Create your first app ›

Start building Flutter iOS apps on macOS

Get started › Install › macOS › Make iOS apps

Contents

Verify system requirements

To install and run Flutter, your macOS environment must meet the following hardware and software requirements.

Hardware requirements

Your macOS Flutter development environment must meet the following minimal hardware requirements.

Requirement	Minimum	Recommended
CPU Cores	4	8
Memory in GB	8	16
Display resolution in pixels	WXGA (1366 x 768)	FHD (1920 x 1080)
Free disk space in GB	44.0	70.0

Software requirements

To write and compile Flutter code for iOS, install the following packages.

Tip

You do not need to install Dart separately as the Flutter SDK includes the full Dart SDK.

Operating system

Flutter supports developing on macOS 11 (Big Sur) or later. This guide presumes your Mac runs the `zsh` as your default shell.

▶ **To verify your shell configuration, expand this section**

Some Flutter components require the Rosetta 2 translation process on Macs running Apple silicon. To run all Flutter components on Apple silicon, install Rosetta 2.

```
$ sudo softwareupdate --install-rosetta --agree-to-license
```

Development tools

Download and install the following packages.

- Xcode 16 to debug and compile native Swift or ObjectiveC code. The Xcode installation includes Git 2.27 or later to manage source code.

- CocoaPods 1.16 to compile and enable Flutter plugins in your native apps.

The developers of the preceding software provide support for those products. To troubleshoot installation issues, consult that product's documentation.

When you run the current version of `flutter doctor`, it might list a different version of one of these packages. If it does, install the version it recommends.

Text editor or integrated development environment

You can build apps with Flutter using any text editor or integrated development environment (IDE) combined with Flutter's command-line tools.

Using an IDE with a Flutter extension or plugin provides code completion, syntax highlighting, widget editing assists, debugging, and other features.

Popular options include:

- Visual Studio Code 1.77 or later with the Flutter extension for VS Code.
- Android Studio 2023.3.1 (Jellyfish) or later with the Flutter plugin for IntelliJ.
- IntelliJ IDEA 2023.3 or later with both the Flutter plugin for IntelliJ and the Android plugin for IntelliJ.

> **Recommended**
>
> The Flutter team recommends installing Visual Studio Code 1.77 or later and the Flutter extension for VS Code. This combination simplifies installing the Flutter SDK.

Install the Flutter SDK

To install the Flutter SDK, you can use the VS Code Flutter extension or download and install the Flutter bundle yourself.

Use VS Code to install   Download and install

Download then install Flutter

To install Flutter, download the Flutter SDK bundle from its archive, move the bundle to where you want it stored, then extract the SDK.

1. Download the following installation bundle to get the latest stable release of the Flutter SDK.

Intel Processor	Apple Silicon
flutter_macos_3.27.2-stable.zip	flutter_macos_arm64_3.27.2-stable.zip

   For other release channels, and older builds, check out the SDK archive.

   The Flutter SDK should download to the macOS default download directory: `~/Downloads/`.

2. Create a folder where you can install Flutter.

   Consider creating a directory at `~/development/`.

3. Extract the file into the directory you want to store the Flutter SDK.

```
$ unzip ~/Downloads/flutter_macos_arm64_3.27.2-stable.zip \
       -d ~/development/
```

   When finished, the Flutter SDK should be in the `~/development/flutter` directory.

Add Flutter to your `PATH`

To run Flutter commands in the Terminal, add Flutter to the `PATH` environment variable. This guide presumes your Mac runs the latest default shell, `zsh`. Zsh uses the `.zshenv` file for environment variables.

1. Launch your preferred text editor.

2. If it exists, open the Zsh environmental variable file `~/.zshenv` in your text editor. If it doesn't, create `~/.zshenv`.

3. Copy the following line and paste it at the end of your `~/.zshenv` file.

   ```bash
   export PATH=$HOME/development/flutter/bin:$PATH
   ```

4. Save your `~/.zshenv` file.

5. To apply this change, restart all open terminal sessions.

If you use another shell, check out this tutorial on setting your PATH.

Configure iOS development

Install and configure Xcode

To develop Flutter apps for iOS, install Xcode to compile to native bytecode.

1. Open the **App Store** and sign in.

2. Search for `Xcode`.

3. Click **Install**.

   The Xcode installer takes up 6+ GB of storage. The download might take some time.

4. To configure the command-line tools to use the installed version of Xcode, use the following commands.

   ```
   $ sudo sh -c 'xcode-select -s /Applications/Xcode.app/Contents/Developer && xcodebuild -
   runFirstLaunch'
   ```

   Use this path for the latest version of Xcode. If you need to use a different version, specify that path instead.

5. Sign the Xcode license agreement.

   ```
   $ sudo xcodebuild -license
   ```

Try to keep to the current version of Xcode.

Configure your target iOS device

With Xcode, you can run Flutter apps on an iOS device or on the simulator.

Physical device

Install CocoaPods

If your apps depend on Flutter plugins with native iOS code, install CocoaPods. This program bundles various dependencies across Flutter and iOS code.

To install and set up CocoaPods, run the following commands:

1. Install `cocoapods` following the CocoaPods install guide.

   ```
   $ sudo gem install cocoapods
   ```

2. Launch your preferred text editor.

3. Open the Zsh environmental variable file `~/.zshenv` in your text editor.

4. Copy the following line and paste it at the end of your `~/.zshenv` file.

```bash
export PATH=$HOME/.gem/bin:$PATH
```

5. Save your `~/.zshenv` file.

6. To apply this change, restart all open terminal sessions.

Check your development setup

Help

Run Flutter doctor

The `flutter doctor` command validates that all components of a complete Flutter development environment for macOS.

1. Open your Terminal.

2. To verify your installation of all the components, run the following command.

   ```
   $ flutter doctor
   ```

As you chose to develop for iOS, you do not need *all* components. If you followed this guide, the result of your command should resemble:

```
Running flutter doctor...
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.27.0, on macOS 14.4.0 23E214 darwin-arm64, locale en)
[!] Android toolchain - develop for Android devices
[!] Chrome - develop for the web
[✓] Xcode - develop for iOS and macOS (Xcode 16)
[!] Android Studio (not installed)
[✓] VS Code (version 1.95)
[✓] Connected device (1 available)
[✓] Network resources


! Doctor found issues in 3 categories.
```

Troubleshoot Flutter doctor issues

When the `flutter doctor` command returns an error, it could be for Flutter, VS Code, Xcode, the connected device, or network resources.

If the `flutter doctor` command returns an error for any of these components, run it again with the verbose flag.

```
$ flutter doctor -v
```

Check the output for other software you might need to install or further tasks to perform.

If you change the configuration of your Flutter SDK or its related components, run `flutter doctor` *again* to verify the installation.

Start developing iOS on macOS apps with Flutter

Congratulations. Having installed all prerequisites and the Flutter SDK, you can start developing Flutter apps for iOS on macOS.

To continue on your learning journey, consult the following guides:

- Learn how to write your first Flutter app
- Flutter fundamentals docs

Manage your Flutter SDK

To learn more about managing your Flutter SDK install, consult the following resources.

- Upgrade Flutter
- Add Android compilation tools
- Add web as a build target
- Uninstall Flutter

Create your first app 〉