

Guidelines for Distributing Ice Applications

This page provides some guidance for developers that are planning to distribute an Ice-based application. We can start by listing items that typically should not be included in your binary distribution:

- Slice compilers
- Slice files (unless you are using a scripting language, as discussed below)
- Executables and libraries for Ice services and tools that your application does not use
- For C++ programs on Windows:
 - DLLs built in debug mode (such as `ice35d.dll`)
 - Program database (PDB) files
 - Header files
 - Import library (LIB) files

Each of the language mappings is discussed in its own subsection below. In the following discussion, we use the term *library* to refer to a shared library or DLL as appropriate for the platform.

On this page:

- [C++ Distribution](#)
 - [Discovering Dependencies](#)
- [.NET Distribution](#)
- [Java Distribution](#)
- [Python and Ruby Distributions](#)
- [PHP Distribution](#)

C++ Distribution

The `Ice` library contains the implementation of the core Ice run time. Supplemental libraries provide the stubs and skeletons for the Ice services, as well as utility functions used by Ice, its services, and Ice applications:

- `Glacier2`
- `IceBox`
- `IceGrid`
- `IcePatch2`
- `IceSSL`
- `IceStorm`

The `IceUtil` library is a dependency of the `Ice` library and therefore must be distributed with any Ice application. The `IceXML` library is required by certain Ice services.

Your distribution needs to include only those libraries that your application uses. If your application implements an `IceBox` service, you must also distribute the `IceBox` server executable (`icebox`).

Discovering Dependencies

On Windows, you can use the `dumpbin` utility in a command window to display the dependencies of a DLL or executable. For example, here is the output for the `glacier2router` executable:

```
> dumpbin /dependents glacier2router.exe
ice35.dll
iceutil35.dll
LIBEAY32.dll
glacier235.dll
icessl35.dll
MSVCP100.dll
MSVCR100.dll
KERNEL32.dll
```

We can deduce from the names of the Microsoft Visual C++ run time DLLs that this Ice installation was compiled with Visual Studio 2010. Note that each of these DLLs has its own dependencies, which can be inspected using additional `dumpbin` commands. However, tracking down the dependencies recursively through each DLL can quickly become tedious, therefore you should consider using the [Dependency Walker](#) graphical utility instead.

On Unix, the `ldd` utility displays the dependencies of shared libraries and executables.

.NET Distribution

The `Ice` assembly contains the implementation of the core Ice run time. Supplemental assemblies provide the stubs and skeletons for the Ice services:

- `Glacier2`
- `IceBox`
- `IceGrid`
- `IcePatch2`
- `IceSSL`
- `IceStorm`

Your distribution needs to include only those assemblies that your application uses. If your application implements an `IceBox` service, you must also distribute the `IceBox` server executable (`iceboxnet.exe`).

On Mono, the file `Ice.dll.config` provides a mapping for the Bzip2 DLL. If your application does not use Ice's protocol compression feature, you do not need to distribute this file. Otherwise, you should include the file and verify that its mapping is appropriate for your target platform.

Java Distribution

The `Ice.jar` file contains the core Ice run time and the `IceSSL` plug-in. Supplemental JAR files provide the stubs and skeletons for the Ice services:

- `Glacier2`
- `IceBox`
- `IceGrid`
- `IcePatch2`
- `IceStorm`

If your application uses `Freeze`, you must also distribute `Freeze.jar` along with the Berkeley DB run time libraries and JAR file.

For assistance with packaging your Java application, consider using a utility such as [ProGuard](#).

Python and Ruby Distributions

The Ice run time for a Python or Ruby application consists of the following components:

- the library for the scripting language extension: `IcePy` or `IceRuby`
- the libraries required by the extension: `Ice`, `IceUtil`, and `Slice`
- the source code generated from the `Slice` files in the Ice distribution

In addition, your distribution should include the source code generated for your own `Slice` files, or the `Slice` files themselves if your application loads them dynamically.

PHP Distribution

The Ice run time for a PHP application consists of the following components:

- the library for the scripting language extension: `IcePHP` or `php_ice`
- the libraries required by the extension: `Ice`, `IceUtil`, and `Slice`
- the source code generated from the `Slice` files in the Ice distribution

In addition, your distribution should include the source code generated for your own `Slice` files.

See Also

- [IceGrid Persistent Data](#)
- [Configuring IceStorm](#)