About Xcode and iPhone SDK

Xcode 3.2 with iPhone SDK 3.1 for Mac OS X 10.6 Snow Leopard

Contents

Introduction
Compatibility
What's New
Installation
Deprecation Notice

Introduction

Xcode 3.2 and iPhone SDK 3.1 is the complete developer tools release for Mac OS X 10.6 Snow Leopard. This package includes the Xcode 3.2 developer tools for Mac OS X development as well as the SDK components and iPhone Simulator for iPhone OS 3.1 development.

We encourage developers to apply to the iPhone Developer Program for access to additional support resources, as well as provisioning resources to enable development directly on an iPhone or iPod touch. For more information visit:

http://developer.apple.com/iphone/program/

Compatibility: this package requires an Intel-based Mac running Mac OS X Snow Leopard version 10.6.0 or later.

What's New

• Static code analysis is fully integrated within the Xcode IDE via the Build and Analyze option under the Build menu or via custom build settings
• New Build Results window collects warnings and errors in a persistent log
• GCC 4.2 is the default system compiler for the 10.6 SDK
• The optional LLVM compiler is included using two different front ends - the Clang compiler is a leading-edge parser that offers dramatically improved compile times. For maximum compatibility, the GCC LLVM compiler utilizes the LLVM back-end with the GCC 4.2 parser.
• New template chooser and project assistants
• New opening window for quick access to recent projects
About SDKs

This package includes two types of SDKs: Mac and iPhone. The Mac OS X SDKs include frameworks and headers allowing you to target the corresponding Mac OS X version for which they are numbered. The iPhone SDK works similarly, but also includes additional features such as custom compilers, and the iPhone Simulator, tied to the corresponding iPhone OS. For example, the Simulator may behave differently, according to the iPhone SDK chosen at the time.

Within Xcode you can choose specific Mac OS X SDKs to create applications that run on particular versions of Mac OS X, or use the iPhone SDK to build applications for the iPhone Simulator, the iPhone, or iPod touch. The tools included with the iPhone SDK will upload your application to a device, and debug directly from the IDE.

New in Xcode

- The assistant interface has been completely revamped, making it easier to create "New Project", "New Target", and "New File" resources. This release of Xcode also adds new assistant templates for both Mac OS X and iPhone OS applications.
- New, less-obtrusive message bubbles stay right-justified and take up less room within the editor window, without re-flowing the source code.
- New build menu item “Build and Analyze” will generate build warnings using the new static analyzer, identifying potential coding mistakes by analyzing most possible code paths. These build warnings can also be viewed using the new message bubbles which, when clicked, will display arrows that walk through the steps that can create the coding error.
- New build results window persists results so that old warnings are not lost, allowing a quit and re-launch of Xcode to more easily return to the previous state.
- New optional Clang-LLVM 1.0 compiler uses the much faster Clang front-end parser coupled with the LLVM back-end compiler for fast compiles and fast executable code. Many projects will benefit from this compiler combination, although GCC 4.2 is still the system default. The Clang-LLVM 1.0 compiler will fall back to using LLVM-GCC 4.2 when it encounters C++ code.
- New “Quick Help” feature (option-double-click on an API) gives instant access to the most common documentation information, replacing the Research Assistant. Quick Help will disappear when focus is changed, or the window may be dragged to a more docked position on the screen.
- Documentation is now downloaded from the web by default after installation, and will be automatically updated in the background. If you do not wish to download the documentation to save disk space, you may uncheck the documentation at install time and the docs will instead be viewed from an online server.
• New Developer Documentation window presents search results along the side by category, making it easier to navigate the search results.
• A new Overview toolbar item lets you see and set the project's active Target, Configuration, Executable, Architecture, and SDK with a single control.
• Setting Architectures and SDKs are now done with provided pop-up lists rather than manually entered strings or long paths.
• It is now easier to add known frameworks to your project
• The source code editor now allows you to "Edit All In Scope", to edit all instances of a symbol in a particular scope.
• Build settings can be set for any combination of architecture and SDK.
• Using the Clang compiler requires the 10.6 SDK
• Using GCC LLVM or GCC 4.2 requires the 10.5 SDK or 10.6 SDK

New in Dashcode
• You can now use Dashcode to create web applications optimized for the Safari on the iPhone, Mac, and PC.

For the latest security information visit: http://support.apple.com/kb/HT1222

For more detailed information please see the complete Xcode 3.2 release notes.

Installation
The Xcode and iPhone SDK installer provides six options for configuring the installation from the “Customize...” button:
• **Essentials.** Contains the essential components of Xcode, installed by default in /Developer or another directory of your choice, including external volumes.
• **iPhone SDK.** Contains the iPhone SDK, tools, and iPhone Simulator
• **System Tools.** Includes CHUD performance tools. Its components are always placed in /Developer.
• **UNIX Dev Support.** Command-line tools used for UNIX-based development. Its components are always placed in /usr - only one version installed at a time.
• **Documentation.** Configures Xcode to automatically download updates to developer documentation. If left unchecked the documentation will be read from the web instead of the local disk. Can be changed via Preferences.
• **Mac OS X 10.4 Support.** This option, off by default, adds support for developing applications that target Mac OS X 10.4 APIs.

Throughout this document `<Xcode>` refers to the path in which the Developer Tools Essentials components are installed.
You can now have multiple versions of the Xcode toolset installed. You can move or rename the `<Xcode>` directory, but must not alter its internal structure. If you need to refer to a developer application from a different directory, it is best to use an alias or symbolic link for the full path.

**NOTE:** Only one version of the System Tools and the UNIX Development Support components can be installed on a computer at a time. The last installed set of these components replaces any previously installed set.

**Step-by-Step Instructions**

**Note:** By default, all versions of Xcode (and iPhone SDK) install in `/Developer` with previous versions automatically upgraded. If you want to keep a previously installed version of Xcode, you must either do a "Custom Install" and specify a new `<Xcode>` directory name (e.g. `/Xcode3.2`) or you must move the pre-existing `/Developer` directory to a new name before you install Xcode 3.2. See step 5 below.

1. Boot into a partition with the Mac OS X v10.6.0 or later installed.
2. Download the Xcode and iPhone SDK software.
4. Follow the instructions in the Installer.
5. To install Xcode in a directory other than the default `/Developer`, you must specify a new `<Xcode>` directory name on the "Custom Install" pane
   - Select the folder icon under "Location" next to the "Essentials" package.
   - Select "Other..." at the top of the pop-up.
   - Navigate to the location for the new `<Xcode>` folder. Select "New Folder".
   - Enter a folder name in the "New Folder" window; Select "Create".
   - Finally, select "Choose" in the "Install Xcode Tools" pane.
6. If you need to install Mac OS X 10.4 support you must select "Customize", and check that package on the "Custom Install" panel.
7. Authenticate as the administrative user. The first user you create when setting up Mac OS X has administrator privileges by default.

Once you have installed the Xcode developer tools, you can access the documentation by launching Xcode and choosing any of the items in the Help menu. Developer applications such as Xcode, Instruments, and Interface Builder are installed in `<Xcode>/Applications`.

**Uninstalling Xcode Developer Tools**

To uninstall Xcode developer tools on the boot volume along with the `<Xcode>` directory, from a Terminal window type:
$ sudo <Xcode>/Library/uninstall-devtools --mode=all

To remove the underlying developer content on the boot volume, but leave the <Xcode> directory and supporting files untouched, from a Terminal window type:

$ sudo <Xcode>/Library/uninstall-devtools --mode=systemsupport

To just remove the UNIX development support on the boot volume, but leave the <Xcode> directory and supporting files untouched, from a Terminal window type:

$ sudo <Xcode>/Library/uninstall-devtools --mode=unixdev

Finally, to just uninstall the <Xcode> directory you can simply drag it to the trash, or from a Terminal window type:

$ sudo <Xcode>/Library/uninstall-devtools --mode=xcodedir

**NOTE:** The uninstaller that ships with previous versions of the Xcode developer tools will not clean everything off of your system properly. You should use the one installed with the Xcode 3.2 developer tools.

**Deprecation Notice**

The following directories inside the Xcode are deprecated and will be removed in a future version of the Xcode developer tools:

- `<Xcode>/Tools` (content will move to `<Xcode>/usr/bin`)
- `<Xcode>/Makefiles` (content will move to `<Xcode>/usr/share/makefiles`)

© 2009 Apple Inc. All rights reserved. Xcode, Apple, the Apple logo, Mac, Mac OS, and Macintosh are trademarks of Apple Inc., registered in the U.S. and other countries.