Using the Binary Distribution

This page provides important information for users of the Ice Touch binary distribution. You can obtain this distribution at the ZeroC web site.

On this page:

Overview of the binary distribution

The binary distribution of Ice Touch includes the following components:

- Command line Objective-C SDK for OS X
- Xcode Objective-C SDKs for OS X, iPhone iOS, iPhone Simulator
- Xcode C++ SDKs for OS X, iPhone iOS, iPhone Simulator
- Xcode Plug-in for use with Ice Touch

The binary distribution was compiled on OS X 10.9 using Xcode 5.1. The binaries in this distribution are fat binaries with support for both Intel 32-bit and Intel 64-bit architectures on OS X and support for ARMv7, ARMv7s and ARM64 on iOS.

The binaries are installed in the `/Library/Developer/IceTouch-1.3.3` directory.

Back to Top ^

Setting up your OS X environment to use Ice Touch

Using Xcode SDKs

For Cocoa and iPhone applications, you must add the appropriate directory to the *Additional SDKs* setting in your Xcode project:

Objective-C SDK	`/Library/Developer/IceTouch-1.3/SDKs/ObjC/$(PLATFORM_NAME).sdk`
C++ SDK	`/Library/Developer/IceTouch-1.3/SDKs/Cpp/$(PLATFORM_NAME).sdk`

You also need to update the project setting "Header Search Paths" to include the Ice Touch SDK include directory.

Objective-C SDK	`/Library/Developer/IceTouch-1.3/SDKs/ObjC/$(PLATFORM_NAME).sdk/usr/local/include`
C++ SDK	`/Library/Developer/IceTouch-1.3/SDKs/Cpp/$(PLATFORM_NAME).sdk/usr/local/include`

In addition, when creating a new iPhone Xcode project, you must set the Code Signing Resource Rules Path to:

```
$(SDKROOT)/ResourceRules.plist
```

You must also add the following to the `Frameworks` folder:

```
CFNetwork.framework
Security.framework
Foundation.framework
```

When using the Objective-C SDK you must also add the following to the `Frameworks` folder:

```
ExternalAccessory.framework
```

See the Xcode Plug-in documentation for additional information on the plug-in build options.

Back to Top ^

Using the Ice Touch command-line SDK

In order to use the `slice2objc` Slice translators included with the Ice Touch distribution, you need to add the location of the Ice Touch binaries to your `PATH` as shown in the bash command below:

```
$ export PATH=/Library/Developer/IceTouch-1.3/bin:$PATH
```

The Ice Touch binary distribution includes two sets of Objective-C libraries built with two different C++ run times. These libraries are installed in in `<Ice Touch installation directory>/lib`. The libraries with the `-libc++` suffix use LLVM `libc++` (e.g., `libIceObjC-libc++.dylib`), while the libraries with no suffix use `libstdc++` (`libIceObjC.dylib`).

When compiling Ice for C++ programs, you must pass the `-pthread` option and a `-I` option specifying the Ice Touch include directory. A typical compile command would look like this:

```
$ c++ -I /Library/Developer/IceTouch-1.3/include -c -pthread myprogram.cpp
```

When linking a program you must pass the Ice Touch library directory with the `-L` option and set the program run path using the `-rpath` linker option. Furthermore, an Objective-C program needs to link with at least `libIceObjC` or `libIceObjC-libc++`. A typical link command would look like this:

```
$ c++ -o myprogram myprogram.o -Wl,-rpath,/Library/Developer/IceTouch-1.3/lib -L/Library/Developer/IceTouch-1.3
/lib -lIceObjC -framework Foundation
```

Additional libraries are necessary if you are using an Ice service such as IceGrid or Glacier2.

To build fat binaries or binaries using an architecture that differs from the default architecture, you can specify the Clang `-arch` compiler flag. For example, use `-arch i386 -arch x86_64` to build Intel 32-bit and 64-bit fat binaries.

As shown in the previous example, we need to set the application run path when we link the application. This is required because Ice libraries use `@rpath` as the prefix in their install names, which allows the application developer to relocate the Ice libraries without needing to rebuild them. You should use the `/Library/Developer/IceTouch-1.3` symbolic link for the run path if you want your executables to automatically use new Ice patch releases. This symbolic link is updated by the Ice installer to point to the latest installed Ice version.

If you want to include Ice Touch in your application bundle, you will need to copy the necessary IceTouch libraries to the `Contents/Frameworks` subdirectory of your bundle and use `@loader_path/../Frameworks` as the run path when linking the application.

Please refer to the `dyld` man page on your OS X system to learn more about `@rpath` and `@loader_path`.

For C++11 builds you must use libraries with the `-libc++` suffix.

Back to Top ^

Using the sample programs on OS X

Sample programs are provided in a separate archive, which can be downloaded from the ZeroC web site.

Back to Top ^