Release Notes

Ice Touch offers C++ and Objective-C SDKs for building iOS and Cocoa applications. Ice Touch also includes an Objective-C run time for use in OS X applications.

The Ice Touch distribution does not include any Ice services, but its support for the complete Ice protocol means that your Ice Touch applications can work seamlessly with existing Ice servers as well as Ice services such as IceGrid, Glacier2, and IceStorm.

On this page:

Back to Top ^

New features in Ice Touch 1.3

This section outlines changes and improvements in this release that may affect the operation of your applications or have an impact on your source code.

For a detailed list of the changes in this release, please refer to the CHANGES file included in your Ice Touch distribution.

Support for Ice 3.5 features

Ice Touch is based on the Ice 3.5.1 release. For more information see the Ice 3.5.1 Release Notes.

Changes and fixes in Ice Touch 1.3.3

- Ice Touch now requires Xcode 6
- Updated Ice Touch to support Xcode 6.0
- Updated Ice Touch to support iOS 8.0
- Fixed a bug where servers wouldn't be able to accept new connections after the application was put in the background.

Back to Top ^

Corresponding Ice release

The Slice definitions included in Ice Touch 1.3 are the same as the Slice definitions included in Ice 3.5.1. In particular, the Glacier2, IceGrid, and IceStorm client libraries included in this Ice Touch release use the Ice 3.5.1 definitions. If a future Ice release adds new APIs (such as a new operation, or a new interface) to one of these services, you will need to rebuild these libraries using the newer Slice definitions in order to use the new APIs.

Back to Top ^

Upgrading your application from previous Ice Touch versions

ZeroC does not guarantee binary compatibility between Ice Touch 1.3 and previous Ice Touch versions, therefore you must recompile your Slice files and rebuild your application.

Xcode project settings

For Xcode iOS and Cocoa applications, you need to update the project setting "Additional SDKs" to match the location of the new Ice Touch SDK installation. You should use:

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

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

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

If you're not using the Cocoa or iOS SDKs but rather the command-line SDK, you will need to update the "Ice Home" setting.

Command line SDK settings

The command line SDK is installed in `/Library/Developer/IceTouch-1.3.` The install name of the Ice Touch shared libraries is prefixed with `@rpath/`. You will need to change your build system accordingly. Specifically, you will need to link your executable with `-Wl,-rpath,/Library/Developer/IceTouch-1.3/lib` if you want the Ice Touch shared libraries to be located without having `DYLD_LIBRARY_PATH` set.

Back to Top ^

Ice Touch feature set

Ice Touch supports the following features:

- Objective-C mapping
- C++ mapping

Ice Touch currently lacks support for the following Ice features:

- Protocol plug-ins
- Local interfaces
- `Ice::Application` and `Ice::Service` helper classes

The Objective-C mapping currently lacks support for the following Ice features:

- Asynchronous method dispatch (AMD)
- Collocation optimization
- Servant locators

Ice Touch has limited support for:

- SSL
  See below for more information.

- UDP
  On iPhone, UDP requests do not transparently establish a 3G/Edge connection.

Back to Top ^

SSL support

Ice Touch for OS X and Cocoa uses the Ice for C++ SSL protocol plug-in.

For iOS devices, Ice Touch SSL provides only a subset of this functionality. Due to limitations in iOS SSL support, the following restrictions apply:

- Ice Touch servers cannot authenticate SSL clients.

- Clients on iOS 5 and later reject server certificates that use MD5 hashes.

Furthermore, the semantics of some IceSSL configuration properties have changed, and new properties have been added. The IceSSL property reference provides complete details.

Back to Top ^

Logger notes

Objective-C applications must install custom loggers via `ICEInitializationData`. You cannot use any of the Ice for C++ logger properties, such as `Ice.UseSyslog`, and you cannot install a custom logger with a plug-in (`Ice.Plugin.*`).

Back to Top ^

Memory management

Graphs of objects with cycles

With the introduction of automatic reference counting (ARC), you no longer need to explicitly retain and release parameters or class members. However, you are still required to explicitly break cycles, otherwise graphs of objects containing cycles will leak. Consider this Slice definition:

Slice

```
class Foo
{
    Foo next;
};
```

Suppose we use these definitions as follows:

Objective-C

```
Foo a = [[Foo alloc] init];
Foo b = [[Foo alloc] init];
a.next = b;
b.next = a;
```

If you send this graph over the wire, your application will leak memory unless you somehow retain the graph and manually break the cycle.

Back to Top ^

Auto release pool

The Ice run time creates an `NSAutoReleasePool` object before dispatching server-side invocations and client-side AMI callbacks. The pool is released once the dispatch is complete.

Back to Top ^

Support for background applications

IceTouch supports VoIP applications as documented in Executing Code in the Background.

Follow these steps to ensure your application is correctly configured:

- Add the `voip` flag to the `UIBackgroundModes` key of your application's `Info.plist`.
- Set the configuration property `Ice.Voip` to `1` in your communicator configuration properties. As described in the Configuring Sockets for VoIP Usage, this causes the Ice run time to set the `kCFStreamNetworkServiceType` property to `kCFStreamNetworkServiceTypeVoIP` for all sockets.
- Before moving your application to the background, call the `setKeepAliveTimeout:handler:` method to specify the frequency at which your application must be woken to maintain your service.

The sample application `demo/iPhone/voip` provides a demonstration of these steps.

Back to Top ^

Targeting iOS >= 5.1.1

Xcode 6 targets iOS 8 by default. To build your application for an iOS >= 5.1.1 target, you will need to modify your Xcode project as follows:

- Set the Xcode `Deployment Target` property to the desired iOS version.

Back to Top ^