Using the macOS Binary Distribution

This page provides important information for users of the Ice binary distribution for macOS.

On this page:

Overview of the macOS Binary Distribution

Ice for C++, Java, Objective-C, and PHP on macOS are provided through the Homebrew formulas.

This `ice` formula includes the following components by default:

- Run-time libraries for C++ and Objective-C (macOS)
- Executables for IceGrid, IceStorm, Glacier2, IceBridge, and IcePatch2 services (macOS)
- Tools and libraries for developing Ice applications (macOS)
- Xcode SDKs for C++ and Objective-C (macOS, iOS and iPhone Simulator)

The `php56-ice`, `php70-ice`, and `php71-ice` formulas include:

- Run-time libraries for their respective version of PHP (macOS)

Back to Top ^

Homebrew Taps

The `ice` formula is available in two taps:

- `homebrew/core`, the default homebrew tap
- `zeroc-ice/tap`, ZeroC's homebrew tap

The `php56-ice`, `php70-ice` and `php71-ice` formulas are available in these taps:

- `homebrew/php`, the default homebrew tap for PHP formulas
- `zeroc-ice/tap`, ZeroC's homebrew tap

These formulas are identical in all these taps, and you can install `ice` and related formulas from homebrew or zeroc-ice taps. One exception is the `ice-builder-xcode` formula which is currently available only from `zeroc-ice/tap`.

When a new version of Ice is released, `zeroc-ice/tap` provides the new release immediately. It can take several days (and occasionally more) for the `homebrew` taps to be updated.

Back to Top ^

Installing the macOS binary distribution

Using Homebrew, you can install the distribution with this command:

```
brew install zeroc-ice/tap/ice [--with-java] [--with-additional-compilers] [--without-xcode-sdk]
```

The `--with-java` option builds the Java components and the IceGrid Admin app. You can also install IceGrid GUI on its own by downloading IceGrid GUI.dmg.

The `--with-additional-compilers` option installs `slice2py`, `slice2js`, and `slice2rb`.

The `--without-xcode-sdk` option skips the Xcode SDKs.

If you want to install a binary bottle, as opposed to building everything from sources, do not specify any build option when installing `ice`.

Separate formulas are available for PHP 5.6, 7.0, and 7.1:

```
brew install zeroc-ice/tap/php56-ice
brew install zeroc-ice/tap/php70-ice
brew install zeroc-ice/tap/php71-ice
```

Back to Top ^

Setting up your macOS environment to use Ice

After installing Ice, read the relevant language-specific sections below to learn how to configure your environment and start programming with Ice.

C++

Compiling and Linking

When compiling Ice for C++ programs, you must pass the `-pthread` option. A typical compile command would look like this:

C++11

```
c++ -c -DICE_CPP11_MAPPING -pthread myprogram.cpp
```

C++98

```
c++ -c -pthread myprogram.cpp
```

`C++11` and `C++98` in the tabs above correspond to the Ice C++ mapping you're using.

When linking a program you must link with at least the Ice library. A typical link command would look like this:

C++11

```
c++ -o myprogram myprogram.o -lIce++11
```

C++98

```
c++ -o myprogram myprogram.o -lIce
```

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

If you want to include Ice in your application bundle, you will need to copy the necessary Ice 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 macOS system to learn more about `@loader_path`.

Back to Top ^

Objective-C

Compiling and Linking

When compiling Ice for Objective-C programs, you must pass the `-pthread` option. A typical compile command would look like this:

```
cc -c -pthread myprogram.m
```

When linking a program you must link with `libIceObjC`. A typical link command would look like this:

```
cc -o myprogram myprogram.o -lIceObjC -framework Foundation
```

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

If you want to include Ice in your application bundle, you will need to copy the necessary Ice 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 macOS system to learn more about `@loader_path`.

Back to Top ^

Using Xcode SDKs

In order to use one of the Ice Xcode SDKs with your Xcode project, you need to:

- add `/usr/local/opt/ice/sdk/$(PLATFORM_NAME).sdk` to your *Additional SDKs*
- link your application with at least
  - `C++11: -lIce++11 -liconv -lbzip2`
  - `C++98: -lIce -liconv -lbzip2`
  - `Objective-C: -ObjC -lIce -lIceObjC -lc++ -liconv -lbzip2`
- On `iphoneos` and `iphonesimulator`, link your application with `CFNetwork.framework` and `UIKit.framework`

The following optional libraries are included in the Ice Xcode SDKs:

Name	Link your C++11 application with	Link your C++98 application with	Link your Objective-C application with	Additional dependencies
Glacier2 client library	`-lGlacier2++11`	`-lGlacier2`	`-lGlacier2ObjC`	
IceDiscovery plug-in	`-lIceDiscovery++11`	`-lIceDiscovery`	`-lIceDiscovery`	
IceGrid client library	`-lIceGrid++11`	`-lIceGrid`	`-lIceGridObjC`	
IceIAP plug-in (iphoneos and iphonesimulator only)	`-lIceIAP++11`	`-lIceIAP`	`-lIceIAP -lIceIAPObjC`	`ExternalAccessory.framework`
IceLocatorDiscovery plug-in	`-lIceLocatorDiscovery++11`	`-lIceLocatorDiscovery`	`-lIceLocatorDiscovery`	
IceSSL plug-in	`-lIceSSL++11`	`-lIceSSL`	`-lIceSSL -lIceSSLObjC`	`Security.framework`
IceStorm client library	`-lIceStorm++11`	`-lIceStorm`	`-lIceStormObjC`	

The Ice Xcode SDKs include only static libraries.

We also recommend installing the Ice Builder for Xcode, which helps you compile Slice files to C++ and Objective-C within Xcode . The builder is not included in the `ice` formula and must be installed separately with:

```
brew install zeroc-ice/tap/ice-builder-xcode
```

Back to Top ^

PHP

The Ice extension for PHP is loaded automatically when the interpreter loads the contents of the file `/usr/local/etc/php/{{version}}/php.d/ext-ice.ini` (generated on install). This file contains the following:

```
[ice]
extension="/usr/local/opt/php{{version}}-ice/ice.so"
include_path="/usr/local/opt/php{{version}}-ice"
```

At run time, the PHP interpreter requires the Ice shared libraries.

You can verify that the Ice extension is installed properly by examining the output of the `php -m` command, or by calling the `phpinfo()` function from a script.

Back to Top ^

Using the Sample Programs on macOS

Sample programs for all programming languages are available in a separate GitHub repository. Simply clone this repository:

```
git clone -b 3.7 https://github.com/zeroc-ice/ice-demos.git
cd ice-demos
```

Back to Top ^

Starting IceGrid GUI on macOS

You can launch IceGrid GUI with the `IceGridGUI` application installed in your `/Applications` directory. IceGrid GUI is a Java 8-based application.

Back to Top ^