ScriptWorX32 for Fast Multi-threading VBA Applications

APPLICATIONS NOTE

August 2004

Description: Configure ScriptWorX32 to manage your scripts.
OS Requirement: The operating system required is Win 2003/XP/2000/NT/95/98/Me.
General Requirement:

The main function of ScriptWorX32 is to manage and maintain scripts that perform different functions. It takes advantage of a new Visual Basic 6.3 feature called multi-threading. The use of multi-threading enables operating systems to run more efficiently because it assigns different tasks to different threads so that the CPU usage will be split up amongst these threads. This is very useful if you have a need or desire to run different tasks, which do not rely on each other, at the same time. This documentation will give the user a quick start on the principles and configuration of ScriptWorX32. It consists of three different modules: Server, User Interface and the Visual Basic Editor.

ScriptWorX32 Server

The server which is not visible to the user runs as a service in NT (or as a standard application) executing the compiled code from the VBA Editor, at times and intervals configured in the User Interface.

ScriptWorX32 User Interface

The User Interface indicates what type of script is being created and how it will be executed. ScriptWorX32 supports the following types:

- **Global Scripts** are either manually launched from outside the application or upon entering runtime mode.
- **Periodic Scripts** are launched according to configured time settings.
- **Event Scripts** are based on changes, based on OPC data, expressions or local variables, using the standard Expression Editor.
- **Alarm Scripts** are based on the same principle as Event Scripts.

ScriptWorX32 VBA 6.3

The Visual Basic Editor is where the scripts will be created according to the configuration in the User Interface. All scripts are written in Visual Basic. Please refer to the Visual Basic documentation for more information.

Creating a script using the script wizard:

1. Open ScriptWorX32.
2. Select **Global Script** and right mouse click to insert a new script.
3. In the right window pane, specify the Trigger Name, **ExecuteBeep**, and Script Name, **BeepScript**. If desired, specify a VBA project name other than the default **Untitled**.
4. To have this configuration go into effect select the **Apply Changes** button.

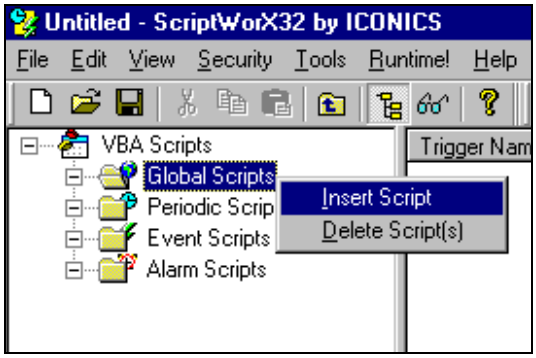

Fig. 1: Inserting a new script

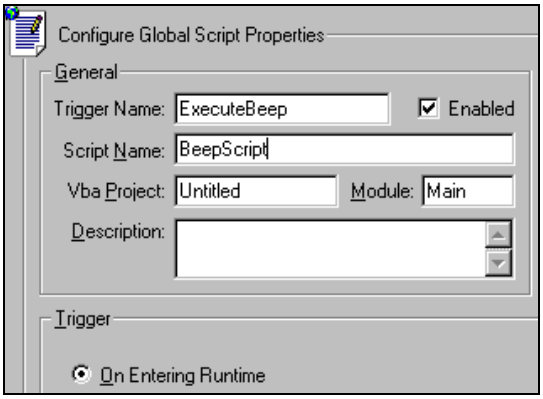

Fig. 2: Script Configuration Window

Now the script BeepScript is identified in ScriptWorX32, but the code still needs to be written. To create a script through the script wizard:

1. Select the **Script Wizard** Button.
2. For the Category select **General Scripts** and for the Scripts select **Beeper**.
3. Click on the **Generate Script** button.
4. Select "**Yes**" to enter the VBA editor. You will see the created script and can make any modifications if necessary.
5. Now switch back to the ScriptWorX32 Configurator and save the configuration to **Untitled.swx** using the floppy disk icon in the ScriptWorX32 menu bar.
6. To run the script you can start the ScriptWorX32 Engine by clicking on the traffic light in the ScriptWorX32 Configurator.
7. You will hear a short beep upon entering runtime, since the script is configured to trigger on entering runtime. To manually start the script, select the Trigger Name and click on the Start Instance Command.
8. If you need to edit this script, stop the ScriptWorX32 Engine with the traffic light, and click **Edit VBA**.