Flutter Cross-platform UI toolkit

START HERE

`flutter create <name>`
New project

`flutter analyze`
Lint the code

`flutter doctor`
Check your setup

`flutter build apk`
Release build

`flutter devices`
List targets

`flutter run`
Run with hot reload

`flutter pub add <pkg>`
Add a package

`flutter test`
Run tests

Edit Dart code → `save / r` → **Hot reload** keeps state → `R` → **Hot restart** fresh state → `flutter build` → **Release** apk / ipa / web

CONCEPTS

Press r in a flutter run session to hot reload; state is kept. R restarts and resets state.

flutter pub replaces dart pub and flutter test replaces dart test inside Flutter projects.

PROJECT

`flutter create <name>`
Create a new app

`flutter create --platforms=android,web .`
Create into current dir for chosen platforms

`flutter create --template=package <name>`
Create a Dart package
Other templates: plugin, module, skeleton.

`flutter create --org <com.example> <name>`
Set the reverse-domain org id

`flutter clean`
Delete build/ and .dart_tool/

`flutter gen-l10n`
Generate localization classes

RUN AND DEBUG

`flutter run`
Run the app on the selected device

`flutter run -d <device-id>`
Pick a device

RUN AND DEBUG (cont.)

`flutter run -d all`
Run on every connected device

`flutter run -t <file.dart>`
Use another entrypoint

`flutter run --release`
Run an optimized build

`flutter run --profile`
Profile performance on a real device

`flutter run --flavor <name>`
Build a flavor / scheme

`flutter run --dart-define=<K>=<V>`
Pass a compile-time constant

`flutter logs`
Stream device logs

IN A RUN SESSION

`r`
Hot reload

`R`
Hot restart

`q`
Quit and stop the app

`h`
List all session commands

`d`
Detach, leave the app running

`p`
Toggle the widget debug overlay

IN A RUN SESSION (cont.)

`P`
Toggle performance overlay

DEVICES

`flutter devices`
List connected devices

`flutter emulators`
List available emulators

`flutter emulators --launch <id>`
Start an emulator

`flutter config --enable-<platform>-desktop`
Enable a desktop platform
Platforms: linux, windows, macos; web uses --enable-web.

PACKAGES

`flutter pub get`
Fetch dependencies

`flutter pub add <pkg>`
Add a dependency

`flutter pub add dev:<pkg>`
Add a dev dependency

`flutter pub remove <pkg>`
Remove a dependency

`flutter pub upgrade`
Upgrade within constraints

`flutter pub upgrade --major-versions`
Raise constraints to latest majors

PACKAGES (cont.)

`flutter pub outdated`
Show packages with newer versions

`flutter pub deps`
Print the dependency tree

`flutter pub run build_runner build`
Run code generation

TEST AND ANALYZE

`flutter test`
Run all tests in test/

`flutter test <path/to/file_test.dart>`
Run one test file

`flutter test --plain-name <text>`
Run tests by name substring

`flutter test --name <regex>`
Run tests matching a regex

`flutter test --coverage`
Write coverage/lcov.info

`flutter test --update-goldens`
Regenerate golden images

`flutter test integration_test -d <id>`
Run integration tests on a device

`flutter analyze`
Static analysis

`dart format .`
Format all Dart files

TEST AND ANALYZE (cont.)

`dart fix --apply`
Apply automated lint fixes

BUILD

`flutter build apk`
Android APK

`flutter build apk --split-per-abi`
One APK per CPU architecture

`flutter build appbundle`
Android App Bundle for Play

`flutter build ios`
iOS build (macOS only)

`flutter build ipa`
iOS archive for the App Store

`flutter build web`
Web build into build/web

`flutter build linux`
Linux desktop build

`flutter build windows`
Windows desktop build

`flutter build macos`
macOS desktop build

`flutter build <target> --build-name=<x.y.z> --build-number=<n>`
Override version

SDK AND SETUP

`flutter doctor`
Check toolchain and platforms

`flutter doctor -v`
Detailed doctor output

SDK AND SETUP (cont.)

`flutter doctor --android-licenses`
Accept Android SDK licenses

`flutter --version`
Show SDK, engine, Dart versions

`flutter upgrade`
Upgrade the SDK

`flutter channel`
List channels

`flutter channel <stable|beta|main>`
Switch channel

`flutter help <command>`
Help for one command

RECIPES

New app, verify, run

```
flutter create --org
com.example my_app
cd my_app
flutter analyze && flutter test
flutter run -d chrome
```

Release APK for sideloading

```
flutter clean
flutter pub get
flutter build apk --release --
split-per-abi
# output: build/app/outputs/
flutter-apk/
```

Play Store bundle with a build number

```
flutter build appbundle --
release \
  --build-name=1.2.0 --build-
number=42
# output: build/app/outputs/
bundle/release/
```