

Description: Guide to setting up GWX64 Display Caching.

OS Requirement: Windows Server 2003 x64/Vista x64/ Windows Server 2008 x64/Windows 7 x64/ Windows Server 2008 R2 x64

General Requirement: Installation of GENESIS64

Introduction

Before you publish and deploy GraphWorX64 displays, it is important that you pay attention to the cache settings that are in place for GraphWorX64 and the displays themselves. GraphWorX64 displays can become quite large and can take time to open in runtime. This may not be an issue for a display that acts as a console which, once it is open it stays open. However, it may become an issue for runtime users whose GraphWorX64 display opens other windows that take a long time to load.

Cache Settings

Although you can improve the performance by eliminating unnecessary details from models in a display, you can further enhance performance by using an effective strategy for caching displays. By default, GraphWorX64 displays are cached in accordance with the following:

- Four **Runtime Options** preferences that you set for GraphWorX64.
- The overriding **CachePriority** property setting for the display's ThisDisplay object.

Using these settings, you can specify whether displays are to be cached as they open or are preloaded into the cache so that they are resident in memory when a runtime user opens them. You can also specify when and whether they are removed from the cache.

Strategy

You don't want to simply enable caching for every display because you may end up consuming an unnecessary amount of memory. Instead, cache any displays that are used frequently or large displays that need to open quickly. If you know that a set of displays are typically used in tandem on individual client machines, consider their memory requirements in regard to each other. For example, if there are several large displays, caching them all may create more problems than it will solve. If there are several small displays, caching may not be necessary.

Runtime Overrides

Keep in mind that the ability to cache displays can be overridden during runtime as shown in Figure 1 and Figure 2.

NOTE: Figure 2 can be achieved by unchecking the Ribbon option in the Show/Hide group, then right clicking on the canvas and selecting "Show Main Menu".

The runtime user has access to and the ability to override the Cache Displays runtime option preference, and can enable and disable caching at will. When a runtime user enables caching, displays are cached in accordance with the caching settings. For that reason, it is important that you pay attention to the settings, even if you publish the displays with caching disabled.

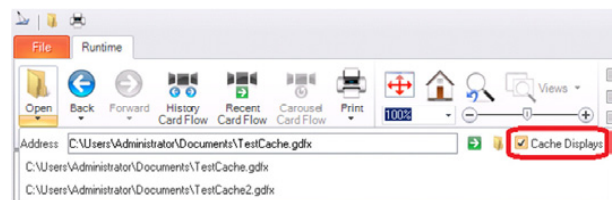


Figure 1 - Runtime Cache Displays Checkbox

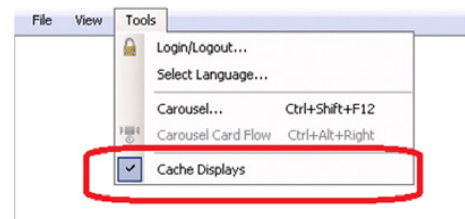


Figure 2 - Runtime Menu Option: Cache Displays

Table 1 shows a comparison of performance of different displays when caching is enabled versus caching disabled. We also compared the performance loading Silverlight displays.

NOTE: for more information on running GraphWorX64 in Silverlight, please refer to the application note *GENESIS64 – Silverlight Support*.

Table 1 - Performance Comparison

V10.7 – AMD Athlon 64 X2 Dual Core 4600+ 2.4 GHz, 3GB RAM						
GWX test display with 5000 unique process points	Non-cached		Cached		Silverlight	
	First Load	Reload	First Load	Reload	First Load	Reload
Load duration [s]	4.5	6.1	2.2	2.3	4.2	6.1
Load duration + Time to receive all good data [s]	11.3	13.5	9.1	9.7	15.3	13.7

“This Display” Preferences: Runtime Options

Caching Section:

- **CacheDisplays** - Set this preference to **True** to enable caching and make GraphWorX64 displays load faster; selecting **True** enables the three additional preferences described below. The default is **False**. If you enable caching, caching can be overridden for individual displays and at runtime. You can use the display's **CachePriority** property to opt out for individual GraphWorX64 displays or specify the conditions under which individual windows get removed from the cache if memory runs low. Runtime users can enable and disable caching during runtime, which is helpful if memory gets low.
- **CacheDisplayExpiration** - Specify the amount of time that each individual display will remain in memory since the last time the display was loaded. The default is **No Expiration**, and therefore displays will not be removed from memory. To override the default, specify a length of time in *hours:minutes:seconds* (0:00:00 to 9:59:59).
- **PreloadLinkedDisplays** - **True** preloads all displays that a main display references. For example, a display that is a Main Console has 20 buttons, each with a Pick dynamic Command property of Load Display that opens a separate GraphWorX64 display. Setting this preference to **True** would preload the additional 20 displays when the Main Console opens, and hold them in memory so that they don't have to open at the time a runtime user clicks a button. The default is **False**.
- **PreloadCarouselDisplays** - **True** preloads all displays in a Carousel playlist. If you set this preference to **True**, for each display you can create a playlist of the

windows that preload when the display is opened at runtime. (The files will preload, but the carousel will not play.) The default is **False**.

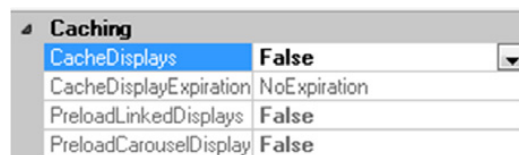


Figure 3 - Caching Section in Runtime Options

“This Display” Properties – CachePriority

CachePriority - Displays are automatically removed from the cache if the cache is full. Specify your preference for handling caching when memory gets low during runtime. Options are:

- **NotCached** - Displays with this setting are not added to the cache, regardless of the **CacheDisplays** preference setting. Select this option if you want to exclude most displays from being cached; then override this property for individual displays.
- **Low** - Displays with this setting are most likely to be removed from the cache when memory is low.
- **BelowNormal**
- **Normal** - The default value.
- **AboveNormal**
- **High** - Displays with this setting are least likely to be removed from the cache when memory is low.
- **NotRemovable** - Displays with this setting are not removed from the cache, even when memory is low.

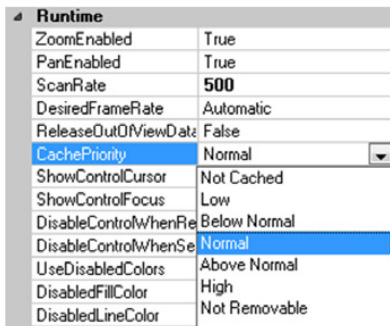


Figure 4 - Caching Section in Runtime Options