Windows install

Docs Get started Install Windows

Contents

System requirements

To install and run Flutter, your development environment must meet these minimum requirements:

- **Operating Systems**: Windows 7 SP1 or later (64-bit)
- **Disk Space**: 400 MB (does not include disk space for IDE/tools).
- **Tools**: Flutter depends on these tools being available in your environment.
  - Windows PowerShell 5.0 or newer (this is pre-installed with Windows 10)
  - Git for Windows 2.x, with the **Use Git from the Windows Command Prompt** option.

    If Git for Windows is already installed, make sure you can run `git` commands from the command prompt or PowerShell.

Get the Flutter SDK

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

   flutter_windows_v1.12.13+hotfix.9-stable.zip

For other release channels, and older builds, see the SDK archive page.

2. Extract the zip file and place the contained `flutter` in the desired installation location for the Flutter SDK (for example, `C:\src\flutter`; do not install Flutter in a directory like `C:\Program Files\` that requires elevated privileges).

If you don't want to install a fixed version of the installation bundle, you can skip steps 1 and 2. Instead, get the source code from the Flutter repo on GitHub, and change branches or tags as needed. For example:

```
C:\src>git clone https://github.com/flutter/flutter.git -b stable
```
content_copy

You are now ready to run Flutter commands in the Flutter Console!

Update your path

If you wish to run Flutter commands in the regular Windows console, take these steps to add Flutter to the `PATH` environment variable:

- From the Start search bar, enter ‘env’ and select **Edit environment variables for your account**.
- Under **User variables** check if there is an entry called **Path**:
  - If the entry exists, append the full path to `flutter\bin` using `;` as a separator from existing values.
  - If the entry doesn’t exist, create a new user variable named `Path` with the full path to `flutter\bin`as its value.

Note that you have to close and reopen any existing console windows for these changes to take effect.

Run `flutter doctor`

From a console window that has the Flutter directory in the path (see above), run the following command to see if there are any platform dependencies you need to complete the setup:

```
C:\src\flutter>flutter doctor content_copy
```

This command checks your environment and displays a report of the status of your Flutter installation. Check the output carefully for other

software you might need to install or further tasks to perform (shown in **bold** text).

For example:

```
[-] Android toolchain - develop for
Android devices
    • Android SDK at D:\Android\sdk
    ✗ Android SDK is missing command
line tools; download from
https://goo.gl/XxQghQ
    • Try re-installing or updating your
Android SDK,
      visit
https://flutter.dev/setup/#android-setup
for detailed instructions.
```


The following sections describe how to perform these tasks and finish the setup process. Once you have installed any missing dependencies, you can run the `flutter doctor` command again to verify that you've set everything up correctly.

Warning: The `flutter` tool uses Google Analytics to anonymously report feature usage statistics and basic crash reports. This data is used to help improve Flutter tools over time.

Flutter tool analytics are not sent on the very first run. To disable reporting, type `flutter config --no-analytics`. To display the current setting, type `flutter config`. If you opt out of analytics, an opt-out event will be sent, and then no further information will be sent by the Flutter tool.

By downloading the Flutter SDK, you agree to the Google Terms of Service. Note: The Google Privacy Policy describes how data is handled in this service.

Moreover, Flutter includes the Dart SDK, which may send usage metrics and crash reports to Google.

Android setup

Note: Flutter relies on a full installation of Android Studio to supply its Android platform dependencies. However, you can write your Flutter apps in a number of editors; a later step will discuss that.

Install Android Studio

1. Download and install Android Studio.
2. Start Android Studio, and go through the 'Android Studio Setup Wizard'. This installs the latest Android SDK, Android SDK Command-line Tools, and Android SDK Build-Tools, which are required by Flutter when developing for Android.

Warning: In Android Studio 3.6 or later, you need to manually add the old version of the Android SDK Tools for Flutter to work. To do this:

1. Open the **Android Studio SDK Manager**
2. In the Android SDK tab, uncheck **Hide Obsolete Packages**
3. Check **Android SDK Tools (Obsolete)**

The dialog below shows the appropriate settings:

This is a known issue that will be addressed in an upcoming version of Flutter.

Set up your Android device

To prepare to run and test your Flutter app on an Android device, you'll need an Android device running Android 4.1 (API level 16) or higher.

1. Enable **Developer options** and **USB debugging** on your device. Detailed instructions are available in the Android documentation.
2. Windows-only: Install the Google USB Driver.
3. Using a USB cable, plug your phone into your computer. If prompted on your device, authorize your computer to access your device.
4. In the terminal, run the `flutter devices` command to verify that Flutter recognizes your connected Android device. By default, Flutter uses the version of the Android SDK where your `adb` tool is based. If you want Flutter to use a different installation of the Android SDK, you must set the `ANDROID_HOME`environment variable to that installation directory.

Set up the Android emulator

To prepare to run and test your Flutter app on the Android emulator, follow these steps:

1. Enable VM acceleration on your machine.
2. Launch **Android Studio > Tools > Android > AVD Manager** and select **Create Virtual Device**. (The **Android** submenu is only present when inside an Android project.)
3. Choose a device definition and select **Next**.
4. Select one or more system images for the Android versions you want to emulate, and select **Next**. An *x86* or *x86_64* image is recommended.
5. Under Emulated Performance, select **Hardware - GLES 2.0** to enable hardware acceleration.
6. Verify the AVD configuration is correct, and select **Finish**.

   For details on the above steps, see Managing AVDs.

7. In Android Virtual Device Manager, click **Run** in the toolbar. The emulator starts up and displays the default canvas for your selected OS version and device.

Web setup

Note: As of 1.12, Flutter has early support for running web applications, but you need to be running the beta channel of Flutter. If you experience a problem that hasn't yet been reported, please file an issue and make sure that "web" appears in the title.

To prepare to run, test, and debug your Flutter app on the web, you must install Chrome, if you haven't already.