Best Practices for Selecting Dart and Flutter Packages

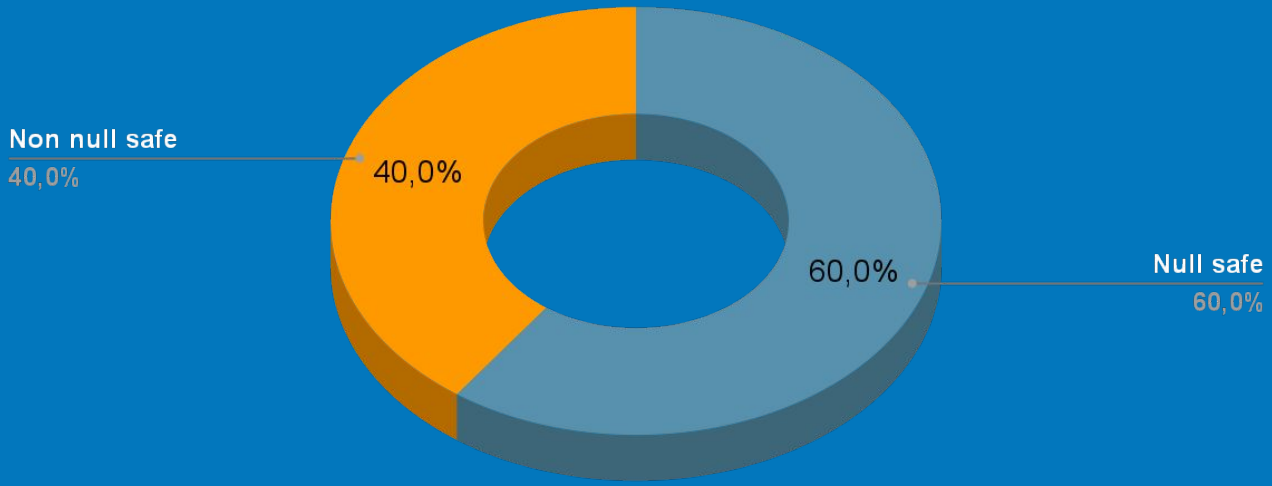
by Guillaume Roux

Using packages: pros & cons

PROS	CONS
• Easy • Saves time • Get updates and fixes • Simplify complexity	• Breaking changes and migrations • Hide how things work • Not always well maintained

Packages supporting null safety

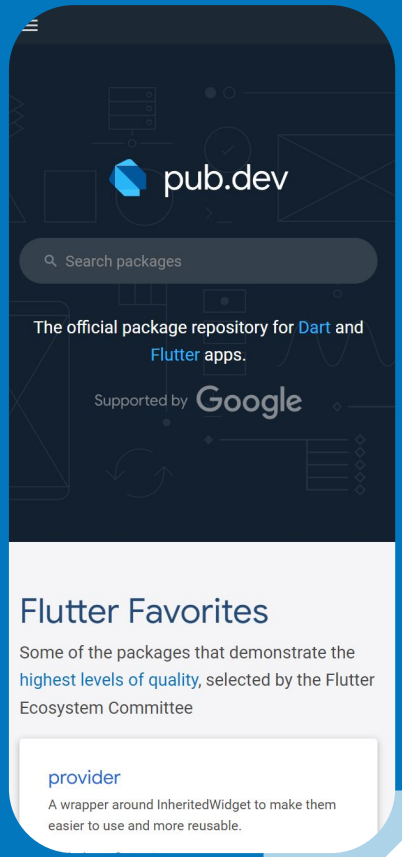
~40,000 packages



How to identify a discontinued or a low quality package?

Informations provided by pub.dev

- **Likes**: Number of developers who liked the package.
- **Pub points**: Metric used to have a general idea of the package's quality (code style, documentation, examples, supported platforms).
- **Popularity**: Measures the number of apps that depend on the package over the past 60 days.
- **Publication date**: Date of the last published version.
- **Flutter Favorite**: A badge indicating that the package has passed high quality standards in the eye of a "Flutter Ecosystem Committee".
- **Discontinued badge**: A badge which can be set by a publisher to explicitly indicate the package's discontinuation.



Informations provided by the Repository

- **Commits**: If there's no recent commit on the repository it might be either that the package has attained its purpose and doesn't need further improvements (but it should have at least releases to stay updated with the latest Dart/Flutter versions) or that it has been silently discontinued (accumulating issues can be an indicator)
- **Tests**: A package with well written tests indicates that the codebase was made with maintainability in mind which is always a good sign.
- **Open Issues**: If the package hasn't been updated for some time there might be compatibility problems or deprecated APIs. (Or even issues asking if the repo is still maintained)

Great! But... How does it guarantee that a package will be maintained?


THAT'S THE NEAT PART: IT DOESN'T

Different types of maintainers

Framework/Language maintainers

Packages made by the **Dart or Flutter team** will probably stay around for as long as the main technologies are around.

Flutter

Dart

Companies

Companies: If a company relies on Flutter/Dart to make money they are going to keep maintaining packages they have published as long as needed. (ex: **Very Good Ventures**, **Serverpod**, **Shorebird, Invertase**)

Individual Developers

Individual Developers*: The riskier category, if a developer stops working on the package without giving the appropriate rights to a new maintainer, then you will have to look for an alternative.

** Consider sponsoring individuals who make good packages to keep them motivated*

Now, a few tips!

Ask yourself if you really need a package for what you want

Dependency Inversion for widgets

```dart
class AppAssetImage extends StatelessWidget {
  const AppAssetImage(
      // ...
      );

  final String assetName;
  final double? width;
  final double? height;
  final BoxFit? fit;
  final Color? color;

  @override
  Widget build(BuildContext context) {
    if (assetName.endsWith('.svg')) {
      final localColor = color;

      return SvgPicture.asset(
        assetName,
        width: width,
        height: height,
        fit: fit ?? BoxFit.contain,
        colorFilter: localColor != null
            ? ColorFilter.mode(localColor, BlendMode.srcIn)
            : null,
      );
    }

    return Image.asset(
      assetName,
      width: width,
      height: height,
      fit: fit,
      color: color,
    );
  }
}
```

Dependency Inversion for services

```dart
abstract interface class AppNetwork {
  Future<bool> get isConnected;
  Stream<bool> get onConnectivityChanged;
}

class AppNetworkImpl implements AppNetwork {
  @override
  Future<bool> get isConnected async {
    final result = await Connectivity().checkConnectivity();
    return result.isConnected;
  }

  Stream<bool>? _connectivityStream;

  @override
  Stream<bool> get onConnectivityChanged {
    return _connectivityStream ??= Connectivity().onConnectivityChanged.map(
          (result) => result.isConnected,
        );
  }
}

extension on ConnectivityResult {
  bool get isConnected {
    return this == ConnectivityResult.wifi ||
        this == ConnectivityResult.mobile ||
        this == ConnectivityResult.ethernet;
  }
}
```

Bonus: Check Flutter Gems

- Packages curated and examined by the community
- Filtered by category and usage
- Combine info from Pub.dev and GitHub

Thank you for your attention!
Find more here