iOS Embedded Deployment

Table of Contents

Overview

Traditionally, LiveCode provides you a complete environment for developing an application. Your application is written entirely within LiveCode using the LiveCode language and feature-set provided by the engine, and although its possible to extend the functionality through the use of externals, it is still the LiveCode environment that has ultimate control.

The embedded version of LiveCode turns this situation on its head. Instead of writing your complete application in LiveCode (with optional native code extensions through externals), you can compile a LiveCode application in a way that allows it to be included in a native Xcode project as a component – meaning you can choose to only implement certain parts of your whole application in LiveCode, or add functionality written in LiveCode to an existing native application.

An embedded component has access to almost the entire feature set that the LiveCode engine has to offer – the only exceptions being those features that access the 'application' level of a native app since it is up to the native code host for the component to deal with those (for example, push notifications, status bar manipulation etc.)

A stack built as an embedded component using the *LiveCode Embedded Builder Plugin* produces a static library for linking into an Xcode project which enables the native code to create a *LiveCodeView* which can then be placed anywhere a normal UIView can. The native code host can then post events to the LiveCode component, and register to respond to messages that pass through the LiveCode message path.

Note: *The embedded platform requires at least 5.5.2-gm-2 for Mac to be used – previous versions don't have the necessary IDE support for the builder plugin.*

Getting Started

The current release of the embedded platform consists of three things:

- The *LiveCode Embedded Builder Plugin* – this builds a stack as an embedded component (static library) for use in Xcode.
- An example *EmbeddedView* project – this is a simple Xcode 4 project demonstrating how to use an embedded LiveCode component.
- The release notes – the document you are currently reading.

To get started, place the LiveCodeEmbeddedPlugin.rev file in your LiveCode user extensions Plugins folder (this is usually ~/Documents/My LiveCode/Plugins – but can be configured in Preferences). When you restart the IDE you will be able to select the builder plugin from the Plugins menu in the normal way.

Note: *In order to use the embedded plugin, you will need to relicense LiveCode after you've purchased the product in the store so that the IDE knows you are licensed to use the platform (look for 'iOS Embedded' in the license activation screen to make sure its present).*

The plugin itself is a simple interface for specifying basic aspects of the component you want to build (there's a lot less to configure for embedded components compared to standalone applications since the native code host is responsible for most iOS application configuration aspects):

The plugin allows you to configure:

- *stack* – this is the mainstack that comprises the component. This drop down will list all current mainstacks in memory that have been saved to disk.
- *externals* – this allows you to select which externals to include. In this release only the standard LiveCode externals are supported, but third-party externals will be supported in a subsequent release.
- *sdk* – this allows you to specify which iOS SDK to build against. In this release only 5.0 or 5.1 are supported. The SDK you use in Xcode must match the choice you make here. Note that device builds (whether you choose 5.0 or 5.1) will work on any iOS version from 4.0 onwards assuming the Xcode project is set to support those (check the minimum iOS setting

there).

- *output* – the filename of the static library to produce. A good place to choose is next to the stack being built. (The filename is recorded relative to the stack if its in the same folder or one of its descendants, otherwise its recorded as an absolute path).

Once a stack has been configured in this way, the *Build* button is used to create the static library which can then be used in the Xcode project.

Trying it out

This release comes with a simple example Xcode project *EmbeddedView*.

The first thing to do is to build the component's static library:

1. First of all make sure the IDE is loaded and has been relicensed after purchase to ensure the ios-embedded deployment license is present.
2. Open up the builder plugin from the Plugins menu.
3. Open the *EmbeddedViewStack.livecode* stack from the EmbeddedView project at which point you should be able to choose it from the stack drop-down in the plugin:

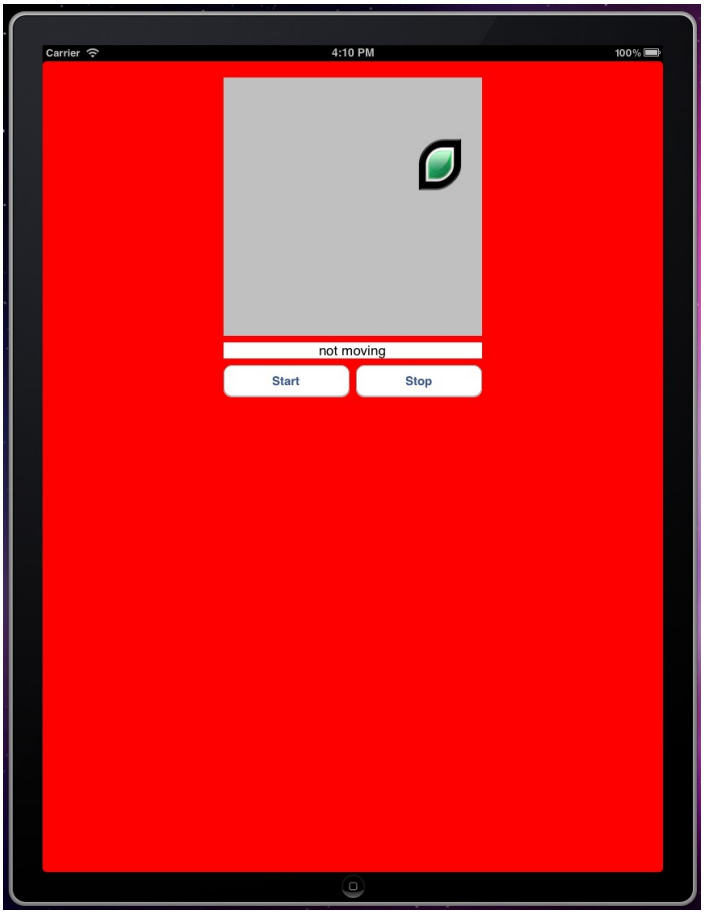
   As you can see, this stack is (initially) configured to use the 5.0 SDK and output a static library called EmbeddedView.a – it will be created in the folder next to the stack.
4. Choose an appropriate SDK (whether you can use 5.0 or 5.1 will depend on the iOS SDKs you have installed and which developer roots are configure in the IDE – either will work fine).
5. Click 'Build' and after a few moments the static library should appear. Note that the library is quite big as its a universal build for both devices and the simulator – you can choose to build for either in Xcode using the same library.

With the component's static library built, the next step is to load up the Xcode project:

1. Load up the EmbeddedView Xcode project into Xcode 4.
2. Choose to target the iOS iPad Simulator – making sure you select the version that matches

the SDK version that was chosen in the plugin.

3. Click 'Run' and the application should leap into life into the simulator:



This simple application consists of the LiveCodeView containing the stack we used in the plugin along with a number of native iOS elements created by the native code side of the project. Clicking Start will post a message to the component to tell it to start moving the logo, whilst clicking Stop will post a message to the component telling it to stop moving the logo. Additionally, the component dispatches a message *imagePositionChanged* which the native code project connects to and updates the label in response.

Using an existing Xcode project

To add a LiveCode embedded component to an existing Xcode project is similarly straightforward, needing you to do three things.

First of all, copy the LiveCodeView.h header file into the source folder for the project – this contains the LiveCodeView interface definition and external reference to *LiveCodeGetView()*. The LiveCodeGetView() function returns the UIView (at runtime) that you can then place wherever you wish.

Next build the static library for your component using the builder plugin and drag the resulting file to your project so that Xcode knows to link with it.

Finally, make sure the list of Frameworks and libraries your application links with includes all those

listed in the 'deps' file that is created alongside the static library. (The easiest way to add these dependencies is to select them all in the example project, copy, and then paste them into your project).

With these steps done, the existing Xcode project should build and link with your LiveCode component – the only thing left for you to do being to write the code that uses it!

For more information about how to use the LiveCode view, take a look at the example project and the LiveCodeView.h header file, but at its simplest you need do only three things:

1. Create the LiveCodeView by calling LiveCodeGetView() when your application starts up and place it somewhere in your UIView hierarchy.
2. Configure the location and bounds of the view by using the *setViewBounds:* method (at the moment you can't use *setFrame:* or *setBounds:* to do this so you must do it manually).
3. Use the *startup* method to initialise the engine and open the component's stack into the view and the *shutdown* method on termination to finalise the engine and clean up its resources.

Limitations

This release of the iOS Embedded deployment option has a number of limitations all of which will be addressed in future releases:

1. An embedded component will only create a single LiveCodeView.
2. You can only embed one LiveCode component per project.
3. To configure the size and position of the LiveCodeView you must use the *setViewBounds:* method (*setFrame:* and *setBounds:* will not currently work, and should not be sent to the LiveCodeView).
4. Third-party externals are not currently supported.
5. You can only *post* (i.e. *send in 0 millisecs*) messages to the LiveCodeView. (In a subsequent version you will be able to *send* messages and therefore get a result back immediately).
6. Only iOS 5.0 or 5.1 simulators are supported, and you need to use either 5.0 or 5.1 SDK for device builds (although such builds will run on 4.0 onwards, assuming the appropriate minimum deployment setting is configured in Xcode).

Change Logs and History

Embedded Engine Change History

R1 (2012-09-13) MW Initial version.

Embedded Builder Change History

R1 (2012-09-13) MW Initial version.

Document History

Revision 1 (2012-09-13) MW Initial version.