Building a web application with Flutter

Contents

This page covers the following steps for getting started with web support:

- Configure the `flutter` tool for web support.
- Create a new project with web support.
- Run a new project with web support.
- Build an app with web support.
- Add web support to an existing project.

Requirements

To create a Flutter app with web support, you need the following software:

- Flutter SDK. See the Flutter SDK installation instructions.
- Chrome; debugging a web app requires the Chrome browser.
- Optional: An IDE that supports Flutter. You can install Android Studio, IntelliJ IDEA, or Visual Studio Code and install the Flut and Dart plugins to enable language support and tools for refactoring, running, debugging, and reloading your web app with an editor. See setting up an editor for more details.

For more information, see the web FAQ.

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

Create a new project with web support

You can use the following steps to create a new project with web support.

Set up

Run the following commands to use the latest version of the Flutter SDK from the beta channel and enable web support:

```
$ flutter channel beta
$ flutter upgrade
$ flutter config --enable-web
```

Warning: Running `flutter channel beta` replaces your current version of Flutter with the beta version and can take time i your connection is slow. After this, running `flutter upgrade` upgrades your install to the latest `beta`. Returning to the stable channel (or any other) requires calling `flutter channel <channel>` explicitly.

Note: The `flutter upgrade` command silently fails when `origin` points to a personal fork. To validate that `origin` points to `https://github.com/flutter/flutter.git`, run the following commands in the root directory of your local copy of the `https://github.com/flutter/flutter` repository:

```
$ cd <inside local copy of the flutter/flutter repo>
$ git remote get-url origin
https://github.com/flutter/flutter.git
```

Once web is enabled, the `flutter devices` command outputs a `Chrome` device that opens the Chrome browser with your app running, and a `Web Server` that provides the URL serving the app.

```
$ flutter devices
2 connected device:

Web Server • web-server • web-javascript • Flutter Tools
Chrome     • chrome     • web-javascript • Google Chrome 81.0.4044.129
```

After enabling web support, restart your IDE. You should now see **Chrome (web)** and **Web Server (web)** in the device pulldown.

Note: You only need to execute `flutter config --enable-web` once. You can always check the status of your configuration using the no-argument `flutter config` command.

Create and run

Creating a new project with web support is no different than creating a new Flutter project for other platforms.

Once you've configured your environment for web support, you can create and run a web app either in the IDE or from the comma line.

IDE

After you've configured your environment to support the web, make sure you restart the IDE if it was already running.

Create a new app in your IDE and it automatically creates iOS, Android, and web versions of your app. (And macOS, too, if you've enabled desktop support.) From the device pulldown, select **Chrome (web)** and run your app to see it launch in Chrome.

Command line

To create a new app that includes web support (in addition to mobile support), run the following commands, substituting `myapp` w the name of your project:

```
$ flutter create myapp
$ cd myapp
```

To serve your app from `localhost` in Chrome, enter the following from the top of the package:

```
$ flutter run -d chrome
```

Note: If there aren't any other connected devices, the `-d chrome` is optional.

The `flutter run` command launches the application using the development compiler in a Chrome browser.

Build

Run the following command to generate a release build:

```
$ flutter build web
```

A release build uses dart2js (instead of the development compiler) to produce a single JavaScript file `main.dart.js`. You can cre a release build using release mode (`flutter run --release`) or by using `flutter build web`. This populates a `build/web` direc with built files, including an `assets` directory, which need to be served together.

For more information, see Build and release a web app.

Add web support to an existing app

To add web support to an existing project, run the following command in a terminal from the root project directory:

```
$ flutter create .
```