This provides an overview of the structure of the flutter/plugins and flutter/packages repository.

flutter/plugins

Plugins

flutter/plugins uses the federated plugin model. If you are not familiar with federated plugins, start with reading the federated plugin overview to understand the terms below.

All plugins are located in the `packages/` directory. In theory, each plugin should have the following layout:

- `some_plugin/` - A directory containing the individual packages of the federated plugin:
  - `some_plugin/` - The app-facing package
  - `some_plugin_platform_interface/` - The platform interface
  - `some_plugin_android/`, `some_plugin_ios/`, `some_plugin_web/`, `some_plugin_windows/`, `some_plugin_macos/`, and/or `some_plugin_linux/` - The individual platform implementations, as applicable
    - In some special cases, implementation packages have different names; examples include `webview_flutter_wkwebview` and `in_app_purchase_storekit`. These would normally be named `_ios`, but have more generic names because they are expected to include macOS implemntations in the future. Sharing a package allows sharing the code, as the OS APIs are largely the same across the two platforms.

This layout reflects the goal of having all plugins in flutter/plugins being fully federated. (While this is not strictly necessary, as all packages are being maintained by the Flutter team, using a fully federated structure ensures that we are testing the federated model and finding issues and areas for improvement specific to federation.)

In practice (as of November 2021) most plugins are not yet fully federated, and some are not federated at all, since the process of converting them from their pre-federated forms is ongoing. If you are looking for fully federated plugins as examples, consider `in_app_purchase`, `path_provider`, `shared_preferences`, or `url_launcher`.

The two most common alternate structures are:

Semi-federated

- `some_plugin/`
  - `some_plugin/` - The app-facing package, also containing `ios/` and `android/` folders containing their implementations
  - `some_plugin_platform_interface/` - The platform interface
  - `some_plugin_web/`, `some_plugin_windows/`, `some_plugin_macos/`, and/or `some_plugin_linux/` - Federated implementations

These are packages that have mobile implementations that predate the federated model, but have had a platform implementation extracted to add new platforms via federation. Eventually the `ios/` and `android/` implementations will be extracted to their own packages.

Unfederated

- `some_plugin/` - An unfederated plugin containing mobile implementations

These are packages that predate the federated model and have not had web or desktop implementations added. They should be federated at some point as well, unless extending them to other platforms would never make sense.

Tools

`script/tool/` contains the tooling used to manage tasks across all plugins in the repository. See its README for more information.

flutter/packages

Most packages are located in `packages`. A few which are derived heavily from third-party code are instead in `third_party/packages/`.