GLX Application Framework   2

GLX Application Framework

Introduction

The GLX Application Framework provides the underlying infrastructure you need for creating professional-grade applications on Mac and Windows. This allows you to focus on making your application unique and valuable instead of wasting time setting up mundane features such as undo support. Using the framework saves you time and makes you more productive.

The framework takes care of common tasks such as splash stacks, loading stacks and externals at launch, working with user preferences, auto updating, undo management and more.

The framework is free to use in your personal or commercial projects.

The framework does not currently have Linux support because we do not create applications for Linux. Someone is welcome to add support for this. Contact me if you are interested and we can discuss what would need to be done.

Thanks to Daniels & Mara for the use of the GLX name. Jerry has created tools that help Revolution developers be far more productive and hopefully this framework will help do the same. GLX2 is the must-have script editor for Revolution developers and is available at http://daniels-mara.com/glx2/.

If you have questions, suggestions or would like to contribute a modification to the framework please send an email to glxapp at bluemangolearning dot com.

Manual

The manual is available at http://revolution.screenstepslive.com/spaces/revolution.

About This Document

This document describes messages and properties available in the framework. Take a look at the on-line tutorials for information on setting up, configuring and building applications using the framework.

You can also look at the scripts used in the framework. See the stack scripts of the glxapplicationFramework stack and it's substacks as well as the glxappManagedEngineMessages button (used as a front script) on card 1 of the glxapplicationFramework stack.

Framework Messages

The GLX Application Framework sends the following custom messages.

Message	Description
glxapp_build *pProfile, pStandaloneOrUpdater*	When you build your application this message will be sent. The first parameter is the name of the profile being built. The second parameter is either "standalone" or "updater" depending on what type of build is being performed. You can show/hide controls based on the parameters, set certain properties, etc. All of your application stacks will be in memory but libraries and externals will not be in use.
glxapp_cleanupApplication	Sent right before the application quits. You can save any data or perform any other necessary cleanup.
glxapp_firstRun	Sent the first time your application is launched on a computer. This is determined based on the presence of a user preference file on the computer. The message is sent just before glxapp_initializeApplication and the application has been full loaded at this point.
glxapp_preloadApplication	Sent after the framework stack has been loaded into memory but before any of your applications stacks have been loaded into memory. This is a good place to load the revDataGridLibrary.rev stack if you are manually including it with your application.
glxapp_initializeApplication	Sent when all application stacks have been loaded, libraries have been put in use and externals have been loaded. Perform any initialization routines here such as setting fonts for stacks, set the navigationArrows, etc. If you return "false" from this command then the framework will quit the application. Returning false is an appropriate way to quit if something goes wrong during initialization.

Message	Description
glxapp_openApplication	Sent after glxapp_initializeApplication. This is where you would typically open your main application window. glxapp_initializeApplication and glxapp_openApplication are separated into two separate messages strictly for organizational purposes
glxapp_openFiles *pFiles*	Sent when the operating system is requesting that your application open a file. The pFiles parameter is passed in. It is a line delimited list of file names that have file extensions/ creator codes that you have setup for your application.
glxapp_relaunch	Sent when the user has attempted to relaunch your application but your application only allows a single instance. This is a good place to bring your main application out of iconic mode if need by. Windows only.

Stack and Card Messages

The GLX Application Framework sends the following custom messages to stacks and cards of stacks that have the uGLXApp["manage window"] property set to true.

Message	Description
PreOpenWindow *pHasBeenOpened*	Same behavior as preOpenStack but with one parameter. pHasBeenOpened is true if the stack has been opened once during the current session, false otherwise.
OpenWindow *pHasBeenOpened*	Same behavior as openStack but with one parameter. pHasBeenOpened is true if the stack has been opened once during the current session, false otherwise.
PreOpenView *pHasBeenOpened*	Same behavior as preOpenCard but with one parameter. pHasBeenOpened is true if the stack has been opened once during the current session, false otherwise.

Message	Description
OpenView *pHasBeenOpened*	Same behavior as openCard but with one parameter. pHasBeenOpened is true if the stack has been opened once during the current session, false otherwise.
ResizeView *pWidth, pHeight, pOldWidth, pOldHeight*	Similar to resizeStack but sent more often: * When window first opens. * When navigating to a different card in a stack. If you do all of your resizing in this handler then the controls on your cards will always appear correctly.

The GLX Application Framework sends the following custom messages to cards of stacks that are listed in the uGLXApp["palettes"] of a stack. The the uGLXApp["palettes"] property of a stack is a list of stack names that are palettes of the stack, one per line.

Message	Description
CloseParentStack	Send to the card of the palette stack when the parent stack closes.
IconifyParentStack	Send to the card of the palette stack when the parent stack is iconified.
OpenParentStack	Send to the card of the palette stack when the parent stack opens.
UnIconifyParentStack	Send to the card of the palette stack when the parent stack is uniconified.

Read-Only Properties

These properties can be used with:

function `glxapp_getProp(pProperty)`

Property	Description
32x32 app icon id	The id of the 32x32 pixel application icon.

Property	Description
64x64 app icon id	The id of the 64x64 pixel application icon.
application folder	The path of the folder containing the application executable (.exe or .app). Same as *root folder* property.
application has loaded	Returns true if the application has finished loading, false otherwise. This value is set to true after the glxapp_openApplication message is sent.
application stack file path	The full path to the application.dat stack file.
application update available	Returns true if an update for the application is available on the web. Only valid after calling *glxapp_checkForUpdate*.
application update version	The version of the application on the update server. Only valid after calling glxapp_check-ForUpdate.
build profile	The profile used to build the application. Returns development if you are in the development environment or if no build profile was used to build the application.
build profiles	Control references of all build profile objects. One per line.
command line files	List of application supported files that were passed to your application when it was first launched. One per line.

Check this property to determine if you should open a specific document when your application first opens. |
| engine update available | Returns true if an update for the engine your application uses is available on the web. Only valid after calling *glxapp_checkForUpdate*. |
| executable folder | The full path to the folder the executable is in. On OS X this will be inside of the application bundle. |

Last Modified: July 22, 2009

Property	Description
externals	Control references of all application external objects. One per line.
file extensions	Control references of all application file extension objects. One per line.
file type filters	Control references of all application file type filter objects. One per line.
framework stack file path	The full path to the glxapp_framework.dat stack file.
long version	The major, minor, revision, first character of each word of the stage and build number. Examples: 1.0.4r3 1.2.4rc5 2.5.2b1
macos externals	Control references of all application external objects for OS X. One per line.
macos required external packages	Control references of all application external objects for OS X that are required for application to launch. One per line.
macos standalone name	Name of the application standalone on OS X.
macos standalone path	Full path to the application standalone on OS X.
preference broadcaster	Reference to the control used to store preference broadcasting settings.
resources	Control references to application resource controls. One per line. Currently this is limited to the 32x32 and 64x64 image controls.
root folder	The path of the folder containing the application executable (.exe or .app). Same as *application folder* property.

Property	Description
shared data folder	The path to the application shared data folder on the computer the application is running on. This will be empty if *use shared application data* is false.
shared preference file	Returns the full path to the file used to store shared preferences. The value is empty if the preference file name for the platform has not been set (i.e. *macos preference file* or *win32 preference file*). This file is stored in the "shared data folder" so make sure you have "use shared application data" set to true.
short version	The major, minor and revision numbers. Examples: 1.0.0 2.1.2
stacks	Control references of all application stack objects. One per line.
standalone name without extension	The name of the standalone executable without the .exe or .app extension.
standalone path	The full path to the application standalone executable. This property is based off of *standalone name without extension.*
standalone resources	Control references of all application standalone resource objects. One per line.
update available	Returns true if an update for the application or engine is available on the web. Only valid after calling *glxapp_checkForUpdate*.

Property	Description
use shared preference file	Set to true if your application uses a shared preference file as well as a user preference file. This can be handy for storing system wide preferences for your application. This file is stored in the "shared data folder" so make sure you have "use shared application data" set to true. If using a shared preference file then pass "shared" as the last parameter when calling glxapp_setPref, glxapp_getPref, goxapp_prefIsSet and glxapp_savePref.
user preference file	Returns the full path to the file used to store user preferences. The value is empty if the preference file name for the platform has not been set (i.e. *macos preference file* or *win32 preference file*).
user data folder	The path to the application user data folder on the computer the application is running on. This will be empty if *use user application data* is false.
version	If the stage is *Release* then the major, minor and revision are returned. Example: 1.0.2 2.1.4 If the stage is anything other than release then the major, minor, revision, first character of each word of the stage and build number are returned. Example: 1.0.2 b2 2.1.1 rc3
version array	Returns an array representing the version. The keys are *major, minor, revision, stage* and *build.*

Last Modified: July 22, 2009

Property	Description
win32 externals	Control references of all application external objects for Windows. One per line.
win32 preference file	The name of the file to store preferences in on Windows.
win32 required external packages	Control references of all application external objects for Windows that are required for application to launch. One per line.
win32 standalone name	Name of the application standalone on OS X.
win32 standalone path	Full path to the application standalone on OS X.
working standalone name	The actual name of the standalone running the application. This could be the Revolution IDE engine.
working standalone path	The full path to the folder of the standalone running the application. This could be the folder containing the Revolution IDE engine.

Read/Write Properties

These properties can be used with following handlers:

```
function glxapp_getProp(pProperty)
command glxapp_setProp pProperty, pValue
```

Property	Description
application data folder	The relative folder where your application will store files. The path is relative to the user/ shard application data folders.
application name	The name of your application.
auto update base url	The base url where auto updates will be stored. This base url is combined with the build profile name when looking for updates for an application.
auto update password	The password used to authenticate with the update server (Optional).

Property	Description
auto update username	The username used to authenticate with the update server (Optional).
email address for errors	The email address that users should send error reports to when non-fatal (script) errors are encountered while using your application.
engine update available	Returns true if an update for the engine your application uses is available on the web. Only valid after calling *glxapp_checkForUpdate*.
engine version	The version of the engine that your application is distributed with. This combines the version (everything up to first "-") and the buildNumber to give a value such as 2.9.0.580.
macos preference file	The name of the file to store preferences in on OS X.
shutdown when all windows are closed	Set to true if you want the application to behave the same on Mac as it does on Windows. That is that the application will quit when all stacks of mode 1 or 2 are closed AND stacks of mode 3 are closed or invisible.
splash screen duration	The minimum amount of time (in milliseconds) to display the splash screen. The actual time the screen is displayed may be longer depending on how long it takes to launch your application.
splash stack file name	The relative path to the stack file to display as a splash screen when application launches. The path is relative to the root application folder. Leave empty to not display a splash screen.
stacks password	The password to encrypt stacks with when building an application. This property will return empty in the runtime environment.

Property	Description
stacks to unlock in standalone	A return delimited list of stack names that will be unlocked using the "stacks password" property when your application launches. This is only applicable after you package an application and the "stacks password" has been applied to your stacks. This property is useful if you plan on copying controls from or pasting controls onto stacks at runtime as you can't do either when a stack is password protected unless you have set the "passkey" property of the stack.
standalone name	The name of the standalone executable for current the platform. This property is configured and does not necessarily represent the name of the executable currently running the application.
use shared application data	Set to true if your application will store files in the shared data folder on the computer. The *shared data folder* property only returns a value if this is set to true.
use user application data	Set to true if your application will store files in the user data folder on the computer. The *user data folder* property only returns a value if this is set to true.
win32 preference file	The name of the file to store preferences in on Windows.