;AddingPublicAPI.rtfd;;¬   Next Section   ;InstallingAPrecompiledHeader.rtfd;;¬   Previous Section

Providing backward compatibility

1 Click the Inspector button.

2 Choose Project Attributes from the Inspector panel.

3 Set the Deploy with version name field in the Project Attributes inspector if you have removed or changed API.

4 Build the project.

When you change a framework, you want to make sure not to break existing programs. If you do one of the following to your framework, you are in danger of breaking programs that link with it:

SquareBullet.eps ¬Remove any public API.
SquareBullet.eps ¬Change any API, such as a method or function declaration or a class name.
SquareBullet.eps ¬Add instance variables to a class.
SquareBullet.eps ¬Rearrange the order of instance variables in a class.
SquareBullet.eps ¬Remove any of the architectures the framework is built for.

FrameworkProjectInspector.eps ¬

Whenever you make one of these changes, you should increment the framework's major version letter and provide both the new and old binary to your users. That way, programs linked against the older version of the framework will still run. New programs or modified programs will link with the newer version of the framework.

A dynamic shared library's name contains the major version letter. This name is recorded in the executable when a program links with the library. Thus, any program that links with a dynamic shared library knows that library's major version. The program won't launch if it can't find a library with the correct name.

For example, if the program **MyProg** links with version A of the framework **Misc**, the path **/LocalLibrary/Frameworks/Misc.framework/Versions/A/Misc** is recorded in **MyProg**. Suppose you add an instance variable to a class in **Misc** and change version to B. This builds **Misc.framework/Versions/B/Misc** but leaves **Versions/A/Misc** intact. Because version A still exists, **MyProg** can still run. If you change **MyProg** and rebuild it, it links with version B.

> The Install in: field of the Build Attributes inspector provides the first half of the framework's full name. Variables such as **$(HOME)** are expanded *before* the path name is recorded. For more information on the Build Attributes inspector, see Chapter€9. ;../../04_BuildingDebugging/09_Building/Building.rtfd;;¬

Related Concept: ;FrameworksLibrariesConcepts.rtfd;linkMarkername TipsandTrickstoChangingtheMajorVersion;, Tips and Tricks to Changing the Major Version