About Xcode for Mac, iPad, and iPhone

Xcode 3.2.2 developer tools for Mac SDK 10.6 and iPhone SDK 3.2

Contents

Introduction

Xcode is the complete developer toolset for creating Mac OS X and iPhone OS applications. This package installs the Xcode IDE, performance analysis tools, iPhone Simulator, and OS framework bundles in the form of Mac SDKs and iPhone SDKs.

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

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

Compatibility: Xcode 3.2 requires an Intel-based Mac running Mac OS X Snow Leopard version 10.6.2 or later. Note: some distributions of Xcode do not include the iPhone SDKs; you can verify their presence in the Installer's "Custom Install" pane.

What's New

- Xcode 3.2.2 adds support for developing iPad and Universal iPad/iPhone apps
- New menu option: Project -> Upgrade Current Target for iPad...
- To configure Universal apps that run on both iPad and iPhone, in Build Settings:
  - set `Base SDK` in the Architectures section to "iPhone SDK 3.2"
  - set `iPhone OS Deployment Target` to iPhone OS 3.1.3 or earlier
- For iPad-only development, use iPhone SDK 3.2 and target iPhone OS 3.2
- For iPhone-only, use iPhone SDK 3.1.3 and target iPhone OS 3.1.3 or earlier
- Be sure to set `Targeted Device Family` to iPad, iPhone, or iPhone/iPad

About SDKs and the iPhone Simulator

This package includes two types of SDKs: Mac and iPhone. The Mac OS X SDKs include frameworks and headers that allow you to target the corresponding Mac OS X version number. The iPhone SDK works similarly, but also includes additional features such as bundled compilers, and the iPhone Simulator.

There are two important settings in the build system related to SDKs and the OS you are targeting: the Base SDK, and the Mac OS X or iPhone OS Deployment Target. The Base SDK refers to the sets of frameworks used to build your application, and should usually be the most recent. The Deployment Target refers to the minimum OS you wish to support with your application, perhaps one major release back.

NOTE: the Simulator will always run an iPhone OS version that matches the Base SDK used to build your app, regardless of your iPhone OS Deployment Target choice.

For final qualification on a specific version of iPhone OS you must test on a device.

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 optional LLVM compiler uses the much faster Clang front-end parser coupled with the LLVM back-end compiler for fast compiles and fast executable code. The LLVM GCC 4.2 compiler benefits from the improved back-end code generation of LLVM, but uses the GCC 4.2 parser to maintain backward compatibility and add C++ support. The LLVM compiler will fall back to using LLVM GCC 4.2 when it encounters C++ 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, less-obtrusive message bubbles stay right-justified and take up less room within the editor window, without re-flowing the source code.
- 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 “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 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.
- GCC 4.2 is the default system compiler for the 10.6 SDK
- Using the LLVM compiler requires the 10.6 SDK
- Using LLVM GCC or GCC 4.2 requires the 10.5 SDK or 10.6 SDK

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

For more detailed information please see the complete Xcode 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 Development.** 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 SDK.** 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.

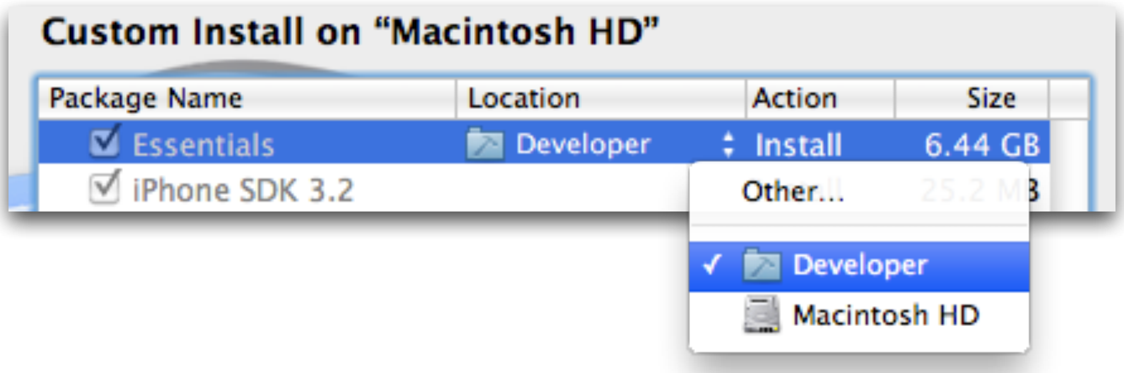
You can now have multiple versions of the Xcode toolset installed. You can move or rename the `<Xcode>` directory, but must not alter its internal structure. If you need to refer to a developer application from a different directory, it is best to use an alias or symbolic link for the full path.

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.

Custom Install: Step-by-Step

By default, Xcode installs in `/Developer` with previous versions automatically upgraded. To keep a previously installed version of Xcode, you must either select "Custom Install" and specify a new <Xcode> directory, or rename the existing Xcode home directory (`/Developer`) before installing this new version. See step 4 below.

1. Boot into a partition with the required Mac OS X 10.6 version, or later installed.
2. Download the Xcode installation package.
3. Double-click on the .mpkg Installer package (icon is an open brown box).
4. 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.



Custom Install on "Macintosh HD"

Package Name	Location	Action	Size
✓ Essentials	Developer	Install	6.44 GB
✓ iPhone SDK 3.2			25.2 MB

   - 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.
5. Authenticate as an 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 these Xcode 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)