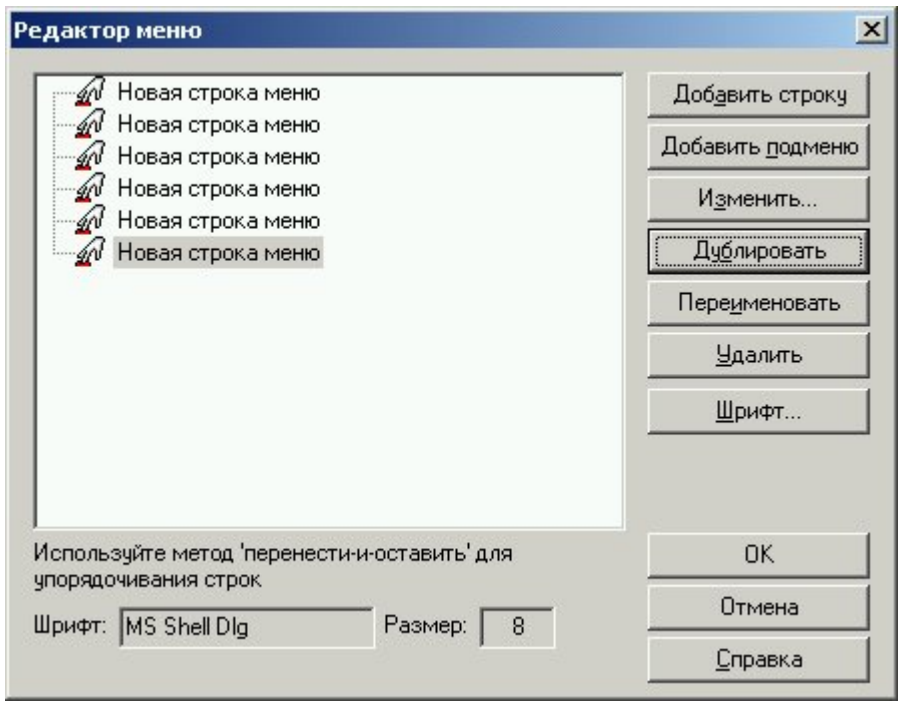


Рис. 8-21. Диалоговая панель настройки параметров всплывающего меню

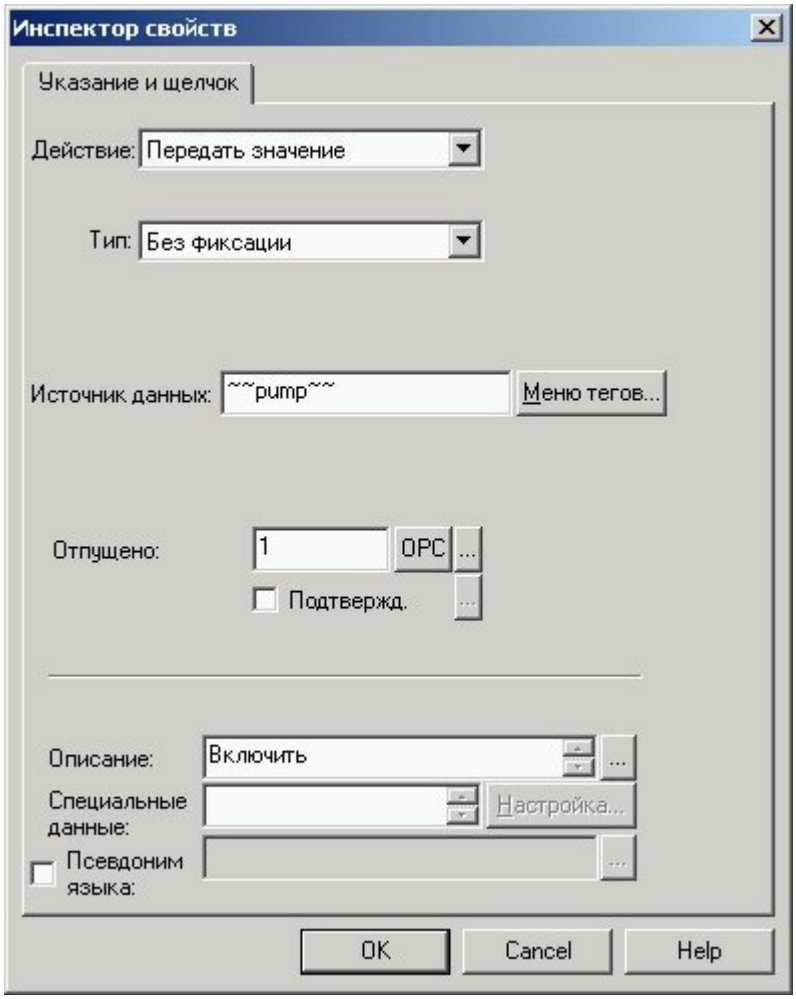


Рис. 8-22. Диалоговая панель назначения действия по выбору команды меню

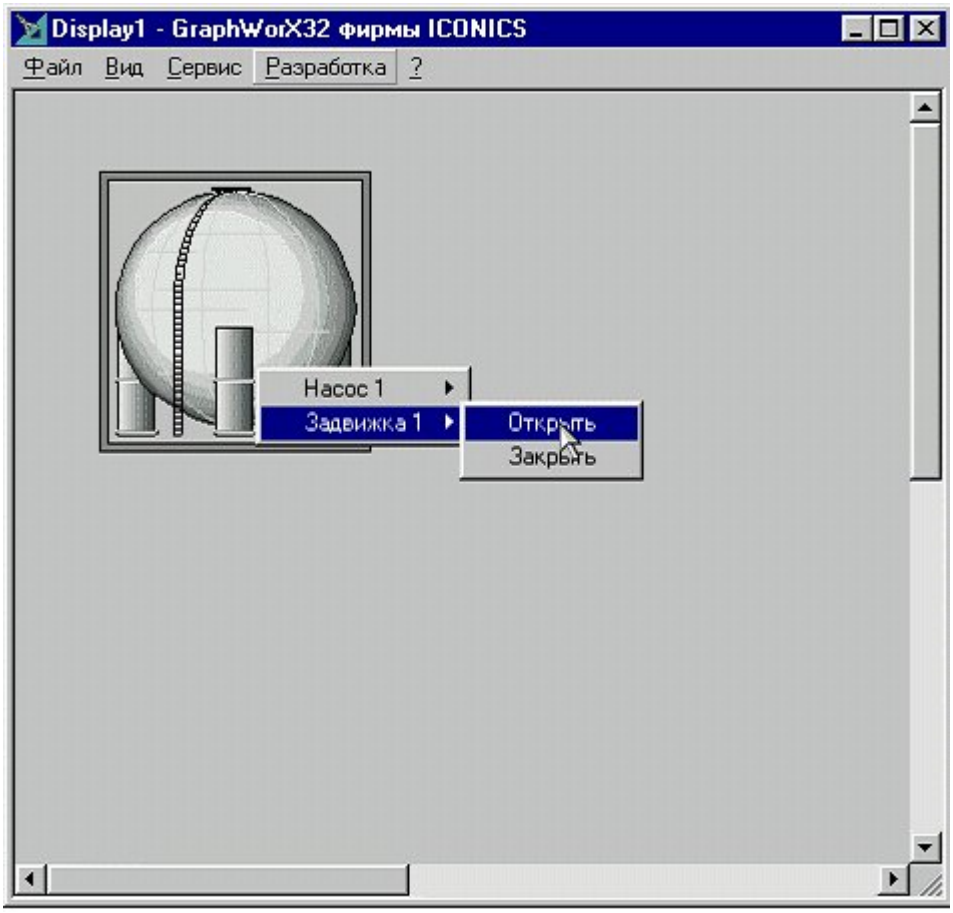
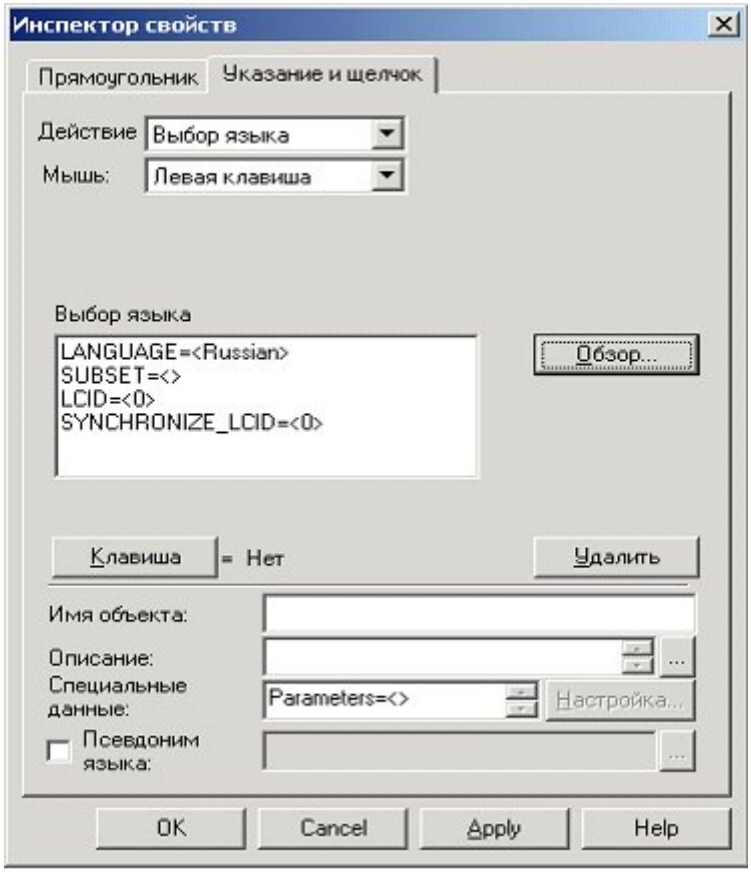


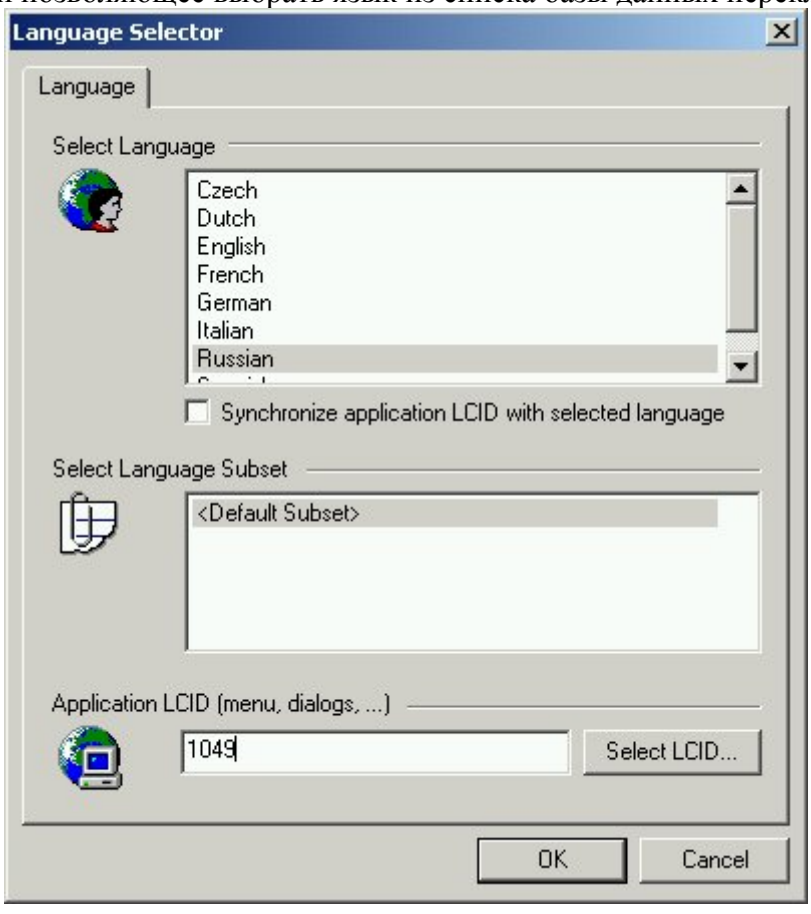
Рис. 8-23. Использование всплывающего (контекстного) меню

16. Действие *Выбор языка*

Данная операция позволяет выбирать язык по нажатию кнопки.

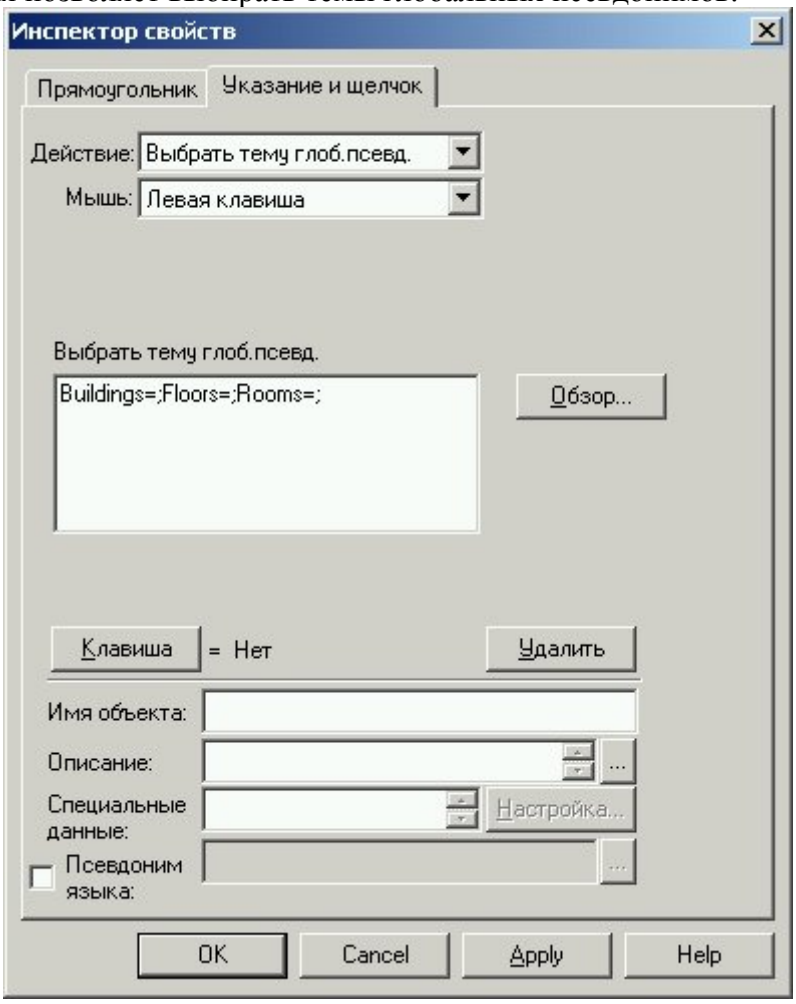


Нажатие на кнопке **Обзор** открывает диалоговое окно **Выбор языка**, показанное на рисунке ниже, и позволяющее выбрать язык из списка базы данных переключения языков.

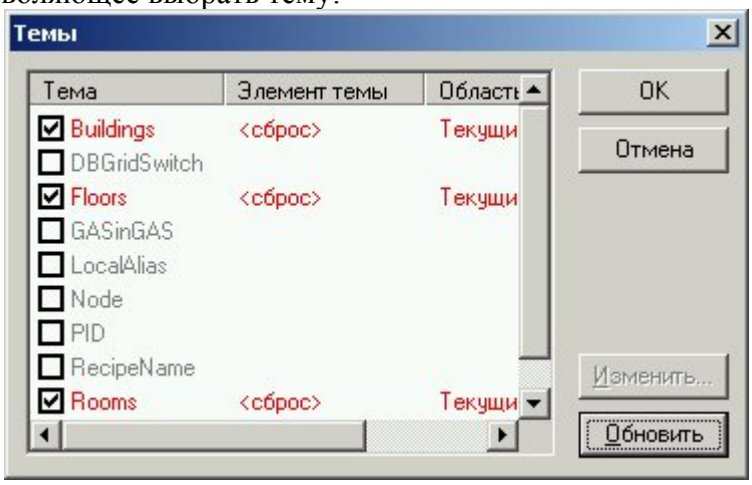


16. Действие *Выбор темы глобальных псевдонимов.*

Данная операция позволяет выбирать темы глобальных псевдонимов.



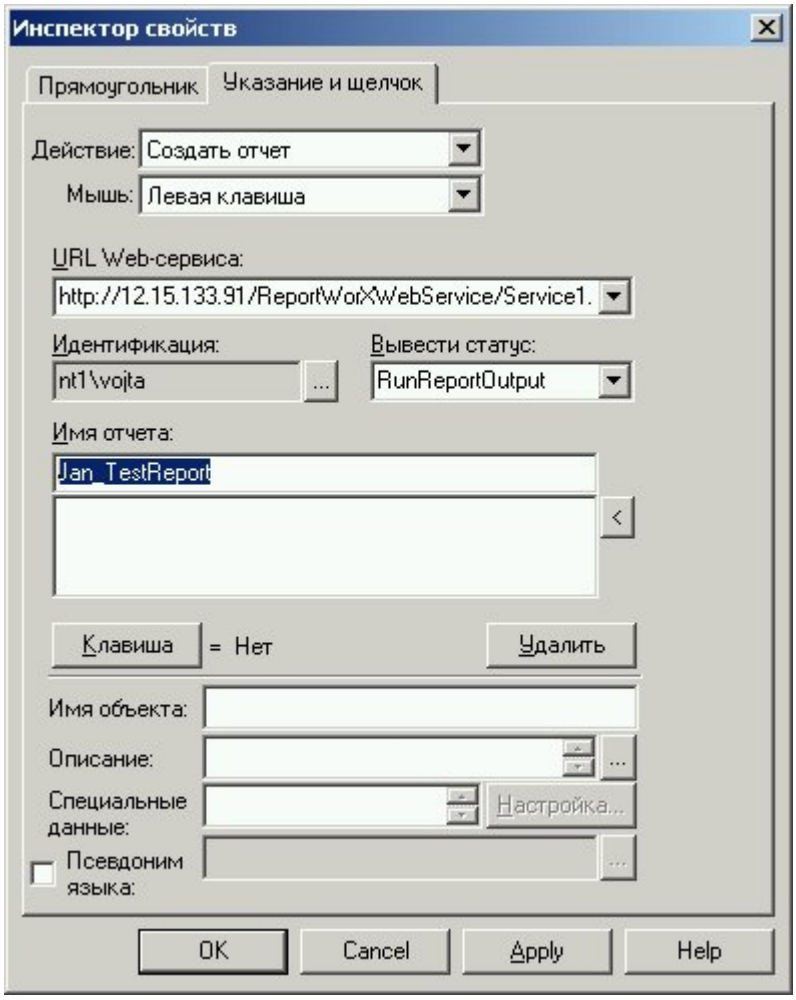
При нажатии кнопки **Обзор** открывается диалоговое окно Навигатора тем глобальных псевдонимов, позволяющее выбрать тему:



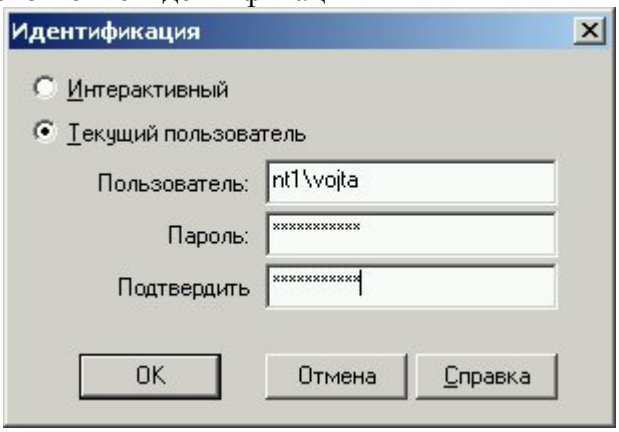
Для изменения свойств темы следует выделить тему и нажать кнопку **Изменить**. При этом будет открыт **Редактор тем**, где можно определить элементы и области темы. Дополнительная информация содержится в описании Доп. настроек режима исполнения.

17. *Действие Создать отчет.*

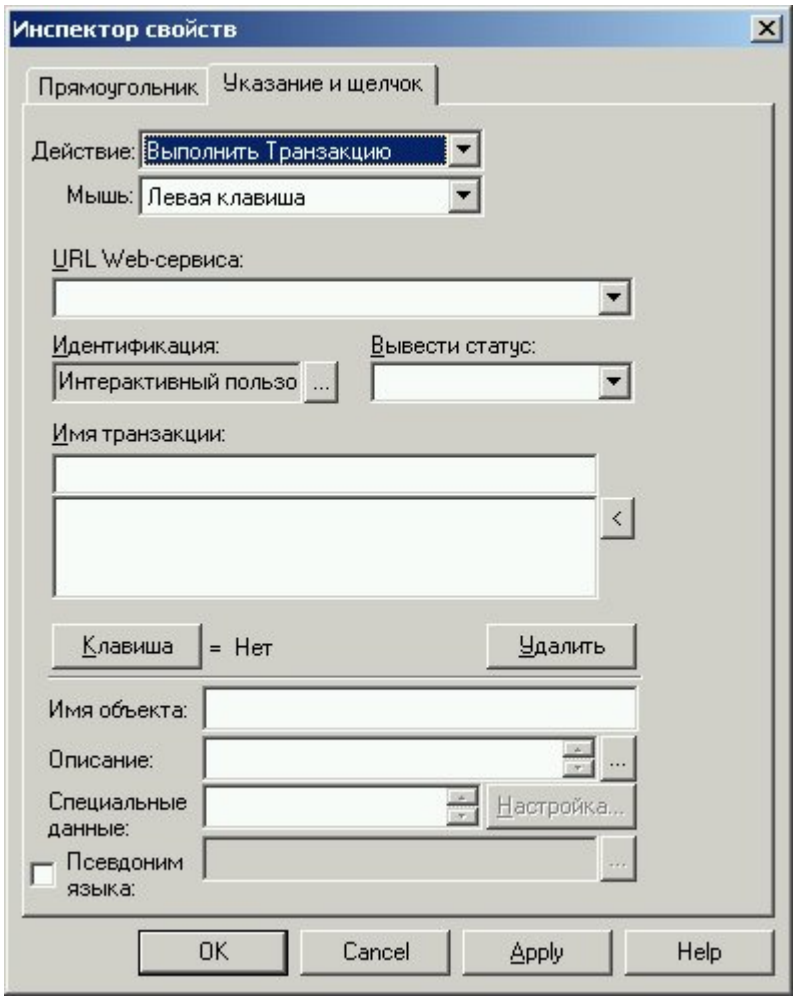
Данная операция позволяет запускать отчет ReportWorX в режиме исполнения.
Для более подробной информации, просьба обратиться к документации ReportWorX на CD Iconics- BizViz.



В поле **URL Web-сервиса** вводится адрес сервера ReportWorX.
В поле **Вывести статус** из выпадающего списка выбирается текстовый объект, для которого будет установлено имя объекта. Если объект указан, на него будет выведен статус отчета.
Имя отчета, который должен быть активирован, вводится в поле **Имя отчета** или выбирается из списка. Для обновления списка, нажмите кнопку **<**.
Если для доступа к серверу ReportWorX необходимо ввести имя пользователя и пароль, нажмите кнопку **…** справа от поля **Идентификация** и введите необходимую информацию в появившемся диалоговом окне Идентификация

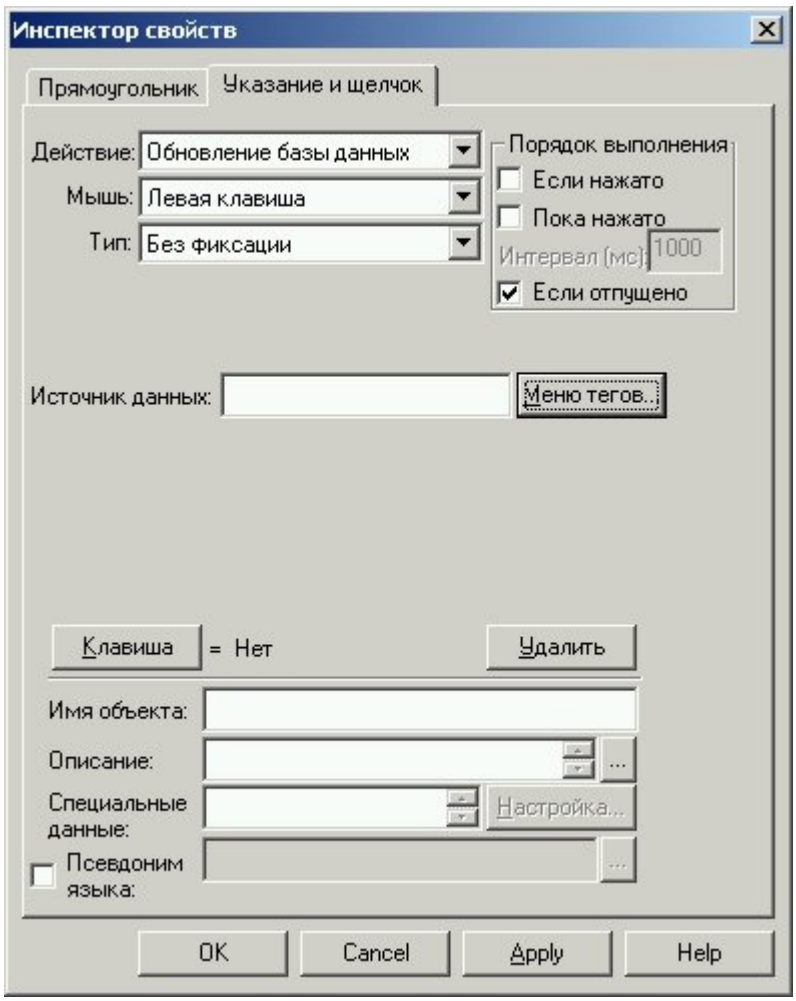


18. *Действие Выполнить Транзакцию.*
Данная операция позволяет выполнять транзакцию BridgeWorX в режиме исполнения. Для более подробной информации, просьба обратиться к документации BridgeWorX на CD Iconics- BizViz.

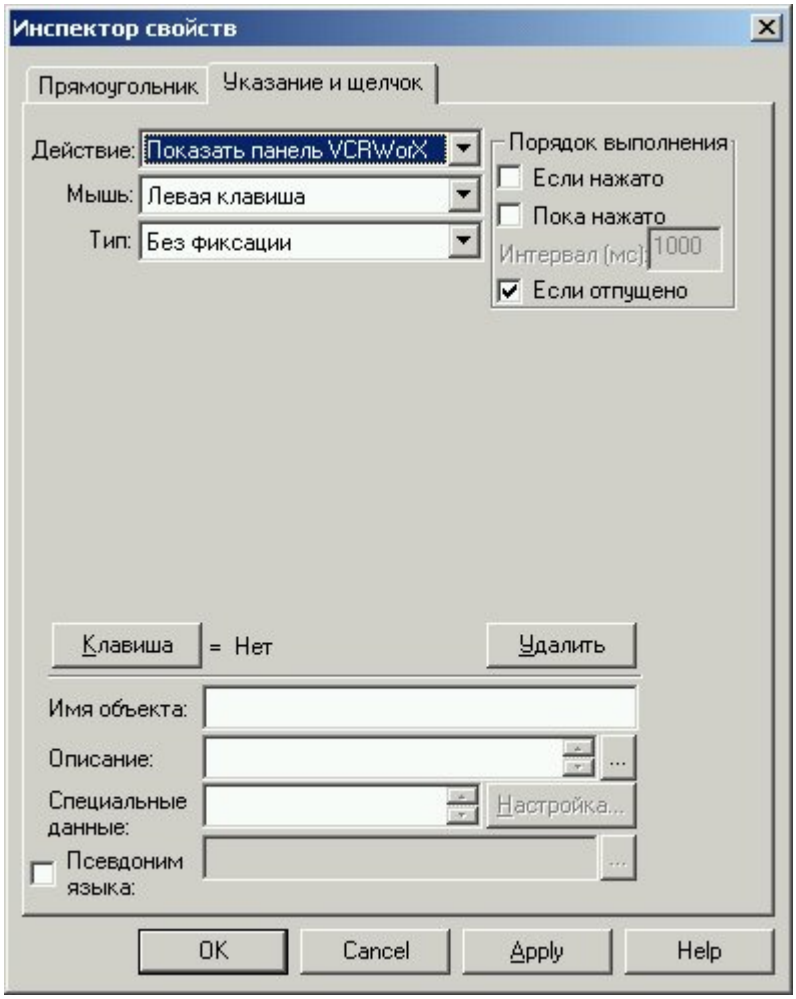


В поле **URL Web-сервиса** вводится адрес сервера BridgeWorX.
В поле **Вывести статус** из выпадающего списка выбирается текстовый объект, для которого установлено имя объекта. Если объект указан, на него будет выведен статус транзакции.
Имя транзакции, которая должна быть активирована, вводится в поле **Имя транзакции**.
Для обновления списка имен транзакций, нажмите кнопку <.
Если для доступа к серверу BridgeWorX необходимо ввести имя пользователя и пароль, нажмите кнопку **…** справа от поля **Идентификация** и введите необходимую информацию в появившемся диалоговом окне Идентификация.

19. *Действие Обновление базы данных.*
Данная операция позволяет обновить сервер данных путем записи любого значения в тег данных сервера.
Для определения источника данных можно воспользоваться Универсальным навигатором данных: нажмите кнопку **Меню тегов** и выберите тег из Database Access. Для более поддробной информации, воспользуйтесь справочной документацией Data Mining Configurator.



20. *Показать/Скрыть панель VCRWorX*
Данная операция позволяет показать/скрыть панель *VCRWorX* воспроизведения архивных данных в режиме исполнения. Для более подробной информации, воспользуйтесь справочной документацией VCRWorX.



8.4 Индикаторы

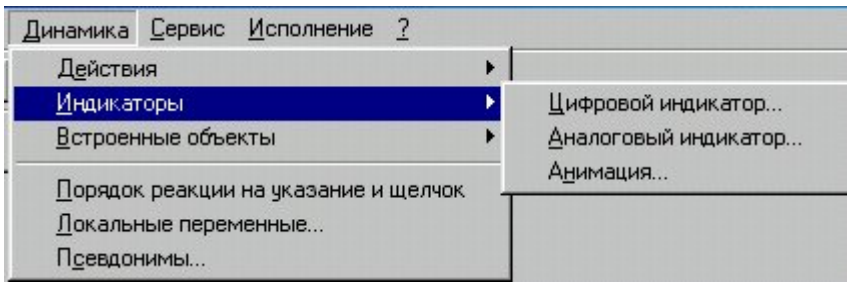


*Рис. 8-24. Подменю **Индикаторы***

Индикаторы предназначены для отображения графических объектов выбранной группы, когда значения связанного источника данных равны заданным или находятся в установленных поддиапазонах. Имеются 3 типа индикаторов:

- *Аналоговый индикатор* – один из нескольких графических объектов группы, к которым применен данный тип динамики, отображается в месте своего создания в рабочей области экранной формы, когда значение связанного источника данных находится в пределах выбранного поддиапазона. При этом остальные объекты группы не отображаются. Переход значения источника данных в другой поддиапазон приводит к скрытию ранее отображаемого объекта и включению отображения другого графического объекта группы, которому соответствует текущий поддиапазон. Количество поддиапазонов равно количеству графических объектов в группе. Единицы измерения в поддиапазонах – проценты от полной шкалы значения связанного источника данных.

- *Цифровой индикатор* – графические объекты группы, к которым применен данный тип динамики, отображаются в месте своего создания в рабочей области экранной формы, когда связанные источники данных находятся в заданном логическом состоянии *ИСТИНА* (*True*) или *ЛОЖЬ* (*False*). Если источники данных принимают противоположное состояние, установленное во время создания индикатора, то это приводит к скрытию ранее отображаемого объекта. Количество источников данных равно количеству графических объектов в группе.
- *Анимация* – каждый графический объект группы, к которой применен указанный тип динамики, рассматривается в виде кадра с заданным номером в последовательности отображения. Автоматическая смена кадров с заданным периодом осуществляется в режиме *Исполнение*, когда связанный источник данных находится в выбранном логическом состоянии *ИСТИНА* (*True*) или *ЛОЖЬ* (*False*).

Цифровой индикатор



Динамическое соединение *Цифровой индикатор* позволяет связывать отдельные графические объекты из объединенных в группу с соответствующими источниками данных, имеющими булев тип. В режиме *Исполнение* переход значения связанного источника данных в состояние *ИСТИНА* (*True*) (или *ЛОЖЬ* (*False*)) приводит к появлению соответствующего графического объекта на экране.
В каждый момент времени может отображаться только один графический объект из нескольких, связанных с одним и тем же источником данных. Например, если два объекта связаны с одним и тем же источником данных, то переход указанного источника в выбранное логическое состояние приведет к появлению объекта, который находится ближе к вершине списка связанных объектов, чем второй объект.
Для создания цифрового индикатора:

1. Выделите два или более графических объектов.
2. Выберите строку **Цифровой индикатор** подменю **Индикаторы** меню **Динамика** или нажмите кнопку *Цифровой индикатор*. На экран монитора будет выведена диалоговая панель **Инспектор свойств** с выбранной страницей **Цифровой индикатор**, внешний вид которой показан на рис. 8-25.

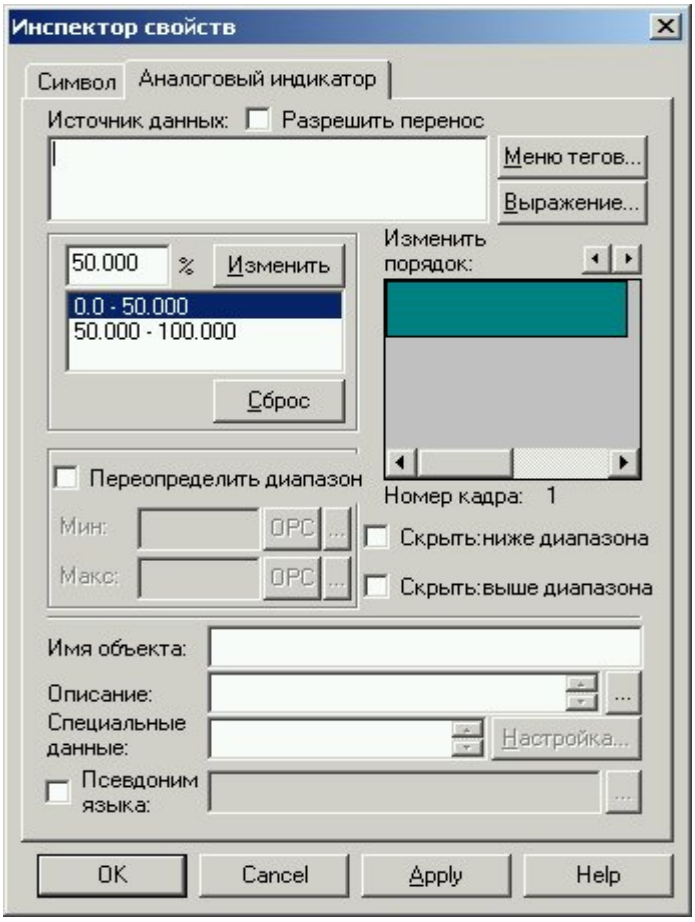


Рис. 8-25. ***Инспектор свойств*** *– страница* ***Цифровой индикатор***

3. Выполните настройку параметров цифрового индикатора. Описание полей диалоговой панели приведено в табл. 8-11.

Таблица 8-11

Поле	Описание
Источник данных	Данное поле предназначено для ввода имени источника данных (тега OPC, переменной, константы или псевдонима), булево значение которого будет определять наличие на экране связанного графического объекта. Верхняя область поля содержит вводимое или текущее редактируемое имя источника данных или выражение. Нижняя область содержит список источников данных, номера элементов которого задают последовательность отображения связанных графических объектов.
Меню тегов	Данная кнопка открывает Универсальный навигатор данных, позволяющий выбрать соединение с тегами OPC, переменными и псевдонимами. Для более подробной информации, обратитесь к Справочной документации Unified Data Browser.
Выражение	Данная кнопка открывает Редактор выражений. Для более подробной информации, обратитесь к Справочной документации Expression_Editor.
Изменить	Данная кнопка предназначена для ввода или изменения существующего источника данных, выбранного в списке поля **Источник данных**.
Удалить	Данная кнопка предназначена для удаления текущего выбранного источника данных в списке.
Номер объекта	Данное поле содержит номер графического объекта в последовательности отображения, который связан с текущим выбранным источником данных в списке.
Изменить порядок	Данные кнопки служат для изменения нумерации объектов в последовательности отображения.
Показать объект, если True	При установке данного переключателя выбранный объект будет появляться на экране, если связанный с ним источник данных находится в состоянии логической 1.

Показать объект, если False	При установке данного переключателя выбранный объект будет появляться на экране, если связанный с ним источник данных находится в состоянии логического нуля.

4. Установите динамические соединения для объектов, образующих цифровой индикатор.
5. По завершении нажмите кнопку **OK** или нажмите клавишу Enter.

Аналоговый индикатор



Динамическое соединение *Аналоговый индикатор* позволяет связывать отдельные графические объекты из объединенных в группу со значением источником данных. В режиме *Исполнение* каждый объект группы отображается в месте своего создания в рабочей области экранной формы, когда значение связанного с группой источника данных находится в пределах поддиапазона, соответствующего данному объекту.

Для создания аналогового индикатора:

1. Выделите два или более графических объектов.
2. Выберите строку **Аналоговый индикатор** подменю **Индикаторы** меню **Динамика** или нажмите кнопку *Аналоговый индикатор* панели инструментов *Динамика*. На экран монитора будет выведена диалоговая панель **Инспектор свойств** с выбранной страницей **Аналоговый индикатор**, внешний вид которой показан на рис. 8-26.
3. Введите имя тега, переменной или псевдоним в поле **Источник данных** или воспользуйтесь Универсальным навигатором данных (нажав на кнопку **Меню тегов)**. Чтобы воспользоваться **Редактором выражений**, нажмите кнопку **Выражение**.
4. Количество поддиапазонов, на которое разбивается полный диапазон значений связанного источника данных, равно количеству графических объектов группы. Выберите поддиапазон в списке **%.** Верхнее граничное значение диапазона будет отображено в поле, расположенном слева от символа '**%**'.
5. С помощью кнопок **Изменить порядок** выберите графический объект группы, который должен появляться на экране при попадании значения связанного источника данных в пределы выбранного диапазона.
6. Повторите действия 4–5 для связи остальных поддиапазонов с графическими объектами группы.
7. При необходимости переопределите диапазон значений связанного источника данных, используя поля **Мин.**, **Макс** группы параметров **Переопределить диапазон**.
8. Установите флажок **Скрыть, если ниже диапазона** или **Скрыть, если выше диапазона**, что приведет к скрытию всей группы объектов в случае, когда значение связанного источника данных меньше нижней границы или, соответственно, больше верхней границы диапазона, заданного в группе параметров **Переопределить диапазон**.
9. По завершении настройки динамического соединения нажмите кнопку **OK** диалоговой панели.

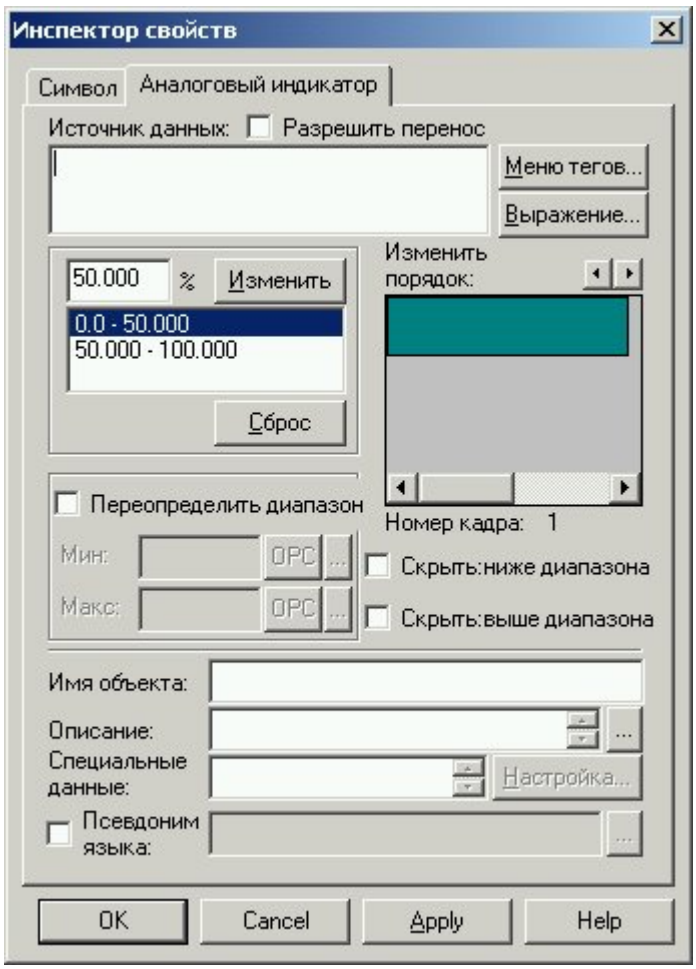


Рис. 8-26. ***Инспектор свойств*** *– страница* ***Аналоговый индикатор***

Анимация



Каждый графический объект группы, к которой применен указанный тип динамики, рассматривается в виде кадра с заданным номером в последовательности отображения. Автоматическая смена кадров с заданным периодом осуществляется в режиме *Исполнение*, когда связанный источник данных находится в выбранном логическом состоянии *ИСТИНА* (*True*) или *ЛОЖЬ* (*False*). Последовательное отображение объектов (кадров) происходит до тех пор, пока связанный источник данных не примет противоположное заданному для анимации логическое состояние.
Для создания динамического соединения *Анимация*:

1. Выделите два или более графических объектов.
2. Выберите строку **Анимация** подменю **Индикаторы** меню **Динамика** или нажмите кнопку *Анимация* панели инструментов *Динамика*. На экран монитора будет выведена диалоговая панель **Инспектор свойств** с выбранной страницей **Анимация**, внешний вид которой показан на рис. 8-27.
3. Введите имя тега, переменной или псевдоним в поле **Источник данных** или воспользуйтесь Универсальным навигатором данных (нажав на кнопку **Меню тегов)**. Источник данных должен иметь булевы значения.

   Чтобы воспользоваться **Редактором выражений**, нажмите кнопку **Выражение**.
4. Установите порядок смены графических объектов в последовательности отображения, для чего воспользуйтесь кнопками **Порядок смены кадров**. Номер в последовательности, который присваивается текущему объекту, отображаемому в области предварительного просмотра, выводится в поле **Номер кадра**.

   Нажмите правой кнопкой мыши на дочерний объект в окне **Порядок смены кадров**, и в выпадающем меню можно выбрать пункт **Инспектор свойств**

объекта или **Показать выделенное** (выделенный дочерний объект) в основной экранной форме. Можно закрыть Инспектор свойств и отредактировать или изменить дочерний объект (выделенный). Эта функция использует свойство подсимвольного редактирования и открывает символ Индикатор/Анимация для редактирования. Это позволяет добавлять, редактировать или удалять все дочерние объекты с этим символом. Чтобы закрыть подсимвольное редактирование, нажмите Esc или дважды щелкните мышью вне границ рабочего прямоугольника.

5. В поле **Период смены кадров** диалоговой панели введите период смены кадров при отображении последовательности.
6. Установите в требуемое положение переключатель **Анимация, если True/False**.
7. Определите режим отображения группы объектов, к которым применяется *Анимация*, когда связанный источник данных меняет свое логическое состояние на противоположное. При установке переключателя **Невидимый, если ВЫКЛ**, смена “активного” логического состояния на противоположное приведет к скрытию всех графических объектов группы. Если установлен переключатель **Видимый, если ВЫКЛ**, то объект группы, который будет статически отображаться при смене активного состояния на противоположное, выбирается с помощью переключателя **Текущий кадр, если ВЫКЛ**/**Первый кадр, если ВЫКЛ**. Положение **Текущий кадр, если ВЫКЛ** соответствует статическому отображению графического объекта, который в текущий момент времени выбран в области предварительного просмотра диалоговой панели. Положение **Первый кадр, если ВЫКЛ** соответствует статическому отображению графического объекта, имеющему 1-й номер в последовательности отображения.
8. По завершении настройки динамического соединения *Анимация* нажмите кнопку **OK** диалоговой панели.

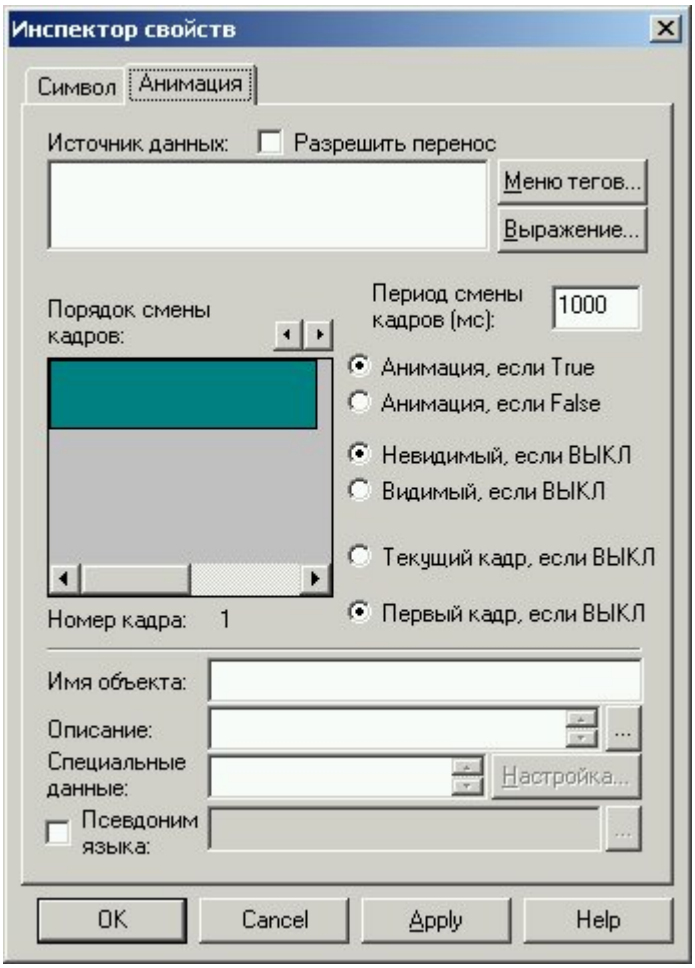


*Рис. 8-27. Диалоговая панель **Инспектор свойств**. Закладка **Анимация**.*

8.5 Встроенные динамические объекты

Встроенные динамические объекты предназначены для создания элементов и органов управления в рабочей области экранных форм. В режиме *Исполнение* оператор, воздействуя на органы управления, выполненные на базе встроенных динамических объектов, может изменять значения и логические состояния связанных источников данных. Доступ к встроенным динамическим объектам в режиме *Разработка* осуществляется в подменю **Встроенные объекты** меню **Динамика** или в правой части панели инструментов *Динамика*. Внешний вид подменю **Встроенные объекты** и соответствующих кнопок панели инструментов *Динамика* показан на рис. 8-28 – 8-29.

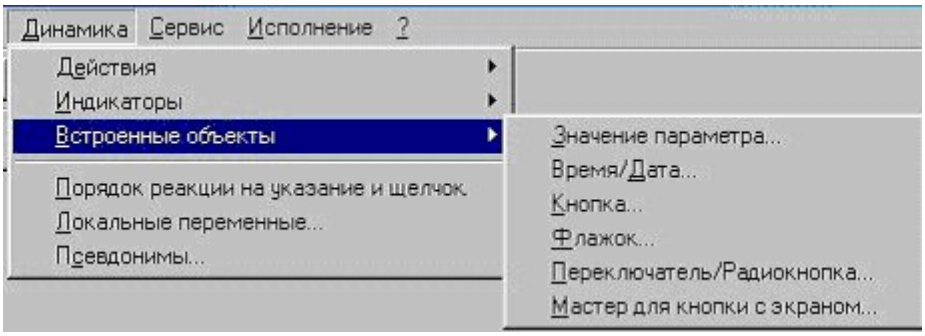


*Рис. 8-28. Подменю **Встроенные объекты** меню **Динамика***



Рис. 8-29. Кнопки доступа к встроенным динамическим объектам панели инструментов Динамика

Встроенный динамический объект *Значение параметра* (*Параметр*)



Данный динамический объект позволяет создавать в экранных формах алфавитно-цифровые индикаторы, на которых в режиме *Исполнение* отображаются значения связанных источников данных. Кроме того, указанный динамический объект может одновременно являться полем, в которое оператор вводит требуемые значения, которые должны передаваться связанному источнику данных в режиме *Исполнение*.
Обновление значений, отображаемых динамическим объектом *Параметр*, производится по извещению от OPC-сервера об изменении связанного с ним тега OPC.
Если объект одновременно используется в качестве поля ввода данных, то, помимо отображения значений связанного источника данных, имеется возможность ввода оператором и передачи значений соответствующему источнику данных.
Для создания объекта *Значение параметра*:

1. Выберите строку **Значение параметра** подменю **Встроенные объекты** меню **Динамика** или нажмите кнопку *Значение параметра* панели инструментов *Динамика*.
2. Щелкните левой клавишей мыши в точке экранной формы, в которой требуется создать динамический объект *Значение параметра*. На экран монитора будет выведена диалоговая панель **Инспектор свойств** с выбранной закладкой **Параметр**, внешний вид которой показан на рис. 8-30.
3. Введите имя тега, локальной переменной, псевдоним или выражение в поле **Источник данных** или воспользуйтесь кнопками **Меню тегов** или **Выражение** для указания имени источника данных.

4. Если требуется использовать объект в качестве поля ввода значений для передачи связанному источнику данных, установите флажок **Ввод данных** в группе **Тип параметра**.
5. Если необходимо только вводить значения, одновременно не запрашивая их у связанного источника данных, снимите отметку флажка **Значение** и установите флажок **Ввод данных**.
6. Если каждый ввод значения должен сопровождаться выводом сообщения о необходимости подтвердить намерение ввода, отметьте флажок **Подтвержд**.
7. Если требуется вводить значения, а рабочая станция, на которой предполагается использовать GraphWorX32, не будет иметь клавиатуры для ввода значений, но будет оснащена указательным устройством типа "мышь" (например, сенсорным экраном), отметьте флажок **Клавиатура**. При этом в режиме *Исполнение* ввод данных может осуществляться при помощи указательного устройства с использованием управляющего элемента *Клавишная панель*. Размеры клавишной панели определяются на странице **Исполнение** в диалоговой панели **Параметры приложения** GraphWorX32.
8. В выпадающем списке **Тип данных** выберите тип отображаемых или вводимых значений. Описание возможных типов приведено в табл. 8-10.
9. В поле **Формат** выберите формат представления значений или введите его вручную.
10. Если объект используется в качестве поля ввода данных, имеется возможность установки начального значения и диапазона изменения в полях **Начальное значение** (после установки соответствующего флажка) и группе параметров **Диапазон** соответственно.
11. Дополнительные визуальные свойства динамического объекта могут быть установлены на странице **Текст** в **Инспекторе свойств**.
12. По завершении настройки параметров объекта щелкните левой клавишей мыши на кнопке **OK**.

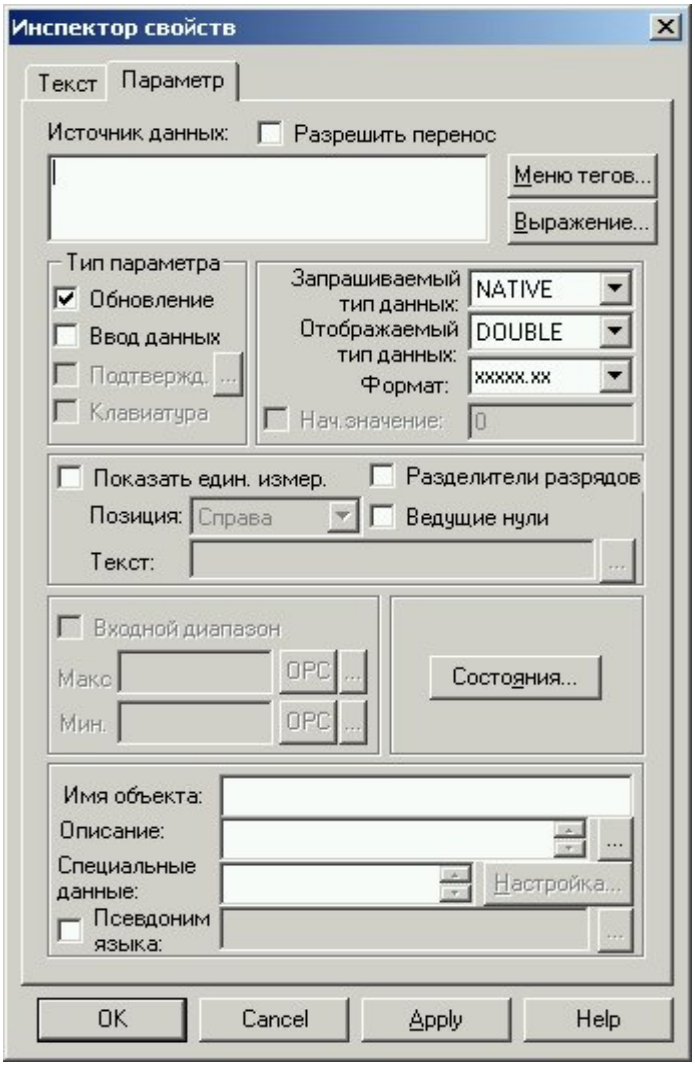


Рис. 8-30. ***Инспектор свойств*** *– страница* ***Значение параметра****.*

Таблица 8-9

Поле	Описание
Источник данных	Данное поле предназначено для ввода имени тега (тега OPC, локальной переменной, псевдонима) или выражения, значение которого будет отображаться на экране.
Меню тегов	Данная кнопка открывает Универсальный навигатор данных, позволяющий выбрать соединение с тегами OPC, переменными и псевдонимами. Для более подробной информации, обратитесь к Справочной документации Unified Data Browser.
Выражение	Данная кнопка открывает Редактор выражений. Для более подробной информации, обратитесь к Справочной документации Expression_Editor.
Тип параметра	Флажки данной группы предназначены для установки режима работы динамического объекта *Параметр*.
Обновление	При установке данного флажка в режиме *Исполнение* происходит обновление отображаемых значений. Обновление происходит асинхронно (путем получения извещений от соответствующего сервера OPC об изменении значения связанного элемента данных.)
Ввод данных	При установке данного флажка появляется возможность ввода оператором значений для передачи связанному источнику данных.
Подтвержд.	Определяет подтверждающее сообщение, которое будет вызываться GraphWorX32 при вводе пользователем значений. После установки флажка, можно воспользоваться Конфигуратором подтверждающих сообщений. Для этого нужно нажать на кнопку ... Для более подробной информации, обратитесь к пункту Конфигуратор подтверждающих сообщений.

Поле	Описание
Клавиатура	Если требуется вводить значения, а рабочая станция, на которой предполагается использовать GraphWorX32, не будет иметь клавиатуры для ввода значений, но будет оснащена указательным устройством типа "мышь" (например, сенсорным экраном), отметьте флажок **Клавиатура**. При этом в режиме *Исполнение* ввод данных может осуществляться при помощи указательного устройства с использованием управляющего элемента *Клавишная панель*. Размеры клавишной панели определяются на странице **Исполнение** в диалоговой панели **Параметры приложения** GraphWorX32
Тип данных	Данный выпадающий список предназначен для выбора типа отображаемых значений из следующего перечня: *Float* (действительное), *Double* (действительное двойной точности), *Bool* (булево), *Byte* (байт), Word (16-разрядное целое без знака), Dword (32-разрядное длинное целое), Char (символ), *Short* (16-разрядное целое со знаком), *Long* (32-разрядное целое со знаком), *String* (строка), Native
Формат	Данный выпадающий список предназначен для выбора или ввода представления отображаемых значений.
Float	По умолчанию содержит 10 форматов представления в виде xx.xxx, где символы ‘xxx’, расположенные справа от позиции плавающей запятой, представляют количество отображаемых знаков после запятой.
Double	Аналогично *Float*.
Short	По умолчанию содержит 6 форматов представления.
Long	Аналогично *Short*.
Bool	Допускает ввод пользовательских названий логических состояний. Например, *Норма/Не норма*. Для отображения введенных названий состояний в режиме *Исполнение* символами *Кириллицы*, следует в режиме *Разработка* выбрать соответствующий набор символов.
String	Перед размещением объекта, с помощью которого предполагается отображать строковые сообщения, следует ввести максимальную длину строки.
Byte	Аналогично *Short*.
Word	Аналогично *Short*.
DWord	Аналогично *Short*.
Char	Аналогично *Short*.
Hexadecimal	Шестнадцатеричная константа , т.е. 0x20A = 2 * (16^2) + 0 *(16^1) + 10 * (16^0) = 2*256 + 0*16 + 10 * 1 = 512 + 0 +10=522
Octal	Восьмеричная константа, т.е. 0t36= 3 * (7^1) + 6 *(7^0) = 3* 7 + 6* 1= 21 + 6 = 27
Binary	Двоичная константа, т.е. 0b110 = 1 * (2^2) + 1 * (2^1) + 0 * (2^ 0) = 1 * 4 + 1 * 2 + 0 * 1 = 4+2+0 = 6
Native	Пользовательский тип данных
Начальное значение	Данное поле позволяет ввести значение, которое будет передано связанному источнику данных в момент перевода экранной формы в режим *Исполнение*.
Показать един.измер.	При установке этого флажка в режиме исполнения рядом (**справа** или **слева**) с величиной будет выводиться единица её измерения. **Текс**т единицы измерения может быть введен вручную, а также с помощью меню **Глобальных** и **Языковых** псевдонимов (вызываемых по кнопке ...).
Разделители разрядов	Установите флажок, чтобы разделять тысячи. Символ разделителя разрядов зависит от региональных установок, например: в режиме исполнения число «13200» будет показано в виде «13,200» в США и «13.200» в Европе.
Ведущие нули	Установите флажок, чтобы показать ведущие нули слева от величины в режиме исполнения. Возможно использование столько ведущих нулей, сколько необходимо для достижения нужного формата. Например, если формат «xxxx», значение «23», то параметр будет выводиться в формате «0023»

Поле	Описание
Входной диапазон	Поля данной группы предназначены для задания диапазона вводимых значений.
Макс.	Верхняя граница диапазона вводимых значений.
Мин.	Нижняя граница диапазона вводимых значений.
Состояния	Данная кнопка позволяет определить набор состояний, которые будут передаваться связанному источнику данных при выборе оператором соответствующих строк в выпадающем списке, появляющимся под объектом *Параметр* в момент ввода данных, как показано на рис. 8-31.

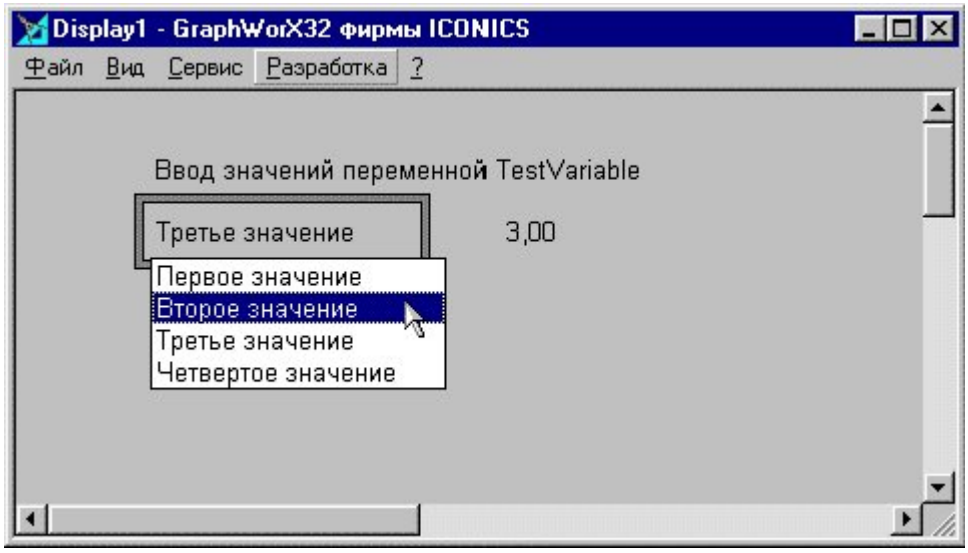


Рис. 8-31. Использование строк состояния для ввода данных

Для определения строк состояний:

1. На странице свойств **Параметр** в **Инспекторе свойств** нажмите кнопку **Состояния**. На экран монитора будет выведена диалоговая панель **Конфигурация поля состояния**, показанная на рис. 8-32.
2. В поле **Строка для данного состояния** введите текстовую строку, которая в режиме *Исполнение* будет появляться в выпадающем списке.
3. В поле **Значение состояния** введите значение, которое должно передаваться в связанный источник данных при выборе строки, соответствующей данному состоянию.
4. Нажмите кнопку **Добавить** после ввода значения и строки настраиваемого состояния.
5. Выполните настройку остальных состояний аналогичным образом.
6. Для начального состояния, определяемого в поле **Начальное значение** в **Инспекторе свойств**, определите строку в поле **Исходная строка** в диалоговой панели **Конфигурация поля состояния**.
7. Сконфигурированный набор состояний может быть сохранен в текстовом файле нажатием кнопки **Экспорт**. Загрузка набора состояний из файла выполняется нажатием кнопки **Импорт**.
8. Для изменения значения или строки для какого-либо состояния, следует выбрать соответствующую строку в таблице сконфигурированных состояний диалоговой панели **Конфигурация поля состояния**, изменить строку или значение в поле **Значение состояния** или **Строка для данного состояния** и нажать кнопку **Изменить**.
9. Для удаления всех сконфигурированных состояний нажмите кнопку **Сбросить все**.

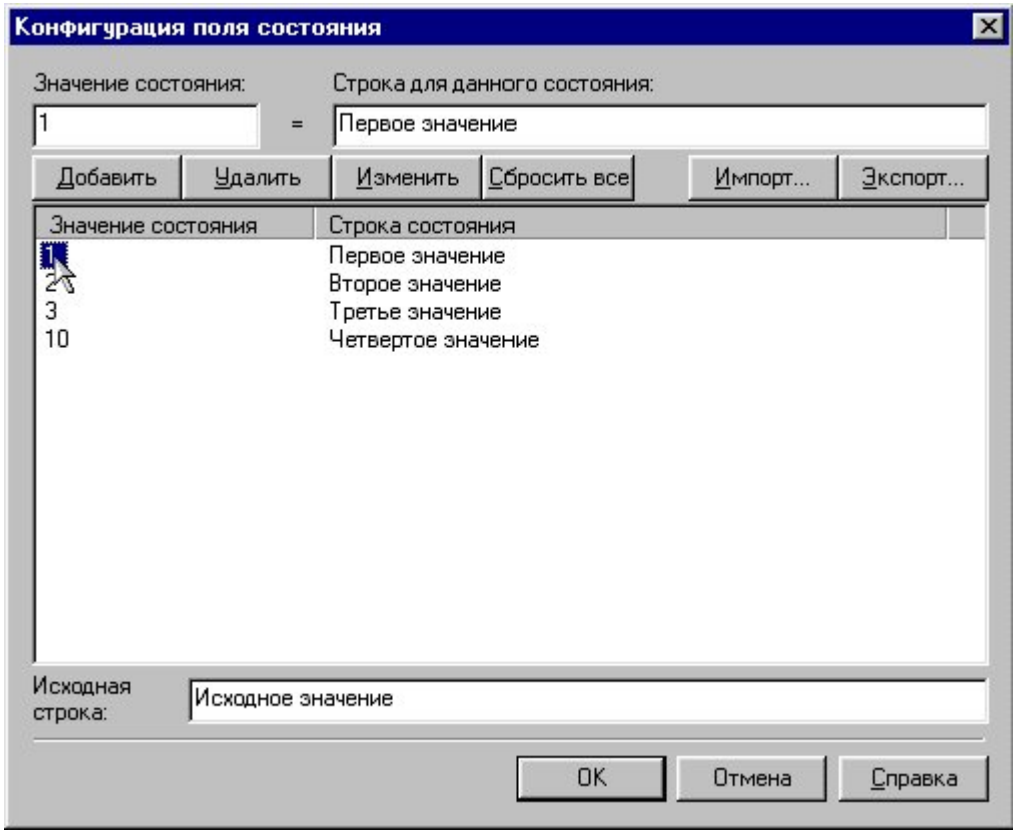


*Рис. 8-32. Диалоговая панель **Конфигурация поля состояния***

Время/Дата



Данный динамический объект предназначен для отображения текущих времени и даты в режиме *Исполнение*. Возможны различные варианты представления времени и даты.
Для создания элемента отображения времени и даты:

1. Выберите строку **Время/Дата** подменю **Встроенные объекты** меню **Динамика** или щелкните левой клавишей мыши на кнопке *Время/Дата* панели инструментов *Динамика*. Данный динамический объект предназначен для отображения текущих времени и даты в режиме *Исполнение*. Возможны различные варианты представления времени и даты
2. Щелкните левой клавишей мыши в точке экранной формы, в которой требуется создать динамический объект *Время/Дата*. На экран монитора будет выведена диалоговая панель **Инспектора свойств** с выбранной закладкой **Время/Дата**, внешний вид которой показан на рис. 8-33.
3. В выпадающем списке **Отобразить** выберите содержание элемента отображения *Время/Дата*.
4. В списках **Формат времени** и **Формат даты** выберите формат представления времени и даты.
5. Дополнительные визуальные свойства динамического объекта могут быть установлены на странице **Текст** в **Инспекторе свойств**. Для отображения символьных представлений в режиме *Исполнение* символами *Кириллицы*, следует в режиме *Разработка* выбрать соответствующий набор символов.
6. По завершении настройки параметров объекта щелкните левой клавишей мыши на кнопке **OK**.

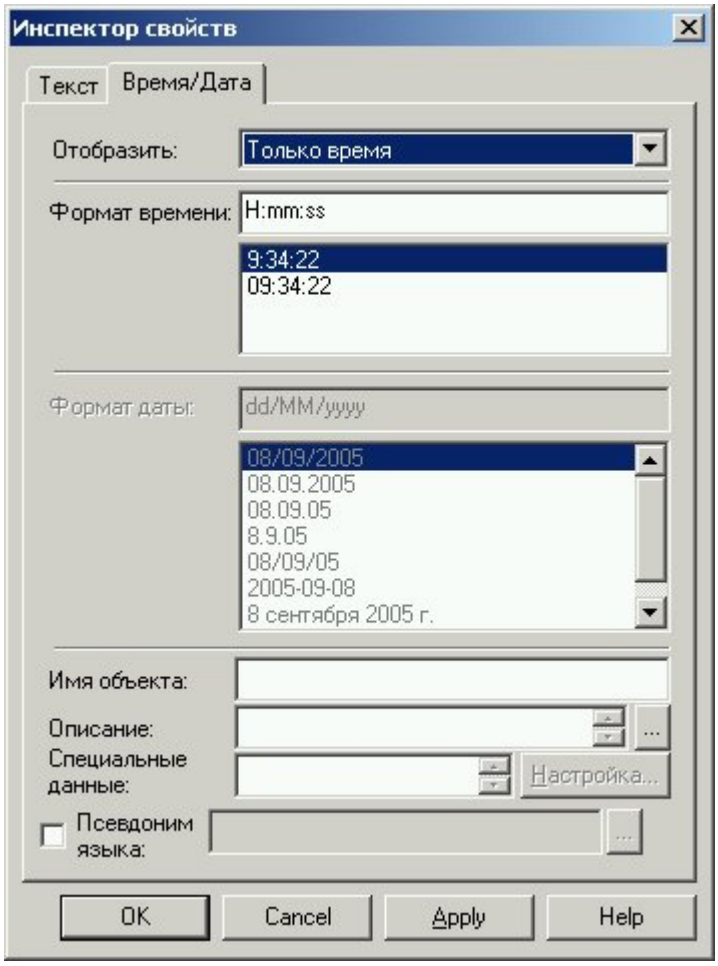


*Рис. 8-33. **Инспектор свойств** -- страница **Время/Дата***

Встроенный объект *Кнопка*



Данный динамический объект предназначен для создания изображений кнопок в экранных формах GraphWorX32. Функционирование кнопки абсолютно идентично описанному выше динамическому соединению *Указание и щелчок* за исключением того, что состояние кнопки (*Нажато/Отпущено*) соответствующим образом отображается в режиме *Исполнение*.

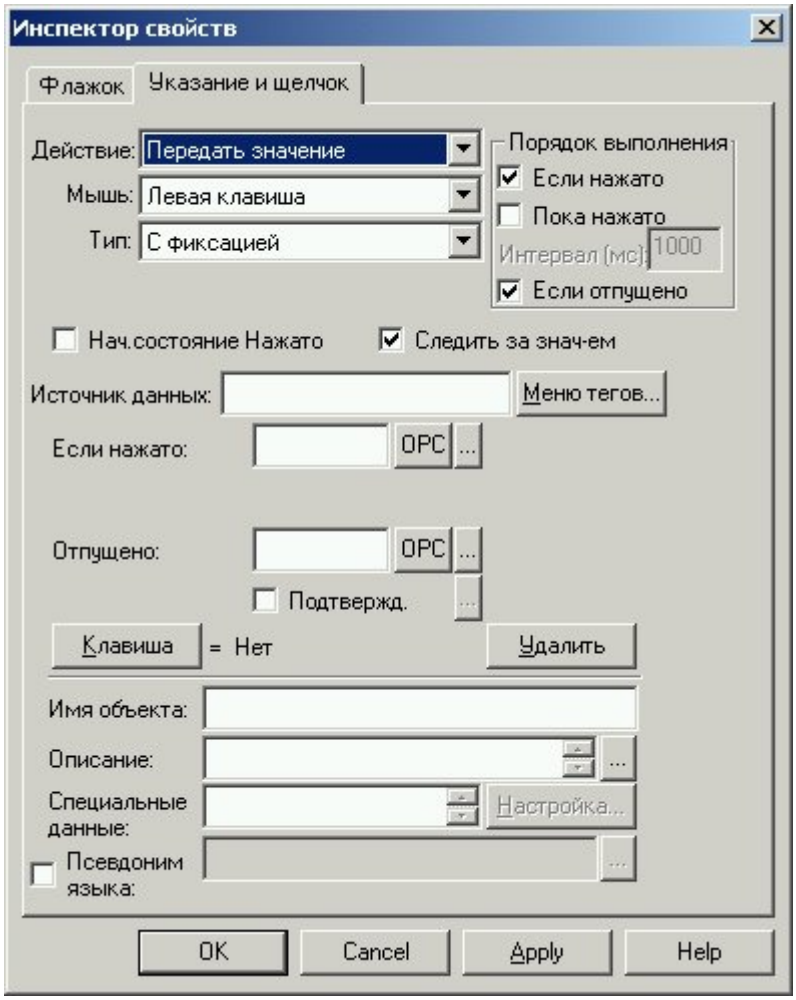
Для создания кнопки:

1. Выберите строку **Кнопка** подменю **Встроенные объекты** меню **Динамика** или выберите соответствующий инструмент в панели инструментов *Динамика*.
2. Щелкните левой клавишей мыши в точке экранной формы, в которой требуется создать динамический объект *Кнопка*. На экран монитора будет выведена диалоговая панель **Инспектора свойств** с выбранной закладкой.
3. Настройте параметры операции, выполняемой при нажатии и отпускании кнопки.
4. По завершении настройки параметров объекта нажмите кнопку **OK**.

Встроенный объект *Флажок*



Данный встроенный объект позволяет создавать элемент управления *Флажок* таким же образом, каким создается объект *Кнопка*. В режиме *Исполнение* щелчок мышью на объекте *Флажок* приводит к его установке(отметка флажка) или снятию.
Закладка **Указание и щелчок** для объекта **Флажок** приведена на рисунке:



Если установлен флажок «Следить за значением», в режиме исполнения флажок будет устанавливаться /сниматься в зависимости от изменения источника данных.

Встроенный объект *Переключатель/Радиокнопка*



Данный встроенный объект позволяет создавать элемент управления *Переключатель* таким же образом, каким создается объект *Кнопка*. Переключатель в режиме *Исполнение* функционально отличается от обычной кнопки как внешним видом, так и логикой работы. Щелчок мышью на переключателе приводит к его установке с одновременным сбросом всех связанных с ним переключателей и радиокнопок, которые объединяются в группу с одним именем. Переключатель остается в установленном состоянии до тех пор, пока не произведен щелчок на любом другом переключателе или радиокнопке, входящем в одноименную группу.

Кнопка с экраном



Выразительные средства GraphWorX32 имеют в своем составе объект, называемый *Кнопка с экраном*. *Кнопка с экраном* после настройки, выполняемой при помощи **Мастера кнопки с экраном**, может выполнять одну из четырех функций:

- Загрузка новой экранной формы
- Открытие экранной формы во всплывающем окне
- Открытие экранной формы во встроенном окне
- Работа в режиме объекта, ссылающегося на экранную форму, которая может быть вставлена в контейнер GraphWorX32 ActiveX в стиле "Перенести-и-Оставить" (Drag-and-Drop)

Кнопки с экраном позволяют улучшить доступность пользовательского интерфейса и устранить излишнюю перегруженность экранных форм при создании больших проектов, поскольку различные элементы графических мнемосхем могут отображаться на экране только по мере надобности путем щелчка мышью на кнопках с экраном.
Вызов **Мастера кнопки с экраном** осуществляется выбором строки **Мастер для кнопки с экраном** подменю **Встроенные объекты** меню **Динамика** или щелчком мышью на кнопке  панели инструментов *Динамика* GraphWorX32. На экран монитора выводится диалоговая панель, внешний вид которой показан на рис. 8-34.

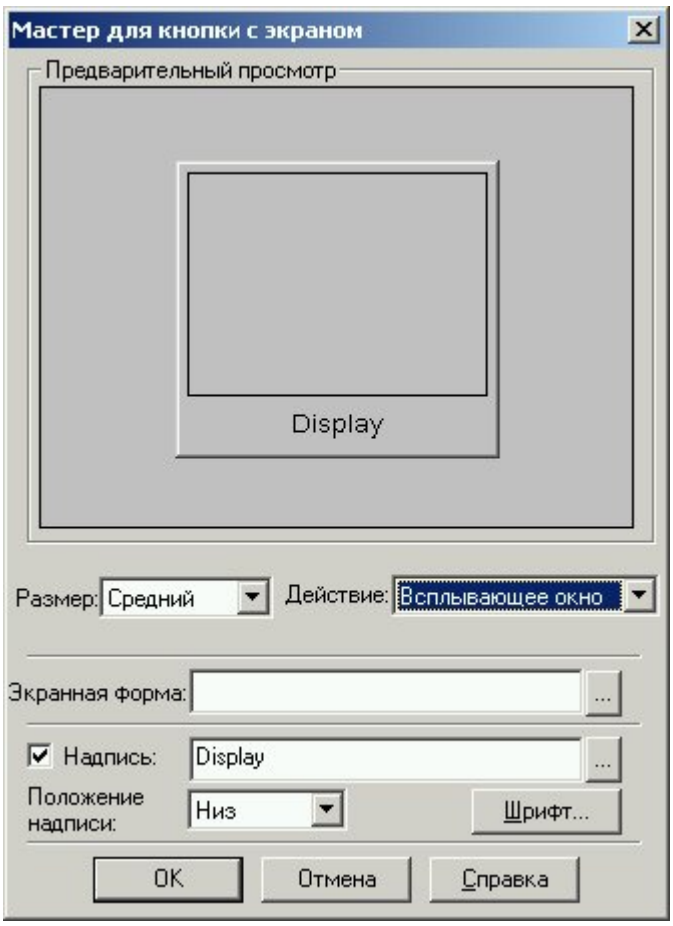
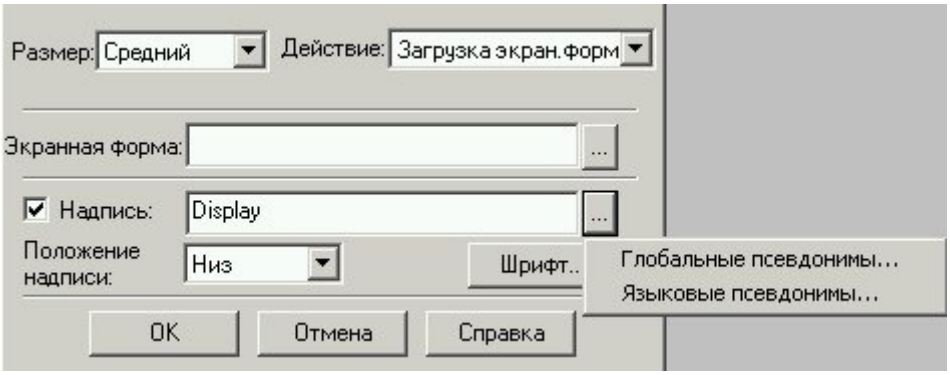


*Рис. 8-34. Диалоговая панель **Мастер для кнопки с экраном***

Область **Предварительный просмотр** показывает, каким образом будет выглядеть поверхность кнопки после настройки ее параметров.
Выпадающий список **Размер** позволяет выбрать один из пяти размеров кнопки.
Поле **Надпись** задает наличие и содержание надписи на поверхности кнопки.
Местоположение надписи определяется выпадающим списком **Положение надписи**.
Примечание 1. Перед вводом надписи символами кириллицы следует предварительно установить соответствующий набор символов для шрифта путем нажатия кнопки **Шрифт**.
Примечание 2. После настройки параметров создаваемой кнопки с экраном повторный вызов *Мастера* для данной кнопки невозможен. Для изменения функциональности кнопки с экраном следует щелкнуть на ней правой клавишей мыши, удерживая клавишу SHIFT в нажатом состоянии, после чего дважды щелкнуть на надписи кнопки. Это приведет к вызову **Инспектора свойств** объекта с возможностью настройки динамического действия *Указание и щелчок*.

Поле **Экранная форма** и пункт **Обзор файлов** в выпадающем меню кнопки ..., предназначено для ввода имени файла экранной формы, вызов которой будет происходить при выполнении действия, установленного в выпадающем списке **Действие**.

Кнопки ... справа от поля **Экранная форма** и **Надпись** на кнопке позволяют выбрать также **Глобальные** и **Языковые псевдонимы как показано на рисунке**.



В выпадающем списке **Действие** может быть выбрано одно из следующих действий:
Загрузка экранной формы
Закрывает текущую и загружает новую экранную форму в текущее окно GraphWorX32. Имя экранной формы указывается в поле **Экранная форма**.
Встроенное окно
Загружает экранную форму во встроенное окно в пределах текущего окна GraphWorX32. Имя экранной формы указывается в поле **Экранная форма**
Всплывающее окно
Загружает экранную форму во всплывающее окно, которое будет дочерним для текущего окна GraphWorX32. Имя экранной формы указывается в поле **Экранная форма**.
Загрузка в стиле Перенести/Оставить
Кнопка с экраном выполняет функцию объекта, который может быть "перенесен и отпущен" внутрь контейнера GraphWorX32 ActiveX. Роль объекта играет экранная форма, указанная в поле **Экранная форма.**

8.6 Копирование динамических действий, примененных к графическим объектам

Помимо удаления динамических действий, примененных к графическому объекту, имеется возможность применения этого же набора действий к любому другому динамическому объекту.
Для копирования динамики, связанной с одним объектом в другой объект:

1. Выделите объект, набор примененных динамических действий которого должен быть связан с другим графическим объектом, и выберите команду **Копировать** меню **Формат**.
2. Выделите объект, с которым должен быть связан набор динамических действий, щелкните правой клавишей мыши и выберите строку **Вставить динамику** в появившемся контекстном меню.

8.7 Использование специальных данных

Настройка

Каждая закладка **Инспектора свойств** GraphWorX32 имеет поле **Специальные данные** и кнопку **Настройка**, расположенную справа от поля **Специальные данные**. Данная кнопка может быть связана с утилитой настройки специальных данных, функциональность которой определяется пользователем.
Щелчок на кнопке **Специальные данные – Настройка** приводит к созданию объекта автоматизации, имеющего определенный идентификатор программы (*ProgID*). По умолчанию *ProgID* равен *IcoCustom.Configure*. Данный идентификатор может быть переопределен путем установки альтернативного *ProgID* в следующем ключе системного реестра Windows:

HKEY_LOCAL_MACHINE\SOFTWARE\ICONICS\Gwx32\OEM Information\Custom Configure ProgID = "***ProgID_пользователя***"

При этом может потребоваться, чтобы объект автоматизации создавался из внешней библиотеки динамической компоновки (DLL).
GraphWorX32 будет затем использовать интерфейс диспетчеризации этого объекта для вызова метода:

boolean CustomConfigure(long hWndParent, IDispatch* ObjectOne, BSTR ObjectOneType, IDispatch* ObjectTwo, BSTR ObjectTwoType);

Описание.

hWndParent

Идентификатор окна (*handle*) **Инспектора свойств** GraphWorX32. Пользователю может потребоваться вызвать диалоговую панель настройки каких-либо параметров в качестве элемента пользовательского интерфейса. Данный аргумент используется в качестве идентификатора родительского окна для указанной диалоговой панели, создаваемой пользователем.

ObjectOne

Диспетчеризуемый указатель объекта GraphWorX32, связанного с текущей отображаемой страницей **Инспектора свойств** (например, если выбрана страница **Указание и щелчок** в диалоговой панели **Инспектор свойств**, **ObjectOne** будет указывать на объект класса **GwxPick** для этой страницы).

ObjectOneType

Строковое имя класса объекта **ObjectOne** (например, "*GwxPick*"). Пользователю может потребоваться выполнить проверку типа объекта с тем, чтобы быть уверенным в том, что он поддерживается в пользовательской конфигурации.

ObjectTwo

Диспетчеризуемый указатель экранной формы GraphWorX32, которой принадлежит **ObjectOne**.

ObjectTwoType

Строковое имя класса объекта **ObjectTwo** (например, " *GwxDisplay* ").
Возвращаемое значение равно TRUE, если пользователем внесены изменения в конфигурацию объектов, доступ к которым был получен с помощью данного метода. FALSE возвращается в случае, если объекты оставлены без изменений. **Не освобождайте диспетчеризуемые указатели, передаваемые этому методу, путем вызова метода Release()!!!**По завершении выполнения данного метода, GraphWorX32 освобождает указатель для объекта автоматизации "*IcoCustom.Configure*".

Ниже приведен пример исходного текста возможной реализации метода **CustomConfigure**.

```
BOOL CAutoConnect::CustomConfigure(long hWndParent, LPDISPATCH ObjectOne, LPCTSTR
ObjectOneType, LPDISPATCH ObjectTwo, LPCTSTR ObjectTwoType)
{
  CString obj1Type(ObjectOneType);
  if (obj1Type == _T("GwxPick"))
  {
    CString strCustomData;
//Вызов пользовательской диалоговой панели настройки здесь...
if (returnValOfConfigDialog)
    {
      COleDispatchDriver obj1(ObjectOne, FALSE);
      //установка пользовательских данных
      obj1.SetProperty(0x10001, VT_BSTR, (LPCTSTR)strRCustomData);
      //установка для Указание и щелчок = Специальная команда
      obj1.SetProperty(0x1, VT_I4, (long)8);
```

```
      return TRUE;
    }
  }
  else
  {
    AfxMessageBox(_T("Специальная конфигурация для объектов данного типа не
поддерживается."));
  }
  return FALSE;
}
```

Потоки данных и таймера.

Настройка потоков данных и таймера (если необходимо увеличить приоритет потока GraphWorX) производится с помощью двух параметров системного реестра:

HKEY_LOCAL_MACHINE\SOFTWARE\ICONICS\Gwx32\Runtime Settings\DataThread\Priority

HKEY_LOCAL_MACHINE\SOFTWARE\ICONICS\Gwx32\Runtime Settings\TimerThread\Priority

Значения параметров являются строковыми переменными. Возможны следующие значения строк:

-15	THREAD_PRIORITY_IDLE
-2	THREAD_PRIORITY_LOWEST
-1	THREAD_PRIORITY_BELOW_NORMAL
0	THREAD_PRIORITY_NORMAL [default]
1	THREAD_PRIORITY_ABOVE_NORMAL
2	THREAD_PRIORITY_HIGHEST
15	THREAD_PRIORITY_TIME_CRITICAL

Выполнение специальной команды

В предыдущем подразделе приведен один из возможных примеров использования динамического действия *Указание и щелчок* для настройки определяемых пользователем параметров системы в GraphWorX32. Кроме того, имеется возможность выполнения специальных функций и запуска приложений. Пользовательское приложение может быть исполняемым модулем (EXE) или библиотекой динамической компоновки (DLL).
При первом щелчке на объекте, с которым связано динамическое действие *Указание и щелчок* с операцией *Специальная команда*, GraphWorX32 создает объект автоматизации с определенным идентификатором программы (*ProgID*).
По умолчанию *ProgID* равен *IcoCustom.Configure*. Данный идентификатор может быть переопределен путем установки альтернативного *ProgID* в следующем ключе системного реестра Windows:

HKEY_LOCAL_MACHINE\SOFTWARE\ICONICS\Gwx32\OEM Information\Custom Configure ProgID = "***ProgID_пользователя***"

При этом может потребоваться, чтобы объект автоматизации создавался из внешнего исполняемого модуля (EXE). Исполняемый модуль должен позволять создавать несколько экземпляров указанного объекта автоматизации без запуска нескольких копий одного и того же приложения.GraphWorX32 затем использует интерфейс диспетчеризации данного объекта для вызова метода:

```
boolean CustomCommand(long hWndParent, IDispatch* ObjectOne,  BSTR
ObjectOneType, IDispatch* ObjectTwo, BSTR ObjectTwoType);
```

Описание аргументов аналогично приведенному в п. 17.1.**Не освобождайте диспетчеризуемые указатели, передаваемые этому методу, путем вызова Release()!!!** По завершении исполнения данного метода, GraphWorX32 **НЕ ОСВОБОЖДАЕТ** диспетчеризуемый указатель объекта автоматизации "*IcoCustom.Command*".

Данный указатель используется GraphWorX32 при последующих вызовах *Специальной команды* (с тем, чтобы не потребовалось многократно создавать один и тот же объект). Указатель не освобождается до тех пор, пока не будет завершен владеющий им экземпляр GraphWorX32.

8.8 Конфигуратор подтверждающих сообщений

Конфигуратор подтверждающих сообщений позволяет создавать сообщения, которые выводятся при каждом вводе пользователем значения на экран GraphWorX32. Использовать Конфигуратор можно в одной из следующих динамических связей:

- Параметр
- Действие Передать значение по указанию и щелчку над объектом
- Действие Передать значение по указанию и щелчку
- Динамическое изменение положения (Положение)
- Динамическое вращение (Вращение)

Например, на странице **Параметр** диалоговой панели **Инспектор свойств**, показанной на рис.8-35, установка флажка **Подтвержд**. (Подтверждать) позволяет определить подтверждающее сообщение, которое будет выводиться каждый раз при вводе пользователем значения.

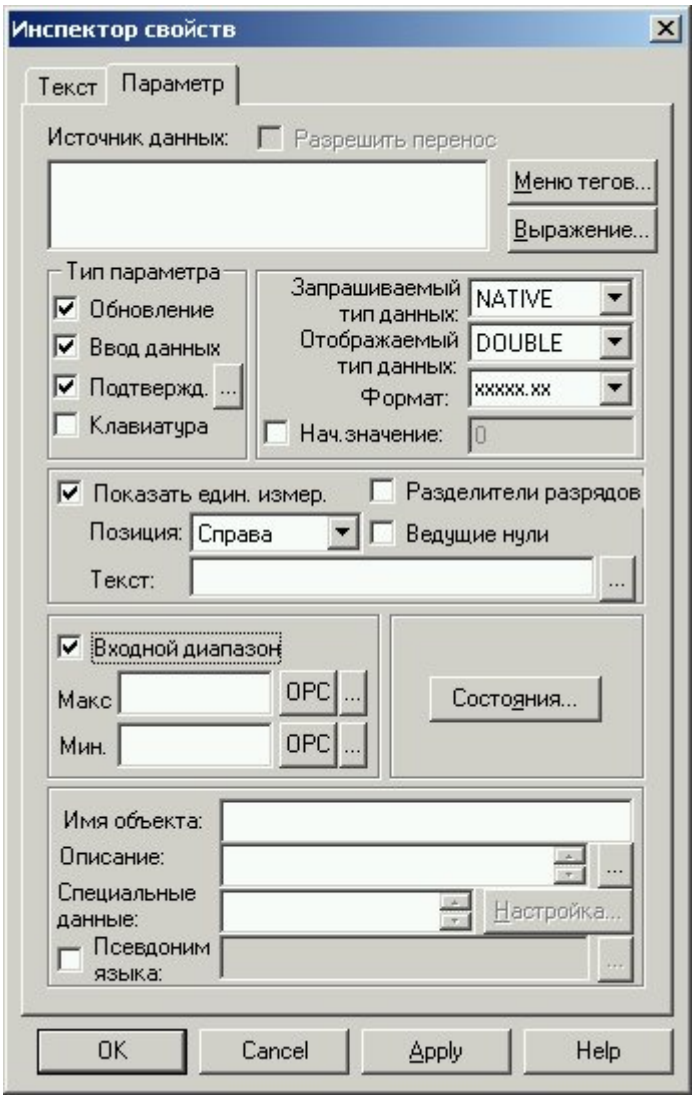


Рис. 8-35. Подтверждение вводимого Параметра

Нажатие кнопки ··· справа от поля **Подтвержд.** открывает Конфигуратор подтверждающих сообщений, показанный на рис.8-36. Верхний левый угол диалогового окна Конфигуратора позволяет ввести и отредактировать сообщение. Подтверждающее сообщение содержит текст, языковые псевдонимы, глобальные псевдонимы и заменяемые параметры, которые представляют собой динамическое соединение или данные (напр. Величина, Тег, Имя и т.д.).

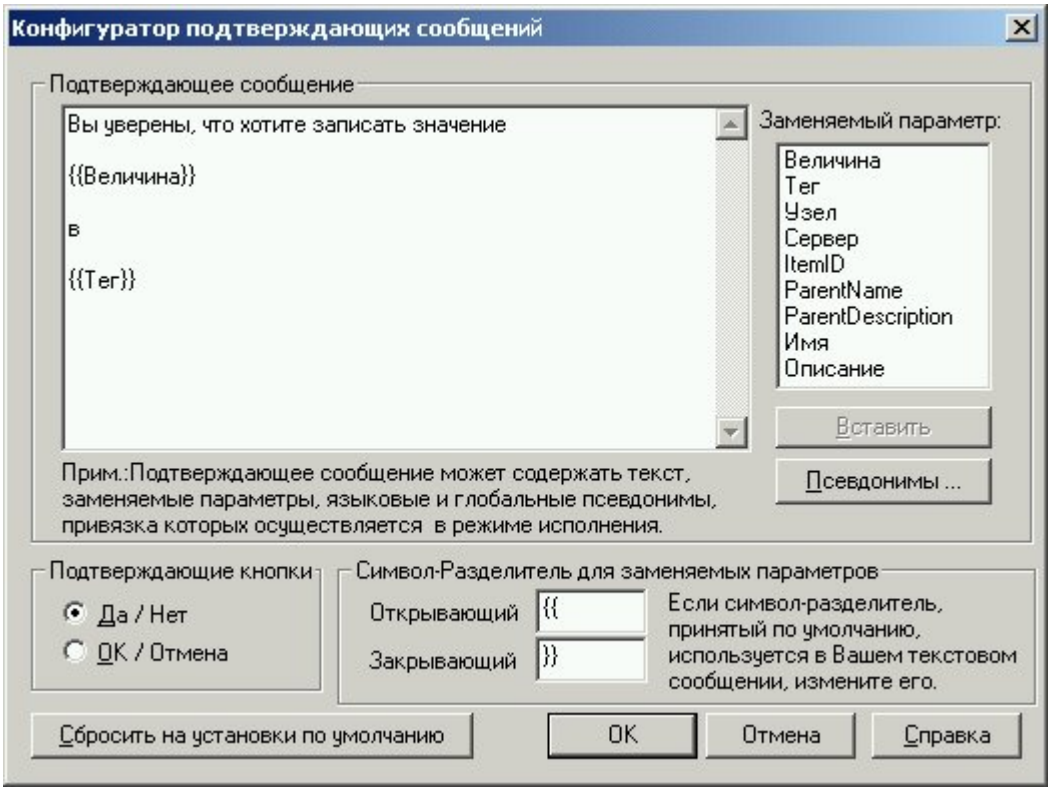


Рис. 8-36. Конфигуратор подтверждающих сообщений

Чтобы выбрать глобальный или языковой псевдоним, нажмите кнопку **Псевдонимы** и выберите из выпадающего меню нужный навигатор.
Чтобы выбрать заменяемый параметр в подтверждающее сообщение, нужно выбрать в окне **Заменяемый параметр** (верхняя правая часть диалогового окна Конфигуратора подтверждающих сообщений) и нажать кнопку **Вставить** (или двойной щелчок мыши на выбранном параметре).
В примере на рис.8-36 используется следующий синтаксис:
«Вы уверены, что хотите записать значение {{Величина}} в {{Тег}}»
Заметьте, что {{Величина}} и {{Тег}}являются заменяемыми параметрами текущей величины и имя тега соответственно. По умолчанию используются символы-разделители {{и}}, но вы можете изменить их, если эти символы уже используются в вашем текстовом сообщении.
После того, как вы определите текстовое сообщение, выберите Подтверждающие кнопки, которые будут выводиться пользователю: Да / Нет и Ок /Отмена.

8.9 Доступ к данным и их визуализация

GraphWorX32 поддерживает с открытыми базами данных соединения, описанные в Конфигураторе Сервера обмена данными GENESIS32 (Data Mining Configurator). Следующим шагом после конфигурации связей с базами данных, источниками и элементами данных, является визуализация данных. Демонстрационные примеры доступа к данным GraphWorX32 (Data Mining Demo) показывают примеры настройки соединений с БД и экранов с визуализацией данных из базы данных при той или иной конфигурации связей. Примечание: Для более подробной информации, см. Справочную документацию Data Mining Configurator.

Один из демо-примеров расположен в директории: **C:\Program Files\ICONICS\GENESIS-32\Examples\Database Mining Examples ,** файл DBOPC.gdf.
В этом примере экранная форма открывается в режиме разработки, как показано на рисунке 8-37. Экран содержит два различных примера визуализации доступа к данным: статический и динамический. Статический экран просто отображает значения данных из связанных с ними баз. Динамический экран позволяет управлять доступом к данным из связанных с ними баз.
Таблица статического сектора содержит несколько столбцов с именами из таблицы базы данных Employees (Сотрудники):

- Name: Имя сотрудника
- Surname: Фамилия сотрудника
- Salary: Зарплата сотрудника
- PID: Идентификационный номер сотрудника

Каждый столбец содержит параметры (например, Name_1, Name_2, Name_3, PID_1, PID_2 и т.д.), связанные с данными из демонстрационной базы Employees_By_PID . Каждый параметр должен быть настроен на элемент данных. Для примера рассмотрим значение Name_2.
1. В колонке Name экрана GraphWorX выполните двойной щелчок мышью во второй строке, как показано на *Рис. 8-37*.

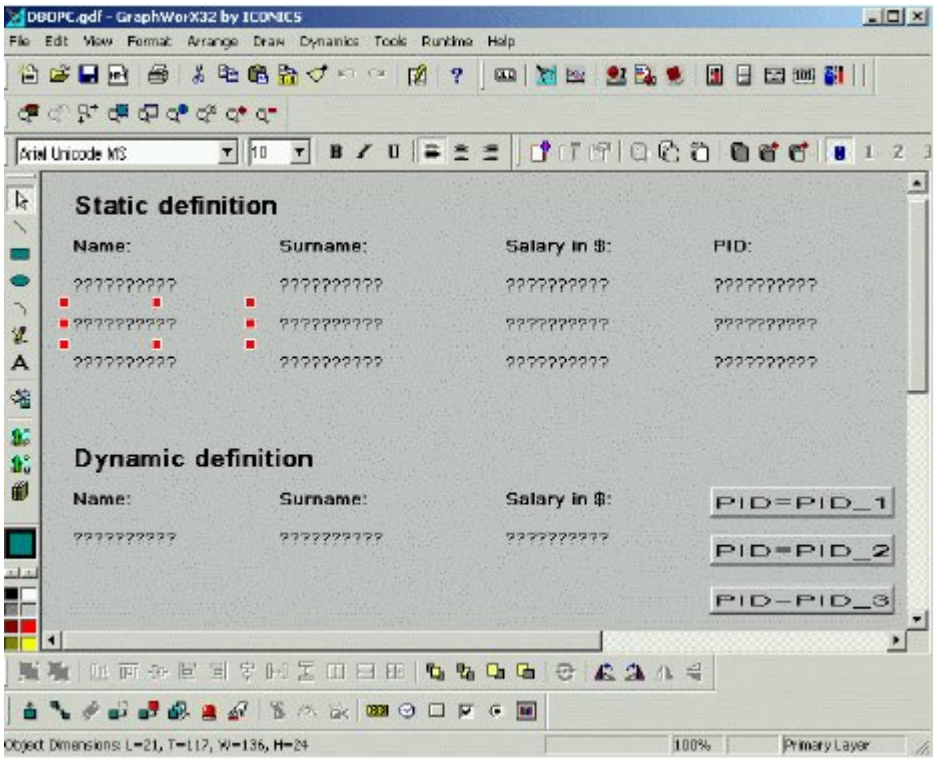


Рис. 8-37 Демонстрационный пример Data Mining в GraphWorX32: Режим разработки

2. Откроется Инспектор свойств.
3. Выберите закладку **PPT/DE** (Параметр) как показано на Рис. 8-38 и убедитесь, что источником является AdministrationDB.Employees_By_PID.Name_2.

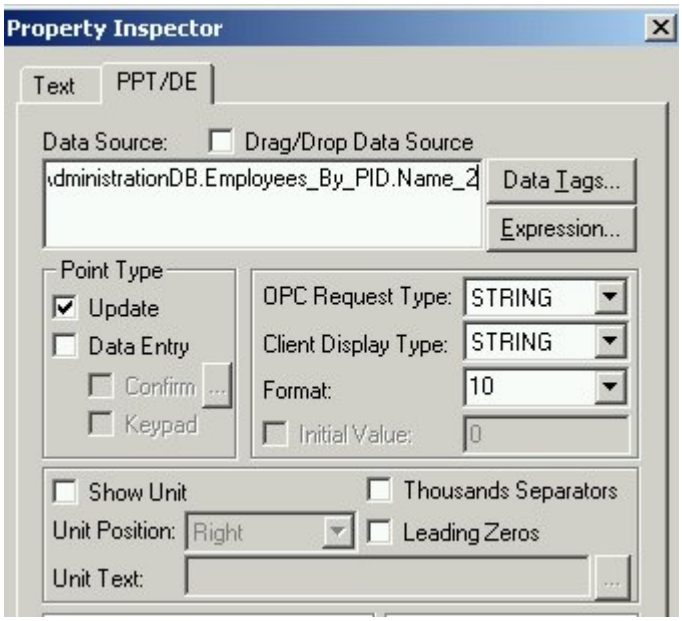


Рис. 8-38 Инспектор свойств для Name_2

4. Нажмите кнопку **Data Tags** (Меню тегов) для открытия Универсального навигатора данных. Выбрать закладку Databass Access как показано на рисунке 8-39. Вы можете увидеть, что структура дерева аналогична структуре Data Mining Configurator. Вы можете выбрать из базы Employees_By_PID любой источник данных. Выбираем Name_2.
Путь к каждому выбранному источнику (например ICONICS.DatabaseOPCServer.3\AdministrationDB.Employees_By_PID.Name_2) содержит следующую информацию:

- Имя сервера (ICONICS.DatabaseOPCServer.3\)
- Имя базы данных (AdministrationDB)
- Имя таблицы базы данных (Employees_By_PID)
- Имя поля из базы данных (Name_2)

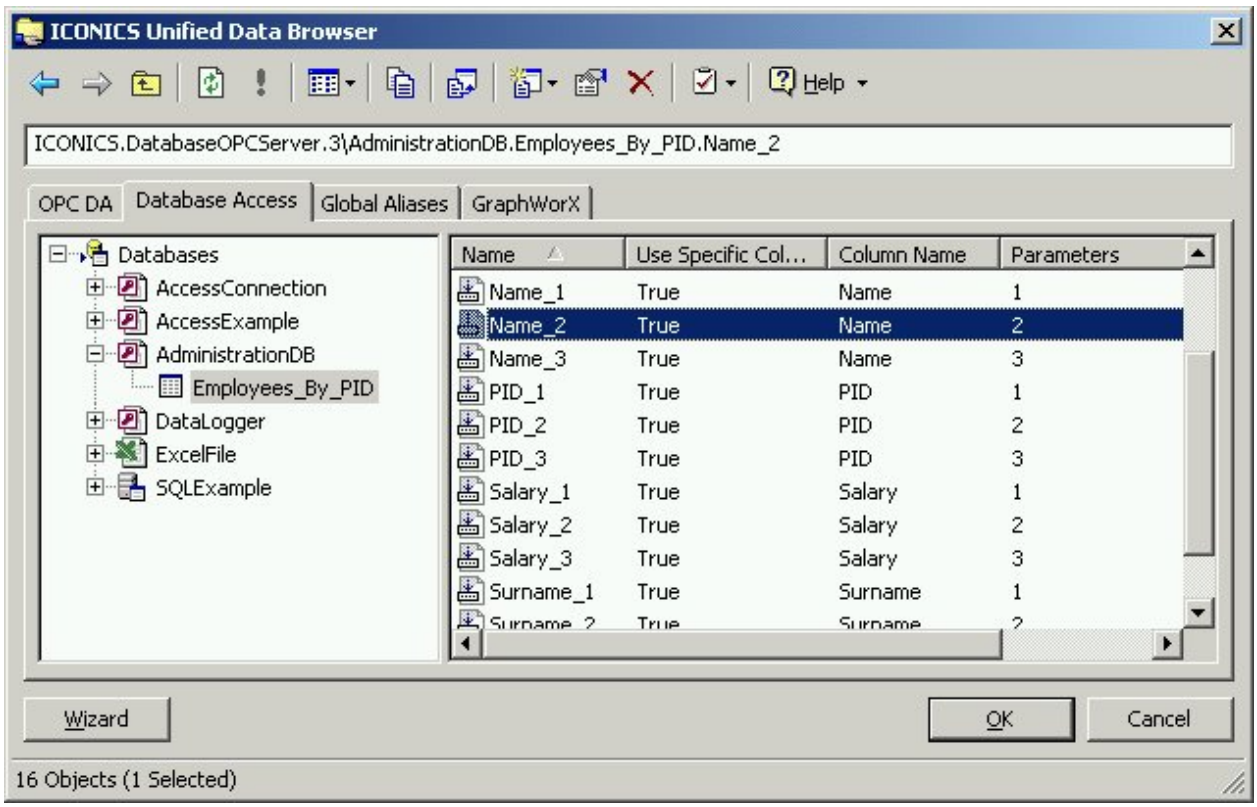


Рис. 8-39 Универсальный навигатор данных

5. Закройте Универсальный навигатор данных и диалоговое окно Инспектора свойств.
6. Переведите экран GraphWorX32 в режим Исполнения. Вы увидите данные из таблицы Employee как показано на рис. 8-40. Имя рассматриваемого нами значения Name_2 будет “Russel”.

Примечение: Статический сектор примера просто отображает значения из базы данных Employee. Эти значения не могут быть изменены. Динамический сектор примера также отображает значения из таблицы Employee, но показывает только одну строку с выбранным идентификационным номером (в примере выбран PID_1) . Вы можете увидеть информацию по другому сотруднику – для этого нажмите кнопку с интересующим Вас PID. Динамическая конфигурация использует локальные и глобальные псевдонимы, которые позволяют переключаться между строками в режиме исполнения.



Рис. 8-40 Демонстрационный пример Data Mining в GraphWorX32: Режим исполнения

7. Данные, отображаемые в GraphWorX, соответствуют информации в таблице Employees базы данных Administration database. Например, AdministrationDB.Employees_By_PID.Name_2 имеет значение Russel, т.к. Russel- это второе имя по порядку строк таблицы базы данных.

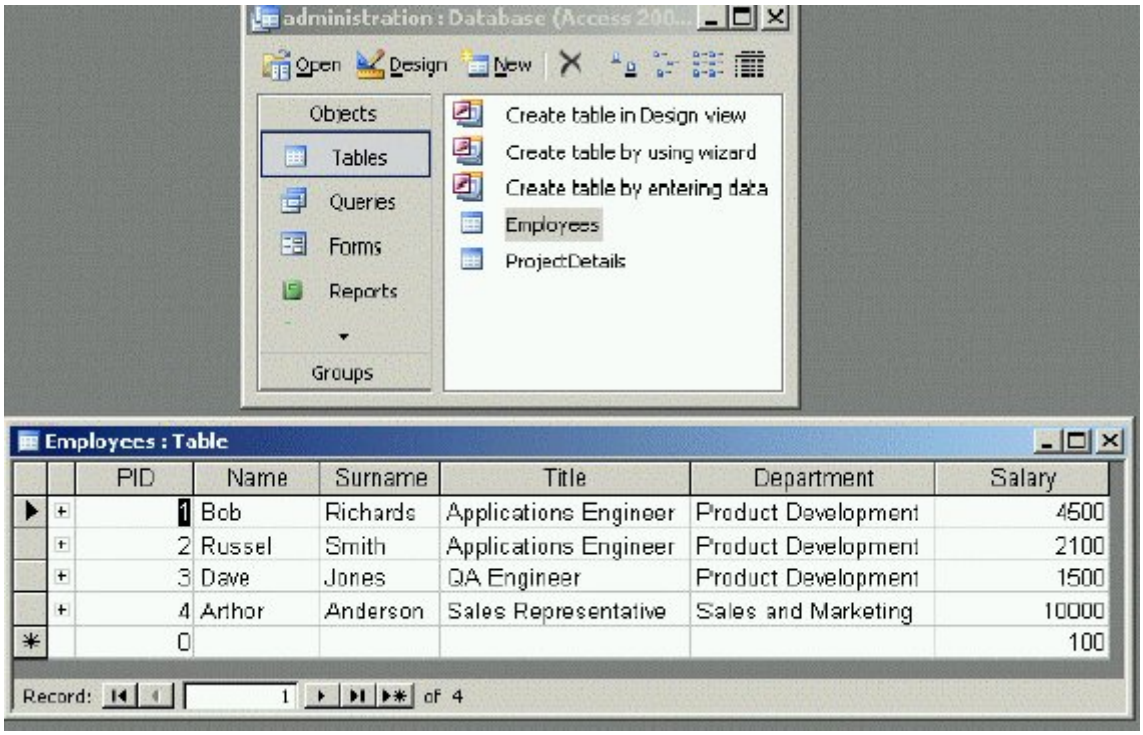


Рис. 8-41 Таблица Employees в базе данных AdministrationDB

Глава 9. Сервис.

9.1 Введение

9.2 Функциональные клавиши

9.3 Установка рабочего каталога

9.4 Настройка прав доступа.

9.5 Редактор псевдонимов

9.6 Экспорт в HTML

9.7 Поддержка Windows CE

9.1 Введение

Меню **Сервис** содержит различные инструменты для работы с данными GraphWorX32. Ниже приведен список опций:
Макрос (создание, редактирование, отладка, запуск, удаление неиспользуемых модулей VBA и редактор Visual Basic). Дополнительная информация о редакторе Visual Basic содержится в документации на Visual Basic for Applications.
Функциональные клавиши
Установка рабочего каталога
Настройка прав доступа (доп. информация в документации на Security Configurator)
Редактор файла псевдонимов
Конфигуратор глобальных псевдонимов (доп. информация в документации на Global Aliasing)
Конфигуратор псевдонимов языков (доп. информация в документации на Language Aliasing)
Преобразование экранной формы GFW 3.xx (доп. информация в документации на GraphWorX32 Translation Utility)
Экспорт в HTML
Windows CE (разработка для Windows CE и настройки Windows CE).

9.2 Функциональные клавиши

Опция **Функциональные клавиши** из меню **Сервис** предоставляет возможность настройки функциональных клавиш и иерархической структуры. При выборе пункта **Функциональные клавиши** из меню **Сервис** будет открыто диалоговое окно **Редактора сценариев функциональных клавиш**, показанное на рисунке 9-1, где можно добавлять, редактировать и удалять события. Событие представляет собой связь комбинации клавиш со сценарием VBA, который будет запущен на исполнение по нажатию этой комбинации клавиш.

Нажатие на кнопках **Добавить** или **Изменить** приводит к открытию диалогового окна **Свойства функциональной клавиши**, показанного на рисунке 9-2.
Необходимо ввести в соответствующем поле имя сценария. При щелчке на кнопке **Редактор VBA** сценарий открывается для редактирования.
Поле **Область действия** определяет область действия сценария. По умолчанию установлен уровень **Для экранной формы**, являющийся первой ступенью в иерархии функциональных клавиш.

Поле **Описание** – текстовое поле, занимающее последний столбец в окне **Редактора сценариев функциональных клавиш.**

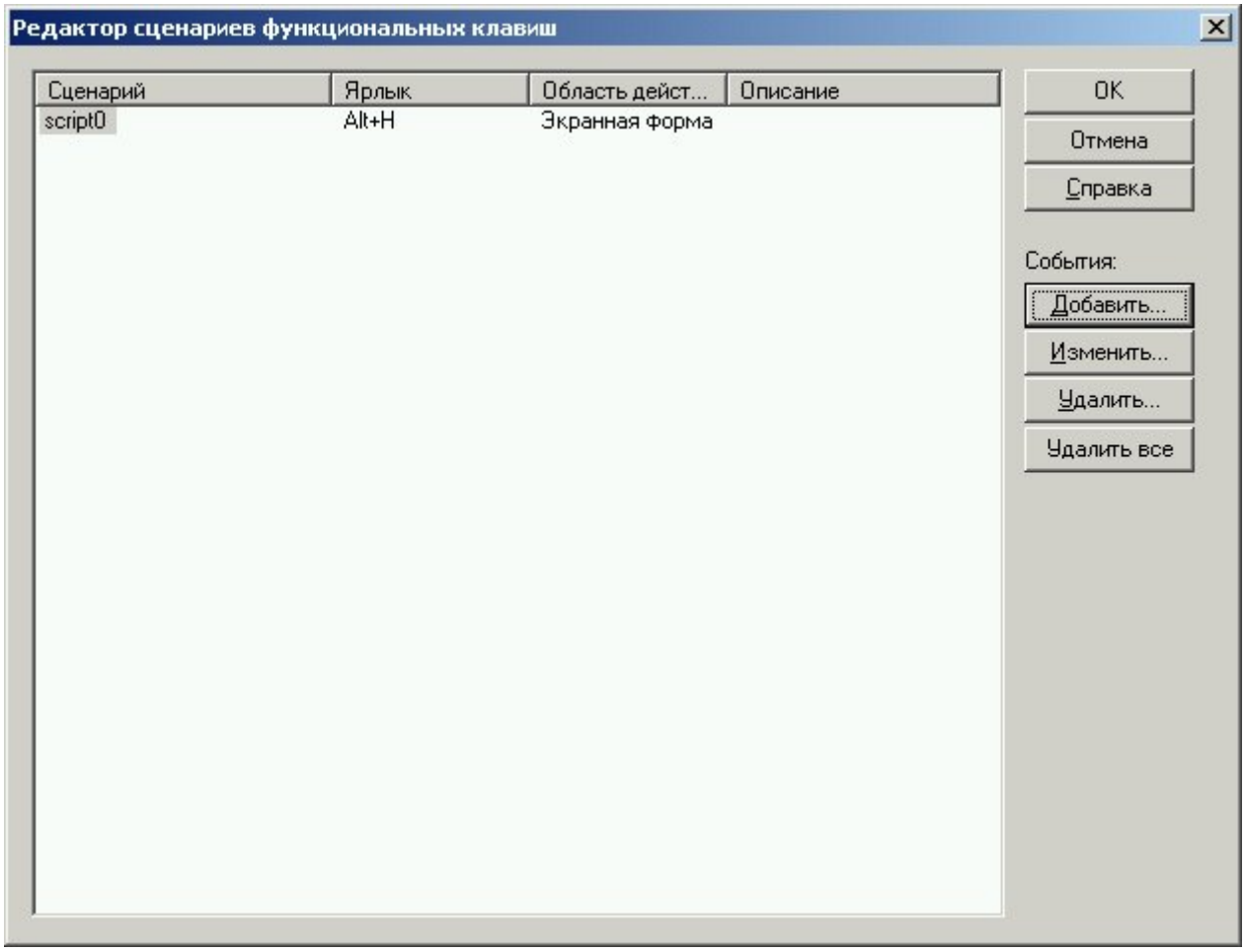


Рис. 9-1. Окно Редактора сценариев функциональных клавиш

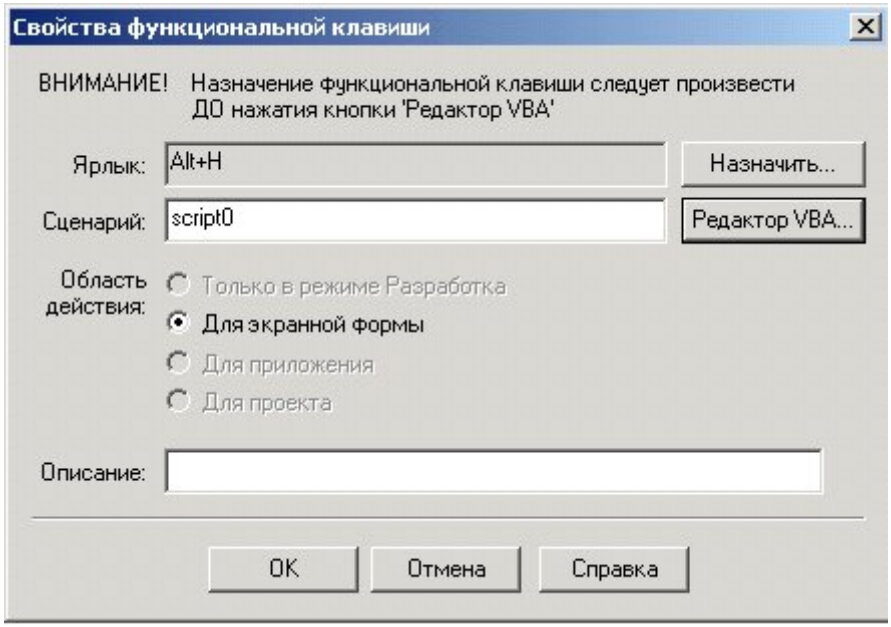


Рис. 9-2. Окно Свойства функциональной клавиши

9.3 Установка рабочего каталога

При выборе пункта **Рабочий каталог** из меню **Сервис** открывается диалоговое окно **Выбор рабочего каталога**, показанное на рисунке 9-3, позволяющее определить каталог, в котором хранятся конфигурационные файлы приложений.

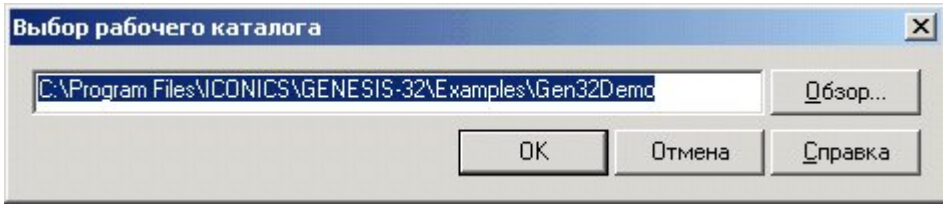


Рис. 9-3. Выбор рабочего каталога

При нажатии на кнопке **Обзор** открывается диалог выбора файла.

9.4 Настройка прав доступа.

При выборе пункта **Настройка прав доступа** из меню **Сервис** будет открыт Конфигуратор сервера безопасности. Доп. информация - в документации на *Конфигуратор сервера безопасности*.

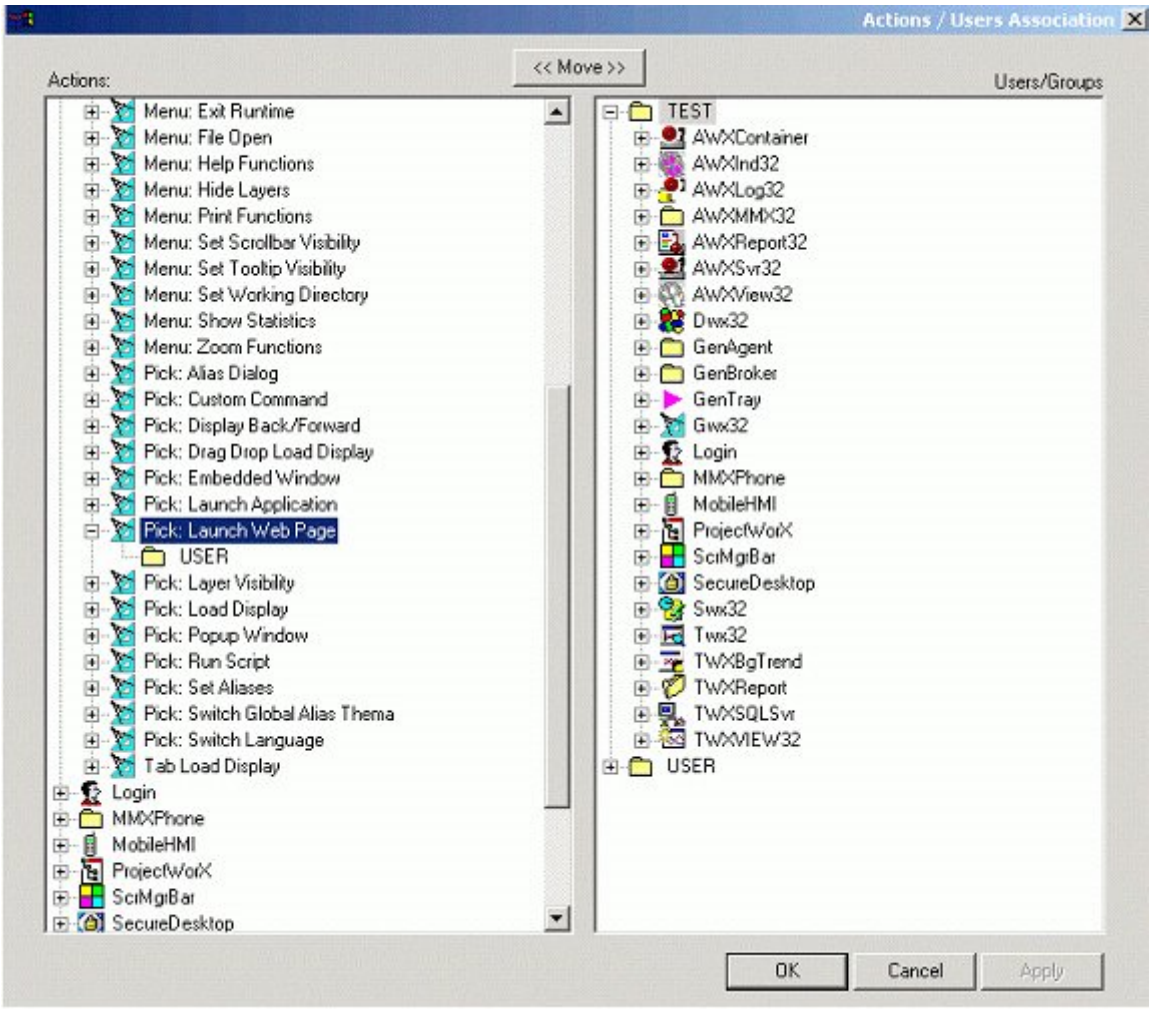


Рис. 9-4 Операции в приложениях (Конфигуратор сервера безопасности)

Отключение сообщений Сервера безопасности и приложения регистрации
Следующие опции позволяют отключать сообщение сервера безопасности «Недостаточные права доступа» и приложение login.exe:
HKLM / GWX32 / Compatibility Properties / SuppressSecurityLogin
HKLM / GWX32 / Compatibility Properties / SuppressSecurityMessageBox

9.5 Редактор псевдонимов

При выборе пункта **Редактор файла псевдонимов** из меню **Сервис** будет открыто диалоговое окно Редактора файла псевдонимов, показанное на рисунке 9-5, в котором можно добавлять, удалять и редактировать псевдонимы. Щелчок на кнопке **Псевдонимы** открывает список псевдонимов экранной формы. Кроме того, можно импортировать и экспортировать псевдонимы с их определениями. По нажатию кнопки **Меню тегов** можно выбрать тег, выражение, локальную или симуляционную переменную в качестве определения псевдонима.

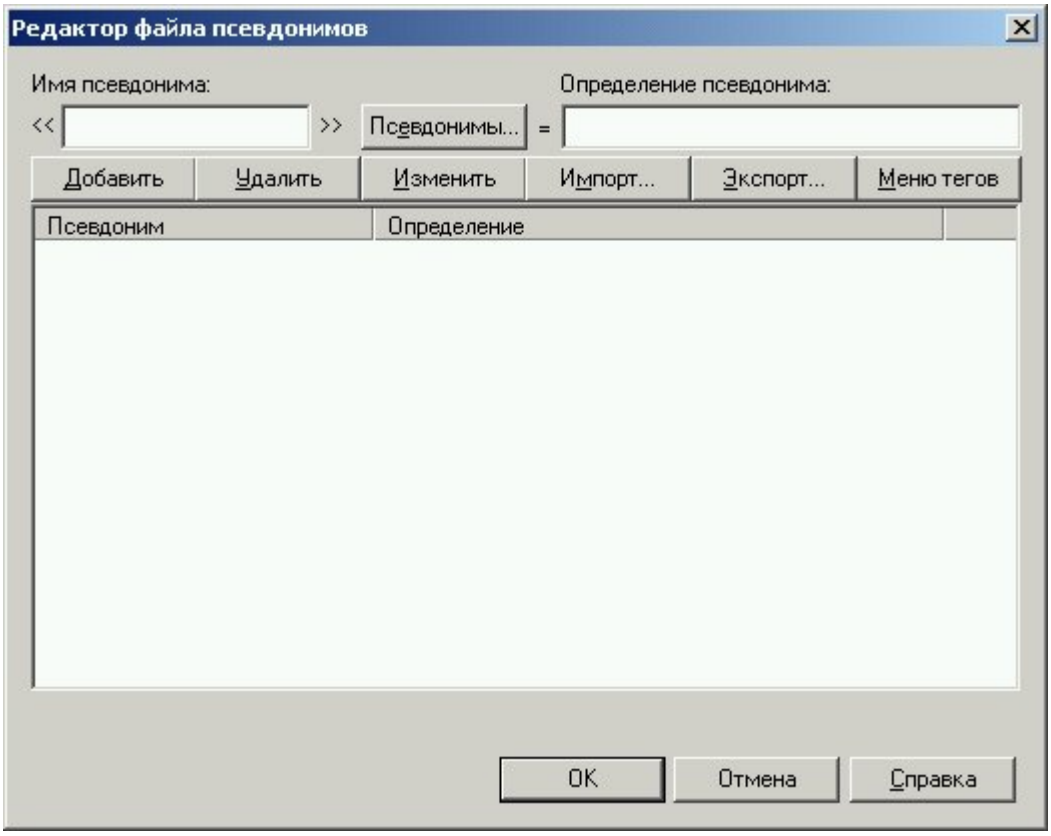


Рис. 9-5. Редактор файла псевдонимов.

9.6 Экспорт в HTML

Мастер Web-публикаций Genesis32 позволяет экспортировать экранные формы GraphWorX32(.gdf), TrendWorX32 (.t32) b AlarmWorX32 (.a32) в файлы HTML и/или публиковать файлы HTML на Web-сервере (в локальной сети или в Интернет). При публикации на Web-сервере WebHMI использует HTML для ссылок на файлы в формате, подходящем для размещения в Интернет. После экспорта экранной формы GraphWorX32 в файл HTML и публикации на Web-сервере, его можно просматривать на компьютере клиента с помощью Интернет-браузера, такого как Microsoft Internet Explorer. Каждая экранная форма GraphWorX32 просматривается как Web-страница.

Публикация экранной формы GraphWorX32:

1. Создать экранную форму GraphWorX32 (файл .gdf) на компьютере с установленным GraphWorX32, где производится разработка проекта.
2. С помощью Мастера Web-публикаций создать файл HTML и опубликовать его на Web-сервере с заданным URL-адресом.
3. На компьютере, являющемся Web-сервером (с установленным GENESIS32), запустить GenBroker Server из GenTray.
4. На компьютере клиента открыть Web-браузер, например, Microsoft Internet Explorer, и задать URL-адрес файла HTML. Этот файл будет загружен на компьютер клиента с Web-сервера. Web-сервер пересылает все необходимые компоненты (например, ActiveX-компоненты) на компьютер клиента, что позволит открыть экран GraphWorX32 в окне браузера. Так можно в реальном времени просматривать экран GraphWorX32 как Web-страницу.

Исходный код публикуемого файла HTML содержит ссылки на элементы plug-in, требующие пересылки GraphWorX32 Viewer ActiveX на компьютер клиента. GraphWorX32 Viewer ActiveX обеспечивает запуск файла .gdf через сеть. Собственно экран GraphWorX32 реально не «преобразуется» в HTML. Вместо этого ActiveX plug-in ссылается на файл .gdf в коде HTML.

Для пояснения пункта 2, указанного выше, приведен пример с использованием Microsoft Internet Explorer. В нем показана часть кода HTML со ссылкой на файл .gdf. Ссылка на файл **GWXview32.cab** для пересылки «plug-in» GraphWorX32 Viewer ActiveX содержится в поле **codeBase.** Это открывает экранную форму .gdf . В примере, приведенном ниже, файл экрана называется **Building1.gdf**, что показано в поле параметра **DisplayName.**

```
<OBJECT classid=clsid:98A5DDE3-563B-11CF-A343-487C03C10000
codeBase="http://www.myserver.com/webhmi/cabs/GWXview32.cab"
id=GWXview321 style="HEIGHT: 420px; WIDTH: 620px" height=420 width=620>
<param name="_Version" value="65537">
<param name="_ExtentX" value="16404">
<param name="_ExtentY" value="11113">
<param name="_StockProps" value="160">
<param name="BorderStyle" value="1">
<param name="Appearance" value="1">
<param name="DisplayName"
value="http://www.myserver.com/webhmi/Building1.gdf">
<param name="UseAmbientBackColor" value="0">
<param name="AutoStartRuntime" value="-1">
<param name="OverrideScrollbarSettings" value="1">
<param name="VerticalScrollbar" value="0">
<param name="HorizontalScrollbar" value="0">
</OBJECT>
```

При экспортировании файла .gdf в HTML создаются ссылки не только на файл GWXview32.cab, как показано в примере, но и на другие файлы .cab («plug-in»), необходимые для других ActiveX-компонентов, встроенных в файл экранной формы (например, TrendWorX32 или AlarmWorX32 Viewer ActiveX).

Перед тем как получить данные с OPC-сервера, на компьютер клиента должны пересылаться файл **IcoSetServer.cab**, содержащий необходимые настройки системы безопасности и лицензирования, а также файл **GenBroker configuration (.gbc or .gbx)**, содержащий сетевые настройки для OPC-соединений. Ссылки на эти файлы содержатся в коде HTML:

<object id="SetServer2" classid="clsid:57802C16-9A15-11D4-B2A8-0090272E599B"
codeBase=http://www.myserver.com/WebHMI/cabs/IcoSetServer.cab height=28 width=17>
<PARAM NAME="CfgName"
VALUE="http://www.myserver.com/WebHMI/Samples/Default.gbc">
</object>
Мастера Web-публикаций Genesis32 автоматически создает все необходимые ссылки в коде HTML. Microsoft Internet Explorer использует файлы .cab.
Примечание: если Вы используете многочисленные фреймы для страниц WebHMI (например, основной «навигационный» фрейм для переключения между страницами и фрейм «с содержанием», включающий в себя «тело» страницы) , убедитесь, что ссылки на файл **IcoSetServer.cab** и файлы **.gbc or .gbx** настроены корректно в исходном коде HTML для основного фрейма. Таким образом, настройки безопасности и лицензирования будут доступны при загрузке компонента.

Пересылка необходимых Web-компонентов клиенту.
WebHMI разработан для применения технологии **Нулевой инсталляции** и **Тонкого Клиента**. Это означает, что компьютер клиента не должен иметь ничего, кроме Windows или Internet Explorer. Поэтому все необходимые Web-компоненты должны пересылаться с удаленного компьютера. Тип и количество компонентов, необходимых для пересылки клиенту, определяется содержимым экранных форм GraphWorX32. Например, экран со встроенным компонентом Trend Viewer требует пересылки соответствующего Web-компонента TrendWorX32 Viewer ActiveX, но не AlarmWorX32 Viewer ActiveX.

Процесс пересылки может происходить медленно из-за низкой скорости в сети. Чтобы избежать потери времени, первое требование заключается в том, чтобы пересылать только необходимые компоненты. Процесс пересылки строго соответствует коду HTML. Внутри кода HTML существуют ссылки на файлы .dpl files или .cab- основание для пересылки компонентов. Таким образом, построение страницы HTML является критическим моментом. МастерWeb-публикаций просматривает экранную форму, определяет, какие компоненты будут нужны на стороне клиента и создает соответствующий код HTML.

Требования к станции клиента.
Для просмотра опубликованных экранов GraphWorX32, на компьютере клиента должен быть установлен Web-браузер (Microsoft Internet Explorer).

Требования к станции разработки.
Для разработки требуется GraphWorX32

Требование к станции Web-сервера.
На компьютере, являющемся Web-сервером, должно быть установлено следующее:
WebHMI
GENESIS32 (с GenBroker)
Для Windows NT, Internet Information Server (IIS) или Personal Web Server
Для Windows 2000 и Windows XP, Internet Information Server (IIS)

Поддержка нескольких экранов.
Предположим, имеется основной экран GraphWorX32 (файл .gdf), который связан с другими экранными формами (например, форма содержит операции «Указание и щелчок», по которым открывается всплывающее окно или загружается другая экранная форма). Когда основной экран публикуется как файл HTML, необходимо, чтобы все связи и ссылки работали корректно, когда страница HTML будет загружена браузером клиента.

МастерWeb-публикаций позволяет определить все связанные файлы экранных форм, проверяя все динамические действия, где встречаются имена файлов в качестве параметров. Поддержка нескольких экранов при Web-публикациях используется для следующих действий динамики:
Загрузка экранной формы
Встроенное окно
Всплывающее окно
Загрузка в стиле «перенести-и-оставить»
Настройка псевдонимов
Диалог настройки псевдонимов

Функция экранной формы Закладки также поддерживается для публикации нескольких экранов GraphWorX32.

Поддержка встроенных ActiveX-компонентов.
МастерWeb-публикаций обеспечивает поддержку встроенных ActiveX-компонентов, входящих в состав экранных форм, связанных с основным экраном, что позволяет клиенту с помощью Web-браузера просматривать встроенные во все экраны ActiveX-компоненты (например, Alarm Viewer ActiveX или Trend Viewer ActiveX).

Применение Мастера Web-публикаций.
Мастер Web – публикаций выполняет две основные функции:
1. Создает HTML – файлы для экранов GraphWorX32 (.gdf), TrendWorX32 (.t32) или AlarmWorX32 (.a32).
2. Мастер или «экспортирует» (сохраняет) HTML – файлы в каталоге, определенном пользователем, на локальном диске, и / или «публикует» (загружает) HTML – файлы на URL-адрес Web – сервера, определенный пользователем (т.е. через Internet или intranet).

Запуск Мастера Web-публикаций (WEB Publishing Wizard).
Для запуска Мастера Web –публикаций:
1. Открыть или создать файл экрана GraphWorX32 (.gdf).
2. Выбрать пункт **Сохранить как** из меню **Файл**. Будет открыто диалоговое окно, позволяющее выбрать имя и расположение сохраняемого файла. Введите имя в поле **Имя файла**. Важно: в поле **Сохранить как** следует выбрать **GraphWorX32 – экраны без VBA**, как показано на рисунке 9-6, т.к. Мастер Web – публикаций не поддерживает Microsoft Visual Basic for Applications (VBA). Нажать кнопку **Save**.

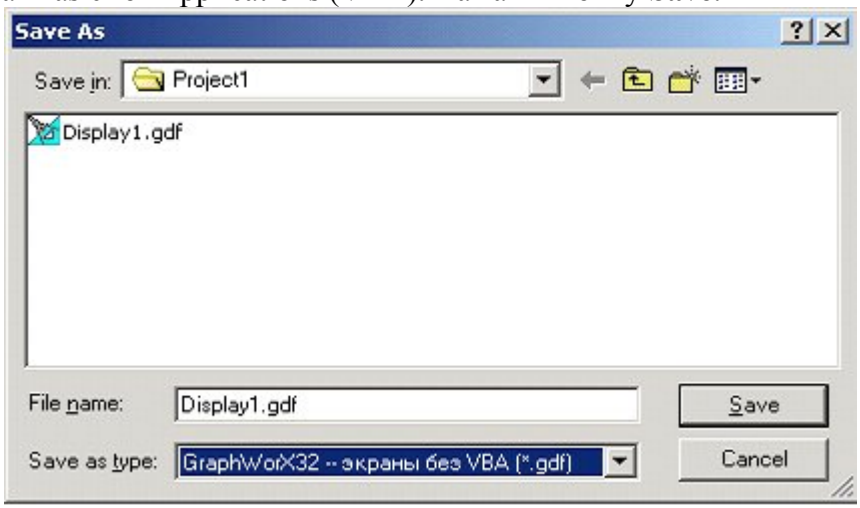


Рис. 9-6 Сохранение экранной формы без VBA

3.Выбрать пункт Экспорт в HTML из меню Сервис.

Примечание. Если попытаться запустить Мастер Web – публикаций для экранной формы, которая была сохранена с VBA, будет выведено следующее сообщение (Рис.9-7):

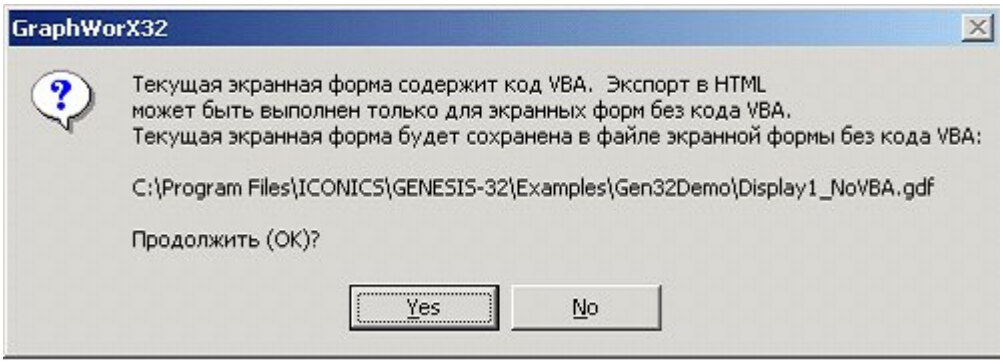


Рис. 9-7

При нажатии ДА (Yes) Мастер Web – публикаций автоматически сохранит экран без VBA, при нажатии НЕТ операция публикации будет отменена.

4. Будет запущен Мастер Web – публикаций.

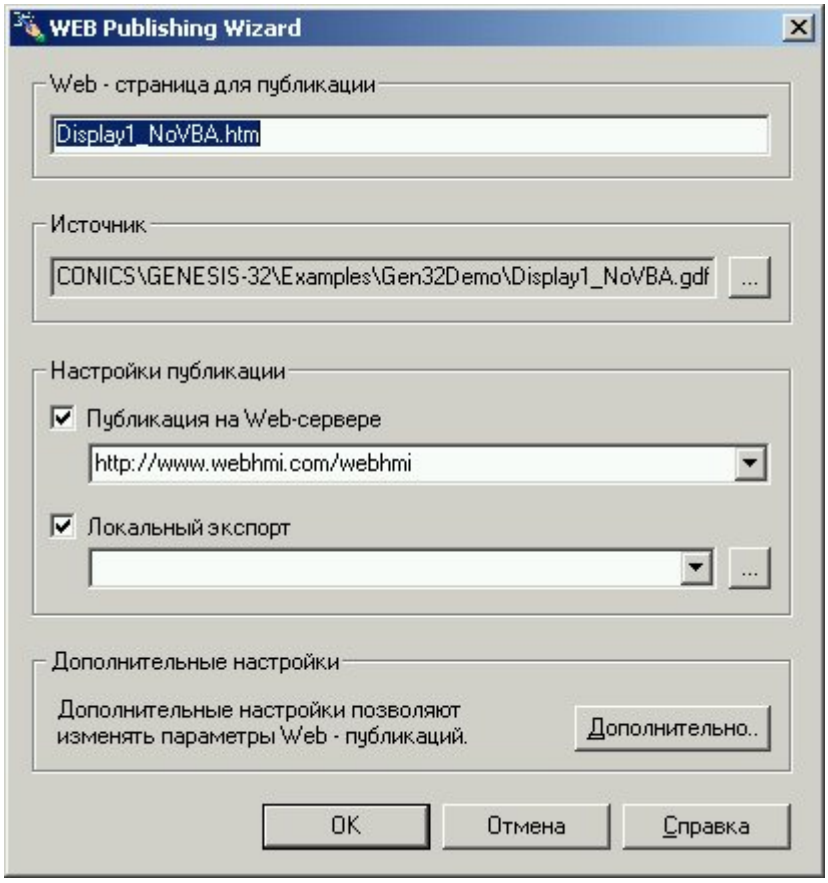


Рис. 9-8 Мастер Web – публикаций

Примечание: оба действия - **Публикация на Web – сервере** и **Локальный экспорт** могут быть выполнены одновременно.

Экспорт локальной копии экранной формы.

1. В поле **Источник н**ужно определить экран GraphWorX32 (.gdf), который будет экспортирован. Для выбора файла нажать кнопку просмотра (...), выбрать файл и щелкнуть **Открыть**. Путь и имя файла появятся в текстовом поле.
2. В поле **Web – страница для публикации** Мастера Web – публикаций необходимо определить имя

нового HTML – файла, который будет создан.
3. В поле **Настройки публикации** установите флажок **Локальный экспорт.** Определите путь к каталогу, куда следует экспортировать HTML – файл. Можно выбрать его, открыв список путей, вводившихся ранее, или воспользоваться кнопкой просмотра (...) для выбора каталога и нажать ОК.
4. Выбранный путь появляется в поле **Локальный экспорт** Мастера Web публикаций.
5. Нажать кнопку ОК для создания HTML - файла. Новый HTML – файл сохранится в каталоге, определенном в поле **Локальный экспорт** Мастера Web – публикаций.
Примечание: оба действия - **Публикация на Web – сервере** и **Локальный экспорт** могут быть выполнены одновременно.

Публикация экрана GraphWorX32 на Web - сервере.

1. В поле **Источник н**ужно определить экран GraphWorX32 (.gdf), который будет опубликован. Для выбора файла нажать кнопку просмотра (...), выбрать файл и щелкнуть **Открыть**. Путь и имя файла появятся в текстовом поле.

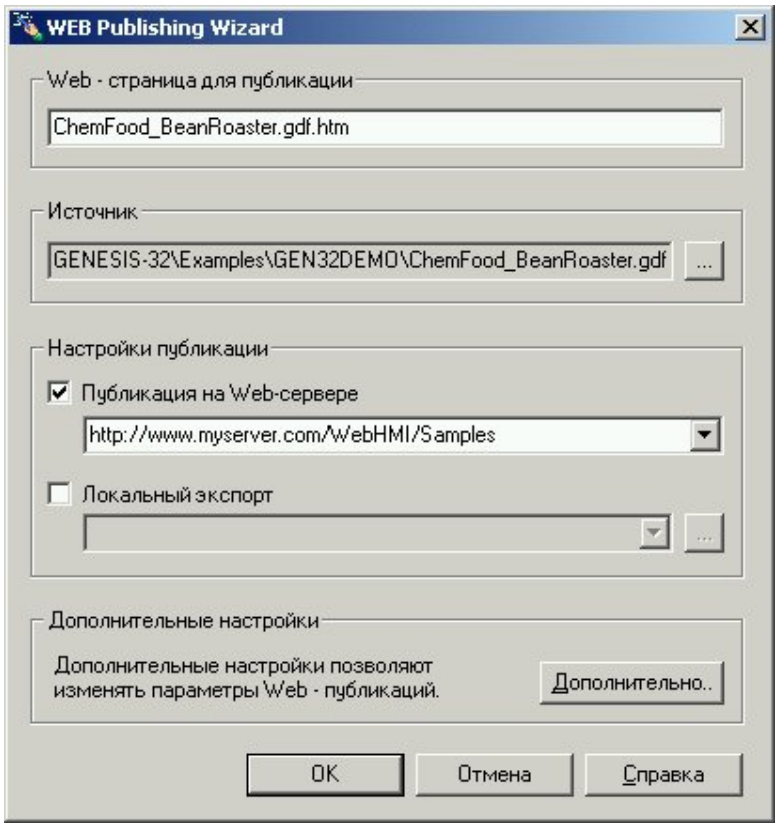


Рис. 9-9 Мастер Web – публикаций

2. В поле **Web – страница для публикации** Мастера Web – публикаций необходимо определить имя нового HTML – файла, который будет создан.
3. В поле **Настройки публикации** установить флажок **Публикация на Web – сервере** и ввести URL – адрес Вашего Web – сервера с полным путем, показывающим, где Вы хотите публиковать HTML - файл. В примере, показанном выше, URL – адрес Web – сервера ("http://www.myserver.com/WebHMI/Samples") состоит из следующих компонентов:

http://www.myserver.com: IP – адрес (имя сервера) WebHMI Server.
WebHMI: Имя корневого каталога сервера WebHMI.
Samples: Имя каталога, где находятся HTML – файлы.

4. URL – адрес Web – сервера появляется в поле **Публикация на Web – сервере** Мастера

Web – публикаций.
5. Нажать кнопку ОК для создания HTML - файла. Новый HTML – файл загружается на Web – сервер и сохраняется по URL – адресу, определенном в поле **Публикация на Web – сервере** Мастера Web – публикаций.

Примечание: оба действия - **Публикация на Web – сервере** и **Локальный экспорт** могут быть выполнены одновременно.

Дополнительные опции публикации.
Мастер Web – публикаций содержит дополнительные опции для экспорта/публикации HTML - файлов. При нажатии на кнопку **Дополнительно** в диалоге Мастера Web – публикаций открывается диалоговое окно **Свойства** Мастера Web-публикаций, содержащее следующие закладки:
Разрешение экрана
Опции публикации
Выбор канала
Переключение серверов
Общие настройки
Общие настройки .
Закладка **Общие** в диалоге **Свойства Web-публикаций** содержит настройки, позволяющие определить исходный адрес URL для сервера WebHMI. Путь к корневому каталогу WebHMI используется в качестве исходного пути URL, только если в диалоге Мастера Web- публикаций установлена опция Экспорт локальной копии. (Если там установлена опция Публикация на Web-сервере, вы должны ввести другой адрес URL для сервера WebHMI).

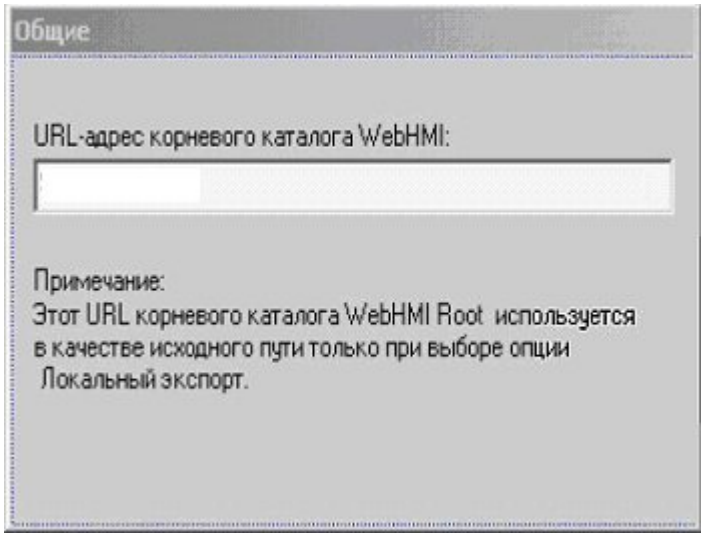


Рис. 9-10 Адрес URL для сервера WebHMI.

Переключение серверов.
Закладка **Переключение серверов** позволяет включать/отключать поддержку GenBroker для экспортированных / опубликованных HTML – файлов. Здесь можно определить, какой из файлов конфигураций GenBroker будет активным. В конфигурационном файле GenBroker устанавливаются настройки OPC – соединений между клиентами и Web – сервером.

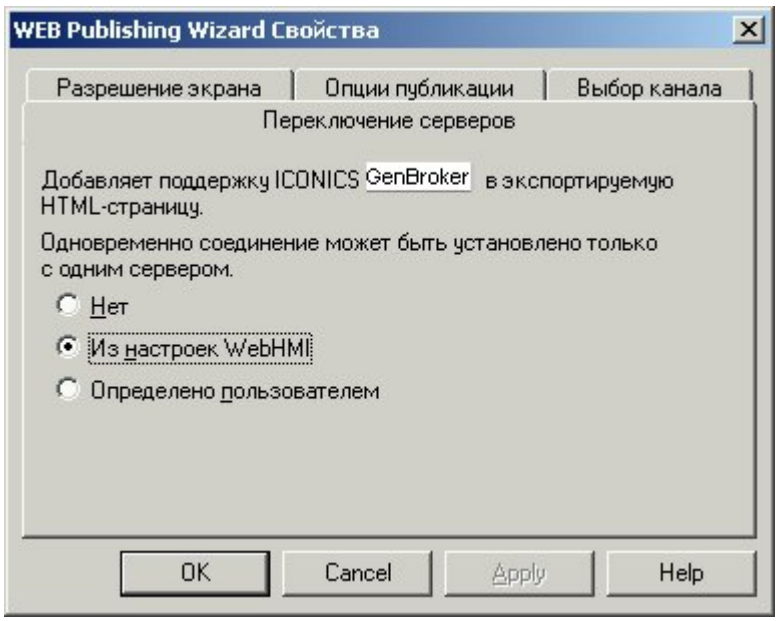


Рис. 9-11 Переключение серверов

Три опции поддержки GenBroker:

Нет: поддержка GenBroker отключена.

Из настроек WebHMI: используется определенный по умолчанию URL – адрес файла конфигурации GenBroker (.gbc или .gbx), заданный WebHMI.

Определенно пользователем: при выборе этой опции поле **URL файла конфигурации GenBroker** становится доступным для редактирования, что позволяет ввести URL – адрес файла конфигурации GenBroker (.gbc или .gbx).

Клиенты могут получать данные от различных серверов WebHMI, но одновременно соединение может быть установлено только с одним сервером. От запрашиваемой клиентом информации зависит, к какому серверу подсоединится клиент. Например, если сервер А не имеет компонентов, необходимых для ответа на запрос клиента, запрос будет переадресован серверу В, серверу С ит.д.

Примечание: дополнительная информация о настройках GenBroker содержится в документации GenBroker Configurator.

Настройки разрешения экрана.

Закладка **Разрешение экрана** определяет разрешение экрана и размер GraphWorX32 Viewer ActiveX control, который связан с HTML – файлом и загружается на компьютер клиента при просмотре HTML – файла браузером. Можно определить размер экрана, введя значения в поля

Ширина и **Высота** в пикселах или процентах, а также определить разрешение экрана, щелкнув на кнопке **Разрешения** и выбрав его из стандартных разрешений экрана.

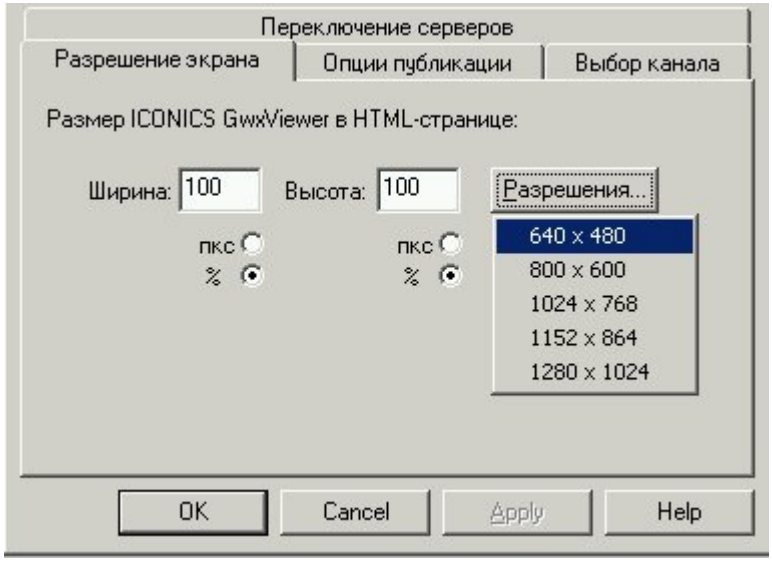


Рис. 9-12 Настройки разрешения экрана

Опции публикации.

Закладка **Опции публикации** в диалоге **Свойства Web-публикаций** содержит настройки, позволяющие улучшить процесс публикации. По умолчанию, Мастер Web-публикаций поддерживает поиск связанных файлов (доп. информ. в разделе **Поддержка нескольких экранов**).Поиск экранов может быть достаточно длительным процессом. Можно ускорить его, отключив опцию **Разрешить поддержку нескольких экранов**. Это особенно полезно, если Вы уже публиковали проект на Web – сервере, и хотите обновить только экран, в который внесли изменения.

Механизм поиска экранов обеспечивает , что Web – страница будет опубликована корректно, в соответствии с исходным файлом. Отключение **Поддержки нескольких экранов** может привести к публикации неполной страницы, и, как следствие, к непредсказуемому поведению. Поэтому есть возможность запретить публикацию страницы. Это можно сделать, включив опцию **Публикация только файла экрана**.

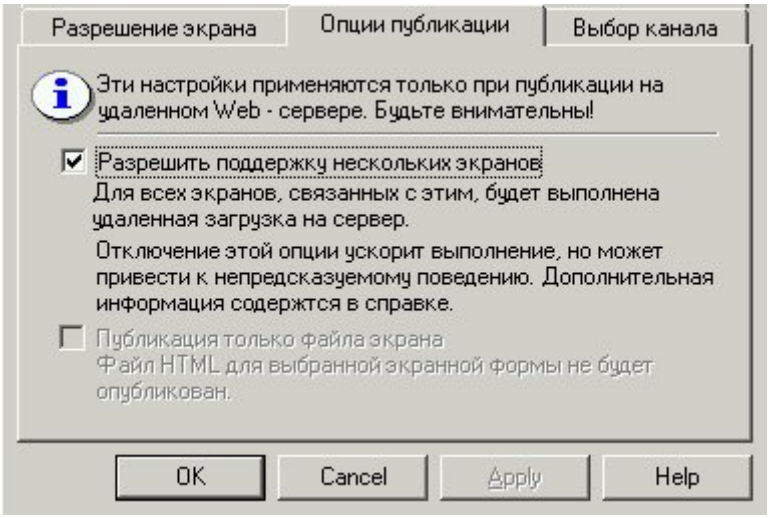


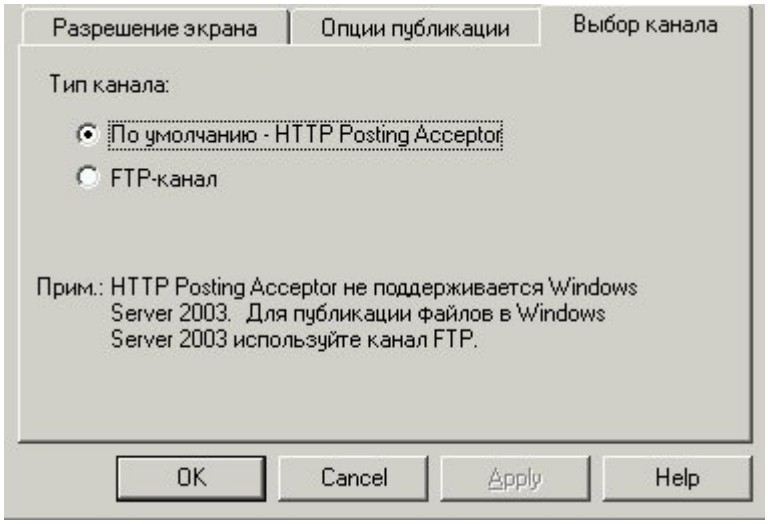
Рис. 9-13 Опции публикации

Выбор канала публикации

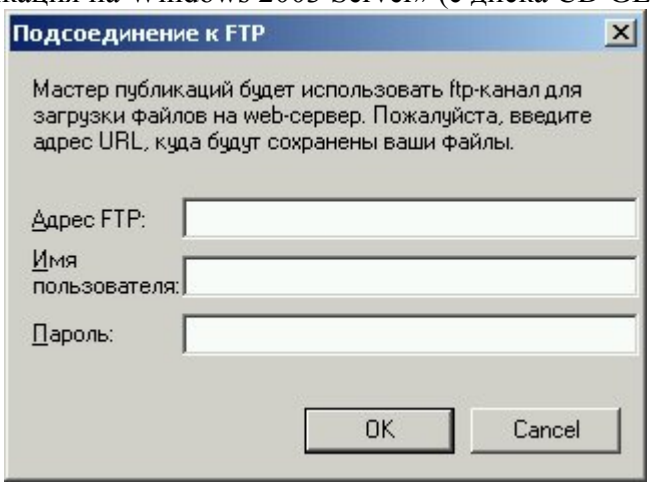
Публикация через компонент Microsoft Posting Acceptor больше не поддерживается Windows Server 2003, поэтому предлагается другая возможность пересылки публикуемых файлов из GENESIS32 на WebHMI-сервер. Это FTP-канал.

Изменение канала соединения
По умолчанию для публикации файлов на Web-сервере используется метод Posting Acceptor. На странице **Выбор канала** диалогового окна **Свойства WEB-публикаций** есть возможность изменить путь файла загрузки на Web-сервер, выбрав **Тип канала** FTP-канал.

Примечание: Выбор типа канала FTP не отменит конфигурацию основного диалогового окна Мастера WEB-публикаций. Установки останутся такими же, как если бы вы осуществляли публикацию файлов, используя метод по умолчанию (т.е. Posting Acceptor).



Конфигурация подсоединения к FTP
Перед загрузкой файлов на Web-сервер нужно определить параметры FTP-соединения, показанные на рисунке. Введите адрес FTP, имя пользователя и пароль.
Примечание: Для получения информации о конфигурации FTP-сайта, воспользуйтесь приложением «Публикация на Windows 2003 Server» (с диска CD GENESIS32).



Просмотр HTML-файлов, экспортированных локально.
Для просмотра файла HTML, который был экспортирован в локальный каталог, следует щелкнуть на нем, и файл будет открыт Web-браузером. Можно также открыть браузер и ввести путь и имя файла в поле Адрес, как показано на рисунке 9-14



Рис. 9-14 Путь и имя файла HTML.

Просмотр опубликованных HTML-файлов.
Перед тем, как файл HTML будет загружен с сервера WebHMI, необходимо запустить на узле сервера WebHMI **GenBroker** с помощью GenTray:
1. Запустить GenTray, выбрав **Start >Programs > ICONICS GENESIS-32 > Tools > GenTray.**
2. После запуска GenTray в строке инструментов Windows появится его иконка в виде треугольника. Щелкните на нем и выберите **GenBroker > Запуск,** как показано на рисунке 9-15. При этом будет запущен GenBroker Server.

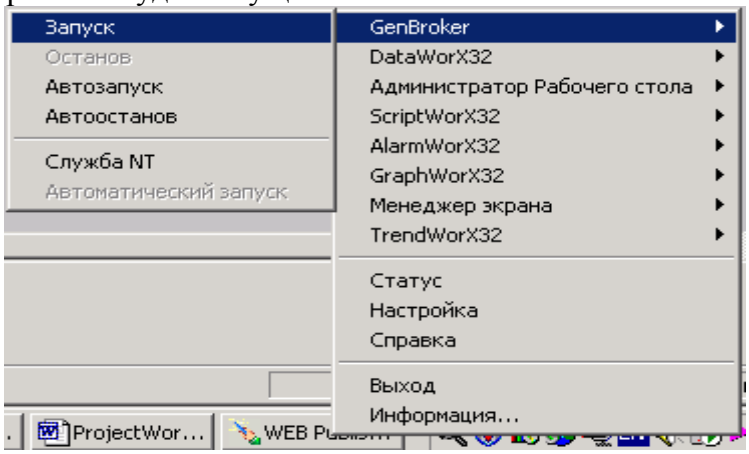


Рис. 9-15 Запуск GenBroker Server

3. После запуска GenBroker Server можно просматривать файлы HTML, опубликованные на Web-сервере, с помощью Web-браузера на любом компьютере клиента. Для этого следует ввести URL – адрес в поле **Адрес** браузера. Например, полный адрес может быть http://www.myserver.com/webhmi/Samples/ChemFood_BeanRoaster.htm. Убедитесь, что в адресе присутствует имя опбликованного файла HTML с расширением "htm". Web-браузер клиента загружает файл HTML, в котором содержатся ссылки на файл .gdf.
Все, что нужно клиенту – это Web-браузер, установленный GENESIS32 на компьютере клиента не нужен. Web-страница загружается с Web-сервера через Internet/intranet и появляется в окне Web-браузера клиента. Значения OPC-тегов динамически обновляются в реальном времени, выглядит экранная форма так же, как и экран GraphWorX32 на сервере.

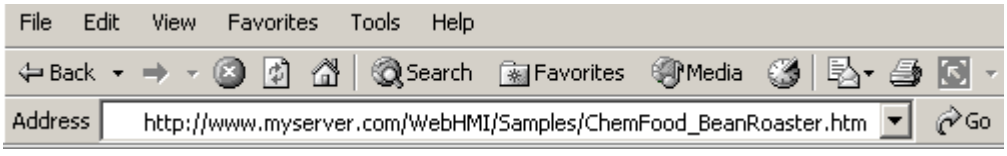


Рис. 9-16 URL – адрес Web-страницы.

9.7 Поддержка Windows CE

Все экранные формы для Pocket GENESIS вначале должны быть разработаны с помощью приложений GENESIS32 (GraphWorX32, TrendWorX32, AlarmWorX32) на настольном компьютере, и затем сохранены в формате, совместимом с Pocket PC и Windows CE. После этого файл экранной формы загружается в Pocket PC или в устройство с Windows CE для просмотра в реальном времени. Если файл экранной формы запущен в соответствующем приложении Pocket GENESIS (например, Pocket GraphWorX, Pocket TrendWorX, или Pocket AlarmWorX) в режим исполнения, то могут исполняться любые динамические функции, созданные при конфигурировании. За несколькими исключениями, в режиме исполнения приложений Pocket GENESIS поддерживаются те же функции, что и для приложений GENESIS32.

Разработка для Windows CE в GENESIS32.

Если необходима разработка для Windows CE в GraphWorX32, TrendWorX32 и AlarmWorX32, следует выбрать пункты меню **Сервис - Windows CE - Разработка для Windows CE**. Если инсталлирован только Pocket GENESIS, пункт установлен по умолчанию.
После создания экранной формы следует выбрать пункт **Сохранить как** из меню **Файл**, и в появившемся диалоговом окне **Сохранить как**, показанном на рисунке 9-17, задать имя файла и один из следующих типов:
В GraphWorX32 выбрать "GraphWorX32 экраны для WinCE (*.gdc)" или "GraphWorX32 шаблоны для WinCE (*.gdc)."
В AlarmWorX32 выбрать "AWXview32 CE File (*.awv)."
В TrendWorX32 выбрать "TWXviewerCE File (*.tce)."
Нажать кнопку Сохранить. При этом все функции, не поддерживаемые Pocket PC или устройством Windows CE, будут удалены из экранной формы Pocket GENESIS.
Примечание: при открытии экранной формы Pocket GENESIS в GENESIS32 следует определить правильное расширение файла Windows CE.

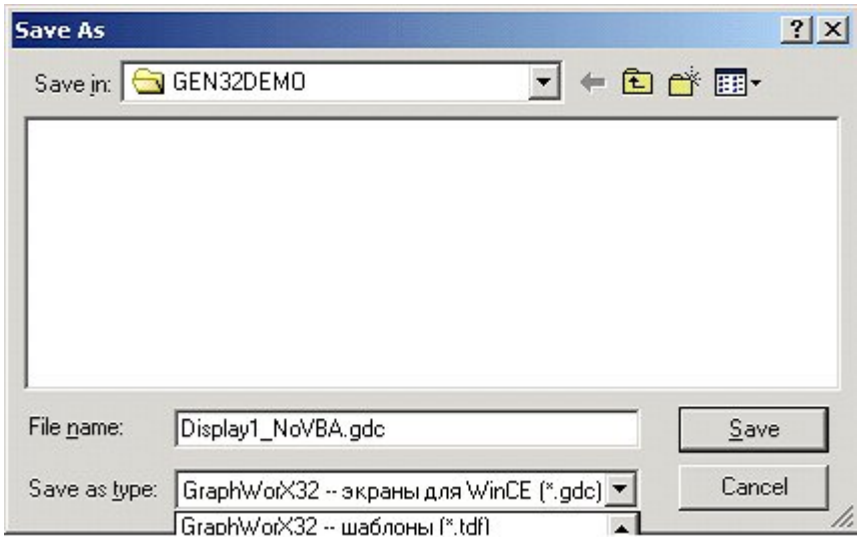


*Рис. 9-17 Окно **Сохранить как***

Удаленные соединения.
Для удаленного подключения к OPC-серверам через сеть, необходимо установить GenBroker. Дополнительная информация по настройке GenBroker содержится в соответствующей документации.

Загрузка конфигурационных файлов GENESIS32 в Pocket PC.
После создания и сохранения экранной формы Pocket GENESIS необходимо загрузить конфигурационные файлы GraphWorX32, TrendWorX32, и AlarmWorX32 в Pocket PC или устройство Windows CE с помощью программы Download to Windows CE Utility.
Для соединения с устройством и загрузки файла используется программа Microsoft ActiveSync.

Настройка загрузки.
Инструмент, выполняющий загрузку, требует наличия модулей как на настольном PC или рабочей станции, так и на Pocket PC или устройстве Windows CE, т.к. сервисы Microsoft ActiveSync используются для соединения и аутентификации устройства CE.

Настройка настольного компьютера.
Настольный компьютер – единственное устройство, где необходимо производить настройку. Если инструмент загрузки установлен корректно, будет выведен список модулей ActiveSync в закладке **Sync Options** из диалогового окна Windows **Options.**
Необходимо разрешить загрузку файлов с помощью модуля ActiveSync, установив флажок Download, как показано на рисунке 9-18. Для синхронизации загрузки необходимо установить связь с устройством CE (рис.9-19).
Если устройство CE отвечает без проблем, модуль синхронизации загрузки становится активным. Если загрузка разрешена, конфигурационный файл для Windows CE будет немедленно загружен в устройство CE. Если нет, то только сохранен.
Дополнительная информация содержится в документации на Microsoft ActiveSync.

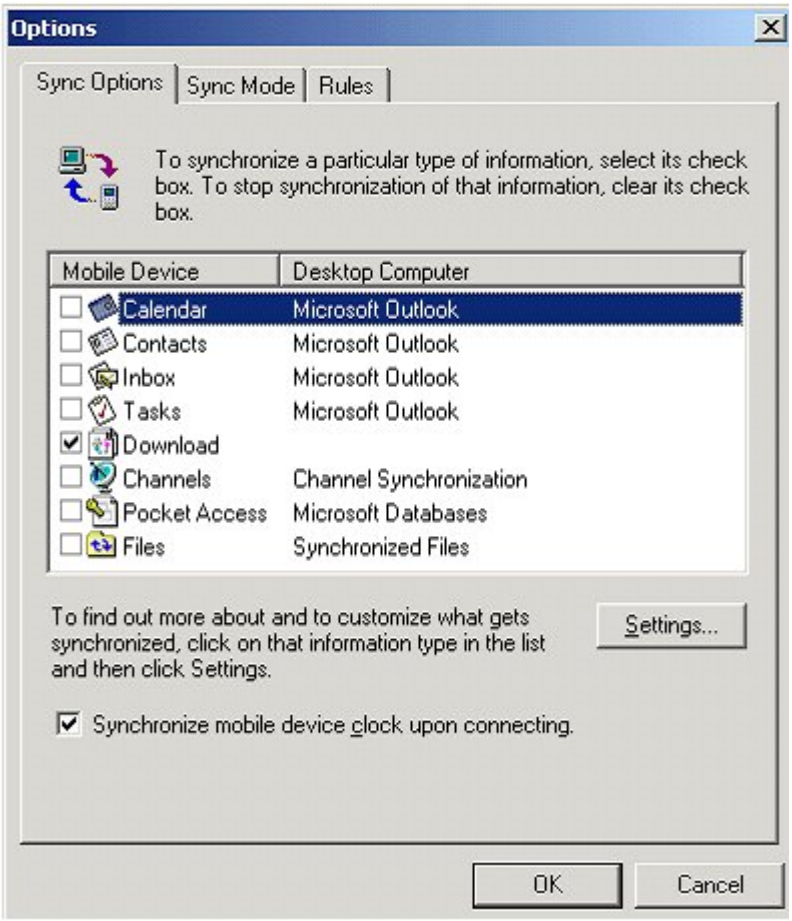


Рис. 9-18 Диалоговое окно Windows Options

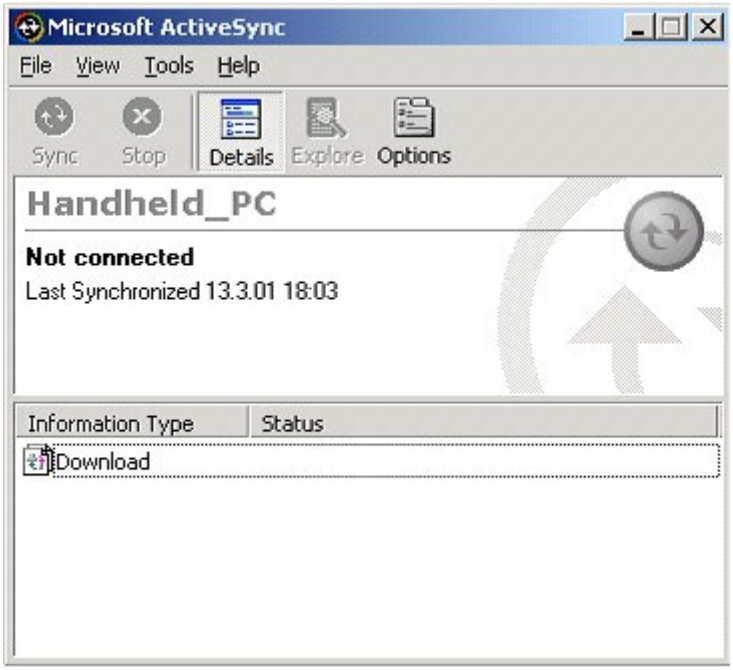


Рис. 9-19 Microsoft ActiveSync

Настройка загрузки файлов в приложениях.
В GraphWorX32, TrendWorX32, и AlarmWorX32 есть собственная версия инструмента загрузки. При выборе пунктов меню **Сервис - Windows CE - Параметры для Windows CE** будет открыто диалоговое окно **Параметры для Windows CE ,** показанное на рисунке 9-20, где можно включить или отключить инструмент загрузки файлов (флажок **Загрузка в устройство при сохранении**). Кроме того, следует определить **Каталог загрузки** – целевой каталог на устройства CE.

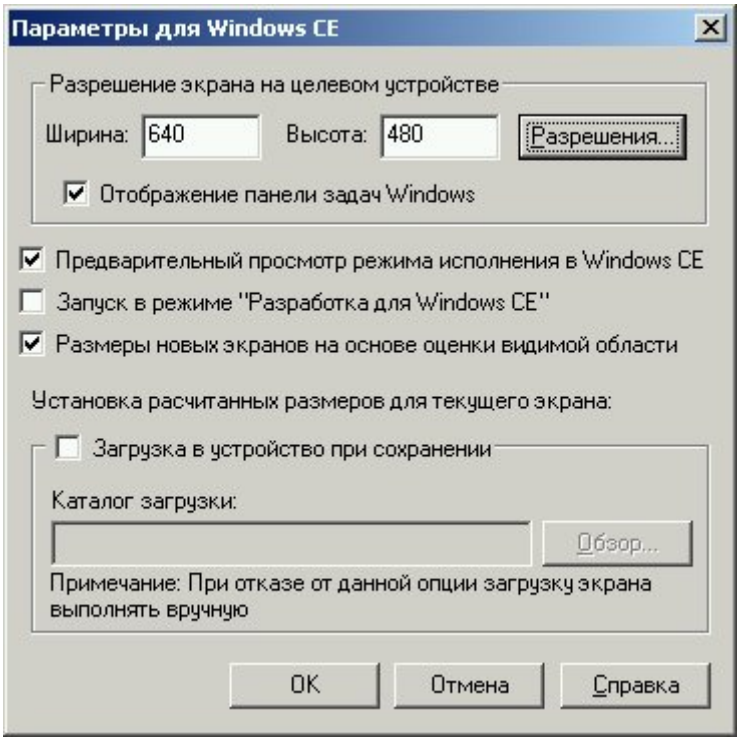


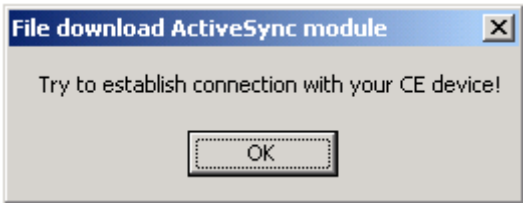
Рис. 9-20 Параметры для Windows CE

Можно также нажать кнопку Обзор для выбора каталога. При этом будет открыто диалоговое окно **Browse for Folder,** в котором показаны каталоги устройства CE.



Рис. 9-21 Просмотр файлов

Если модуль ActiveSync загрузки файла неактивен, будет выдано предупреждающее сообщение.



Для установления соединения с устройством CE нажмите OK.
В диалоговом окне **Параметры для Windows CE** можно настроить другие параметры для устройства CE. Чтобы выбрать разрешающую способность целевого устройства (от 240x320 до 1024 x 768), необходимо нажать кнопку **Разрешения**.

Загрузка файла в устройство CE.
После создания экранной формы следует выбрать пункт **Сохранить как** из меню **Файл**, и в появившемся диалоговом окне **Сохранить как**, показанном на рисунке 9-17, задать имя файла и один из следующих типов:
В GraphWorX32 выбрать "GraphWorX32 экраны для WinCE (*.gdc)" или "GraphWorX32 шаблоны для WinCE (*.gdc)."
В AlarmWorX32 выбрать "AWXview32 CE File (*.awv)."
В TrendWorX32 выбрать "TWXviewerCE File (*.tce)."
Будет открыто диалоговое окно **Импорт файла**, показывающее загрузку файлов в устройство Windows CE.

Запуск приложений Pocket GENESIS на Pocket PC.
После загрузки конфигурационных файлов GraphWorX32, AlarmWorX32, или TrendWorX32 из настольного компьютера в Pocket PC или устройство Windows CE, можно запускать приложения на Pocket PC. Клиентские приложения Pocket GENESIS (Pocket GraphWorX, Pocket AlarmWorX, and Pocket TrendWorX) размещаются в каталоге **ICONICS/Pocket GENESIS** на Pocket PC. Откройте приложение, которое нужно запустить и выберите **Open** из меню **File.** Файлы имеют расширения .gdc, .awv, or .tce.
Пример с использованием Pocket GraphWorX:
1. Открыть Pocket GraphWorX из каталога **ICONICS/Pocket GENESIS** на Pocket PC. Выбрать **Open** из меню **File.**

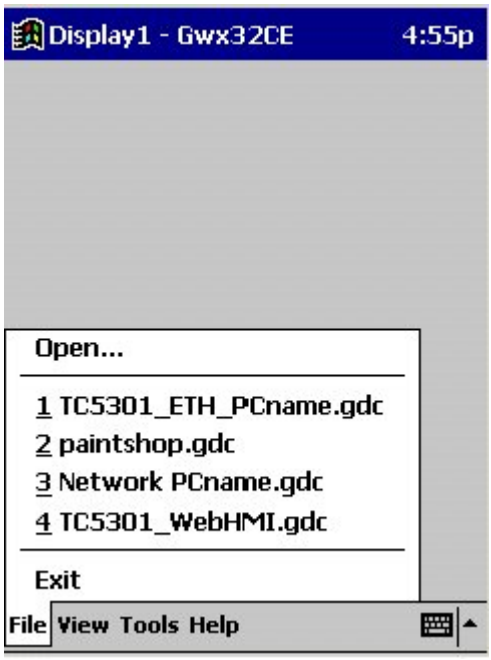
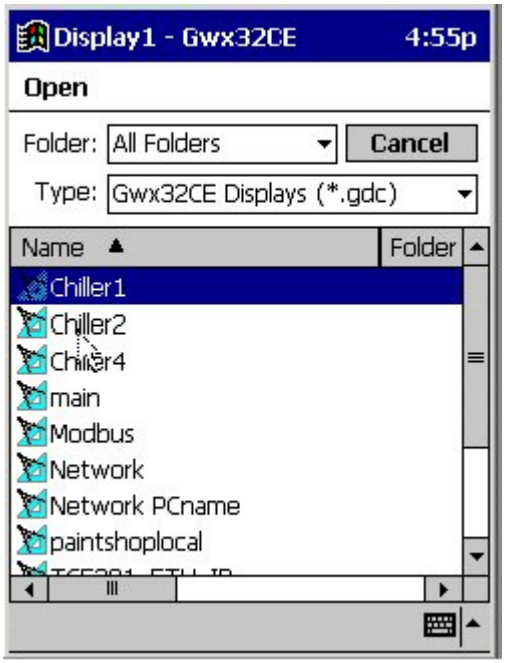


Рис. 9-21 Запуск приложений Pocket GENESIS

2. Найти файл экрана Pocket GraphWorX (.gdc) с помощью обзора. В данном примере выбран **chiller1.gdc.**



3. Экранная форма открывается в Pocket GraphWorX, как показано на рисунке. Запуск Pocket GENESIS позволяет просматривать в реальном времени данные OPC.



Примечание: процедура открытия и просмотра файлов в Pocket AlarmWorX и Pocket TrendWorX выполняется таким же образом.

Ядро исполнения сценариев и редактор сценариев VBScript and Jscript.
GraphWorX32 содержит встроенный редактор Visual Basic for Applications (VBA), в котором можно создавать сценарии VBA. Это мощный инструмент, позволяющий существенно расширять возможности GraphWorX32. Для того, чтобы можно было исполнять сценарий VBA, на компьютере должно быть **Ядро исполнения сценариев VBA** (VBA Runtime Engine). Это ядро не может пересылаться через Интернет на компьютер с «тонким клиентом», т.к. оно занимает большое количество места на диске и требует большого времени на загрузку. Поэтому в прошлом ICONICS приняла решение удалить поддержку VBA из WebHMI и из других «облегченных» продуктов, таких как Pocket GraphWorX. Таким образом, использовать мощность VBA и преимущества «тонкого клиента» в WebHMI невозможно. Многие пользователи разрабатывают два различных варианта экранных форм: с VBA для настольных компьютеров и без VBA для тонкого клиента WebHMI. Это требует дополнительных трудозатрат по переработке экранной формы в связи с удалением VBA.
В качестве решения этой проблемы было добавлено новое ядро исполнения кода VBScript и Jscript. Это облегченное ядро может легко передаваться через Интернет на любые компьютеры «тонких клиентов». Если Вы собираетесь использовать технологию сценариев для расширения возможностей GraphWorX32 и технологию «тонкого клиента», например, WebHMI, следует писать сценарии на VBScript или Jscript вместо VBA и использовать новое встроенное ядро исполнения сценариев.
Новое ядро исполнения сценариев не содержит всех функций ядра исполнения VBA, поэтому некоторые возможности не поддерживаются. Есть определенные ограничение, но основное заключается в том, что невозможно управлять событиями, приходящими от ActiveX или других элементов управления, содержащихся в экране GraphWorX32.
Новое ядро исполнения сценариев позволяет управлять наиболее важными событиями экранной формы и действий управления, но вызывать новые события в настоящий момент невозможно.
Еще одно ограничение состоит в том, что VBScript и Jscript не поддерживают формы, поэтому невозможно создавать элементы интерфейса пользователя, отличающиеся от простых окон сообщения или ввода.

Примечание: Перечисляемые константы не могут быть использованы в редакторе VBScript. Поэтому должны использоваться соответствующие числовые переменные. Например, следующий код открывает экранную форму и устанавливает значения псевдонима путем чтения их из файла.
ThisDisplay.FileOpenSetAliases("myFileName.gdf", AliasSetFromFile, "AliasCommandString.txt")
В сценариях VBScript и Jscript необходимо заменить константы AliasSetFromFile соответствующими числовыми переменными, например 1.
Т.е. рассматриваемый код в VBScript примет вид:
ThisDisplay.FileOpenSetAliases("myFileName.gdf", 1,"AliasCommandString.txt")

Редактор сценариев.
Редактор сценариев отображается в экране GraphWorX32 как панель инструментов. Она может перемещаться по экрану (плавающая панель) или быть фиксированной на любой из четырех сторон экрана. Можно изменять размеры панели (как фиксированной, так и плавающей) путем «перетаскивания» ее границ. Поскольку Редактор сценариев – встроенная панель инструментов, не нужно ее «открывать». Вместо этого ее можно «показывать» или «скрывать», отмечая пункт **Скрыть панель скриптов** из меню **Вид**, или нажимая комбинацию клавиш **CTRL+ALT+J.**
Кроме того, можно переключать видимость панель инструментов Редактора сценариев щелчком на иконке «карандаш» на основной панели инструментов.

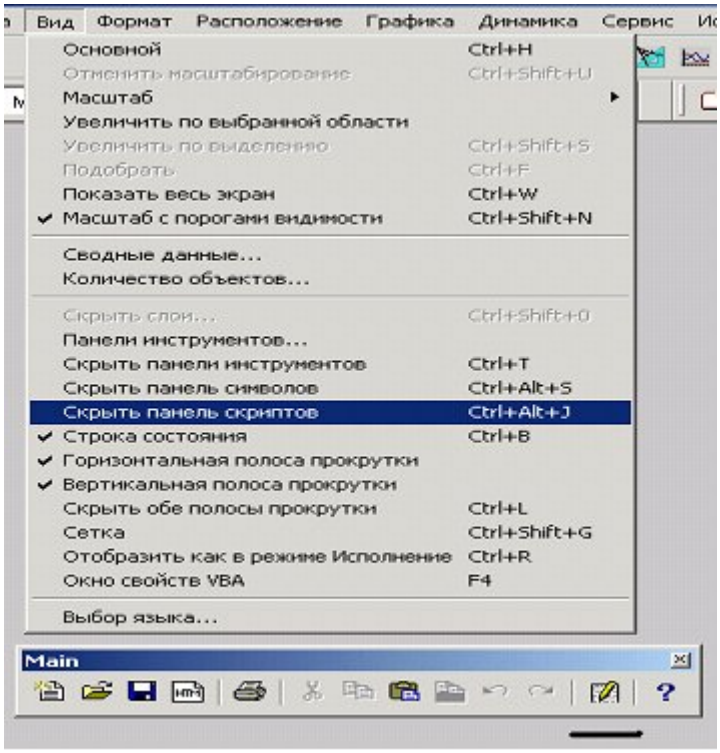


Рис. 9-22

Вид редактора сценариев показан на рисунке 9-23. Окно редактора разделено на левую и правую панель. Левая панель содержит список сценариев, правая – код для выбранного сценария. Каждый сценарий на левой панели помечается - VBScript **(VB)** или JScript **(JS).**

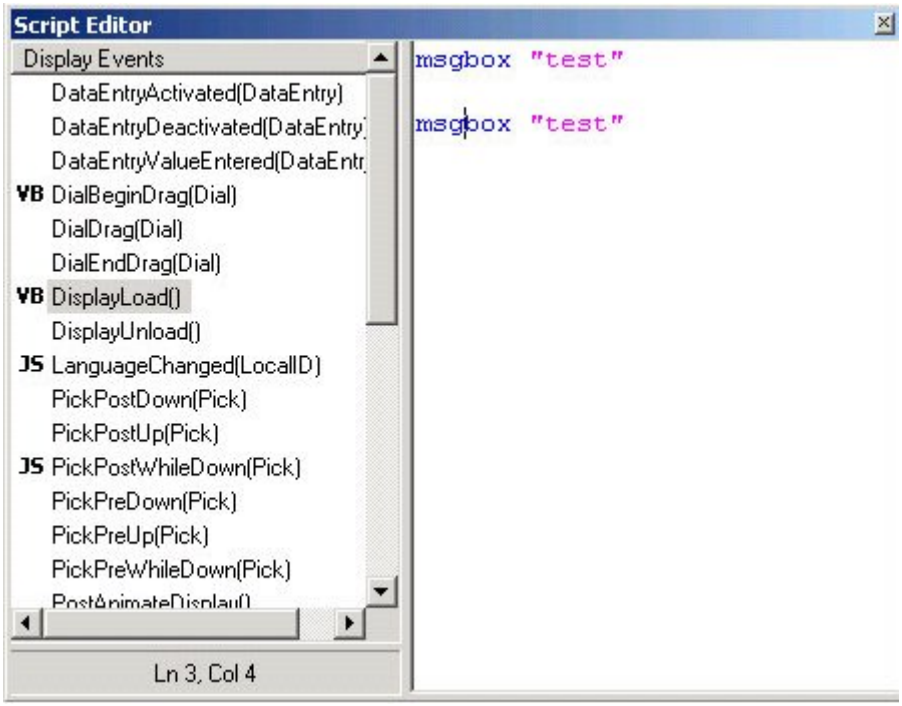


Рис. 9-23 Редактор сценариев в режиме событий

Редактор сценариев содержит два основных режима редактирования – режим событий и режим объектов. В первом случае на левой панели отображается список всех событий, по которым могут запускаться сценарии, во втором – список объектов экранной формы. Переключение режимов Редактора сценариев производится щелчком на заголовке левой панели.

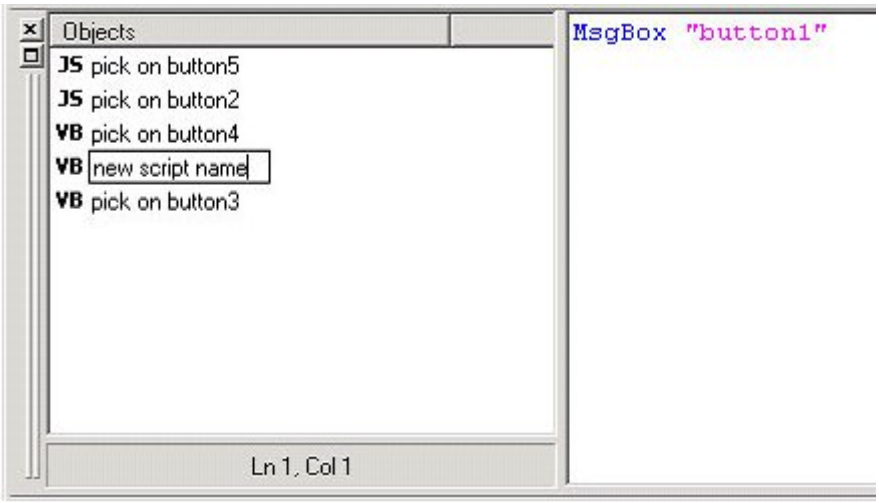


Рис. 9-24 Редактор сценариев в режиме объектов

Сценарий и соответствующий ему объект в экранной форме синхронизированы. Поэтому, если вы в экранной форме выбираете объект, с которым связан сценарий, то сценарий будет отображаться в Редакторе сценариев автоматически. И наоборот, если в Редакторе сценариев выбран сценарий, соответствующий объект экранной формы тоже будет выделен.

Режим событий.

Когда Редактор сценариев находится в Режиме событий, на левой панели отображается список всех событий, по которым могут запускаться сценарии. Полный список событий находится в документации GraphWorX (Справочник по OLE Automation)

Для того, чтобы связать сценарий с событием щелкните на событии в списке Редактора сценариев. Появляется выпадающее меню, позволяющее выбрать язык, который Вы предполагаете использовать для создания сценария (VBScript or JScript).

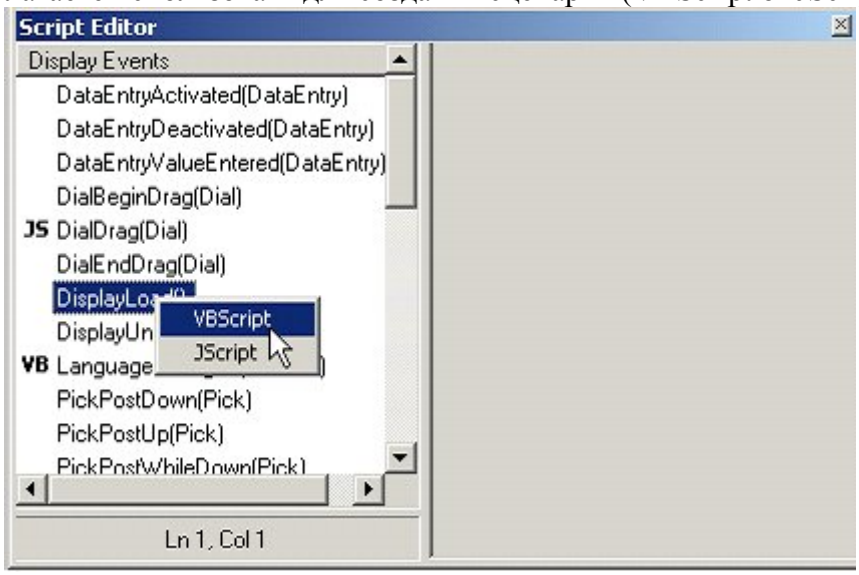


Рис. 9-25 Выбор языка сценария.

После выбора типа сценария можно начинать его редактирование. При следующем открытии этого меню тип сценария будет отмечен флажком. При необходимости можно изменить тип сценария.
Примечание: если вы снимете этот флажок, сценарий будет удален. Перед удалением будет выведен диалог для подтверждения удаления.

Режим объектов.
Когда Редактор сценариев находится в Режиме объектов, можно редактировать сценарии, связанные с объектом. Сценарий можно связывать с объектом через действие динамики «указание и щелчок»- действие, которое выполняется каждый раз, как произведен щелчок на объекте. На левой панели в этом случае отображается список объектов со сценариями – т.е. объектов, которые имеют динамику «указание и щелчок» с действием **Выполнить сценарий VBA** , если установлен тип сценария - **VBScript or JScript**.
В Редакторе сценариев не показываются объекты с действием **Выполнить сценарий VBA** и типом VBA. Сценарии VBA редактируются стандартным редактором VBA.

Для создания нового сценария и связи его с объектом необходимо:
1. Добавить к объекту динамику «указание и щелчок».
2. В Инспекторе свойств выбрать действие **Выполнить сценарий VBA** и имя сценария в поле **Сценарий**. Если имя сценария не будет указано, GraphWorX32 использует в качестве него имя объекта «указание и щелчок (Если имя объекта тоже отсутствует, в качестве имени сценария используется слово **«Script»**).
3. Выбрать тип сценария из выпадающего меню в поле **Тип**:
4. Нажать **ОК**. Если был выбран тип VBScript or JScript, в Редакторе сценариев будет создан новый сценарий, и его имя появится в окне Редактора сценариев. Если Редактор сценариев находится в Режиме событий, он автоматически переключится в Режим объектов, и вновь созданный сценарий будет выделен. После этого можно начинать редактирование сценария.

Изменение имени сценария.
Имя сценария можно изменить в любое время как через страницу **Свойств объекта**, так и в Редакторе сценариев. Если Редактор сценариев находится в Режиме объектов, можно

переименовать выделенный сценарий, щелкнув на его имени в списке на левой панели Редактора сценариев.
Примечание: переименовать сценарий невозможно, если Редактор сценариев находится в Режиме событий. Исходные имена событий изменять нельзя.

Основные операции с текстом.
При щелчке правой клавиши мыши на правой панели появляется меню, содержащее основные операции с текстом: **Отменить, Вырезать, Копировать, Вставить, Удалить и Выбрать все**.

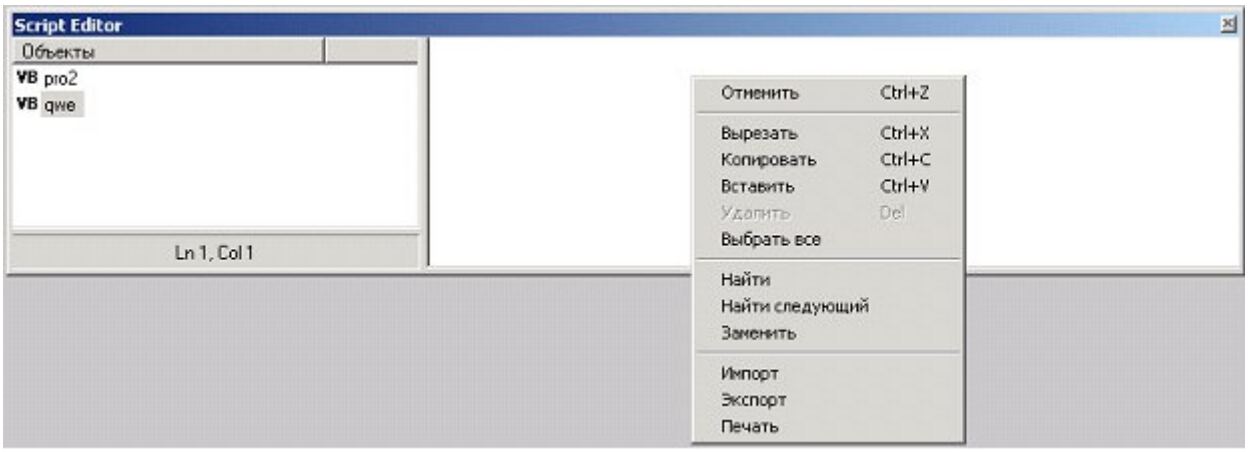


Рис. 9-26 Операции с текстом

Поиск и замена текста.
Функции поиска и замены текста вызываются при выборе пунктов **Найти, Найти следующий** и **Заменить** из меню, открывающегося при щелчке правой клавиши мыши на правой панели. Можно выбрать дополнительные опции, позволяющие **Учитывать слово целиком** и **Учитывать регистр**. Также можно указать направление поиска, и необходимость заменять только первый фрагмент найденного текста или все фрагменты в сценарии. Поиск по последнему введенному шаблону производится при выборе из меню пункта **Найти следующий.**

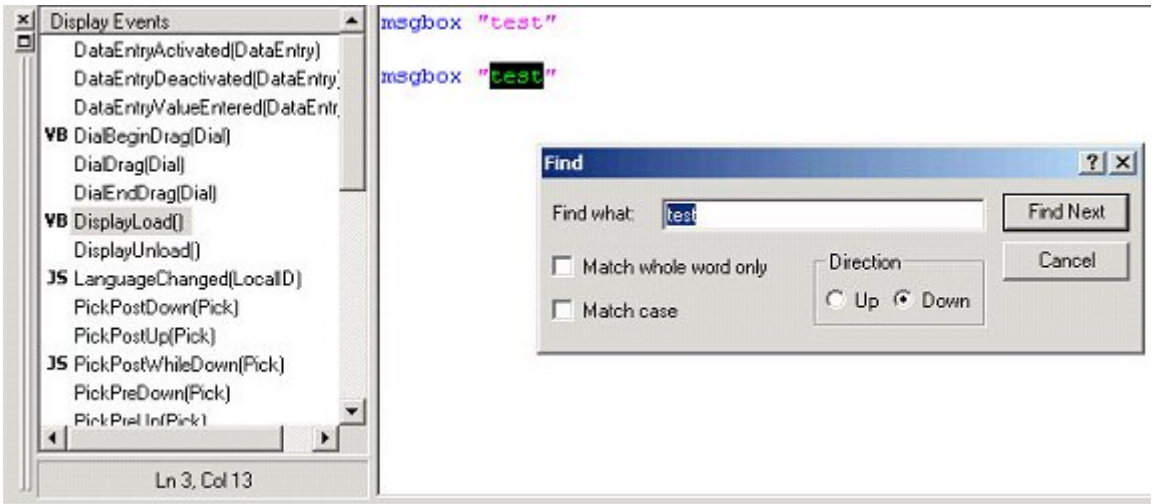


Рис. 9-27 Поиск и замена

Импорт и экспорт сценариев.
С помощью пунктов меню **Импорт** и **Экспорт** можно экспортировать сценарий в текстовый файл и импортировать из любого текстового файла. Также можно импортировать сценарий из стандартных файлов VBScript и JScript (*.vbs и *.js). Экспортировать сценарий можно из текстового файла или файлов VBScript и JScript. При экспорте GraphWorX32 автоматически добавляет заголовок, содержащий дату и время, создания файла, имя и тип сценария. Если сценарий связан с событием, в качестве имени сценария указывается название события.

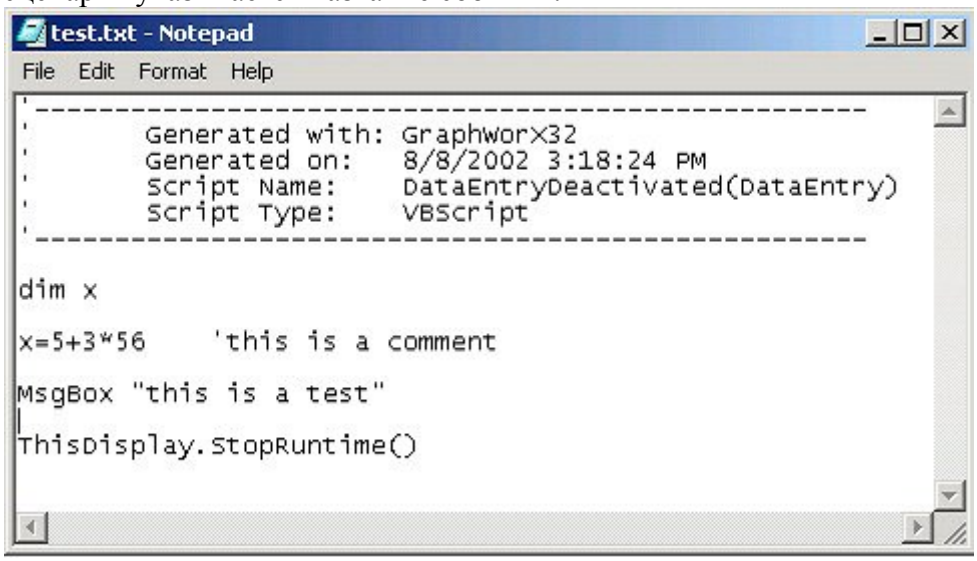


Рис. 9-28 Экспорт сценария.

Печать сценариев.
Можно печатать сценарий непосредственно из Редактора сценариев, выбрав пункт **Печать,** из меню, открывающегося при щелчке правой клавиши мыши на правой панели. При этом будет открыто стандартное диалоговое окно печати Windows, в котором можно установить настройки принтера: принтер, количество копий. Не поддерживается режим печати нескольких страниц или выделенного фрагмента текста. Нумерация страниц также не поддерживается.

Параметры и настройки.
Для Редактора сценариев могут быть сделаны пользовательские настройки. Для этого следует выбрать меню Формат – Параметры приложения - Редактор сценариев.
Можно изменять следующие параметры Редактора сценариев:
Тип, размер и стиль шрифта;
Ключевые слова и константы VBScript;
Ключевые слова и константы JScript;
Цвет для ключевых слов, констант, строк, чисел и комментариев;
Разрешать/запрещать выделение цветом;
Разрешать/запрещать автоматическую смену регистра.

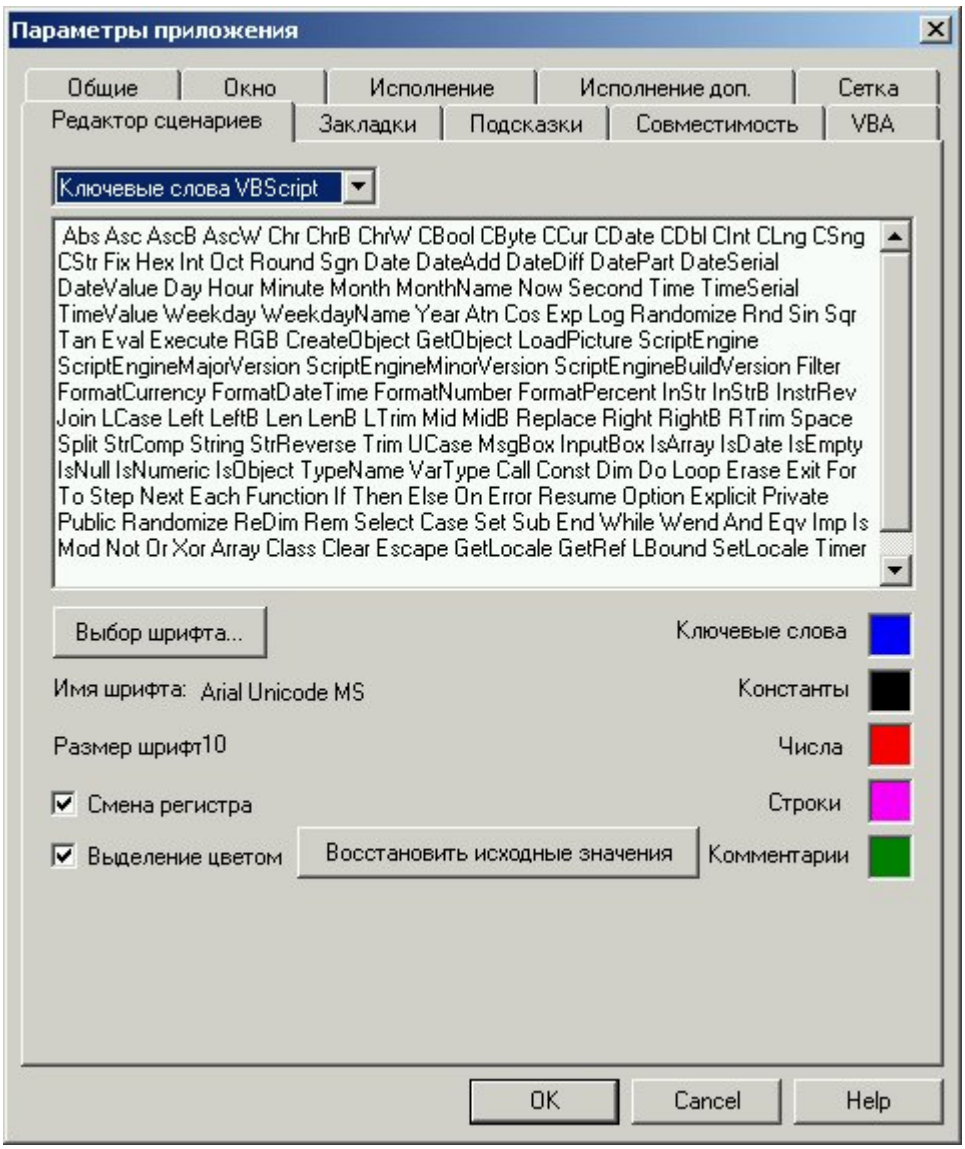


Рис. 9-29 Настройки Редактора сценариев

Глава 10. Среда исполнения.

10. 1 Введение

10.2 Активизация режима *Исполнение*

10.3 Меню Файл режима *Исполнение*

10.4 Меню Вид режима *Исполнение*

10.5 Меню Администрирование режима *Исполнение*

10.6 Команды меню окна Утилита регистрации в системе

10.7 Ввод данных в режиме *Исполнение*

10.8 Поддержка *TraceWorX32*

10.9 Совместимость с *OLExpress*

10.10 VCRWorX32 Data Replay

10.11 Установки Таймаута

10. 1 Введение

Настоящий раздел содержит описание *среды исполнения* GraphWorX32.
Среда исполнения предназначена для просмотра множества окон экранных форм в режиме *Исполнение*. В отличие от режима *Разработка*, в режиме *Исполнение* происходит периодическое обновление динамических соединений, примененных к графическим объектам экранных форм, в зависимости от значений соответствующих источников данных в серверах OPC. Имеется возможность одновременного просмотра любой комбинации окон приложений набора GENESIS32 фирмы ICONICS, включая TrendWorX32, AlarmWorX32 и др., а также других приложений Windows, например, Word, Excel и др.
Функциональность системы в режиме *Исполнение* зависит от настроек, выполненных для данного режима в процессе создания проекта.
В режиме *Исполнение* возможна динамическая загрузка других экранных форм, вывод на печать содержимого экранных форм, получение справочной информации о системе и перевод системы в режим *Разработка*.

10.2 Активизация режима *Исполнение*

Запуск из меню GraphWorX32
Для активизации режима *Исполнение* щелкните левой клавишей мыши на заголовке меню **Исполнение** главного окна GraphWorX32.
Ярлык на *Рабочем столе*

1. Щелкните правой клавишей мыши на ярлыке GraphWorX32 и выберите строку **Свойства** в появившемся контекстном меню.
2. Выберите закладку **Ярлык** диалоговой панели **Свойства: GraphWorX32**. В поле **Объект** после пути к исполняемому файлу Gwx32.exe введите следующую строку: "Экран.gdf" "–runtime"
   , где "Экран" – имя файла экранной формы, которая должна быть загружена в режиме *Исполнение*.
3. В поле **Рабочий каталог** укажите путь к каталогу, в котором находится файл экранной формы, введенный в поле **Объект**.

Пример создания ярлыка для автоматического запуска экранной формы VBARobot.gdf, которая находится в каталоге "C:\Program Files\ICONICS\GENESIS-32\Examples\GraphWorX32 Examples\", описан ниже.

Пример создания ярлыка запуска в режиме *Исполнение*:

1. На *Рабочем столе* щелкните правой клавишей мыши и в появившемся контекстном меню выполните команду **Создать–Ярлык**. На экран монитора будет выведена диалоговая панель **Создание ярлыка**, внешний вид которой показан на рис. 10- 1.

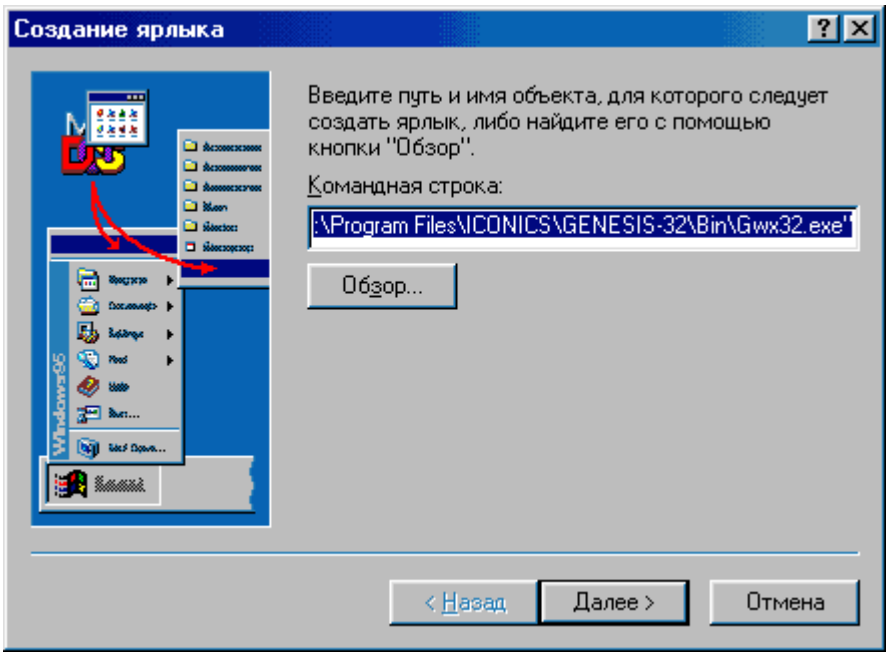


Рис. 10-1. Создание ярлыка для запуска экранной формы в режиме Исполнение

2. Щелкните на кнопке **Обзор** диалоговой панели и в появившейся диалоговой панели **Обзор** в каталоге *C:\Program Files\ICONICS\GENESIS-32\Bin* выберите имя исполняемого файла GraphWorX32 – *Gwx32.exe*, после чего нажмите кнопку **Далее**.
3. В появившейся диалоговой панели **Выбор названия программы** введите имя ярлыка, как показано на рис. 10-2, после чего нажмите кнопку **Готово**. Ярлык *VBARobot* появится на *Рабочем столе*.

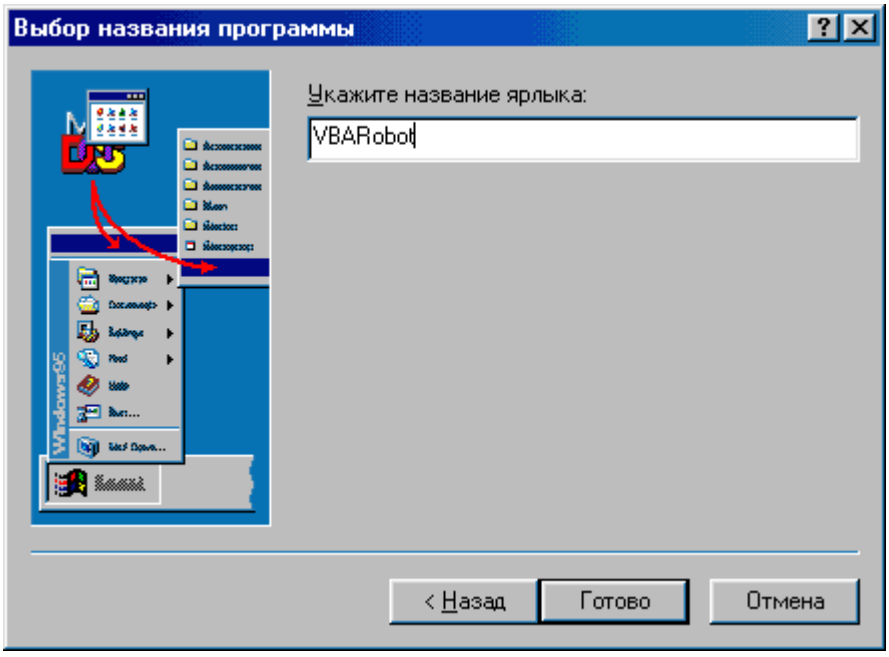


Рис. 10-2. Ввод имени ярлыка

4. Щелкните правой клавишей мыши на ярлыке *VBARobot* и выберите строку **Свойства** в появившемся контекстном меню. На экран монитора будет выведена диалоговая панель **Свойства: VBARobot**, показанная на рис. 10-3.

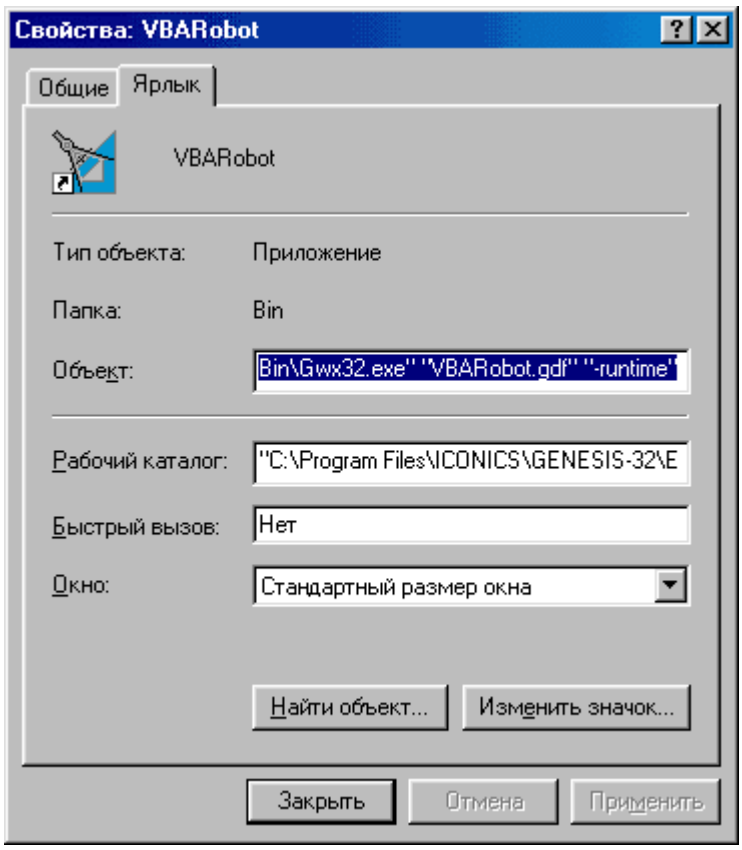


Рис. 10-3. Свойства ярлыка VBARobot

5. В поле **Объект** после строки, содержащей путь и имя исполняемого файла GraphWorX32, введите имя файла и ключ запуска в режиме *Исполнение* "VBARobot.gdf" "–runtime"
6. В поле **Рабочий каталог** введите путь к файлу VBARobot.gdf и нажмите на кнопку **Применить**.
7. Если при вводе не было допущено ошибок, кнопка **Применить** будет блокирована (надпись на кнопке отобразится серым цветом). Нажмите кнопку **Закрыть**.

Запуск в режиме *Исполнение* из командной строки:

1. Выберите команду **Выполнить** системного меню **Пуск**. В появившейся диалоговой панели **Запуск программы** введите путь и имя исполняемого файла GraphWorX32 – Gwx32.exe.
2. Введите путь и имя файла экранной формы, подлежащей запуску, а также ключ запуска.

Пример:

"C:\Program Files\ICONICS\GENESIS-32\Bin\Gwx32.exe" "C:\Program Files\ICONICS\GENESIS-32\Examples\GraphWorX32 Examples\VBARobot.gdf" "-runtime"

После перевода GraphWorX32 в режим *Исполнение*, произойдет изменение внешнего вида панели меню главного окна GraphWorX32, как показано на рис. 10-4.

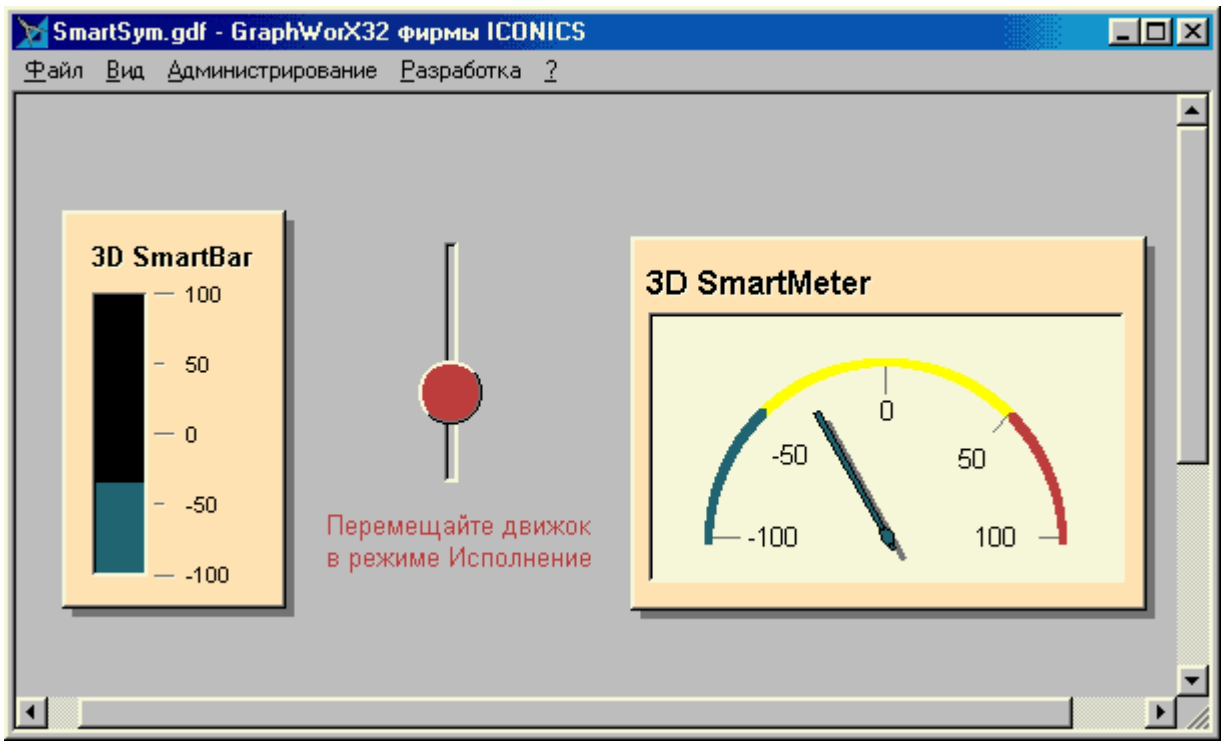


Рис. 10-4. Окно GraphWorX32 в режиме Исполнение

10.3 Меню Файл режима *Исполнение*

Внешний вид меню **Файл** GraphWorX32 в режиме *Исполнение* показан на рис. 10-5.

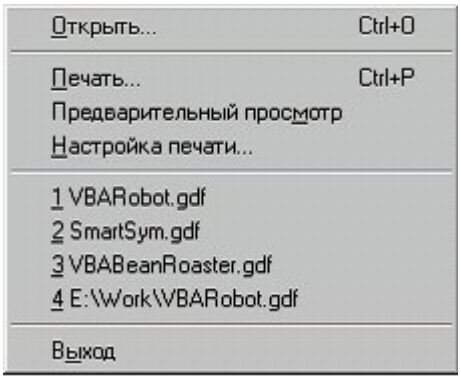


*Рис. 10-5. Меню **Файл** GraphWorX32 в режиме Исполнение*

Команда Открыть

Данная команда предназначена для загрузки экранной формы GraphWorX32 без остановки режима *Исполнение*.

Для загрузки экранной формы:

1. Выберите одно из окон GraphWorX32, если одновременно происходит исполнение нескольких экранных форм.
2. Выберите строку **Открыть** меню **Файл**.
3. В появившейся диалоговой панели **Открыть** выберите файл экранной формы GraphWorX32, который требуется загрузить, и нажмите кнопку **OK**. Текущая исполняющаяся экранная форма будет выгружена, а выбранная – загружена.

Печать экранных форм

В процессе исполнения экранной формы имеется возможность вывода содержимого всей экранной формы на печатающее устройство. Для выполнения указанной операции служит команда **Печать** меню **Файл**.

Для вывода экранной формы на печатающее устройство:

1. Выберите команду **Печать** меню **Файл**.
2. В появившейся диалоговой панели **Печать** введите требуемые значения параметров печати и нажмите кнопку **OK** или клавишу Enter.
3. Содержимое экранной формы будет выведено на печать.

Команда Предварительный просмотр

Данная команда позволяет поочередно отобразить на экране видеокадры в виде, в котором они будут выведены на печать.

Для выполнения предварительного просмотра:

1. Выберите строку **Предварительный просмотр** меню **Файл**. Окно экранной формы GraphWorX32 будет переведено в режим предварительного просмотра печати, внешний вид которого показан на рис. 10-6.

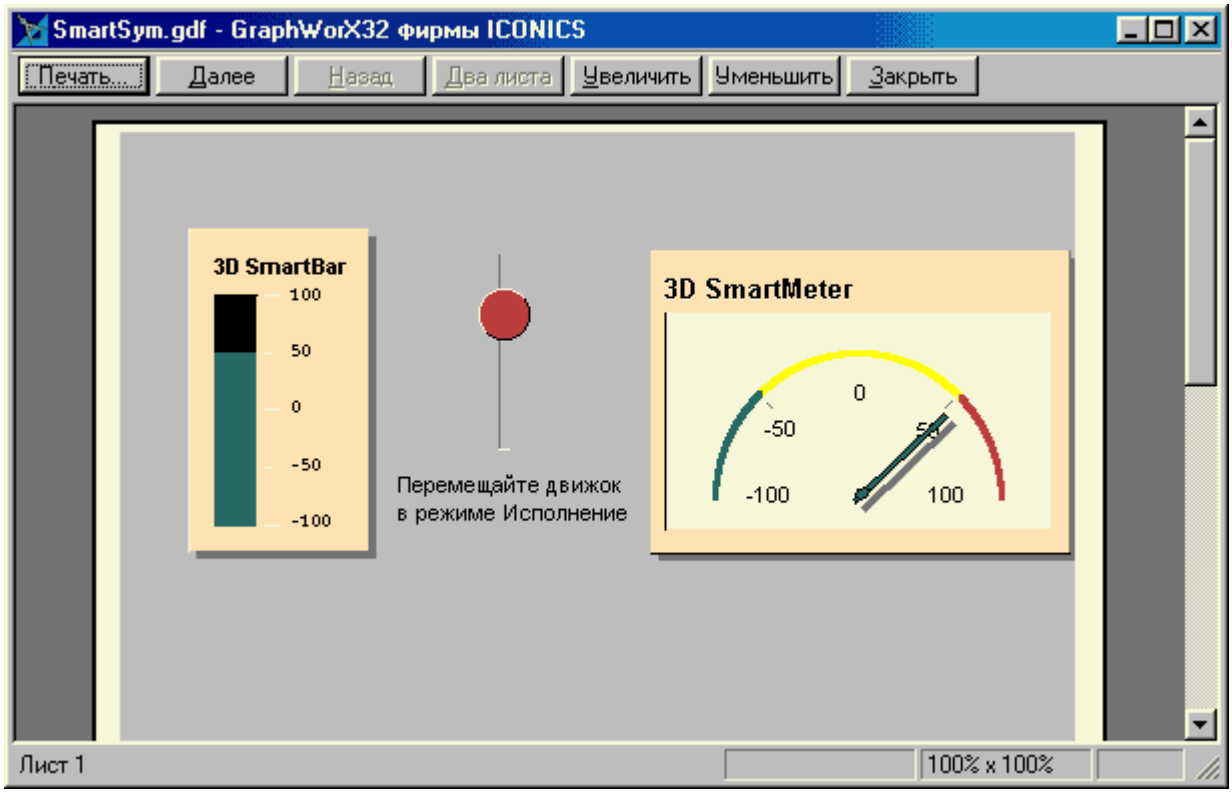


Рис. 10-6. Предварительный просмотр печати

2. Кнопка **Печать** позволяет начать печать текущего видеокадра. Кнопка **Далее** предназначена для отображения следующего по времени видеокадра.
3. Кнопки **Увеличить**, **Уменьшить** и **Закрыть** позволяют увеличить или уменьшить размеры отображаемого видеокадра, а также завершить предварительный просмотр.

Команда Настройка печати

Данная команда предназначена для настройки параметров печатающего устройства, включая выбор принтера, размер и ориентацию листа бумаги, а также лоток принтера, в который будут подаваться листы.

10.4 Меню Вид режима *Исполнение*

Меню **Вид** в режиме *Исполнение* показано на рис. 10-7. Как видно из рисунка, содержимое меню **Вид** в режиме *Исполнение* практически идентично используемому в режиме *Разработка*, за исключение команды **Статистика**.

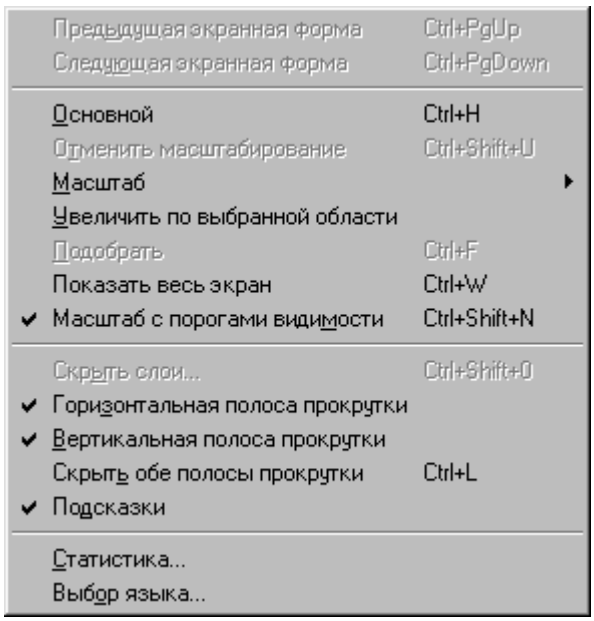


*Рис. 10-7. Меню **Вид** режима Исполнение*

Команда **Статистика** позволяет вывести на экран монитора информацию диалогового окна Statistics Viewer, показанного на рис. 10-8. Этот диалог реализован на библиотеке GenClient/Olexpress DLL и доступен через приложения GENESIS32. Для более подробной информации воспользуйтесь справочной документацией GenStatistics Viewer.

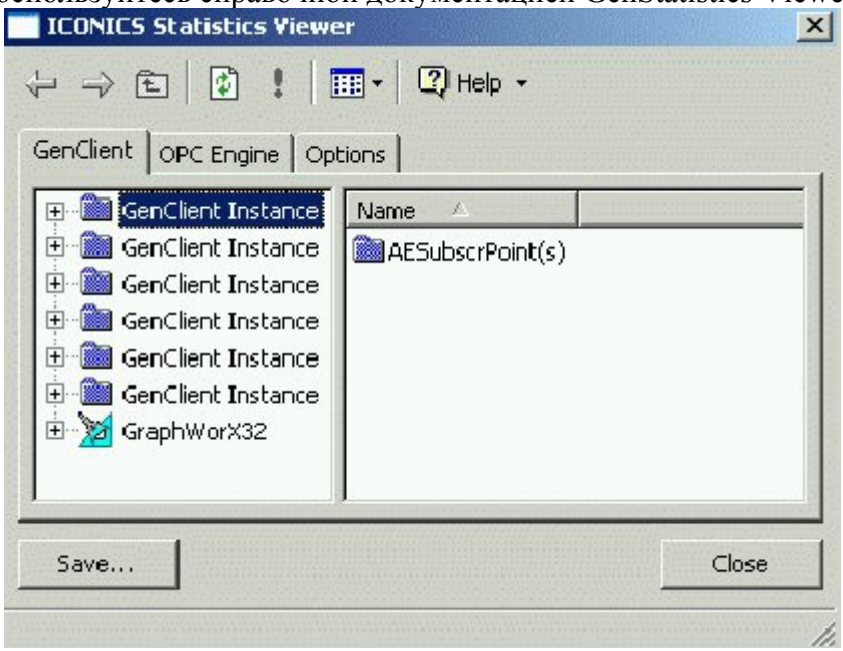


*Рис. 10-8. **GenStatistics Runtime Viewer***

Список ранее загруженных экранных форм

GraphWorX32 сохраняет список последних 50-ти загруженных экранных форм. Поочередная загрузка экранных форм из указанного списка производится с помощью операций *Предыдущая экранная форма* и *Следующая экранная форма* динамического соединения *Указание и щелчок*. Кроме того, выполнение данных операций возможно из сценарных процедур, написанных на языке VBA, или из внешних приложений Windows путем использования соответствующих методов OLE Automation.

Примечание. Сохраняемый список последних 50-ти экранных форм содержит исходные настройки псевдонимов для каждой предыдущей экранной формы. Таким образом, при переходах к предыдущим экранным формам настройки всех псевдонимов будут

сброшены в соответствующие начальные состояния. Для получения более подробной информации обратитесь к разделу 7 настоящего руководства.

10.5 Меню Администрирование режима *Исполнение*

Настоящий подраздел содержит сведения о регистрируемом входе пользователей в систему исполнения GraphWorX32 с контролем прав доступа к различным операциям и функциям исполняемого проекта.

Уровни доступа к системным функциям задаются на этапе создания проекта и устанавливают полноту прав персонала, эксплуатирующего и обслуживающего систему, по выполнению различных системных операций.

Для входа в систему исполнения с регистрацией прав пользователя:

1. Выберите команду **Вход в систему** меню **Сервис**. На экран монитора будет выведена диалоговая панель **Вход в систему**(рис. 10-9).

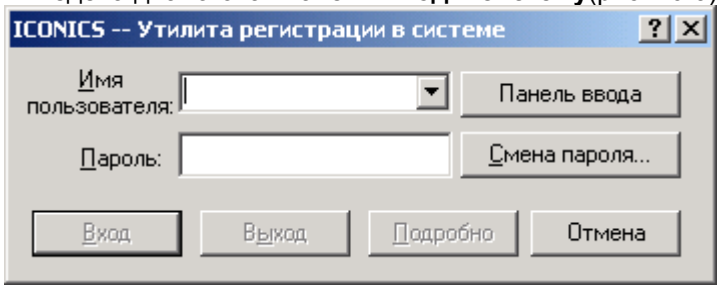


*Рис. 10-9. Диалоговая панель **Вход в систему***

2. Введите имя пользователя и пароль в соответствующие поля диалоговой панели и нажмите кнопку **Вход**. На экран будет выведено окно *Сервера администрирования*, показанное на рис. 10-10.

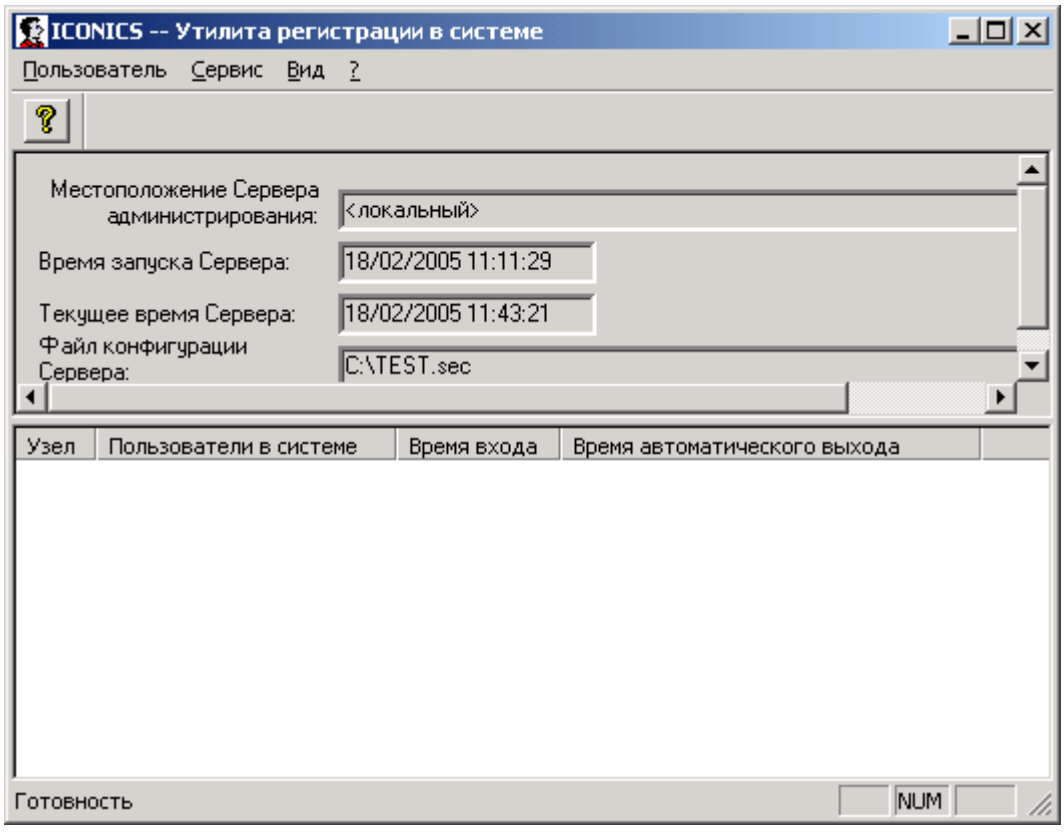


*Рис. 10-10. Окно **Утилита регистрации в системе***

Описание полей этого окна приведено в табл. 10-1.

Таблица 10-1

*Описание полей окна **Утилита регистрации в системе***

Поле	Описание
Местоположение Сервера администрирования	Содержит информацию о местонахождении *Сервера системного администрирования*.
Время запуска сервера	Содержит информацию о дате и времени запуска *Сервера системного администрирования*.
Текущее время сервера	Содержит текущие дату и время.
Файл конфигурации сервера	Содержит имя конфигурационного файла сервера системного администрирования.
Узел	Данный столбец содержит расположение зарегистрированных в системе пользователей.
Пользователи в системе	Данный столбец содержит список пользователей, зарегистрировавшихся в системе.
Время входа	Данный столбец содержит времена и даты регистрации пользователей в системе.
Время автоматического выхода	Данный столбец содержит времена и даты, при наступлении которых произойдет автоматический выход соответствующих пользователей из системы.

10.6 Команды меню окна Утилита регистрации в системе

Меню Пользователь

Данное меню содержит команды, показанные на рис. 10-11.

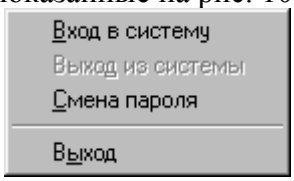


*Рис. 10-11. Команды меню **Пользователь***

Команда **Вход в систему** предназначена для регистрации пользователей на *Сервере администрирования* GENESIS32.

Команда **Смена пароля** позволяет пользователю сменить текущий пароль.

Для смены пароля:

1. Выберите строку **Смена пароля** меню **Пользователь**. На экран монитора будет выведена диалоговая панель **Смена пароля**, показанная на рис. 10-12.
2. Введите имя пользователя, текущий используемый пароль, новый пароль в соответствующие поля диалоговой панели и нажмите кнопку **OK**.

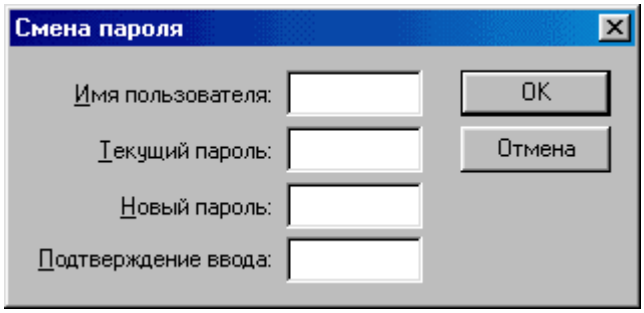


*Рис. 10-12. Диалоговая панель **Смена пароля***

Команда **Выход** предназначена для завершения работы с окном *Сервера системного администрирования*.

Меню Сервис

Данное меню содержит команды, показанные на рис. 10-13.

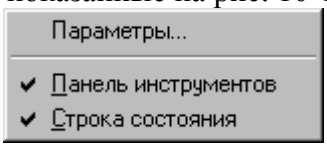


*Рис. 10-13. Меню **Сервис***

Команда **Параметры** предназначена для настройки части параметров *Сервера системного администрирования* в диалоговой панели, показанной на рис. 10-14.

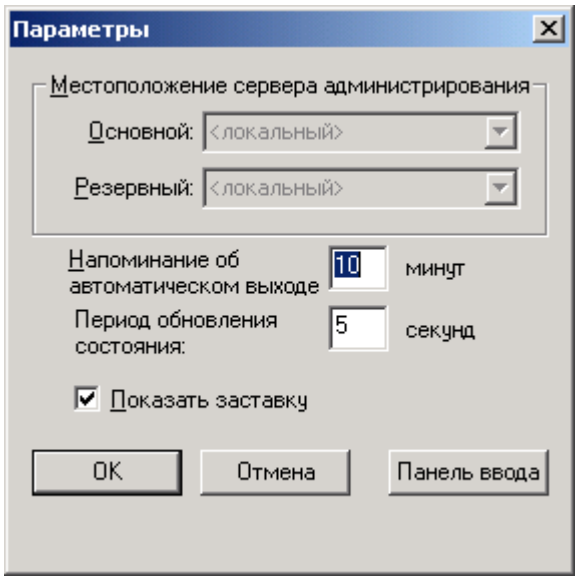


Рис. 10-14. Диалоговая панель настройки некоторых параметров Сервера системного администрирования в режиме Исполнение

Описание полей диалоговой панели **Параметры** приведено в табл. 10-2.

Таблица 10-2

*Описание полей диалоговой панели **Параметры***

Поле	Описание
Местоположение сервера администрирования	Введите имена основного и резервного узлов для соединения с *Сервером системного администрирования.* По умолчанию устанавливается узел <локальный>. Примечание: Формирование расширенного выпадающего списка узлов занимает много времени. Если Вы знаете точное имя рабочей станции, удобнее внести его вручную.
Напоминание об автоматического выходе, мин	Данное поле содержит и позволяет ввести время, по истечении которого будет происходить автоматическое оповещение пользователей, «вошедших в систему», об их последующем автоматическом выводе из числа зарегистрированных в системе. Чтобы на экран не выводилось всплывающее окно с этой информацией, введите «0».
Период обновления состояния	Содержит и позволяет установить период времени, выраженный в секундах, по истечении которого *Сервер системного администрирования* обновляет свое состояние, включая всю информацию, связанную с правами и настройками, которые заданы для пользователей.
Показать заставку	Показать/скрыть заставку при старте Security Login.

Команды **Строка состояния** и **Панель инструментов** предназначены для управления отображением панели инструментов и строки состояния окна **Вход в систему**.

Меню Справка

Данное меню содержит команды, позволяющие вывести список разделов справочной системы *Сервера системного администрирования*, а также информацию о сервере, номере версии и авторских правах в диалоговой панели **ICONICS About Box**.

10.7 Ввод данных в режиме *Исполнение*

Во время режима исполнения можно переключаться между элементами ввода данных с помощью клавиши **TAB**.

10.8 Поддержка *TraceWorX32*

Благодаря встроенной во все продукты ICONICS технологии, в системе GENESIS32 можно проводить диагностику с помощью утилиты TraceWorX32, а также настройку приложений. TraceWorX32 была создана специально для системных интеграторов, OEM пользователей, которым был нужен инструмент для проведения диагностики и устранения неисправностей.

TraceWorX32 отслеживает действия приложений GENESIS32 в режиме исполнения и записывает данные в файл отчета, в соответствии с уровнями слежения, определенными пользователем. В отчете находится полная информация о производимых действиях, включая время, дату, уровень доступа и описание события или проблемы.

TraceWorX32 также может формировать отчеты для службы технической поддержки. Есть возможность сжатия файлов отчетов и пересылки их по электронной почте.

Активизация TraceWorX32 производится с помощью установки следующего ключа системного реестра для соответствующего компонента GENESIS32:

параметр **ReleaseTraceLevel** должен иметь значение от 0 (трассировка всей информации) до 1000 (функция отключена). Если функция включена, отчет будет располагаться как в текущем каталоге компонента, так и в папке **Documents and Settings**, и иметь имя ***.log.xml**

[HKEY_LOCAL_MACHINE\SOFTWARE\ICONICS\GWX32]

"ReleaseTraceLevel"=dword:00000000

[HKEY_LOCAL_MACHINE\SOFTWARE\ICONICS\GWXview32]

"ReleaseTraceLevel"=dword:00000000

Дополнительная информация содержится в документации на TraceWorX32.

10.9 Совместимость с *OLExpress*

GraphWorX32 может имитировать действия OLExpress 6.0 для обеспечения совместимости. Патч исправляет проблемы с передвижением ползунка, но представляет потенциальную проблему – нарушение целостности данных. Функция по умолчанию отключена.

Она может быть отключена для GWX32.exe следующим ключом системного реестра:

[HKEY_LOCAL_MACHINE\SOFTWARE\ICONICS\GWX32]
"CompatibilityOlexpress60Write"=dword:00000000

и для GWXview32.ocx следующим ключом системного реестра:

[HKEY_LOCAL_MACHINE\SOFTWARE\ICONICS\GWXview32]
"CompatibilityOlexpress60Write"=dword:00000000

Функция немедленно обновляет другие динамические объекты экранной формы, поэтому Вы можете разместить элемент Параметр после Ползунка и просматривать точное значение, которое было введено.

10.10 VCRWorX32 Data Replay

VCRWorX32 – мощный инструмент GENESIS32, позволяющий воспроизводить графические, исторические данные и тревоги. Основные свойства VCRWorX32:
Панель управления, показанная на рис. 10-15, имеет пользовательский интерфейс, позволяющий устанавливать время воспроизведения исторических данных, а также дату.
Возможности VCRWorX32:
Увидеть то, что видел оператор на графических дисплеях
Определить временные границы и дату для воспроизведения
Быстрая перемотка вперед и назад
Изменения скорости воспроизведения
Поиск по ключевым событиям, тегам или выражениям
Изменение оболочек, шаблонов

Запуск управляющей панели VCRWorX из режима Исполнение:

1. Выбрать пункт Запуск VCRWorX из меню Сервис
2. Экран GraphWorX перейдет в режим воспроизведения архивных данных по установленному пользователем дате и времени. Есть возможность увидеть значения параметров и управляющие элементы ActiveX, обновленные в экране GraphWorX, а также поля время/дата панели VCRWorX в режиме воспроизведения, промотки назад.



Рис. 10-15. Панель управления VCRWorX

Для более подробной информации о VCRWorX, воспользуйтесь справочной документацией VCRWorX.

10.11 Установки Таймаута

GraphWorX позволяет устанавливать внутренние таймауты для нескольких операций системного реестра:
[HKEY_LOCAL_MACHINE\SOFTWARE\ICONICS\GWX32\Runtime Settings]
"SecurityTimeoutSecs"=dword:0000003c
"LaunchAppTimeoutSecs"=dword:0000003c
"SystemWideMsgTimeout"=dword:0000003c
По умолчанию, величина приведенных выше таймаутов равна 60 секунд.
SecurityTimeoutSecs –определяет таймаут получения настроек безопасности
LaunchAppTimeoutSecs – определяет таймаут запуска внешних приложений
SystemWideMsgTimeout – таймаут пересылки системных сообщений

Глава 11. Мастера VBA.

11.1 Введение

11.2 Принцип работы Мастера VBA

11.3 Соглашения по именам сценариев, относящиеся к Мастеру VBA

11.4 Средство реализации Мастера VBA

11.5 Описание полей диалоговой панели Мастер VBA-сценариев

11.6 Более подробная информация о *Мастерах VBA*

11.7 Другие источники информации

11.1 Введение

Мастера VBA являются объектами GraphWorX32 со связанным с ними кодом VBA. Данный код исполняется как в режиме *Разработка*, что обеспечивает дополнительную гибкость и функциональность системы при конфигурировании объектов GraphWorX32, так и в режиме *Исполнение* – для выполнения специальных задач, связанных с объектом. Код VBA, как правило, берется из редактора VBA и сохраняется в файле текущей экранной формы (текущем документе), когда выполняется операция ее сохранения. При открытии экранной формы GraphWorX32 происходит автоматическая загрузка связанного с ней кода VBA в редактор Visual Basic.
Однако если следовать приведенным ниже указаниям, код VBA может быть привязан к отдельным объектам GraphWorX32. При вставке или переносе таких объектов в окно другого запущенного экземпляра GraphWorX32, в *Библиотеку символов* или на *Рабочий стол*, связанный код VBA "последует" за объектом.

11.2 Принцип работы Мастера VBA

Режим *Разработка*

GraphWorX32 может активизировать (запустить) макрос Мастера VBA, находясь в режиме *Разработка*, при двойном щелчке левой клавишей мыши на объекте или символе, с которым связан соответствующий макрос.
По умолчанию двойной щелчок левой клавишей мыши на объекте или символе GraphWorX32 приводит к вызову **Инспектора свойств** соответствующего объекта. Однако если в поле **Специальные данные Инспектора свойств** объекта или символа в начале строки указано специальное ключевое слово, будет осуществлена попытка запуска соответствующего макроса.

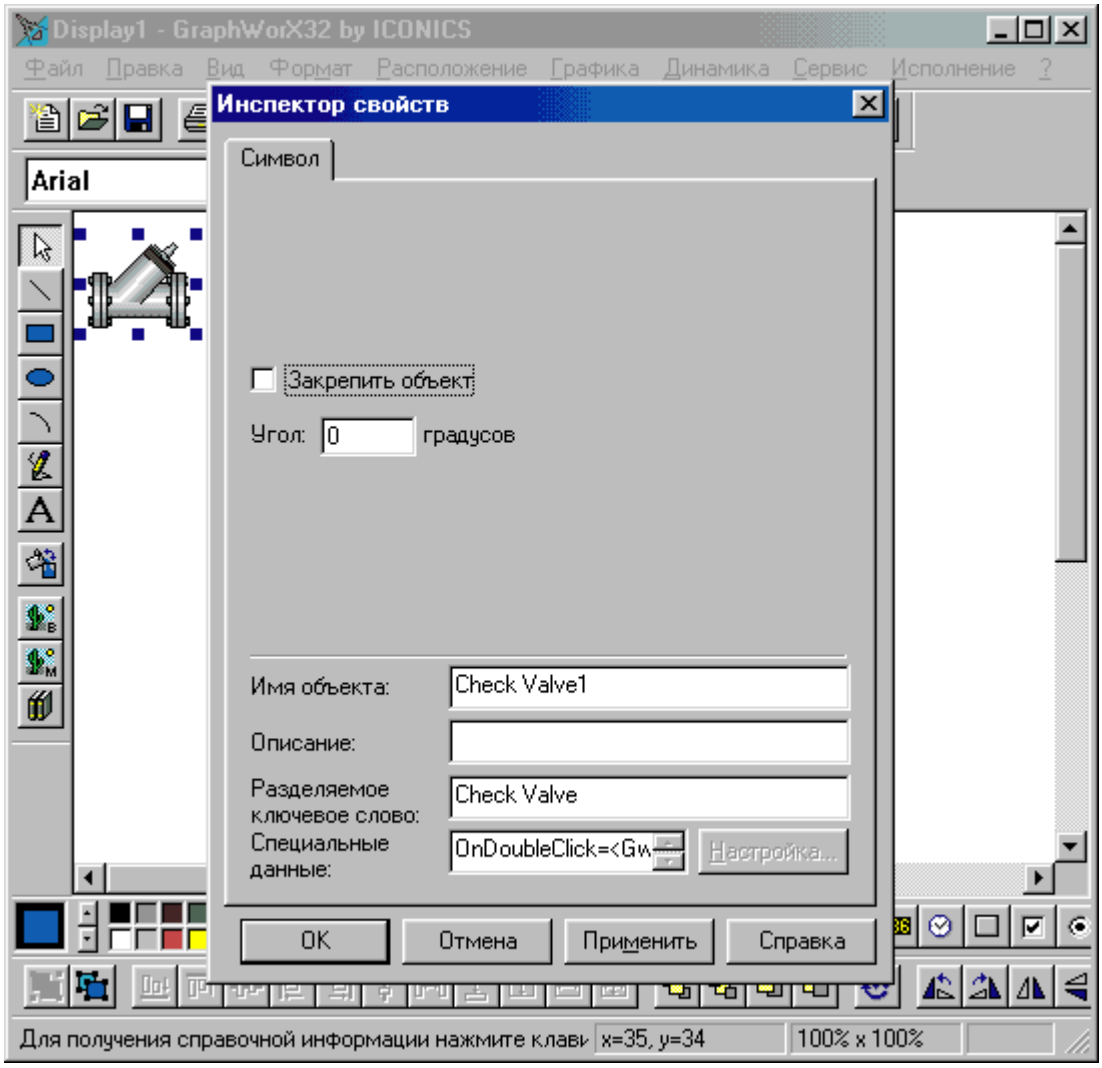


Рис. 11-1. Мастер VBA: Запись в поле Специальные данные

Формат ключевого слова, позволяющего вызывать макрос с именем *ИмяМакроса*, имеет следующий вид:

OnDoubleClick=<GwX*ИмяМакроса*_Main.*ИмяМакроса*>,Parameters=<>

Для успешного запуска макроса при двойном щелчке левой клавишей мыши на объекте в составе проекта должен существовать макрос *ИмяМакроса* в модуле *GwXИмяМакроса_Main*. Секция **Parameters (Параметры)** может содержать любые данные, вводимые пользователем между угловыми скобками <>. Любые данные, которые по требованию пользователя подлежат передаче в макрос для дополнительной обработки, могут быть размещены между угловыми скобками.

Примечание. Имя макроса не должно содержать пробелы!

Режим *Исполнение*

Запуск *Мастера VBA* в режиме *Исполнение* осуществляется щелчком на объекте *Кнопка* или любом другом графическом объекте, к которому применено действие *Указание и щелчок* с выбранной операцией *Выполнить сценарий VBA*. При этом поле **Имя сценария** страницы **Указание и щелчок** диалоговой панели **Инспектор свойств** должно содержать имя макроса в следующем формате:

Gwx*ИмяМакроса*_Main.*ИмяМакроса*

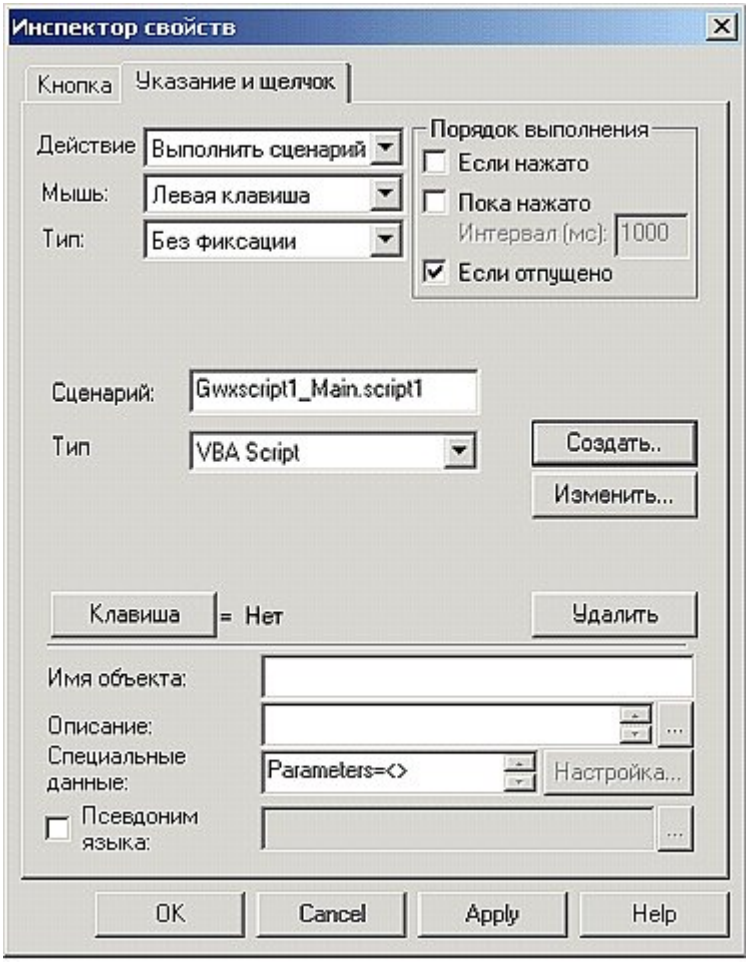


*Рис. 11-2. Назначение вызова **Мастера VBA** на кнопку*

Поле **Специальные данные** может содержать любую строку требуемых данных для передачи в сценарий. Строка помещается между угловыми скобками <>.

11.3 Соглашения по именам сценариев, относящиеся к Мастеру VBA

GraphWorX32 осуществляет полный контроль над содержимым кода, создаваемого при помощи *Мастера VBA*. Благодаря специальным соглашениям по присвоению имен всем модулям, относящимся к конкретному *Мастеру VBA*, код в данных модулях может согласованно перемещаться вместе с символом, к которому он привязывается в момент создания.
Если макрос *Мастера VBA* имеет имя *ИмяМакроса*, то все программные модули должны начинаться со строки Gwx*ИмяМакроса*_. Данная технология обеспечивает возможность использования большего количества программных модулей и форм совместно с одним объектом *Мастера VBA*, а также при необходимости облегчает перенос всего указанного кода вместе с объектом.

11.4 Средство реализации Мастера VBA

GraphWorX32 имеет в своем составе удобное средство создания макросов – диалоговую панель **Мастер VBA-сценариев**, с помощью которой возможно преобразовать любой объект GraphWorX32 в *Мастер VBA* и создать шаблон программного кода VBA. Созданный шаблон кода может быть легко расширен и модифицирован.
Ниже приведен пример использования указанного средства. Запустите GraphWorX32, нарисуйте несколько эллипсов и сгруппируйте их в символ. Выполните щелчок правой

клавишей мыши на полученном символе и выберите строку **Создать VBA-сценарий** в появившемся контекстном меню.

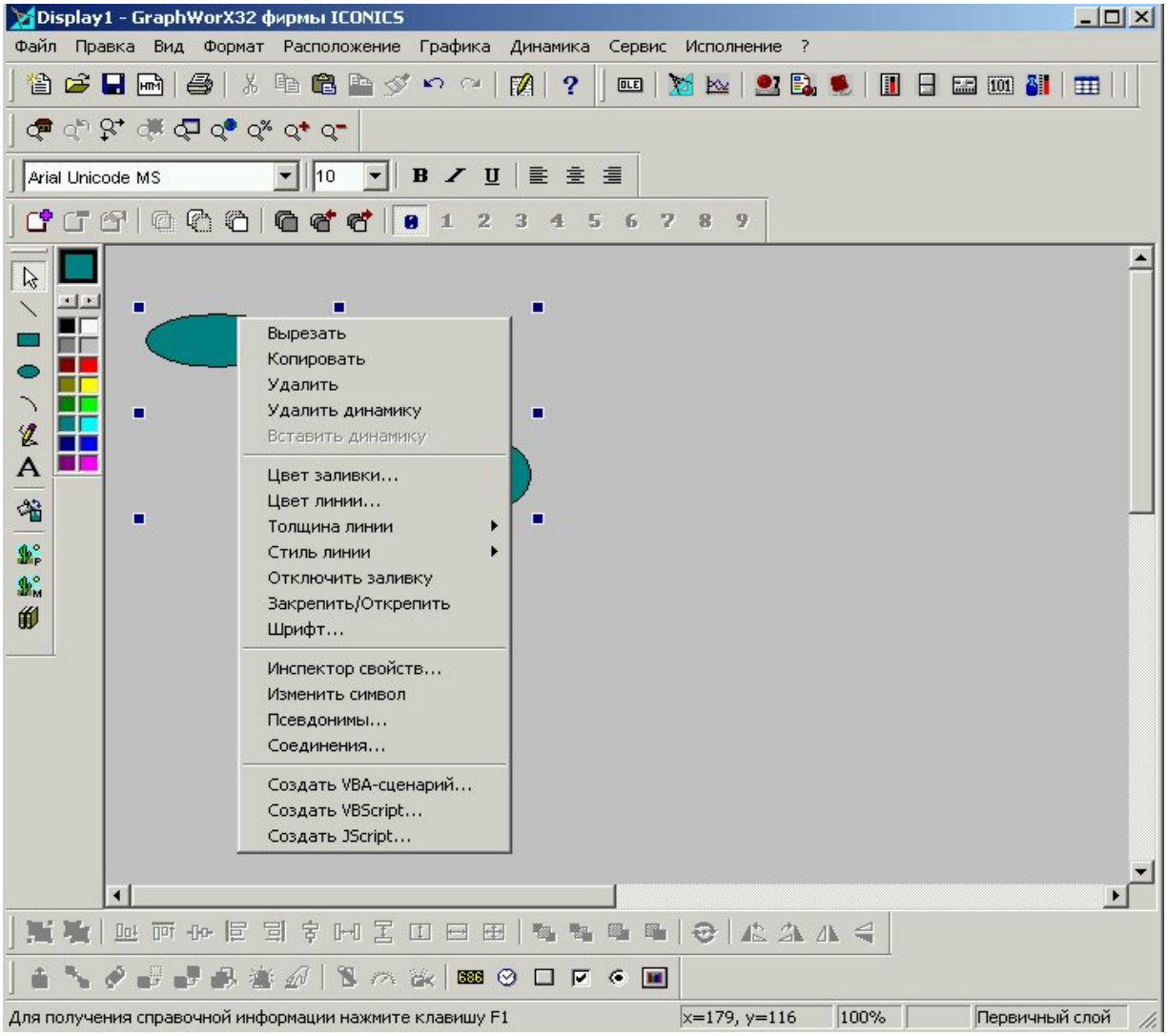


*Рис. 11-3. Вызов диалоговой панели **Мастер VBA-сценариев** с помощью контекстного меню*

Другим способом вызова указанной диалоговой панели является выделение объекта с последующим выбором строки **Создать** подменю **Макрос** меню **Сервис** GraphWorX32.

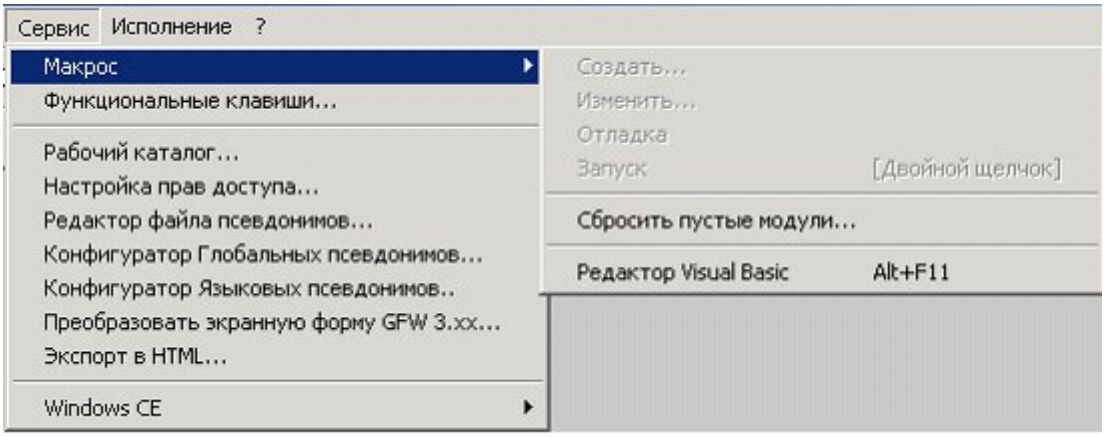


*Рис. 11-4. Вызов диалоговой панели **Мастер VBA-сценариев** с помощью меню GraphWorX32*

На экран монитора будет выведена диалоговая панель **Мастер VBA-сценариев**, показанная на рис. 11-5. Введите имя макроса в поле **Имя сценария**, например, "*Test*".

Обратите внимание, что в поле **Модуль**, недоступном для ввода, автоматически появится имя модуля, основанное на введенном имени макроса.

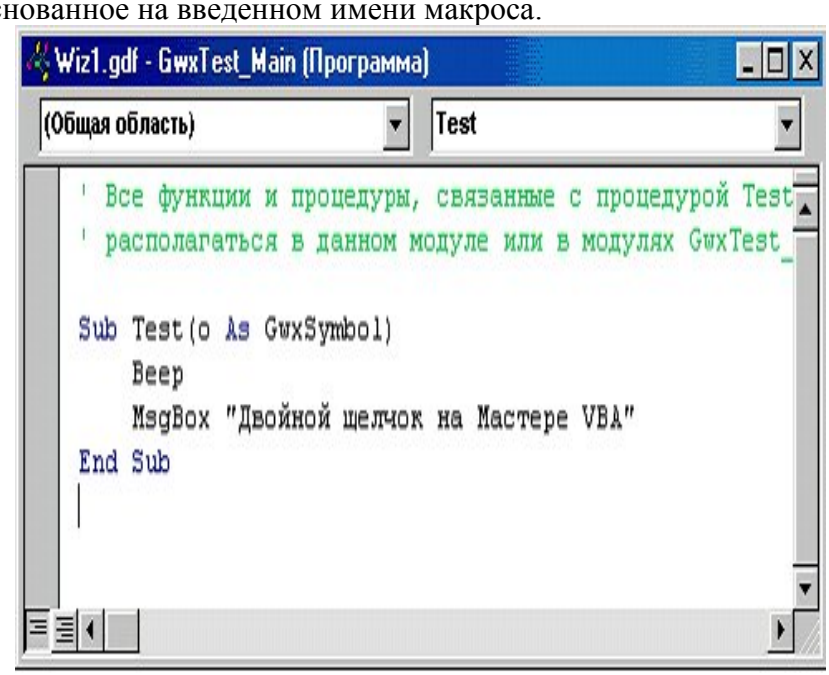


*Рис. 11-5. Диалоговая панель **Мастер VBA-сценариев***

Нажмите кнопку **OK** диалоговой панели. Произойдет запуск редактора VBA и курсор будет помещен в тело процедуры Test в программном модуле *GwxTest_Main*. В настоящий момент возможен ввод исходного текста сценария, который будет выполнен при двойном щелчке левой клавишей мыши на созданном символе GraphWorX32 в режиме *Разработка*. Введите программный код процедуры Test, показанный на рис. 11-6.

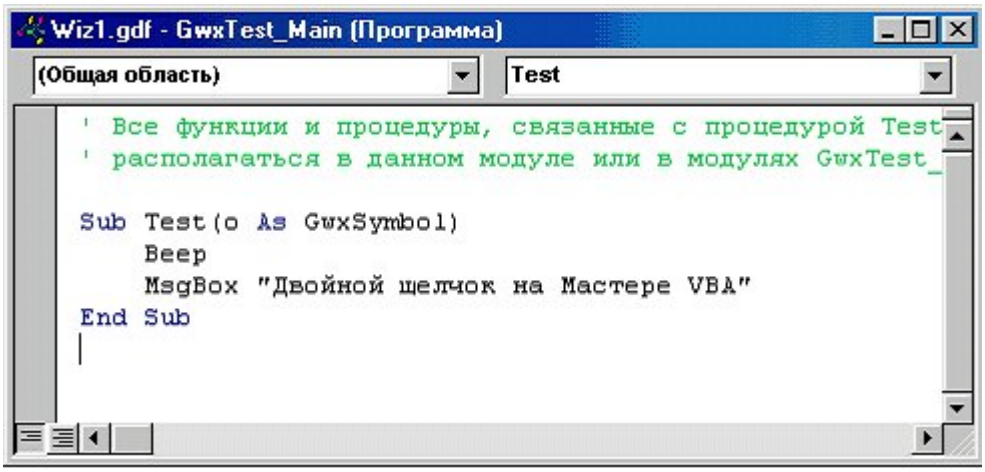
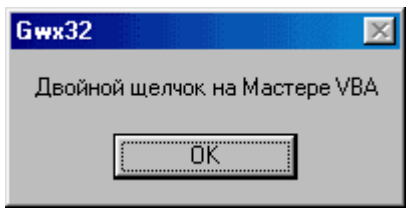


*Рис. 11-6. Процедура Test, созданная **Мастером VBA***

Вернитесь в окно экранной формы GraphWorX32 и дважды щелкните левой клавишей мыши на символе. Раздастся короткий звуковой сигнал и на экран будет выведено сообщение:



11.5 Описание полей диалоговой панели Мастер VBA-сценариев

Поле Имя сценария

Данное поле предназначено для ввода имени создаваемого макроса. Имя должно начинаться с буквы, а остальные символы должны быть алфавитно-цифровыми. Использование одинаковых имен нескольких макросов в одном модуле не допускается.

Поле Модуль

Данное поле недоступно для ввода и содержит автоматически формируемое на основе введенного имени макроса имя программного модуля.

Флажок Добавить в сценарий код для разбора параметров

Если данный флажок отмечен (установлен), то в исходный текст тела процедуры будет включен некоторый дополнительный программный код, который облегчает задачи сохранения и восстановления данных, связанных с объектом *Мастера VBA*. Указанный программный код использует модуль *GwxTools* для преобразования параметров, задаваемых пользователем, в строковую переменную с именем *StrPar*, которая является локальной в создаваемом сценарии.

Поле Параметры

Данное поле может содержать любую строку, вводимую пользователем на этапе создания макроса. Данная строка будет передана в макрос при его запуске. Указанное поле разработано для обеспечения передачи специальных пользовательских данных каждому экземпляру объекта *Мастера VBA*. Различные экземпляры одних и тех же объектов могут иметь разные передаваемые данные. Использование данного поля необязательно.

Флажок Добавить форму

Если при вызове создаваемого макроса на экран должна выводиться форма VBA, отметьте (установите) указанный флажок. Имеется возможность включения в *Мастер VBA* любого количества форм. При этом предполагается, что пользователь придерживается соглашений по присвоению имен, принятых в *Мастере VBA*. В противном случае перемещение кода VBA вместе со связанным символом или графическим объектом станет невозможным.

Поле Форма

Данное поле недоступно для ввода и содержит автоматически формируемое на основе введенного имени макроса имя основной формы, связанной с макросом.

11.6 Более подробная информация о *Мастерах VBA*

Режим разработки *Мастера VBA*

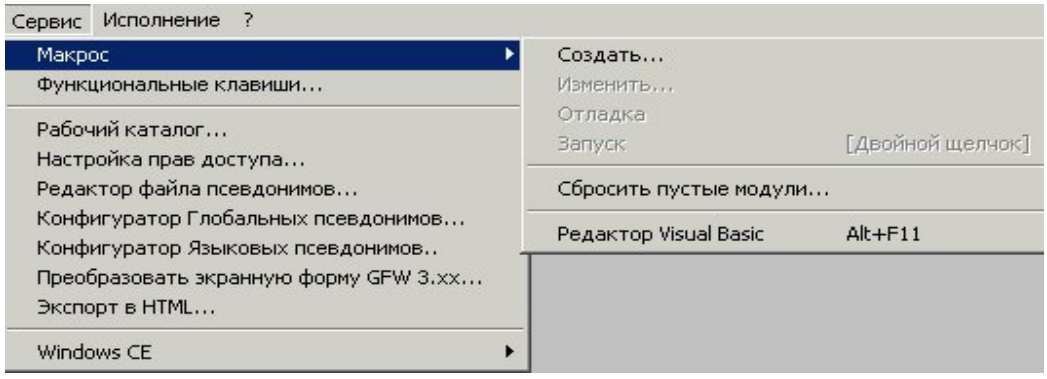


*Рис. 11-7. Меню **Мастера VBA***

Подменю **Макрос** меню **Сервис** GraphWorX32 содержит ряд команд, предназначенных для работы с *Мастерами VBA* в режиме разработки. Команды показаны на рис. 11-7 справа.

Макрос – Создать

Данная команда предназначена для создания макроса *Мастера VBA*. Аналогичная операция может быть выполнена путем щелчка правой клавишей мыши на символе или объекте, с которым связывается создаваемый макрос, с последующим выбором строки **Создать VBA-сценарий** в появившемся контекстном меню. Команды **Создать VBA-сценарий** контекстного меню и команда **Создать** подменю **Макрос** GraphWorX32 доступны только в том случае, когда в пределах рабочей области экранной формы выделен графический объект или символ, с которым ранее не был связан другой макрос *Мастера VBA*.

Изменить

При выполнении данной команды происходит вызов редактора VBA с помещением текстового курсора в тело процедуры, представляющей макрос. Аналогичная операция может быть выполнена путем щелчка правой клавишей мыши на символе или объекте, с которым ранее был связан макрос *Мастера VBA*, с последующим выбором строки **Изменить VBA-сценарий** в появившемся контекстном меню. Команды **Изменить VBA-сценарий** контекстного меню и команда **Изменить** подменю **Макрос** GraphWorX32 доступны только в том случае, когда в пределах рабочей области экранной формы выделен графический объект или символ, с которым ранее был связан макрос *Мастера VBA*.

Отладка

Данная команда предназначена для запуска макроса *Мастера VBA* выбранного графического объекта или символа в пошаговом (отладочном) режиме.

Запуск

Данная команда предназначена для запуска макроса *Мастера VBA* выбранного графического объекта или символа. Аналогичная операция может быть выполнена путем двойного щелчка левой клавишей мыши на символе или объекте, с которым ранее был связан макрос *Мастера VBA*. Команда доступна, когда в пределах рабочей области экранной формы выделен графический объект или символ, с которым ранее был связан макрос *Мастера VBA*.

Сбросить пустые модули

Если объект или символ, с которым связан макрос *Мастера VBA*, удален или перенесен из рабочей области экранной формы, соответствующий программный код не удаляется из текущего проекта автоматически. Данная функция позволяет удалить все неиспользуемые модули, которые были связаны с объектами или символами, в текущий момент отсутствующими в рабочей области экранной формы.

Режим исполнения *Мастера VBA*

При необходимости создания сценария, вызов которого должен происходить в режиме *Исполнение*, средствами *Мастера VBA*, создайте кнопку или объект с динамическим соединением *Указание и щелчок* и в выпадающем списке **Действие** выберите строку *Выполнить сценарий VBA*, после чего с помощью кнопок **Создать** и **Изменить** страницы **Указание и щелчок** диалоговой панели **Инспектор свойств** вызовите *Мастер VBA*.
Примечание. Для редактирования имеющегося сценария достаточно вызвать редактор Visual Basic нажатием комбинации клавиш Alt+F11.

11.7 Другие источники информации

После установки GENESIS32 пользователь сможет найти множество примеров графических объектов и символов, включая находящиеся в *Библиотеке символов*, в которых применяется *Мастер VBA*.

Глава 12. GraphWorX32 ActiveX

12.1 Введение

12.2 Вставка *GraphWorX32 ActiveX* в приложение-контейнер

12.3 Панель инструментов *GraphWorX32 ActiveX*

12.4 Настройка параметров *GraphWorX32 ActiveX*

12.1 Введение

GraphWorX32 ActiveX (GWXView32.OCX) является управляющим элементом ActiveX, который предназначен для просмотра экранных форм GraphWorX32 в различных приложениях Windows, поддерживающих технологию ActiveX. Основным достоинством технологии ActiveX является возможность вставки и использования управляющего элемента в любом приложении, которое является контейнером ActiveX, в результате чего экранные формы GraphWorX32 могут размещаться на Web-страницах и просматриваться с помощью Microsoft Internet Explorer. Кроме того, данная технология дает возможность использовать GraphWorX32 в качестве компонента программной системы верхнего уровня АСУТП, разрабатываемой пользователем в средах Microsoft Visual Basic, C++, Inprise C++Builder, Microsoft Visual J++ 6.0 и др.
По существу *GraphWorX32 ActiveX* является компонентом, который поддерживает только просмотр экранных форм, разработанных при помощи GraphWorX32, в режиме *Исполнение* с минимальными возможностями конфигурирования. В результате существенным образом снижаются системные требования к аппаратным и программным ресурсам. Единственным ограничением *GraphWorX32 ActiveX* в режиме *Исполнение* является отсутствие поддержки исполнения VBA-сценариев, встроенных в экранные формы.

12.2 Вставка *GraphWorX32 ActiveX* в приложение-контейнер

Методика вставки управляющих элементов ActiveX в контейнеры может иметь незначительные отличия в разных приложениях-контейнерах, однако основы остаются неизменными. В настоящем разделе приведены указания по вставке *GraphWorX32 ActiveX* в экранную форму GraphWorX32.
Выберите строку **Вставить новый объект** меню **Правка** GraphWorX32 или щелкните мышью на кнопке  . На экран монитора будет выведена диалоговая панель **Вставка объекта**, показанная на
рис. 12-1.

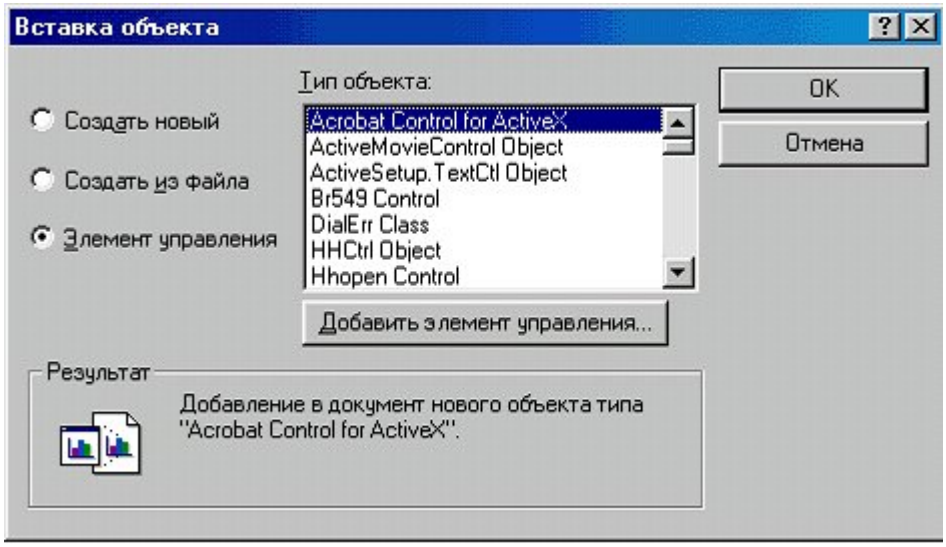


Рис. 12-1. Диалоговая панель ***Вставка объекта***

В списке **Тип объекта** диалоговой панели **Вставка объекта** выберите строку *ICONICS GraphWorX32 Viewer ActiveX* и нажмите кнопку **OK**. Управляющий элемент просмотра экранных форм GraphWorX32 будет вставлен в центр рабочей области экранной формы.

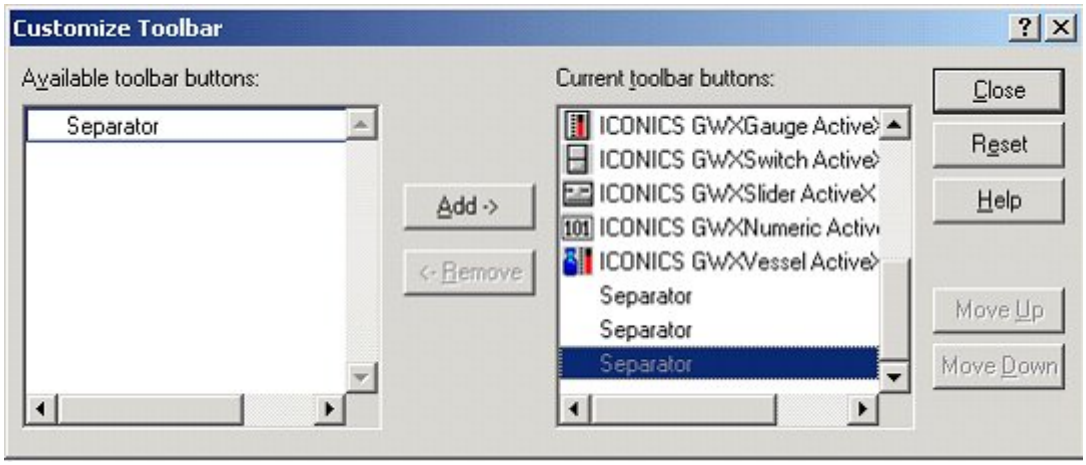
Для вставки указанного управляющего элемента можно воспользоваться кнопкой  , находящейся в панели инструментов *ActiveX* главного окна GraphWorX32.

12.3 Панель инструментов *GraphWorX32 ActiveX*

Панель инструментов *GraphWorX32 ActiveX* является частью экрана GraphWorX32:



Настройка панели инструментов.
Каждый *ActiveX,* вставляемый в экранную форму с помощью кнопки OLE, добавляется на панель инструментов как новая кнопка. Удалить инструменты можно с помощью панели Настройки инструментов, которая открывается при щелчке правой клавиши на Панели инструментов ActiveX:



Кнопки «Добавить» и «Удалить» переносят ActiveX из списка доступных в список текущих и обратно.
Следующие параметры системного реестра определяют расположение ActiveX на панели инструментов:

Количество ActiveX-компонентов на панели инструментов:
HKEY_LOCAL_MACHINE\Software\Iconics\Gwx32\OEM Information\ActiveX Toolbar Default Info\Number of ProgIDs

Программные идентификаторы (Prog ID) ActiveX-компонентов, расположенных на панели инструментов (например, "GWXVIEW32.GWXview32Ctrl.1").
HKEY_LOCAL_MACHINE\Software\Iconics\Gwx32\OEM Information\ActiveX Toolbar Default Info\ProgID_1
HKEY_LOCAL_MACHINE\Software\Iconics\Gwx32\OEM Information\ActiveX Toolbar Default Info\ProgID_2
HKEY_LOCAL_MACHINE\Software\Iconics\Gwx32\OEM Information\ActiveX Toolbar Default Info\ProgID_3
...etc.
Примечание: Эти параметры реестра не выполняют никаких действий, если GraphWorX32 уже запущен (т.к. при сохранении эти настройки будут перезаписаны текущими настройками GraphWorX32). Если GraphWorX32 уже запущен, необходимо удалить следующий параметр реестра:
HKEY_CURRENT_USER\Software\Iconics\Gwx32\Custom Toolbar Settings.

OLE ActiveX

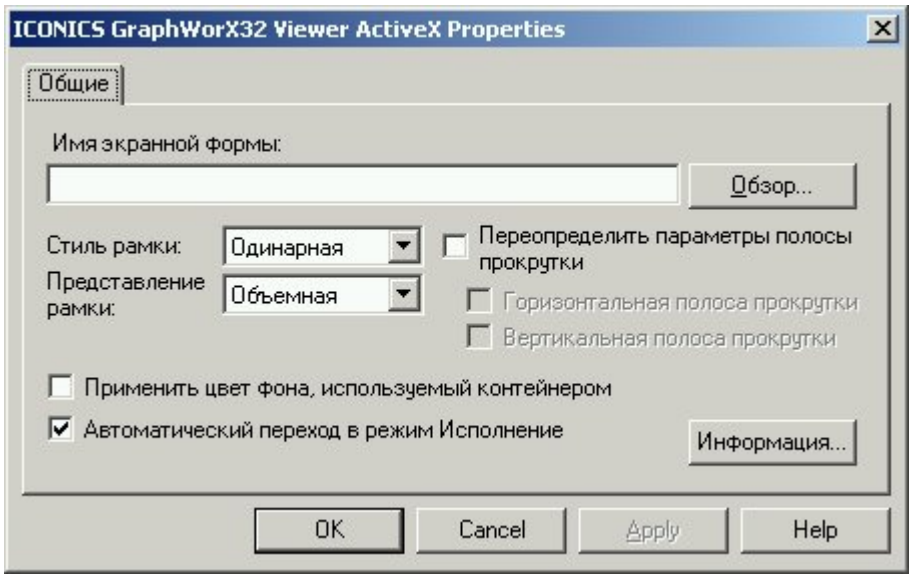


Щелчок на кнопке **OLE** на панели инструментов **ActiveX** вставляет ActiveX-компонент/объект OLE.

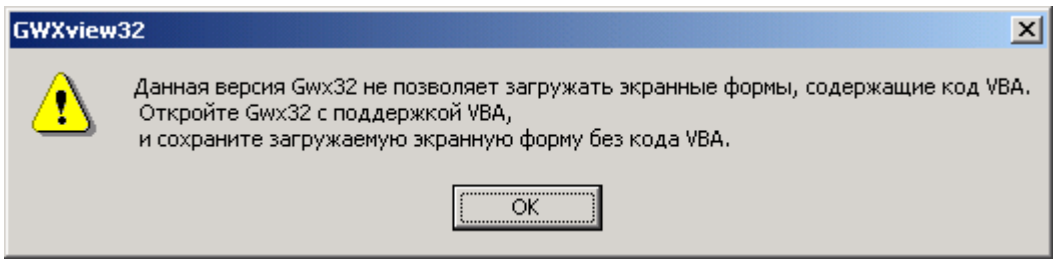
Graphics ActiveX



Щелчок на кнопке **GWXView32 ActiveX** на панели инструментов **ActiveX** открывает Graphics ActiveX.



Диалоговое окно, показанное выше, позволяет настроить параметры вставляемого в экранную форму ActiveX-компонента GraphworX32 Display. По умолчанию экранная форма сохраняется с кодом VBA. Но вставляемая экранная форма должна быть сохранена без VBA, иначе возникнет следующее сообщение об ошибке:



Trend ActiveX



Щелчок на кнопке **TWXView32** на панели инструментов **ActiveX** открывает TrendWorX32 Viewer ActiveX. Дополнительная информация содержится в документации на TrendWorX32 Viewer ActiveX.

Alarm ActiveX



Щелчок на кнопке **AWXView32** на панели инструментов **ActiveX** открывает AlarmWorX32 Container. Дополнительная информация содержится в документации на **AlarmWorX32 Viewer ActiveX**.

AlarmWorX32 Report ActiveX



Щелчок на кнопке AlarmWorX32 Report ActiveX на панели инструментов **ActiveX** открывает диалоговую панель **Свойства AlarmWorX32 Report ActiveX** . Дополнительная информация содержится в документации на AlarmWorX32 Report ActiveX.

AlarmWorX32 Indicator ActiveX

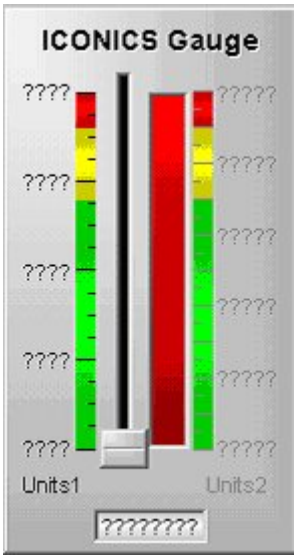


Щелчок на кнопке AlarmWorX32 Indicator ActiveX на панели инструментов **ActiveX** выводит на экран Индикатор тревог. Дополнительная информация содержится в документации на AlarmWorX32 Indicator ActiveX.

Gauge ActiveX



Щелчок на кнопке **GWXGauge ActiveX** на панели инструментов **ActiveX** открывает Gauge ActiveX. Дополнительная информация содержится в документации на **GWXGauge ActiveX.**



Switch ActiveX



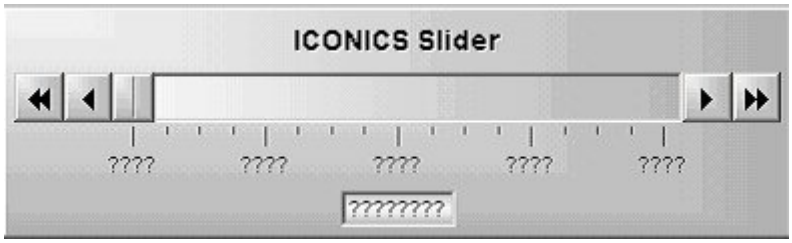
Щелчок на кнопке **GWXSwitch ActiveX** на панели инструментов **ActiveX** открывает Switch ActiveX. Дополнительная информация содержится в документации на **GWXSwitch ActiveX**



Slider ActiveX



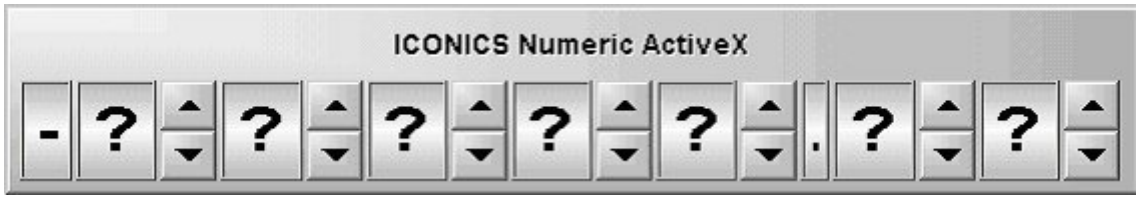
Щелчок на кнопке **GWXSlider ActiveX** на панели инструментов **ActiveX** открывает Slider ActiveX. Дополнительная информация содержится в документации на **GWXSlider ActiveX**



Numeric ActiveX



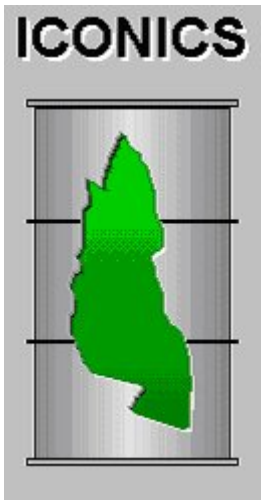
Щелчок на кнопке **GWXNumeric ActiveX** на панели инструментов **ActiveX** открывает Numeric ActiveX. Дополнительная информация содержится в документации на **GWXNumeric ActiveX**



Vessel ActiveX



Щелчок на кнопке **GWXVessel ActiveX** на панели инструментов **ActiveX** открывает Vessel ActiveX. Дополнительная информация содержится в документации на **GWXVessel ActiveX**



Data Mining Control



Щелчок на кнопке **Data Mining Control** на панели инструментов **ActiveX** открывает элемент управления Data Mining. Дополнительная информация содержится в документации на **Data Mining ActiveX**.

12.4 Настройка параметров *GraphWorX32 ActiveX*

Диалоговая панель свойств, вызываемый двойным щелчком на вставленном управляющем элементе, позволяет выполнить настройку ряда параметров *GraphWorX32 ActiveX*.
Внешний вид диалоговой панели свойств *GraphWorX32 ActiveX* показан на рис. 12-2.

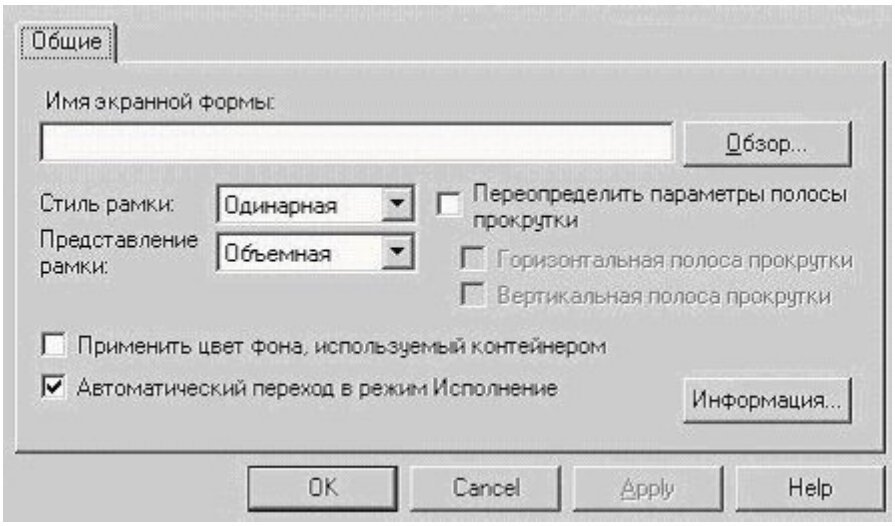


Рис. 12-2. Диалоговая панель свойств GraphWorX32 ActiveX

Описание полей страницы свойств приведено в табл. 12-1.

Таблица 12-1

Описание полей страницы свойств управляющего элемента GraphWorX32 ActiveX

Поле	Описание
Имя экранной формы	Данное поле позволяет задать имя файла экранной формы GraphWorX32, которая будет загружаться для просмотра. Кнопка **Обзор** позволяет выполнить поиск и выбор требуемого файла экранной формы.
Стиль рамки	Данный выпадающий список предназначен для выбора стиля рамки, охватывающей внутреннюю область управляющего элемента. Возможно выбрать рамку в виде линии (*Одинарная*) или отказаться от использования рамки (*Без рамки*).
Представление рамки	Данный выпадающий список позволяет выбрать тип представления рамки, если в списке **Стиль рамки** установлено использование одинарной рамки. При выборе строки *Плоская* рамка будет представлена тонкой линией. При выборе строки *Объемная* рамка примет вид выпуклой линии.
Применить цвет фона, используемый контейнером	Если данный флажок отмечен, то цвет фона внутренней области управляющего элемента будет совпадать с цветом фона приложения-контейнера, в который вставлен *GraphWorX32 ActiveX*. Данная возможность функционирует только в том случае, если приложение-контейнер поддерживает свойство **AmbientBackColor**. GraphWorX32 и Microsoft Visual Basic поддерживают указанное свойство для своих форм.
Автоматический переход в режим Исполнение	Если данный флажок отмечен, *GraphWorX32 ActiveX* будет сразу переводиться в режим *Исполнение*, как только приложение-контейнер перейдет в аналогичный режим. Данная возможность функционирует только в том случае, если приложение-контейнер поддерживает свойство **AmbientUserMode**. GraphWorX32 и Microsoft Visual Basic поддерживают указанное свойство для своих форм.
Переопределить параметры полосы прокрутки	Если данный флажок отмечен, то настройка параметров экранной формы, относящихся к отображению полос прокрутки, будет отменена.
Горизонтальная полоса прокрутки	Если данный флажок отмечен, экранная форма будет иметь горизонтальную полосу прокрутки, расположенную вдоль нижней границы внутренней области *GraphWorX32 ActiveX*. Данное свойство вступает в силу только после установки флажка **Переопределить параметры**

Поле	Описание
	полосы прокрутки.
Вертикальная полоса прокрутки	Если данный флажок отмечен, экранная форма будет иметь вертикальную полосу прокрутки, расположенную вдоль правой границы внутренней области *GraphWorX32 ActiveX*. Данное свойство вступает в силу только после установки флажка **Переопределить параметры полосы прокрутки**.

Глава 13. Справочник по OLE Automation.

13.1 Введение
13.2 Регистрация сервером GenRegistrar в режиме разработки
13.3 Иерархия объектов
13.4 Программирование OLE Automation на C++ и Visual Basic
13.5 Примеры экранов GraphWorX32 OLE Automation
13.6 События
13.7 Перечисляемые константы
13.8 Свойства и методы

13.1 Введение

OLE Automation используется для получения доступа к свойствам и методам объектов GraphWorX32.
Свойства играют роль ссылок на атрибуты объекта (например, экранная форма GraphWorX32 имеет атрибут Цвет фона, доступ к которому осуществляется посредством чтения или изменения свойства BackgroundColor). Методы предназначены для выполнения объектами GraphWorX32 различных действий (например, экранная форма содержит метод FileOpen, вызов которого приводит к загрузке нового файла экранной формы).
Доступ к свойствам и методам может быть получен из внешних приложений Windows, разработанных на Visual Basic, C++ или Delphi.
GraphWorX32 состоит из множества различных типов объектов, каждый из которых обладает собственным набором свойств и методов. Настоящий раздел содержит информацию о методах и свойствах различных типов объектов, входящих в состав GraphWorX32.

13.2 Регистрация сервером GenRegistrar в режиме разработки.

GraphWorX32 регистрируется сервером GenRegistrar в режиме исполнения – регистрируется диспетчерский указатель (**dispatch pointer**) экранной формы с ключом “GWX32_DESIGN_MODE”.
При переходе в режим исполнения он перерегистрирует диспетчерский указатель на <заголовок текущей экранной формы> и сохраняет этот ключ до выхода из режима исполнения, независимо от других загруженных экранных форм. Это обеспечивает распознавание нескольких экземпляров GraphWorX32 в режиме исполнения (на основании заголовка, с которым он был запущен).
Функция **Регистрация Времени Исполнения** по умолчанию отключена. Её включение определяется ключом системного реестра **DesignTimeRegistration** ветви GWX32 / Compatibility Properties:
0 – отключено
1 – включено.



При выходе их режима исполнения GraphWorX32 снова регистрируется с ключом “GWX32_DESIGN_MODE”.
В примере ниже показано, как экземпляр GraphWorX32 в режиме разработки может быть получен из приложения Visual Basic.

Пример.

```
Sub AutomateGwx()
    ' Note: Add ICONICS AutoGenRegistrar to Project | References
    ' Note: Add GWX32 to Project | References

    ' Create an Automation GenRegistrar Wrapper
    Dim genreg As New AutoGenRegistrar
    If genreg Is Not Nothing Then Exit Sub  ' failure

    ' Declare GWX reference and get it from GenRegistrar
    ' Note that we ask for GWX instance in design mode
    Dim gwx As Gwx32.GwxDisplay
    Call genreg.GetDispatch("GWX32_DESIGN_MODE", "GWX32", "*",
gwx)
    If gwx Is Nothing Then Exit Sub         ' failure

    ' Automate GWX instance
    gwx.ShowWindow
    gwx.BringWindowToTop
    Randomize
    gwx.BackgroundColor = RGB(Rnd * 255, Rnd * 255, Rnd * 255)
End Sub
```

13.3 Иерархия объектов

Иерархия объектов GraphWorX32, доступ к которым может осуществляться через OLE Automation, показана на рис. 13-1.

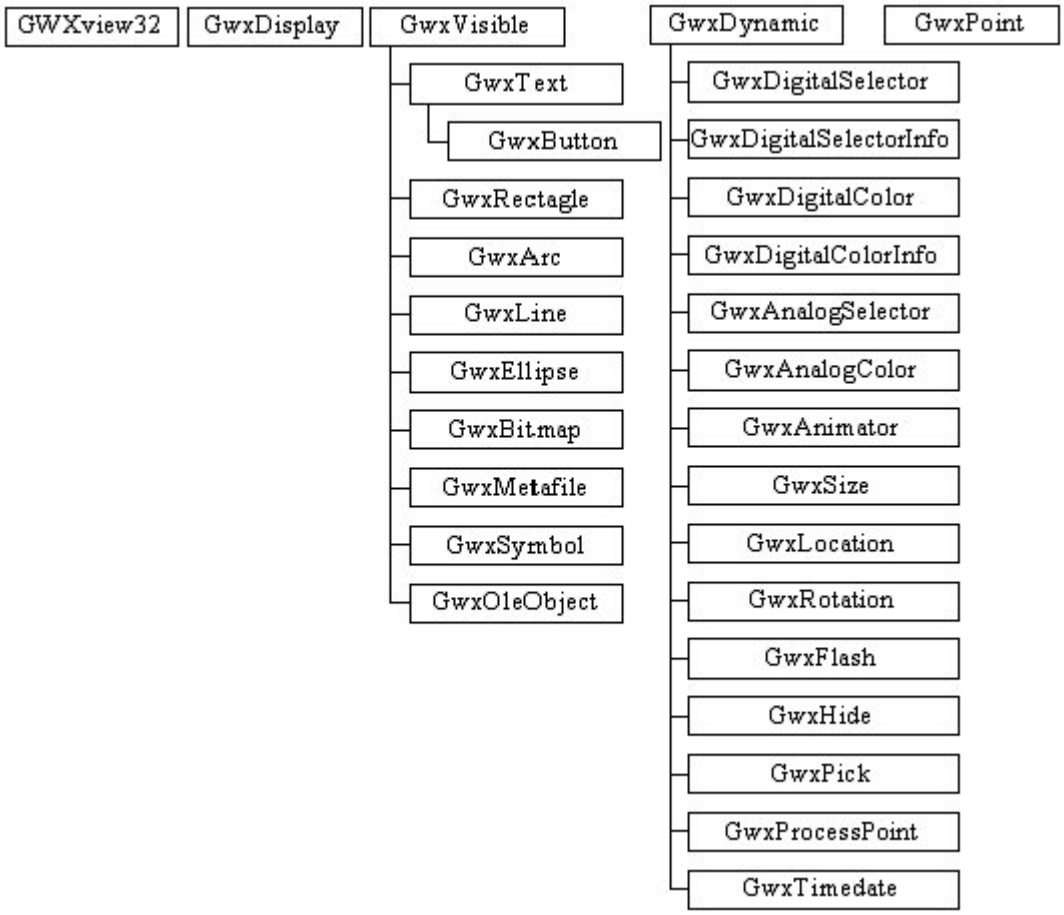


Рис. 13-1. Иерархия объектов OLE Automation GraphWorX32

GWXview32 является базовым типом данных (классом) управляющего элемента *GWXview32 ActiveX* фирмы ICONICS. Для программных компонентов, разрабатываемых при помощи инструментального средства *ICONICS ActiveX ToolWorX*, все создаваемые управляющие элементы ActiveX наследуются от **GWXview32** (наследуют свойства и методы класса **GWXview32**). Каждый объект класса **GWXview32** содержит единственный объект класса **GwxDisplay**.

Класс **GwxDisplay** представляет объекты экранных форм GraphWorX32. Экранные формы GraphWorX32 содержат множество объектов классов, производных от **GwxVisible**, **GwxDynamic** и **GwxPoint**. Объект **GwxDisplay** предоставляет методы для создания новых и получения существующих экземпляров объектов трех перечисленных классов.

GwxVisible является базовым типом объекта (классом) для всех объектов экранных форм GraphWorX32, которые могут быть отображены в ее рабочей области (могут быть визуально восприняты пользователем). Все видимые объекты GraphWorX32 (например, **GwxRectangle**, **GwxEllipse**, **GwxText** и т.д.) наследуются от **GwxVisible**, а значит, приобретают все его свойства и методы.

GwxDynamic является базовым классом объектов экранной формы GraphWorX32, которые выполняют динамическое изменение визуальных атрибутов связанных с ними видимых объектов. Например, объект класса **GwxSize**, связанный с объектом класса **GwxRectangle**, может изменять размеры последнего в зависимости от значения переменной в OPC-сервере. Все классы динамических объектов GraphWorX32 (например, **GwxSize**, **GwxLocation**, **GwxRotation** и т.д.) наследуются от класса **GwxDynamic** и наследуют все его методы и свойства. Динамические объекты не могут восприниматься визуально. Они служат для выполнения динамических действий над свойствами связанных с ними видимых объектов. Объект класса **GwxVisible** может иметь множество связанных с ним объектов класса **GwxDynamic**, тогда как один объект класса **GwxDynamic** может быть связан с одним и только одним объектом класса **GwxVisible**.

GwxPoint является базовым классом объектов в экранных формах GraphWorX32, которые связаны с элементами данных в серверах OPC, значениями выражений, локальными переменными или константами. С каждым динамическим объектом класса **GwxVisible** могут быть связаны один или более объектов класса **GwxPoint**. Объекты класса **GwxPoint** создаются и уничтожаются объектами класса **GwxDynamic**.

13. 4 Программирование OLE Automation на C++ и Visual Basic

Основы наследования

Иерархия объектов OLE Automation GraphWorX32 широко использует наследование. Из этого вытекает ряд особенностей при разработке программного обеспечения на языках программирования C++ и Visual Basic. Для понимания указанных особенностей необходимо предварительно ознакомиться со способами получения доступа к методам и свойствам объектов через интерфейсы OLE Automation.

Примечание. Программистам, предпочитающим язык Си, настоятельно рекомендуется ознакомиться с книгой Дейла Роджерсона "Основы COM", выпущенной российской редакцией Microsoft Press. Особенно следует обратить внимание на раздел, посвященный автоматизации OLE (OLE Automation), и, в частности, на способы доступа к объектам через IDispatch и vtable. Доступ к любому методу, описываемых ниже классов, может быть получен непосредственно через **IUnknown::QueryInterface()**.

Доступ к методу или свойству OLE Automation может быть получен по имени метода/свойства или по числовому идентификатору (ID) метода/свойства. Когда объект производного класса использует унаследованные методы/свойства объекта базового класса, числовые идентификаторы для унаследованных методов/свойств в производном классе должны быть изменены таким образом, чтобы старшее слово числового идентификатора содержало уровень наследования производного класса от базового класса, в котором определены эти свойства и методы. Например, в **GwxVisible** определено свойство **FillColor** (**Цвет заливки**) с числовым идентификатором **0x3**. **GwxRectangle** наследует свойство **FillColor**, имея значение его числового идентификатора равным **0x10003** (**GwxRectangle** является производным классом от **GwxVisible**). Аналогичным образом, **GwxButton** наследует свойство **FillColor**, имея значение его числового идентификатора равным **0x20003** (поскольку **GwxButton** является потомком **GwxText**, который, в свою очередь, является потомком базового класса **GwxVisible**).

Способ нумерации идентификаторов при наследовании не приводит к возникновению каких-либо проблем при осуществлении доступа к реально существующим объектам каких-либо типов. Некоторые сложности проявляются при попытке нисходящего обращения к базовому объекту через объект-потомок. Например, метод **GwxVisibleObjectFromName**, вызываемый для объекта класса **GwxDisplay**, возвращает объект производного класса **GwxVisible**, однако может не быть способа выяснить, какой именно реальный объект на самом деле стоит над возвращенным . В результате этого программист, присваивающий возвращенное значение объявленному ранее объекту типа **GwxVisible** (а не объекту производного класса типа **GwxRectangle** или **GwxEllipse**) и затем пытающийся получить доступ к его свойствам и методам, не достигнет желаемого, поскольку числовой идентификатор вызываемого метода/свойства будет неправильным (если на самом деле возвращенный объект имел тип **GwxRectangle**, а его свойство **FillColor** равнялось **0x10003**, то попытка обращения к свойству **FillColor** ожидаемого объекта типа **GwxVisible** будет неудачной, поскольку идентификатор данного свойства для объектов типа **GwxVisible** равен **0x3**).

В Visual Basic данная проблема решается наиболее просто. Если заранее неизвестен реальный тип возвращаемого объекта, а требуется работать с методами и свойствами, принадлежащими всей последовательности составляющих его объектов базовых и производных классов, используйте объект типа **Object** (интерфейс **IDispatch** при доступе на Си). Это приведет к тому, что Visual Basic будет обращаться к методам и свойствам объекта по именам, а не по числовым идентификаторам.

При использовании C++ для решения описанной выше проблемы потребуется приложить несколько больше усилий. Как правило, при использовании объектов OLE Automation в приложениях, разрабатываемых на C++, можно позволить Microsoft Developer Studio

сгенерировать код диспетчеризации для интерфейса OLE Automation. Например, можно использовать *Мастер классов* (Class Wizard) для генерации производного класса диспетчеризации **COleDispatchDriver**. Либо можно воспользоваться директивой #import для генерации кода диспетчеризации (файлы TLI и TLH). Однако оба описанных способа подразумевают доступ к свойствам и методам через неудобные для восприятия числовые идентификаторы. Следовательно, при использовании подобных классов диспетчеризации всегда необходимо корректно работать с вложенными типами объектов, наследуемых друг от друга. Наилучший способ обхода проблемы состоит в генерации классов диспетчеризации для **GwxVisible** и **GwxDynamic**, что обеспечивает возможность обращения к методам и свойствам по именам, а не по громоздким числовым идентификаторам.

Свойства и методы объектов приведены в представлении Visual Basic. Для получения доп. информации о синтаксисе Visual Basic выберите строку **Просмотр объектов** меню **Вид** в среде разработки VBA и в окне **Просмотр объектов** выберите требуемый метод или свойство. Представление выбранного свойства или метода на Visual Basic будет отображено в нижней области окна **Просмотр объектов**.

13.5 Примеры экранов GraphWorX32 OLE Automation

Инсталяционный диск GENESIS32 содержит примеры экранов GraphWorX32 (.gdf), демонстрирующих использование функций OLE automation.
Несколько экранных форм содержат примеры кода VBA или VBScript для различных функций. Эти экраны разделены на две категории. Первая отражает все аспекты VBA-программирования в GraphWorX. Вторая – VBScript-программирование. Экраны показывают объектное программирование в работе, а также справочную документацию. Всего восемь основных групп экранных форм, использующиеся как источник кода для примеров, дополняющих справочную документацию GraphWorX32.

- GwxdisplaycreateVBA.gdf – Содержит функции и свойства, имеющих отношение к созданию функций. Также эта группа описывает функции и свойства класса GwxVisible. Список этих функций, расположенных по алфавиту, находится в файле GwxdisplaycreateVBA.txt
- GwxDisplayFileVBA.gdf, GwxDisplayFileVB.gdf – Содержат функции и свойства действий над файлами. Список этих функций, расположенных по алфавиту, находится в файлах GwxDisplayFileVBA.txt, GwxDisplayFileVB.txt
- GwxDisplayMiscVBA.gdf, GwxDisplayMiscVB.gdf – Содержат функции и свойства, имеющие разнообразные цели. Они объединены в один общий файл. Перечень этих функций содержится в текстовых файлах GwxDisplayMiscVBA.txt, GwxDisplayMiscVB.txt.
- GwxDisplayObjectsVBA.gdf, GwxDisplayObjectsVB.gdf - Содержат функции и свойства действий над объектами. Полный список этих функций – в GwxDisplayObjectsVBA.txt, GwxDisplayObjectsVB.txt
- GwxDisplayWindowVBA.gdf, GwxDisplayWindowVB.gdf - Содержат функции и свойства действий с окнами режима исполнения и разработки. Список функций содержится в текстовых файлах GwxDisplayWindowVBA.txt, GwxDisplayWindowVB.txt
- GwxDisplayWindow2VBA.gdf, GwxDisplayWindow2VB.gdf- Содержат функции и свойства действий с окнами режима исполнения и разработки. Список функций содержится в текстовых файлах GwxDisplayWindow2VBA.txt, GwxDisplayWindow2VB.txt
- GwxDynamicVBA.gdf, GwxDynamicVB.gdf - Содержат функции и свойства действий с объектами GwxDynami. Список этих функций по алфавиту - в файлах GwxDynamicVBA.txt, GwxDynamicVB.txt
- GwxDisplayAlaisesVBA.gdf, GwxDisplayAlaisesVB.gdf – Описывают использование псевдонимов. Список – в файлах GwxDisplayAlaisesVBA.txt, GwxDisplayAlaisesVB.txt

Примеры экранных форм содержат следующие функции OLE automation:

GwxDisplayAlaisesVB.gdf
•SetAliasDefinition (GwxDisplay)
•SetAliases (GwxDisplay)
•SetAlaisesFromFile (GwxDisplay)
•OpenSetAliasesFromFile (GwxDisplay)
•OpenSetAliasesDialog (GwxDisplay)

•GetLocalAliases (GwxDisplay)
•FileOpenSetAliases (GwxDisplay)
•OpenPopupWinSetAliases (GwxDisplay)
•OpenEmbeddedWinSetAliases (GwxDisplay)
•GetGASThemes (GwxDisplay)
•SetGasThemes (GwxDisplay)
•SetAliasDefinition (GwxVisible)
•SetAlaisesFromFile (GwxVisible)
•SetAliases (GwxVisible)
•SetAliases (GwxDynamic)
•SetAlaisesFromFile (GwxDynamic)
•SetAliasDefinition (GwxDynamic)
•GetAliasDefinition (GwxDynamic)

GwxDisplayAliasesVBA.gdf
•SetAliasDefinition (GwxDisplay)
•SetAliases (GwxDisplay)
•SetAlaisesFromFile (GwxDisplay)
•OpenSetAliasesFromFile (GwxDisplay)
•OpenSetAliasesDialog (GwxDisplay)
•GetLocalAliases (GwxDisplay)
•FileOpenSetAliases (GwxDisplay)
•OpenPopupWinSetAliases (GwxDisplay)
•OpenEmbeddedWinSetAliases (GwxDisplay)
•GetGASThemes (GwxDisplay)
•SetGasThemes (GwxDisplay)
•SetAliasDefinition (GwxVisible)
•SetAlaisesFromFile (GwxVisible)
•SetAliases (GwxVisible)
•GetLanguageAliasString (GwxDynamic)
•SetLanguageAliasString (GwxDynamic)
•SetAliases (GwxDynamic)
•SetAlaisesFromFile (GwxDynamic)
•SetAliasDefinition (GwxDynamic)
•GetAliasDefinition (GwxDynamic)

GwxDisplayCreateVB.gdf
•CreateAnalogColorDynamic
•CreateAnalogSelector
•CreateAnimator
•CreateArc
CreateBitmap
•CreateButton
•CreateRectangle
•CreateDigitalSelector
•CreateEllipse
•CreateFlashDynamic
•CreateHideDynamic
•CreateLayer
•CreateRectangle
•CreateMetafile
•CreatePickDynamic
•CreatePolyline
•CreateProcessPoint
•CreateRectangle
•CreateRotationDynamic
•CreateSizeDynamic
•CreateStateField
•CreateSymbol
•CreateText
•CreateTiemdate

GwxDisplayFileVB.gdf
•CloseWindow
•DisplayBack
•DisplayForward
•ExportBitmapCaptureOfView
•FileNew
•FileOpen
•FileOpenSetAliases
•FilePrint
•FileSave
•FileSaveAs
•FileSaveAsType
•GetFileContaintsVBA
•GetFileName
•GetFilePath
•GetFileVersion
•GetPrintViewArea
•GetPrintWhiteBackground
•GetWorkingDirectory
•OpenEmbeddedWinSetAliases
•SetAliasesFromFile
SetFileVersion
•SetPrintwhiteBackground
•SetWorkingDirectory

GwDisplayFileVBA.gdf
•FileNew
•FileOpen
•FilePrint
•FileSave
•GetFileContaintsVBA
•GetFileName
•FileSaveAsType
•FileOpenSetAliases
•FileSaveAs
•GetFilePath
•GetFileVersion
•SetFileVersion
•SetWorkingDirectory
•GetWorkingDirectory
•DisplayBack
•DisplayForward
•SetAliasesFromFile
•SetPrintwhiteBackground
•GetPrintWhiteBackground
•GetPrintViewArea
•ExportBitmapCaptureOfView
•OpenEmbeddedWinSetAliases
•CloseWindow

GwxDisplayMiscVB.gdf
•GetLocalAliases
•GetOpenPopupByIndex
•GetOpenPopupCount
•GetVCRMode
•SetVCRMode
•VCRControlPanel
•LogicalPointToClient
•LogicalPointToScreen
•OpenTagBrowser
•OpenPopupWinSetAliases
•GetSimulationvariables

•GetLocalVariables
SetAliasDefinition
•SetAliases
•RepalceHostEx
•ReplaceHost
•GetLoadTab
•OpenSetAliasesDialog
•ReplaceStringInString
•SetSystemWideLanguage

GwxDisplayMiscVBA.gdf
•RemoveLayer
•CreateLayer
•DuplicateSelection
•GetEventScriptText
•GetLoadTab
•GetLocalAliases
•GetLocalVariables
•GetOpenPopupByIndex
•GetOpenPopupCount
•GetSimulationVariables
•GetSystemWideLanguage
•GwxGetVBAProject
•GetVCRMode
•LogicalPointToClient
•LogicalPointToScreen
•MoveSelectionBackward
•MoveSelectionForward
•MoveSelectionToBack
•MoveSelectionToFront
•OpenPopupWinSetAliases
•OpenSetAliasesDialog
•OpenTagBrowser
•PopAllCurrentSymbol
•PushCurrentSymbol
•QueryDataType
•QueryRanges
•RemoveLayer
•ReplaceHostEx
•ReplaceHost
•SetAliasDefinition
•SetAliases
•SetLayerOverrideHide
SetModifiedFlag
•SetSystemWideLanguage
•SetVCRMode
•TestCustomSecurityItem
•UngroupSymbol
•VCRControlPanel

GwxDisplayObjectsVB.gdf
•GetObjectTypeName
•GetPointObjectFromName
•GetVisibleObjectFromIndex
•GetVisibleObjectFromName
•GetVisibleObjectFromPosition

GwxDisplayObjectsVBA.gdf
•ConvertGwxSymbolToLibraryObject
•DeleteObject
•DeselectAllObjects
•GetHeadObject
•GetObjectTypeName

•GetPointObjectFromName
•GetVisibleObjectFromIndex
•GetVisibleObjectFromName
•GetVisibleObjectFromPosition
•InsertLibraryObject
•InsertLibraryObjectByIndex
•InsertLibraryObjectByName
•InsertOleObject
•SetHeadObject

GwxDisplayWindow2VB.gdf
•ViewHome
•ViewUnzoom
•ViewPan
•ViewZoomPercent
•ViewFitToWindow
•IsModified
•ViewShowWholeDisplay
•ViewBoxZoom
•ViewZoomCustomDialog
•GetFrameWindowHandle
•EnableRuntimeZoomAndPan
IsEnabledRuntimeZoomAndPan
•DisplayBack
•DisplayForward
•OpenEmbeddedWindow
•IsEmbeddedPopupWindow
•GetClientDimensionsPixels
•GetNumberOfTopLevelVisibles
•OpenPopupWindow
•GetParentDislay
•BackgroundColor
•CloseWindow

GwxDisplayWindow2VBA.gdf
•BackgroundColor
•CloseWindow
•DisplayBack
•DisplayForward
•EnableRuntimeZoomAndPan
•GetClientDimensionsPixels
•GetFrameWindowHandle
•GetNumberOfTopLevelVisibles
•GetParentDislay
•IsEmbeddedPopupWindow
•IsEnabledRuntimeZoomAndPan
•IsModified
•OpenEmbeddedWindow
•OpenPopupWindow
•ViewBoxZoom
•ViewFitToWindow
•ViewHome
•ViewPan
•ViewShowWholeDisplay
•ViewUnzoom
•ViewZoomCustomDialog
•ViewZoomPercent

GwxDisplayWindowVB.gdf
•ShowWindow
•RefreshWindow
•BringWindowToTop
•HideWindow

•StartRuntime
StopRuntime
•ToggleRuntime
•ExitApplication
•MinimizeWindow
•MaximizeWindow
•RestoreWindow
•GetDisplayDimensions2
•SetDisplayDimensions
•SetViewDimensions
•GetViewDimensions2
•IsRuntimeMOde
•SetWindowDimensionsPixels
•SetWindowDimensionsPercent
•GetWindowDimensionsPercent2
•FetWindowDimensionsPixels2
•isPopupWindow
•OpenPopupWindow
•CloseAllPopupWindow
•closePopupwithTitle
•CloseWindow

GwxDynamicVB.gdf
•dataSource
•highRange
•lowRange
•ObjectName
•RangeOverride
•RequestDataType
•TimeRate
•UserCustomData
•UserDescriptuon
•GetAliasdefinition
•GetdataSourcePointObject
•GetHighRangePointObject
•GetLowRangePointObject
•GetNumberOfSubDynamic
•GetObjectTypeName
•GetSubDynamicObjectFromIndex
•GetVisibleObject
•SetAliasDefinition
•SetAliases
•SetAliasesFromFile

GwxDisplayWindowVBA.gdf
•ShowWindow
•RefreshWindow
•BringWindowToTop
•HideWindow
•StartRuntime
•StopRuntime
•ToggleRuntime
•ExitApplication
•MinimizeWindow
•MaximizeWindow
•RestoreWindow
•GetDisplayDimensions2
•SetDisplayDimensions
•SetViewDimensions
•GetViewDimensions2
•IsRuntimeMOde
•SetWindowDimensionsPixels
•SetWindowDimensionsPercent

•GetWindowDimensionsPercent2
•FetWindowDimensionsPixels2
•isPopupWindow
•OpenPopupWindow
•CloseAllPopupWindow
•closePopupwithTitle
•CloseWindow

GwxDynamicVBA.gdf
•dataSource
•GetAliasdefinition
•GetdataSourcePointObject
•GetHighRangePointObject
•GetLowRangePointObject
•GetNumberOfSubDynamic
•GetObjectTypeName
•GetSubDynamicObjectFromIndex
•GetVisibleObject
•highRange
•lowRange
•ObjectName
•RangeOverride
•RequestDataType
SetAliasDefinition
•SetAliases
•SetAliasesFromFile
•TimeRate
•UserCustomData
•UserDescription

13. 6 События

Ниже описаны события, которые могут обрабатываться во встроенной среде разработки и исполнения сценарных процедур Visual Basic для приложений. Данные события также генерируются управляющим элементом *GWXview32 ActiveX*.
Примечание. Для разработчиков, использующих ActiveX ToolWorX, имеется ряд виртуальных функций, соответствующих данным событиям.

Event DataEntryActivated(DataEntry As Object)

Данное событие возникает в момент, когда пользователь активизирует объект ввода данных (например, щелкает мышью на элементе ввода данных для ввода нового значения). Аргумент **DataEntry** является объектом **GwxProcessPoint**, с которым связано данное событие.

Event DataEntryDeactivated(DataEntry As Object)

Данное событие возникает в момент, когда объект ввода данных теряет фокус ввода (например, пользователь щелкнул мышью вне пределов области отображения объекта ввода данных, нажал клавишу Esc либо завершил ввод данных). Аргумент **DataEntry** является объектом **GwxProcessPoint**, с которым связано данное событие.

Event DataEntryValueEntered(DataEntry As Object)

Данное событие возникает в момент, когда пользователь вводит новое значение в поле объекта ввода данных. Аргумент **DataEntry** является объектом **GwxProcessPoint**, с которым связано данное событие.

Event DialBeginDrag(Dial As Object)

Данное событие возникает в момент, когда пользователь нажал левую клавишу мыши, поместив курсор в область отображения объекта типа *Круговая шкала* GraphWorX32. Аргумент **Dial** является объектом типа **GwxRotation**, с которым связано данное событие.

Event DialDrag(Dial As Object)

Данное событие возникает, когда пользователь вращает "рукоятку" *круговой шкалы*. Аргумент **Dial** является объектом типа **GwxRotation**, с которым связано данное событие.

Event DialEndDrag(Dial As Object)

Данное событие возникает в момент, когда пользователь отпустил левую клавишу мыши, завершив вращение "рукоятки" *круговой шкалы*. Аргумент **Dial** является объектом типа **GwxRotation**, с которым связано данное событие.

Event DisplayKeyDown(KeyCode As Long, Shift As Long, CancelProcessing)

Event DisplayKeyUp(KeyCode As Long, Shift As Long, CancelProcessing)

Данные события возникают при нажатии (DisplayKeyDown) или отпускании (DisplayKeyUp) клавиши. Можно подавлять возникновение событий, установив параметру **CancelProcessing** значение **True**.
Примечание: событие будет возникать только в случае разрешения соответствующих событий программно(EnableRuntimeEvents) или вручную, установив флажок в диалоговом окне **Формат** – **Параметры приложения** – **VBA**.
Параметры:

KeyCode
: Key code, such as **vbKeyHome** (the HOME key).

Shift
: Целое число, соответствующее состоянию клавишей SHIFT, CTRL, и ALT во время события.

CancelProcessing
: Значение типа Variant, подавляющее возникновение событий в GraphWorX32, если установлено в True

Таблица, показывающая значения для констант *Shift*.

Константа	Значение	Описание
vbShiftMask	1	SHIFT key bit mask.
vbCtrlMask	2	CTRL key bit mask.
vbAltMask	4	ALT key bit mask.

Event DisplayLoad()

Данное событие возникает после загрузки экранной формы GraphWorX32.

Event DisplayMouseDblClick(Button As Long, Shift As Long, X As Single, Y As Single, CancelProcessing)

Event DisplayMouseDown(Button As Long, Shift As Long, X As Single, Y As Single, CancelProcessing)

Event DisplayMouseMove(Button As Long, Shift As Long, X As Single, Y As Single, CancelProcessing)

Event DisplayMouseUp(Button As Long, Shift As Long, X As Single, Y As Single, CancelProcessing)

Данные события возникают, если на клавише мыши произведен двойной щелчок (DisplayMouseDblClick), клавиша нажата (DisplayMouseDown), отпущена (DisplayMouseUp), или изменилось положение мыши (DisplayMouseMove). Можно подавлять возникновение событий, установив параметру **CancelProcessing** значение **True**.
Примечание: событие будет возникать только в случае разрешения соответствующих событий программно(EnableRuntimeEvents) или вручную, установив флажок в диалоговом окне **Формат** – **Параметры приложения** – **VBA**.
Параметры:

button

Целое число, определяющее состояние клавишей мыши: соответствующий бит значения установлен , если клавиша мыши нажата. Биты в значении соответствуют клавишам мыши следующим образом:
левая клавиша – бит 0 (бит установлен – значение равно 1)
правая клавиша – бит 1 (бит установлен – значение равно 2)
средняя клавиша – бит 2 (бит установлен – значение равно 4)

CancelProcessing

Shift

Целое число, определяющее состояние клавишей SHIFT, CTRL, и ALT: соответствующий бит значения установлен , если клавиша нажата. Биты в значении соответствуют клавишам следующим образом:
SHIFT – бит 0 (бит установлен – значение равно 1)
CTRL – бит 1 (бит установлен – значение равно 2)
ALT – бит 2 (бит установлен – значение равно 4)
Пример: если нажаты клавиши CTRL и ALT, значение будет равно 6.

x, *y*

Значение, определяющее текущее положение указателя мыши. Значения *x* и *y* всегда определяются в терминах системы координат, определенной свойствами объекта **ScaleHeight**, **ScaleWidth**, **ScaleLeft**, и **ScaleTop.**

CancelProcessing

Значение типа Variant, подавляющее возникновение событий в GraphWorX32, если установлено в True

Event DisplayUnload()

Данное событие возникает перед выгрузкой экранной формы GraphWorX32 (например, перед загрузкой новой экранной формы или перед завершение работы GraphWorX32).

Event LanguageChanged(long LocaleID)

Данное событие возникает при переключении GraphWorX32 на другой язык. Параметр **localeID** является идентификатором (locale ID) нового языка. Не поддерживается для экранных форм, просматриваемых внутри объектов **GWXview32**.

Event PickPostDown(Pick As Object)

Данное событие возникает в момент, когда пользователь совершил щелчок мышью на графическом объекте GraphWorX32, с которым связано динамическое действие *Указание и щелчок*. Событие возникает после выполнения операции, связанной с моментом "**Если нажато**". Аргумент **Pick** является объектом типа **GwxPick**, с которым связано данное событие.

Event PickPostUp(Pick As Object)

Данное событие возникает в момент, когда пользователь отпустил клавишу мыши после щелчка на графическом объекте GraphWorX32, с которым связано динамическое действие *Указание и щелчок*. Событие возникает после выполнения операции, связанной с моментом "**Если отпущено**". Аргумент **Pick** является объектом типа **GwxPick**, с которым связано данное событие.

Event PickPostWhileDown(Pick As Object)

Данное событие удерживается, пока пользователь удерживает нажатой клавишу мыши после щелчка на графическом объекте GraphWorX32, с которым связано динамическое действие *Указание и щелчок*. Событие возникает после выполнения операции, связанной с моментом "**Пока нажато** ". Аргумент **Pick** является объектом типа **GwxPick**, с которым связано данное событие.

Event PickPreDown(Pick As Object)

Данное событие возникает в момент, когда пользователь совершает щелчок мышью на графическом объекте GraphWorX32, с которым связано динамическое действие *Указание и щелчок*. Событие возникает перед выполнением операции, связанной с моментом "**Если нажато**". Аргумент **Pick** является объектом типа **GwxPick**, с которым связано данное событие.

Event PickPreUp(Pick As Object)

Данное событие возникает в момент, когда пользователь отпускает клавишу мыши после щелчка на графическом объекте GraphWorX32, с которым связано динамическое действие *Указание и щелчок*. Событие возникает перед выполнением операции, связанной с моментом "**Если отпущено**". Аргумент **Pick** является объектом типа **GwxPick**, с которым связано данное событие.

Event PickPreWhileDown(Pick As Object)

Данное событие возникает, пока пользователь удерживает нажатой клавишу мыши после щелчка на графическом объекте GraphWorX32, с которым связано динамическое действие *Указание и щелчок*. Событие возникает перед выполнением операции, связанной с моментом "**Пока нажато** ". Аргумент **Pick** является объектом типа **GwxPick**, с которым связано данное событие.

Event PostAnimateDisplay()

Данное событие возникает после того, как GraphWorX32 обновил рабочую область экранной формы в момент перехода из режима *Разработка* в режим *Исполнение*.

Event PostDeanimateDisplay()

Данное событие возникает после того, как GraphWorX32 обновил рабочую область экранной формы в момент перехода из режима *Исполнение* в режим *Разработка*.

Event PostRuntimeStart()

Данное событие возникает после того, как GraphWorX32 перешел в режим *Исполнение*. Событие следует после **PostAnimateDisplay**.

Event PostRuntimeStop()

Данное событие возникает после того, как GraphWorX32 перешел в режим *Разработка*. Событие следует после **PostDeanimateDisplay**.

Event PreAnimateDisplay()

Данное событие возникает перед тем, как GraphWorX32 обновил рабочую область экранной формы в момент перехода из режима *Разработка* в режим *Исполнение*. Событие возникает после **PreRuntimeStart**. Обновление экранной формы состоит в выполнении запроса списка тегов от серверов OPC, которые связаны с динамическими графическими объектами экранной формы. (Фактически, в момент обновления экранной формы на серверах OPC создаются группы элементов OPC, указывающих на реальные теги в серверах, которые отображаются в GraphWorX32). При переходе в режим *Разработка* из режима *Исполнение* происходит отсоединение от серверов OPC (удаление OPC-групп).

Event PreDeanimateDisplay()

Данное событие возникает перед тем, как GraphWorX32 прекратил обновление экранной формы (сразу после щелчка в области меню **Исполнение** GraphWorX32 или вызова соответствующего метода OLE Automation). Событие предшествует удалению OPC-групп и освобождению соответствующих тегов OPC-серверов.

Event PreRuntimeStart()

Данное событие возникает перед тем, как GraphWorX32 перешел в режим *Исполнение*.

Event PreRuntimeStop()

Данное событие возникает перед тем, как GraphWorX32 перешел в режим *Разработка* (до события **PreDeanimateDisplay**).

Event SliderBeginDrag(Slider As Object)

Данное событие возникает в момент, когда пользователь нажал левую клавишу мыши, поместив курсор в область отображения объекта типа *Движковый регулятор* GraphWorX32. Аргумент **Slider** является объектом типа **GwxLocation**, с которым связано данное событие.

Event SliderDrag(Slider As Object)

Данное событие возникает, когда пользователь перемещает движок *Движкового регулятора*. Аргумент **Slider** является объектом типа **GwxLocation**, с которым связано данное событие.

Event SliderEndDrag(Slider As Object)

Данное событие возникает в момент, когда пользователь отпустил левую клавишу мыши, завершив перемещение движка регулятора. Аргумент **Slider** является объектом типа **GwxLocation**, с которым связано данное событие. Указанное событие также возникает при управлении регулятором с помощью клавиатуры.

Event PreAnimateLayer(LayerName As String)

Данное событие возникает перед тем, как будет выполнена операция запроса данных OPC для слоя.

Event PostAnimateLayer(LayerName As String)

Данное событие возникает после того, как выполнена операция запроса данных OPC для слоя.

Event PreDeanimateLayer(LayerName As String)

Данное событие возникает перед тем, как данные OPC, получаемые слоем, будут отпущены.

Event PostDeanimateLayer(LayerName As String)

Данное событие возникает после того, как данные OPC, получаемые слоем, отпущены.

13.7 Перечисляемые константы

Перечисляемые константы не могут быть использованы в редакторе VBScript. Поэтому должны использоваться соответствующие числовые переменные. Например, следующий код открывает экранную форму и устанавливает значения псевдонима путем чтения их из файла.
ThisDisplay.FileOpenSetAliases("myFileName.gdf", AliasSetFromFile, "AliasCommandString.txt")
В сценариях VBScript и Jscript необходимо заменить константы AliasSetFromFile соответствующими числовыми переменными, например 1.
Т.е. рассматриваемый код в VBScript примет вид:
ThisDisplay.FileOpenSetAliases("myFileName.gdf", 1,"AliasCommandString.txt")

GWX3DEDGESTYLE

```
helpstring("3D Edge Style")]
enum {
EdgeNone = 0,
EdgeRaised = 5,
```

EdgeEtched = 6,
EdgeBump = 9,
EdgeSunken = 10
} GWX3DEDGESTYLE;

GWXLINESTYLE

helpstring("Line Style")]
enum {
LineSolid = 0,
LineDash = 1,
LineDot = 2,
LineDashDot = 3,
LineDashDotDot = 4,
LineNone = 5
} GWXLINESTYLE;

GWXTEXTALIGNMENT

helpstring("Text Alignment")]
enum {
TextAlignLeft = 0,
TextAlignCenter = 1,
TextAlignRight = 2
} GWXTEXTALIGNMENT;

GWXSIZEDYNTYPE

helpstring("Size Dynamic Direction")]
enum {
SizeLeft = 0,
SizeRight = 1,
SizeUp = 2,
SizeDown = 3,
SizeUpLeft = 4,
SizeUpRight = 5,
SizeDownLeft = 6,
SizeDownRight = 7,
SizeLeftRight = 8,
SizeUpDown = 9,
SizeLeftRightBias = 10,
SizeUpDownBias = 11,
SizeAllFour = 12,
SizeLeftRightUp = 13,
SizeLeftRightDown = 14,
SizeUpDownLeft = 15,
SizeUpDownRight = 16
} GWXSIZEDYNTYPE;

GWXPICKACTION

helpstring("Pick Action")]
enum {
PickLoadDisplay = 0,
PickDragDropLoad = 1,
PickPopupWindow = 2,
PickDownloadValue = 3,
PickToggleValue = 4,
PickLaunchApp = 5,
PickClose = 6,
PickRunScript = 7,
PickCustomCommand = 8,
PickEmbeddedWindow = 9,
PickDisplayBack = 10,
PickDisplayForward = 11,
PickSetAliases = 12,
PickAliasesDialog = 13,
PickLayerHideOn = 14,
PickLayerHideOff = 15,
PickLayerHideToggle = 16,

PickPopupUserMenu = 17,
PickSwitchLanguage = 18,
PickSelectThema = 19,
PickLaunchWebPage = 20
} GWXPICKACTION;

GWXSCRIPTTYPE

helpstring("Script Type")]
enum {
ScriptTypeNone = -1,
ScriptTypeVBA = 0,
ScriptTypeVBScript = 1,
ScriptTypeJScript = 2
} GWXSCRIPTTYPE;

GWXEVENTNAME

helpstring("Event Name")]
enum {
DataEntryActivated = 0,
DataEntryDeactivated = 1,
DataEntryValueEntered = 2,
DialBeginDrag = 3,
DialDrag = 4,
DialEndDrag = 5,
DisplayLoad = 6,
PostRuntimeStop = 26,
SliderDrag = 29,
GlobalAliasChanged = 32,
DisplayUnload = 7,
LanguageChanged = 8,
PickPostDown = 9,
PickPostUp = 10,
PickPostWhileDown = 11,
PickPreDown = 12,
PickPreUp = 13,
PickPreWhileDown = 14,
PostAnimateDisplay = 15,
PostDeanimateDisplay = 16,
PreAnimateDisplay = 17,
PreDeanimateDisplay = 18,
PreAnimateLayer = 19,
PostAnimateLayer = 20,
PreDeanimateLayer = 21,
PostDeanimateLayer = 22,
PreRuntimeStart = 23,
PostRuntimeStart = 24,
PreRuntimeStop = 25,
SliderBeginDrag = 27,
SliderEndDrag = 28,
VCRStart = 30,
VCRStop = 31,
ViewChanged = 33,
DisplayChanged = 34,
WindowChanged = 35,
PrePrint = 36,
PostPrint = 37
} GWXEVENTNAME;

GWXALIASCOMMAND

helpstring("Alias Command")]
enum {
AliasNoCommand = 0,
AliasSetFromFile = 1,
AliasSetDirect = 2
} GWXALIASCOMMAND;

GWXBUTTONTYPE

```
helpstring("Button Type")]
enum {
ButtonNormal = 0,
ButtonCheck = 1,
helpstring("Data Type")]
ButtonRadio = 2
} GWXBUTTONTYPE;
```

GWXMOUSEBUTTON

```
helpstring("Mouse Button")]
enum {
MouseButtonLeft = 0,
MouseButtonMiddle = 1,
MouseButtonRight = 2
} GWXMOUSEBUTTON;
```

GWXEXECUTIONTRIGGER

```
helpstring("Execution Trigger")]
enum {
TriggerOnDown = 1,
TriggerWhileDown = 2,
TriggerOnDnWhileDn = 3,
TriggerOnUp = 4,
TriggerOnDnOnUp = 5,
TriggerWhileDnOnUp = 6,
TriggerOnDnWhileDnOnUp = 7
} GWXEXECUTIONTRIGGER;
```

GWXTIMEDATEFORMATTYPE

```
helpstring("Timedate Format Type")]
enum {
TimeOnly = 0,
DateOnly = 1,
TimeAndDate = 2
} GWXTIMEDATEFORMATTYPE;
```

GWXDATATYPE

```
enum {
DataTypeEmpty = 0,
DataTypeShort = 2,
DataTypeLong = 3,
DataTypeFloat = 4,
DataTypeDouble= 5,
DataTypeString = 8,
DataTypeBool = 11,
DataTypeChar = 16,
DataTypeByte = 17,
DataTypeWord = 18,
DataTypeDword= 19
} GWXDATATYPE;
```

GWXSCALEMODE

```
helpstring("Scale Mode")]
enum {
FixedScale = 0,
Scaleable = 1,
ScaleablePreserveAspect = 2
} GWXSCALEMODE;
VerMGraphics40 = 3,
```

GWXARCTYPE

```
helpstring("Arc Type")]
enum {
ArcArc = 0,
```

ArcPie = 1,
ArcChord = 2
} GWXARCTYPE;

GWXGRADIENTSTYLE

helpstring("Gradient Style")]
enum {
GradientHorizontal = 0,
GradientVertical = 1,
GradientSquare = 2
} GWXGRADIENTSTYLE;

GWXGETFILEVERSION

helpstring("Get File Version")]
enum {
OpenFileError = -1,
Ver520 = 0,
Ver600 = 1,
Ver600SP = 2,
Ver610 = 4,
Ver620 = 5,
VerLatest = 9999
} GWXGETFILEVERSION;

GWXSETFILEVERSION

helpstring("Set File Version")]
enum {
SetVer600SP = 2,
SetVerMGraphics40 = 3,
SetVer610 = 4,
SetVerLatest = 9999
} GWXSETFILEVERSION;

GWXSECURITYWRITESTATUS

helpstring("Get Security Write Status")]
enum {
WriteUninitialized = -1,
WriteEnabled = 0,
WriteDisabled = 1,
WriteCritical = 2
} GWXSECURITYWRITESTATUS;

13.8 Свойства и методы.

GraphWorX32 включает следующие классы OLE Automation:
GwxView32
GwxDisplay
GwxVisible
GwxDynamic
GwxPoint

GwxView32

GWXview32 является базовым классом управляющего элемента *GWXview32 ActiveX* фирмы ICONICS. Для программных компонентов, разрабатываемых при помощи инструментального средства *ICONICS ActiveX ToolWorX*, все создаваемые управляющие элементы ActiveX наследуются от **GWXview32**.
Объекты **GWXview32** могут создаваться через программный идентификатор (ProgID)
GWXVIEW32.GWXview32Ctrl.1

Свойства

AutoStartRuntime As Boolean

При равенстве TRUE элемент управления автоматически переходит в режим *Исполнение*, как только содержащий его контейнер переводится в режим *Исполнение*. Данное свойство имеет силу только в том случае, если контейнер поддерживает свойство AmbientUserMode.

HorizontalScrollbar As Boolean

При равенстве TRUE горизонтальная полоса прокрутки элемента управления становится видимой независимо от параметров полос прокрутки, заданных в свойствах загруженной экранной формы. При равенстве FALSE горизонтальная полоса прокрутки всегда невидима. Состояние данного свойства не имеет значения, если свойство OverrideScrollbarSettings равно FALSE.

OverrideScrollbarSettings As Boolean

При равенстве TRUE будут переопределены свойства экранной формы, относящиеся к полосам прокрутки, в соответствии с установками свойств HorizontalScrollbar и VerticalScrollbar. При равенстве FALSE наличие полос прокрутки у нижней и правой границ внутренней области экранной формы будет определяться свойствами, заданными для загруженной экранной формы.

UseAmbientBackColor As Boolean

При равенстве TRUE цвет фона внутренней области управляющего элемента будет совпадать с цветом фона контейнера, в который он вставлен. Данное свойство работает только в том случае, если контейнер поддерживает свойство AmbientBackColor.

VerticalScrollbar As Boolean

При равенстве TRUE вертикальная полоса прокрутки элемента управления становится видимой независимо от параметров полос прокрутки, заданных в свойствах загруженной экранной формы. При равенстве FALSE вертикальная полоса прокрутки всегда невидима. Состояние данного свойства не имеет значения, если свойство OverrideScrollbarSettings равно FALSE.

DisplayName As String

Имя файла экранной формы, загружаемой в элемент управления. Изменение значения данного свойства во время исполнения приводит к выгрузке текущей экранной формы и загрузке вновь заданной.

Appearance As Integer

Внешний вид рамки окна элемента управления. Допустимые значения:

- Плоская = 0
- Трехмерная = 1

BorderStyle As Integer

Стиль рамки окна элемента управления. Допустимые значения:

- Рамка отсутствует = 0
- Обычная рамка в виде тонкой линии = 1

Методы

Function GetDisplay() As Object

Возвращает объект типа **GwxDisplay** для текущей экранной формы, загруженной в управляющий элемент. Указанный объект может использоваться для доступа ко всем свойствам и методам всех объектов, описание которых приведено в настоящем разделе.

Function ReplaceFilePath(OldSubstring As String, NewSubstring As String) As Long

Заменяет путь к файлу в динамических элементах. Возвращает 0 при успешном выполнении и HRESULT – при неудачном.

```
'similar to ReplaceTag
'VBA example, works only in configure mode
'replaces file path in whole display

Dim Status As Long
Status = ThisDisplay.ReplaceFilePath("C:\Windows\Temp",
"D:\Temp")
'if Status <> 0 then there was no replacement performed or an
error occurs
If Status <> 0 Then
 MsgBox "No replacements"
Else
 MsgBox "Tags replaced"
End If
```

Function ReplaceHost(OldHostName As String, NewHostName As String) As Long

Заменяет имя узла в строках тегов и пути к файлам. Возвращает 0 при успешном выполнении и HRESULT – при неудачном.

```
'similar to ReplaceTag
'VBA example, works only in configure mode
'replaces host name in whole display

Dim Status As Long
Status = ThisDisplay.ReplaceHost("Host1", "Host2")
'if Status <> 0 then there was no replacement performed or an
error occurs
If Status <> 0 Then
 MsgBox "No replacements"
Else
 MsgBox "Tags replaced"
End If
```

Function ReplaceHostEx(OldHostNameSubstring As String, NewHostNameSubstring As String, MatchCase As Boolean, MatchWholeWord As Boolean) As Long

Заменяет имя узла в строках тегов и пути к файлам. Поддерживает регистр символов, групповые символы и флаг MatchWholeWord. Возвращает 0 при успешном выполнении и HRESULT – при неудачном.

```
'similar to ReplaceTag
'VBA example, works only in configure mode
'replaces host name in whole display, regarding "case" and "whole
words" options

Dim Status As Long
Status = ThisDisplay.ReplaceHostEx("Host1", "Host2", True, True)
'if Status <> 0 then there was no replacement performed or an
error occurs
If Status <> 0 Then
```

```
 MsgBox "No replacements"
Else
 MsgBox "Tags replaced"
End If
```

Function ReplaceTag(OldSubstring As String, NewSubstring As String) As Long

Заменяет теги объектов динамики во всей экранной форме. Заменяет **OldSubstring** на **NewSubstring** в определенных ситуациях. Возвращает 0 при успешном выполнении и HRESULT – при неудачном.
Пример применения метода:

```
'VBA example, works only in configure mode
'replaces tag name in whole display

Dim Status As Long
Status = ThisDisplay.ReplaceTag("gfwsim.ramp.float",
"gfwsim.sine.double")
'if Status <> 0 then there was no replacement performed or an
error occurs
If Status <> 0 Then
 MsgBox "No replacements"
Else
 MsgBox "Tags replaced"
End If
```

long ReplaceStringInString (BSTR StringToReplace, BSTR OldSubstring, BSTR NewSubsring)

Заменяет **OldSubstring** на **NewSubstring** в **StringToReplace**. Возвращает 0 при успешном выполнении и HRESULT – при неудачном.

Sub SelectLanguage(Language As String, LanguageSubset As String, LocaleID As Long, SynchronizeLCID As Boolean)

Sub SetGasParentCookie(newCookie As Long)

Внутреннее использование для соединения между ActiveX и контейнером. Не подлежит изменению.

Sub StartRuntime()

Переводит управляющий элемент в режим *Исполнение*.

Sub StopRuntime()

Переводит управляющий элемент в режим *Разработка*.

Пример

```
'Процедура вызывается по событию от таймера ICONICS Timer32,
'который вставлен в экранную форму GraphWorX32
'Работа процедуры показана на рис. 13-2
Private Sub AxTimer1_Timer()

Dim gwxDisplayInner As GwxDisplay 'Экранная форма, загруженная в GWXview32 ActiveX
Set gwxDisplayInner = GWXview321.GetDisplay()

Dim pptOuter As GwxPoint 'Параметр в экранной форме-контейнере
Dim pptInner As GwxPoint 'Параметр в экранной форме ActiveX

'Получаем локальную переменную ~~OuterSlider~~ в экранной форме-контейнере
Set pptOuter = ThisDisplay.GetPointObjectFromName("~~OuterSlider~~")
```

```
'Получаем локальную переменную ~~Slider1~~ в экранной форме в ActiveX
Set pptInner = gwxDisplayInner.GetPointObjectFromName("~~Slider1~~")
'Присваиваем локальной переменной в контейнере значение переменной
'из ActiveX
pptOuter.Value = pptInner.Value
'Отпускаем объекты
Set pptInner = Nothing
Set pptOuter = Nothing
Set gwxDisplayInner = Nothing
End Sub

'Процедура вызывается в экранной форме-контейнере по нажатию кнопки и переключает
'режим экранной формы, загруженной в ActiveX. Расположена в отдельном модуле.
Sub StartStopRuntimeForInner(o As GwxPick)

Dim gwxDisplayInner As GwxDisplay 'Экранная форма, загруженная в GWXview32 ActiveX

'Получаем объект GwxDisplay из модуля ThisDisplay
Set gwxDisplayInner = ThisDisplay.GWXview321.GetDisplay()

If Not (gwxDisplayInner Is Nothing) Then
   If gwxDisplayInner.IsRuntimeMode Then
      gwxDisplayInner.StopRuntime 'Останов режима Исполнение
   Else
      gwxDisplayInner.StartRuntime 'Переход в режим Исполнение
   End If
Set gwxDisplayInner = Nothing
End If
End Sub
```

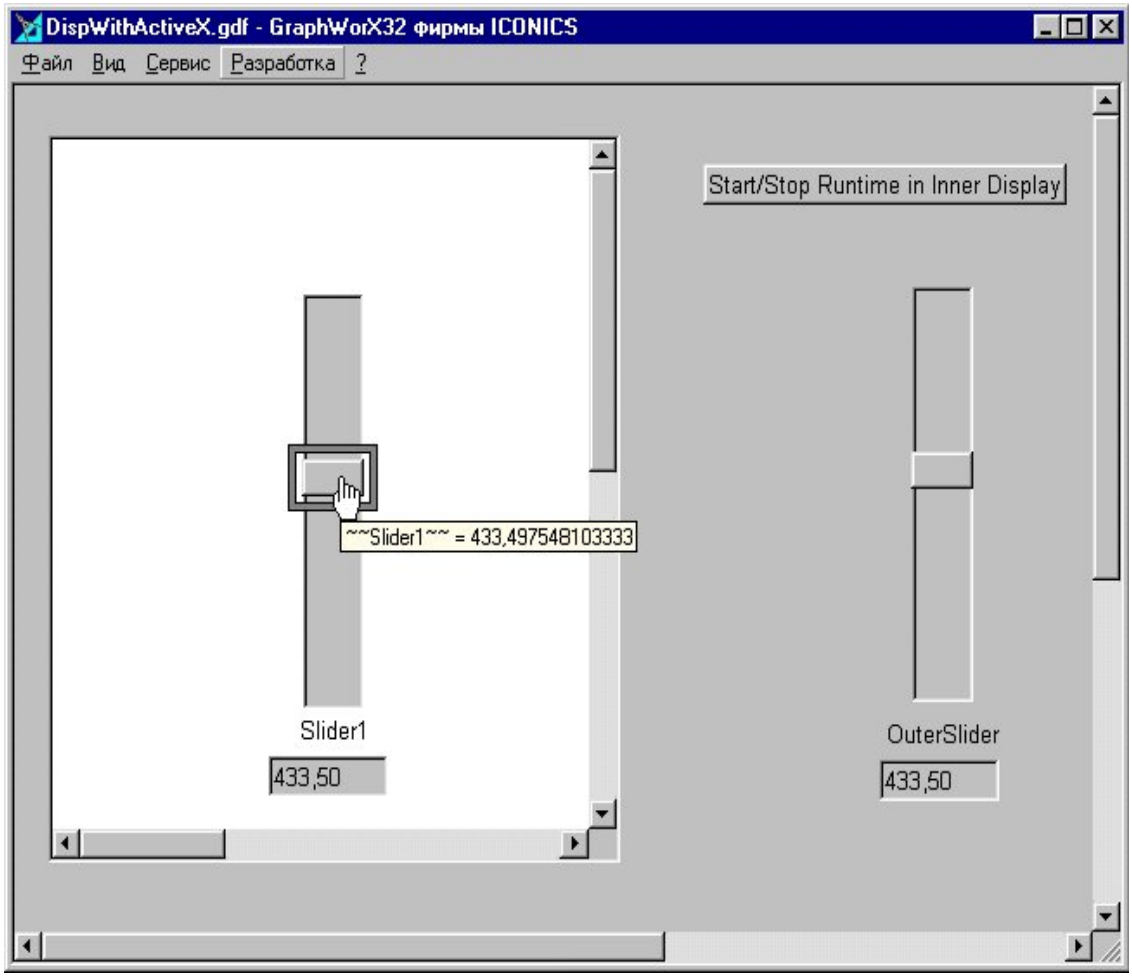


Рис. 13-2. Работа экранной формы с вставленным управляющим элементом GWXview32 ActiveX

GwxDisplay

Данный тип представляет экранные формы GraphWorX32. Свойства и методы объектов данного типа описаны ниже. Кроме того, ниже приведено описание событий, обрабатываемых во встроенной среде разработки и исполнения сценарных процедур GraphWorX32 Microsoft VBA. Объекты типа **GwxDisplay** могут создаваться по программному идентификатору (ProgID) **Gwx32.Display**.

Свойства

BackgroundColor As OLE_COLOR

Устанавливает или возвращает цвет фона для текущей экранной формы.

Пример VBA:

```
Dim red As Byte
Dim green As Byte
Dim blue As Byte
Dim rgbOLE As OLE_COLOR

rgbOLE = ThisDisplay.BackgroundColor

red = 255
green = 255
blue = 0

ThisDisplay.BackgroundColor = rgb(red, green, blue)

MsgBox "Undo BacgroundColor"

ThisDisplay.BackgroundColor = rgbOLE
```

Пример VB:

```
Dim red
Dim green
Dim blue
Dim rgbOLE

rgbOLE = ThisDisplay.BackgroundColor

red = 255
green = 255
blue = 0

ThisDisplay.BackgroundColor = RGB(red, green, blue)

MsgBox "Undo BacgroundColor"

ThisDisplay.BackgroundColor = rgbOLE
```

IsEnabledRuntimeZoomAndPan As Boolean

Пример VBA:

```
Dim flag As Boolean
MsgBox "Is runtime zoom and pan enabled?" +
Chr(13) + CStr(ThisDisplay.IsEnabledRuntimeZoomAndPan)
```

Пример VB:

```
MsgBox "Is runtime zoom and pan enabled?" +
Chr(13) + CStr(ThisDisplay.IsEnabledRuntimeZoomAndPan)
```

Redraw As Boolean

При равенстве TRUE вызовы методов и свойств OLE Automation, приводящие к изменению видимых атрибутов графических объектов экранной формы, будут автоматически вызывать обновление (перерисовку) внутренней области экранной формы.

При равенстве FALSE пользователь должен явно выполнять обновление экрана (например, можно изменить видимые атрибуты большого количества объектов, после чего выполнить обновление экрана один раз).

```
'To run this script you need to create an object with
'name "Test2".
'Toggle Redraw so the display will
'not be able to automatically
'refresh. At the end of this script
'we will manualy call RefreshWindow.
ThisDisplay.Redraw = False

'Get rectangle from name:
Dim oRec As GwxRectangle Set oRec =
ThisDisplay.GetVisibleObjectFromName("Test2")
oRec.Selected = True

'Set Redraw to True:
ThisDisplay.Redraw = True

'and finally use refresh:
Call ThisDisplay.RefreshWindow
```

Пример VB:

```
'This routine changes the dimensions of display with following
'display recovery.

'Declare coordinates as long values
 Dim widthDisplay
 Dim heightDisplay

 Dim widthDisplayMem
 Dim heightDisplayMem

'Remember current dimensions:

Call ThisDisplay.GetDisplayDimensions2(widthDisplayMem,heightDisplayMem)
widthDisplayMem = 1036
heightDisplayMem = 767

'Set up new parameters:
widthDisplay = 600
heightDisplay = 400

'Set the display parameters:
Call ThisDisplay.SetDisplayDimensions(widthDisplay, heightDisplay)
'and refresh the window:
ThisDisplay.RefreshWindow

MsgBox "When you click on OK the dimensions will be restored."

'Again, set display parameters:
Call ThisDisplay.SetDisplayDimensions(widthDisplayMem, heightDisplayMem)

'and once more refresh the window:
 ThisDisplay.RefreshWindow
```

SaveAsVBA As Boolean

Сохраняет экран по умолчанию как экран VBA (TRUE) и как экран без VBA (FALSE). Это можно также установить в Закладке **Общие** диалогового окна **Параметры приложения** в поле **Сохранить экран как**...

Примечание: Полный код редактора VBA будет утерян при сохранении экрана как формы без VBA *(SaveAsVBA = Fals*e).

ScaleMode As GWXSCALEMODE

Устанавливает режим масштабирования текущей экранной формы. Допустимые значения:

- Фиксированный масштаб = 0
- Масштабирование разрешено = 1
- Масштабирование с сохранением пропорций = 2

Методы

Методы, доступные только в режиме *Разработка*, не снабжены примерами на Visual Basic, поскольку предполагается что указанные методы будут вызываться только из внешних приложений, разрабатываемых на языке Си.

Sub BringWindowToTop()

Размещает главное окно GraphWorX32 над окнами остальных приложений. (Не выполняется для экранных форм, исполняющихся внутри объектов **GWXview32**).

Пример VBA:

```
'Declare a new GwxDisplay
 Dim newGwxDisplay As New Gwx32.GwxDisplay

'Show the display
 Call newGwxDisplay.ShowWindow

Call newGwxDisplay.BringWindowToTop

MsgBox "This function works properly only" +
 " within a dialog which follows."
```

Пример VB:

```
'Declare a new GwxDisplay
 Dim newGwxDisplay

MsgBox "New window is launched..."

'Assign an object to newGwxDisplay
 Set newGwxDisplay = CreateObject("Gwx32.Display")

'Show the display...
 Call newGwxDisplay.ShowWindow

'and bring it to top.
 Call newGwxDisplay.BringWindowToTop
```

Sub ClearGradientFill()

Отменяет установленную ранее градиентную заливку фона (уст. сплошной фон).

Sub CloseAllPopupWindows()

Закрывает все открытые всплывающие окна. (Не доступен в ActiveX ToolWorX).

Пример VBA:

```
'Close the window from he parent display.
 ThisDisplay.CloseAllPopupWindows
```

Пример VB:

```
'Close the window from he parent display.
ThisDisplay.CloseAllPopupWindows
```

Function ClosePopupWithTitle(titleSubstring As String) As Boolean

Закрывает всплывающее окна, область заголовка которого содержит строку, являющуюся частью заголовка или полным заголовком окна. Возвращает TRUE, если заданное окно успешно закрыто и FALSE, – в противном случае. (Не доступен в ActiveX ToolWorX).

Пример VBA:

```
'Ignore trailing string "GraphWorx32" from Iconics
 ThisDisplay.ClosePopupWithTitle ("PopupWindow")
```

Пример VB:

```
'Ignore trailing string "GraphWorx32" from Iconics
 ThisDisplay.ClosePopupWithTitle ("PopupWindow")
```

Sub CloseWindow()

Закрывает окно данной экранной формы. Данный метод аналогичен ExitApplication, но может выполняться для экранных форм, работающих во всплывающих или встроенных окнах. (Недоступен в ActiveX ToolWorX).

Пример VBA:

```
'Create a new instance from the parent display
 'where you can use this function.
 ThisDisplay.CloseWindow
```

Пример VB:

```
'Create a new instance from the parent display
 'where you can use this function.
 ThisDisplay.CloseWindow
```

Function ConvertGwxSymbolToLibraryObject(gwxSymbol As Object, libraryObject As Object) As Boolean

Преобразует объект **GwxSymbol** в объект **Библиотеки Символов**. Такой объект может быть вставлен в категории Библиотеки Символов с помощью средств Библиотеки Символов. (Дополнительная информация содержится в документации Библиотеки Символов)

Примечание: Данный метод может быть вызван только в режиме Разработка.

Пример VBA:

```
Dim newSym As Object
 Dim libObj As Object

If Not ThisDisplay.InsertLibraryObjectByName("Misc",
 "Check") Then
MsgBox "Error with InsertLibraryObjectByName!"
 End If

Set newSym = ThisDisplay.GetHeadObject

MsgBox newSym.GetObjectTypeName

If Not ThisDisplay.ConvertGwxSymbolToLibraryObject(newSym,
 libObj) Then
MsgBox "Error converting symbol to library object!"
 Else
 MsgBox "GwxSymbol successfully converted
 End If
```

Function CreateAnalogColorDynamic(visibleObjectName As String, dynamicObjectName As String, changeFill As Boolean, changeLine As Boolean, changeShadow As Boolean, startFillColor As OLE_COLOR, endFillColor As OLE_COLOR, startLineColor As OLE_COLOR, endLineColor As OLE_COLOR, startShadowColor As OLE_COLOR, endShadowColor As OLE_COLOR, defaultColorAbove As Boolean, defaultColorBelow As Boolean) As Object)

Создает объект **GwxAnalogColor** (плавное изменение цвета) с заданными атрибутами и связывает его с видимым объектом, чье имя передается в первом аргументе. В случае успеха возвращает вновь созданный объект **GwxAnalogColor**, и NULL – в противном случае. (См. также **GwxAnalogColor** и **GwxDynamic**). Данный метод может быть вызван только в режиме Разработка.

Пример VBA:

```
Dim objRect As GwxRectangle

‘button
 o.Selected = False

Call ThisDisplay.DeleteObject("TheRectangle")
Set objRect = ThisDisplay.CreateRectangle(

442,
213,
80,
60,
True,
RGB(100, 200, 230),
RGB(100, 200, 230),
1,
LineNone,
False,
RGB(192, 192, 192),
EdgeEtched,
False,
"TheRectangle",
True)

If objRect Is Nothing Then
 MsgBox "Operation failed."
 Exit Sub
Else
MsgBox "A new object has been created."
End If

objRect.UserDescription = "The Rectangle"

Dim obj As GwxAnalogColor
ThisDisplay.DeleteDynamic ("ObjAnalogColor")

Set obj = ThisDisplay.CreateAnalogColorDynamic(
"TheRectangle",
"ObjAnalogColor",

True,
True,
RGB(223, 223, 223),
RGB(100, 100, 200),
RGB(223, 223, 223),
RGB(100, 100, 200),
RGB(223, 223, 223),
RGB(10, 10, 200),
True,
True)

If obj Is Nothing Then
MsgBox "Operation failed."
Exit Sub
```

```
Else
MsgBox "A new object has been created."
End If
obj.dataSource = "gfwsim.sine.double"
obj.UserDescription = "GwxAnalogColor"
```

Function CreateAnalogSelector(symbolName As String, dynamicObjectName As String, hiddenWhenAbove As Boolean, hiddenWhenBelow As Boolean) As Object

Создает объект **GwxAnalogIndicator** (аналоговый индикатор) с заданными атрибутами и связывает его с символом, чье имя передается в первом аргументе. В случае успеха возвращает вновь созданный объект **GwxAnalogSelector**, и NULL – в противном случае. (См. также **GwxAnalogSelector**, **GwxSymbol**, **GwxVisible** и **GwxDynamic**). Данный метод может быть вызван только в режиме Разработка.

Пример VBA:

```
'button
o.Selected = False

Dim obj As GwxAnalogSelector

ThisDisplay.DeleteDynamic ("ObjectDyn")

Dim objRect As GwxRectangle
 Set objRect = ThisDisplay.GetVisibleObjectFromName(
                                "Rectangle 0")
objRect.Selected = True

Dim objEll As GwxEllipse
Set objEll = ThisDisplay.GetVisibleObjectFromName(
                                "Ellipse 0")
objEll.Selected = True

Dim objCir As GwxEllipse
Set objCir = ThisDisplay.GetVisibleObjectFromName(
                                "Circle 0")
objCir.Selected = True

'Deselect the button:
o.Selected = False

'Use selected visibles to create a group:
ThisDisplay.CreateSymbol ("Symbol 0")

Set obj = ThisDisplay.CreateAnalogSelector(
"Symbol 0",
"ObjectDyn",
False,
False)

If obj Is Nothing Then

MsgBox "Operation failed."
Exit Sub
Else
MsgBox "A new object has been created."
End If

obj.dataSource = "gfwsim.sine.double"
obj.UserDescription = "GwxAnalogSelector"
```

Function CreateAnimator(symbolName As String, dynamicObjectName As String, animateWhenTrue As Boolean, visibleWhenOff As Boolean, currentFrameWhenOff As Boolean) As Object

Создает объект **GwxAnimator** (анимация) с заданными атрибутами и связывает его с символом, чье имя передается в первом аргументе. В случае успеха возвращает вновь

созданный объект **GwxAnimator**, и NULL – в противном случае. (См. также **GwxAnimator**, **GwxSymbol**, **GwxVisible** и **GwxDynamic**). Данный метод может быть вызван только в режиме Разработка.

Пример VBA:

```
' button o.Selected = False

ThisDisplay.DeleteDynamic ("ObjectDyn")

Dim objRect As GwxRectangle
 Set objRect = ThisDisplay.GetVisibleObjectFromName("Rectangle 0")

objRect.Selected = True
Dim objEll As GwxEllipse
Set objEll = ThisDisplay.GetVisibleObjectFromName("Ellipse 0")

objEll.Selected = True

Dim objCir As GwxEllipse
Set objCir = ThisDisplay.GetVisibleObjectFromName("Circle 0")

objCir.Selected = True

'Deselect the button:
o.Selected = False

'Use selected visibles to create a group:
ThisDisplay.CreateSymbol ("Symbol 0")

Dim obj As GwxAnimator
Set obj = ThisDisplay.CreateAnimator(
                                     "Symbol 0",
                                     "ObjectDyn",
                                     True,
                                     True,
                                     False)
If obj Is Nothing Then
MsgBox "Operation failed."
Exit Sub
Else
MsgBox "A new object has been created." End If
obj.dataSource = "gfwsim.sine.double"
obj.UserDescription = "GwxAnimator"
```

Function CreateArc(arcType As GWXARCTYPE, centerX As Single, centerY As Single, radiusX As Single, radiusY As Single, startAngle As Single, endAngle As Single, isFilled As Boolean, fillColor As OLE_COLOR, lineColor As OLE_COLOR, lineWidth As Long, lineStyle As GWXLINESTYLE, hasShadow As Boolean, shadowColor As OLE_COLOR, edgeStyle As GWX3DEDGESTYLE, isHidden As Boolean, objectName As String) As Object

Создает и возвращает объект типа **GwxArc** (дуга) с заданными атрибутами. Значения углов (float startAngle, float endAngle) представляются в градусах. (См. также **GwxArc** и **GwxVisible**). Данный метод может быть вызван только в режиме Разработка.

Пример VBA:

```
‘ deselect button
o.Selected = False
ThisDisplay.DeleteObject ("TheArc")

Dim obj As GwxArc
Set obj = ThisDisplay.CreateArc(
                               ArcPie,
                               470, _
                               242, _
                               30, _
                               40, _
```

```
                                    270,
                                    45,
                                    True,
                                    RGB(198, 255, 190),
                                    RGB(10, 100, 200),
                                    2,
                                    LineSolid, _
                                    False, _
                                    RGB(50, 50, 50), _
                                    EdgeEtched, _
                                    False,
                                    "TheArc")
If obj Is Nothing Then
MsgBox "Operation failed."
Exit Sub
Else
MsgBox "A new object has been created."
End If
obj.UserDescription = "The Arc"
```

Function CreateBitmap(filename As String, left As Single, top As Single, width As Single, height As Single, isTransparent As Boolean, transparentColor As Long, lineColor As Long, lineWidth As Long, lineStyle As GWXLINESTYLE, hasShadow As Boolean, shadowColor As Long, edgeStyle As GWX3DEDGESTYLE, isHidden As Boolean, objectName As String) As Object)

Создает и возвращает объект типа **GwxBitmap** (*растровое изображение*) с заданными атрибутами в текущей экранной форме. Аргумент **fileName** должен содержать имя файла импортируемого растрового изображения. (См. также **GwxVisible**). Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
'deselect button
o.Selected = False
ThisDisplay.DeleteObject ("ThePicture")
Dim obj As GwxBitmap
Set obj = ThisDisplay.CreateBitmap(
"meter.bmp", _
700, _
127,
125,
False, _
RGB(192, 192, 192),
RGB(192, 192, 192),
1,
LineNone,
False, _
RGB(192, 192, 192), _
EdgeEtched, _
False, _
"ThePicture")
If obj Is Nothing Then
MsgBox "Operation failed."
Exit Sub
Else
MsgBox "A new object has been created."
End If
obj.UserDescription = "The Picture"
```

Function CreateButton(buttonType As GWXBUTTONTYPE, x As Single, y As Single, label As String, alignment As GWXTEXTALIGNMENT, stretchText As Boolean, isFilled As Boolean, fillColor As OLE_COLOR, lineColor As OLE_COLOR, lineWidth As Long, lineStyle As Long, hasShadow As Boolean, shadowColor As OLE_COLOR, edgeStyle As Long, isHidden As Boolean, objectName As String) As Object

Создает объект типа **GwxButton** (*кнопка*) с заданными атрибутами в текущей экранной форме. Созданная таким образом кнопка не будет выполнять какие-либо операции при

нажатии до тех пор, пока не будет выполнено соединение с объектом **GwxPick** (*указание и щелчок*). (См. также **GwxButton**, **GwxPick**, **GwxDynamic**, **GwxText** и **GwxVisible**). Данный метод может быть вызван только в режиме *Разработка*.

Function CreateDigitalColorDynamic(visibleObjectName As String, dynamicObjectName As String, changeColorWhenTrue As Boolean, changeFill As Boolean, changeLine As Boolean, changeShadow As Boolean, fillColor As OLE_COLOR, lineColor As OLE_COLOR, shadowColor As OLE_COLOR, dataSource As String) As Object

Создает объект **GwxDigitalColor** (*дискретное изменение цвета*) с заданными атрибутами и связывает его с видимым объектом, чье имя передается в первом аргументе. Объект создается с одним соединением с источником данных. Дополнительные соединения создаются путем повторных вызовов данного метода. В случае успеха возвращает вновь созданный объект **GwxDigitalColor**, и NULL – в противном случае. (См. **GwxDigitalColor**, **GwxDigitalColorInfo** и **GwxDynamic**). Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
Dim objRect As GwxRectangle
‘button
o.Selected = False
Call ThisDisplay.DeleteObject("TheRectangle")
Set objRect = ThisDisplay.CreateRectangle( _
442, _
213, _
80, _
60,
True,
RGB(100, 200, 230),
RGB(100, 200, 230),
1,
LineNone,
False, _
RGB(192, 192, 192), _
EdgeEtched, _
False, _
"TheRectangle",
True)
If objRect Is Nothing Then
MsgBox "Operation failed."
Exit Sub
Else
MsgBox "A new object has been created."
End If
objRect.UserDescription = "The Rectangle"
ThisDisplay.DeleteDynamic ("ObjDyn")
Call ThisDisplay.CreateDigitalColorDynamic(
"TheRectangle",
"ObjDyn",
True,
True,
False, _
False, _
RGB(213, 200, 200), _
RGB(213, 120, 120), _
RGB(213, 120, 120), _
"gfwsim.toggle.bool")
```

Function CreateDigitalSelector(symbolName As String, dynamicObjectName As String) As Object

Создает объект **GwxDigitalSelector** (*цифровой индикатор*) с заданными атрибутами и связывает его с символом, чье имя передается в первом аргументе. В случае успеха возвращает вновь созданный объект **GwxDigitalSelector**, и NULL – в противном случае. Вновь созданный объект не имеет соединений с источниками данных. Для установления соединений с источниками данных используйте метод **GwxDigitalSelector::SetConnectionInfo**. (См. также **GwxDigitalSelector**,

GwxDigitalSelectorInfo, **GwxSymbol**, **GwxVisible** и **GwxDynamic**). Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
Selected = False
Dim objRect As GwxRectangle
ThisDisplay.UngroupSymbol ("Symbol_0")
Set objRect = ThisDisplay.GetVisibleObjectFromName( _
"Rectangle 0")
objRect.Selected = True
Dim objEll As GwxEllipse
Set objEll = ThisDisplay.GetVisibleObjectFromName( _
"Ellipse_0")
objEll.Selected = True
Dim objCir As GwxEllipse
Set objCir = ThisDisplay.GetVisibleObjectFromName(
"Circle_0")
objCir.Selected = True
'Deselect the button:
o.Selected = False
'Use selected visibles to create a group:
ThisDisplay.DeleteDynamic ("ObjectDyn")
Dim objDyn As GwxDigitalSelector
Set objDyn = ThisDisplay.CreateDigitalSelector( _
"Symbol_0", _
"ObjectDyn")
If objRect Is Nothing Then
MsgBox "No dynamic object has been created."
Exit Sub
Else
MsgBox "A new dynamic object has been created."
End If
objDyn.UserDescription = "The Digital Selector"
 'Call objDyn.SetConnectionInfo(0, "gfwsim.sine.double", True)
Call objDyn.SetConnectionInfo(1, "gfwsim.cosine.double", False)
Call objDyn.SetConnectionInfo(2, "gfwsim.toggle.bool", True)
```

Function CreateEllipse(left As Single, top As Single, width As Single, height As Single, isFilled As Boolean, fillColor As OLE_COLOR, lineColor As OLE_COLOR, lineWidth As Long, lineStyle As GWXLINESTYLE, hasShadow As Boolean, shadowColor As OLE_COLOR, edgeStyle As GWX3DEDGESTYLE, isHidden As Boolean, objectName As String) As Object

Создает объект типа **GwxEllipse** (*эллипс*) с заданными атрибутами в текущей экранной форме. (См. также **GwxEllipse** и **GwxVisible**). Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
Dim objEll As GwxEllipse
ThisDisplay.DeleteObject ("TheEllipse")
’deselect button
o.Selected = False
Set objEll = ThisDisplay.CreateEllipse( _
442,
230,
80,
30,
True,
RGB(123, 200, 123),
RGB(123, 200, 123),
1, _
LineNone, _
False, _
RGB(123, 200, 123),
EdgeEtched,
False,
"TheEllipse")
```

```
If objEll Is Nothing Then
MsgBox "Operation failed." E
xit Sub
Else
MsgBox "A new object has been created."
End If
objEll.UserDescription = "The Ellipse"
```

Function CreateFlashDynamic(visibleObjectName As String, dynamicObjectName As String, hideObject As Boolean, flashWhenTrue As Boolean, altStateWhenOff As Boolean, changeFill As Boolean, changeLine As Boolean, changeShadow As Boolean, altFillColor As OLE_COLOR, altLineColor As OLE_COLOR, altShadowColor As OLE_COLOR) As Object

Создает объект **GwxFlash** (*прерывистое отображение*) с заданными атрибутами и связывает его с видимым объектом, чье имя передается в первом аргументе. В случае успеха возвращает вновь созданный объект **GwxFlash**, и NULL – в противном случае. (См. **GwxFlash** и **GwxDynamic**). Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
    Dim objRect As GwxRectangle
    'deselect button
    o.Selected = False

    Call ThisDisplay.DeleteObject("TheRectangle")
    Set objRect = ThisDisplay.CreateRectangle( _
                                442, _
                                213, _
                                80, _
                                60, _
                                True, _
                                RGB(100, 200, 230), _
                                RGB(100, 200, 230), _
                                1, _
                                LineNone, _
                                False, _
                                RGB(192, 192, 192), _
                                EdgeEtched, _
                                False, _
                                "TheRectangle", _
                                True)
    If objRect Is Nothing Then
        MsgBox "Operation failed."
        Exit Sub
    Else
        MsgBox "A new object has been created."

    End If

    objRect.UserDescription = "The Rectangle"

    ThisDisplay.DeleteDynamic ("TheFlashObject")
    Dim obj As GwxFlash

    Set obj = ThisDisplay.CreateFlashDynamic( _
                            "TheRectangle", _
                            "TheFlashObject", _
                            False, _
                            True, _
                            False, _
                            True, _
                            True, _

                            True, _
                            RGB(230, 230, 100), _
                            RGB(100, 100, 230), _
                            RGB(100, 100, 100))
```

```
If obj Is Nothing Then
    MsgBox "No dynamic object has been created."
    Exit Sub
Else
    MsgBox "A new dynamic object has been created."
End If

obj.UserDescription = "The Flash Object"
obj.dataSource = "gfwsim.toggle.bool"
```

Function CreateHideDynamic(visibleObjectName As String, dynamicObjectName As String, hideWhenTrue As Boolean, disableObject As Boolean) As Object)

Создает объект **GwxHide** (*скрыть/блокировать*) с заданными атрибутами и связывает его с видимым объектом, чье имя передается в первом аргументе. В случае успеха возвращает вновь созданный объект **GwxHide**, и NULL – в противном случае. (См. **GwxHide** и **GwxDynamic**). Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
Dim objRect As GwxRectangle

o.Selected = False

Call ThisDisplay.DeleteObject("TheRectangle")
Set objRect = ThisDisplay.CreateRectangle( _
                            442, _
                            213, _
                            80, _
                            60, _
                            True, _
                            RGB(100, 200, 230), _
                            RGB(100, 200, 230), _
                            1, _
                            LineNone, _
                            False, _
                            RGB(192, 192, 192), _
                            EdgeEtched, _
                            False, _
                            "TheRectangle", _
                            True)
If objRect Is Nothing Then
    MsgBox "Operation failed."
    Exit Sub
Else
    MsgBox "A new object has been created."

End If

objRect.UserDescription = "The Rectangle"

ThisDisplay.DeleteDynamic ("TheHideObject")

Dim obj As GwxHide
Set obj = ThisDisplay.CreateHideDynamic( _
                      "TheRectangle", _
                      "TheHideObject", _
                      True, _
                      False)

If obj Is Nothing Then
    MsgBox "No dynamic object has been created."
    Exit Sub

Else
    MsgBox "A new dynamic object has been created."
End If
```

```
obj.UserDescription = "The Hide Object"
obj.dataSource = "gfwsim.toggle.bool"
```

Sub CreateLayer(layerName As String)

Создает новый слой и присваивает ему имя, определенное параметром)

Пример VBA:

```
'If any exists...
ThisDisplay.RemoveLayer ("Layer9999")

ThisDisplay.CreateLayer ("Layer9999")
MsgBox "A new layer has been created."

MsgBox "On the Format menu, click Layers -> Remove" + _
       " to remove the layer Layer9999"
```

Function CreateLocationDynamic(visibleObjectName As String, dynamicObjectName As String, offsetX As Single, offsetY As Single, slider As Boolean, tracking As Boolean, numberOfDetents As Integer, continuousUpdate As Boolean) As Object)

Создает объект типа **GwxLocation** (*изменение положения*) с заданными атрибутами и связывает его с видимым объектом, чье имя передается в первом аргументе. Аргументы **offsetX** и **offsetY** определяют значения сдвига объекта по горизонтальной и вертикальной осям относительно первоначального положения. В случае успеха возвращает вновь созданный объект **GwxLocation**, и NULL – в противном случае. (См. также **GwxLocation** и **GwxDynamic**). Данный метод может быть вызван только в режиме *Разработка*.

ПримерVBA example:

```
Dim objRect As GwxRectangle

o.Selected = False

Call ThisDisplay.DeleteObject("TheRectangle")
Set objRect = ThisDisplay.CreateRectangle( _
                            442, _
                            213, _
                            80, _
                            60, _
                            True, _
                            RGB(100, 200, 230), _
                            RGB(100, 200, 230), _
                            1, _
                            LineNone, _
                            False, _
                            RGB(192, 192, 192), _
                            EdgeEtched, _
                            False, _
                            "TheRectangle", _
                            True)
If objRect Is Nothing Then
    MsgBox "Operation failed."
    Exit Sub
Else
    MsgBox "A new object has been created."

End If

objRect.UserDescription = "The Rectangle"

'If any exists ...
ThisDisplay.DeleteDynamic ("TheLocationDynamicObject")

Dim obj As GwxLocation
Set obj = ThisDisplay.CreateLocationDynamic( _
                      "TheRectangle", _
```

```
                        "TheLocationDynamicObject", _
                        50, _
                        50, _
                        True, _
                        True, _
                        0, _
                        True)

If obj Is Nothing Then
    MsgBox "No dynamic object has been created."
    Exit Sub
Else
    MsgBox "A new dynamic object has been created."
End If

obj.UserDescription = "The Location Dynamic Object"
obj.dataSource = "gfwsim.cosine.double"
```

Function CreateMetafile(filename As String, left As Single, top As Single, width As Single, height As Single, objectName As String) As Object

Создает объект типа **GwxMetafile** (*векторное изображение*) с заданными атрибутами в текущей экранной форме. Аргумент **fileName** должен содержать имя файла импортируемого векторного изображения. (См. также **GwxVisible**). Данный метод может быть вызван только в режиме *Разработка*.
VBA example:

```
o.Selected = False
ThisDisplay.DeleteObject ("ThePicture")

Dim objPic As GwxMetafile
Set objPic = ThisDisplay.CreateMetafile( _
                        "valve.emf", _
                        700, _
                        125, _
                        90, _
                        120, _
                        "ThePicture")
If objPic Is Nothing Then
    MsgBox "Operation failed."
    Exit Sub
Else

    MsgBox "A new object has been created."
End If

objPic.UserDescription = "The Picture"
```

Function CreatePickDynamic(visibleObjectName As String, dynamicObjectName As String, pickAction As GWXPICKACTION, pickType As GWXBUTTONTYPE, executionTrigger As GWXEXECUTIONTRIGGER, mouseButton As GWXMOUSEBUTTON, initiallySelected As Boolean, groupName As String, filename As String, modal As Boolean, center As Boolean, value1 As String, value2 As String, value3 As String) As Object

Создает объект типа **GwxPick** (указание и щелчок) с заданными атрибутами и связывает его с видимым объектом, чье имя передается в первом аргументе.

- PickLoadDisplay (загрузка экранной формы) = 0
- PickDragDropLoad (загрузка методом Drag-and-Drop) = 1
- PickPopupWindow (всплывающее окно) = 2
- PickDownloadValue (передача значения) = 3
- PickToggleValue (переключение значения) = 4

- PickLaunchApp (запуск приложения) = 5
- PickClose (закрытие окна) = 6
- PickRunScript (запуск VBA-сценария) = 7
- PickCustomCommand (специальная команда) = 8
- PickEmbeddedWindow (встроенное окно) = 9
- PickDisplayBack (предыдущая экранная форма) = 10
- PickDisplayForward (следующая экранная форма) = 11
- PickSetAliases (настройка псевдонимов) = 12
- PickAliasesDialog (диалог настройки псевдонимов) = 13
- PickLayerHideOn (скрыть слой) = 14
- PickLayerHideOff (показать слой) = 15
- PickLayerHideToggle (переключить отобр. слоя) = 16
- PickPopupUserMenu (польз. меню) = 17
- PickSwitchLanguage (выбор языка) = 18
- PickSelectThema (выбор темы) = 19
- PickLaunchWebPage (открыть Web-страницу) = 20

Допустимые значения аргумента **GWXBUTTONTYPE** (**Тип кнопки**):
ButtonNormal (кнопка без фиксации) = 0
ButtonCheck (кнопка с фиксацией) = 1
ButtonRadio (зависимая кнопка) = 2
Допустимые значения аргумента **GWXEXECUTIONTRIGGER** (**Порядок выполнения**):

- TriggerOnDown (если нажато) = 1
- TriggerWhileDown (пока нажато) = 2
- TriggerOnDnWhileDn (если нажато или пока нажато) = 3
- TriggerOnUp (если отпущено) = 4
- TriggerOnDnOnUp (если нажато или отпущено) = 5
- TriggerWhileDnOnUp (пока нажато или отпущено) = 6
- TriggerOnDnWhileDnOnUp
  (если нажато или отпущено или пока нажато) = 7

Допустимые значения аргумента **GWXMOUSEBUTTON** (**Клавиша мыши**):

- MouseButtonLeft (левая) = 0
- MouseButtonMiddle (средняя) = 1
- MouseButtonRight (правая) = 2

В зависимости от значения **pickAction**, некоторые параметры могут не приниматься во внимание GraphWorX32. В случае успеха возвращает вновь созданный объект типа **GwxPick**, и NULL, – в противном случае. (См. также **GwxPick** и **GwxDynamic**). Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
o.Selected = False
ThisDisplay.DeleteObject ("Button_0")

Dim obj As GwxButton
Set obj = ThisDisplay.CreateButton( _
                    ButtonNormal, _
                    440, _
                    250, _
                    "The Button", _
                    TextAlignCenter, _
                    False, _
                    True, _
                    RGB(120, 220, 120), _
                    RGB(0, 0, 120), _
```

```
                        1, _
                        EdgeNone, _
                        False, _
                        RGB(100, 100, 100), _
                        EdgeNone, _
                        False, _
                        "Button_0")

If obj Is Nothing Then
    MsgBox "Operation failed."
    Exit Sub
Else
    MsgBox "A new object has been created."
End If

obj.UserDescription = "The Button"


Call ThisDisplay.CreatePickDynamic("Button_0", _
                                   "PickDyn_0", _
                                   PickRunScript, _
                                   ButtonNormal, _
                                   TriggerWhileDown, _
                                   MouseButtonLeft, _
                                   False, _
                                   "", _
                                   "", _
                                   False, _

                                   True, _
                                   "", _
                                   "", _
                                   "")
```

Function CreatePolyline(vertices, isFilled As Boolean, fillColor As OLE_COLOR, lineColor As OLE_COLOR, lineWidth As Long, lineStyle As GWXLINESTYLE, hasShadow As Boolean, shadowColor As OLE_COLOR, edgeStyle As GWX3DEDGESTYLE, isHidden As Boolean, objectName As String) As Object

Создает и возвращает объект типа **GwxLine** (*линия*) с заданными атрибутами в текущей экранной форме. Аргумент **vertices** является массивом действительных значений. Элементы массива с четными индексами (0, 2, 4, 6, ...) содержат значения координат вершин ломаной линии по горизонтальной оси, а с нечетными (1, 3, 5, 7, ...) – по вертикальной. (См. также **GwxLine** и **GwxVisible**). При вызове метода из приложения на C++ аргумент **vertices** должен быть массивом, нижняя граница которого задана равной 1. Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
   'If any exists
ThisDisplay.DeleteObject ("ThePolyLineObject")
o.Selected = False
Dim objPol As GwxLine

Dim lVertices(8) As Double

lVertices(1) = 683     'x
   lVertices(2) = 149 'y
lVertices(3) = 911     'x
   lVertices(4) = 239 'y
lVertices(5) = 708     'x
   lVertices(6) = 269 'y
```

```
lVertices(7) = 876     'x
   lVertices(8) = 274 'y

Set objPol = ThisDisplay.CreatePolyline( _
                lVertices, _
                True, _
                RGB(100, 200, 255), _
                RGB(100, 200, 255), _
                1, _
                LineNone, _
                False, _
                RGB(0, 0, 0), _
                EdgeEtched, _
                False, _
                "ThePolyLineObject")

If objPol Is Nothing Then
   MsgBox "Operation failed."
   Exit Sub
Else
   MsgBox "A new object has been created."
End If

objPol.UserDescription = "The PolyLine"
```

Function CreateProcessPoint(textObjectName As String, dynamicObjectName As String, dataType As GWXDATATYPE, update As Boolean, dataEntry As Boolean, hasInitialValue As Boolean, initialValue, format As String) As Object

Создает объект типа **GwxProcessPoint** (*параметр процесса*) с заданными атрибутами в текущей экранной форме и связывает его с объектом типа **GwxText**, имя которого передается через первый аргумент. В случае успеха возвращает вновь созданный объект типа **GwxProcessPoint**, и NULL, – в противном случае. (См. также **GwxProcessPoint**, **GwxText**, **GwxVisible** и **GwxDynamic**). Данный метод может быть вызван только в режиме *Разработка*.

VBA example:

```
o.Selected = False

ThisDisplay.DeleteObject ("TheText")


Dim objText As GwxText
Set objText = ThisDisplay.CreateText( _
                           448, _
                           232, _
                           "The Text", _
                           TextAlignCenter, _
                           False, _
                           True, _
                           RGB(200, 200, 230), _
                           RGB(10, 10, 50), _

                           1, _
                           LineNone, _
                           False, _
                           RGB(0, 0, 0), _
                           EdgeEtched, _
                           False, _
                           "TheText")
```

```
If objText Is Nothing Then
    MsgBox "Operation failed."
    Exit Sub
Else
    MsgBox "A new text object has been created."
End If

objText.UserDescription = "The Text"



ThisDisplay.DeleteDynamic ("TheProcessPointObject")

Dim obj As GwxProcessPoint
Set obj = ThisDisplay.CreateProcessPoint( _
                      "TheText", _
                      "TheProcessPointObject", _
                      DataTypeDouble, _
                      True, _
                      False, _
                      True, _
                      0, _

                      "x.xe")

ThisDisplay.RefreshWindow

If obj Is Nothing Then
    MsgBox "No dynamic object has been created."
    Exit Sub
Else
    MsgBox "A new dynamic object has been created."
End If

obj.UserDescription = "The Process Point"
obj.dataSource = "gfwsim.cosine.double"
```

Function CreateRectangle(left As Single, top As Single, width As Single, height As Single, isFilled As Boolean, fillColor As OLE_COLOR, lineColor As OLE_COLOR, lineWidth As Long, lineStyle As GWXLINESTYLE, hasShadow As Boolean, shadowColor As OLE_COLOR, edgeStyle As GWX3DEDGESTYLE, isHidden As Boolean, objectName As String, rounded As Boolean) As Object

Создает объект типа **GwxRectangle** (*прямоугольник*) с заданными атрибутами в текущей экранной форме. (См. также **GwxRectangle** и **GwxVisible**). Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
Dim objRect As GwxRectangle

'button
o.Selected = False

Call ThisDisplay.DeleteObject("TheRectangle")
Set objRect = ThisDisplay.CreateRectangle( _
                          442, _
                          213, _
                          80, _
                          60, _
                          True, _
                          RGB(100, 200, 230), _
                          RGB(100, 200, 230), _

                          1, _
                          LineNone, _
                          False, _
                          RGB(192, 192, 192), _
```

```
                                EdgeEtched, _
                                False, _
                                "TheRectangle", _
                                True)
If objRect Is Nothing Then
    MsgBox "Operation failed."
    Exit Sub
Else
    MsgBox "A new object has been created."

End If

objRect.UserDescription = "The Rectangle"


Dim obj As GwxAnalogColor

ThisDisplay.DeleteDynamic ("ObjAnalogColor")

Set obj = ThisDisplay.CreateAnalogColorDynamic( _
                    "TheRectangle", _
                    "ObjAnalogColor", _
                    True, _
                    True, _
                    True, _
                    RGB(223, 223, 223), _
                    RGB(100, 100, 200), _

                    RGB(223, 223, 223), _
                    RGB(100, 100, 200), _
                    RGB(223, 223, 223), _
                    RGB(10, 10, 200), _
                    True, _
                    True)

If obj Is Nothing Then
    MsgBox "Operation failed."
    Exit Sub
Else
    MsgBox "A new object has been created."
End If

obj.dataSource = "gfwsim.sine.double"

obj.UserDescription = "GwxAnalogColor"
```

Function CreateRotationDynamic(visibleObjectName As String, dynamicObjectName As String, startAngle As Single, endAngle As Single, pivotX As Single, pivotY As Single, clockwise As Boolean, dial As Boolean, tracking As Boolean, numberOfDetents As Integer, continuousUpdate As Boolean) As Object)

Создает объект типа **GwxRotation** (вращение) с заданными атрибутами и связывает его с видимым объектом, чье имя передается в первом аргументе. Аргументы **pivotX** и **pivotY** определяют значения сдвига объекта по горизонтальной и вертикальной осям относительно центра прямоугольной области, в которую вписан весь объект. Значения углов передаются в градусах. В случае успеха возвращает вновь созданный объект **GwxRotation**, и NULL – в противном случае. (См. также **GwxRotation** и **GwxDynamic**). Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
Dim objRect As GwxRectangle

o.Selected = False

Call ThisDisplay.DeleteObject("TheRectangle")
Set objRect = ThisDisplay.CreateRectangle( _
                                442, _
                                213, _
```

```
                                80, _
                                60, _
                                True, _
                                RGB(100, 200, 230), _
                                RGB(100, 200, 230), _

                                1, _
                                LineNone, _
                                False, _
                                RGB(192, 192, 192), _
                                EdgeEtched, _
                                False, _
                                "TheRectangle", _
                                True)
If objRect Is Nothing Then
    MsgBox "Operation failed."
    Exit Sub
Else
    MsgBox "A new object has been created."

End If

objRect.UserDescription = "The Rectangle"

ThisDisplay.DeleteDynamic ("TheRotationDynamic")

Dim obj As GwxRotation
Set obj = ThisDisplay.CreateRotationDynamic( _
                        "TheRectangle", _
                        "TheRotationDynamic", _
                        30, _
                        270, _
                        20, _
                        30, _
                        True, _

                        True, _
                        True, _
                        0, _
                        True)

If obj Is Nothing Then
    MsgBox "No dynamic object has been created."
    Exit Sub
Else
    MsgBox "A new dynamic object has been created."
End If

obj.UserDescription = "The Rotation Dynamic Object"
obj.dataSource = "gfwsim.cosine.double"
'obj.RangeOverride = True


'Dim objPoint As GwxPoint
'Set objPoint = obj.GetDataSourcePointObject
'objPoint.highRange = 1
'objPoint.lowRange = -1
```

Function CreateSizeDynamic(visibleObjectName As String, dynamicObjectName As String, sizeType As GWXSIZEDYNTYPE, clip As Boolean, startSize As Single, endSize As Single) As Object

Создает объект типа **GwxSize** (*изменение размера*) с заданными атрибутами и связывает его с видимым объектом, чье имя передается в первом аргументе. Аргументы **startSize** и **endSize** представляются значения в процентах от диапазона от 0.0 до 1.0.

Допустимые значения аргумента **GWXSIZEDYNTYPE** (тип изменения размера):

SizeLeft	Изменение размера вдоль горизонтальной оси. Начальному значению размера соответствует точка объекта, расположенная правее остальных.	= 0
SizeRight	Изменение размера вдоль горизонтальной оси. Начальному значению размера соответствует точка объекта, расположенная левее остальных.	= 1
SizeUp	Изменение размера вдоль вертикальной оси. Начальному значению размера соответствует самая нижняя точка объекта.	= 2
SizeDown	Изменение размера вдоль вертикальной оси. Начальному значению размера соответствует самая верхняя точка объекта.	= 3
SizeUpLeft	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины и высоты справа налево и снизу вверх соответственно.	= 4
SizeUpRight	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины и высоты слева направо и снизу вверх соответственно.	= 5
SizeDownLeft	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины и высоты справа налево и сверху вниз соответственно.	= 6
SizeDownRight	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины и высоты слева направо и сверху вниз соответственно.	= 7
SizeLeftRight	Изменение размера вдоль горизонтальной оси. Начальному значению размера соответствует горизонтальная координата экранной формы, расположенная в центре воображаемого прямоугольника, в который вписывается объект..	= 8
SizeUpDown	Изменение размера вдоль вертикальной оси. Начальному значению размера соответствует вертикальная координата экранной формы, расположенная в центре воображаемого прямоугольника, в который вписывается объект..	= 9
SizeLeftRightBias	Изменение размера вдоль горизонтальной оси. При выполнении указанного динамического действия объект делится на две части воображаемой прямой линией, проведенной параллельно вертикальной оси экранной формы через точку, расположенную в центре воображаемого прямоугольника, в который вписан объект. Увеличение значения связанного источника данных сначала приводит к уменьшению горизонтального размера (ширины) объекта слева до координаты, расположенной в центре воображаемого прямоугольника, а затем, к увеличению вертикального размера от центра вправо.	= 10
SizeUpDownBias	Изменение размера вдоль вертикальной оси. При выполнении указанного динамического действия объект делится на две части воображаемой прямой линией, проведенной параллельно горизонтальной оси экранной формы через точку, расположенную в центре воображаемого прямоугольника, в который вписан объект. Увеличение значения связанного источника данных сначала приводит к уменьшению вертикального размера (высоты) объекта от самой нижней точки до координаты, расположенной в центре воображаемого прямоугольника, а затем, к увеличению вертикального размера от центра до самой верхней точки.	= 11

SizeAllFour	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины в обоих направлениях и высоты снизу вверх соответственно.	= 12
SizeLeftRightUp	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины и высоты справа налево и снизу вверх соответственно.	= 13
SizeLeftRightDown	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины и высоты справа налево и сверху вниз соответственно.	= 14
SizeUpDownLeft	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины справа налево и высоты сверху в обоих направлениях соответственно.	= 15
SizeUpDownRight	Одновременное изменение высоты и ширины объекта. Увеличение значения связанного источника данных приводит к увеличению ширины слева направо и высоты сверху в обоих направлениях соответственно.	= 16

В случае успеха возвращает вновь созданный объект **GwxSize**, и NULL – в противном случае. (См. также **GwxSize** и **GwxDynamic**). Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
Dim objRect As GwxRectangle

o.Selected = False

Call ThisDisplay.DeleteObject("TheRectangle")
Set objRect = ThisDisplay.CreateRectangle( _
                            442, _
                            213, _
                            80, _
                            60, _
                            True, _
                            RGB(100, 200, 230), _
                            RGB(100, 200, 230), _
                            1, _
                            LineNone, _
                            False, _
                            RGB(192, 192, 192), _
                            EdgeEtched, _
                            False, _
                            "TheRectangle", _
                            True)
If objRect Is Nothing Then
    MsgBox "Operation failed."
    Exit Sub
Else
    MsgBox "A new object has been created."

End If

objRect.UserDescription = "The Rectangle"

ThisDisplay.DeleteDynamic ("TheSizeDynamic")

Dim obj As GwxSize
Set obj = ThisDisplay.CreateSizeDynamic( _
                        "TheRectangle", _
                        "TheSizeDynamic", _
                        SizeUpDown, _
                        True, _
```

```
                        -1, _
                        1)

If obj Is Nothing Then

    MsgBox "No dynamic object has been created."
    Exit Sub
Else
    MsgBox "A new dynamic object has been created."
End If

obj.UserDescription = "The Rotation Dynamic Object"
obj.dataSource = "gfwsim.cosine.double"
```

Function CreateStateField(textObjectName As String, dynamicObjectName As String, update As Boolean, dataEntry As Boolean, hasInitialValue As Boolean, initialValue, stateConfiguration As String, defaultState As String) As Object

Создает и возвращает объект типа *Поле состояния* на основе объекта *Параметр процесса*.
Параметр **stateConfiguration** имеет следующий формат:
Значение1<TAB>Строка1<CR-LF>Значение2<TAB>Строка2<CR-LF> … итд.
Параметр **defaultState** – строка, отображаемая если не обнаружено никакого состояния.
Доп. информация в описании **CreateProcessPoint()**. Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
o.Selected = False
'if any exists...

ThisDisplay.DeleteObject ("TheText")

Dim objText As GwxText
Set objText = ThisDisplay.CreateText( _
                        448, _
                        232, _
                        "The Text", _
                        TextAlignCenter, _
                        False, _
                        True, _
                        RGB(200, 200, 230), _

                        RGB(10, 10, 50), _
                        1, _
                        LineNone, _
                        False, _
                        RGB(0, 0, 0), _
                        EdgeEtched, _
                        False, _
                        "TheText")
If objText Is Nothing Then
    MsgBox "Operation failed."
    Exit Sub
Else
    MsgBox "A new text object has been created."

End If

objText.UserDescription = "The Text"


ThisDisplay.DeleteDynamic ("TheStateField")

Dim obj As Object

Dim strStateConfiguration As String

strStateConfiguration = _
```

```
        "0" + Chr(9) + "Line1" + Chr(13) + _
        "1" + Chr(9) + "Line2"

Set obj = ThisDisplay.CreateStateField( _
                    "TheText", _
                    "TheStateField", _
                    True, _

                    False, _
                    True, _
                    0, _
                    strStateConfiguration, _
                    "<#Building#>")
If obj Is Nothing Then
    MsgBox "No dynamic object has been created."
    Exit Sub
Else
    MsgBox "A new dynamic object has been created."
End If


obj.dataSource = "gfwsim.toggle.bool"
```

Function CreateSymbol(objectName As String) As Object

Группирует все выделенные графические объекты в объект типа **GwxSymbol** (*символ*), которому присваивается имя, передаваемое аргументом метода. Возвращает полученный в результате группировки объект типа **GwxSymbol**. (См. также **GwxSymbol** и **GwxVisible**). Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
o.Selected = False

Dim objRect As GwxRectangle

ThisDisplay.UngroupSymbol ("Symbol_0")
ThisDisplay.DeselectAllObjects

Set objRect = ThisDisplay.GetVisibleObjectFromName( _
                        "Rectangle_0")
MsgBox "Select the Rectangle_0."
objRect.Selected = True


Dim objEll As GwxEllipse
Set objEll = ThisDisplay.GetVisibleObjectFromName( _
                        "Ellipse_0")
MsgBox "Select the Ellipse_0."

objEll.Selected = True

Dim objCir As GwxEllipse
Set objCir = ThisDisplay.GetVisibleObjectFromName( _
                        "Circle_0")
MsgBox "Select the Circle_0."
objCir.Selected = True

'Deselect the button:
o.Selected = False

MsgBox "And end up create a symbol."
'Use selected visibles to create a group:
ThisDisplay.CreateSymbol ("Symbol_0")
```

Function CreateText(x As Single, y As Single, text As String, alignment As GWXTEXTALIGNMENT, stretchText As Boolean, isFilled As Boolean, fillColor As OLE_COLOR, lineColor As OLE_COLOR, lineWidth As Long, lineStyle As GWXLINESTYLE, hasShadow As Boolean, shadowColor As OLE_COLOR, edgeStyle As GWX3DEDGESTYLE, isHidden As Boolean, objectName As String) As Object

Создает и возвращает объект типа **GwxText** (*текст*) с заданными атрибутами в текущей экранной форме. (См. также **GwxText** и **GwxVisible**). Данный метод может быть вызван только в режиме *Разработка*.

VBA example:

```
o.Selected = False
'if any exists...

ThisDisplay.DeleteObject ("TheText")


Dim objText As GwxText
Set objText = ThisDisplay.CreateText( _
                          448, _
                          232, _
                          "The Text", _
                          TextAlignCenter, _
                          False, _
                          True, _
                          RGB(200, 200, 230), _

                          RGB(10, 10, 50), _
                          1, _
                          LineNone, _
                          False, _
                          RGB(0, 0, 0), _
                          EdgeEtched, _
                          False, _
                          "TheText")
If objText Is Nothing Then
    MsgBox "Operation failed."
    Exit Sub
Else
    MsgBox "A new text object has been created."

End If

objText.UserDescription = "The Text"
```

Function CreateTimedate(textObjectName As String, dynamicObjectName As String, formatType As GWXTIMEDATEFORMATTYPE, timeFormat As String, dateFormat As String) As Object

Создает объект типа **GwxTimedate** (*Дата/время*) с заданными атрибутами в текущей экранной форме и связывает его с объектом типа **GwxText**, чье имя передается в качестве первого аргумента. В случае успеха возвращает вновь созданный объект **GwxTimedate**, и NULL – в противном случае. (См. также **GwxTimedate**, **GwxText**, **GwxVisible** и **GwxDynamic**). Данный метод может быть вызван только в режиме *Разработка*.

GWXTIMEDATEFORMATTYPE:

TimeOnly = 0,
DateOnly = 1,
TimeAndDate = 2

Пример VBA:

```
o.Selected = False
'if any exists...

ThisDisplay.DeleteObject ("TheText")


Dim objText As GwxText
Set objText = ThisDisplay.CreateText( _
                          448, _
                          232, _
                          "The Text", _
                          TextAlignCenter, _
                          False, _
                          True, _
```

```
                                RGB(200, 200, 230), _

                                RGB(10, 10, 50), _
                                1, _
                                LineNone, _
                                False, _
                                RGB(0, 0, 0), _
                                EdgeEtched, _
                                False, _
                                "TheText")
    If objText Is Nothing Then
        MsgBox "Operation failed."
        Exit Sub
    Else
        MsgBox "A new text object has been created."

    End If

    objText.UserDescription = "The Text"

    ThisDisplay.DeleteDynamic ("TheTimeAndDateObject")

    Dim obj As GwxTimedate

    Set obj = ThisDisplay.CreateTimedate( _
                            "TheText", _
                            "TheTimeAndDateObject", _
                            TimeAndDate, _
                            "h:mm:ss tt", _
                            "M/d/yyyy")
    If obj Is Nothing Then
        MsgBox "No dynamic object has been created."

        Exit Sub
    Else
        MsgBox "A new dynamic object has been created."
    End If
```

Function DeleteDynamic(objectName As String) As Boolean

Вызов данного метода приводит к удалению всей динамики, связанной с объектом, чье имя передается в качестве аргумента. В случае успеха возвращает TRUE, и FALSE, – в противном случае. Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
ThisDisplay.DeleteDynamic ("Rectangle2")
```

Function DeleteObject(objectName As String) As Boolean

Удаляет видимый объект с заданным именем. В случае успеха возвращает TRUE, и FALSE, – в противном случае. Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
ThisDisplay.DeleteObject ("Rectangle2")
```

Sub DeselectAllObjects()

Отменяет выделение всех выделенных в настоящий момент видимых объектов. Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
    Dim obj As Object
    Dim obj2 As Object

    'Select rectangles:
    Set obj = ThisDisplay.GetVisibleObjectFromName("Rectangle")
    Set obj2 = ThisDisplay.GetVisibleObjectFromName("Rectangle2")

    MsgBox "Now we will select the rectangles."
    obj.Selected = True
```

```
obj2.Selected = True

MsgBox "Now we are going to deselect both rectangles."

ThisDisplay.DeselectAllObjects
```

Function DisplayBack() As Boolean

Открывает предыдущую в списке запоминаемых экранную форму. (Не доступен в ActiveX ToolWorX).

Function DisplayForward() As Boolean

Открывает следующую в списке запоминаемых экранную форму. (Не доступен в ActiveX ToolWorX).

Пример VBA:

```
ThisDisplay.DisplayForward
```

Пример VB:

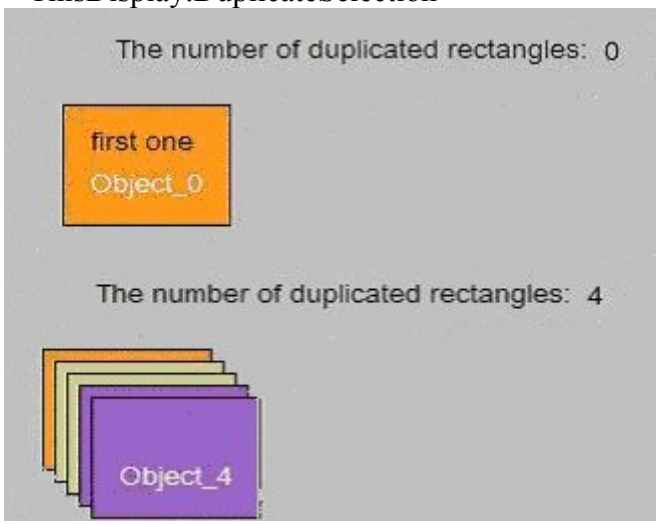
```
ThisDisplay.DisplayForward
```

Sub DuplicateSelection()

Создает копии (дублирует) текущих выделенных объектов. Метод доступен только в режиме *Разработка*. Не может вызываться для объектов **GWXview32**.

Пример VBA:

ThisDisplay.DuplicateSelection



Пример VBA:

```
'We assume we have created a rectangle (Object_X)
'with a text object (rectText) to watch how many
'rectangles is duplicated.
'"Counter" is a helper variable to indicate
'the number of the newly created rectangles.

'Helper declarations:
Dim x As Single
Dim y As Single
Dim width As Single
Dim height As Single

Dim obj As Object
'Get the helper index (Counter) to select
'the right rectangle.
Set obj = ThisDisplay.GetVisibleObjectFromName("Counter")

Dim txtCounter As GwxText 'intellisense
Set txtCounter = obj
Set obj = ThisDisplay.GetVisibleObjectFromName( _
    "Object_" + CStr(txtCounter.text))
```

```
Dim rect As GwxRectangle
Set rect = obj 'intellisense

rect.Selected = True
o.Selected = False 'Do not duplicate the button selection.

'Duplicate currantly selected object.
ThisDisplay.DuplicateSelection

'increase the number of rectangles
txtCounter.text = CStr(Val(txtCounter.text) + 1)


'Object_0 is automatically copied to a new object with
'the name Object_0,1,2...
Set obj = ThisDisplay.GetVisibleObjectFromName( _
    "Object_" + CStr(txtCounter.text))
Set rect = obj
Call rect.GetObjectDimensions(x, y, width, height)
Call rect.SetObjectDimensions(x + 6, y + 6, width, height)

'Set the random color:
Dim a As Long
Randomize (Second(Time) + Minute(Time) * 60 + Hour(Time) * 3600)
rect.fillColor = RGB(Int(255 * Rnd), _

                     Int(255 * Rnd), _
                     Int(255 * Rnd))
Dim txt As GwxText
Set txt = ThisDisplay.GetVisibleObjectFromName("rectText")


Call txt.GetObjectDimensions(x, y, width, height)
'Shift the label by 6 pixels.
Call txt.SetObjectDimensions(x + 6, y + 6, width, height)
txt.Selected = True
txt.text = "Object_" + CStr(txtCounter.text)
rect.Selected = False
ThisDisplay.MoveSelectionToFront


If Val(txtCounter.text) > 10 Then
    Dim j As Long
    For j = 1 To 11
        ThisDisplay.DeleteObject ("Object_" + CStr(j))
    Next j
    'Set the former text and position of the Object rectText
    Call txt.SetObjectDimensions(x - 6 * 10, y - 6 * 10, width, height)
    txt.objectName = "rectText"
    txt.text = "Object_0"
    txtCounter.text = "0"
End If

txt.Selected = False
```

Sub EnableRuntimeEvents(LeftButton As Boolean, MiddleButton As Boolean, RightButton As Boolean, MouseMove As Boolean, Keyboard As Boolean)

Разрешает /запрещает возникновение событий VBA от мыши и клавиатуры .
LeftButton – при значении True можно получать события DisplayMouse, связанные с левой клавишей мыши .
MiddleButton - при значении True можно получать события DisplayMouse, связанные со средней клавишей мыши .
RightButton - при значении True можно получать события DisplayMouse, связанные с правой клавишей мыши .
MouseMove - при значении True можно получать события DisplayMouse, связанные с перемещением мыши.

Keyboard - при значении True можно получать события DisplayMouse, связанные с клавиатурой.

Пример VBA:

```
    If ThisDisplay.IsEnabledRuntimeZoomAndPan Then
        ThisDisplay.EnableRuntimeZoomAndPan (False)
    Else
        ThisDisplay.EnableRuntimeZoomAndPan (True)
    End If
```

Пример VBA:

```
Private Sub GwxDisplay_PreRuntimeStart()

    ' Enable right click events only
    Call ThisDisplay.EnableRuntimeEvents( _

False, False, True, False, False)

End Sub
```

Пример VB:

```
    If ThisDisplay.IsEnabledRuntimeZoomAndPan Then
        ThisDisplay.EnableRuntimeZoomAndPan (False)
    Else
        ThisDisplay.EnableRuntimeZoomAndPan (True)
    End If
```

Sub ExitApplication()

Завершает работу GraphWorX32. Не может вызываться для экранных форм внутри объектов GWXview32.

Пример VBA:

```
ThisDisplay.ExitApplication
```

Пример VB:

```
ThisDisplay.ExitApplication
```

Function ExportBitmapCaptureOfView(filename As String) As Boolean

Экспортирует текущий вид экрана GraphWorX32 в растровый файл (BMP) с определенным именем. Не может вызываться для экранных форм внутри объектов GWXview32.

Пример VBA:

```
Dim strCurrDir As String
strCurrDir = ThisDisplay.GetFilePath

Dim strFileName As String
strFileName = "screenshot.bmp"

'Add a backslash at the end.
strFileName = strCurrDir + "\" + strFileName
ThisDisplay.ExportBitmapCaptureOfView (strFileName)
```

Пример VB:

```
Dim strCurrDir
strCurrDir = ThisDisplay.GetFilePath
```

```
Dim strFileName
strFileName = "screenshot.bmp"

'Add a backslash at the end.
strFileName = strCurrDir + "\" + strFileName
ThisDisplay.ExportBitmapCaptureOfView (strFileName)
```

Function FileNew() As Boolean

Создает новую экранную форму. В случае успеха возвращает TRUE, и FALSE, – в противном случае.

Пример VBA:

```
'Save the current display.
ThisDisplay.FileSave


'Starts a new empty display.
ThisDisplay.FileNew
```

Пример VB:

```
'Save the current display.
ThisDisplay.FileSave


'Starts a new empty display.
ThisDisplay.FileNew
```

Function FileOpen(filename As String) As Boolean

Загружает экранную форму, имя которой содержится в передаваемом аргументе. В случае успеха возвращает TRUE, и FALSE, – в противном случае. (В ActiveX ToolWorX этот метод открывает только экранные формы источников).

Примеры VBA:

```
'Save the current display.
ThisDisplay.FileSave

MsgBox "Open the display ""EmbeddedWindow.gdf""."

'Open a display.
ThisDisplay.FileOpen ("EmbeddedWindow.gdf")
```

Примеры VB:

```
'Save the current display.
ThisDisplay.FileSave

MsgBox "Open the display ""EmbeddedWindow.gdf""."

'Open a display.
ThisDisplay.FileOpen ("EmbeddedWindow.gdf")
```

Function FileOpenSetAliases(filename As String, aliasCommandType As GWXALIASCOMMAND, aliasCommandString As String) As Boolean

Загружает экранную форму, имя которой содержится в передаваемом аргументе, и выполняет настройку псевдонимов. Если GraphWorX32 находится в режиме *Исполнение*,

настройка псевдонимов должна выполняться до запроса значений каких-либо тегов. Аргумент **aliasCommandType** может принимать следующие значения:

- AliasNoCommand (настройка псевдонимов не выполняется) = 0
- AliasSetFromFile (настройка псевдонимов из файла) = 1
- AliasSetDirect (непосредственная настройка псевдонимов) = 2

Если аргумент **aliasCommandType** равен AliasSetFromFile, аргумент **aliasCommandString** должен содержать имя текстового файла привязки псевдонимов, каждая строка которого содержит имя псевдонима и, отделенное символом табуляции, имя источника данных (см. также **SetAliasesFromFile()**). Если аргумент **aliasCommandType** равен AliasSetDirect, аргумент **aliasCommandString** должен иметь следующий формат:

ИмяПсевдонима1<TAB>ОпределениеПсевдонима1<CR-LF>ИмяПсевдонима2<TAB>ОпределениеПсевдонима2<CR-LF> … и т.д.

Примечание: В ActiveX ToolWorX этот метод открывает только экранные формы источников.

Пример VBA:

```
'There are assumed 3 process points
'in the popup display to run correctly this
'example.
'Put this code (optional) in "DataSource" in the
'Property Inspector in each of the process points:
'ProcessPont N.1:
    'x= $"SourceAlias1: The value is "$ + ...
    '... tostring(<<SourceAlias1>>) + $" units."$
'ProcessPont N.2:
    'x= $"SourceAlias2: The value is "$ + ...
    '... tostring(<<SourceAlias2>>) + $" units."$
'ProcessPont N.3:

    'x= $"SourceAlias3: The value is "$ + ...
    '... tostring(<<SourceAlias3>>) + $" units."$
'Adjust properties according to the string lenght.
'See "Creating Expressions Using the Expression Editor"
'in the help.

'Open a popup window:
Dim strFileName As String
strFileName = "DisplayWithAliases.gdf"

'Save your display before you launch the new one.
'It is not necessary.
ThisDisplay.FileSave

'Compose a command string:

Dim strAliasCommand As String
strAliasCommand = "SourceAlias1" + Chr(9) + _
                    "100+abs({{gfwsim.sine.double}})*500" + _
                  Chr(13) + _
                  "SourceAlias2" + Chr(9) + _
                    "abs({{gfwsim.sine.double}})*1000" + _
                  Chr(13) + _
                  "SourceAlias3" + Chr(9) + _
                    "120+5*{{gfwsim.sine.double}}"
'Execute the function:
Call ThisDisplay.FileOpenSetAliases(strFileName, _

                                    AliasSetDirect, _
                                    strAliasCommand)
'... or you can use a file, where the same
'commands are written. For the file applyFor the tab spaces :
'Call ThisDisplay.FileOpenSetAliases(strFileName, _
                                    AliasSetFromFile, _
                                    "AliasCommandString.txt")
'Assume we have this file in the same directory.
```

Пример VB:

```
'There are assumed 3 process points
'in the popup display to run correctly this
'example.
'Put this code (optional) in "DataSource" in the
'Property Inspector in each of the process points:
'ProcessPont N.1:
    'x= $"SourceAlias1: The value is "$ + ...
    '... tostring(<<SourceAlias1>>) + $" units."$
'ProcessPont N.2:
    'x= $"SourceAlias2: The value is "$ + ...
    '... tostring(<<SourceAlias2>>) + $" units."$
'ProcessPont N.3:

    'x= $"SourceAlias3: The value is "$ + ...
    '... tostring(<<SourceAlias3>>) + $" units."$
'Adjust properties according to the string lenght.
'See "Creating Expressions Using the Expression Editor"
'in the help.

'Open a popup window:
Dim strFileName
strFileName = "DisplayWithAliases.gdf"

'Save your display before you launch the new one.
'It is not necessary.
ThisDisplay.FileSave

'Compose a command string:

Dim strAliasCommand
strAliasCommand = "SourceAlias1" + Chr(9) + _
                    "100+abs({{gfwsim.sine.double}})*500" + _
                  Chr(13) + _
                  "SourceAlias2" + Chr(9) + _
                    "abs({{gfwsim.sine.double}})*1000" + _
                  Chr(13) + _
                  "SourceAlias3" + Chr(9) + _
                    "120+5*{{gfwsim.sine.double}}"
'Execute the function:
Call ThisDisplay.FileOpenSetAliases(strFileName, _

                                    AliasSetDirect, _
                                    strAliasCommand)
'... or you can use a file, where the same
'commands are written. For the file applyFor the tab spaces :
'Call ThisDisplay.FileOpenSetAliases(strFileName, _
'                                     AliasSetFromFile, _
'                                    "AliasCommandString.txt")
'Assume we have this file in the same directory.
```

Function FilePrint() As Boolean

Выводит на печать текущую экранную форму. В случае успеха возвращает TRUE, и FALSE, – в противном случае. (Не поддерживается в ActiveX ToolWorX).

Пример VBA:

```
'Prints the current display.
ThisDisplay.FilePrint
```

Пример VB:

```
'Prints the current display.
ThisDisplay.FilePrint
```

Function FileSave() As Boolean

Сохраняет на диске текущую экранную форму. В случае успеха возвращает TRUE, и FALSE, – в противном случае. (Не поддерживается в экранных формах, исполняющихся внутри объектов GWXview32).

Пример VBA:

```
ThisDisplay.FileSave
```

Пример VB:

```
ThisDisplay.FileSave
```

Function FileSaveAs(filename As String) As Boolean

Сохраняет на диске текущую экранную форму в файле с именем, передаваемым в качестве аргумента. В случае успеха возвращает TRUE, и FALSE, – в противном случае. (Не поддерживается в экранных формах, исполняющихся внутри объектов GWXview32).

Пример VBA:

```
 Dim fileName As String
fileName = ThisDisplay.GetFileName

If ThisDisplay.FileSaveAs(fileName) Then
    MsgBox "The file has been successfully saved."
End If
```

Пример VB:

```
Dim fileName
fileName = ThisDisplay.GetFileName

If ThisDisplay.FileSaveAs(fileName) Then
    MsgBox "The file has been successfully saved."

End If
```

Function FileSaveAsType(filename As String, displayType As Integer) As Boolean

Сохраняет на диске текущую экранную форму в файле с именем, передаваемым в качестве аргумента. В случае успеха возвращает TRUE, и FALSE, – в противном случае. Величина Типа дисплея должна быть «0» для экранов VBA и «1» для экранов без VBA. (Не поддерживается в экранных формах, исполняющихся внутри объектов GWXview32).

Пример VBA:

```
'Saves the current file with the
'current name.

Dim VBAflag As Byte
'Does this display contain a VBA part?
If ThisDisplay.GetFileContainsVBA Then
    VBAflag = 0
Else
    VBAflag = 1
End If

Call ThisDisplay.FileSaveAsType( _
    ThisDisplay.GetFileName, _
    VBAflag)
```

Пример VB:

```
'Saves the current file with the
'current name.

Dim VBAflag
'Does this display contain a VBA part?
If ThisDisplay.GetFileContainsVBA Then
    VBAflag = 0
Else
```

```
    VBAflag = 1
End If

Call ThisDisplay.FileSaveAsType( _
    ThisDisplay.GetFileName, _
    VBAflag)
```

Sub GetClientDimensionsPixels(left As Long, top As Long, width As Long, height As Long)

Возвращает размеры и положение внутренней (клиентской) прямоугольной области GraphWorX32 в пикселах. Внутренней областью является прямоугольная область главного окна GraphWorX32 без рамки, области меню и области заголовка.

Пример VBA:

```
Dim left As Long
Dim top As Long
Dim width As Long
Dim height As Long

Call ThisDisplay.GetClientDimensionsPixels( _
                        left, _
                        top, _
                        width, _
                        height)

MsgBox "Client dimensions in pixels: " + _
   Chr(13) + _
   "left: " + CStr(left) + Chr(13) + _

   "top: " + CStr(top) + Chr(13) + _
   "width: " + CStr(width) + Chr(13) + _
   "height: " + CStr(height)
```

Пример VB:

```
Dim Left
Dim top
Dim width
Dim height

Call ThisDisplay.GetClientDimensionsPixels2( Left, _

                                             top, _

                                             width, _

                                             height)

MsgBox "Client dimensions in pixels: " + _
   Chr(13) + _
   "left: " + CStr(Left) + Chr(13) + _
   "top: " + CStr(top) + Chr(13) + _
   "width: " + CStr(width) + Chr(13) + _

   "height: " + CStr(height)
```

Sub GetDisplayDimensions(width As Long, height As Long)

Поиск массивов экранных форм (рабочих областей/внеш.границ) текущих загружаемых экранов.

Примеры VBA:

```
'declare coordinates as long values
Dim dispWidth As Long
Dim dispHeight As Long

'By reference values we get coordinates:
```

```
Call ThisDisplay.GetDisplayDimensions(dispWidth, _
                                      dispHeight)

MsgBox "The display dimensions are:" + _
   Chr(13) + " Width = " + _
   Conversion.CStr(dispWidth) + _
   ", Height = " + _
   Conversion.CStr(dispHeight)
```

Примечание: В сценарии VB должна быть использована подпрограмма **GetDisplayDimensions2**

Примеры VB:

```
'declare coordinates as long values
Dim dispWidth As Long
Dim dispHeight As Long

'TO DO!
'By reference values we get coordinates:
Call ThisDisplay.GetDisplayDimensions2(dispWidth, _
                                       dispHeight)

MsgBox "The display dimensions are:" + _
   Chr(13) + " Width = " + _
   CStr(dispWidth) + _
   ", Height = " + _
   CStr(dispHeight)
```

Function GetDynamicObjectFromName(objectName As String) As Object

Возвращает динамический объект по имени, передаваемому аргументом. Возможны следующие типы возвращаемого объекта: **GwxDynamic** или один из производных от **GwxDynamic** типов (**GwxSize**, **GwxLocation**, **GwxRotation**, **GwxHide**, **GwxFlash**, **GwxPick**, **GwxDigitalColor**, **GwxAnalogColor**, **GwxAnimator**, **GwxAnalogSelector**, **GwxDigitalSelector**, **GwxProcessPoint**, **GwxTimedate**). Возвращает NULL если объекты соответствующих типов не обнаружены.

Пример VBA:

```
Dim obj As Object
   Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")

   Dim oProcessPoint As GwxProcessPoint
   Set oProcessPoint = obj
   'You can complete your code with intellisence.

   'We are going to change
   oProcessPoint.dataSource = "gfwsim.random.long"
```

Function GetEventScriptText(Event As GWXEVENTNAME) As String

GraphWorX32 позволяет выполнять по событиям сценарии VBScript и JScript. Сценарии, связанные с событием, выполняются при возникновении события (например, после запуска приложения или перед запросом данных для слоя). Сценарии могут редактироваться с помощью новой интегрированной панели инструментов или с помощью интерфейса **OLE automation**. Сценарий может быть написан с использованием языков VBScript or Jscript независимо от другого события. Эта функция позволяет просматривать (не изменять) исходный код, связанный с определенным событием.

```
Enum  GWXEVENTNAME:
enum {
   DataEntryActivated = 0
   DataEntryDeactivated = 1
   DataEntryValueEntered = 2
   DialBeginDrag = 3
   DialDrag = 4
   DialEndDrag = 5
   DisplayLoad = 6
   DisplayUnload = 7
   LanguageChanged = 8
```

```
   PickPostDown = 9
   PickPostUp = 10
   PickPostWhileDown = 11
   PickPreDown = 12
   PickPreUp = 13
   PickPreWhileDown = 14
   PostAnimateDisplay = 15
   PostDeanimateDisplay = 16
   PreAnimateDisplay = 17
   PreDeanimateDisplay = 18
PreAnimateLayer = 19
   PostAnimateLayer = 20
   PreDeanimateLayer = 21
   PostDeanimateLayer = 22
   PreRuntimeStart = 23
   PostRuntimeStart = 24
   PreRuntimeStop = 25
   PostRuntimeStop = 26
   SliderBeginDrag = 27
   SliderEndDrag = 28
   SliderDrag = 29
   VCRStart = 30
   VCRStop = 31
   GlobalAliasChanged = 32
   ViewChanged = 33
   DisplayChanged = 34
   WindowChanged = 35
   PrePrint = 36
   PostPrint = 37
} GWXEVENTNAME;
```

Пример VBA:

```
MsgBox ThisDisplay.GetEventScriptText(PickPostDown)
```

Пример VB:

```
MsgBox ThisDisplay.GetEventScriptText(9)
```

Function GetEventScriptType(Event As GWXEVENTNAME) As GWXSCRIPTTYPE

GraphWorX32 8.0 позволяет выполнять по событиям сценарии VBScript и JScript. Сценарии, связанные с событием, выполняются при возникновении события (например, после запуска приложения или перед запросом данных для слоя). Сценарии могут редактироваться с помощью новой интегрированной панели инструментов или с помощью интерфейса **OLE automation**. Сценарий может быть написан с использованием языков VBScript or Jscript независимо от другого события. Эта функция позволяет просматривать (не изменять) исходный код, связанный с определенным событием.

Enum GWXSCRIPTTYPE

ScriptTypeNone = -1

ScriptTypeVBA = 0

ScriptTypeVBScript = 1

ScriptTypeJScript = 2

Пример VBA:

'работает в режиме Разработки и Исполнения
'создает сценарий VB по событию PostRuntimeStart

```
'sets PostRuntimeStart event script type to VB script
ThisDisplay.SetEventScriptType _
PostRuntimeStart, ScriptTypeVBScript

'adds text to the script
ThisDisplay.SetEventScriptText PostRuntimeStart, _
```

```
"MsgBox " & Chr(34) & "Test OK" & Chr(34)
MsgBox ThisDisplay.GetEventScriptType(PostRuntimeStart)
MsgBox ThisDisplay.GetEventScriptText(PostRuntimeStart)
```

Function GetFileContainsVBA() As Boolean

Функция **GetFileContainsVBA()** возвращает значение True value если экранная форма содержит код VBA и False – если не содержит. Присутствие VBA контролируется при загрузке экранной формы без VBA и при сохранении экранной формы без VBA, или при использовании методов **FileSaveAsType(1)** (содержит VBA) or **FileSaveAsType(0)** (не содержит VBA).

Пример VBA:

```
If ThisDisplay.GetFileContainsVBA Then
    MsgBox "This display currently contains the VBA part."
End If
```

Пример VB:

```
If ThisDisplay.GetFileContainsVBA Then
    MsgBox "This display currently contains the VBA part."
End If
```

Function GetFileName() As String

Возвращает имя текущей загруженной экранной формы, включая путь к файлу.

Пример VBA :

```
MsgBox "The name of this file is: " + _
    ThisDisplay.GetFileName
```

Пример VB:

```
MsgBox "The name of this file is: " + _
    ThisDisplay.GetFileName
```

Function GetFilePath() As String

Возвращает путь к файлу текущей экранной формы без имени файла.

Пример VBA:

```
Dim strPath As String

strPath = ThisDisplay.GetFilePath

MsgBox "The file path of the current display is: " + _
    StrPath
```

Пример VB:

```
Dim strPath

strPath = ThisDisplay.GetFilePath

MsgBox "The file path of the current display is: " + _
    StrPath
```

Function GetFileVersion() As Long

Функция **GetFileVersion()** возвращает следующие значения:

```
-1 .. OpenFileError
0  .. Ver520
1  .. Ver600
2  .. Ver600SP
3  .. VerMGraphics40
```

```
4  .. Ver610

5  .. Ver620

9999      .. VerLatest
```

VBA пример:

```
    'Get the version of the current
    'release.
    Dim lFileVersion As Long
    lFileVersion = ThisDisplay.GetFileVersion

    Dim strVersion As String
    Select Case lFileVersion
        Case -1
            strVersion = "OpenFileError"
        Case 0
            strVersion = "Ver520"
        Case 1
            strVersion = "Ver600"
        Case 2
            strVersion = "Ver600SP"
        Case 3
            strVersion = "VerMGraphics40"
        Case 4

            strVersion = "Ver610"
        Case 8 To 9999 'year 2004
            strVersion = "VerLatest"
    End Select

    MsgBox "The file version is: " + _
        CStr(lFileVersion) + _
        "(" + strVersion + ")"
```

VB пример:

```
    'Get the version of the current
    'release.
    Dim lFileVersion
    lFileVersion = ThisDisplay.GetFileVersion

    Dim strVersion
    Select Case lFileVersion
        Case -1
            strVersion = "OpenFileError"
        Case 0
            strVersion = "Ver520"
        Case 1
            strVersion = "Ver600"
        Case 2
            strVersion = "Ver600SP"
        Case 3
            strVersion = "VerMGraphics40"
        Case 4
            strVersion = "Ver610"

        Case 8
            strVersion = "VerLatest"
    End Select

    MsgBox "The file version is: " + _
        CStr(lFileVersion) + _
        "(" + strVersion + ")"
```

Function GetFrameWindowHandle() As Long

Возвращает идентификатор (handle) окна, в котором запущен данный объект типа GwxDisplay. (Не поддерживается в ActiveX ToolWorX).

VBA пример:

```
Dim handle As Long
handle = ThisDisplay.GetFrameWindowHandle

'Use the handle to specify the parent window.
Dim tagName As String
Call ThisDisplay.OpenTagBrowser(handle, tagName)
```

VB пример:

```
Dim handle
handle = ThisDisplay.GetFrameWindowHandle

'Use the handle to specify the parent window.
Dim tagName
Call ThisDisplay.OpenTagBrowser(handle, tagName)
```

Function GetGASThemes() As String

Возвращает текущую тему Глобальных псевдонимов в формате, определенном в Навигаторе Тем Глобальных псевдонимов.

```
'VBA/VB Script example, works only in runtime mode
'example for setting and getting themes (Floors theme is changed
in child windows)
ThisDisplay.SetGASThemes _
      "Buildings=BuildingB;Floors=Floor2/+1;"
MsgBox ThisDisplay.GetGASThemes
```

Function GetHeadObject() As Object

Возвращает текущий выделенный объект. Объект, на котором был произведен щелчок клавишей мыши, вписанный в прямоугольник с небольшими символами квадратной формы красного или синего цвета, называется выделенным. В режиме *Исполнение* выделенным объектом считается объект, которому передан фокус ввода. Если выделенных объектов не обнаружено, возвращается NULL.

Пример VBA:

```
Dim obj As Object
Set obj = ThisDisplay.GetHeadObject

MsgBox "The head visible object is: " + _
    obj.objectName
```

Пример VBA в режиме разработки:

```
Dim obj As Object
 Set obj = ThisDisplay.GetHeadObject

 MsgBox "The head visible object is: " + _
     obj.objectName
```

Function GetLoadTab(index As Long, label As String, File As String, AliasCmdType As Long, AliasCmdString As String, GlobalAliasCmdString As String) As Boolean

Возвращает значение TRUE, когда успешно получена информация настроек Закладок. Параметр index – индекс закладки (с отсчетом от нуля). Для получения информации о Закладке необходимо установить остальные параметры.

VBA пример:

```
Dim strlabel As String
Dim strfile As String
Dim lCommandType As Long
Dim strAliasCommand As String
Dim strGlAliasCommand As String

If ThisDisplay.GetLoadTab(0, _
                          strlabel, _
                          strfile, _
                          lCommandType, _
                          strAliasCommand, _
                          strGlAliasCommand) _
                       Then
    MsgBox "Properties of the display(0): " + _
        Chr(13) + _
        "label: " + strlabel + Chr(13) + _
        "file: " + strfile + Chr(13) + _
        "AliasCmdType: " + CStr(lCommandType) + Chr(13) + _
        "AliasCmdString: " + strAliasCommand + Chr(13) + _
        "GlobalAliasCmdString: " + strGlAliasCommand
End If
```

Function GetLoadTabsCount() As Long

Возвращает количество Закладок в экранной форме.

Function GetLocalAliases(DisplayName As String)

Возвращает локальные псевдонимы экранной формы DisplayName.

VBA пример:

```
Dim strAliases As Variant

Dim gwxNewDisp As New Gwx32.GwxDisplay

strAliases = gwxNewDisp.GetLocalAliases( _
                ThisDisplay.GetFilePath + _
                "\GwxDynamicVBA.gdf")

MsgBox "Have a look in the GwxDynamicVBA.gdf file..."
MsgBox "The first alias is: " + _
        strAliases(0)
```

VB пример:

```
Dim strAliases
Dim newGwxDisplay
Set newGwxDisplay = CreateObject("Gwx32.Display")


strAliases = newGwxDisplay.GetLocalAliases( _
                ThisDisplay.GetFilePath + _
                "\GwxDynamicVBA.gdf")

MsgBox "Have a look in the GwxDynamicVBA.gdf file..."
MsgBox "The first alias is: " + _
        strAliases(0)
```

Function GetLocalVariables(DisplayName As String)

Возвращает локальные переменные экранной формы DisplayName.

VBA пример:

```
Dim strVars As Variant

Dim gwxNewDisp As New Gwx32.GwxDisplay
strVars = gwxNewDisp.GetLocalVariables( _
                ThisDisplay.GetFilePath + _
                "\GwxDynamicVBA.gdf")

MsgBox "Have a look in the GwxDynamicVBA.gdf file..."
MsgBox "The first variable is: " + _
        strVars(0)
```

VB пример:

```
Dim strVars

Dim newGwxDisplay
Set newGwxDisplay = CreateObject("Gwx32.Display")

strVars = newGwxDisplay.GetLocalVariables( _
                ThisDisplay.GetFilePath + _
                "\GwxDynamicVBA.gdf")

MsgBox "Have a look in the GwxDynamicVBA.gdf file..."
MsgBox "The first local alias is: " + _
        strVars(0)
```

Function GetNumberOfTopLevelVisibles() As Long

Возвращает количество видимых объектов, из которых сгруппирован корневой символ экранной формы (называемых "видимыми объектами верхнего уровня", поскольку они являются объектами на самом верхнем уровне иерархии символа).

Пример VBA:

```
Dim visObj As Long
'returns the number of visible objects:
visObj = ThisDisplay.GetNumberOfTopLevelVisibles

MsgBox "The number of visible objects is: " + _
    CStr(visObj)
```

Пример VB:

```
Dim visObj
'returns the number of visible objects:
visObj = ThisDisplay.GetNumberOfTopLevelVisibles

MsgBox "The number of visible objects is: " + _
    CStr(visObj)
```

Function GetObjectTypeName() As String

Возвращает имя типа данного объекта ("**GwxDisplay**").

Пример VBA:

```
Dim obj As Object
    Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")

    Dim oDynamic As GwxDynamic
    'GwxProcessPoint has all the properties
    'and methods of GwxDynamic.
```

```
    Set oDynamic = obj
    'We can complete code via intellisence...

    Dim oPP As GwxProcessPoint
    Set oPP = oDynamic

    MsgBox "The type name of the Process Point is: " + _
        oPP.GetObjectTypeName
```

Пример VB:

```
    Dim obj
    Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")
    MsgBox "The type name of the Process Point is: " + _
        obj.GetObjectTypeName
```

Function GetOpenPopupByIndex(popupIndex As Long) As Object

Возвращает объект **GwxDisplay** текущего открытого всплывающего или встроенного окна, имеющий заданный индекс, начиная с 0 (**popupIndex** – всем встроенным и всплывающим окнам, которыми владеет данное главное окно экранной формы присваиваются числовые индексы). (Не поддерживается в ActiveX ToolWorX).

Пример VBA:

```
'Index number:
    '#0
    Call ThisDisplay.OpenPopupWindow("PopupWindow.gdf", _
                                     False, _
                                     True, _
                                     False)
    '#1
    Call ThisDisplay.OpenPopupWindow("PopupWindow.gdf", _
                                     False, _
                                     False, _
                                     False)
    Dim obj As Object
    Set obj = ThisDisplay.GetOpenPopupByIndex(1)

    Dim oPopup As GwxDisplay
    Set oPopup = obj
    oPopup.BackgroundColor = RGB(255, 0, 0)
```

Пример VB:

```
    'Index number:
    '#0
    Call ThisDisplay.OpenPopupWindow("PopupWindow.gdf", _
                                     False, _
                                     True, _
                                     False )
    '#1
      Call ThisDisplay.OpenPopupWindow("PopupWindow.gdf", _
                                     False, _
                                     False, _
                                     False)
    Dim obj
    Set obj = ThisDisplay.GetOpenPopupByIndex(1)

    obj.BackgroundColor = RGB(255, 0, 0)
```

Function GetOpenPopupCount() As Long

Возвращает количество открытых всплывающих и встроенных окон, принадлежащих текущей экранной форме. (Не поддерживается в ActiveX ToolWorX).

Пример VBA:

```
Call ThisDisplay.OpenPopupWindow("PopupWindow.gdf", _
                                     False, _
                                     True, _
```

```
                                    False)
    MsgBox "The number of popupWindows is " + _
           CStr(ThisDisplay.GetOpenPopupCount) + "." + _
           Chr(13) + "Click on the button once more please."
```

VB example:

```
    Call ThisDisplay.OpenPopupWindow("PopupWindow.gdf", _
                                    False, _
                                    True, _
                                    False)
    MsgBox "The number of popupWindows is " + _
           CStr(ThisDisplay.GetOpenPopupCount) + "." + _
           Chr(13) + "Click on the button once more please."
```

Function GetOpenPopupWithTitle(titleSubstring As String) As Object

Возвращает объект экранной формы типа **GwxDisplay** во всплывающем окне, заголовок которой содержит текстовый фрагмент, передаваемый методу в качестве аргумента. В случае неудачи – Nothing . (Не поддерживается в ActiveX ToolWorX).

Пример VBA:

```
Dim parentDisplay As Object

    'Assign an object to popup
    Set popup = ThisDisplay.GetOpenPopupWithTitle("PopupWindow.gdf")

    If Nothing Is popup Then
        MsgBox "There is no popup."
    Else
        MsgBox "The path to the popup window is:" + _
        popup.GetFileName
    End If
```

Пример VB:

```
    Dim popup
    'Assign an object to popup
    Set popup = ThisDisplay.GetOpenPopupWithTitle("PopupWindow.gdf")

    If Nothing Is popup Then
        MsgBox "There is no popup."
    Else
        MsgBox "The path to the popup window is:" + _
        popup.GetFileName
    End If
```

Function GetParentDisplay() As Object

Данный метод, вызываемый из сценария во всплывающем или встроенном окне, возвращает объект родительской экранной формы (из которой выполнялась загрузка встроенного или всплывающего окна). Если метод вызывается из сценария не во всплывающем или встроенном окне, возвращается NULL. (Не поддерживается в ActiveX ToolWorX).

Пример VBA:

```
Dim parentDisplay As Object
        'Assign an object to parentDisplay
    Set parentDisplay = ThisDisplay.GetParentDisplay

    If Nothing Is parentDisplay Then
        MsgBox "There is no parent window."
    Else
        MsgBox "The path to the parent display is:" + _
        parentDisplay.GetFileName
    End If
```

Пример VB:

```
    Dim parentDisplay
```

```
    'Assign an object to parentDisplay
Set parentDisplay = ThisDisplay.GetParentDisplay

If Nothing Is parentDisplay Then
    MsgBox "There is no parent window."
Else
    MsgBox "The path to the parent display is:" + _
    parentDisplay.GetFileName
End If
```

Function GetPointObjectFromName(pointName As String) As Object

Возвращает объект типа **GwxPoint** с заданным именем. Возвращает NULL при отсутствии объекта данного типа с указанным именем.

Пример VBA:

```
Dim obj As Object
    Set obj = _
ThisDisplay.GetPointObjectFromName("gfwsim.cosine.double")


    Dim oPoint As GwxPoint
    Set oPoint = obj

    Dim strName As String
    Dim strType As String
    Dim strPointName As String
    Dim val As Double
    Dim low As Double
    Dim high As Double

    strName = oPoint.GetObjectTypeName
    strType = oPoint.dataType
    strPointName = oPoint.GetPointName
    high = oPoint.highRange
    low = oPoint.lowRange
    val = oPoint.Value

    MsgBox "Source gfwsim.cosine.double: " + _
        "Object type name: " + strName + Chr(13) + _

        "Data Type: " + strType + Chr(13) + _
        "Point Name: " + strPointName + Chr(13) + _
        "High Range: " + CStr(high) + Chr(13) + _
        "Low Range: " + CStr(low) + Chr(13) + _
        "Current Value: " + CStr(val)
```

Пример VB:

```
    Dim oPoint
    Set oPoint = _
                ThisDisplay.GetPointObjectFromName("gfwsim.cosine.double")

    Dim strName
    Dim strType
    Dim strPointName
    Dim val
    Dim low
    Dim high

         If oPoint Is Nothing Then
                MsgBox "There is no source gfwsim.cosine.double to render."
         End If

    strName = oPoint.GetObjectTypeName
    strType = oPoint.dataType
    strPointName = oPoint.GetPointName
    high = oPoint.highRange
    low = oPoint.lowRange

    val = oPoint.Value
```

```
MsgBox "Source gfwsim.cosine.double: " + _
    "Object type name: " + strName + Chr(13) + _
    "Data Type: " + strType + Chr(13) + _
    "Point Name: " + strPointName + Chr(13) + _
    "High Range: " + CStr(high) + Chr(13) + _
    "Low Range: " + CStr(low) + Chr(13) + _
    "Current Value: " + CStr(val)
```

Function GetPrintViewArea()

Возвращает вид области дисплея для печати.

```
PV_NOCHANGE = 0,
PV_HOME = 1,
PV_FITTOWINDOW = 2,
PV_WORLD = 3
```

VBA пример:

```
Dim printViewArea As Boolean
printViewArea = ThisDisplay.GetPrintViewArea

Dim strprintViewArea As String
Select Case printViewArea
    Case 0
        strprintViewArea = "PV_NOCHANGE"
    Case 1
        strprintViewArea = "PV_HOME"
    Case 2
        strprintViewArea = "PV_FITTOWINDOW"
    Case 3
        strprintViewArea = "PV_WORLD"
End Select

MsgBox "The layout for printing is: " + _
    StrprintViewArea
```

VB пример

```
Dim printViewArea
printViewArea = ThisDisplay.GetPrintViewArea

Dim strprintViewArea
Select Case printViewArea
    Case 0
        strprintViewArea = "PV_NOCHANGE"
    Case 1
        strprintViewArea = "PV_HOME"
    Case 2
        strprintViewArea = "PV_FITTOWINDOW"
    Case 3
        strprintViewArea = "PV_WORLD"

End Select

MsgBox "The layout for printing is: " + _
    StrprintViewArea
```

Function GetPrintWhiteBackground () As Long

Возвращает область печати текущего дисплея.

VBA пример:

```
Dim layout As Long
layout = ThisDisplay.GetPrintWhiteBackground

'The optional layouts are:
'PV_NOCHANGE = 0,
'PV_HOME = 1,
'PV_FITTOWINDOW = 2,
'PV_WORLD = 3

Dim strLayout As String
Select Case layout
    Case 0
        strLayout = "PV_NOCHANGE"
    Case 1
        strLayout = "PV_HOME"
    Case 2
        strLayout = "PV_FITTOWINDOW"
    Case 3
        strLayout = "PV_WORLD"

End Select

MsgBox "The print view area is: " + strLayout
```

VB пример:

```
Dim layout
layout = ThisDisplay.GetPrintWhiteBackground

'The optional layouts are:
'PV_NOCHANGE = 0,
'PV_HOME = 1,
'PV_FITTOWINDOW = 2,
'PV_WORLD = 3

Dim strLayout
Select Case layout
    Case 0
        strLayout = "PV_NOCHANGE"
    Case 1
        strLayout = "PV_HOME"
    Case 2
        strLayout = "PV_FITTOWINDOW"
    Case 3
        strLayout = "PV_WORLD"
End Select


MsgBox "The print view area is: " + strLayout
```

Function GetSimulationVariables(DisplayName As String)

Получение переменных из другой экранной формы.

VBA пример:

```
    Dim strVars As Variant

    Dim gwxNewDisp As New Gwx32.GwxDisplay
    MsgBox "Before you use this function you have " + _
            Chr(13) + "to specify the path to display " + _
            "that you are interested in. e.g." + _
            "\\SERVER\...\myDisplay.gdf"

    'strVars = _
gwxNewDisp.GetSimulationVariables("\\SERVER\...\GwxDynamicVBA.gdf")

    'MsgBox "The first variable is: " + _
    '        strVars(0)
```

VB пример:

```
    Dim strVars

    Dim newGwxDisplay

    Set newGwxDisplay = CreateObject("Gwx32.Display")

    MsgBox "Before you use this function you have " + _
            Chr(13) + "to specify the path to display " + _
            "that you are interested in. e.g." + _
           "\\SERVER\...\myDisplay.gdf"

    'strVars = _
newGwxDisplay.GetSimulationVariables("\\SERVER\...\GwxDynamicVBA.gdf")

    'MsgBox "The first variable is: " + _
    '        strVars(0)
```

Function GetSystemWideLanguage(Language As String, LanguageSubset As String, LocaleID As Long, SynchronizeLCID As Boolean) As Boolean

Получает текущий выбранный язык для приложений GENESIS

Пример VBA:

```
    Dim strLanguage As String
    Dim strLanguageSubSet As String
    Dim lLocaleID As Long
    Dim bSynchronizeLCID As Boolean

    Call ThisDisplay.GetSystemWideLanguage(strLanguage, _
                                           strLanguageSubSet, _
                                           lLocaleID, _
                                           bSynchronizeLCID)
```

Function GetTransparency (TransparencyEnabled As Boolean, transparentColor As OLE_COLOR, AlphaEnabled As Boolean, AlphaBlending As Integer) As Boolean

Получает информацию о настройках прозрачности. Метод может быть вызван только в операционной системе Windows 2000 или выше.

Пример VBA:

```
   'VBA example, works in configure and runtime mode
```

```
'this VBA example gets transparency and transluency settings

Dim   TE As Boolean, _
      TC As OLE_COLOR, _
      AE As Boolean, _
      AB As Integer
ThisDisplay.GetTransparency TE, TC, AE, AB
MsgBox "Transparency: " & TE & vbCrLf & "Color: " & _
      TC & vbCrLf & "Transluency: " & AE & _
      vbCrLf & "Color: " & AB
```

Sub SetViewDimensions(left As long, top As long, width As long, height As long)

Возвращает размеры и положение рабочей области GraphWorX32. Определяет какая часть рабочей/всей области экрана будет видна. Можно использовать масштаб и прокрутку для просмотра экрана.

Пример:

```
Sub SetViewDimensions(o As GwxPick)
    'Declare coordinates as long values
    Dim viewLeft As Long
    Dim viewTop As Long
    Dim viewWidth As Long
    Dim viewHeight As Long

    'Backup values
    Dim bckLeft As Long
    Dim bckTop As Long
    Dim bckWidth As Long
    Dim bckHeight As Long

    'Remember current dimensions:
    Call ThisDisplay.GetViewDimensions(bckLeft, _
                                       bckTop, _

                                       bckWidth, _
                                       bckHeight)

    'Set new dimensions:
    viewLeft = 0
    viewTop = 0
    viewWidth = 1036
    viewHeight = 767

    'Set the view parameters:
    Call ThisDisplay.SetViewDimensions(viewLeft, _
                                       viewTop, _
                                       viewWidth, _
                                       viewHeight)
    'and refresh the window:

    ThisDisplay.RefreshWindow

    MsgBox "When you click on OK the view dimensions" + _
        " will be restored to the previous ones." + _
        Chr(13) + " (Left = " + _
        Conversion.CStr(bckLeft) + _
        " , Top = " + _
        Conversion.CStr(bckTop) + _
        " , Width = " + _
        Conversion.CStr(bckWidth) + _
        " and Height = " + _
        Conversion.CStr(bckHeight) + _
        ")"

    'Again, set the view parameters:

    Call ThisDisplay.SetViewDimensions(bckLeft, _
                                       bckTop, _
```

```
                                        bckWidth, _
                                        bckHeight)

    'and once more refresh the window:
    ThisDisplay.RefreshWindow
End Sub
```

Пример VB:

```
    'Declare coordinates as long values
    Dim viewLeft
    Dim viewTop
    Dim viewWidth
    Dim viewHeight
    'Backup values
    Dim bckLeft
    Dim bckTop
    Dim bckWidth
    Dim bckHeight

    'Remember current dimensions:
    'TO DO!
         Call ThisDisplay.GetViewDimensions2(bckLeft, _
                                        bckTop, _
                                        bckWidth, _
                                        bckHeight)
         bckLeft = 0
         bckTop = 0
         bckWidth = 1131
         bckHeight = 767
    'New dimensions: <0, 0, 502, 341>
    viewLeft = 0
    viewTop = 0
    viewWidth = 502
    viewHeight = 341

    'Set the view parameters:
    Call ThisDisplay.SetViewDimensions(viewLeft, _
                                        viewTop, _
                                        viewWidth, _
                                        viewHeight)
    'and refresh the window:
    ThisDisplay.RefreshWindow
        MsgBox "When you click on OK the view dimensions" + _
        " will be restored to the previous ones."


    'Again, set the view parameters:
    Call ThisDisplay.SetViewDimensions(bckLeft, _
                                        bckTop, _
                                        bckWidth, _
                                        bckHeight)

    'and once more refresh the window:
    ThisDisplay.RefreshWindow
```

Function GetVisibleObjectFromIndex(index As Long) As Object

Возвращает видимый объект верхнего уровня вложенности с заданным индексом. Объект, имеющий нулевой индекс, расположен под всеми остальными видимыми объектами. Вызов данного метода наиболее удобен при последовательном доступе ко всем объектам верхнего уровня. Получить общее количество объектов верхнего уровня можно с помощью вызова метода **GetNumberOfTopLevelVisibles**.

Пример:

```
Dim visibleObject As Object
Set visibleObject = ThisDisplay.GetVisibleObjectFromIndex(1)
MsgBox (visibleObject.GetObjectTypeName)
```

Function GetVisibleObjectFromName(objectName As String) As Object

Возвращает видимый объект с заданным именем. Возвращаемый объект будет иметь тип **GwxVisible** или один из производных от **GwxVisible** (**GwxRectangle**, **GwxEllipse**, **GwxLine**, **GwxText**, **GwxArc**, **GwxBitmap**, **GwxMetafile**, **GwxSymbol**, **GwxOleObject**, **GwxButton**). При отсутствии видимого объекта с заданным именем возвращает NULL.

Пример VBA:

```
'If you do not know the actual type
'of an object and you want to access
'it generically, use the object type
'Object.  This will cause VB to access
'methods and properties of the object
'by name rather than by ID, thereby
'avoiding ID numbering issues.

Dim obj As Object
'Dim rect As GwxRectangle 'is optional
'Set rect = obj

Set obj = ThisDisplay.GetVisibleObjectFromName("Rectangle")

MsgBox "Change the color of the rectangle above." + _

    " Its name is: " + obj.objectName

obj.fillColor = rgb(200, 200, 255)
MsgBox "Undo fillcolor."
obj.fillColor = rgb(255, 191, 0)
```

Пример VB:

```
'If you do not know the actual type
'of an object and you want to access
'it generically, use the object type
'Object.  This will cause VB to access
'methods and properties of the object
'by name rather than by ID, thereby
'avoiding ID numbering issues.

Dim obj
'Dim rect As GwxRectangle 'is optional
'Set rect = obj

Set obj = ThisDisplay.GetVisibleObjectFromName("Rectangle")

MsgBox "Change the color of the rectangle above." + _

    " Its name is: " + obj.objectName

obj.fillColor = RGB(200, 200, 255)
MsgBox "Undo fillcolor."
obj.fillColor = RGB(255, 191, 0)
```

Function GetVisibleObjectFromPosition(x As Long, y As Long, GoIntoSymbol As Boolean) As Object

Возвращает объект, расположенный в определенном месте на экране. Начинает с основного экрана и проходит далее, включая слои, пока не будет найден самый дальний объект. Если аргумент **GoIntoSymbol** имеет значение True, поиск продолжается до последнего видимого объекта данной позиции. Иначе функция возвращает верхний символ, содержащий объект.

Пример VBA:

```
'Works in configure and runtime mode
'this example gets the lowest level visible at random position
Dim w As Long, h As Long
Dim o As Object
ThisDisplay.GetDisplayDimensions w, h
'gets display width and height
If ThisDisplay.GetNumberOfTopLevelVisibles > 0 Then
```

```
'checks if display has at least one visible object!
'and runs a loop for find one

Do
  Set o = ThisDisplay.GetVisibleObjectFromPosition( _
Rnd * w, Rnd * h, True)

'tries to get visible object from random position - _

!if object is GwxSymbol, it goes into symbol and _
!continues searching

Loop Until Not (o Is Nothing)

'after any object is found, loop ends and example _
!shows name of the object
 MsgBox o.objectName
Else
 MsgBox "Example needs at least one visible object"
End If
```

Пример VBA:

```
    Dim obj As Object
    'We assume you have created a rectangle on the
    'specific position.
    Set obj = ThisDisplay.GetVisibleObjectFromPosition(588, _
                                                       256, _
                                                       True)
    Dim rect As GwxRectangle
    Set rect = obj
    MsgBox "Change the color of the rectangle." + _
        " Its name is: " + CStr(rect.objectName)

    rect.fillColor = rgb(200, 200, 255)

    MsgBox "Undo fillcolor."
    rect.fillColor = rgb(255, 191, 0)
```

Пример VB:

```
    Dim obj
        Set obj = ThisDisplay.GetVisibleObjectFromPosition(588, _
                                                       256, _
                                                       True)
    Dim rect
    Set rect = obj
    MsgBox "Change the color of the rectangle." + _
        " Its name is: " + CStr(rect.objectName)

    rect.fillColor = RGB(200, 30, 255)
         MsgBox "Undo fill color."
         rect.fillColor = RGB(200, 200, 255)
```

Sub GetWindowDimensionsPercent(left As Single, top As Single, width As Single, height As Single)

Возвращает размеры и положение главного окна GraphWorX32 в процентах от размеров всего экрана. Возвращаемые значения лежат в диапазоне от 0,0 до 1,0. (Для VB-сценариев используется подпрограмма GetWindowDimensionsPercent2).

Пример VBA:

```
Sub GetWndDimPercent(o As GwxPick)
    'declare resolution as long values
    Dim percLeft As Single
    Dim percTop As Single
    Dim percWidth As Single
    Dim percHeight As Single

    'By reference values we can get dimensions:
```

```
    Call ThisDisplay.GetWindowDimensionsPercent(percLeft, _
                                                percTop, _
                                                percWidth, _
                                                percHeight)

        MsgBox "GraphWorX32 main window size in percent is:" + _
    Chr(13) + " Left = " + _
    Conversion.CStr(percLeft) + _
    ", Top = " + _
    Conversion.CStr(percTop) + _
    ", Width = " + _
    Conversion.CStr(percWidth) + _
    ", Height = " + _
    Conversion.CStr(percHeight)
End Sub
```

Пример VB:

```
    'declare resolution
    Dim percLeft
    Dim percTop
    Dim percWidth
    Dim percHeight

    'By reference values we can get dimensions:
    Call ThisDisplay.GetWindowDimensionsPercent2(percLeft, _
                                                percTop, _
                                                percWidth, _
                                                percHeight)
        MsgBox "GraphWorX32 main window size in percent is:" + _
    Chr(13) + " Left = " + _
    CStr(percLeft) + _
    ", Top = " + _
    CStr(percTop) + _
    ", Width = " + _
    CStr(percWidth) + _
    ", Height = " + _
    CStr(percHeight)
```

Sub GetWindowDimensionsPixels(left As Long, top As Long, width As Long, height As Long)

Возвращает размеры и положение главного окна GraphWorX32 в пикселах.

Пример VBA:

```
Sub GetWndDimPixels(o As GwxPick)
    'declare resolution as long values
    Dim pixLeft As Long
    Dim pixTop As Long
    Dim pixWidth As Long
    Dim pixHeight As Long
            'By reference values we can get dimensions:
    Call ThisDisplay.GetWindowDimensionsPixels(pixLeft, _
                                               pixTop, _
                                               pixWidth, _
                                               pixHeight)
        MsgBox "GraphWorX32 main window size in pixels is:" + _
    Chr(13) + " Left = " + _
    Conversion.CStr(pixLeft) + _
    ", Top = " + _
    Conversion.CStr(pixTop) + _
    ", Width = " + _
    Conversion.CStr(pixWidth) + _
    ", Height = " + _
    Conversion.CStr(pixHeight)
End Sub
```

Пример VBA:

```
   'Declare resolution as long values
    Dim pixLeft
    Dim pixTop
    Dim pixWidth
```

```
    Dim pixHeight


TO DO!
    'By reference values we can get dimensions:
    Call ThisDisplay.GetWindowDimensionsPixels2(pixLeft, _
                                                pixTop, _
                                                pixWidth, _
                                                pixHeight)
        MsgBox "GraphWorX32 main window size in pixels is:" + _
    Chr(13) + " Left = " + _
    CStr(pixLeft) + _
    ", Top = " + _
    CStr(pixTop) + _
    ", Width = " + _

    CStr(pixWidth) + _
    ", Height = " + _
    CStr(pixHeight)
```

Function GetWorkingDirectory() As String

Возвращает рабочий каталог Genesis32. Поиск файлов экранных форм и конфигурационных файлов Genesis32 производится в этом каталоге.

Пример VBA:

```
MsgBox "The working directory is: " + _
  ThisDisplay.GetWorkingDirectory
```

Пример VB:

```
MsgBox "The working directory is: " + _
  ThisDisplay.GetWorkingDirectory
```

Function GwxGetVbaProject() As Object

Возвращает интерфейс проекта VBA, связанного с текущей экранной формой. Не поддерживается для экранных форм, просматриваемых внутри объектов **GWXview32**.

Пример VBA:

```
    'You need to add a Project | References | button Browse and
    'locate .../Program Files/Common/Microsoft...
    '...Shared/VBA/VBA6/VBE6EXT.OLB file.
    'Add it into references and check it.
    'It should point to Microsoft for Visual Basic Extensibility.

    Dim x As VBIDE.VBProject
    Set x = ThisDisplay.GwxGetVBAProject()

    ' Add a string programmatically above this subroutine.
    Call x.VBComponents.Item(1).CodeModule.AddFromString( _

 "' I am a comment!")

    Debug.Print x.HelpContextID
```

Sub HideWindow()

Скрывает главное окно GraphWorX32.

Пример VBA:

```
'You need to add a Project | References | button Browse and
  'locate .../Program Files/Common/Microsoft...
  '...Shared/VBA/VBA6/VBE6EXT.OLB file.
  'Add it into references and check it.
  'It should point to Microsoft for Visual Basic Extensibility.

  Dim x As VBIDE.VBProject
  Set x = ThisDisplay.GwxGetVBAProject()
```

```
  ' Add a string programmatically above this subroutine.
  Call x.VBComponents.Item(1).CodeModule.AddFromString( _

'" I am a comment!")

  Debug.Print x.HelpContextID
```

Function InsertLibraryObject(libraryObject As Object) As Boolean

Вставляет объект в экранную форму. Библиотечный объект может быть создан/получен с помощью средств OLE Automation Библиотеки Символов.
(Дополнительная информация – в документации Библиотеки Символов).
Примечание: Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
Dim newSym As Object
Dim libObj As Object

If Not ThisDisplay.InsertLibraryObjectByName("Misc",_

"Check") Then

  MsgBox "Error with InsertLibraryObjectByName!"
End If

Set newSym = ThisDisplay.GetHeadObject

MsgBox newSym.GetObjectTypeName

If Not ThisDisplay.ConvertGwxSymbolToLibraryObject(newSym, _
     libObj) Then
  MsgBox "Error converting symbol to library object!"
Else
  MsgBox "GwxSymbol successfully converted to library object!"
End If
```

Function InsertLibraryObjectByIndex(category As String, index As Long) As Boolean

Вставляет символ из Библиотеки Символов с помощью средств OLE automation.
Параметр **category** соответствует имени файла категории (.sdf) Библиотеки Символов; параметр **index** – индекс символа в категории с отсчетом от нуля.
Примечание: Данный метод может быть вызван только в режиме *Разработка*.

Пример VBA:

```
'We assume that the SymbolLibrary object
'is already loaded into your project.

Dim sl As SymbolLibrary.SymbolLibrary
Dim sc As SymbolLibrary.SymbolCategory

Set sl = New SymbolLibrary.SymbolLibrary
Set sc = sl.GetCategoryFromIndex(1)

MsgBox sl.GetRootDirectory

If sc.GetNumberOfSymbols > 0 Then
    If Not ThisDisplay.InsertLibraryObjectByIndex( _
                            Dir(sc.GetCategoryName), _
                            0) Then

        MsgBox "Error with InsertLibraryObjectByIndex!"
    Else
        MsgBox "Library object successfully inserted!"
    End If
Else
    MsgBox "Example needs at least one symbol"
```

```
End If
```

Function InsertLibraryObjectByName(category As String, symbol As String) As Boolean

Вставляет символ из Библиотеки Символов с помощью средств OLE Automation. Параметр **category** соответствует имени файла категории (.sdf) Библиотеки Символов; параметр **symbol** – имя символа в категории.

Примечание: Данный метод может быть вызван только в режиме *Разработка*.

Пример применения метода:

```
Dim SL As SymbolLibrary.SymbolLibrary
Dim SC As SymbolLibrary.SymbolCategory
Dim o As Object
Set SL = New SymbolLibrary.SymbolLibrary
Set SC = SL.GetCategoryFromIndex(0)
If SC.GetNumberOfSymbols > 0 Then
 ThisDisplay.InsertLibraryObjectByIndex _
      Dir(SC.GetCategoryName), _
      Round(Rnd * SC.GetNumberOfSymbols)
 ThisDisplay.InsertLibraryObjectByName _
      Dir(SC.GetCategoryName), _
      SC.GetSymbolName(Round(Rnd * SC.GetNumberOfSymbols))
 Set o = SC.GetSymbolFromIndex( _
      Round(Rnd * SC.GetNumberOfSymbols))
 ThisDisplay.InsertLibraryObject o
Else
 MsgBox "Example needs at least one symbol"
End If
```

Пример2 VBA:

```
'We assume that the SymbolLibrary object
    'is already loaded into your project.
    'In Microsoft Visual Basic -> Tools -> References check
    'the option SymbolLibrary.
    If Not ThisDisplay.InsertLibraryObjectByName("Components", "Batteries")
Then
        MsgBox "Error with InsertLibraryObjectByName!"
    Else
        MsgBox "Library object successfully inserted!"
    End If
```

Function InsertOleObject(progID As String, objName As String) As Boolean

Метод создает объект OLE Automation и вставляет его в экранную форму GraphWorX32. Этот интерфейс OLE Automation может быть вызван из VBA или из VBScript/JScript или из C++. Необходимо определить имя создаваемого объекта. Можно создавать объекты любых типов, но нужно знать имя объекта. Например, ICONICS использует следующие объекты:

- "ICONICS.SCRAlarmCtrl.1"
- "ICONICS.Event32.1"
- "ICONICS.Periodic32.1"
- "GWXGAUGE.GWXGaugeCtrl.1"
- "GWXSWITCH.GWXSwitchCtrl.1"
- "TWXViewer.TWXViewerCtrl.1"
- "AWXview32.AWXview32Ctrl.1"
- "GWXVIEW32.GWXview32Ctrl.1"
- "GWXNUMERIC.GWXNumericCtrl.1"
- "GWXVESSEL.GwxVesselCtrl.1"

- "ICONICS.FunctionalKeyCtrl.1"
- "MarqueeLaunch.MarqueeCtl.1"
- "PopupLaunch.PopupCtl.1"
- "Gwx32.Display"

VBA пример:

```
'ICONICS uses the following objects:
'ICONICS.SCRAlarmCtrl.1
'ICONICS.Event32.1
'ICONICS.Periodic32.1
'GWXGAUGE.GWXGaugeCtrl.1
'GWXSWITCH.GWXSwitchCtrl.1
'TWXViewer.TWXViewerCtrl.1
'AWXview32.AWXview32Ctrl.1
'GWXVIEW32.GWXview32Ctrl.1
'GWXNUMERIC.GWXNumericCtrl.1
'GWXVESSEL.GwxVesselCtrl.1
'ICONICS.FunctionalKeyCtrl.1
'MarqueeLaunch.MarqueeCtl.1
'PopupLaunch.PopupCtl.1
'Gwx32.Display

'To create object "Gwx32.Display" (e.g. in VB Script) you have

'to write this code:
'Dim newGwxDisplay
'Set newGwxDisplay = CreateObject("Gwx32.Display")


Call ThisDisplay.InsertOleObject("AWXview32.AWXview32Ctrl.1", _
     "view")
```

Function IsEmbeddedPopupWindow() As Boolean

Возвращает TRUE, если экранная форма, из которой вызывается данный метод, исполняется во встроенном окне, и FALSE, – в противном случае. (Не поддерживается в ActiveX ToolWorX).

Пример VBA:

```
Dim flag As Boolean

    flag = ThisDisplay.IsEmbeddedPopupWindow

    If flag Then
        MsgBox "The display is running in" + _
            " embedded popup window."
    Else
        MsgBox "The display is not running in" + _
            " embedded popup window."
    End If
```

Пример VB:

```
    Dim flag

    flag = ThisDisplay.IsEmbeddedPopupWindow

    If flag Then
        MsgBox "The display is running in" + _
            " embedded popup window."
    Else
        MsgBox "The display is not running in" + _
            " embedded popup window."
    End If
```

Function IsModified() As Boolean

Возращает TRUE, если в экранную форму были внесены изменения со времени последнего сохранения.

VBA пример:

```
Dim Modified As Boolean
Modified = ThisDisplay.IsModified
If Modified Then
    MsgBox "The display has been modified since the last save."
Else
    MsgBox "The display has not been modified since the last save."
End If
```

VB пример:

```
Dim Modified
Modified = ThisDisplay.IsModified

If Modified Then
    MsgBox "The display has been modified since the last save."
Else
    MsgBox "The display has not been modified since" + _
                  " the last save."
End If
```

Function IsPopupWindow() As Boolean

Возвращает TRUE, если экранная форма, из которой вызывается данный метод, исполняется во всплывающем окне, и FALSE, – в противном случае. (Не поддерживается в ActiveX ToolWorX).

Пример VBA

```
Dim Modified As Boolean
    Modified = ThisDisplay.IsModified
    If Modified Then
        MsgBox "The display has been modified since the last save."
    Else
        MsgBox "The display has not been modified since the last save."
    End If
```

Пример VB

```
    Dim Modified
    Modified = ThisDisplay.IsModified

    If Modified Then
        MsgBox "The display has been modified since the last save."
    Else
        MsgBox "The display has not been modified since" + _
                      " the last save."
    End If
```

Function IsRuntimeMode() As Boolean

Возвращает TRUE, если GraphWorX32 находится в режиме Исполнение, и FALSE, – в противном случае.

Пример VBA:

```
  'Returns current mode.
    Select Case (ThisDisplay.IsRuntimeMode)
        Case True:
            MsgBox "You are in runtime mode."
        Case False:
            MsgBox "You are in design mode."
    End Select
```

Пример VB:

```
    'Returns current mode.
    Select Case (ThisDisplay.IsRuntimeMode)
```

```
        Case True:
            MsgBox "You are in runtime mode."
        Case False:
            MsgBox "You are in design mode."
    End Select
```

Sub LogicalPointToClient(x As Long, y As Long)

Преобразует логические координаты GraphWorX32 в клиентские координаты. (Не поддерживается в ActiveX ToolWorX).

Пример:

```
Dim obj As Object
    Dim xL As Long
    Dim yL As Long

    Set obj = ThisDisplay.GetVisibleObjectFromName("ObjectX_gridL")
    xL = Val(obj.text)
    Set obj = ThisDisplay.GetVisibleObjectFromName("ObjectY_gridL")
    yL = Val(obj.text)

    Call ThisDisplay.LogicalPointToClient(xL, yL)

    Set obj = ThisDisplay.GetVisibleObjectFromName("ObjectX_gridC")
    obj.text = CStr(xL)
    Set obj = ThisDisplay.GetVisibleObjectFromName("ObjectY_gridC")

    obj.text = CStr(yL)
```

Sub LogicalPointToScreen(x As Long, y As Long)

Преобразует логические координаты GraphWorX32 в экранные координаты. (Не поддерживается в ActiveX ToolWorX).

Пример VBA:

```
'We assume we have created a text object to store
    'the data.

    Dim obj As Object
    Dim xL As Long
    Dim yL As Long

    Set obj = ThisDisplay.GetVisibleObjectFromName("ObjectX_gridL")
    xL = Val(obj.text)
    Set obj = ThisDisplay.GetVisibleObjectFromName("ObjectY_gridL")
    yL = Val(obj.text)

    Call ThisDisplay.LogicalPointToScreen(xL, yL)

    Set obj = ThisDisplay.GetVisibleObjectFromName("ObjectX_gridS")
    obj.text = CStr(xL)

    Set obj = ThisDisplay.GetVisibleObjectFromName("ObjectY_gridS")
    obj.text = CStr(yL)
```

Sub MaximizeWindow()

Увеличивает главное окно GraphWorX32 до максимального размера. (Не поддерживается для экранных форм, исполняющихся в объектах **GWXview32**).

Пример VBA:

```
ThisDisplay.MaximizeWindow
```

Пример VB:

```
ThisDisplay.MaximizeWindow
```

Sub MinimizeWindow()

Сворачивает главное окно GraphWorX32 в пиктограмму. (Не поддерживается для экранных форм, исполняющихся в объектах **GWXview32**).

Пример VBA:

```
ThisDisplay.MinimizeWindow
```

Пример VB:

```
ThisDisplay.MinimizeWindow
```

Function MoveSelectionBackward() As Boolean

Перемещает отмеченный объект на одну позицию назад. Используется для сортировки объектов и размещения одного объекта под другим.

Пример VBA:

```
Dim obj As Object
Dim obj2 As Object
Set obj = ThisDisplay.GetVisibleObjectFromName("Object_red")
Dim oRec As GwxEllipse
Set oRec = obj

ThisDisplay.DeselectAllObjects
oRec.Selected = True

ThisDisplay.MoveSelectionBackward
```

Function MoveSelectionForward() As Boolean

Перемещает отмеченный объект на одну позицию вперед. Используется для сортировки объектов и размещения одного объекта над другим.

Пример VBA:

```
Dim obj As Object
Dim obj2 As Object
Set obj = ThisDisplay.GetVisibleObjectFromName("Object_red")
Dim oRec As GwxEllipse
Set oRec = obj
    ThisDisplay.DeselectAllObjects
oRec.Selected = True
ThisDisplay.MoveSelectionForward
```

Function MoveSelectionToBack() As Boolean

Перемещает отмеченный объект позади других объектов экранной формы.

Пример VBA:

```
Dim obj As Object
Set obj = ThisDisplay.GetVisibleObjectFromName("Object_red")
Dim oRec As GwxEllipse
Set oRec = obj
   ThisDisplay.DeselectAllObjects
oRec.Selected = True

ThisDisplay.MoveSelectionToBack
```

Function MoveSelectionToFront() As Boolean

Перемещает отмеченный объект впереди других объектов экранной формы.

Пример VBA:

```
Dim obj As Object
Set obj = ThisDisplay.GetVisibleObjectFromName("Object_red")
Dim oRec As GwxEllipse
Set oRec = obj

ThisDisplay.DeselectAllObjects
oRec.Selected = True

ThisDisplay.MoveSelectionToFront
```

Function OpenEmbeddedWindow(filename As String, center As Boolean, hidden As Boolean) As Object

Открывает встроенное окно GraphWorX32 и загружает в него экранную форму с заданным именем. В случае успеха возвращает объект экранной формы (типа **GwxDisplay**), загруженной во встроенное окно, и NULL, – в случае неудачи. (Не поддерживается в ActiveX ToolWorX).

Пример VBA:

```
Call ThisDisplay.OpenEmbeddedWindow("EmbeddedWindow.gdf", _
 True, False)
```

Пример VB:

```
Call ThisDisplay.OpenEmbeddedWindow("EmbeddedWindow.gdf", _
 True, False)
```

Function OpenEmbeddedWinSetAliases(filename As String, center As Boolean, hidden As Boolean, aliasCommandType As GWXALIASCOMMAND, aliasCommandString As String) As Object

Открывает встроенное окно GraphWorX32, загружает в него экранную форму с заданным именем и выполняет настройку псевдонимов. Если экранная форма, в которой вызывается данный метод, находится в режиме *Исполнение*, настройка псевдонимов должна выполняться до запроса значений каких-либо тегов. Описание аргументов **aliasCommandType** и **aliasCommandString** приведено в описании метода **FileOpenSetAliases**(). (Не поддерживается в ActiveX ToolWorX).

Пример VBA:

```
'Specify a text file. Take care all
'of the white spaces (only TABS) dividing strings.
'Set to use real tabs in your editor.
'Template is:
'aliasName1<TAB>aliasDef1
'aliasName2<TAB>aliasDef2
'<TAB> is the real TAB character.
Dim strTextFileName As String
strTextFileName = "AliasCommandString.txt"

'The file contents:
'SourceAlias1    100+abs({{gfwsim.sine.double}})*500
'SourceAlias2    abs({{gfwsim.sine.double}})*1000

'SourceAlias3    120+5*{{gfwsim.sine.double}}



'Specify an embedded window:
Dim strEmbdWndFileName As String
strEmbdWndFileName = "EmbeddedWindowWithAliases.gdf"
'To make this example working properly we
'assume 3 process points in the display.
'Put this code in "DataSource" in
'Property Inspector in each of the process points:

'ProcessPont N.1:
    'x= $"SourceAlias1: The value is "$ + ...
    '... tostring(<<SourceAlias1>>) + $" units."$

'ProcessPont N.2:
    'x= $"SourceAlias2: The value is "$ + ...
    '... tostring(<<SourceAlias2>>) + $" units."$
'ProcessPont N.3:
    'x= $"SourceAlias3: The value is "$ + ...
    '... tostring(<<SourceAlias3>>) + $" units."$

'Adjust properties according to the string lenght and format.
'See "Creating Expressions Using the Expression Editor"
```

```
    'in the help.


    Call ThisDisplay.OpenEmbeddedWinSetAliases(strEmbdWndFileName, _

                                               True, _
                                               False, _
                                               AliasSetFromFile, _
                                               strTextFileName)
```

Пример VB:

```
    'Specify a text file. Take care all
    'of the white spaces (only TABS) dividing strings.
    'Set to use real tabs in your editor.
    'Template is:
    'aliasName1<TAB>aliasDef1
    'aliasName2<TAB>aliasDef2
    '<TAB> is the real TAB character.
    Dim strTextFileName
    strTextFileName = "AliasCommandString.txt"

    'The file contents:
    'SourceAlias1    100+abs({{gfwsim.sine.double}})*500
    'SourceAlias2    abs({{gfwsim.sine.double}})*1000

    'SourceAlias3    120+5*{{gfwsim.sine.double}}


    'Specify an embedded window:
    Dim strEmbdWndFileName
    strEmbdWndFileName = "EmbeddedWindowWithAliases.gdf"
    'To make this example working properly we
    'assume 3 process points in the display.
    'Put this code in "DataSource" in
    'Property Inspector in each of the process points:
        'ProcessPont N.1:
        'x= $"SourceAlias1: The value is "$ + ...
        '... tostring(<<SourceAlias1>>) + $" units."$

    'ProcessPont N.2:
        'x= $"SourceAlias2: The value is "$ + ...
        '... tostring(<<SourceAlias2>>) + $" units."$
    'ProcessPont N.3:
        'x= $"SourceAlias3: The value is "$ + ...
        '... tostring(<<SourceAlias3>>) + $" units."$

    'Adjust properties according to the string lenght and format.
    'See "Creating Expressions Using the Expression Editor"
    'in the help.

    Call ThisDisplay.OpenEmbeddedWinSetAliases(strEmbdWndFileName, _
                                               True, _
                                               False, _
                                               AliasSetFromFile, _
                                               strTextFileName)
```

Function OpenPopupWindow(filename As String, modal As Boolean, center As Boolean, hidden As Boolean) As Object

Открывает всплывающее окно GraphWorX32 и загружает в него экранную форму с заданным именем. В случае успеха возвращает объект экранной формы (типа **GwxDisplay**), загруженной во всплывающее окно, и NULL, – в случае неудачи. (Не поддерживается в ActiveX ToolWorX).

Пример VBA:

```
Sub OpenPopupWnd(o As GwxPick)
    'Open display "PopupWindow.gdf":
    Call ThisDisplay.OpenPopupWindow("PopupWindow.gdf", False, _
```

```
 True, False)
End Sub
```

Function OpenPopupWinSetAliases(filename As String, modal As Boolean, center As Boolean, hidden As Boolean, aliasCommandType As GWXALIASCOMMAND, aliasCommandString As String) As Object)

Открывает всплывающее окно GraphWorX32, загружает в него экранную форму с заданным именем и выполняет настройку псевдонимов. Если экранная форма, в которой вызывается данный метод, находится в режиме *Исполнение*, настройка псевдонимов должна выполняться до запроса значений каких-либо тегов. Описание аргументов **aliasCommandType** и **aliasCommandString** приведено в описании метода **FileOpenSetAliases**(). (Не поддерживается в ActiveX ToolWorX).

Пример VBA:

```
'Index number:
'#0
Call ThisDisplay.OpenPopupWindow("PopupWindow.gdf", _
                                 False, _
                                 True, _
                                 False)
'#1
Call ThisDisplay.OpenPopupWindow("PopupWindow.gdf", _
                                 False, _
                                 False, _
                                 False)
Dim obj As Object
Set obj = ThisDisplay.GetOpenPopupByIndex(1)

Dim oPopup As GwxDisplay
Set oPopup = obj
oPopup.BackgroundColor = RGB(255, 0, 0)
```

Пример VB:

```
'Index number:
'#0
Call ThisDisplay.OpenPopupWindow("PopupWindow.gdf", _
                                 False, _
                                 True, _
                                 False )
'#1
  Call ThisDisplay.OpenPopupWindow("PopupWindow.gdf", _
                                 False, _
                                 False, _
                                 False)
Dim obj
Set obj = ThisDisplay.GetOpenPopupByIndex(1)

obj.BackgroundColor = RGB(255, 0, 0)
```

Function OpenSetAliasesDialog(aliasCommandType As GWXALIASCOMMAND, aliasCommandString As String, objectName As String) As Boolean

Открывает диалоговую панель настройки псевдонимов. Данная диалоговая панель предназначена для ввода оператором новых определений для передаваемого набора псевдонимов. Описание аргументов **aliasCommandType** и **aliasCommandString** приведено в описании метода **FileOpenSetAliases**(). Аргумент **aliasCommandString** предназначен для конфигурирования диалоговой панели. Для каждого уникального имени псевдонима, заданного в **aliasCommandString**, в диалоговой панели будет создана отдельная закладка. Определения каждого псевдонима будут отображаться в выпадающих списках соответствующих страниц диалоговой панели, выбираемых при помощи созданных закладок. Аргумент **objectName** указывает имя видимого объекта, к которому будет применена настройка псевдонимов. Если имя объекта не указано, настройка псевдонимов будут применена ко всей экранной форме. (Не поддерживается в ActiveX ToolWorX).

Пример VBA:

```
'1. AliasSetFromFile
'Specify a text file. Take care all
'of the white spaces (only TABS) dividing strings.
'Set to use real tabs in your editor.
'Template is:
'aliasName1<TAB>aliasDef1
'aliasName2<TAB>aliasDef2
'<TAB> is the real TAB character.
Dim strTextFileName As String
strTextFileName = "AliasCommandString.txt"

'The file contents:
'SourceAlias1    100+abs({{gfwsim.sine.double}})*500
'SourceAlias2    abs({{gfwsim.sine.double}})*1000

'SourceAlias3    120+5*{{gfwsim.sine.double}}

MsgBox "AliasSetFromFile"

Call ThisDisplay.OpenSetAliasesDialog(AliasSetFromFile, _
                                      strTextFileName, _
                                      "ProcessPoint")

'2.
'Compose a command string:
Dim strAliasCommand As String
strAliasCommand = "SourceAlias1" + Chr(9) + _
                    "100+abs({{gfwsim.sine.double}})*500" + _

                  Chr(13) + _
                  "SourceAlias1" + Chr(9) + _
                    "abs({{gfwsim.sine.double}})*1000" + _
                  Chr(13) + _
                  "SourceAlias2" + Chr(9) + _
                    "120+5*{{gfwsim.sine.double}}"


Call ThisDisplay.OpenSetAliasesDialog(AliasSetDirect, _
                                       strAliasCommand, _
                                       "ProcessPoint")
```

Function OpenTagBrowser(hWndParent As Long, tagName As String) As Boolean

Выводит на экран окно **Навигатора тегов**. Первый аргумент должен содержать идентификатор (handle) родительского окна экранной формы, в котором находится создаваемое динамическое соединение с тегом OPC. Второй аргумент после успешного выполнения метода будет содержать имя тега OPC. В случае, если пользователь щелкнул на кнопке **OK** в окне **Навигатора тегов**, возвращается TRUE. Если пользователь щелкнул на кнопке **Отмена** (Cancel) – возвращается FALSE.

Пример

```
Dim handle As Long
    handle = ThisDisplay.GetFrameWindowHandle

    'Use the handle to specify the parent window.
    Dim tagName As String
    Call ThisDisplay.OpenTagBrowser(handle, tagName)
```

Пример VB :

```
    Dim handle
    handle = ThisDisplay.GetFrameWindowHandle

    'Use the handle to specify the parent window.
    Dim tagName
    Call ThisDisplay.OpenTagBrowser2(handle, tagName)
```

Function PopAllCurrentSymbol() As Boolean

Перемещает текущий уровень редактирования символа вплоть до корневого уровня. Возвращает TRUE в случае успеха и FALSE, – в случае неудачи. (См. также **GwxSymbol::PushCurrentSymbol()**). Может вызываться только в режиме *Разработка*.

Пример VBA

```
ThisDisplay.PopAllCurrentSymbol
```

Function PopCurrentSymbol() As Boolean

Перемещает текущий уровень редактирования символа на один уровень вверх. Возвращает TRUE в случае успеха и FALSE, – в случае неудачи. (См. также **GwxSymbol::PushCurrentSymbol()**). Может вызываться только в режиме *Разработка*.

Пример VBA

```
ThisDisplay.PopAllCurrentSymbol
```

Function QueryDataType(dataSource As String, dataType As GWXDATATYPE) As Boolean

Запрашивает у OPC-сервера тип данных для тега, имя которого задается первым аргументом. Допустимые значения **GWXDATATYPE**:

- DataTypeShort = 2
- DataTypeLong = 3
- DataTypeFloat = 4
- DataTypeDouble = 5
- DataTypeString = 8
- DataTypeBool = 11
- DataTypeChar = 16
- DataTypeByte = 17
- DataTypeWord = 18
- DataTypeDword = 19

Пример VBA

```
Dim strSource As String
    strSource = "ICONICS.Simulator.1\SimulatePLC.Sine"

    Dim lDataType As GWXDATATYPE
    Call ThisDisplay.QueryDataType(strSource, _
                                   lDataType)

    Dim strDataType As String

    strDataType = "DataType"
    Select Case lDataType
        Case 0
            strDataType = strDataType + "Empty"
        Case 2
            strDataType = strDataType + "Short"

        Case 3
            strDataType = strDataType + "Long"
        Case 4
            strDataType = strDataType + "Float"
        Case 5
            strDataType = strDataType + "Double"
        Case 8
            strDataType = strDataType + "String"
        Case 11
            strDataType = strDataType + "Bool"
        Case 16
            strDataType = strDataType + "Char"
        Case 17
            strDataType = strDataType + "Byte"
        Case 18
```

```
            strDataType = strDataType + "Word"

        Case 19
            strDataType = strDataType + "Dword"
        Case Else
            strDataType = "Type undefined."
    End Select


    MsgBox "Data type of the SimulatePLC.Sine source is: " + _
            CStr(strDataType)
```

Пример VB:

```
    Dim strSource
    strSource = "ICONICS.Simulator.1\SimulatePLC.Sine"

    Dim lDataType
    Call ThisDisplay.QueryDataType2(strSource, _
                                    lDataType)

    Dim strDataType

    strDataType = "DataType"
    Select Case lDataType
        Case 0
            strDataType = strDataType + "Empty"

        Case 2
            strDataType = strDataType + "Short"
        Case 3
            strDataType = strDataType + "Long"
        Case 4
            strDataType = strDataType + "Float"
        Case 5
            strDataType = strDataType + "Double"
        Case 8
            strDataType = strDataType + "String"
        Case 11

            strDataType = strDataType + "Bool"
        Case 16
            strDataType = strDataType + "Char"
        Case 17
            strDataType = strDataType + "Byte"
        Case 18
            strDataType = strDataType + "Word"
        Case 19
            strDataType = strDataType + "Dword"
        Case Else
            strDataType = "Type undefined."

    End Select


    MsgBox "Data type of the SimulatePLC.Sine source is: " + _
            CStr(strDataType)
```

Function QueryRanges(dataSource As String, lowRange As Double, highRange As Double) As Boolean

Запрашивает у OPC-сервера верхнюю и нижнюю границы диапазона значений для тега, имя которого задается первым аргументом.

Пример VBA

```
Dim strSource As String
    strSource = "ICONICS.Simulator.1\SimulatePLC.Sine"

    Dim lowRange As Double
    Dim highrange As Double
```

```
If ThisDisplay.QueryRanges(strSource, _
                           lowRange, _
                           highrange) _
                           Then
   MsgBox "The High Range is: " + CStr(highrange) + _
       Chr(13) + "The Low Range is: " + CStr(lowRange)
End If
```

Пример VB:

```
Dim strSource
strSource = "ICONICS.Simulator.1\SimulatePLC.Sine"

Dim lowRange
Dim highrange

If ThisDisplay.QueryRanges2(strSource, _
                           lowRange, _
                           highrange) _
                           Then
   MsgBox "The High Range is: " + CStr(highrange) + _
       Chr(13) + "The Low Range is: " + CStr(lowRange)

End If
```

Sub RefreshWindow()

Redraws the visible portion of a GraphWorX32 display.

VBA пример:

```
'To run this script you need to create an object with _
'name "Test2".

'Toggle Redraw so the display will
'not be able to automatically
'refresh. At the end of this script
'we will manualy call RefreshWindow.
ThisDisplay.Redraw = False

'Get rectangle from name:
Dim oRec As GwxRectangle
Set oRec = ThisDisplay.GetVisibleObjectFromName("Test2")
oRec.Selected = True

'Set Redraw to True:

ThisDisplay.Redraw = True

'and finally use refresh:
Call ThisDisplay.RefreshWindow
```

VB пример:

```
'This routine changes the dimensions of display with following _
     'display recovery.


     'Declare coordinates as long values
Dim widthDisplay
Dim heightDisplay

Dim widthDisplayMem
Dim heightDisplayMem
```

```
'Remember current dimensions:
     'TO DO!
     Call ThisDisplay.GetDisplayDimensions2(widthDisplayMem, _
           heightDisplayMem)
     widthDisplayMem = 1036
heightDisplayMem = 767

'Set up new parameters:

widthDisplay = 600
heightDisplay = 400

'Set the display parameters:
Call ThisDisplay.SetDisplayDimensions(widthDisplay, _
            heightDisplay)
'and refresh the window:
ThisDisplay.RefreshWindow

MsgBox "When you click on OK the dimensions will be restored."

'Again, set display parameters:
Call ThisDisplay.SetDisplayDimensions(widthDisplayMem, _
            heightDisplayMem)

 'and once more refresh the window:

ThisDisplay.RefreshWindow
```

Sub RemoveLayer(layerName As String)

Удаляет слой из экранной формы.

Function ReplaceFilePath(OldSubstring As String, NewSubsring As String) As Long

Заменяет путь к файлу в дин. элементах. Возвращает 0 при успешном выполнении и HRESULT – при неудачном.

Пример VBA:

```
'similar to ReplaceTag
'VBA example, works only in configure mode
'replaces file path in whole display

Dim Status As Long
Status = ThisDisplay.ReplaceFilePath("C:\Windows\Temp", "D:\Temp")
'if Status <> 0 then there was no replacement performed or an error occurs

If Status <> 0 Then
 MsgBox "No replacements"
Else
 MsgBox "Tags replaced"
End If
```

Function ReplaceHost(OldHostName As String, NewHostName As String) As Long

Заменяет имя узла в строках тегов и пути к файлам. Возвращает 0 при успешном выполнении и HRESULT – при неудачном.

VBA пример 1:

```
If 0 <> ThisDisplay.ReplaceHost( _
     "\\host1\Simulator\SimulatePLC.Sine", _
     "\\host2\Simulator\SimulatePLC.Sine") _
     Then
 MsgBox "Something fails."
```

```
    Else
        MsgBox "The node name has been successfully replaced."
    End If
```

VBA пример 2:

```
'similar to ReplaceTag
'VBA example, works only in configure mode
'replaces host name in whole display

Dim Status As Long
Status = ThisDisplay.ReplaceHost("Host1", "Host2")
'if Status <> 0 then there was no replacement performed or an error occurs

If Status <> 0 Then
 MsgBox "No replacements"
Else
 MsgBox "Tags replaced"
End If
```

Function ReplaceHostEx(OldHostNameSubstring As String, NewHostNameSubstring As String, MatchCase As Boolean, MatchWholeWord As Boolean) As Long

Заменяет имя узла в строках тегов и пути к файлам. Поддерживает регистр символов, групповые символы и флаг MatchWholeWord. Возвращает 0 при успешном выполнении и HRESULT – при неудачном.

Function ReplaceTag(OldSubstring As String, NewSubstring As String) As Long

Заменяет теги объектов динамики во всей экранной форме. Заменяет **OldSubstring** на **NewSubstring** в определенных ситуациях. Возвращает 0 при успешном выполнении и HRESULT – при неудачном.

Пример применения метода:

```
'VBA example, works only in configure mode
'replaces tag name in whole display

Dim Status As Long
Status = ThisDisplay.ReplaceTag("gfwsim.ramp.float",
"gfwsim.sine.double")
'if Status <> 0 then there was no replacement
'performed or an error occurs
If Status <> 0 Then
 MsgBox "No replacements"
Else
 MsgBox "Tags replaced"
End If
```

Function ReplaceStringInString (StringToReplace As String, OldSubstring As String, NewSubsring As String) As Long

Заменяет **OldSubstring** на **NewSubstring** в **StringToReplace**. Возвращает 0 при успешном выполнении и HRESULT – при неудачном.

Sub RestoreWindow()

Восстанавливает размер и положения окна экранной формы до первоначальных. (Не поддерживается для экранных форм, исполняющихся в объектах **GWXview32**).

Пример VBA

```
ThisDisplay.RestoreWindow
```

Пример VB

```
ThisDisplay.RestoreWindow
```

Function SelectPaletteColor(SelectedColor As OLE_COLOR) As Boolean

Открывает диалоговую панель цветовой палитры GraphWorX32. Аргумент **SelectedColor** после выполнения метода будет содержать цвет, выбранный пользователем. В случае, если пользователь щелкнул на кнопке **OK** диалоговой панели, возвращается TRUE. Если пользователь щелкнул на кнопке **Отмена** (Cancel) – возвращается FALSE. (Не поддерживается для экранных форм, исполняющихся в объектах **GWXview32**).

Пример VBA

```
Dim color As OLE_COLOR
    Call ThisDisplay.SelectPaletteColor(color)
    MsgBox "OLE color is defined as: " + CStr(color)
```

VB пример:

```
    Dim color
    Call ThisDisplay.SelectPaletteColor2(color)
    MsgBox "OLE color is defined as: " + CStr(color)
```

Function SelectPaletteColorRGB(red, green, blue) As Boolean

Открывает диалоговую панель цветовой палитры и позволяет выбрать цвет. Returns Возвращает TRUE если цвет выбран, иначе FALSE.
Любой цвет, отображаемый на мониторе, состоит из 3 компонентов: красный, зеленый и синий. Каждый из компонентов может иметь диапазон интенсивности от 0 (компонента нет) до 255 (максимум). Когда пользователь выбирает цвет, три параметра определяются значением интенсивности красного, зеленого и синего в выбранном цвете.
Например, если был выбран ярко желтый цвет, значения будут следующие:
Red=255
Green=255
Blue=0
Метод может быть вызван в VBA, только если тип параметров определен как Variant.

Пример VBA:

```
    Dim lr As Variant
    Dim lg As Variant
    Dim lb As Variant

    r = 0
    g = 255
    b = 255

    Call ThisDisplay.SelectPaletteColorRGB(lr, lg, lb)

    MsgBox "RGB color is defined as: " + _
            " " + CStr(lr) + _
            " " + CStr(lg) + _
            " " + CStr(lb)
```

Пример VB:

```
  Dim r
   Dim g
     Dim b
    r = 0
    g = 255
    b = 255
    Call ThisDisplay.SelectPaletteColorRGB(r, g, b)

    MsgBox "RGB color is defined as " + _
                               CStr(r) + " " + CStr(g) + " " + CStr(b)
```

Function SetAliasDefinition(aliasName As String, newDefinition As String) As Boolean

Выполняет настройку псевдонима для всех динамических объектов экранной формы, с которым он связан. Данный метод удобен для переприсоединения источников данных с графическими объектами экранной формы "на-лету" в режиме *Исполнение*. Возвращает FALSE, если в экранной форме не обнаружены заданные имена псевдонимов, и TRUE, – в противном случае. (См. также **GwxVisible::SetAliasDefinition** и **GwxDynamic::SetAliasDefinition**).

Пример VBA

```
Dim strAliasName As String
    strAliasName = "SourceAlias1"

    Call ThisDisplay.setAliasDefinition(strAliasName, _

   "{{gfwsim.cosine.double}}*20")

    MsgBox "The new definition for the alias" + _
        " ""SourceAlias1"" is {{gfwsim.cosine.double}}*20."
```

Function SetAliases(commandString As String) As Boolean

Выполняет настройку всех псевдонимов для всех динамических объектов экранной формы. Данный метод удобен для переприсоединения источников данных с графическими объектами экранной формы "на-лету" в режиме *Исполнение*. Возвращает FALSE, если в экранной форме не обнаружены заданные имена псевдонимов, и TRUE, – в противном случае. Аргумент **commandString** должен иметь следующий формат:

ИмяПсевдонима1<TAB>ОпределениеПсевдонима1<CR-LF>ИмяПсевдонима2<TAB>
ОпределениеПсевдонима2<CR-LF> … и т.д.

Пример

```
'Compose a command string:
    'See GwxDisplay::SetAliasDefinition

    Dim strAliasCommand As String

    strAliasCommand = "SourceAlias1" + Chr(9) + _
                      "100+abs({{gfwsim.sine.double}})*500"
    'Chr(13) is dividing each command line.

    MsgBox "Now we define another source for the SourceAlias1: " + _
        "100+abs({{gfwsim.sine.double}})*500"
    Call ThisDisplay.SetAliases(strAliasCommand)
```

Function SetAliasesFromFile(filename As String) As Boolean

Выполняет настройку всех псевдонимов для всех динамических объектов экранной формы. Аргумент должен содержать имя текстового файла привязки (определения) псевдонимов, каждая строка которого содержит имя псевдонима и, отделенное символом табуляции, имя источника данных . Данный файл может быть создан текстовым редактором типа *Notepad* (*Блокнот*), экспортирован из Excel, а также создан с помощью **Редактора файла псевдонимов** GraphWorX32.

Пример VBA

```
Dim strTextFileName As String

    'Specify a text file. Take care all
    'of the white spaces (only TABS) dividing the strings.
    'Set to use real tabs in your editor.
    'Template is:
    'aliasName1<TAB>aliasDef1<CR-LF>
    'aliasName2<TAB>aliasDef2<CR-LF>
    '....
    '..
    '.
    strTextFileName = "AliasCommandString.txt"

    MsgBox "Now we define another source for the Local alias." + _
        "See the value in the top right hand corner of the display."


    Call ThisDisplay.SetAliasesFromFile(strTextFileName)
```

VB пример

```
    Dim strTextFileName
        'Specify a text file. Take care all
    'of the white spaces (only TABS) dividing the strings.
```

```
    'Set to use real tabs in your editor.
    'Template is:
    'aliasName1<TAB>aliasDef1<CR-LF>
    'aliasName2<TAB>aliasDef2<CR-LF>
    '....
    '..
    '.
    strTextFileName = "AliasCommandString.txt"

    MsgBox "Now we define another source for the Local alias." + _
        "See the value in the top right hand corner of the display."


    Call ThisDisplay.SetAliasesFromFile(strTextFileName)
```

Sub SetDisplayDimensions(width As Long, height As Long)

Устанавливает размеры рабочей области текущей загруженной экранной формы.

Пример VBA

```
Sub SetDisplayDimensions(o As GwxPick)
    'Declare coordinates as long values
    Dim widthDisplay As Long
    Dim heightDisplay As Long

    Dim widthDisplayMem As Long
    Dim heightDisplayMem As Long

    'Remember current dimensions:
    Call ThisDisplay.GetDisplayDimensions(widthDisplayMem, _
       heightDisplayMem)

    'Set new values:
    widthDisplay = 600
    heightDisplay = 1000

    'Set display parameters:

    Call ThisDisplay.SetDisplayDimensions(widthDisplay, _

 heightDisplay)

    'and refresh the window:
    ThisDisplay.RefreshWindow

    MsgBox "When you click on OK the dimensions" + _
        " will be restored." + _
        Chr(13) + " (width = " + _
        Conversion.CStr(widthDisplayMem) + _
        " and height = " + _
        Conversion.CStr(heightDisplayMem) + _
        ")"

    'Again, set display parameters:
    Call ThisDisplay.SetDisplayDimensions(widthDisplayMem, _
    heightDisplayMem)
    'and once more refresh the window:

    ThisDisplay.RefreshWindow
End Sub
```

VB пример:

```
'Declare coordinates as long values
    Dim widthDisplay
    Dim heightDisplay

    Dim widthDisplayMem
    Dim heightDisplayMem

    'Remember current dimensions:
         'TO DO!
         Call ThisDisplay.GetDisplayDimensions2(widthDisplayMem, _
```

```
                heightDisplayMem)
   widthDisplayMem = 1036
    heightDisplayMem = 767

    'Set up new values:
    widthDisplay = 600
    heightDisplay = 400

    'Set display parameters:
    Call ThisDisplay.SetDisplayDimensions(widthDisplay, _

heightDisplay)
    'and refresh the window:
    ThisDisplay.RefreshWindow

    MsgBox "When you click on OK the dimensions will be restored."

    'Again, set display parameters:
    Call ThisDisplay.SetDisplayDimensions(widthDisplayMem, _
                heightDisplayMem)

     'and once more refresh the window:
    ThisDisplay.RefreshWindow
```

Sub SetEventScriptText(Event As GWXEVENTNAME, text As String)

GraphWorX32 8.0 позволяет выполнять по событиям сценарии VBScript и JScript. Сценарии, связанные с событием, выполняются при возникновении события (например, после запуска приложения или перед запросом данных для слоя). Сценарии могут редактироваться с помощью новой интегрированной панели инструментов или с помощью интерфейса **OLE automation**. Сценарий может быть написан с использованием языков VBScript or Jscript независимо от другого события. Эта функция позволяет изменять исходный код, связанный с определенным событием.

Sub SetEventScriptType(Event As GWXEVENTNAME, Type As GWXSCRIPTTYPE)

GraphWorX32 позволяет выполнять по событиям сценарии VBScript и JScript. Сценарии, связанные с событием, выполняются при возникновении события (например, после запуска приложения или перед запросом данных для слоя). Сценарии могут редактироваться с помощью новой интегрированной панели инструментов или с помощью интерфейса **OLE automation**. Сценарий может быть написан с использованием языков VBScript or Jscript независимо от другого события. Эта функция позволяет изменять исходный код, связанный с определенным событием.

```
'Works in configure and runtime mode
'Creates VB script on event PostRuntimeStart

ThisDisplay.SetEventScriptType _
PostRuntimeStart, ScriptTypeVBScript

ThisDisplay.SetEventScriptText _
PostRuntimeStart, "MsgBox " & Chr(34) & _
"Test OK" & Chr(34)

Соответствующий пример VB:
ThisDisplay.SetEventScriptText 23,"msgbox ""test"" "
ThisDisplay.SetEventScriptType 23,1
```

Function SetFileVersion(NewFileVersion As GWXSETFILEVERSION) As GWXGETFILEVERSION

Функция **SetFileVersion()** устанавливает версию экранной формы для следующего сохранения. Обратите внимание, что количество версий, которые можно установить для сохранения, меньше, чем может быть определено при чтении файла. Другими словами,

GraphWorX32 определяет много версий файлов, но позволяет сохранять только в некоторых поздних версиях, сохранение в версии 5.20 и более старых не поддерживается. При попытке сохранения в неподдерживаемой старой версии GraphWorX32 сохраняет файл в версии `6.0`

SetFileVersion() input values

- `2 .. SetVer600SP`
- `3 .. SetVerMGraphics40`
- `4 .. SetVer610`
- `9999 .. SetVerLatest (&H270F)`

Интерфейс OLE automation предоставляет большие возможности при автоматизации разработки экранных форм (например, если необходимо повысить или понизить версию для группы экранных форм до определенной версии, или внести изменения в экранную форму с сохранением существующей версии).

Пример VBA:

```
    'Keep the latest version.
    MsgBox "Keep the latest version."
    ThisDisplay.SetFileVersion (SetVerLatest)
```

Пример VB:

```
    'Keep the latest version.
    MsgBox "Keep the latest version."
    ThisDisplay.SetFileVersion (SetVerLatest)
```

Пример VB:

```
Private Sub DisplayVersion()
    ' Specify the path and filename of an existing display
    Const strPath As String = _
"C:\PathToYourDisplay\"
    Const strFilename As String = _
strPath & "YourDisplay.gdf"
    Const SetVer600SP as Long = 2
    Const VerLatest as Long = 9999

    ' Create GWX instance
    Dim g As New Gwx32.GwxDisplay

    ' Check if GWX was created
    If g Is Nothing Then
        MsgBox "Failed to create GWX"

        Exit Sub
    End If

    ' Show GWX in the front and load the requested display
    Call g.ShowWindow
    Call g.BringWindowToTop
    Call g.FileNew
    Call g.FileOpen(strFilename)

    ' Check the file version
    Dim ver As long
    ver = GetFileVersion(g)

    ' If this is the latest version, save in the
    ' version 6.00
    If (ver = VerLatest) Then
        g.SetFileVersion(SetVer600SP)
        g.FileSave
```

```
    End If

    ' Exit GWX, the FileNew suppresses the message box
    ' if to save changes.
    g.FileNew
    g.ExitApplication
End Sub

' Get file version and trace the current version in the
' Immediate window for debugging purposes.
Private Function GetFileVersion(g As GwxDisplay) As Long
    Const OpenFileError as Long = -1
    Const Ver520 as Long = 0
    Const Ver600 as Long = 2
    Const Ver600SP as Long = 2
    Const VerMGraphics40 as Long = 2

    Const Ver610 as Long = 2
    Const VerLatest as Long = 9999

    Dim ver As long

    ' Check the version
    ver = g.GetFileVersion
    Select Case (ver)
        Case OpenFileError:
            Debug.Print "Version: FileOpenError"
        Case Ver520:
            Debug.Print "Version: Ver520"
        Case Ver600:
            Debug.Print "Version: Ver600"
        Case Ver600SP:
            Debug.Print "Version: Ver600SP"
        Case VerMGraphics40:

            Debug.Print "Version: VerMGraphics40"
        Case Ver610:
            Debug.Print "Version: Ver610"
        Case VerLatest:
            Debug.Print "Version: VerLatest"
    End Select

    ' Check if the display contains VBA
    If g.GetFileContainsVBA Then
        Debug.Print "VBA: Yes"
    Else
        Debug.Print "VBA: No"
    End If

    GetFileVersion = ver
End Function
```

Function SetGASThemes(Themes As String) As Boolean

Устанавливает тему Глобальных псевдонимов в формате, определенном в Навигаторе Тем Глобальных псевдонимов, поэтому рекомендуется использовать Навигатор для создания требуемой строки и затем вставить строку в функцию через буфер обмена.

```
'VBA/VB Script example, works only in runtime mode
'example for setting and getting themes (Floors theme is changed
in child windows)
ThisDisplay.SetGASThemes "Buildings=BuildingB;Floors=Floor2/+1;"
MsgBox ThisDisplay.GetGASThemes
```

Function SetGradientFill(style As GWXGRADIENTSTYLE, color1 As OLE_COLOR, color2 As OLE_COLOR, isTwoColor As Boolean, brightness As Single, reverseColors As Boolean, steps As Integer, horizontalOffset As Single, verticalOffset As Single) As Boolean

Устанавливает градиентную заливку для фона экранной формы.
Параметры, передаваемые функции:
GWXGRADIENTSTYLE style – перечислитель, определяющий **стиль** градиентной заливки. Возможные значения:

- GradientHorizontal (горизонтально) = 0
- GradientVertical (вертикально) = 1
- GradientSquare (квадрат) = 2

OLE_COLOR color1 – основной цвет заливки графического объекта
OLE_COLOR color2 – цвет второго воображаемого источника света, освещающего объект
boolean isTwoColor – признак использования двух цветов для формирования заливки. Если установлен в True, параметр brightness (яркость) игнорируется, параметр color2 – воспринимается. Если установлен в False – наоборот.
float brightness – определяет яркость воображаемого основного источника света в диапазоне от 0.0 до 1.0
boolean reverseColors – признак взаимной перестановки воображаемых источников света. При использовании одного источника света значение True приводит к зеркальному перемещению источника.
int steps – целое число, определяющее плавность перехода фрагментов градиентной заливки. Может принимать следующие значения: 5, 10, 15, 20, 25, 30, 40, 50, 75, 100, 150, 200, 250
float horizontalOffset – определяет степень сдвига воображаемого источника по горизонтали в сторону своего зеркального отображения в диапазоне от –1.0 до 1.0.
float verticalOffset – определяет степень сдвига воображаемого источника по вертикали в сторону своего зеркального отображения в диапазоне от 0.0 до 1.0.

Пример

```
Sub Gradient(o As GwxPick)
Dim result As Boolean
result = ThisDisplay.SetGradientFill _
(GradientHorizontal, RGB(0, 128, 255), RGB(255, 255, 255), False, 0.65, _
False, 5, -1, -1)
End Sub
```

Function SetHeadObject(objectName As String) As Boolean

Делает выделенным объект, имя которого задано аргументом. В случае успеха возвращается TRUE, и FALSE, – в случае неудачи.

Пример VBA

```
'If you want to select an object with the name "Object"
ThisDisplay.SetHeadObject ("Object")
```

Function SetLayerOverrideHide(layerName As String, hidden As Boolean) As Boolean

Устанавливает параметр **Явно скрыть слой** для определенного слоя.

Sub SetModifiedFlag(Modified As Boolean)

Флаг изменения используется для определения, вносились ли изменения в экранную форму со времени последнего сохранения. Если Флаг изменения установлен (true), GraphWorX32 будет запрашивать о необходимости сохранения изменений перед закрытием экранной формы или загрузкой новой.

Пример VBA:

```
ThisDisplay.SetModifiedFlag (True)
    MsgBox "The modified flag has been set to ""True""."
```

Function SetRuntimeFocusProperties(ShowHandCursor As Boolean, ShowFocusRectangle As Boolean, fillColor As OLE_COLOR, BorderColor As OLE_COLOR) As Boolean

Устанавливает атрибуты обозначения фокуса ввода в режиме *Исполнение*.

Пример

```
Dim result As Boolean
result = _
ThisDisplay.SetRuntimeFocusProperties _
(True, False, RGB(127, 0, 127), RGB(127, 127, 0))
```

Sub SetPrintWhiteBackground(WhiteBackground As Boolean, PrintArea As Long)

Устанавливает вид области и белый фон при печати

VBA пример:

```
'Set an appropriate display layout:
'PV_NOCHANGE = 0,
'PV_HOME = 1,
'PV_FITTOWINDOW = 2,
'PV_WORLD = 3
MsgBox "The display layout is: PV_NOCHANGE"
Call ThisDisplay.SetPrintWhiteBackground(True, 0)
```

VB пример:

```
Call ThisDisplay.SetPrintWhiteBackground(True, 0)

MsgBox "The white background is set to True."
```

Function SetRuntimeUpdateRate(updateRate As Long) As Boolean

Устанавливает период обновления значений OPC-тегов для экранной формы (в миллисекундах).

Пример

```
Dim result As Boolean
result = ThisDisplay.SetRuntimeUpdateRate(100)
```

Function SetSystemWideLanguage(Language As String, LanguageSubset As String, LocaleID As Long, SynchronizeLCID As Boolean) As Boolean

Устанавливает язык GENESIS и язык документа, LocaleID для приложения и флаг необходимости синхронизации языка документа с приложением.

Language ID

```
---------------------------
Czech           1029
Dutch           2067
English         1033
French          1036
German    1031
Italian         1040
Russian   1049
Spanish   1034
```

Пример VBA:

```
Dim strLanguage As String
Dim strLanguageSubSet As String
Dim lLocaleID As Long
Dim bSynchronizeLCID As Boolean

strLanguage = "Czech"
strLanguageSubSet = "Czech.Test"
lLocaleID = 1029
```

```
bSynchronizeLCID = False
    Call ThisDisplay.SetSystemWideLanguage( _
                                strLanguage, _
                                strLanguageSubSet, _
                                lLocaleID, _
                                bSynchronizeLCID)
```

Function SetTooltipProperties(ShowDynamicTips As Boolean, ShowDynamicObjectName As Boolean, ShowDynamicObjectDescription As Boolean, ShowDataSourceName As Boolean, ShowDataSourceValue As Boolean, ShowStaticTooltips As Boolean, ShowStaticObjectName As Boolean, ShowStaticObjectDescription As Boolean) As Boolean

Устанавливает свойства окон подсказок для экранной формы.

Пример

```
Dim result As Boolean
'В подсказке будет выводиться следующая информация:
'- Имя источника данных,
'- Значение источника данных
result = ThisDisplay.SetTooltipProperties _
(True, False, False, True, True, False, False, False)
```

Function SetTransparency(EnableTransparency As Boolean, transparentColor As OLE_COLOR, EnableAlpha As Boolean, AlphaBlending As Integer) As Boolean

Устанавливает прозрачность в режиме исполнения. Метод может быть вызван только в операционной системе Windows 2000 или выше.

```
'VBA example, works in runtime mode
'this example toggles transparency and transluency settings

Dim   TE As Boolean, TC As OLE_COLOR, AE As Boolean, _
      AB As Integer
ThisDisplay.GetTransparency TE, TC, AE, AB
'following lines enable/disable transparency and transluency
If TE Then TE = False Else TE = True
If AE Then AE = False Else AE = True
'if transluency is enabled, sets alpha blending value to 100 of
255
If AE Then AB = 100 Else AB = 0
'applies changed values
ThisDisplay.SetTransparency TE, TC, AE, AB
MsgBox "Transparency: " & TE & vbCrLf & "Color: " _
      & TC & vbCrLf & "Transluency: " & _
      AE & vbCrLf & "Color: " & AB
```

Sub SetVCRMode(Enable As Boolean)

Эта функция устанавливает режим воспроизведения VCRWorX. (Могут ли воспроизводиться данные из базы HDA).

Пример VBA:

```
If Not ThisDisplay.GetVCRMode Then
    ThisDisplay.SetVCRMode (True)
End If
```

Пример VB:

```
If Not ThisDisplay.GetVCRMode Then
    ThisDisplay.SetVCRMode (True)
End If
```

Sub SetViewDimensions(left As Long, top As Long, width As Long, height As Long)

Устанавливает размеры и положение видимой части рабочей области экранной формы GraphWorX32 (области просмотра). Установка размеров области просмотра может использоваться для полного или фрагментарного увеличения или увеличения экранной формы.

Пример VBA

```
Sub SetViewDimensions(o As GwxPick)
    'Declare coordinates as long values
    Dim viewLeft As Long
    Dim viewTop As Long
    Dim viewWidth As Long
    Dim viewHeight As Long

    'Backup values
    Dim bckLeft As Long
    Dim bckTop As Long
    Dim bckWidth As Long
    Dim bckHeight As Long

    'Remember current dimensions:
    Call ThisDisplay.GetViewDimensions(bckLeft, _
                                       bckTop, _

                                       bckWidth, _
                                       bckHeight)

    'Set new dimensions:
    viewLeft = 0
    viewTop = 0
    viewWidth = 1036
    viewHeight = 767

    'Set the view parameters:
    Call ThisDisplay.SetViewDimensions(viewLeft, _
                                       viewTop, _
                                       viewWidth, _
                                       viewHeight)
    'and refresh the window:

    ThisDisplay.RefreshWindow

    MsgBox "When you click on OK the view dimensions" + _
        " will be restored to the previous ones." + _
        Chr(13) + " (Left = " + _
        Conversion.CStr(bckLeft) + _
        " , Top = " + _
        Conversion.CStr(bckTop) + _
        " , Width = " + _
        Conversion.CStr(bckWidth) + _
        " and Height = " + _
        Conversion.CStr(bckHeight) + _
        ")"

    'Again, set the view parameters:

    Call ThisDisplay.SetViewDimensions(bckLeft, _
                                       bckTop, _
                                       bckWidth, _
                                       bckHeight)

    'and once more refresh the window:
    ThisDisplay.RefreshWindow
End Sub
```

Пример VB:

```
    'Declare coordinates as long values
    Dim viewLeft
    Dim viewTop
```

```
Dim viewWidth
Dim viewHeight
'Backup values
Dim bckLeft
Dim bckTop
Dim bckWidth
Dim bckHeight

'Remember current dimensions:
'TO DO!
     Call ThisDisplay.GetViewDimensions2(bckLeft, _
                                         bckTop, _
                                         bckWidth, _
                                         bckHeight)


     bckLeft = 0
     bckTop = 0
     bckWidth = 1131
     bckHeight = 767


'New dimensions: <0, 0, 502, 341>
viewLeft = 0
viewTop = 0
viewWidth = 502
viewHeight = 341

'Set the view parameters:
Call ThisDisplay.SetViewDimensions(viewLeft, _
                                   viewTop, _
                                   viewWidth, _
                                   viewHeight)
'and refresh the window:

ThisDisplay.RefreshWindow

MsgBox "When you click on OK the view dimensions" + _
    " will be restored to the previous ones."


'Again, set the view parameters:
Call ThisDisplay.SetViewDimensions(bckLeft, _
                                   bckTop, _
                                   bckWidth, _
                                   bckHeight)

'and once more refresh the window:
ThisDisplay.RefreshWindow
```

Sub SetWindowDimensionsPercent(left As Single, top As Single, width As Single, height As Single)

Устанавливает размеры и положение главного окна GraphWorX32 в процентах от размеров всего экрана. Устанавливаемые значения лежат в диапазоне от 0,0 до 1,0. (Не поддерживается для экранных форм, исполняющихся в объектах GWXview32).

Пример VBA

```
Sub SetWndDimPercent(o As GwxPick)
    'Declare coordinates as Single values
    Dim viewLeft As Single
    Dim viewTop As Single
    Dim viewWidth As Single
    Dim viewHeight As Single

    'Set new values:
    viewLeft = -0.002
    viewTop = -0.0025
    viewWidth = 0.45
    viewHeight = 0.5
```

```
    'Set the view parameters:
    Call ThisDisplay.SetWindowDimensionsPercent(viewLeft, _
                                                viewTop, _

                                                viewWidth, _
                                                viewHeight)
    'and refresh the window:
    ThisDisplay.RefreshWindow

    MsgBox "When you click on OK the view dimensions" + _
        " will be restored to the previous ones."

    'Again, set the view parameters:
    Call ThisDisplay.SetWindowDimensionsPercent(1 / viewLeft, _
                                                1 / viewTop, _

                                                1 / viewWidth, _
                                                1 / viewHeight)

    'and once more refresh the window:
    ThisDisplay.RefreshWindow
    'Rise window from the task bar
    ThisDisplay.MaximizeWindow

End Sub
```

Пример VB:

```
'Declare coordinates
    Dim viewLeft
    Dim viewTop
    Dim viewWidth
    Dim viewHeight
       'Set new dimensions:
    viewLeft = -0.002
    viewTop = -0.0025
    viewWidth = 0.5
    viewHeight = 0.45

    'Set the view parameters:
    Call ThisDisplay.SetWindowDimensionsPercent(viewLeft, _
                                                viewTop, _
                                                viewWidth, _
                                                viewHeight)
    'and refresh the window:
    ThisDisplay.RefreshWindow

    MsgBox "When you click on OK the view dimensions" + _
        "will be restored to the previous ones."

    'Again, set the view parameters:
    Call ThisDisplay.SetWindowDimensionsPercent(1 / viewLeft, _
                                                1 / viewTop, _
                                                1 / viewWidth, _
                                                1 / viewHeight)


    'and once more refresh the window:
    ThisDisplay.RefreshWindow
    'Rise window from the task bar
    ThisDisplay.MaximizeWindow
```

Sub SetWindowDimensionsPixels(left As Long, top As Long, width As Long, height As Long)

Устанавливает размеры и положение главного окна GraphWorX32 в пикселах. (Не поддерживается для экранных форм, исполняющихся в объектах GWXview32).

Пример VBA

```
Sub SetWndDimPixels(o As GwxPick)
```

```
    'Declare coordinates as Long values
    Dim viewLeft As Long
    Dim viewTop As Long
    Dim viewWidth As Long
    Dim viewHeight As Long
    'Backup values
    Dim bckLeft As Long
    Dim bckTop As Long
    Dim bckWidth As Long
    Dim bckHeight As Long

    'Remember current dimensions:
    Call ThisDisplay.GetWindowDimensionsPixels(bckLeft, _
                                               bckTop, _

                                               bckWidth, _
                                               bckHeight)

    'Set new values:
    viewLeft = -2
    viewTop = -2
    viewWidth = 516
    viewHeight = 748

    'Set the view parameters:
    Call ThisDisplay.SetWindowDimensionsPixels(viewLeft, _
                                               viewTop, _
                                               viewWidth, _
                                               viewHeight)

    'and refresh the window:
    ThisDisplay.RefreshWindow

    MsgBox "When you click on OK the view dimensions" + _
        " will be restored to the previous ones." + _
        Chr(13) + " (Left = " + _
        Conversion.CStr(bckLeft) + _
        " , Top = " + _
        Conversion.CStr(bckTop) + _
        " , Width = " + _
        Conversion.CStr(bckWidth) + _
        " and Height = " + _
        Conversion.CStr(bckHeight) + _
        ")"

    'Again, set the view parameters:

    Call ThisDisplay.SetWindowDimensionsPixels(bckLeft, _
                                               bckTop, _
                                               bckWidth, _
                                               bckHeight)
    'and once more refresh the window:
    ThisDisplay.RefreshWindow
End Sub
```

Пример VB:

```
    'Declare coordinates as Long values
    Dim viewLeft
    Dim viewTop
    Dim viewWidth
    Dim viewHeight
    'Backup values
    Dim bckLeft
    Dim bckTop
    Dim bckWidth
    Dim bckHeight

    'Remember current dimensions:
    Call ThisDisplay.GetWindowDimensionsPixels2(bckLeft, _
                                               bckTop, _
```

```
                                                    bckWidth, _
                                                    bckHeight)

  bckLeft = -4
     bckTop  = -4
     bckWidth = 1160
     bckHeight = 842


    'New dimensions <0, 0, 502, 341>
    viewLeft = 0
    viewTop = 0
    viewWidth = 502
    viewHeight = 341

    'Set the view parameters:
    Call ThisDisplay.SetWindowDimensionsPixels(viewLeft, _
                                                    viewTop, _
                                                    viewWidth, _
                                                    viewHeight)

    'and refresh the window:
    ThisDisplay.RefreshWindow

    MsgBox "When you click on OK the view dimensions" + _
        " will be restored to the previous ones."

    'Again, set the view parameters:
    Call ThisDisplay.SetWindowDimensionsPixels(bckLeft, _
                                                    bckTop, _
                                                    bckWidth, _
                                                    bckHeight)
    'and once more refresh the window:
    ThisDisplay.RefreshWindow
```

Sub SetWorkingDirectory()

Пример VBA:

```
    'Sets working directory.

    Dim strWorDir As String
    strWorDir = ThisDisplay.GetWorkingDirectory
    MsgBox "Keep the current directory: " + _
        strWorDir
    ThisDisplay.SetWorkingDirectory (strWorDir)
```

Пример VB:

```
    'Sets working directory.
        Dim strWorDir
    strWorDir = ThisDisplay.GetWorkingDirectory
    MsgBox "Keep the current directory: " + _
        strWorDir
    ThisDisplay.SetWorkingDirectory (strWorDir)
```

Sub ShowWindow()

Отображает главное окно GraphWorX32.

Пример VBA:

```
'Declare a new GwxDisplay
    Dim newGwxDisplay As New Gwx32.GwxDisplay
    MsgBox "New window is launched..."

    'Show the display
    Call newGwxDisplay.ShowWindow

    MsgBox "This function works properly only" + _
        " within a dialog which follows."
```

Пример VB:

```
'Declare a new GwxDisplay
Dim newGwxDisplay

  MsgBox "New window is launched..."

  Set newGwxDisplay = CreateObject("Gwx32.Display")

'Show the display
  Call newGwxDisplay.ShowWindow
```

Sub StartRuntime()

Переводит GraphWorX32 в режим *Исполнение*.

Пример VBA

```
Sub StartRuntime(o As GwxButton)
    'Switch display to runtime mode.
    ThisDisplay.StartRuntime
End Sub
```

Пример VB

```
  'You can use it within the function "StopRuntime".
   'Switch to design environment
  ThisDisplay.StopRuntime

    MsgBox "The display is going to quit runtime mode." + _
                " On submit the function StartRuntime is invoked."

  'Switch to runtime environment
  ThisDisplay.StartRuntime
```

Sub StopRuntime()

Переводит GraphWorX32 в режим *Разработка*.

Пример:

```
'Процедура вызывается в экранной форме-контейнере по нажатию кнопки и переключает
'режим экранной формы, загруженной в ActiveX. Расположена в отдельном модуле.
Sub StartStopRuntimeForInner(o As GwxPick)

Dim gwxDisplayInner As GwxDisplay 'Экранная форма, загруженная в GWXview32 ActiveX

'Получаем объект GwxDisplay из модуля ThisDisplay
Set gwxDisplayInner = ThisDisplay.GWXview321.GetDisplay()

If Not (gwxDisplayInner Is Nothing) Then
  If gwxDisplayInner.IsRuntimeMode Then
    gwxDisplayInner.StopRuntime 'Останов режима Исполнение
  Else
    gwxDisplayInner.StartRuntime 'Переход в режим Исполнение
  End If
Set gwxDisplayInner = Nothing
End If
End Sub
```

Пример VBA:

```
Sub StopRuntime(o As GwxPick)
  'Switch to design environment
  ThisDisplay.StopRuntime
End Sub
```

Пример VB:

```
  'Switch to design environment
  ThisDisplay.StopRuntime
```

Function TestCustomSecurityItem(customString As String) As Boolean

Возвращает TRUE если текущий пользователь имеет права для для определенного пункта системы безопасности, иначе FALSE.

```
GWXSECURITYWRITESTATUS:
WriteUninitialized = -1,
WriteEnabled = 0,
WriteDisabled = 1,
WriteCritical = 2
```

Function ToggleLayerOverrideHide(layerName As String) As Boolean

Переключает значение параметра Явно скрыть слой для заданного слоя.

Function ToggleRuntime() As Boolean

Переключает текущий режим работы GraphWorX32. Возвращает TRUE, если произошел переход из режима *Разработка* в режим *Исполнение*. Возвращает FALSE, если произошел переход в режим *Разработка*.

Пример VBA:

```
'Declare flag to watch the mode:
    Dim modeFlag As Boolean

    'Set the flag according to a current mode:
    modeFlag = ThisDisplay.ToggleRuntime
```

Пример VB:

```
'Declare flag to watch the mode:
    Dim modeFlag As Boolean

    'Set the flag according to a current mode:
    modeFlag = ThisDisplay.ToggleRuntime
```

Function UngroupSymbol(objectName As String) As Boolean

Разгруппировывает объект типа **GwxSymbol** с заданным именем. Хотя данная функция разбивает графический объект на примитивы, однако связанный с ним объект OLE Automation остается сгруппированным и доступным. Возвращает TRUE в случае успеха, и FALSE, – в противном случае. (Доступен только в режиме *Разработка*).

Function VCRControlPanel(Show As Boolean) As Boolean

Открывает панель управления VCR

Пример VBA:

```
    ThisDisplay.VCRControlPanel (True)
```

Sub ViewBoxZoom()

Выполняет операцию "**Увеличение по выделенной области**". (Не поддерживается в ActiveX ToolWorX).

Пример VBA

```
ThisDisplay.ViewBoxZoom
```

Пример VB

```
ThisDisplay.ViewBoxZoom
```

Sub ViewFitToWindow()

Выполняет операцию "**Подобрать**", когда все графические объекты экранной формы оказываются в видимой области. (Не поддерживается в ActiveX ToolWorX).

Пример VBA

```
ThisDisplay.ViewFitToWindow

    MsgBox "Zoom back."
    ThisDisplay.ViewHome
```

Пример VB :

```
ThisDisplay.ViewFitToWindow

MsgBox "Zoom back."
ThisDisplay.ViewHome
```

Sub ViewHome()

Выполняет операцию **Вид -- Основной**, когда видимая область экрана принимает размеры, которые она имела в момент последнего сохранения на диске. (Не поддерживается в ActiveX ToolWorX).

Пример VBA

```
Call ThisDisplay.SetViewDimensions(0, 0, 200, 600)
  MsgBox "Zoom back to previous display." + _
    Chr(13) + "(ViewHome)"
  ThisDisplay.ViewHome
```

Пример VB :

```
  Call ThisDisplay.SetViewDimensions(0, 0, 200, 600)
     MsgBox "Zoom back to previous display." + _
    Chr(13) + "(ViewHome)"
  ThisDisplay.ViewHome
```

Sub ViewPan(offsetX As Long, offsetY As Long)

Производит сдвиг видимой области вдоль горизонтальной и вертикальной осей на заданное количество единиц. (Не поддерживается в ActiveX ToolWorX).

Пример VBA

```
'Set the view parameters:
    Call ThisDisplay.SetWindowDimensionsPercent(0.002, _
                                                0.0025, _
                                                0.45, _
                                                0.5)
    MsgBox "The window pans the view on submit."

    Call ThisDisplay.ViewPan(70, 200)

    MsgBox "Confirm return to the primary display."

    'Restore window dimensions.
    Call ThisDisplay.SetWindowDimensionsPercent(1 / 0.002, _

                                                1 / 0.0025, _
                                                1 / 0.45, _
                                                1 / 0.5)
    ThisDisplay.MaximizeWindow
```

Пример VB :

```
    'Set the view parameters:
    Call ThisDisplay.SetWindowDimensionsPercent(0.002, _
                                                0.0025, _
                                                0.45, _
                                                0.5)
    MsgBox "The window pans the view on submit."
```

```
    Call ThisDisplay.ViewPan(70, 200)

    MsgBox "Confirm return to the primary display."

    'Restore window dimensions.
    Call ThisDisplay.SetWindowDimensionsPercent(1 / 0.002, _

                                                1 / 0.0025, _
                                                1 / 0.45, _
                                                1 / 0.5)
    ThisDisplay.MaximizeWindow
```

Sub ViewShowWholeDisplay()

Подбирает размеры видимой области экранной формы таким образом, чтобы они соответствовали заданным в свойствах экранной формы. (Не поддерживается в ActiveX ToolWorX).

Пример VBA

```
Sub ViewShowWholeDisplay(o As GwxPick)
    Call ThisDisplay.SetWindowDimensionsPercent(0.5, 0.5, 0.5, 0.5)

    MsgBox "View the whole display."

    ThisDisplay.ViewShowWholeDisplay

    MsgBox "Restore window."

    ThisDisplay.ViewHome
    ThisDisplay.MaximizeWindow
End Sub
```

Пример VB:

```
    Call ThisDisplay.SetWindowDimensionsPercent(0.5, 0.5, 0.5, 0.5)
    MsgBox "View the whole display."
    ThisDisplay.ViewShowWholeDisplay

    MsgBox "Restore window."
    ThisDisplay.ViewHome
    ThisDisplay.MaximizeWindow
```

Sub ViewUnzoom()

Отменяет последнюю выполненную команду изменения размеров видимой области. (Не поддерживается в ActiveX ToolWorX).

Пример VBA

```
The display is 1.5 times bigger.
    ThisDisplay.ViewZoomPercent (0.5)

    MsgBox "Confirm to call ViewUnzoom."

    'Undo zoom.
    ThisDisplay.ViewUnzoom
```

Пример VB:

```
    'The display is 1.5 times bigger.
    ThisDisplay.ViewZoomPercent (1.5)

    MsgBox "Confirm to call ViewUnzoom."
```

```
'Undo zoom.
ThisDisplay.ViewUnzoom
```

Sub ViewZoomCustomDialog()

Вызывает диалоговую панель **Произвольный масштаб**, которая позволяет ввести в процентах степень увеличения или уменьшения видимой области экранной формы. (Не поддерживается в ActiveX ToolWorX).

Пример VBA

```
ThisDisplay.ViewZoomCustomDialog
```

Пример VB

```
ThisDisplay.ViewZoomCustomDialog
```

Sub ViewZoomPercent(newZoom As Single)

Изменяет размеры видимой области экранной формы в соответствии с величиной, заданной аргументом. (Не поддерживается в ActiveX ToolWorX).

Пример VBA

```
Dim Zoom

    Zoom = 2
    Call ThisDisplay.ViewZoomPercent(Zoom)

    MsgBox "Zoom to home."
    ThisDisplay.ViewHome
```

Пример VB:

```
        Dim Zoom
        Zoom = 2
    Call ThisDisplay.ViewZoomPercent(Zoom)

    MsgBox "Zoom to home."
    ThisDisplay.ViewHome
```

GwxVisible

Классы всех видимых объектов GraphWorX32 (эллипсы, прямоугольники и т.д.) унаследованы от класса **GwxVisible**. Другими словами, все видимые объекты GraphWorX32 имеют свойства и методы **GwxVisible**.

Свойства

Angle As Single

Возвращает/устанавливает угол поворота объекта относительно его первоначального положения. Значение угла задается в градусах.

edgeStyle As GWX3DEDGESTYLE

Тип трехмерного эффекта при отображении рамки объекта. Допустимые значения **GWX3DEDGESTYLE**:

- EdgeNone (трехмерный эффект отсутствует) = 0
- EdgeRaised (выпуклый) = 5
- EdgeEtched (вогнутый) = 6
- EdgeBump (рельефная рамка) = 9
- EdgeSunken (выпуклая рамка) = 10

fillColor As OLE_COLOR

Возвращает/устанавливает цвет заливки объекта.

hasShadow As Boolean

Возвращается TRUE, если объект отображается с тенью, и FALSE, – в противном случае.

isFilled As Boolean

Возвращается TRUE, если объект имеет заливку, и FALSE, – в противном случае.

Keyword As String

Данная строка используется для хранения разделяемого ключевого слова. Данное свойство используется при определении, поддерживает ли объект функцию обновления разделяемых объектов GraphWorX32.

lineColor As OLE_COLOR

Возвращает/устанавливает цвет линии/рамки объекта.

lineStyle As GWXLINESTYLE

Стиль отображения линии или рамки. Допустимые значения **GWXLINESTYLE**:

- LineSolid (сплошная линия) = 0
- LineDash (штрих) = 1
- LineDot (пунктир) = 2
- LineDashDot (штрих-пунктир) = 3
- LineDashDotDot (штрих-двойной пунктир) = 4
- LineNone (линия отсутствует) = 5

lineWidth As Long

Возвращает/устанавливает толщину линии/рамки объекта. Значения лежат в диапазоне от 0 до 10.

objectName As String

Возвращает/устанавливает имя видимого объекта. Имя объекта используется для идентификации объекта при вызове методов его класса. (например, **GwxDisplay::GetVisibleObjectFromName**). Обязательным условием идентификации объектов является уникальность имени каждого объекта. Если пользователь присваивает видимому объекту уже существующее имя, GraphWorX32 добавит к имени порядковый

номер (например, при создании нескольких копий объекта с именем "*Tank*", копиям будут присвоены имена "*Tank1*", "*Tank2*" и т.д.).

Selected As Boolean

Возвращается TRUE, если объект выделен, и FALSE, – в противном случае. Выделенным является объект, на котором пользователь произвел щелчок мышью. При этом вокруг объекта появляется прямоугольный контур с небольшими символами квадратной формы. Данное свойство может быть изменено только в режиме *Исполнение*.

shadowColor As OLE_COLOR

Возвращает/устанавливает цвет тени объекта.

UserCustomData As String

Данная строка используется для хранения специальных пользовательских данных. Данное свойство используется для связывания с видимым объектом дополнительной информации.

UserDescription As String

Строка описания видимого объекта. Обычно данная строка используется в качестве вспомогательной текстовой информации, выводимой в окне подсказки при нахождении курсора мыши над объектом в течение некоторого времени.

Visible As Boolean

Возвращается TRUE, если объект отображается на экране, и FALSE, – если объект скрыт.

Методы

Sub ClearGradientFill()

Отключает градиентную заливку графического объекта. Объект будет иметь цвет заливки, заданный в поле **Цвет заливки** на странице **Общие** диалоговой панели **Инспектор свойств**.

Пример

```
Dim visibleObject As Object
'Доступ через IDispatch к объектам неизвестного типа
Set visibleObject = ThisDisplay.GetVisibleObjectFromName("Rectangle1")
Call visibleObject.ClearGradientFill
Set visibleObject = Nothing
```

FlipHorizontal()

Переворачивает объект относительно горизонтальной оси.

FlipVertical()

Переворачивает объект относительно вертикальной оси

Пример

```
Dim obj as GWXRectangle
set obj=ThisDisplay.GetVisibleObjectFromName("myobject")
obj.FlipHorizontal
obj.FlipVertical
```

Function GetDataSources(includeDuplicates As Boolean)

Получает все имена тегов, связанные с видимым объектом. Возвращает массив строк (тип variant), каждая строка в котором соответствует имени тега. Опционально можно включать или не включать повторяющиеся теги в массив.

Function GetDynamicObjectFromIndex(index As Long) As Object

Возвращает динамический объект по заданному индексу, который связан с данным видимым объектом (нулевой индекс имеет динамическое действие, связанное с видимым объектом первым). Данный метод наиболее удобен при выполнении однотипных или

последовательных операций над динамическими объектами, связанными с текущим видимым объектом.

Пример

```
Sub GetDynamicObject(o As GwxPick)

Dim dynamicsName(10) As String        'Имена динамических действий
Dim numberDynamics As Long                    'Количество действий, примененных к объекту
'Доступ через IDispatch к объектам неизвестного типа
Dim visibleObject As Object
'Доступ через IDispatch к объектам неизвестного типа
Dim dynamicObject As Object
'Получаем видимый объект с именем Rectangle1
Set visibleObject = ThisDisplay.GetVisibleObjectFromName("Rectangle1")
'Получаем количество динамических действий, связанных с видимым объектом
numberDynamics = visibleObject.GetNumberOfDynamics
If numberDynamics > 0 Then
   For i = 0 To numberDynamics - 1
      Set dynamicObject = visibleObject.GetDynamicObjectFromIndex(i)
      dynamicsName(i) = dynamicObject.GetObjectTypeName
      Set dynamicObject = Nothing
   Next i
End If
Set visibleObject = Nothing

End Sub
```

Function GetDynamicObjectFromName(nameSubstring As String) As Object

Возвращает динамический объект с заданным именем (или частью имени), который связан с текущим видимым объектом.

Пример

```
Dim dynamicsName(10) As String        'Имена динамических действий
Dim numberDynamics As Long                    'Количество действий, примененных к объекту
'Доступ через IDispatch к объектам неизвестного типа
Dim visibleObject As Object
'Доступ через IDispatch к объектам неизвестного типа
Dim dynamicObject As Object
'Получаем видимый объект с именем Rectangle1
Set visibleObject = ThisDisplay.GetVisibleObjectFromName("Rectangle1")
'Получаем количество динамических действий, связанных с видимым объектом
numberDynamics = visibleObject.GetNumberOfDynamics
If numberDynamics > 0 Then
     Set dynamicObject = visibleObject.GetDynamicObjectFromName("LevelChanging_")
  dynamicsName(i) = dynamicObject.GetObjectTypeName
  Set dynamicObject = Nothing
End If
Set visibleObject = Nothing
```

Function GetNumberOfDynamics() As Long

Возвращает количество динамических объектов, связанных с текущим видимым объектом.

Пример

```
Dim dynamicsName(10) As String        'Имена динамических действий
Dim numberDynamics As Long                    'Количество действий, примененных к объекту
'Доступ через IDispatch к объектам неизвестного типа
Dim visibleObject As Object
'Доступ через IDispatch к объектам неизвестного типа
Dim dynamicObject As Object
'Получаем видимый объект с именем Rectangle1
Set visibleObject = ThisDisplay.GetVisibleObjectFromName("Rectangle1")
'Получаем количество динамических действий, связанных с видимым объектом
numberDynamics = visibleObject.GetNumberOfDynamics
If numberDynamics > 0 Then
     Set dynamicObject = visibleObject.GetDynamicObjectFromName("LevelChanging_")
  dynamicsName(i) = dynamicObject.GetObjectTypeName
  Set dynamicObject = Nothing
End If
Set visibleObject = Nothing
```

Sub GetObjectDimensions(left As Single, top As Single, width As Single, height As Single)

Возвращает размер и положение объекта. Этот метод не работает в VBScript иJscript.

Sub GetObjectDimensions2(left, top, width, height)

Метод может вызываться как в Visual Basic for Applications, так и в VBScript и Jscript. Для использования его в VBA необходимо объявить параметры как Variant.

```
'VB example, works only in runtime mode
'example for getting object dimensions (GetObjectDimensions2
returns variants)
'visible is GwxVisible object
'example requires an object named "test"

Dim l, t, w, h, visible
Set visible = ThisDisplay.GetVisibleObjectFromName("test")
visible.GetObjectDimensions2 l, t, w, h
MsgBox "Left: " & l & vbCrLf & "Top: " & t & _
      vbCrLf & "Width: " & w & _
      vbCrLf & "Height: " & h
```

Function GetObjectTypeName() As String

Возвращает имя класса объекта.

Пример

```
Dim dynamicsName(10) As String        'Имена динамических действий
Dim numberDynamics As Long                    'Количество действий, примененных к объекту
'Доступ через IDispatch к объектам неизвестного типа
Dim visibleObject As Object
'Доступ через IDispatch к объектам неизвестного типа
Dim dynamicObject As Object
'Получаем видимый объект с именем Rectangle1
Set visibleObject = ThisDisplay.GetVisibleObjectFromName("Rectangle1")
'Получаем количество динамических действий, связанных с видимым объектом
numberDynamics = visibleObject.GetNumberOfDynamics
If numberDynamics > 0 Then
    Set dynamicObject = visibleObject.GetDynamicObjectFromName("LevelChanging_")
  dynamicsName(i) = dynamicObject.GetObjectTypeName
  Set dynamicObject = Nothing
End If
Set visibleObject = Nothing
```

Function GetParentObject() As Object

Метод получает указатель на «объект-родитель» в иерархии объектов GraphWorX32. При рассмотрении следующего примера представьте, что есть два объекта "child1" и "child2", сгруппированные в символ "parent". Начиная с указателя на "child1", Вы получите указатель на parent, и затем имя экранной формы.

```
Dim obj As GwxRectangle
Set obj = ThisDisplay.GetVisibleObjectFromName("child1")
If obj Is Nothing Then
   MsgBox "can't find child1"
Else
   Set parent = obj.GetParentObject
   If parent Is Nothing Then
      MsgBox "can't get parent"
   Else
```

```
        MsgBox parent.objectName
      End If
    End If
```

Sub MoveObject(offsetX As Single, offsetY As Single)

Выполняет смещение объекта относительно его первоначального положения. Данный метод работает только для объектов, которые являются видимыми объектами верхнего уровня.

Пример

```
Dim visibleObject As Object
Dim offsetX As Single
Dim offsetY As Single

Set visibleObject = ThisDisplay.GetVisibleObjectFromName("Rectangle1")
if Not ( visibleObject Is Nothing ) Then
    Call visibleObject.MoveObject (offsetX, offsetY)
End If
Set visibleObject = Nothing
```

Sub RefreshObject()

Вызывает перерисовку видимого объекта.

Пример

```
Dim visibleObject As Object
Set visibleObject = ThisDisplay.GetVisibleObjectFromName("Rectangle1")
if Not ( visibleObject Is Nothing ) Then
    Call visibleObject.RefreshObject
End If
Set visibleObject = Nothing
```

Sub Rotate90(clockwise As Boolean)

Поворачивает объект на 90 градусов. Требуется один параметр типа Boolean, определяющий, в какую сторону будет выполняться поворот: по часовой стрелке или против.

Пример.

```
Dim rect As GwxRectangle
Set rect = ThisDisplay.GetVisibleObjectFromName("rectangle")
If rect Is Nothing Then
    MsgBox "An error has occurred while trying to get the object"
Else
    rect.Rotate90 True
End If
```

Function SetAliasDefinition(aliasName As String, newDefinition As String) As Boolean

Выполняет настройку псевдонимов для всех динамических объектов, связанных с текущим видимым объектом. Если данный видимый объект является объектом класса **GwxSymbol** (символом), то данный метод выполняется рекурсивно для всех объектов, из которых сгруппирован символ. Возвращает FALSE, если не обнаружено псевдонимов с заданными именами, и TRUE, – в противном случае. (См. также **GwxDisplay::SetAliasDefinition** и **GwxDynamic::SetAliasDefinition**).

Пример

```
Sub AliasSet(o As GwxPick)
Dim result As Boolean
Dim visibleObject As Object

Set visibleObject = ThisDisplay.GetVisibleObjectFromName("Rectangle1")

If Not ( visibleObject Is Nothing ) Then
    result = visibleObject.SetAliasDefinition _
    ("Bar", "Fastwel.UniOpc.1\Device1.unit2.analog_154")
```

```
End If
Set visibleObject = Nothing
End Sub
```

Function SetAliases(commandString As String) As Boolean

Выполняет настройку нескольких псевдонимов для текущего видимого объекта. Если данный видимый объект является объектом класса **GwxSymbol** (*символом*), то данный метод выполняется рекурсивно для всех объектов, из которых сгруппирован символ. Возвращает FALSE, если не обнаружено псевдонимов с заданными именами, и TRUE, – в противном случае. (См. также **GwxDisplay::SetAliases()** и **GwxVisible::SetAliasDefinition()**).

Пример

```
Sub AliasSet(o As GwxPick)
Dim result As Boolean
Dim visibleObject As Object
Dim aliasDefinition As String

aliasDefinition = _
"TankLevel"+Chr(9) + "Fastwel.UniOpc.1\Device1.unit1.analog_100" + Chr(13) + Chr(10)

Set visibleObject = ThisDisplay.GetVisibleObjectFromName("Rectangle1")

If Not ( visibleObject Is Nothing ) Then
    result = visibleObject.SetAliases(aliasDefinition)
End If
Set visibleObject = Nothing
```

Function SetAliasesFromFile(filename As String) As Boolean

Выполняет настройку нескольких псевдонимов для текущего видимого объекта по конфигурационной информации из файла настройки псевдонимов. Если данный видимый объект является объектом класса **GwxSymbol** (*символом*), то данный метод выполняется рекурсивно для всех объектов, из которых сгруппирован символ. Возвращает FALSE, если не обнаружено псевдонимов с заданными именами, и TRUE, – в противном случае. (См. также **GwxDisplay::SetAliasesFromFile**() и **GwxVisible::SetAliasDefinition**()).

Пример

```
Sub AliasSet(o As GwxPick)
Dim result As Boolean
Dim aliasFileName As String
Dim visibleObject As Object

aliasFileName = "C:\Station\KNS1.txt"
Set visibleObject = ThisDisplay.GetVisibleObjectFromName("Rectangle1")

If Not ( visibleObject Is Nothing ) Then
    result = visibleObject.SetAliasesFromFile(aliasFileName)
End If

Set visibleObject = Nothing
End Sub
```

Function SetGradientFill(style As GWXGRADIENTSTYLE, color1 As OLE_COLOR, color2 As OLE_COLOR, isTwoColor As Boolean, brightness As Single, reverseColors As Boolean, steps As Integer, horizontalOffset As Single, verticalOffset As Single) As Boolean

Устанавливает градиентную заливку для графического объекта.

Параметры, передаваемые функции:

GWXGRADIENTSTYLE style – перечислитель, определяющий **стиль** градиентной заливки. Возможные значения:

- GradientHorizontal (горизонтально) = 0
- GradientVertical (вертикально) = 1
- GradientSquare (квадрат) = 2

OLE_COLOR color1 – основной цвет заливки графического объекта

OLE_COLOR color2 – цвет второго воображаемого источника света, освещающего объект
boolean isTwoColor – признак использования двух цветов для формирования заливки. Если установлен в True, параметр brightness (яркость) игнорируется, параметр color2 – воспринимается. Если установлен в False – наоборот.
float brightness – определяет яркость воображаемого основного источника света в диапазоне от 0.0 до 1.0
boolean reverseColors – признак взаимной перестановки воображаемых источников света. При использовании одного источника света значение True приводит к зеркальному перемещению источника.
int steps – целое число, определяющее плавность перехода фрагментов градиентной заливки. Может принимать следующие значения: 5, 10, 15, 20, 25, 30, 40, 50, 75, 100, 150, 200, 250
float horizontalOffset – определяет степень сдвига воображаемого источника по горизонтали в сторону своего зеркального отображения в диапазоне от –1.0 до 1.0.
float verticalOffset – определяет степень сдвига воображаемого источника по вертикали в сторону своего зеркального отображения в диапазоне от 0.0 до 1.0.

Пример

```
Sub Gradient(o As GwxPick)
Dim result As Boolean
Dim visibleObject As Object
'Доступ через IDispatch к объектам неизвестного типа
Set visibleObject = ThisDisplay.GetVisibleObjectFromName("Rectangle1")
result = visibleObject.SetGradientFill _
(GradientHorizontal, RGB(0, 128, 255), RGB(255, 255, 255), False, 0.65, _
False, 5, -1, -1)
End Sub
```

Sub SetObjectDimensions(left As Single, top As Single, width As Single, height As Single)

Устанавливает размер и положение объекта. Данный метод работает только для объектов, которые являются видимыми объектами верхнего уровня.

Пример:

```
Dim numberOfTopLevelVisibles As Long
Dim visibleObject As Object
Dim left As Single
Dim top As Single
Dim width As Single
Dim height As Single

left = 100
top = 100
width = 100
height = 100

numberOfTopLevelVisibles = ThisDisplay.GetNumberOfTopLevelVisibles
If numberOfTopLevelVisibles > 0 Then
   For i = 0 To numberOfTopLevelVisibles - 1
      Set visibleObject = ThisDisplay.GetVisibleObjectFromIndex(i)
      Call visibleObject.SetObjectDimensions _
      (left + i * left, top + i * top, width * i, height * i)
         Set visibleObject = Nothing
   Next i
End If
```

Sub StretchObject(scaleX As Single, scaleY As Single, anchorX As Single, anchorY As Single)

Растягивает/сжимает объект по заданным значениям масштаба сжатия/растяжения. Аргументы **scaleX** и **scaleY** могут принимать значения от 0,0 до 1,0, что соответствует диапазону от 0 до 100%. Аргументы **anchorX** и **anchorY** используются для передачи координат точки привязки объекта, что определяет сдвиг различных подобластей объекта

друг относительно друга. Например, если растягивание объекта выполняется относительно точки привязки, расположенной в его центре, аргументы **anchorX** и **anchorY** должны содержать значения координат центра объекта. Данный метод работает только для объектов, которые являются видимыми объектами верхнего уровня.

Пример:

```
Dim numberOfTopLevelVisibles As Long
Dim visibleObject As Object
Dim scaleX As Single
Dim scaleY As Single
Dim anchorX As Single
Dim anchorY As Single

scaleX = 0.9
scaleY = 0.75
anchorX = 100
anchorY = 100

numberOfTopLevelVisibles = ThisDisplay.GetNumberOfTopLevelVisibles
If numberOfTopLevelVisibles > 0 Then
   For i = 0 To numberOfTopLevelVisibles - 1
      Set visibleObject = ThisDisplay.GetVisibleObjectFromIndex(i)
      Call visibleObject.StretchObject _
               (i * scaleX, i * scaleY, i * anchorX, i * anchorY)
   Set visibleObject = Nothing
   Next i
End If
```

GwxText

Объекты класса **GwxText** наследуют все свойства и методы класса **GwxVisible**. Описание набора дополнительных свойств приведено ниже.

Свойства

Font As IFontDisp

Возвращает/устанавливает тип шрифта, который используется для отображения фрагментов текста.

Пример

```
Sub InvokeText(o As GwxText)
Dim textObject As GwxText

Set textObject = ThisDisplay.GetVisibleObjectFromName("StaticText1")
If Not (textObject Is Nothing) Then
    MsgBox (textObject.Font.Name)
End If

Set textObject = Nothing
End Sub
```

alignment As GWXTEXTALIGNMENT

Задает выравнивание текста (по левому краю, по центру, по правому краю). Может быть применено только к многострочным фрагментам текста. Допустимые значения **GWXTEXTALIGNMENT**:

- TextAlignLeft (по левому краю) = 0
- TextAlignCenter (по центру) = 1
- TextAlignRight (по правому краю) = 2

Пример

```
Sub InvokeText(o As GwxText)
Dim textObject As GwxText
Set textObject = ThisDisplay.GetVisibleObjectFromName("StaticText1")
If Not (textObject Is Nothing) Then
    textObject.Alignment = TextAlignCenter
End If
Set textObject = Nothing
End Sub
```

BoundaryAlignment As GWXTEXTALIGNMENT

Задает выравнивание текста (по левому краю, по центру, по правому краю). Может быть применено к однострочным и многострочным фрагментам текста. Допустимые значения **GWXTEXTALIGNMENT**:

- TextAlignLeft (по левому краю) = 0
- TextAlignCenter (по центру) = 1
- TextAlignRight (по правому краю) = 2

Пример

```
Sub InvokeText(o As GwxText)
Dim textObject As GwxText
Set textObject = ThisDisplay.GetVisibleObjectFromName("StaticText1")
If Not (textObject Is Nothing) Then
    textObject.BoundaryAlignment = TextAlignCenter
End If
Set textObject = Nothing
End Sub
```

stretchText As Boolean

TRUE, если размер шрифта автоматически изменяется при растяжении/сжатии фрагмента текста.
FALSE, если размер шрифта остается неизменным.

text As String

Фрагмент текста, отображаемый при помощи объекта **GwxText**.

Пример

```
Sub InvokeText(o As GwxText)
Dim textObject As GwxText
Set textObject = ThisDisplay.GetVisibleObjectFromName("StaticText1")
If Not (textObject Is Nothing) Then
        textObject.Font.Bold = True
       textObject.Font.Name = "Courier New"
       textObject.Font.Size = 20
   textObject.Alignment = TextAlignCenter
   textObject.BoundaryAlignment = TextAlignRight
   textObject.Text = "Test String Entered"
   Call textObject.RefreshObject
   Call ThisDisplay.RefreshWindow
   Set textObject = Nothing
End If
End Sub
```

GwxRectangle

Объекты класса **GwxRectangle** наследуют все свойства и методы класса **GwxVisible**. Описание набора дополнительных свойств приведено ниже.

Свойства

rounded As Boolean

Если значение данного свойства равно TRUE, прямоугольник имеет скругленные углы с параметрами скругления по умолчанию.

RoundingX As Long

Возвращает/устанавливает величину горизонтального скругления углов прямоугольника.

GwxArc

Объекты класса **GwxArc** наследуют все свойства и методы класса **GwxVisible**. Описание набора дополнительных свойств приведено ниже.

Свойства

endAngle As Single

Возвращает/устанавливает конечный угол дуги в градусах.

arcType As GWXARCTYPE

Возвращает/устанавливает тип дуги. Допустимые значения **GWXARCTYPE**:

- ArcArc (дуга) = 0
- ArcPie (сектор) = 1
- ArcChord (хорда) = 2

startAngle As Single

Возвращает/устанавливает начальный угол дуги в градусах.

GwxSymbol

Объект класса **GwxSymbol** представляет набор сгруппированных объектов, классы которых унаследованы от класса **GwxVisible**. Объекты **GwxSymbol** могут использоваться для группировки других объектов класса **GwxSymbol**, обеспечивая вложенную группировку. Объекты класса **GwxSymbol** наследуют все свойства и методы класса **GwxVisible**. Описание дополнительных методов приведено ниже.

Методы

Function GetDataSourcesRecursive(includeDuplicates As Boolean)

Получает все имена тегов, связанные с видимым объектом. Возвращает массив строк (тип variant), каждая строка в котором соответствует имени тега. Опционально можно включать или не включать повторяющиеся теги в массив.

Function GetNumberOfChildVisibles() As Long

Возвращает количество ближайших по уровню вложенности дочерних объектов, из которых сгруппирован текущий символ.

Function GetVisibleObjectFromIndex(index As Long) As Object

Возвращает видимый объект по заданному индексу, который является ближайшим по уровню вложенности дочерним объектом данного символа (первый дочерний объект имеет нулевой индекс). Данная функция удобна при выполнении повторяющихся действий над дочерними объектами, из которых сгруппирован текущий объект.

Function GetVisibleObjectFromName(nameSubstring As String) As Object

Возвращает видимый объект с заданным именем, который является ближайшим по уровню вложенности дочерним объектом данного символа

GetVisibleObjectFromPosition(x as long, y as long) as Object

Возвращает объект, расположенный в определенном месте на экране. Начинает с основного экрана и проходит далее, включая слои, пока не будет найден самый дальний объект. Если аргумент **GoIntoSymbol** имеет значение True, поиск продолжается до последнего видимого объекта данной позиции. Иначе функция возвращает верхний символ, содержащий объект.

```
'VBA example, works in configure and runtime mode
'this example gets the lowest level visible at random position
Dim w As Long, h As Long
Dim o As Object
ThisDisplay.GetDisplayDimensions w, h
'gets display width and height
If ThisDisplay.GetNumberOfTopLevelVisibles > 0 Then
'checks if display has at least one visible object
'and runs a loop for find one
 Do
  Set o = ThisDisplay.GetVisibleObjectFromPosition(Rnd * w, Rnd *
h, True)
'tries to get visible object from random position
'- if object is GwxSymbol, it goes into symbol
'and continues searching
 Loop Until Not (o Is Nothing)
'after any object is found, loop ends and example
'shows name of the object
 MsgBox o.objectName
Else
 MsgBox "Example needs at least one visible object"
End If
```

Пример VBA:

```
Dim obj As Object
'We assume you have created a rectangle on the
'specific position.
Set obj = ThisDisplay.GetVisibleObjectFromPosition(588, _
                                                   256, _
                                                   True)
Dim rect As GwxRectangle
Set rect = obj
MsgBox "Change the color of the rectangle." + _
    " Its name is: " + CStr(rect.objectName)

rect.fillColor = rgb(200, 200, 255)

MsgBox "Undo fillcolor."
rect.fillColor = rgb(255, 191, 0)
```

Пример VB:

```
Dim obj

Set obj = ThisDisplay.GetVisibleObjectFromPosition(588, _
                                                   256, _
                                                   True)
Dim rect
Set rect = obj
MsgBox "Change the color of the rectangle." + _
    " Its name is: " + CStr(rect.objectName)

rect.fillColor = RGB(200, 30, 255)
     MsgBox "Undo fill color."
     rect.fillColor = RGB(200, 200, 255)
```

Function PushCurrentSymbol() As Boolean

Помещает объект класса **GwxSymbol** в стек редактирования символа. По сути дела, это приводит к тому, что становятся доступными методы и свойства объектов, из которых сгруппирован текущий символ (некоторые операции, подобные **GwxVisible::SetObjectDimensions** могут быть выполнены только для объектов верхнего уровня вложенности).

Пример VBA:




```
Dim obj As Object
        MsgBox "PushCurrentSymbol, ""Symbol1""" + _
        Chr(13) + "Press the button ""PushCurrentSymbol2""" + _
        " to reach the Symbol2 in the Symbol1."
        Set obj = ThisDisplay.GetVisibleObjectFromName("Object1")
    Dim oRec As GwxSymbol
    Set oRec = obj
```

```
oRec.PushCurrentSymbol
    Set obj = ThisDisplay.GetVisibleObjectFromName("EditLevel")
obj.text = CStr(Val(obj.text) + 1)
    MsgBox "See the current edit level." + _

        Chr(13) + _
        "We can double-click aside from this symbol or " + _
        "we can call the function PopCurrentSymbol()."
```

GwxOleObject

Объекты класса **GwxOleObject** наследуют все свойства и методы класса **GwxVisible**. Описание дополнительных методов приведено ниже.

Методы

Function GetOLEObject() As Object

Возвращает интерфейс реального объекта OLE, диспетчеризуемый через объект класса **GwxOleObject**.

Пример VBA

```
Dim o As GwxOleObject
Set o = ThisDisplay.GetVisibleObjectFromName("twx_01")

Dim twx As TWXVIEWERLib.TWXViewer
Set twx = o.GetOLEObject()

twx.BackColor = RGB(0, 255, 0)
```

GwxEllipse

Объекты класса **GwxEllipse** наследуют все свойства и методы класса **GwxVisible**.

GwxLine

Объекты класса **GwxLine** наследуют все свойства и методы класса **GwxVisible**.

GwxBitmap

Объекты класса **GwxBitmap** наследуют все свойства и методы класса **GwxVisible**.

GwxMetafile

Объекты класса **GwxMetafile** наследуют все свойства и методы класса **GwxVisible**.

GwxButton

Объект класса **GwxButton** не имеет какой-либо функциональности до тех пор, пока с ним с не связан динамический объект класса **GwxPick**. **GwxButton** имеет все свойства и методы классов **GwxVisible** и **GwxText**.

GwxDynamic

GwxDynamic является классом, от которого унаследованы классы всех динамических объектов (действий) GraphWorX32 (динамическое изменение размера, положения и т.д.). Иными словами, все классы динамических объектов GraphWorX32 имеют свойства и методы класса **GwxDynamic**.

Свойства

dataSource As String

Первичный источник данных динамического объекта. **DataSource** является строкой, представляющей имя тега OPC, выражение, константу или локальную переменную GraphWorX32.

Пример VBA:

```
    Dim obj As Object
    Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")

    Dim oDynamic As GwxDynamic
    'GwxProcessPoint has all the properties
    'and methods of GwxDynamic.

    Set oDynamic = obj
    'We can complete code via intellisence...

    Dim oPP As GwxProcessPoint
    Set oPP = oDynamic

    Dim strName As String
    strName = oPP.GetObjectTypeName

    oPP.UserDescription = "ExampleDataSource"

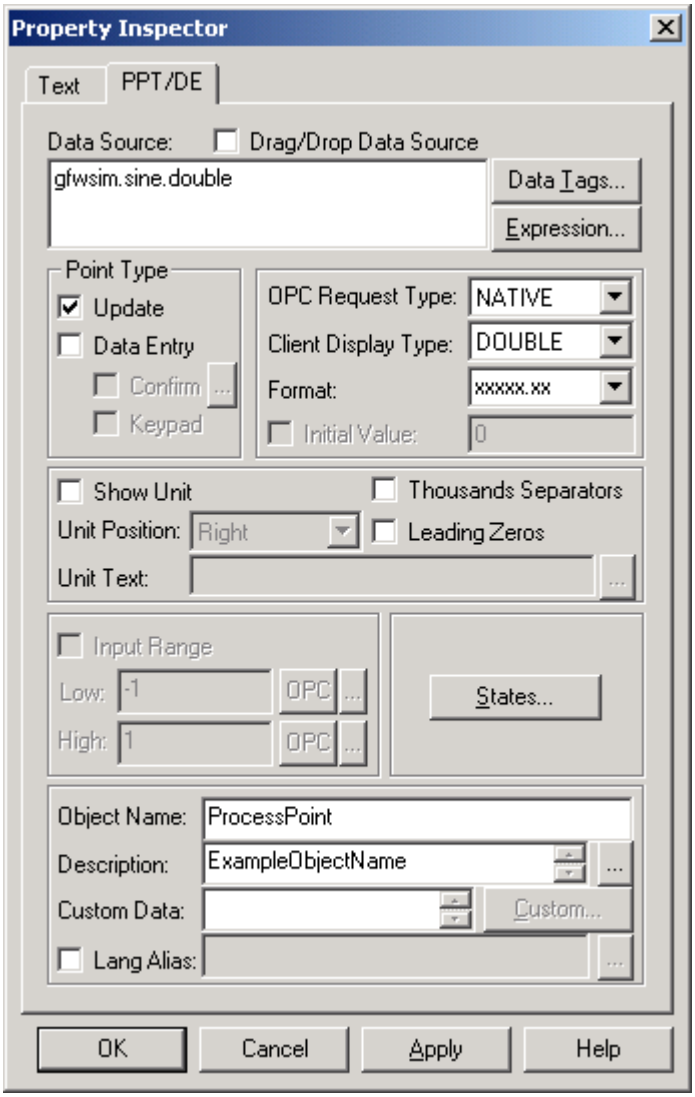

    MsgBox "Changing data source in the ProcessPoint to
gfwsim.cosine.double."
    oPP.dataSource = "gfwsim.cosine.double"
```

Пример VB:

```
    Dim obj
    Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")
        Dim strName
    strName = obj.GetObjectTypeName

    obj.UserDescription = "ExampleDataSource"

    MsgBox "Changing data source in the ProcessPoint to
gfwsim.cosine.double."
    obj.dataSource = "gfwsim.cosine.double"
```



highRange As String

Представляет переопределенное значение верхней границы диапазона для текущего динамического действия. **HighRange** является строкой, представляющей имя тега OPC, выражение, константу или локальную переменную GraphWorX32. Данное свойство используется только для динамики, базирующейся на аналоговой информации, и игнорируется для динамики, основанной на соединениях с цифровыми данными.

Пример VBA:

```
Dim obj As Object
 Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")

 Dim oDynamic As GwxDynamic
 'GwxProcessPoint has all the properties
 'and methods of GwxDynamic.

 Set oDynamic = obj
 'We can complete code via intellisence...

 Dim oPP As GwxProcessPoint
 Set oPP = oDynamic

 Dim strName As String
 strName = oPP.GetObjectTypeName

 oPP.UserDescription = "ExampleHighRange"
 Dim h As String
```

```
Dim l As String
h = oPP.highRange
l = oPP.lowRange

'This properties are ignored for
'dynamics which are based on an analog data
'source.
MsgBox "Set ranges to <-0.5;0.5> units."


oPP.RangeOverride = True

oPP.highRange = "0.5"
oPP.lowRange = "-0.5"
```

Пример VB:

```
MsgBox "This method is valid only for Data Entry."
Dim obj
 Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")
     Dim oDynamic
 'GwxProcessPoint has all the properties
 'and methods of GwxDynamic.
     Set oDynamic = obj
 'We can complete code via intellisence...
     Dim oPP
 Set oPP = oDynamic
     Dim strName
 strName = oPP.GetObjectTypeName
   oPP.UserDescription = "ExampleHighRange"
 Dim h
 Dim l
 h = oPP.highRange
 l = oPP.lowRange

 'This properties are ignored for
 'dynamics which are based on an analog data
 'source.
 MsgBox "Set ranges to <-0.5;0.5> units."

 oPP.RangeOverride = True

 oPP.highRange = "0.5"
 oPP.lowRange = "-0.5"
```

lowRange As String

Представляет переопределенное значение нижней границы диапазона для текущего динамического действия. **LowRange** является строкой, представляющей имя тега OPC, выражение, константу или локальную переменную GraphWorX32. Данное свойство используется только для динамики, базирующейся на аналоговой информации, и игнорируется для динамики, основанной на соединениях с цифровыми данными.

Пример VBA:

```
Dim obj As Object
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")

Dim oDynamic As GwxDynamic
'GwxProcessPoint has all the properties
'and methods of GwxDynamic.

Set oDynamic = obj
'We can complete code via intellisence...

Dim oPP As GwxProcessPoint
Set oPP = oDynamic

Dim strName As String
strName = oPP.GetObjectTypeName
```

```
oPP.UserDescription = "ExampleLowRange"
Dim h As String

Dim l As String
h = oPP.highRange
l = oPP.lowRange

'This properties are ignored for
'dynamics which are based on an analog data
'source.

MsgBox "Set ranges to <-0.5;0.5> units."

oPP.highRange = "0.5"
oPP.lowRange = "-0.5"
```

Пример VB:

```
MsgBox "This method is valid only for Data Entry."
 Dim obj
 Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")

 Dim oDynamic
 'GwxProcessPoint has all the properties
 'and methods of GwxDynamic.

 Set oDynamic = obj
 'We can complete code via intellisence...

 Dim oPP
 Set oPP = oDynamic

 Dim strName
 strName = oPP.GetObjectTypeName

 oPP.UserDescription = "ExampleLowRange"
 Dim h

 Dim l
 h = oPP.highRange
 l = oPP.lowRange

 'This properties are ignored for
 'dynamics which are based on an analog data
 'source.

 MsgBox "Set ranges to <-0.5;0.5> units."

 oPP.highRange = "0.5"
 oPP.lowRange = "-0.5"
```

objectName As String

Возвращает/устанавливает имя для экземпляра динамического объекта. Имя используется для идентификации объекта при обращении к его полям (свойствам и методам), например, – **GwxDisplay::GetDynamicObjectFromName**. Обязательным условием идентификации объектов является уникальность имени каждого объекта. Если пользователь присваивает видимому объекту уже существующее имя, GraphWorX32 добавит к имени порядковый номер (например, при создании нескольких копий объекта с именем "*Tank*", копиям будут присвоены имена "*Size1*", "*Size2*" и т.д.).

Пример VBA:

```
Dim obj As Object
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")

Dim oDynamic As GwxDynamic
'GwxProcessPoint has all the properties
'and methods of GwxDynamic.
```

```
Set oDynamic = obj
'We can complete code via intellisence...

Dim oPP As GwxProcessPoint
Set oPP = oDynamic

Dim strName As String
strName = oPP.GetObjectTypeName

oPP.UserDescription = "ExampleObjektName"
oPP.objectName = "ProcessPoint"
```

Пример VB:

```
Dim obj
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")


Dim strName
strName = obj.GetObjectTypeName

obj.UserDescription = "ExampleObjektName"
MsgBox "The name of the process point above is: " + _
            obj.objectName
```

RangeOverride As Boolean

Возвращает/устанавливает признак переопределения диапазона. Если значение свойства равно FALSE, GraphWorX32 будет использовать диапазон, связанный с первичным источником данных. Если значение свойства равно TRUE, будут использоваться границы диапазона, заданные свойствами **HighRange** и **LowRange**. Данное свойство используется только для динамики, базирующейся на аналоговой информации (**GwxSize**, **GwxLocation**, **GwxRotation**, **GwxAnalogColor**, **GwxAnalogSelect** и **GwxProcessPoint**), и игнорируется для динамики, основанной на соединениях с цифровыми данными.

Пример VBA:

```
Dim obj As Object
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")
    Dim oDynamic As GwxDynamic
'GwxProcessPoint has all the properties
'and methods of GwxDynamic.

Set oDynamic = obj
'We can complete code via intellisence...

Dim oPP As GwxProcessPoint
Set oPP = oDynamic

Dim strName As String
strName = oPP.GetObjectTypeName

oPP.UserDescription = "ExampleRangeOverride"


oPP.RangeOverride = True
oPP.highRange = "0.5"
oPP.lowRange = "-0.5"
```

Пример VB:

```
Dim obj
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")

obj.RangeOverride = True
oPP.highRange = "0.5"
oPP.lowRange = "-0.5"
```

RequestDataType As GWXDATATYPE

Позволяет определить тип данных, запрашиваемых с OPC-сервера.

Пример VBA:

```
Dim obj As Object
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")
    Dim oDynamic As GwxDynamic
'GwxProcessPoint has all the properties
'and methods of GwxDynamic.
    Set oDynamic = obj
'We can complete code via intellisence...
    Dim oPP As GwxProcessPoint
Set oPP = oDynamic
    Dim strName As String
strName = oPP.GetObjectTypeName
     oPP.UserDescription = "ExampleRequestDataType"
    oPP.RequestDataType = DataTypeDouble
```

Пример VB:

```
Dim obj
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")

obj.UserDescription = "ExampleRequestDataType"

obj.RequestDataType = DataTypeDouble
```

TimerRate As Long

Частота обновления таймера, по которому происходит динамическое изменение визуальных свойств графических объектов классов **GwxFlash**, **GwxAnimator** и **GwxPick**.

Пример VBA:

```
Dim obj As Object
 Set obj = ThisDisplay.GetDynamicObjectFromName("AnimatedCar")

 Dim oDynamic As GwxDynamic
 'GwxProcessPoint has all the properties
 'and methods of GwxDynamic.

 Set oDynamic = obj
 'We can complete code via intellisence...

 Dim oAni As GwxAnimator
 Set oAni = oDynamic

 Dim strName As String
 strName = oAni.GetObjectTypeName

 oAni.TimerRate = 1000 'ms
 MsgBox "See the animated car with the frame rate " + _

     CStr(oAni.TimerRate) + " ms."

 oAni.TimerRate = 50

 MsgBox "Now the frame rate has changed to 50 ms."
```

Пример VB:

```
Dim obj
 Set obj = ThisDisplay.GetDynamicObjectFromName("AnimatedCar")


 Dim strName
 strName = obj.GetObjectTypeName

 obj.TimerRate = 1000 'ms
```

```
    MsgBox "See the animated car with the frame rate " + _
        CStr(obj.TimerRate) + " ms."

    obj.TimerRate = 50

    MsgBox "Now the frame rate has changed to 50 ms."
```

UserCustomData As String

Данная строка используется для хранения специальных пользовательских данных. Свойство применяется для связывания дополнительной информации с динамическим объектом.

Пример VB:

```
   Dim obj
    Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")


    obj.UserDescription = "ExampleCustomdata"
    obj.UserCustomData = "Device #XY"
```

Пример VBA:

```
    Dim obj As Object
    Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")

    Dim oDynamic As GwxDynamic
    'GwxProcessPoint has all the properties
    'and methods of GwxDynamic.

    Set oDynamic = obj
    'We can complete code via intellisence...

    Dim oPP As GwxProcessPoint
    Set oPP = oDynamic

    Dim strName As String
    strName = oPP.GetObjectTypeName

    oPP.UserDescription = "ExampleCustomdata"
    oPP.UserCustomData = "Device #XY"
```

UserDescription As String

Строка описания динамического объекта. Обычно данная строка используется в качестве вспомогательной текстовой информации, выводимой в окне подсказки при нахождении курсора мыши над объектом в течение некоторого времени.

Пример

```
Sub InvokeDynamicAccess(o As GwxPick)
Dim dynamicObj As Object
Dim pointObj As Object
Dim objProperties As String
Set dynamicObj = ThisDisplay.GetDynamicObjectFromName("PPT1")
If Not (dynamicObj Is Nothing) Then

    Set pointObj = dynamicObj.GetDataSourcePointObject

    objProperties = "Объект " + dynamicObj.objectName + ": " + Chr(13) _
  + dynamicObj.dataSource + ": " + Chr(13) _
  + Str(pointObj.Value)

  MsgBox (objProperties)
  Set pointObj = Nothing
  Set dynamicObj = Nothing
End If

End Sub
```

Методы

Function GetAliasDefinition(aliasName As String) As String

Возвращает строку настройки привязки для псевдонима с данным именем.

VBA пример:

```
Dim obj As Object
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPointWithAlias")

Dim oDynamic As GwxDynamic
'GwxProcessPoint has all the properties
'and methods of GwxDynamic.

Set oDynamic = obj
'We can complete code via intellisence...

Dim oPP As GwxProcessPoint
Set oPP = oDynamic
Dim p
p = oPP.objectName
Dim strName As String
strName = oPP.GetObjectTypeName

Dim strNewAliasDefinition As String

Dim strAliasName As String
strAliasName = "AliasString"


'Call oPP.SetAliasDefinition(strAliasName, "gfwsim.cosine.double")

strNewAliasDefinition = oPP.GetAliasDefinition(strAliasName)
MsgBox "Alias definition of the ""ProcessPoint"" is: " + _
    strNewAliasDefinition
```

VB пример:

```
Dim obj
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPointWithAlias")

Dim strName
strName = obj.GetObjectTypeName

Dim strNewAliasDefinition
Dim strAliasName
strAliasName = "AliasString"


'Call oPP.SetAliasDefinition(strAliasName, "gfwsim.cosine.double")

strNewAliasDefinition = obj.GetAliasDefinition(strAliasName)
MsgBox "Alias definition of the ""ProcessPoint"" is: " + _
    strNewAliasDefinition
```

Function GetDataSourcePointObject() As Object

Возвращает объект **GwxPoint** для источника данных текущего динамического объекта.

Пример VBA:

```
Dim obj As Object
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")

Dim oDynamic As GwxDynamic
'GwxProcessPoint has all the properties
'and methods of GwxDynamic.

Set oDynamic = obj
'We can complete code via intellisence...

Dim oPP As GwxProcessPoint
Set oPP = oDynamic
```

```
    Dim strName As String
    strName = oPP.GetObjectTypeName

    Dim oPoint As GwxPoint
    Set oPoint = oPP.GetDataSourcePointObject


    Dim str As String

    MsgBox "The Process Point includes an object - GwxPoint." + _
        Chr(13) + "The data type number of the point is: " + _
        CStr(oPoint.dataType)
```

Пример VB:

```
    Dim obj
    Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")

    Dim oDynamic
    'GwxProcessPoint has all the properties
    'and methods of GwxDynamic.

    Set oDynamic = obj
    'We can complete code via intellisence...

    Dim oPP
    Set oPP = oDynamic

    Dim strName
    strName = oPP.GetObjectTypeName

    Dim oPoint
    Set oPoint = oPP.GetDataSourcePointObject

    Dim str

    MsgBox "The Process Point includes an object - GwxPoint." + _

        Chr(13) + "The data type number of the point is: " + _
        CStr(oPoint.dataType)
```

Function GetHighRangePointObject() As Object

Возвращает объект **GwxPoint** для верхней границы диапазона текущего динамического объекта.

Пример VBA:

```
MsgBox "It has no affect untill you use property" + _
         "highRange"
    'Suppose we have created a process point with
    'the data entry checked off. (e.g. <-100; 100>)
    'We will change values of ranges to <-0.5; 0.5>.


    Dim obj As Object
    Set obj = _

 ThisDisplay.GetDynamicObjectFromName("DataEntryProcessPoint")


    Dim oDynamic As GwxDynamic
    'GwxProcessPoint has all the properties
    'and methods of GwxDynamic.

    Set oDynamic = obj
    'We can complete code via intellisence...

    Dim oPP As GwxProcessPoint
    Set oPP = oDynamic
```

```
    Dim oHRPoint As GwxPoint

    Set oHRPoint = oPP.GetHighRangePointObject
    Dim strValue As String

    strValue = oHRPoint.GetPointName
    MsgBox "The High Range Point Object of" + _

" the ""Process Point"" is: " + strValue
```

Пример VB:

```
    'Suppose we have created a process point with
    'the data entry checked off. (e.g. <-100; 100>)
    'We will change values of ranges to <-0.5; 0.5>.

         MsgBox "It has no affect untill you use property" + _
                "highRange"

    Dim obj
    Set obj = _

ThisDisplay.GetDynamicObjectFromName("DataEntryProcessPoint")

    Dim oHRPoint
    Set oHRPoint = obj.GetHighRangePointObject

    Dim strValue
    strValue = oHRPoint.GetPointName

    MsgBox "The High Range Point Object of the ""Process Point"" is: " + _
         strValue
```

Function GetLowRangePointObject() As Object

Возвращает объект **GwxPoint** для нижней границы диапазона текущего динамического объекта.

Пример VBA example:

```
    'Suppose we have created a process point with
    'the data entry checked off. (e.g. <-100; 100>)


    Dim obj As Object

    Set obj = _

 ThisDisplay.GetDynamicObjectFromName("DataEntryProcessPoint")


    Dim oDynamic As GwxDynamic
    'GwxProcessPoint has all the properties
    'and methods of GwxDynamic.

    Set oDynamic = obj
    'We can complete code via intellisence...

    Dim oPP As GwxProcessPoint
    Set oPP = oDynamic

    Dim oHLPoint As GwxPoint

    Set oHLPoint = oPP.GetLowRangePointObject

    Dim strValue As String
    strValue = oHLPoint.GetPointName

    MsgBox "The Low Range Point Object of the ""Process Point"" is: " + _

        strValue
```

Пример VB:

```
MsgBox "It has no affect untill you use property" + _
               "lowRange"
    'Suppose we have created a process point with
    'the data entry checked off. (e.g. <-100; 100>)

    Dim obj
    Set obj = _

ThisDisplay.GetDynamicObjectFromName("DataEntryProcessPoint")

   Dim  oHLPoint
    Set oHLPoint = obj.GetLowRangePointObject

    Dim strValue
    strValue = oHLPoint.GetPointName

    MsgBox "The Low Range Point Object of the ""Process Point"" is: " + _

        strValue
```

Function GetNumberOfSubDynamics() As Long

Возвращает количество дочерних динамических объектов по отношению к текущему объекту. Дочерние динамические объекты используются следующими объектами: **GwxDigitalColor**, **GwxDigitalSelector**, and **GwxPick**.

Пример VBA:

```
    'We sssume we have added a color dynamic
    'to a rectangle with several tags.
    'gfwsim.sine.double, etc.

    Dim obj As Object
    Set obj = ThisDisplay.GetDynamicObjectFromName("SubDynamic")

    Dim oDynamic As GwxDynamic
    'GwxProcessPoint has all the properties
    'and methods of GwxDynamic.

    Set oDynamic = obj
    'We can complete code via intellisence...

    Dim oPick As GwxDigitalColor
    Set oPick = oDynamic


    Dim lNumberOfSubDynamics As Long
    lNumberOfSubDynamics = oPick.GetNumberOfSubDynamics

    MsgBox "The number of immediate child sub-dynamics in" + _
           " GwxDigitalColor rectangle is: " + _
        CStr(lNumberOfSubDynamics)
```

Пример VB:

```
    'We sssume we have added a color dynamic
    'to a rectangle with several tags.
    'gfwsim.sine.double, etc.

    Dim obj
    Set obj = ThisDisplay.GetDynamicObjectFromName("SubDynamic")

    Dim lNumberOfSubDynamics
    lNumberOfSubDynamics = obj.GetNumberOfSubDynamics

    MsgBox "The number of immediate child sub-dynamics in" + _
           " GwxDigitalColor rectangle is: " + _
        CStr(lNumberOfSubDynamics)
```

Function GetObjectTypeName() As String

Возвращает имя класса текущего объекта.

Пример VBA:

```
Dim obj As Object
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")

Dim oDynamic As GwxDynamic
'GwxProcessPoint has all the properties
'and methods of GwxDynamic.

Set oDynamic = obj
'We can complete code via intellisence...

Dim oPP As GwxProcessPoint
Set oPP = oDynamic

MsgBox "The type name of the Process Point is: " + _
    oPP.GetObjectTypeName
```

Пример VB:

```
Dim obj
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")


MsgBox "The type name of the Process Point is: " + _
    obj.GetObjectTypeName
```

Function GetSubDynamicObjectFromIndex(index As Long) As Object

Возвращает дочерний по отношению к текущему динамический объект с определенным индексом (с отсчетом от нуля).Нулевой индекс имеет первый дочерний объект.
См. также **GetNumberOfSubDynamics()**.

Пример VBA:

```
'We sssume we have added a color dynamic
'to a rectangle with several tags.
'gfwsim.sine.double, etc.

Dim obj As Object
Set obj = ThisDisplay.GetDynamicObjectFromName("SubDynamic")

Dim oDynamic As GwxDynamic
'GwxDigitalColor has all the properties
'and methods of GwxDynamic.

Set oDynamic = obj
'We can complete code via intellisence...

Dim oDC As GwxDigitalColor
Set oDC = oDynamic

Dim obj2

Set obj2 = oDC.GetSubDynamicObjectFromIndex(2)
```

Пример VB:

```
'We sssume we have added a color dynamic
'to a rectangle with several tags.
'gfwsim.sine.double, etc.

Dim obj
Set obj = ThisDisplay.GetDynamicObjectFromName("SubDynamic")


Dim obj2
Set obj2 = obj.GetSubDynamicObjectFromIndex(2)
```

Function GetVisibleObject() As Object

Возвращает видимый объект, с которым связан текущий динамический объект.

Пример VBA:

```
Dim obj As Object
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")
'We assume that the process point has the name "ProcessPoint"

Dim oDynamic As GwxDynamic
'GwxProcessPoint has all the properties
'and methods of GwxDynamic.

Set oDynamic = obj
'We can complete code via intellisence...

Dim oPP As GwxProcessPoint
Set oPP = oDynamic

Dim obj2 As Object
Set obj2 = oPP.GetVisibleObject


MsgBox "The name of the visible object is: " + _
    obj2.objectName
```

Пример VB:

```
Dim obj
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPoint")
'We assume that the process point has the name "ProcessPoint"

Dim obj2
Set obj2 = obj.GetVisibleObject

MsgBox "The name of the visible object is: " + _
    obj2.objectName
```

Function SetAliasDefinition(aliasName As String, newDefinition As String) As Boolean

Выполняет настройку псевдонима для текущего динамического объекта. Возвращает FALSE, если не обнаружено псевдонимов с заданными именами, и TRUE, – в противном случае. (См. также **GwxDisplay::SetAliasDefinition** и **GwxVisible::SetAliasDefinition**).

Пример VBA:

```
Dim obj As Object
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPointWithAlias")

Dim oDynamic As GwxDynamic
'GwxProcessPoint has all the properties
'and methods of GwxDynamic.

Set oDynamic = obj
'We can complete code via intellisence...

Dim oPP As GwxProcessPoint
Set oPP = oDynamic

Dim strAliasName As String
strAliasName = "AliasString"

Call oPP.SetAliasDefinition(strAliasName, "{{gfwsim.cosine.double}}*20")

MsgBox "The new definition for the alias" + _
    "in the Process Point is {{gfwsim.cosine.double}}*20."
```

Пример VB:

```
Dim obj
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPointWithAlias")


Dim strAliasName
```

```
strAliasName = "AliasString"

Call obj.SetAliasDefinition(strAliasName, "{{gfwsim.cosine.double}}*20")
MsgBox "The new definition for the alias" + _
    "in the Process Point is {{gfwsim.cosine.double}}*20."
```

Function SetAliases(commandString As String) As Boolean

Выполняет настройку нескольких псевдонимов для текущего динамического объекта. (См. также **GwxDisplay::SetAliases()** и **GwxDynamic::SetAliasDefinition()**).

VBA пример:

```
Dim obj As Object
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPointWithAlias")

Dim oDynamic As GwxDynamic
'GwxProcessPoint has all the properties
'and methods of GwxDynamic.

Set oDynamic = obj
'We can complete code via intellisence...

Dim oPP As GwxProcessPoint
Set oPP = oDynamic

'Compose a command string:
'See GwxDisplay::SetAliasDefinition
Dim strAliasCommand As String
strAliasCommand = "AliasString" + Chr(9) + _

                  "100+abs({{gfwsim.sine.double}})*500"

MsgBox "Now we define another source for the GwxProcessPoint."
Call oPP.SetAliases(strAliasCommand)
```

Пример VB:

```
Dim obj
Set obj = _
ThisDisplay.GetDynamicObjectFromName("ProcessPointWithAlias")


'Compose a command string:
'See GwxDisplay::SetAliasDefinition
Dim strAliasCommand
strAliasCommand = "AliasString" + Chr(9) + _
                   "100+abs({{gfwsim.sine.double}})*500"

MsgBox "Now we define another source for" + _
            " the GwxProcessPoint." + _
       "100+Abs({{gfwsim.sine.double}})*500"

Call obj.SetAliases(strAliasCommand)
```

Function SetAliasesFromFile(filename As String) As Boolean

Выполняет настройку нескольких псевдонимов для текущего динамического объекта по конфигурационной информации из файла настройки псевдонимов. (См. также **GwxDisplay::SetAliasesFromFile()** и **GwxDynamic::SetAliasDefinition()**).

Пример VBA:

```
Dim strTextFileName As String

'Specify a text file. Take care all
'of the white spaces (only TABS) dividing the strings.
'Set to use real tabs in your editor.
'Template is:
'aliasName1<TAB>aliasDef1<CR-LF>
'aliasName2<TAB>aliasDef2<CR-LF>
'....
'..
'.
```

```
strTextFileName = "AliasCommandString.txt"

MsgBox "Now we define another source for the GwxProcessPoint."


Dim obj As Object
Set obj = ThisDisplay.GetDynamicObjectFromName("ProcessPointWithAlias")


Dim oDynamic As GwxDynamic
'GwxProcessPoint has all the properties
'and methods of GwxDynamic.

Set oDynamic = obj
'We can complete code via intellisence...

Dim oPP As GwxProcessPoint
Set oPP = oDynamic


Call oPP.SetAliasesFromFile(strTextFileName)
```

Пример VB :

```
Dim strTextFileName

'Specify a text file. Take care all
'of the white spaces (only TABS) dividing the strings.
'Set to use real tabs in your editor.
'Template is:
'aliasName1<TAB>aliasDef1<CR-LF>
'aliasName2<TAB>aliasDef2<CR-LF>
'....
'..
'.
strTextFileName = "AliasCommandString.txt"

MsgBox "Now we define another source for the GwxProcessPoint."


Dim obj
Set obj = _

ThisDisplay.GetDynamicObjectFromName("ProcessPointWithAlias")



Call obj.SetAliasesFromFile(strTextFileName)
```

GwxDigitalSelector

Класс **GwxDigitalSelector** предназначен для попеременного отображения отдельных объектов, входящих в группу, в зависимости от состояния дискретного (булева) сигнала. Объекты класса **GwxDigitalSelector** могут быть связаны только с объектами класса **GwxSymbol** (объект класса **GwxSymbol** определяет набор видимых объектов, к которым применяется динамическое действие Цифровой индикатор). **GwxDigitalSelector** имеет все свойства и методы класса **GwxDynamic**. Описание дополнительных методов приведено ниже.

Методы

Function SetConnectionInfo(objectNumber As Integer, dataSource As String, showWhenTrue As Boolean) As Boolean

Устанавливает соединения с источниками данных для текущего объекта типа *Цифровой индикатор*. Аргумент **objectNumber** является индексом, представляющим ближайший по уровню вложенности объект символа, с которым связывается данное динамическое действие (нулевому индексу соответствует первый объект). См. также **GwxDigitalSelectorInfo**.

GwxDigitalSelectorInfo

Представляет индивидуальные соединения с источниками данных для объектов класса **GwxDigitalSelector**. **GwxDigitalSelectorInfo** имеет все свойства и методы класса **GwxDynamic**.

GwxDigitalColor

Объект класса **GwxDigitalColor** используется для дискретного изменения цвета видимого объекта, с которым он связан, в зависимости от логического состояния одного или нескольких цифровых сигналов. **GwxDigitalColor** наследует все свойства и методы класса **GwxDynamic**. Описание дополнительных методов приведено ниже.

Методы

Function DeleteSubDynamic(index As Integer) As Boolean

Удаляет дочерний динамический объект с определенным индексом. В случае успешного выполнения возвращает true, в случае неуспешного (обычно если индекс выходит за допустимые пределы) – false.

Function SwapSubDynamics(index1 As Integer, index2 As Integer) As Boolean

Меняет местами индексы дочерних динамических объектов с индексами index1 и index2. В случае успешного выполнения возвращает true, в случае неуспешного (обычно если индекс выходит за допустимые пределы) – false.

GwxDigitalColorInfo

Представляет индивидуальные соединения с источниками данных для объектов класса **GwxDigitalColor**. **GwxDigitalColorInfo** наследует все свойства и методы класса **GwxDynamic**. Описание дополнительных свойств приведено ниже.

Свойства.

UseFillColor As Boolean

UseLineColor As Boolean

UseShadowColor As Boolean

Свойства разрешают изменения цвета видимых объектов.

FillColor As OLE_COLOR

LineColor As OLE_COLOR

ShadowColor As OLE_COLOR

Свойства определяют цвета для операции с цветом.

ChangeColorOnTrue As Boolean

Свойство определяет, выполняется ли действие, когда тег OPC принимает значение true или false.

GwxAnalogSelector

Класс **GwxAnalogSelector** предназначен для попеременного отображения отдельных объектов, входящих в группу, в зависимости от значения аналогового сигнала. Объекты класса **GwxAnalogSelector** могут быть связаны только с объектами класса **GwxSymbol** (объект класса **GwxSymbol** определяет набор видимых объектов, к которым применяется динамическое действие *Аналоговый индикатор*). **GwxAnalogSelector** имеет все свойства и методы класса **GwxDynamic**. Описание дополнительных методов приведено ниже.

Методы

Function SetSelectorRanges(rangeArray) As Boolean

Устанавливает диапазон для динамического объекта. Возвращает TRUE при успешном выполнении, иначе - FALSE.

Параметр **rangeArray** – массив значений с плавающей точкой (float). Количество элементов массива должно быть равно количеству объектов в объекте **analog selector**. Индекс элементов массива соответствует индексу объектов в **analog selector.** Каждый элемент массива представляет собой относительное значение верхнего предела диапазона соответствующего объекта. Значения элементов массива могут быть в диапазоне от 0 до 1 и каждый следующий элемент должен быть больше предыдущего. Значение последнего элемента массива всегда должно быть 1.

GwxAnalogColor

Объект данного класса предназначен для плавного изменения цвета видимого объекта, с которым он связан, в зависимости от значения аналогового сигнала. **GwxAnalogColor** наследует все свойства и методы класса **GwxDynamic**.

GwxAnimator

Объект данного класса предназначен для попеременного отображения в определенной последовательности отдельных объектов из заданного набора в зависимости от прошедшего времени. Объект класса **GwxAnimator** активизируется/блокируется в зависимости от состояния связанного дискретного сигнала. Объекты класса **GwxAnimator** могут быть связаны только с объектами класса **GwxSymbol** (объект класса **GwxSymbol** определяет набор видимых объектов, к которым применяется динамическое действие *Анимация*). **GwxAnimator** имеет все свойства и методы класса **GwxDynamic**.

GwxSize

Объект данного класса предназначен для изменения размеров видимого объекта, с которым он связан, в зависимости от значения аналогового сигнала. **GwxSize** имеет все свойства и методы класса **GwxDynamic**.

GwxLocation

Объект данного класса предназначен для изменения местоположения видимого объекта, с которым он связан, в зависимости от значения аналогового сигнала. **GwxLocation** имеет все свойства и методы класса **GwxDynamic**. Описание дополнительных методов приведено ниже.

Методы

Sub GetCoordinateFromLocationDelta(offset As Single, cx, cy)

Дополнительная возможность для опытных пользователей, которым необходимо разрабатывать сложные приложения. В обычных приложениях этот метод не используется.

Метод позволяет получать координаты, которые будет иметь объект с динамикой **Положение/Движок** при определенном значении тега динамики (параметр **offset**). Метод очень удобен для определения положения, которое будет занимать объект на сложной траектории (например, поезд, начинающий движение из определенной точки). Выполняется преобразование координат из линейных в двумерные (координаты на экране). В методе используется такой же алгоритм, как и в динамике **Положение/Движок** для определения положения объекта на заданной траектории в зависимости от значения тега.
Использование этого метода в комбинации с **setobjectdimension** позволяет изменить положение нескольких объектов экранной формы и перемещать их по траектории. Минимальное и максимальное значения параметра **offset** такие же, как и настройки динамики **Положение/Движок** Мин. и Макс.

Function GetLastEnteredValue()

Возвращает последнее введенное значение, отличающееся от текущего.

GwxRotation

Объект данного класса предназначен для изменения угла поворота видимого объекта, с которым он связан, в зависимости от значения аналогового сигнала. **GwxRotation** имеет все свойства и методы класса **GwxDynamic**. Описание дополнительных методов приведено ниже.

Методы

Function GetLastEnteredValue()

Возвращает последнее введенное значение, отличающееся от текущего.

GwxFlash

Объект данного класса предназначен для переключения цвета или попеременного скрытия/отображения с заданной частотой видимого объекта, с которым он связан, в зависимости от состояния дискретного сигнала. **GwxFlash** имеет все свойства и методы класса **GwxDynamic**.

GwxHide

Объект данного класса предназначен для попеременного скрытия/отображения блокировки/активизации с заданной частотой видимого объекта, с которым он связан, в зависимости от состояния дискретного сигнала. **GwxHide** имеет все свойства и методы класса **GwxDynamic**.

GwxPick

После связывания объекта класса **GwxPick** с видимым объектом, при щелчке мышью на видимом объекте в режиме *Исполнение* будет выполняться заданная операция. Объекты класса **GwxPick** могут быть связаны с объектами типа **GwxButton**, после чего последние получают функциональность реальных кнопок. **GwxPick** имеет все свойства и методы класса **GwxDynamic**. Описание дополнительных свойств приведено ниже.

Свойства

aliasCommandString As String

Если свойство **aliasCommandType** равно **AliasSetFromFile**, свойство **aliasCommandString** должно содержать имя текстового файла привязки псевдонимов, каждая строка которого содержит имя псевдонима и, отделенное символом табуляции, имя источника данных (см. также **SetAliasesFromFile()**).
Если свойство **aliasCommandType** равно **AliasSetDirect**, свойство **aliasCommandString** должно иметь следующий формат:

ИмяПсевдонима1<TAB>ОпределениеПсевдонима1<CR-LF>ИмяПсевдонима2<TAB>
ОпределениеПсевдонима2<CR-LF> … и т.д.

aliasCommandType As GWXALIASCOMMAND

AliasCommandType может принимать одно из следующих значений:

- AliasNoCommand (настройка псевдонимов не выполняется) = 0
- AliasSetFromFile (настройка псевдонимов из файла) = 1
- AliasSetDirect (непосредственная настройка псевдонимов) = 2

NameParameter As String

Назначение параметра **NameParameter** зависит от значения свойства **PickAction**.
Если **PickAction** равно PickLoadDisplay ("*Загрузить экранную форму*"), PickDragDropLoad ("*Загрузка в стиле drag-n-drop*"), PickPopupWindow ("*Всплывающее окно*") или PickEmbeddedWindow ("*Встроенное окно*"), **NameParameter** является именем файла экранной формы GraphWorX32 (.gdf).
Если **PickAction** равно PickLaunchApp ("*Запустить приложение*"), **NameParameter** является именем исполняемого файла (.exe).
Если **PickAction** равно PickRunScript ("*Выполнить VBA-сценарий*"), **NameParameter** является именем VBA-сценария.
Если **PickAction** равно PickSetAliases или PickAliasesDialog ("*Настройка псевдонимов*" или "*Диалог настройки псевдонимов*"), **NameParameter** является именем видимого объекта.

pickAction As GWXPICKACTION

См. **GwxDisplay::CreatePickDynamic** для получения информации о назначении данного свойства.

ScriptText As String

GraphWorX32 8.0 позволяет выполнять по событиям сценарии VBScript и JScript. Сценарии, связанные с событием, выполняются при возникновении события (например, после запуска приложения или перед запросом данных для слоя). Сценарии могут редактироваться с помощью новой интегрированной панели инструментов или с помощью интерфейса **OLE automation**. Сценарий может быть написан с использованием языков VBScript or Jscript независимо от другого события. Это свойство позволяет читать и записывать исходный код сценария, связанного с нажатием кнопки.

ScriptType As GWXSCRIPTTYPE

GraphWorX32 7.0 позволяет выполнять по событиям сценарии VBScript и JScript. Сценарии, связанные с событием, выполняются при возникновении события (например, после запуска приложения или перед запросом данных для слоя). Сценарии могут редактироваться с помощью новой интегрированной панели инструментов или с помощью интерфейса **OLE automation**. Сценарий может быть написан с использованием языков VBScript or Jscript независимо от другого события. Это свойство позволяет получать и устанавливать язык сценария, связанного с нажатием кнопки.

```
'VBA example, works only in configure mode
'example for creating pick action PickRunScript and set script
text and type

Dim o As Object
'creates rectangle
Set o = ThisDisplay.CreateRectangle( _
      50, 50, 100, 100, True, 100, 0, 2, _
      LineSolid, False, 0, EdgeNone, False, _
      "rectangle", False)
'adds pick action PickRunScript on the rectangle
Set o = ThisDisplay.CreatePickDynamic( _
      "rectangle", "dynamic", PickRunScript, _
      ButtonNormal, TriggerOnDown, MouseButtonLeft, _
      False, "", "", False, False, "", "", "")
'sets script properties - name of the script is
'derived from pick action name
o.ScriptType = ScriptTypeVBScript
o.ScriptText = "MsgBox " & Chr(34) & "Test OK" & Chr(34)
```

tracking As Boolean

Для динамических объектов **Кнопка /Передать** или **Переключить значение**, а также для **Флажка** и **Радиокнопки** установка свойству **tracking** значения **TRUE** приводит к установке параметра **Следить за значением**.

Методы

Sub SimulateClick(mouseButton As GWXMOUSEBUTTON)

Выполняет имитацию нажатия кнопки, как если бы кнопка была нажата пользователем.

GwxProcessPoint

Объекты класса **GwxProcessPoint** ("*Параметр процесса*"), связанные с объектами класса **GwxText**, создают на их базе алфавитно-цифровые индикаторы для отображения значений контролируемых параметров, а также поля ввода алфавитно-цифровой информации. **GwxProcessPoint** имеет все свойства и методы класса **GwxDynamic**. Описание дополнительных свойств приведено ниже.

Свойства

ConfirmBeforeWrite As Boolean

TRUE to prompt the user with a confirmation message before writing data values.

UseKeypad As Boolean

TRUE to use an onscreen number pad or keyboard for this data entry.

Методы

Function GetLastEnteredValue()

Returns the last entered value, which might be different from the currently displayed value.

GwxTimedate

Объекты класса **GwxTimedate** ("Время и дата"), связанные с объектами класса **GwxText**, создают на их базе поле отображения текущих даты и/или времени. **GwxTimedate** имеет все свойства и методы класса **GwxDynamic**.

GwxPoint

GwxPoint является классом, который используется GraphWorX32 для представления соединений с источниками данных. Если несколько объектов класса **GwxDynamic** соединены с одним и тем же источником данных, все они ссылаются на один общий объект класса **GwxPoint**. Объекты **GwxPoint** осуществляют взаимодействие с тегами OPC, выражениями, константами и локальными переменными GraphWorX32. Явное (непосредственное) создание и удаление объекта класса **GwxPoint** не допускается. GraphWorX32 автоматически управляет временем жизни объектов класса **GwxPoint**, основываясь на соединениях объектов класса GwxDynamic экранной формы с теми или иными источниками данных.

Свойства

dataType As GWXDATATYPE

Тип данных объекта *Параметр*. Допустимые значения **GWXDATATYPE**:

- DataTypeShort = 2
- DataTypeLong = 3
- DataTypeFloat = 4
- DataTypeDouble = 5
- DataTypeString = 8
- DataTypeBool = 11
- DataTypeChar = 16
- DataTypeByte = 17
- DataTypeWord = 18
- DataTypeDword = 19

highRange As Variant

Верхняя граница диапазона значений объекта *Параметр*.

lowRange As Variant

Нижняя граница диапазона значений объекта *Параметр*.

Value As Variant

Текущее значение объекта *Параметр*. Данное поле объекта обновляется системой исполнения GraphWorX32 новыми значениями связанного источника данных.

Методы

Function GetObjectTypeName() As String

Возвращает имя типа текущего объекта.

Function GetPointName() As String

Возвращает имя объекта типа *Параметр*, которое является строкой идентификатора источника данных.

Глава 14. Проект VBA в GraphWorX32.

14.1 Введение

14.2 Управление объектами GraphWorX32 через интерфейсы автоматизации (OLE Automation) с использованием VBA

14.3 Использование VBA для связи с другими приложениями

14.4 Другие источники информации

14.1 Введение

Проект VBA всегда загружается при запуске редактора Visual Basic из приложения GraphWorX32. Данный проект содержит группы исходных модулей, к которым относятся:

- Встроенные объекты GraphWorX32
- Модули *ThisDisplay* и *GwxTools*
- Формы.

Каждый модуль может содержать исходный текст VBA – функции, процедуры, обработчики событий и глобальные описания (объявления).

Модуль *GwxTools*

GwxTools является специализированным модулем с общими процедурами, которые используются *Мастером VBA-сценариев*, описание которого приведено в следующем разделе настоящего руководства. Модуль *GwxTools* недоступен до тех пор, пока пользователем не был запущен *Мастер VBA-сценариев*.

Модуль *ThisDisplay*

ThisDisplay является специализированным модулем, который представляет текущую загруженную экранную форму GraphWorX32.

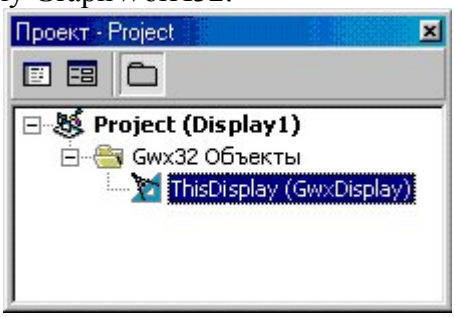


*Рис. 14-3. Окно проекта с выбранным модулем **ThisDisplay***

Для отображения исходного текста модуля *ThisDisplay* выполните двойной щелчок левой клавишей мыши на элементе списка *ThisDisplay (GwxDisplay)* в окне проекта. Исходный текст модуля будет отображен в одном из окон программ, как показано на рис. 14-3.

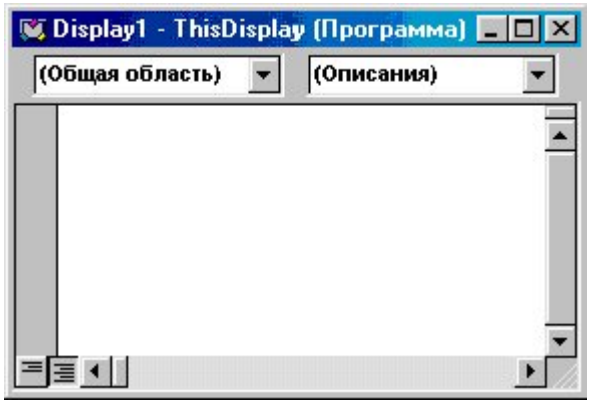


*Рис. 14-4. Окно исходного текста модуля **ThisDisplay***

Выпадающий список, расположенный слева в верхней части окна программы позволяет выбрать следующие элементы:

- (*Общая область*)
- *GwxDisplay*
- Идентификаторы всех управляющих элементов, вставленных в экранную форму GraphWorX32.

Выпадающий список, расположенный справа в верхней части окна программы позволяет выбрать события, относящиеся к объекту, выбранному в левом выпадающем списке. Например, для объекта *ThisDisplay* могут быть выбраны такие события, как *DisplayLoad*, *DisplayUnload* и другие.

Обработка событий в модуле *ThisDisplay*

Обработчики событий могут быть включены в проект GraphWorX32 путем использования VBA. Для этого необходимо иметь загруженную экранную форму.

1. Вызовите редактор VBA путем выбора строки **Редактор Visual Basic** подменю **Макрос** меню **Сервис** или нажатием комбинации клавиш Alt+F11. Произойдет загрузка проекта VBA в GraphWorX32.
2. Откройте модуль *ThisDisplay* двойным щелчком левой клавишей мыши на его названии окне проекта.
3. Выберите элемент *GwxDisplay* в выпадающем списке, расположенном слева в верхней части окна исходного текста модуля *GwxDisplay*.
4. Выберите название требуемого события в выпадающем списке, расположенном справа в верхней части окна исходного текста модуля. Строки заголовка и завершения процедуры будут автоматически вставлены в модуль.
5. Введите исходный текст обработчика выбранного события в тело процедуры.
6. Сохраните проект, для чего выберите строку Сохранить **<Имя_экранной_формы>** меню **Файл**.
7. Завершите работу редактора Visual Basic, выбрав строку **Закрыть и вернуться в Gwx32** меню **Файл**.

Пример

Создание VBA-сценария, выводящего на экран сообщение "Система переведена в режим Исполнение", когда GraphWorX32 переходит в режим *Исполнение*.

1. Выполните действия пп. 1...3 п. 13.4.3 для выбора модуля *GwxDisplay*.
2. Выберите событие PostRuntimeStart в выпадающем списке, расположенном справа в верхней части окна исходного текста модуля.
3. Введите исходный текст, показанный на рис. 13-4.
4. Для перехода в окно экранной формы GraphWorX32 закройте редактор Visual Basic или нажмите комбинацию клавиш Alt+Tab.
5. Переведите GraphWorX32 в режим *Исполнение*, для чего щелкните мышью на заголовке меню **Исполнение**.
6. После перехода GraphWorX32 в режим *Исполнение* будет воспроизведен короткий звуковой сигнал, после чего на экране монитора появится сообщение "Система переведена в режим Исполнение".

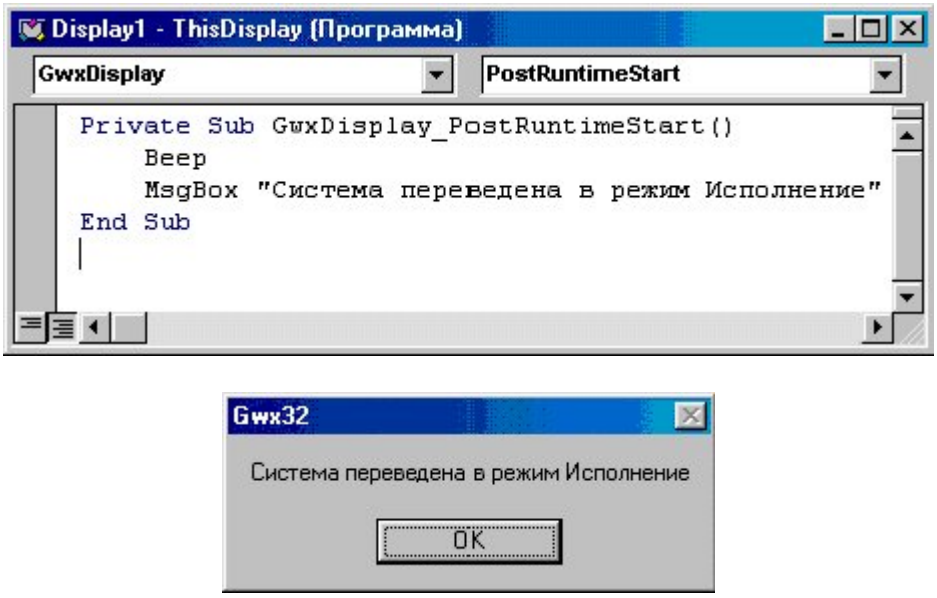


Рис. 14-5. Пример простейшего сценария VBA

14.2 Управление объектами GraphWorX32 через интерфейсы автоматизации (OLE Automation) с использованием VBA

Разработчик на языке VBA имеет возможность управлять всеми свойствами и встроенными объектами текущей экранной формы, включая управление динамизацией графических объектов.

Свойства и методы автоматизации доступны в пределах модуля *ThisDisplay*. Данный модуль, как указывалось выше, является представлением текущей экранной формы GraphWorX32 и содержит методы и свойства всех присутствующих в ней объектов.

Примечание. Методы и свойства автоматизации экранной формы доступны в VBA только в случае, если в диалоговой панели **Ссылки**, вызов которой осуществляется выбором строки **Ссылки** меню **Сервис** редактора VBA, отмечен элемент *Gwx32* списка доступных объектов. При создании новой экранной формы GraphWorX32 данный элемент отмечается автоматически. Внешний вид диалоговой панели **Ссылки** показан на рис.14-6.

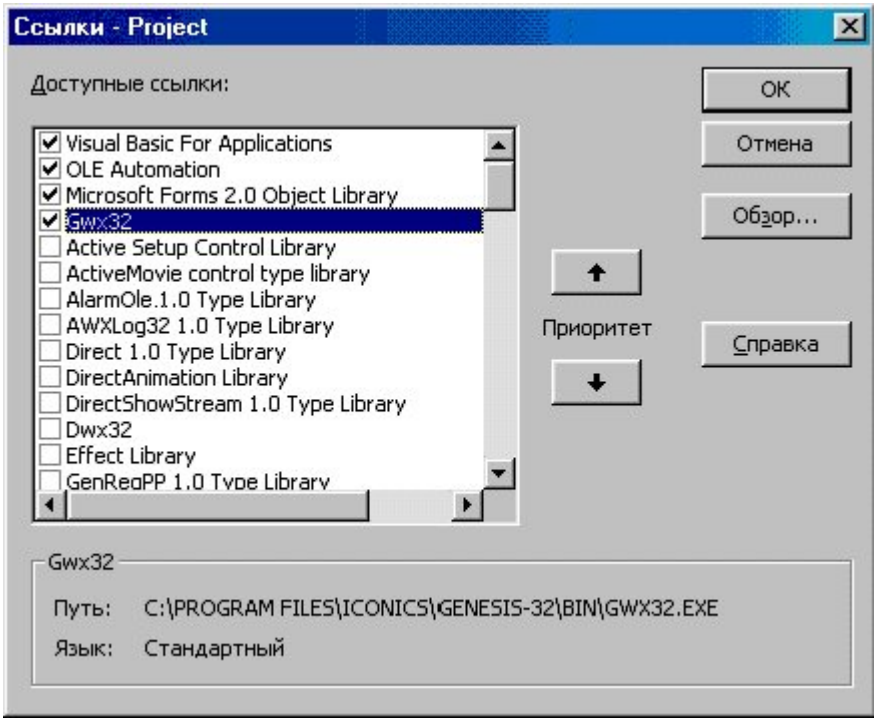


*Рис. 14-6. Диалоговая панель **Ссылки** редактора VBA*

Доступ ко встроенным графическим объектам экранных форм GraphWorX32

Встроенные графические объекты GraphWorX32, включая эллипсы, прямоугольники, символы и динамические соединения, изначально не доступны в VBA. Однако доступ к ним из сценарных процедур VBA может осуществляться путем использования идентификаторов объектов, вводимых в поля **Имя объекта** диалоговой панели **Инспектор свойств**, в качестве ссылок на объекты.
Ссылка на поименованный объект (фактически, являющаяся диспетчеризуемым указателем) может быть разыменована (разрешена) путем использования следующих методов объектов *ThisDisplay*, *symbol* и *Visible*:

- ThisDisplay.GetVisibleObjectFromName
- ThisDisplay.GetVisibleObjectFromIndex
- ThisDisplay.GetVisibleObjectFromName
- Visible.GetDynamicObjectFromIndex
- Visible.GetDynamicObjectFromName
- symbol.GetVisibleObjectFromIndex
- symbol.GetVisibleObjectFromName

ThisDisplay.GetVisibleObjectFromName("*Имя*")

Данная функция возвращает ссылку на графический объект (далее, – "возвращает объект"), связанный с передаваемым именем. После сохранения результата выполнения данной функции в переменной, имеется возможность изменять свойства объекта, находящегося в пределах экранной формы, путем установки соответствующих полей переменной. В приведенном ниже примере производится поворот объекта *Прямоугольник* с именем "*Square*" на угол 45 градусов относительно горизонтальной оси.

```
Dim obj As GwxRectangle
Set obj = ThisDisplay.GetVisibleObjectFromName("Square")
obj.Angle = 45
```

ThisDisplay.GetDynamicObjectFromName("*Имя*")

Данная функция возвращает динамический объект экранной формы, связанный с передаваемым именем. В данном случае динамический объект не является физическим или визуально воспринимаемым, а лишь абстрактным представлением динамического действия, примененного к реальному графическому объекту (например, *Скрыть*, *Вращение*, *Размер* и т.п.). В приведенном ниже примере источник данных, связанный с

динамикой объекта *Скрыть/Блокировать*, которая имеет имя "*hd*", заменяется на константу, равную 0, что приводит к скрытию графического объекта.

```
Dim obj As GwxHide
Set obj = ThisDisplay.GetDynamicObjectFromName("hd")
obj.dataSource = 0
```

ThisDisplay.GetVisibleObjectFromIndex(long lObjectIndex)

Данная функция возвращает объект экранной формы, основываясь на том, в каком порядке происходило создание всех графических объектов текущей экранной формы. Самый первый графический объект, который был размещен в рабочей области вновь созданной экранной формы, получает индекс 0. Данная функция особенно удобна при последовательном выполнении операций над всеми объектами экранной формы. В приведенном ниже примере самый первый созданный объект экранной формы изменяет цвет заливки на зеленый, а второй – на красный.

```
Dim obj1 As Object
Dim obj2 As Object
Set obj1 = ThisDisplay.GetVisibleObjectFromIndex(0)
Set obj2 = ThisDisplay.GetVisibleObjectFromIndex(1)
obj1.fillColor = RGB(0, 255, 0)
obj2.fillColor = RGB(255, 0, 0)
```

SymbolName.**GetVisibleObjectFromIndex(long lObjectIndex)**

Каждый символ GraphWorX32 имеет собственный список индексов всех графических объектов, из которых сгруппирован данный символ. Метод GetVisibleObjectFromIndex, выполняющийся относительно символа, возвращает ссылку на графический объект внутри символа, имеющий индекс, который передается методу. Данная функция удобна при последовательном выполнении операций над графическими объектами, из которых сгруппирован символ. В приведенном ниже примере производится изменение цвета заливки графического объекта с номером 3 внутри символа с именем "*sym*" на зеленый.

```
Dim sym1 As GwxSymbol
Dim obj As Object
Set sym1 = ThisDisplay.GetVisibleObjectFromName("sym")
Set obj = sym1.GetVisibleObjectFromIndex(2)
obj.fillColor = RGB(0, 255, 0)
```

(Обратите внимание на тот факт, что объект с номером 1 имеет индекс 0.)

SymbolName.**GetVisibleObjectFromName("***Имя***")**

Данный метод возвращает объект внутри символа с именем *SymbolName* по передаваемому имени, присвоенному данному графическому объекту GraphWorX32.

VisibleObjectName.**GetDynamicObjectFromIndex(long lObjectIndex)**

При каждом применении к поименованному графическому объекту динамического соединения (действия), данному действию присваивается индекс. Самое первое динамическое действие имеет индекс 0. Поскольку к одному графическому объекту может быть применено множество динамических действий, данная функция особенно удобна для манипуляций с несколькими или со всеми динамическими действиями, связанными с графическим объектом. В приведенном ниже примере получается ссылка на второе динамическое действие, связанное с объектом "*rect*", после чего значение нижней границы диапазона изменения связанного источника данных устанавливается равной 10.

```
Dim o_Vis As GwxRectangle
Dim o_Dyn As Object
Set o_Vis = ThisDisplay.GetVisibleObjectFromName("rect")
Set o_Dyn = o_Vis.GetDynamicObjectFromIndex(1)
o_dyn.lowRange = 10
```

VisibleObjectName.**GetDynamicObjectFromName("***Имя***")**

Данный метод возвращает динамическое соединение внутри графического объекта с именем *VisibleObjectName* по передаваемому имени, присвоенному данному динамическому соединению.

Для получения более подробной информации обратитесь к справочнику по OLE Automation GraphWorX32.

Пример

В данном примере выполняется разработка VBA-сценария, который изменяет цвет встроенного объекта эллипс GraphWorX32 в режиме *Исполнение*.

1. Создайте новую экранную форму GraphWorX32.
2. Нарисуйте эллипс в рабочей области экранной формы.
3. Присвойте графическому объекту *Эллипс* имя *MyEllipse*, для чего:
   - Дважды щелкните левой клавишей мыши на изображении эллипса.
   - В поле **Имя объекта** появившейся диалоговой панели **Инспектора свойств** введите *MyEllipse*.
   - Закройте **Инспектор свойств**.
4. Вызовите редактор VBA, для чего нажмите комбинацию клавиш Alt+F11.

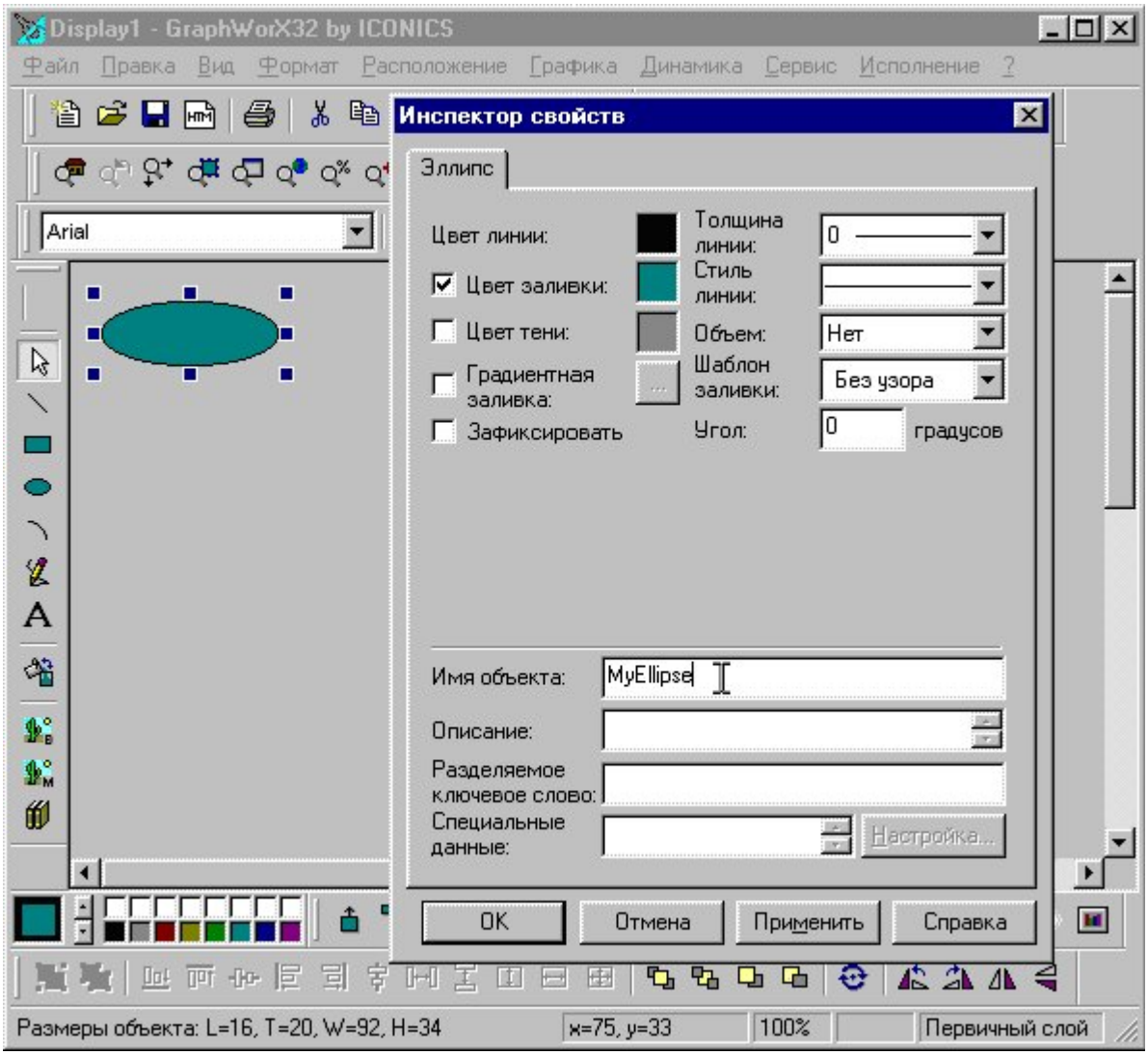


Рис. 14-7. Присвоение имени объекту Эллипс

5. В окне проекта дважды щелкните на названии модуля *ThisDisplay*.
6. В окне программы в верхнем левом выпадающем списке выберите элемент *GwxDisplay*.
7. В окне программы в верхнем правом выпадающем списке выберите название события PostRuntimeStart. Заголовок и строка завершения процедуры будут автоматически вставлены в окно программы. Введите исходный текст процедуры, показанный на рис. 14-7.

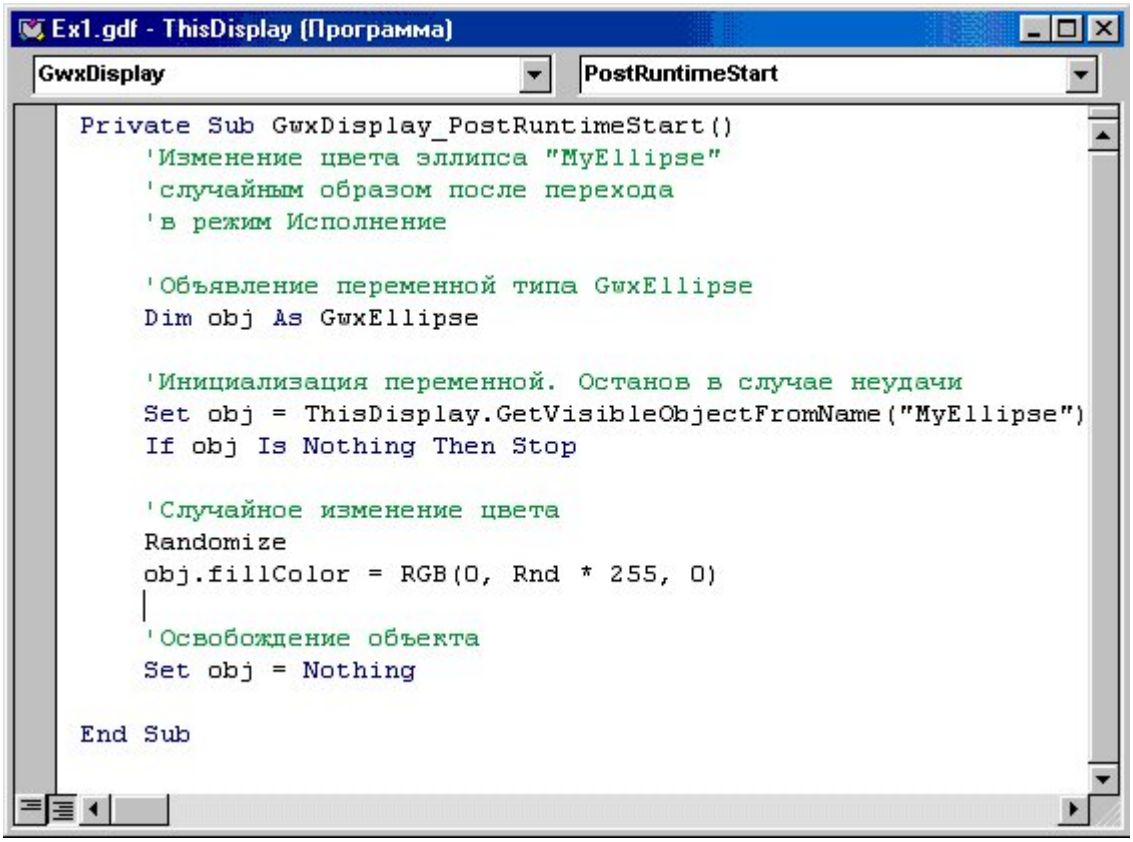


Рис. 14-8. Исходный текст процедуры установки цвета объекта после перехода в режим Исполнение

8. Выберите строку **Сохранить** меню **Файл** для сохранения проекта GraphWorX32.
9. Проверьте работоспособность примера, переведя GraphWorX32 в режим *Исполнение*.
10. Цвет заливки эллипса после перехода в режим Исполнение будет изменен случайным образом.

Уникальность имен объектов

Иногда необходимо получать доступ к одним и тем же графическим объектам внутри всех копий (дубликатов) одного и того же символа. В следующем разделе описан удобный способ выполнения указанного доступа при помощи *Мастера VBA-сценариев*, который позволяет создавать макросы, связанные с символами GraphWorX32. Данный макрос совместно используется всеми копиями каждого символа, и может быть запущен применительно к любой копии.

Для реализации указанной возможности GraphWorX32 поддерживает так называемые "неполные" имена объектов, из которых сгруппирован символ. Неполным является имя, которое заканчивается знаком подчеркивания '_', например, – *MyEllipse_*. В результате копии объекта *Ellipse_* получат следующие имена: *MyEllipse_1*, *MyEllipse_2*, ...

Метод GetVisibleObjectFromName("*Неполное_имя*") может принимать в качестве параметра указанное неполное имя. Результатом выполнения метода является ссылка на первый найденный объект с заданным неполным именем внутри выбранного символа. Ниже приведен пример выполнения доступа по неполному имени.

Пример

Предположим, что имеется символ, состоящий из прямоугольника и эллипса. Необходимо разработать сценарий, который изменяет свойства указанных объектов во всех копиях данного символа. Следует присвоить частичные имена обоим объектам (прямоугольнику и эллипсу), например, *MyRect_* и *MyEll_*. Тогда доступ к указанным объектам внутри всех копий символа выполняется следующим образом.

```
Dim sym As GwxSymbol

' Функция пользователя для выбора символа
Set sym = КакНибудьБеремСимвол()

Dim ell As GwxEllipse, rect As GwxRectangle
Set ell= sym.GetVisibleObjectFromName("MyRect_")
Set rect= sym.GetVisibleObjectFromName("MyEll_")

' Что-нибудь делаем с эллипсом и прямоугольником во всех копиях символа

' Освобождение объектов
Set ell = nothing
Set rect = nothing
Set sym = nothing
```

Пример

Следующий пример демонстрирует описанную выше технологию.

1. Создайте один прямоугольник в GraphWorX32 и присвойте ему имя *Rect_1*.
2. Нажатием комбинации клавиш Ctrl+D создайте копию данного прямоугольника. Обратите внимание на то, что копия получит имя *Rect_2*.
3. Сгруппируйте объекты *Rect_1* и *Rect_2*.
4. Создайте копию сгруппированных объектов нажатием комбинации клавиш Ctrl+D.
5. Вызовите контекстное меню для копии сгруппированных объектов и выберите в нем команду **Редактировать символ**. Поочередно вызывая **Инспектор свойств** для каждого прямоугольника внутри копии символа, убедитесь, что прямоугольники имеют имена *Rect_3* и *Rect_4*.

14.3 Использование VBA для связи с другими приложениями

Одной из наиболее мощных возможностей, предоставляемых VBA, является связь и обмен данными с другими приложениями. В приведенном ниже примере иллюстрируется обмен данными между экранной формой GraphWorX32 и рабочим листом Excel с использованием VBA.

Пример

В данном примере создаются экранная форма GraphWorX32 и лист MicrosoftExcel. Далее производится обмен данными между содержимым двух ячеек рабочего листа Excel и двумя встроенными объектами *Параметр* GraphWorX32.

1. Создайте экранную форму GraphWorX32 и вызовите редактор VBA, для чего нажмите комбинацию клавиш Alt+F11 или выберите строку **Редактор VBA** подменю **Макрос** меню **Сервис**.
2. Первое, что требуется сделать в редакторе VBA GraphWorX32, это установить связь с типами данных, используемых в Microsft Excel. Выберите строку **Ссылки** меню **Сервис** редактора VBA GraphWorX32 и в списке доступных библиотек отметьте флажок *Microsoft Excel 8.0 Object Library*, как показано на рис. 14-8, после чего закройте диалоговую панель **Ссылки**. При использовании предыдущей версии Microsoft Excel отметьте флажок *Microsoft Excel 5.0 Object Library*.

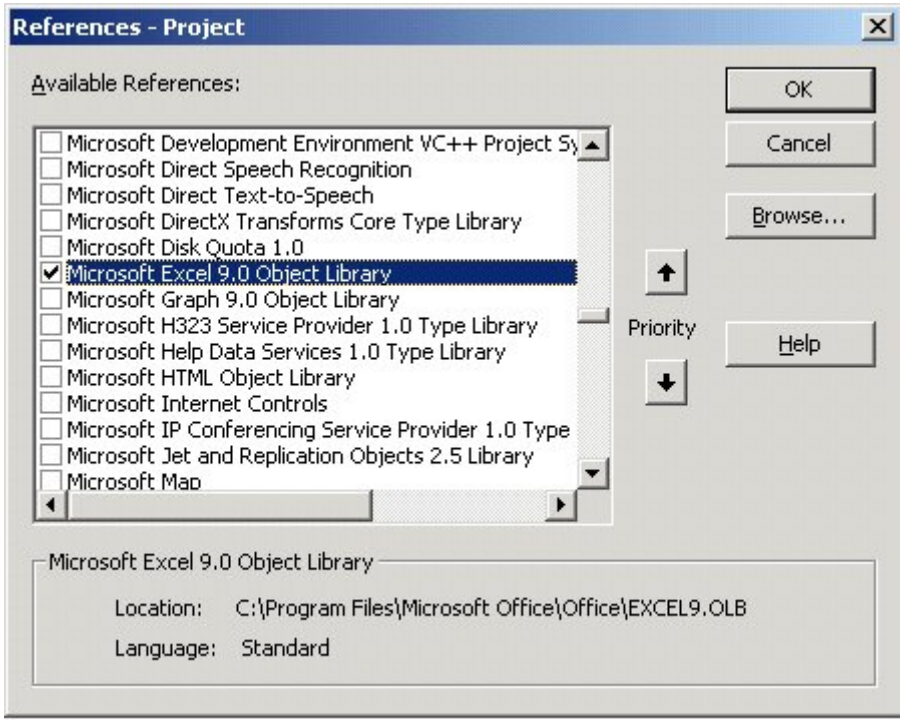
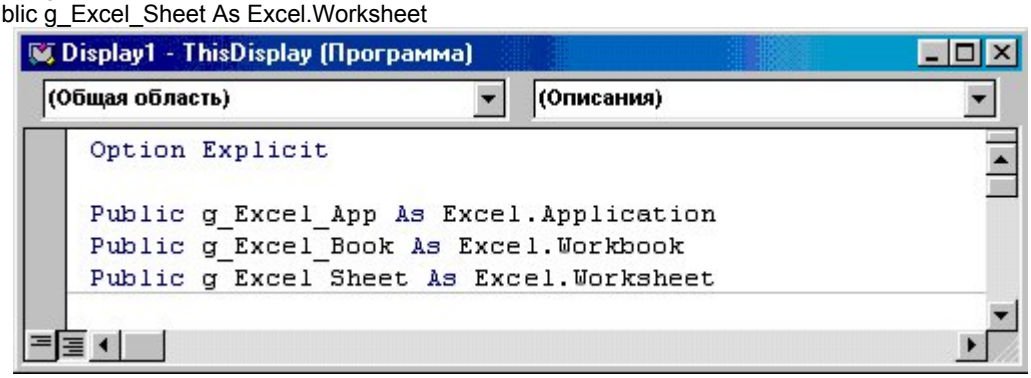


Рис. 14-10. Получение доступа к объектам Excel в инструментальной среде разработки

3. В окне проекта редактора VBA выполните двойной щелчок левой клавишей мыши на элементе *ThisDisplay*. Введите в окне программы строку "Option Explicit". В данной секции окна программы могут быть сделаны объявления глобальных переменных и введены прототипы функций, используемых во всех процедурах.

```
Option Explicit
 Public g_Excel_App As Excel.Application
Public g_Excel_Book As Excel.Workbook
Public g_Excel_Sheet As Excel.Worksheet
```



4. В окне программы в верхнем левом выпадающем списке выберите элемент *GwxDisplay*. В окне программы в верхнем правом выпадающем списке выберите название события PreRuntimeStart. Заголовок и строка завершения процедуры будут автоматически вставлены в окно программы. Введите следующий исходный текст процедуры, которая вызывается непосредственно перед активизацией режима *Исполнение* GraphWorX32.

```
Private Sub GwxDisplay_PreRuntimeStart()
  ' Открыть Excel и сделать его видимым
 Set g_Excel_App = CreateObject("Excel.application")
 g_Excel_App.Visible = True
  ' Открыть лист Excel
 Set g_Excel_Book = g_Excel_App.Workbooks.Add
 Set g_Excel_Sheet = g_Excel_Book.Worksheets(1)
  ' Инициализировать две ячейки,
' которые будут использоваться для обмена данными
  g_Excel_Sheet.Range("a1") = 0
g_Excel_Sheet.Range("a2") = 0
End Sub
```

5. В окне экранной формы GraphWorX32 создайте два встроенных динамических объекта просмотра параметров процесса типа *Параметр* . Отметьте флажок **Ввод данных** на странице **Параметр** диалоговой панели **Инспектор свойств** каждого динамического объекта, а также установите диапазон изменения вводимых значений и, при необходимости начальное значение. В полях **Источник данных** введите имена локальных переменных ~~a1~~ и ~~a2~~ для каждого объекта.

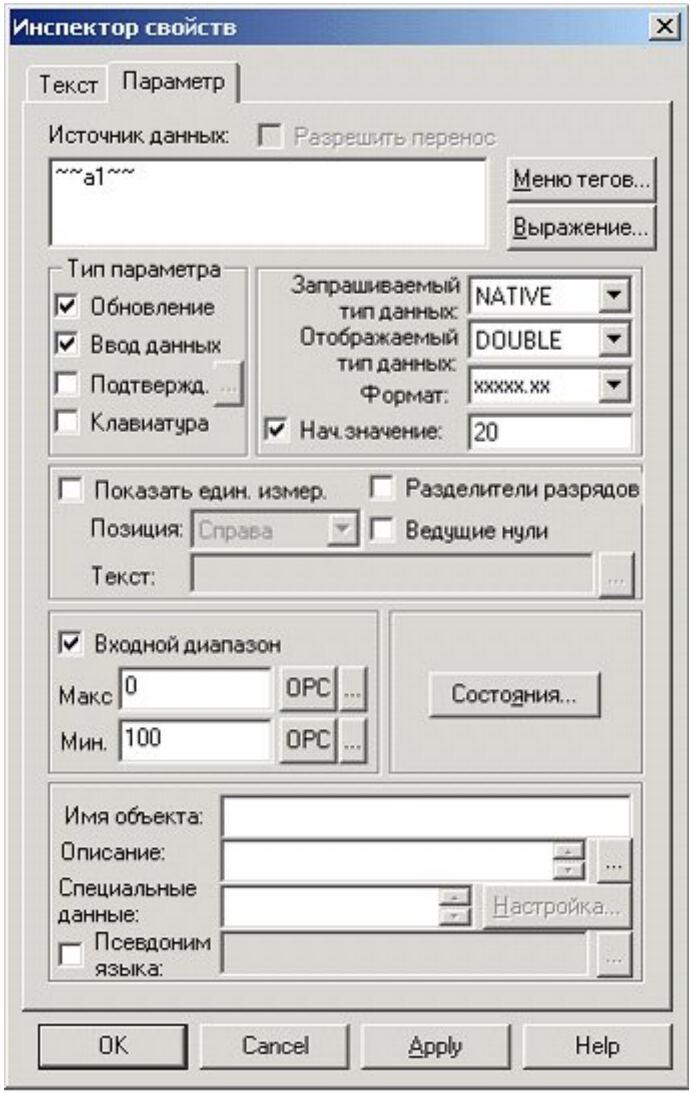


*Рис. 14-12. Страница **Параметр Инспектора свойств** для одного из полей ввода/отображения значений в экранной форме GraphWorX32*

6. Вернитесь в редактор VBA. Создайте процедуру чтения и записи информации между полями ввода/отображения значений в экранной форме GraphWorX32 и ячейками Excel. Выберите строку **Процедура** меню **Вставка** редактора VBA и введите название процедуры "*Read_Write*", после чего введите следующий исходный текст процедуры.

```
Public Sub Read_Write(ByVal Co As Integer)
 ' Данная процедура выполняет чтение или запись между
' ячейками Excel и полями экранной формы GraphWorX32
' по условию, передаваемому в качестве параметра
   Dim Point As GwxPoint
  Dim St As String  'Для хранения имени ячейки
  Dim St2 As String 'Для хранения имени переменной
  Dim X As Integer
  For X = 1 To 2
    ' сложение "a" к значению X с преобразованием в строку
    St = "a" + Mid(Str(X), 2, 1)
    St2 = "~~" + St + "~~"
    Set Point = ThisDisplay.GetPointObjectFromName(St2)
```

```
  If Co = 1 Then  'Записьв Excel
    ThisDisplay.g_Excel_Sheet.Range(St) = Point.Value
  Else            'Чтение из Excel
    Point.Value = ThisDisplay.g_Excel_Sheet.Range(St)
  End If
 Next X
End Sub
```

7. Вернитесь в окно экранной формы GraphWorX32 и создайте две радиокнопки. Назовите одну из них "*Чтение из Excel*", а вторую – "*Запись в Excel*". Настройте каждую кнопку следующим образом:
   – выберите в поле **Тип** страницы **Указание и щелчок Инспектора свойств** обеих кнопок значения *Зависимая*;
   – установите флажок **Пока нажато** в группе параметров **Порядок выполнения**. Снимите отметку с остальных флажков;
   – в выпадающем списке **Операция** выберите строку **Выполнить сценарий VBA**;
   – щелкните на кнопке **Создать** и в поле **Имя сценария** *Мастера VBA-сценариев* введите имя "*Wr*" для кнопки "*Запись в Excel*" и "*Rd*" – для кнопки "*Чтение из Excel*";
   – после нажатия кнопки **OK** *Мастера VBA-сценариев* в окно проекта Visual Basic будут добавлены модули *GwxRd_Main* и *GwxWr_Main*, как показано на рис. 14-11. В области окон программы будут открыты окна указанных модулей;
   – введите исходные тексты в тела процедур записи и чтения:

```
Sub Rd(o As GwxPick)
   Call ThisDisplay.Read_Write(0)
End Sub

Sub Wr(o As GwxPick)
Call ThisDisplay.Read_Write(1)
End Sub
```



Рис. 14-13. Добавленные модули записи и чтения между экранной формой и Excel

8. Вернитесь в окно экранной формы GraphWorX32 и переведите систему в режим *Исполнение*. Произойдет следующее:
   - автоматически запустится экземпляр Microsoft Excel. В ячейках A1 и A2 рабочего листа *Лист1* появятся нулевые значения;
   - при нажатии кнопки **Запись в Excel** экранной формы GraphWorX32 ввод значений с клавиатуры в поля ввода/отображения будет приводить к автоматическому обновлению содержимого ячеек A1 и A2 этими значениями;
   - при нажатии кнопки **Чтение из Excel** кнопка **Запись в Excel** перейдет в отпущенное состояние, а последующий ввод значений с клавиатуры в ячейки A1 и A2 рабочего листа *Лист1* Excel будет приводить к автоматическому обновлению полей ввода/отображения в экранной форме GraphWorX32 этими значениями.

14.4 Другие источники информации

Для получения более подробной информации о программировании с использованием модулей и форм в VBA, обратитесь к руководству программиста MicrosoftVisualBasic. Обратите внимание на то, что в GraphWorX32 используется тот же самый VBA, что и в пакете MicrosoftOffice. Таким образом, имеется возможность получения справочной информации о применении VBA из самых разнообразных источников, включая справочную систему и документацию по программированию на VBA, поставляемую в комплекте MicrosoftOffice.
После установки GENESIS32 пользователь сможет найти множество примеров программирования на VBA в каталоге:
\ICONICS\GENESIS-32\Examples\GraphWorX32 Examples