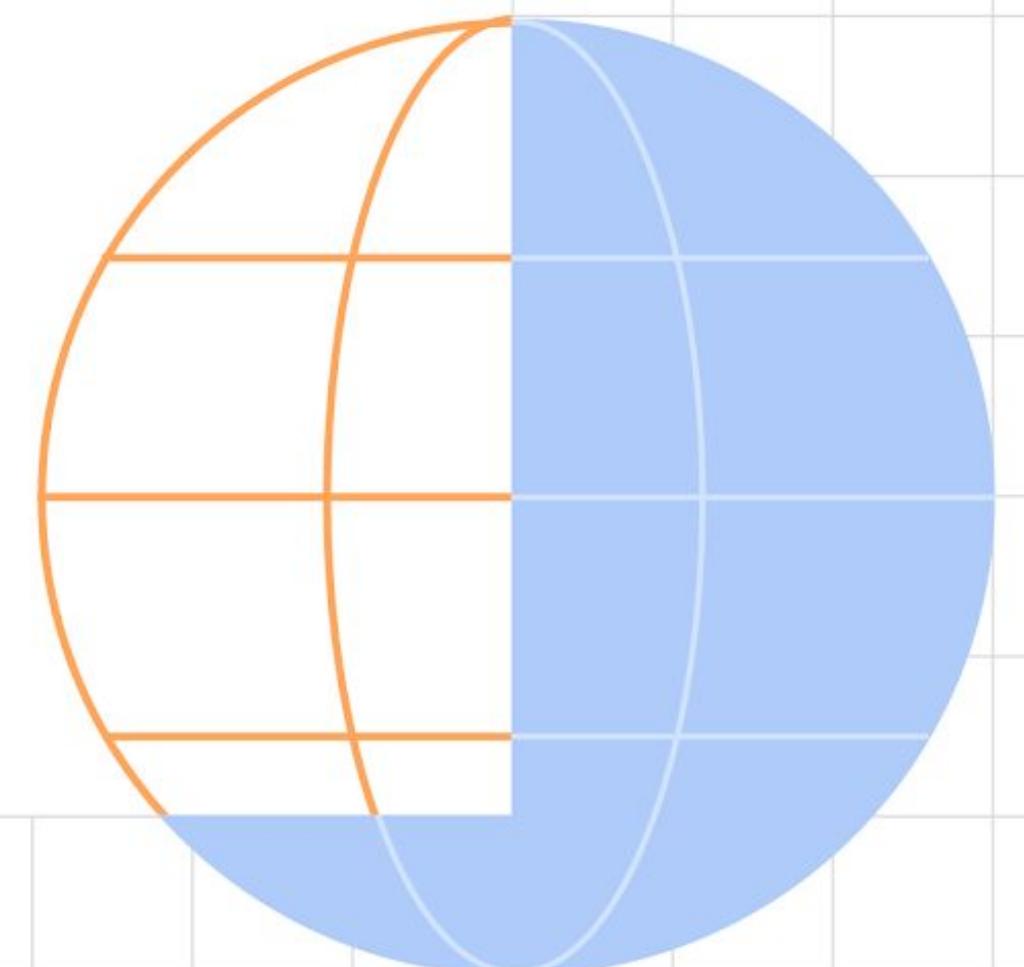




Make the code work for you: Flutter Code Generation



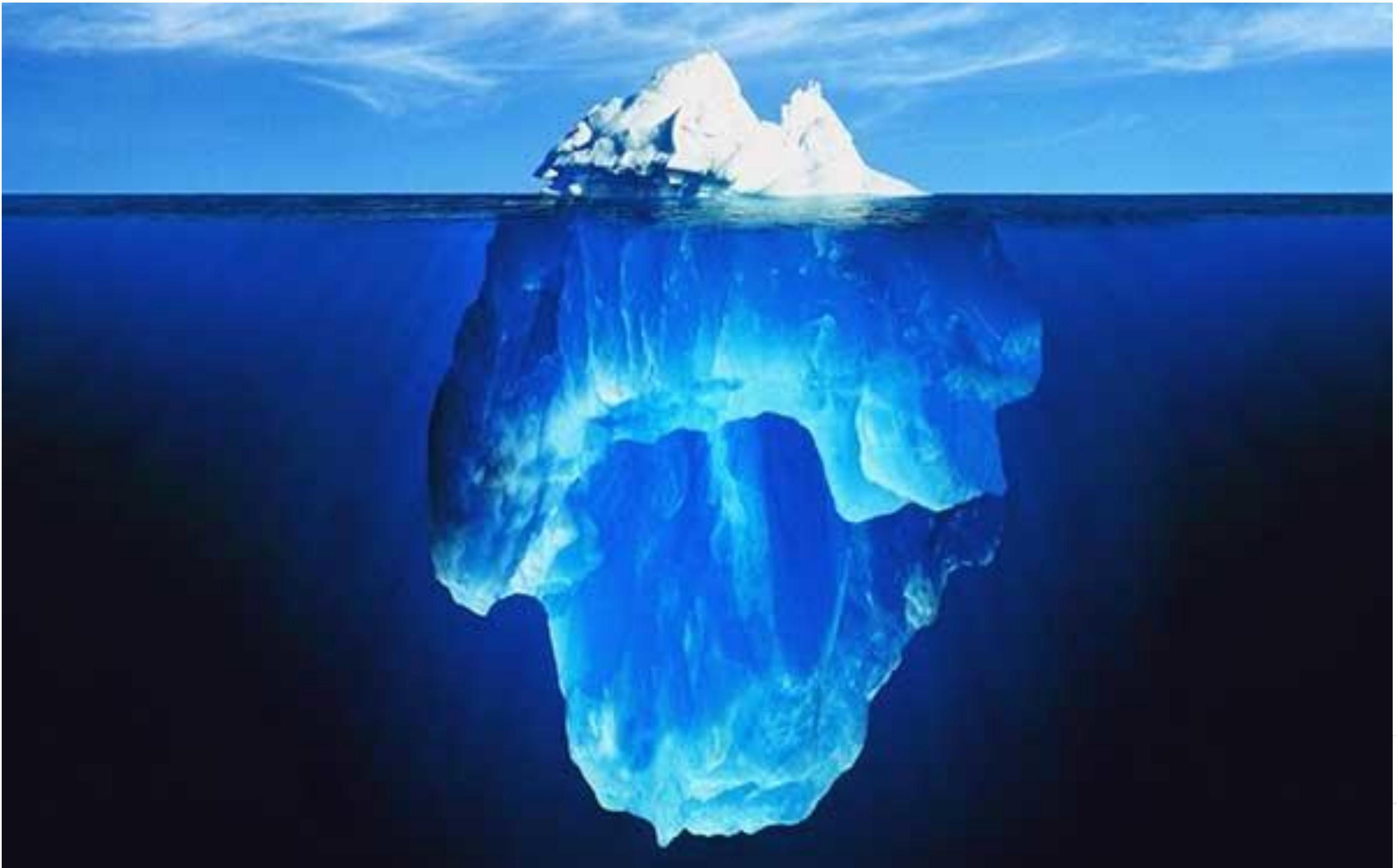
Mangirdas Kazlauskas
GDE for Flutter & Dart
@mkobuolys



Expectations

This talk

Flutter Code Generation



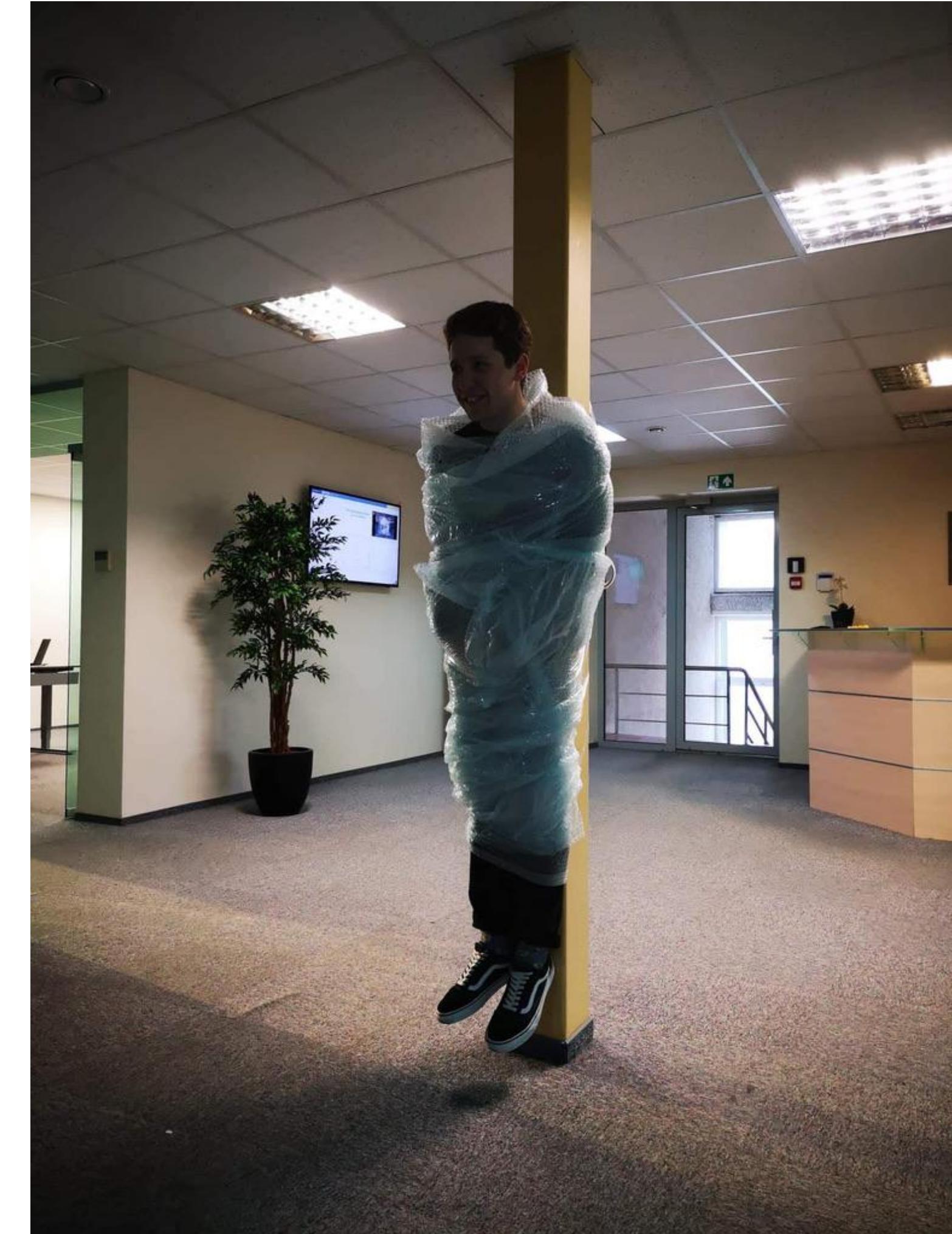
Agenda

- Localization
- Assets
- Objects & State
- Widgets

About me

- Software Engineer from Lithuania 🇲🇹
- Mobile Tech Lead @ Billo
- Google Developer Expert for Flutter & Dart ❤️
- Organiser @ Flutter Lithuania
- Using Flutter since v0.10.2

billo



What is Code Generation



Basic development flow

What is Code Generation

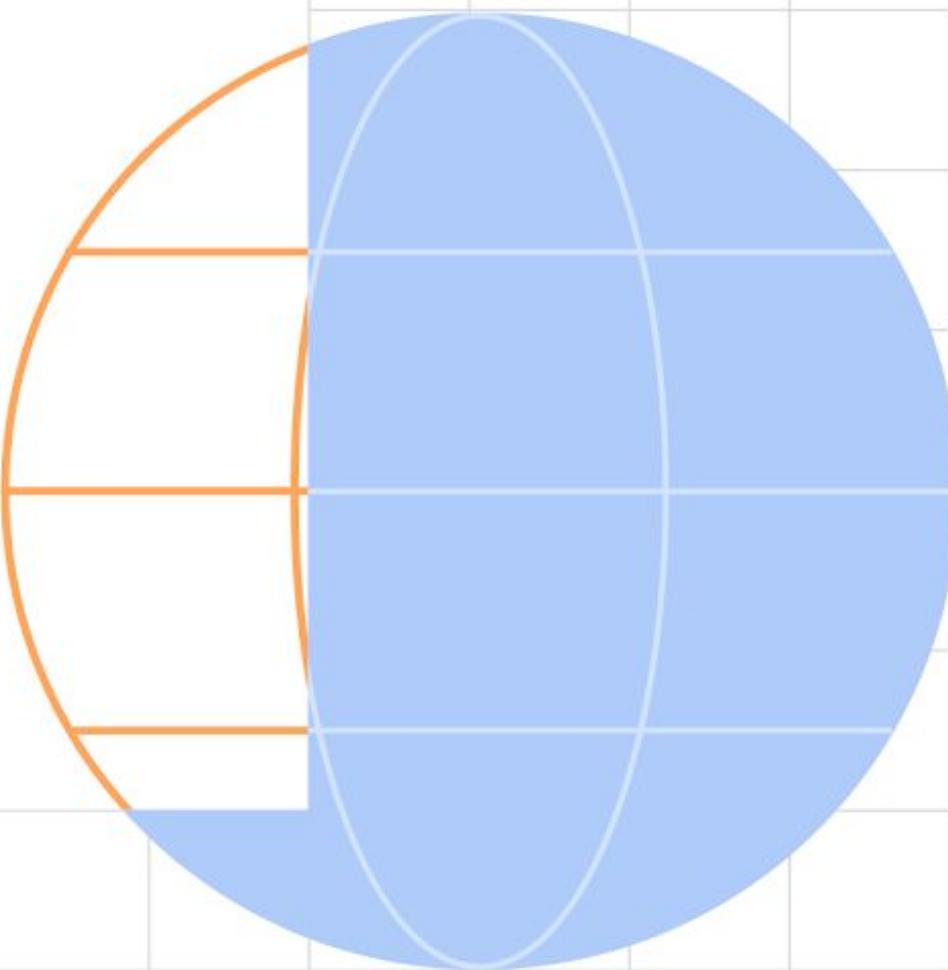


Basic development flow



Development flow with code generation

Localization



intl

Provides internationalization and localization facilities, including message translation, plurals and genders, date/number formatting and parsing, and bidirectional text.



intl 0.17.0

Published Feb 3, 2021 · [dart.dev](#) Null safety

DART NATIVE JS FLUTTER ANDROID IOS LINUX MACOS WEB WINDOWS

1.74K

[Readme](#) [Changelog](#) [Installing](#) [Versions](#) [Scores](#)

Provides internationalization and localization facilities, including message translation, plurals and genders, date/number formatting and parsing, and bidirectional text.

General

The most important library is `intl`. It defines the `Intl` class, with the default locale and methods for accessing most of the internationalization mechanisms. This library also defines the `DateFormat`, `NumberFormat`, and `BidiFormatter` classes.

Current locale



1736 LIKES 120 PUB POINTS 100% POPULARITY

Publisher

[dart.dev](#)

Metadata

Contains code to deal with internationalized/localized messages, date and number formatting and parsing, bi-directional text, and other internationalization issues.

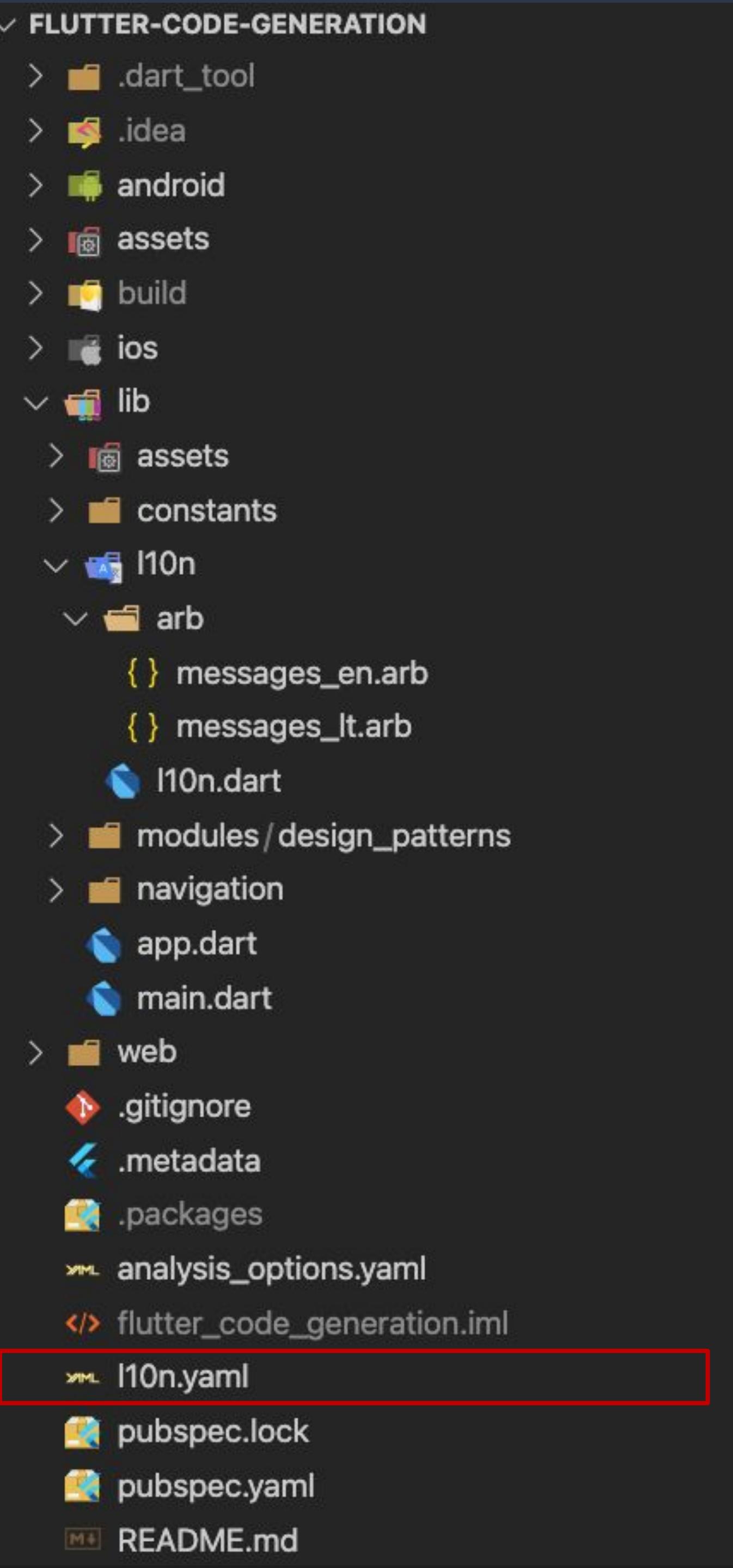
@mkobuolys

pubspec.yaml

```
1 name: flutter_code_generation
2 description: Examples of Flutter Code Generation.
3 version: 1.0.0+1
4
5 environment:
6   sdk: ">=2.12.0 <3.0.0"
7
8 dependencies:
9   flutter:
10     sdk: flutter
11   # Internationalization support.
12   flutter_localizations:
13     sdk: flutter
14   intl: ^0.17.0
15
16 flutter:
17   # Adds code generation (synthetic package) support
18   generate: true
```

l10n.yaml

```
1 arb-dir: lib/l10n/arb
2 template-arb-file: messages_en.arb
3 output-localization-file: app_localizations.dart
```

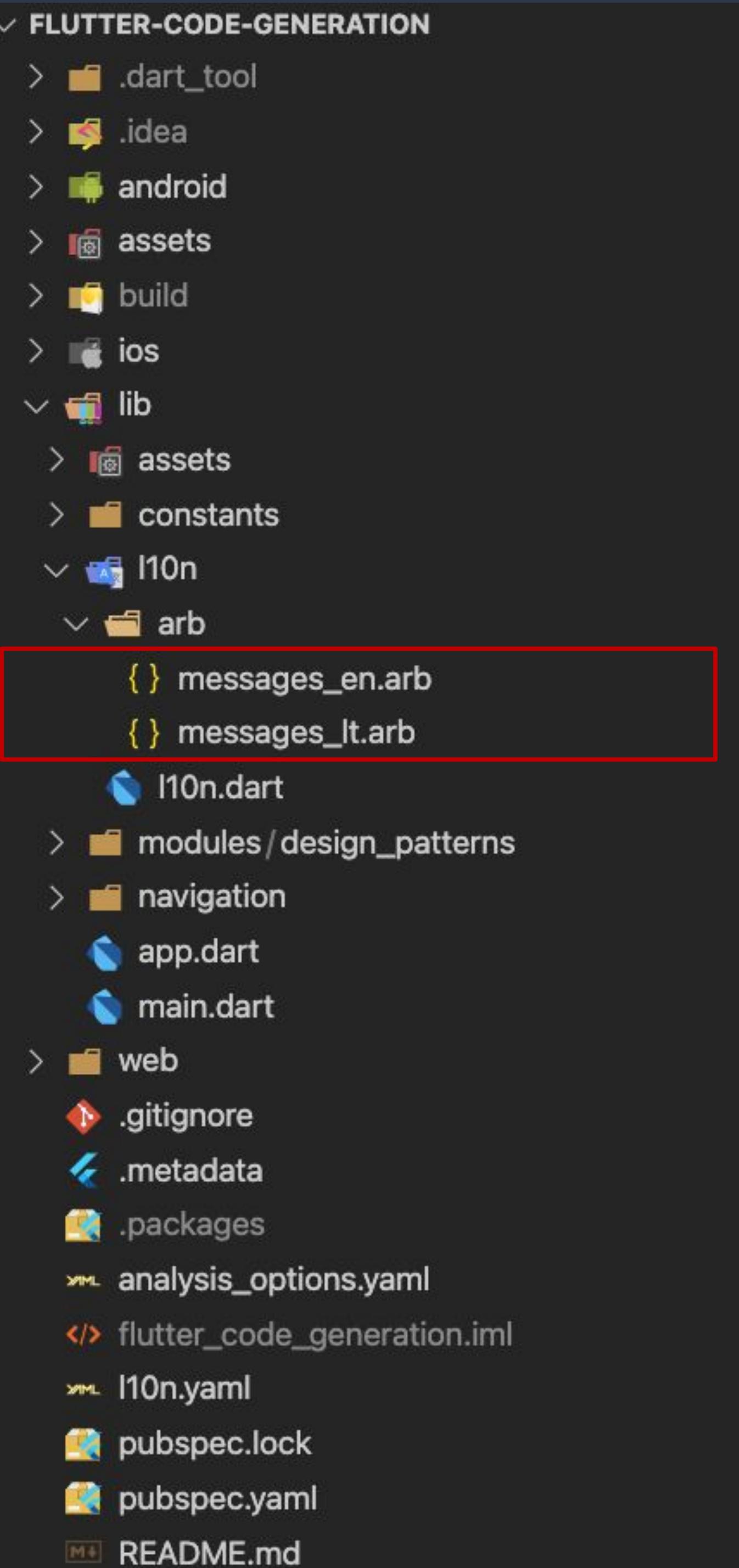


ARB (.arb) files

```
1  {
2      "@@locale": "en",
3      "appTitle": "Flutter Code Generation",
4      "@appTitle": {
5          "description": "Main menu page title."
6      },
7      "localizationExamplesTitle": "Localization Examples",
8      "@localizationExamplesTitle": {
9          "description": "Localization examples page title."
10     },
11     ...
12 }

1  {
2      "@@locale": "lt",
3      "appTitle": "Flutter Code Generation",
4      "localizationExamplesTitle": "Lokalizacijos pavyzdžiai",
5      "localizationExamplesLocaleText": "Lokalė: {locale}",
6      "localizationExamplesCurrentDateText": "Šiandienos data: {date}",
7      "localizationExamplesCurrencyText": "Kaina: {price}",
8      "localizationExamplesButtonPressedText": "{count, plural, =0 {Jūs dar nepaspaudėte mygtuko}
=1 {Jūs paspaudėte mygtuką vieną kartą} other {Jūs paspaudėte mygtuką tiek kartų: {count}}}",
9      ...
10 }
```

@mkobuolys



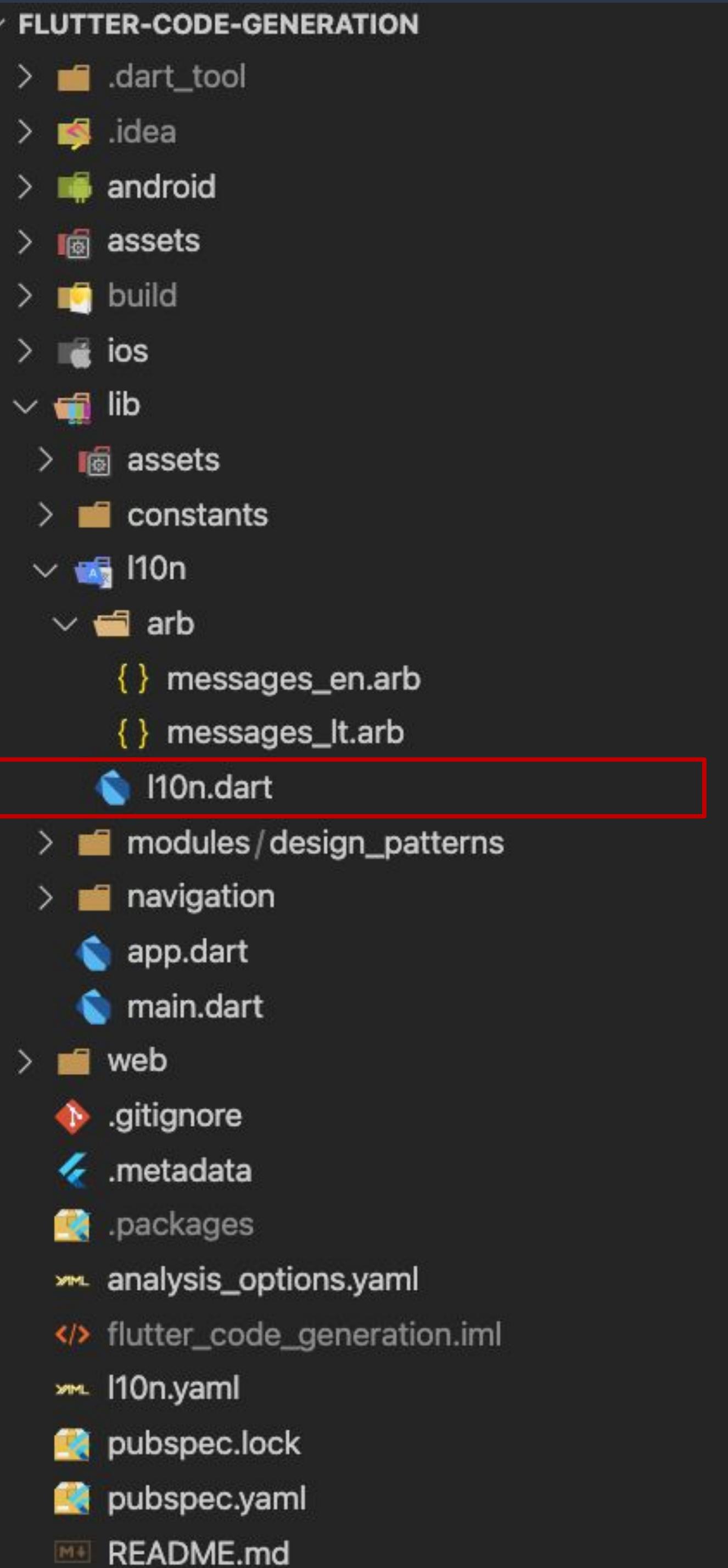
l10n.dart (optional)

```
1 import 'package:flutter/widgets.dart';
2
3 import 'package:flutter_gen/gen_l10n/app_localizations.dart';
4 export 'package:flutter_gen/gen_l10n/app_localizations.dart';
5
6 extension AppLocalizationsExtension on BuildContext {
7   AppLocalizations get l10n => AppLocalizations.of(this)!;
8 }
```

```
1 AppBar(
2   title: Text(context.l10n.appTitle), - w/ extension
3 ),
```

```
1 AppBar(
2   title: Text(AppLocalizations.of(context)!.appTitle), - w/o extension
3 ),
```

@mkobuolys



ARB -> Dart: Simple parameter

```
1 "localizationExamplesTitle": "Localization Examples",  
2 "@localizationExamplesTitle": {  
3   "description": "Localization examples page title."  
4 },  
5 "localizationExamplesLocaleText": "Locale: {locale}",  
6 "@localizationExamplesLocaleText": {  
7   "description": "Localization examples locale text.",  
8   "placeholders": {  
9     "locale": {  
10       "type": "String",  
11       "example": "en"  
12     }  
13   }  
14 },
```

```
1 /// The translations for English (`en`).  
2 class AppLocalizationsEn extends AppLocalizations {  
3   AppLocalizationsEn([String locale = 'en']) : super(locale);  
4  
5   @override  
6   String get localizationExamplesTitle => 'Localization Examples';  
7  
8   @override  
9   String localizationExamplesLocaleText(String locale) {  
10     return 'Locale: $locale';  
11   }  
12 }
```

ARB -> Dart: Date, currency

```
1 "localizationExamplesCurrentDateText": "Today's date: {date}",
2 "@localizationExamplesCurrentDateText": {
3   "description": "Localization examples current date and time text.",
4   "placeholders": {
5     "date": {
6       "type": "DateTime",
7       "format": "yMMMMd"
8     }
9   }
10 },
11 "localizationExamplesCurrencyText": "Price: {price}",
12 "@localizationExamplesCurrencyText": {
13   "description": "Localization examples current date and time text.",
14   "placeholders": {
15     "price": {
16       "type": "double",
17       "format": "simpleCurrency",
18       "optionalParameters": {
19         "decimalDigits": 2
20       }
21     }
22   }
23 },
```

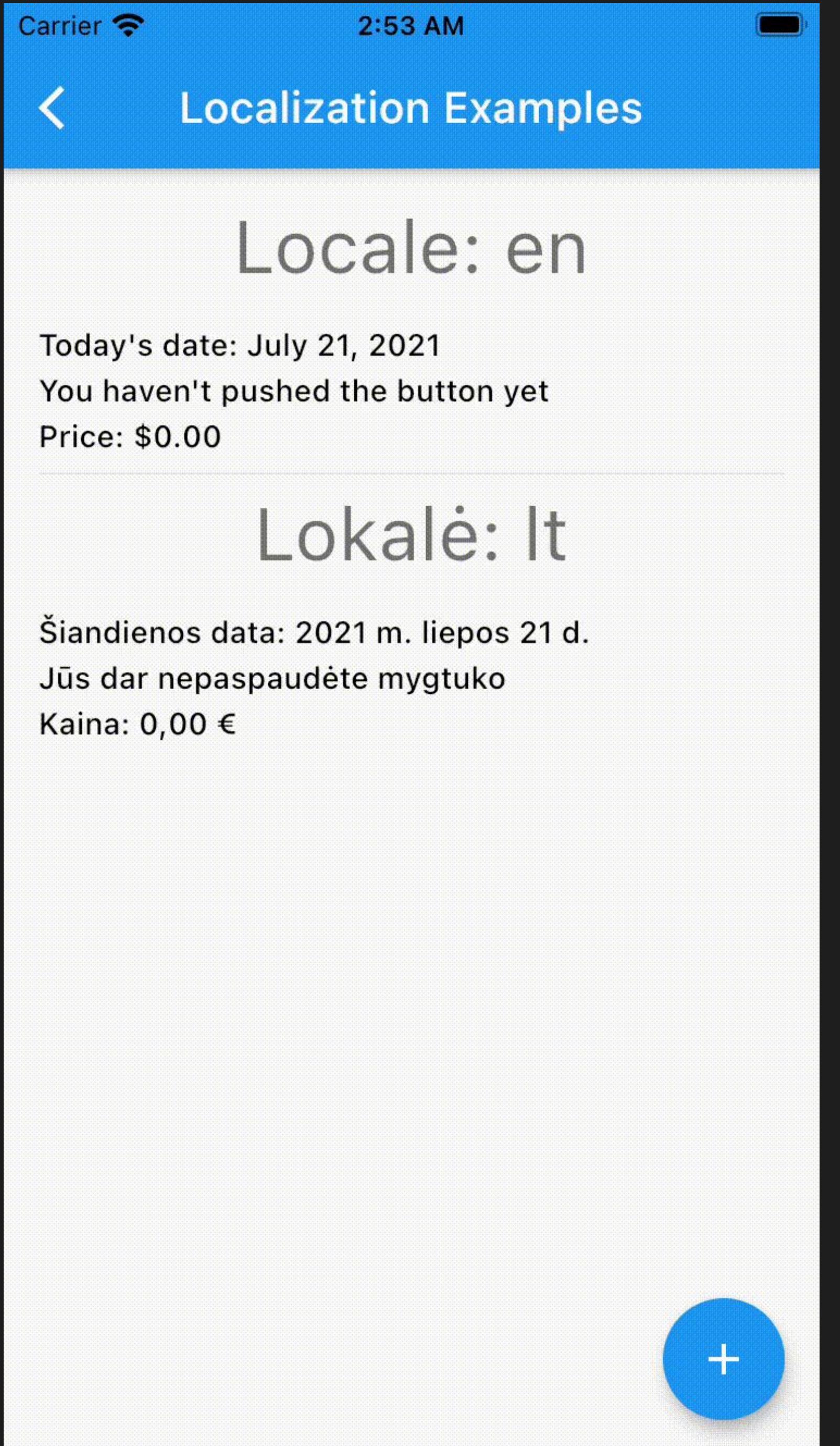
```
1 @override
2 String localizationExamplesCurrentDateText(DateTime date) {
3   final intl.DateFormat dateDateFormat = intl.DateFormat.yMMMMd(localeName);
4   final String dateString = dateDateFormat.format(date);
5
6   return 'Today\'s date: $dateString';
7 }
8
9 @override
10 String localizationExamplesCurrencyText(double price) {
11   final intl.NumberFormat priceNumberFormat = intl.NumberFormat.simpleCurrency(
12     locale: localeName,
13     decimalDigits: 2
14   );
15   final String priceString = priceNumberFormat.format(price);
16
17   return 'Price: $priceString';
18 }
```

ARB -> Dart: Plurals

```
1 "localizationExamplesButtonPressedText": "{count, plural, =0 {You haven't pushed  
the button yet} =1 {You have pushed the button once} other {You have pushed the  
button this many times: {count}}}",  
2 "@localizationExamplesButtonPressedText": {  
3   "description": "Localization examples button pressed example text.",  
4   "placeholders": {  
5     "count": {  
6       "type": "int",  
7       "format": "decimalPattern",  
8       "example": "10"  
9     }  
10    }  
11  },  
12  @override  
13  String localizationExamplesButtonPressedText(int count) {  
14    final intl.NumberFormat countNumberFormat = intl.NumberFormat.decimalPattern(localeName);  
15    final String countString = countNumberFormat.format(count);  
16  
17    return intl.Intl.pluralLogic(  
18      count,  
19      locale: localeName,  
20      zero: 'You haven\'t pushed the button yet',  
21      one: 'You have pushed the button once',  
22      other: 'You have pushed the button this many times: $count',  
23    );  
24  }
```

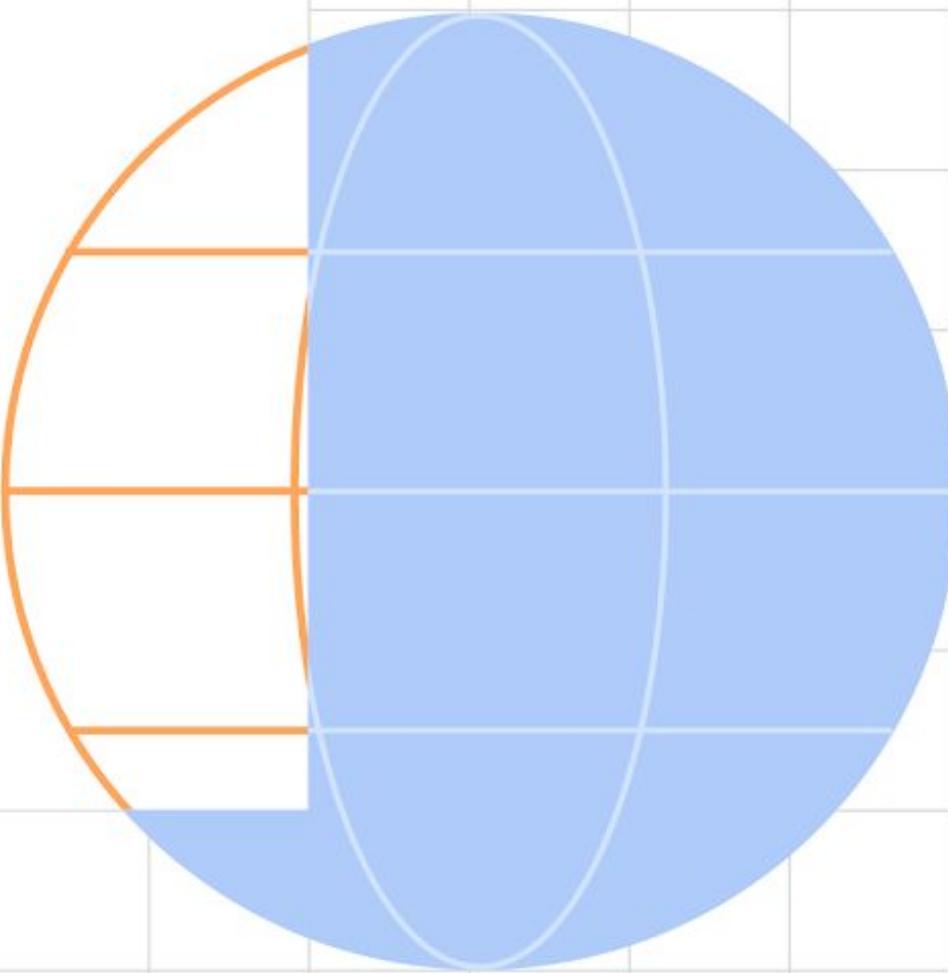
Usage

```
1 Column(  
2   crossAxisAlignment: CrossAxisAlignment.start,  
3   children: [  
4     Center(  
5       child: Text(  
6         context.l10n.localizationExamplesLocaleText(languageCode),  
7         style: Theme.of(context).textTheme.headline4,  
8       ),  
9     ),  
10    const SizedBox(height: LayoutConstants.spaceL),  
11    Text(  
12      context.l10n.localizationExamplesCurrentDateText(DateTime.now()),  
13      style: textStyle,  
14    ),  
15    const SizedBox(height: LayoutConstants.spaceS),  
16    Text(  
17      context.l10n.localizationExamplesButtonPressedText(counter),  
18      style: textStyle,  
19    ),  
20    const SizedBox(height: LayoutConstants.spaceS),  
21    Text(  
22      context.l10n.localizationExamplesCurrencyText(counter * 345.67),  
23      style: textStyle,  
24    ),  
25  ],  
26);
```





Assets



build_runner

Provides a concrete way of generating files using Dart code, outside of tools like pub.



build_runner 2.1.7

Published 30 days ago · [tools.dart.dev](#) Null safety

DART NATIVE

744

[Readme](#) [Changelog](#) [Installing](#) [Versions](#) [Scores](#)

Standalone generator and watcher for Dart using `package:build`.

[open "package: build_runner" issues](#) 38 [pub v2.1.7](#) [dartdocs latest](#) [chat on gitter](#)

The `build_runner` package provides a concrete way of generating files using Dart code, outside of tools like `pub`. Unlike `pub serve/build`, files are always generated directly on disk, and rebuilds are *incremental*-inspired by tools such as `Bazel`.

NOTE: Are you a user of this package? You may be interested in simplified user-facing documentation, such as our [getting started guide](#).

- [Installation](#)
- [Usage](#)

744 | 100 | 99%
LIKES PUB POINTS POPULARITY

Publisher

[tools.dart.dev](#)

Metadata

A build system for Dart code generation and modular compilation.

[Repository \(GitHub\)](#)

[View/report issues](#)

@mkobuolys

flutter_gen

The Flutter code generator
for your assets, fonts,
colors, ... - Get rid of all
String-based APIs.



flutter_gen 4.1.5

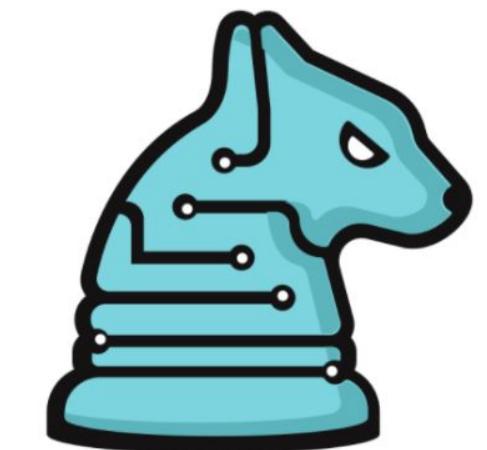
Published 3 days ago • wasabeef.jp Null safety

DART | NATIVE

1 308

Readme Changelog Example Installing Versions Scores

308 | 120 | 92%
LIKES PUB POINTS POPULARITY



FLUTTERGEN

pub v4.1.5 Dart CI passing codecov 99% style flutter_lints

The Flutter code generator for your assets, fonts, colors, ... — Get rid of all String-based APIs.

Publisher
wasabeef.jp

Metadata

The Flutter code generator
for your assets, fonts,
colors, ... — Get rid of all
String-based APIs.

[Repository \(GitHub\)](#)
[View/report issues](#)

Documentation

[Documentation](#)
[API reference](#)

License

[MIT \(LICENSE\)](#)

@mkobuolys

pubspec.yaml – dependencies/assets

```
1 dependencies:  
2   flutter:  
3     sdk: flutter  
4     flare_flutter: ^3.0.2  
5     flutter_svg: ^1.0.1  
6     rive: ^0.8.1  
7  
8   dev_dependencies:  
9     build_runner: ^2.1.7  
10    flutter_gen_runner: ^4.1.5
```

```
flutter:  
  uses-material-design: true  
  # Adds code generation (synthetic package) support  
  generate: true  
assets:  
  - assets/color/  
  - assets/flare/  
  - assets/images/  
  - assets/images/icons/  
  - assets/json/  
  - assets/rive/  
fonts:  
  - family: Fuggles  
    fonts:  
      - asset: assets/fonts/Fuggles-Regular.ttf
```

pubspec.yaml – flutter_gen

```
1 flutter_gen:
2   output: lib/assets/gen # Optional (default: lib/gen/)
3   line_length: 80 # Optional (default: 80)
4   null_safety: true # Optional (default: true)
5
6   integrations:
7     flutter_svg: true
8     flare_flutter: true
9     rive: true
10
11  colors:
12    enabled: true
13    inputs:
14      - assets/color/colors.xml
15
16  assets:
17    enabled: true
18    # Assets.imagesDash
19    # style: camel-case
20
21    # Assets.images_dash
22    # style: snake-case
23
24    # Assets.images.dash (default style)
25    # style: dot-delimiter
26
27  fonts:
28    enabled: true
```

```
1 class $AssetsImagesIconsGen {
2   const $AssetsImagesIconsGen();
3
4   String get githubLogo => 'assets/images/icons/github_logo.svg';
5   String get mediumLogo => 'assets/images/icons/medium_logo.svg';
6   String get twitterLogo => 'assets/images/icons/twitter_logo.svg';
7 }
8
9
10 }
```

- w/o integration

```
1 class $AssetsImagesIconsGen {
2   const $AssetsImagesIconsGen();
3
4   SvgGenImage get githubLogo =>
5       const SvgGenImage('assets/images/icons/github_logo.svg');
6   SvgGenImage get mediumLogo =>
7       const SvgGenImage('assets/images/icons/medium_logo.svg');
8   SvgGenImage get twitterLogo =>
9       const SvgGenImage('assets/images/icons/twitter_logo.svg');
10 }
```

- w/ integration

Generating assets

```
> flutter pub run build_runner build
```

FLUTTER-CODE-GENERATION

- > .dart_tool
- > .idea
- > android
- > assets
- > build
- > ios
- ✓ lib
- ✓ assets

- ✓ gen
 - assets.gen.dart
 - colors.gen.dart
 - fonts.gen.dart
- assets.dart

- constants
- i10n
- modules
- navigation
 - app.dart
 - main.dart

assets.gen.dart - Assets

```
1 class Assets {  
2     Assets._();  
3  
4     static const $AssetsColorGen color = $AssetsColorGen();  
5     static const $AssetsFlareGen flare = $AssetsFlareGen();  
6     static const $AssetsImagesGen images = $AssetsImagesGen();  
7     static const $AssetsJsonGen json = $AssetsJsonGen();  
8     static const $AssetsRiveGen rive = $AssetsRiveGen();  
9 }
```

assets.gen.dart - specific classes

```
1 class $AssetsColorGen {  
2   const $AssetsColorGen();  
3  
4   /// File path: assets/color/colors.xml  
5   String get colors => 'assets/color/colors.xml';  
6 }  
7  
8 class $AssetsFlareGen {  
9   const $AssetsFlareGen();  
10  
11  /// File path: assets/flare/penguin.flr  
12  FlareGenImage get penguin => const FlareGenImage('assets/flare/penguin.flr');  
13 }  
14  
15 class $AssetsImagesGen {  
16   const $AssetsImagesGen();  
17  
18   /// File path: assets/images/dash-1.png  
19   AssetGenImage get dash1 => const AssetGenImage('assets/images/dash-1.png');  
20  
21   /// File path: assets/images/dash-2.jpeg  
22   AssetGenImage get dash2 => const AssetGenImage('assets/images/dash-2.jpeg');  
23  
24   $AssetsImagesIconsGen get icons => const $AssetsImagesIconsGen();  
25 }
```

```
class $AssetsJsonGen {  
  const $AssetsJsonGen();  
  
  /// File path: assets/json/design_patterns.json  
  String get designPatterns => 'assets/json/design_patterns.json';  
}  
  
class $AssetsRiveGen {  
  const $AssetsRiveGen();  
  
  /// File path: assets/rive/vehicles.riv  
  RiveGenImage get vehicles => const RiveGenImage('assets/rive/vehicles.riv');  
}  
  
class $AssetsImagesIconsGen {  
  const $AssetsImagesIconsGen();  
  
  /// File path: assets/images/icons/github_logo.svg  
  SvgGenImage get githubLogo =>  
    const SvgGenImage('assets/images/icons/github_logo.svg');  
  
  /// File path: assets/images/icons/medium_logo.svg  
  SvgGenImage get mediumLogo =>  
    const SvgGenImage('assets/images/icons/medium_logo.svg');  
  
  /// File path: assets/images/icons/twitter_logo.svg  
  SvgGenImage get twitterLogo =>  
    const SvgGenImage('assets/images/icons/twitter_logo.svg');  
}
```

colors.gen.dart

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3   <color name="black">#000000</color>
4   <color name="gray_70">#EEEEEE</color>
5   <color name="gray_410">#979797</color>
6   <color name="crimson_red" type="material">#CF2A2A</color>
7   <color name="yellow_ocher" type="material material-accent">#DF9527</color>
8 </resources>
```

```
1 class ColorName {
2   ColorName._();
3
4   static const Color black = Color(0xFF000000);
5   static const MaterialColor crimsonRed = MaterialColor(
6     0xFFCF2A2A,
7     <int, Color>{
8       50: Color(0xFFFF9E5E5),
9       100: Color(0xFFF1BFBF),
10      200: Color(0xFFE79595),
11      300: Color(0xFFDD6A6A),
12      400: Color(0xFFD64A4A),
13      500: Color(0xFFCF2A2A),
14      600: Color(0xFFCA2525),
15      700: Color(0xFFC31F1F),
16      800: Color(0xFFBD1919),
17      900: Color(0xFFB20F0F),
18    },
19  );
20  static const Color gray410 = Color(0xFF979797);
21  static const Color gray70 = Color(0xFFFFFFFF);
22  static const MaterialColor yellowOcher = MaterialColor(
23    0xFFDF9527,
24    <int, Color>{
25      50: Color(0xFFFFBF2E5),
26      100: Color(0xFFF5DFBE),
27      200: Color(0xFFEFC9A3),
28      300: Color(0xFFE9B568),
29      400: Color(0FFE4A547),
30      500: Color(0xFFDF9527),
31      600: Color(0xFFDB8D23),
32      700: Color(0xFFD7821D),
33      800: Color(0xFFD27817),
34      900: Color(0xFFCA670E),
35    },
36  );
37  static const MaterialAccentColor yellowOcherAccent = MaterialAccentColor(
38    0xFFFFBCA3,
39    <int, Color>{
40      100: Color(0xFFFFE8E0),
41      200: Color(0xFFFFBCA3),
42      400: Color(0xFFFFA989),
43      700: Color(0xFFFF9E7A),
44    },
45  );
46 }
```

fonts.gen.dart

```
1  class FontFamily {  
2    FontFamily._();  
3  
4    /// Font family: Fuggles  
5    static const String fuggles = 'Fuggles';  
6 }
```

Usage: Icons/images

```

1 Row(
2   mainAxisAlignment: MainAxisAlignment.center,
3   children: [
4     Assets.images.icons.githubLogo.svg(width: _iconWidth),
5     const SizedBox(width: LayoutConstants.spaceM),
6     Assets.images.icons.mediumLogo.svg(width: _iconWidth),
7     const SizedBox(width: LayoutConstants.spaceM),
8     Assets.images.icons.twitterLogo.svg(width: _iconWidth),
9   ],
10 );

```

```

1 Row(
2   mainAxisAlignment: MainAxisAlignment.spaceAround,
3   children: [
4     Assets.images.dash1.image(
5       height: _imageHeight,
6       width: _imageWidth,
7     ),
8     Assets.images.dash2.image(
9       height: _imageHeight,
10      width: _imageWidth,
11    ),
12  ],
13 );

```

@mkobuolys

< Assets Examples

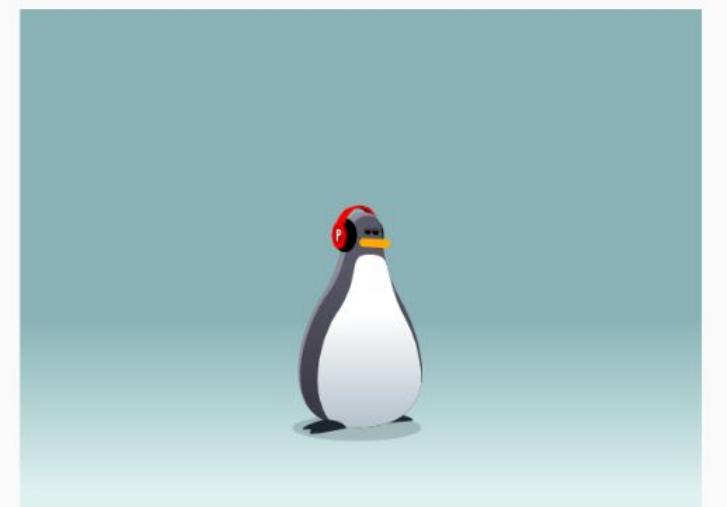
Icons



Images



Flare



Rive



Usage: Flare/Rive/fonts

```

1 Center(
2   child: SizedBox(
3     height: 200.0,
4     width: 200.0,
5     child: Assets.flare.penguin.flare(
6       animation: 'walk',
7     ),
8   ),
9 );
1 Center(
2   child: SizedBox(
3     height: 150.0,
4     width: 200.0,
5     child: Assets.rive.vehicles.rive(),
6   ),
7 );
1 Text(
2   context.l10n.assetsExamplesFontsHelloWorldText,
3   style: Theme.of(context)
4     .textTheme
5     .headline2!
6     .copyWith(fontFamily: FontFamily.fuggles),
7 );

```

@mkobuolys



Assets Examples

Flare



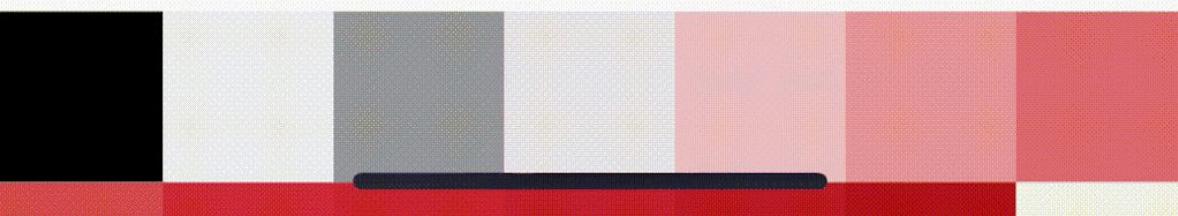
Rive



Fonts

Hello, World!

Colors



Usage: Colors

```

1  Wrap(
2    children: [
3      const _ColorCube(color: ColorName.black),
4      const _ColorCube(color: ColorName.gray70),
5      const _ColorCube(color: ColorName.gray410),
6      _ColorCube(color: ColorName.crimsonRed.shade50),
7      _ColorCube(color: ColorName.crimsonRed.shade100),
8      _ColorCube(color: ColorName.crimsonRed.shade200),
9      _ColorCube(color: ColorName.crimsonRed.shade300),
10     _ColorCube(color: ColorName.crimsonRed.shade400),
11     _ColorCube(color: ColorName.crimsonRed.shade500),
12     _ColorCube(color: ColorName.crimsonRed.shade600),
13     _ColorCube(color: ColorName.crimsonRed.shade700),
14     _ColorCube(color: ColorName.crimsonRed.shade800),
15     _ColorCube(color: ColorName.crimsonRed.shade900),
16     _ColorCube(color: ColorName.yellowOcher.shade50),
17     _ColorCube(color: ColorName.yellowOcher.shade100),
18     _ColorCube(color: ColorName.yellowOcher.shade200),
19     _ColorCube(color: ColorName.yellowOcher.shade300),
20     _ColorCube(color: ColorName.yellowOcher.shade400),
21     _ColorCube(color: ColorName.yellowOcher.shade500),
22     _ColorCube(color: ColorName.yellowOcher.shade600),
23     _ColorCube(color: ColorName.yellowOcher.shade700),
24     _ColorCube(color: ColorName.yellowOcher.shade800),
25     _ColorCube(color: ColorName.yellowOcher.shade900),
26     _ColorCube(color: ColorName.yellowOcherAccent.shade100),
27     _ColorCube(color: ColorName.yellowOcherAccent.shade200),
28     _ColorCube(color: ColorName.yellowOcherAccent.shade400),
29     _ColorCube(color: ColorName.yellowOcherAccent.shade700),
30   ],
31 );

```



Assets Examples



Rive



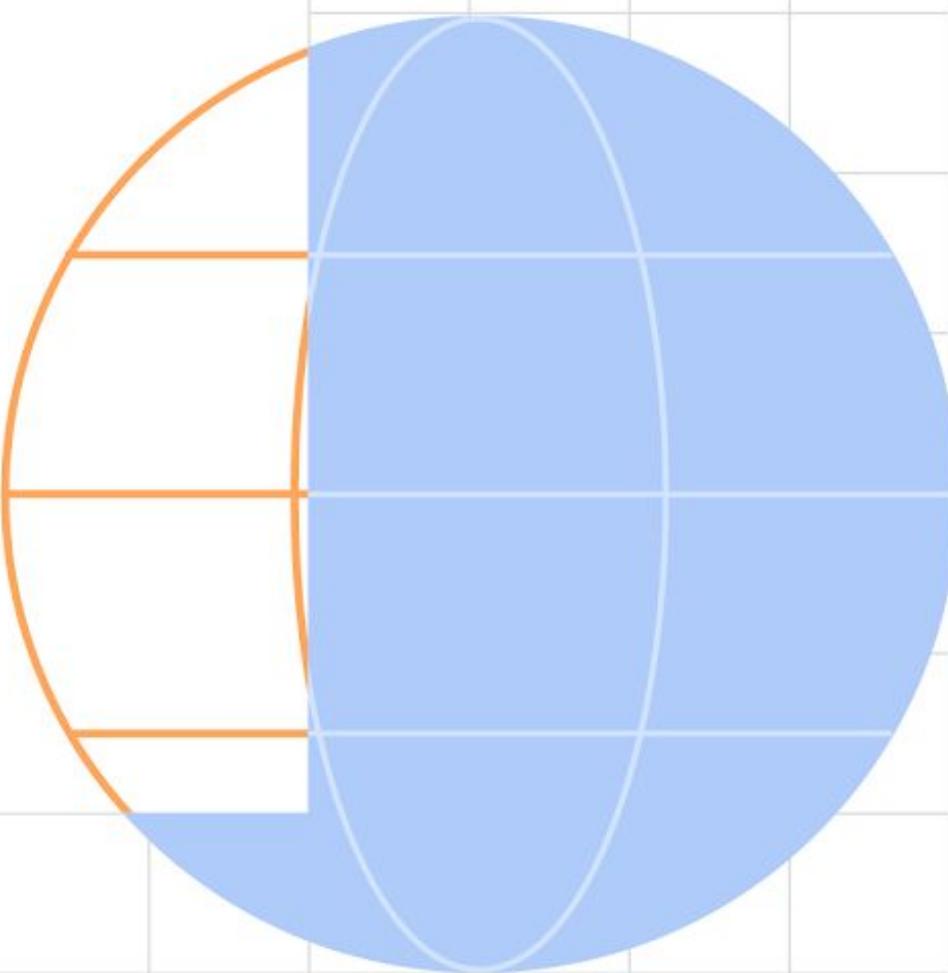
Fonts

Hello, World!

Colors



Objects & State



freezed

“... yet another code generator for unions/pattern-matching/copy.”



The screenshot shows the package page for 'freezed 1.1.1'. At the top right is a 'Flutter Favorite' badge with a blue checkmark and the text 'Flutter Favorite'. Below it are statistics: 1355 LIKES, 120 PUB POINTS, and 98% POPULARITY. The main heading is 'freezed 1.1.1'. Below it, the text 'Published 17 days ago' and 'dash-overflow.net Null safety' are shown. A navigation bar includes 'Readme' (which is underlined), 'Changelog', 'Example', 'Installing', 'Versions', and 'Scores'. Below the navigation bar are status indicators: 'Build passing', 'pub v1.1.1', and 'chat 117 online'. A large circular 'Flutter Favorite' logo is centered below the stats. The text 'Welcome to Freezed, yet another code generator for unions/pattern-matching/copy.' is at the bottom.



Flutter
Favorite



@mkobuolys

freezed_annotation

Annotations for freezed
package.



freezed_annotation 1.1.0

Published 35 days ago · dash-overflow.net Null safety

DART NATIVE JS FLUTTER ANDROID IOS LINUX MACOS WEB WINDOWS

137

Readme Changelog Installing Versions Scores

Annotations for [freezed](#).

This package does nothing without [freezed](#).

137 | 120 | 99%
LIKES PUB POINTS POPULARITY

Publisher

dash-overflow.net

Metadata

Annotations for the
freezed code-generator.
This package does
nothing without freezed
too.

Repository (GitHub)

[View/report issues](#)

@mkobuolys

json_serializable

Automatically generate code for converting to and from JSON by annotating Dart classes.



json_serializable 6.1.4

Published 2 days ago · [google.dev](#) Null safety

DART NATIVE

1.63K

[Readme](#) [Changelog](#) [Example](#) [Installing](#) [Versions](#) [Scores](#)

pub v6.1.4

Provides [Dart Build System](#) builders for handling JSON.

The builders generate code when they find members annotated with classes defined in `package:json_annotation`.

- To generate to/from JSON code for a class, annotate it with `JsonSerializable`. You can provide arguments to `JsonSerializable` to configure the generated code. You can also customize individual fields by annotating them with `JsonKey` and providing custom arguments. See the table below for details on the [annotation values](#).
- To generate a Dart field with the contents of a file containing JSON, use the `JsonLiteral` annotation.



Flutter
Favorite

1632 LIKES | 120 PUB POINTS | 99% POPULARITY

Publisher

[google.dev](#)

Metadata

Automatically generate code for converting to and from JSON by annotating Dart classes.

[Repository \(GitHub\)](#)

[View/report issues](#)

@mkobuolys

pubspec.yaml

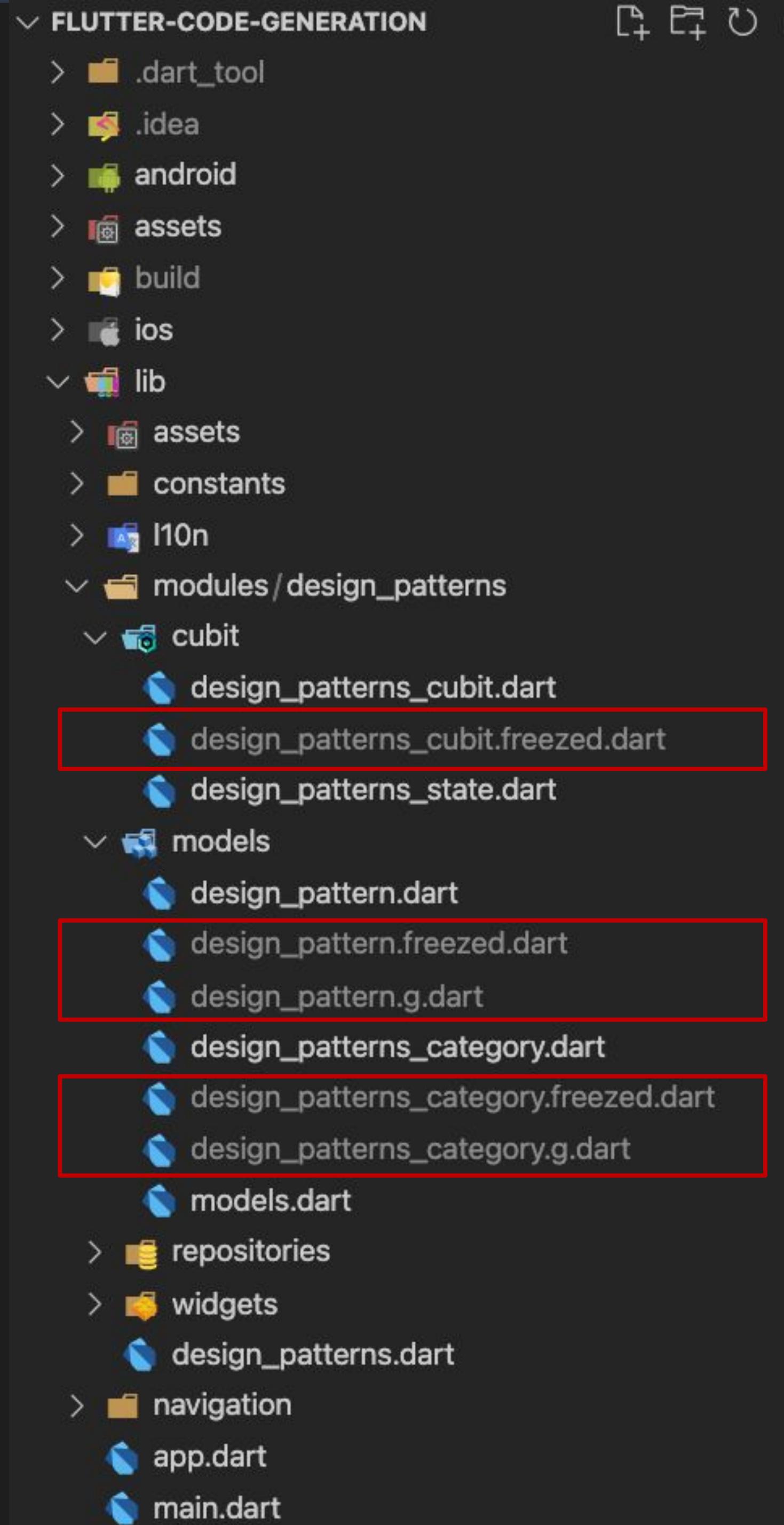
```
1 dependencies:  
2   flutter:  
3     sdk: flutter  
4     freezed_annotation: ^1.1.0  
5  
6 dev_dependencies:  
7   build_runner: ^2.1.7  
8   freezed: ^1.1.1  
9   json_serializable: ^6.1.4
```

Generating files

```
> flutter pub run build_runner build
```

or

```
> flutter pub run build_runner watch
```



Freezed class

```
1 part 'design_patterns_category.freezed.dart';
2 part 'design_patterns_category.g.dart';
3
4 @freezed
5 class DesignPatternsCategory with _$DesignPatternsCategory {
6   const factory DesignPatternsCategory({
7     required String id,
8     required String title,
9     required String color,
10    required List<DesignPattern> patterns,
11  }) = _DesignPatternsCategory;
12
13  factory DesignPatternsCategory.fromJson(Map<String, dynamic> json) =>
14    _$DesignPatternsCategoryFromJson(json);
15 }
```

Generated *.g.dart

```
1  _$DesignPatternsCategory _$$_DesignPatternsCategoryFromJson(
2      Map<String, dynamic> json) =>
3      _$DesignPatternsCategory(
4          id: json['id'] as String,
5          title: json['title'] as String,
6          color: json['color'] as String,
7          patterns: (json['patterns'] as List<dynamic>)
8              .map((e) => DesignPattern.fromJson(e as Map<String, dynamic>))
9              .toList(),
10     );
11
12 Map<String, dynamic> _$$_DesignPatternsCategoryToJson(
13     _$DesignPatternsCategory instance) =>
14     <String, dynamic>{
15         'id': instance.id,
16         'title': instance.title,
17         'color': instance.color,
18         'patterns': instance.patterns,
19     };

```

Generated *.freezed.dart

```
1  ...
2  @override
3  bool operator ==(dynamic other) {
4      return identical(this, other) ||
5          (other.runtimeType == runtimeType &&
6              other is _DesignPatternsCategory &&
7                  const DeepCollectionEquality().equals(other.id, id) &&
8                  const DeepCollectionEquality().equals(other.title, title) &&
9                  const DeepCollectionEquality().equals(other.color, color) &&
10                 const DeepCollectionEquality().equals(other.patterns, patterns));
11 }
12
13 @override
14 int get hashCode => Object.hash(
15     runtimeType,
16     const DeepCollectionEquality().hash(id),
17     const DeepCollectionEquality().hash(title),
18     const DeepCollectionEquality().hash(color),
19     const DeepCollectionEquality().hash(patterns));
20
21 @JsonKey(ignore: true)
22 @override
23 _$DesignPatternsCategoryCopyWith<_DesignPatternsCategory> get copyWith =>
24     _$DesignPatternsCategoryCopyWithImpl<_DesignPatternsCategory>(
25         this, _identity);
26
27 @override
28 Map<String, dynamic> toJson() {
29     return _$$_DesignPatternsCategoryToJson(this);
30 }
31 }
```

State management: State

```
1 abstract class DesignPatternsState extends Equatable {  
2   const DesignPatternsState();  
3  
4   @override  
5   List<Object> get props => [];  
6 }  
7  
8 class DesignPatternsInitial extends DesignPatternsState {}  
9  
10 class DesignPatternsLoadInProgress extends DesignPatternsState {}  
11  
12 class DesignPatternsLoadSuccess extends DesignPatternsState {  
13   final List<DesignPatternsCategory> categories;  
14  
15   const DesignPatternsLoadSuccess([this.categories = const []]);  
16  
17   @override  
18   List<Object> get props => [categories];  
19  
20   @override  
21   String toString() => 'DesignPatternsLoadSuccess { categories: $categories }';  
22 }  
23  
24 class DesignPatternsLoadFailure extends DesignPatternsState {}  
  
1 @freezed  
2 class DesignPatternsState with _$DesignPatternsState {  
3   const factory DesignPatternsState.initial() =  
4     _DesignPatternsStateInitial;  
5   const factory DesignPatternsState.loadInProgress() =  
6     _DesignPatternsStateLoadInProgress;  
7   const factory DesignPatternsState.loadSuccess({  
8     required List<DesignPatternsCategory> categories,  
9   }) = _DesignPatternsStateLoadSuccess;  
10  const factory DesignPatternsState.loadFailure() =  
11    _DesignPatternsStateLoadFailure;  
12 }
```

State management: Cubit

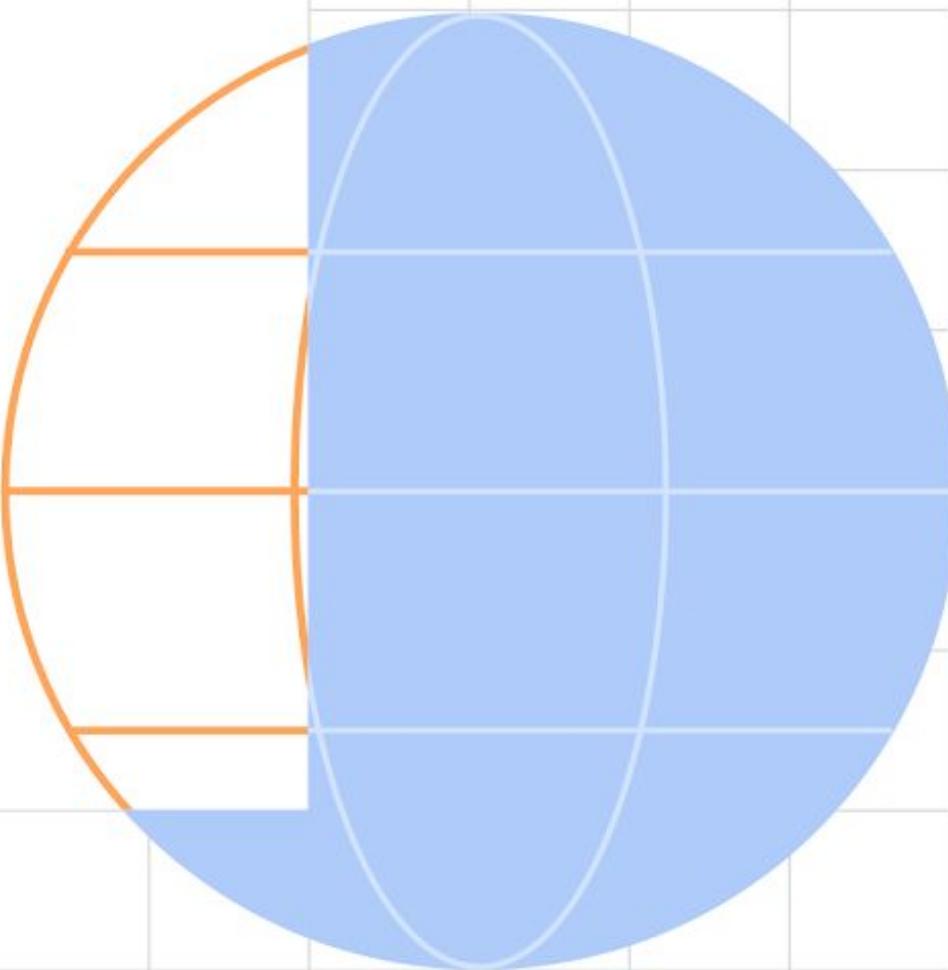
```
1  class DesignPatternsCubit extends Cubit<DesignPatternsState> {
2      final DesignPatternsRepository repository;
3
4      DesignPatternsCubit({
5          required this.repository,
6      }) : super(const DesignPatternsState.initial());
7
8      Future<void> load() async {
9          emit(const DesignPatternsState.loadInProgress());
10
11         try {
12             final categories = await repository.getCategories();
13
14             emit(DesignPatternsState.loadSuccess(categories: categories));
15         } on DesignPatternsJsonNotFoundException {
16             emit(const DesignPatternsState.loadFailure());
17         }
18     }
19 }
```

State management: Usage

```
1  @override
2  Widget build(BuildContext context) {
3      return BlocBuilder<DesignPatternsCubit, DesignPatternsState>(
4          builder: (_, state) {
5              if (state is DesignPatternsStateLoadInProgress) {
6                  return const _LoaderView();
7              } else if (state is DesignPatternsStateLoadFailure) {
8                  return const _ErrorView();
9              } else if (state is DesignPatternsStateLoadSuccess) {
10                  return DesignPatternsListView(
11                      categories: state.categories,
12                  );
13              } else {
14                  return const SizedBox();
15              }
16          },
17      );
18  }

@override
Widget build(BuildContext context) {
    return BlocBuilder<DesignPatternsCubit, DesignPatternsState>(
        builder: (_, state) => state.maybeWhen(
            loadInProgress: () => const _LoaderView(),
            loadFailure: () => const _ErrorView(),
            loadSuccess: (categories) => DesignPatternsListView(
                categories: categories,
            ),
            orElse: () => const SizedBox(),
        ),
    );
}
```

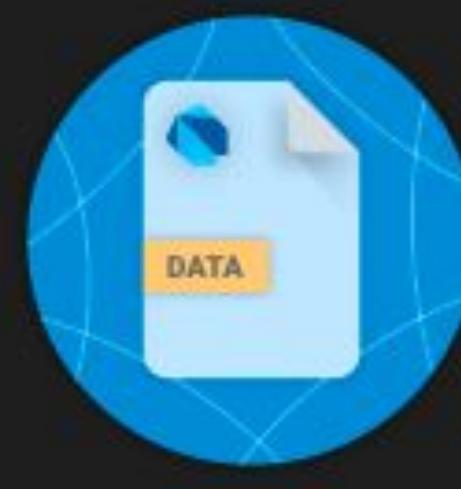
Widgets



IDE Extensions



Awesome Flutter Snippets v3.0.2
Neevash Ramdial | ⚡ 745,678 | ★★★★★(11)
Awesome Flutter Snippets is a collection snippets and shortcuts for commonly used Flutter functions and classes
[Disable](#) [Uninstall](#) [⚙️](#)
This extension is enabled globally.



Dart Data Class Generator v0.5.5
BendixMa | ⚡ 138,378 | ★★★★★(19)
Create dart data classes easily, fast and without writing boilerplate or running code generation.
[Disable](#) [Uninstall](#) [⚙️](#)
This extension is enabled globally.



bloc v6.4.0
Felix Angelov | ⚡ 156,711 | ★★★★★(10)
Support for the bloc library and provides tools for effectively creating blocs for both Flutter and AngularDart apps.
[Disable](#) [Uninstall](#) [⚙️](#)
This extension is enabled globally.

Awesome Flutter Snippets

A screenshot of a code editor showing a tooltip for the word 'state'. The tooltip provides information about stateful and stateless widgets and includes a sample code snippet for creating a StatelessWidget.

```
lib > snippets.dart > state
2 state    Variables must be declared using the keywords 'const', 'final', 'var' or a type name. Try adding the na
  └─ statefulBldr      StatefulWidget
  └─ statefulW        StatefulWidget
  └─ statelessW       StatelessWidget
    └─ State
    └─ State
    └─ State
    └─ StateError
    └─ StateManagementExamplesPage
    └─ StateManagementExamplesPage
    └─ StatefulWidget
    └─ StatefulWidget
    └─ StatefulWidget
Create a Stateless widget (Awesome Flutte ×
r Snippets)

class name extends StatelessWidget {
  const name({Key? key}) : super(key: key);

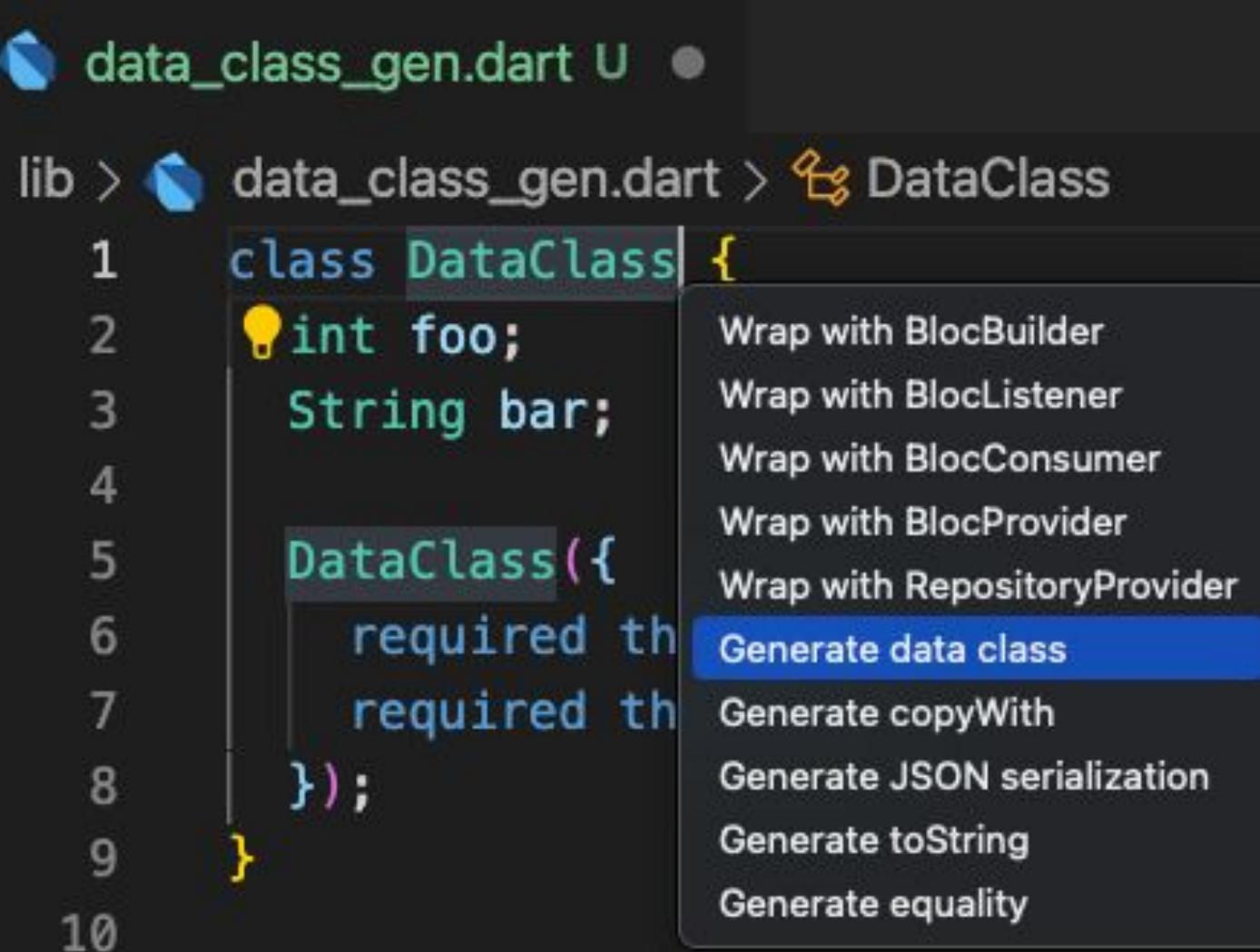
  @override
  Widget build(BuildContext context) {
    return Container(
      child: null,
    );
}
```

snippets.dart U ●

A screenshot of a code editor showing a tooltip for the class 'MyStatelessWidget'. The tooltip provides a sample code snippet for defining a StatelessWidget.

```
lib > snippets.dart > MyStatelessWid
1 import 'package:flutter/material.dart';
2
3 class MyStatelessWid extends StatelessWidget {
4   const MyStatelessWid({Key? key}) : super(key: key);
5
6   @override
7   Widget build(BuildContext context) {
8     return Container();
9   }
10
11 }
```

Dart Data Class Generator



```
1  class DataClass {  
2    int foo;  
3    String bar;  
4  
5    DataClass({  
6      required this.foo,  
7      required this.bar,  
8    });  
9  
10   DataClass copyWith({  
11     int? foo,  
12     String? bar,  
13   }) {  
14     return DataClass(  
15       foo: foo ?? this.foo,  
16       bar: bar ?? this.bar,  
17     );  
18   }  
19  
20   Map<String, dynamic> toMap() {  
21     return {  
22       'foo': foo,  
23       'bar': bar,  
24     };  
25   }  
26  
27   factory DataClass.fromMap(Map<String, dynamic> map) {  
28     return DataClass(  
29       foo: map['foo'],  
30       bar: map['bar'],  
31     );  
32   }  
33   ...  
34 }
```

```
1  ...  
2  String toJson() => json.encode(toMap());  
3  
4  factory DataClass.fromJson(String source) =>  
5    DataClass.fromMap(json.decode(source));  
6  
7  @override  
8  String toString() => 'DataClass(foo: $foo, bar: $bar)';  
9  
10 @override  
11 bool operator ==(Object other) {  
12   if (identical(this, other)) return true;  
13  
14   return other is DataClass && other.foo == foo && other.bar == bar;  
15 }  
16  
17 @override  
18 int get hashCode => foo.hashCode ^ bar.hashCode;  
19 }
```

AI-powered tools

Technical preview

Your AI pair programmer

The screenshot shows a dark-themed code editor interface. At the top, there's a navigation bar with tabs: 'fetch_pic.js' (selected), 'push_to_git.py', 'JS d3_scale.js', 'JS fetch_stock.js', and 'JS material_ui.js'. Below the tabs, a code editor displays the following JavaScript code:

```
1 const fetchNASAPictureOfDay = () => {
2   return fetch('https://api.nasa.gov/planetary/apod?api_key=DEMO_KEY', {
3     method: 'GET',
4     headers: {
5       'Content-Type': 'application/json',
6     },
7   })
8   .then(response => response.json())
9   .then(json => {
10     return json;
11   });
12 }
```

A small blue button labeled 'Copilot' is visible at the bottom left of the code editor. Below the code editor, the GitHub Copilot logo (a white icon of two people) is followed by the text 'GitHub Copilot'.

Too smart?

The image shows a code editor interface with two panes. The left pane displays a Dart file named 'copilot.dart' with the following code:

```
lib > copilot.dart > Copilot > build
1 import 'package:flutter/material.dart';
2
3 class Copilot extends StatelessWidget {
4     const Copilot();
5
6     @override
7     Widget build(BuildContext context) {    The body might compl
8         // Create a Center widget with Flutter Logo as its child.
9     }
10}
11
12
```

A red squiggle underline is under the word 'build' in the 'build' method. A tooltip above the underline reads: 'The body might compl'. The right pane shows the 'Copilot' interface with the following text:

```
...
1 Copilot X ...
1 Synthesizing 10/10 solutions
2 =====
3
4 Accept Solution
5 return new Center(
6     new Center(
7         new MaterialFlutterLogo(),
8         new Center(
9             new Center(
10                new Center(
11                    new Center(
12                        new Center(
13                            new Center(
14                                new Center(
15                                    new Center(
16                                        new Center(
17                                            new Center(
18                                                new Center(
19                                                    new Center(
20                                                        new Ce
```

The text 'Accept Solution' is visible at the top of the Copilot window.

mason

A Dart template generator which helps teams generate files quickly and consistently.



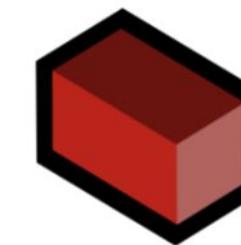
mason 0.1.0-dev.3

Published 4 days ago · brickhub.dev Null safety

DART NATIVE JS FLUTTER ANDROID IOS LINUX MACOS WEB WINDOWS

126

Readme Changelog Example Installing Versions Scores



mason

pub v0.1.0-dev.3 passing coverage 100% style very good analysis license MIT

A Dart template generator which helps teams generate files quickly and consistently.

126 | 130
LIKES PUB POINTS

83%
POPULARITY

Publisher

brickhub.dev

Metadata

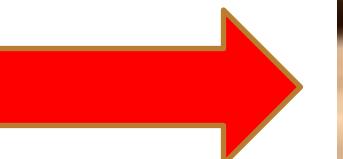
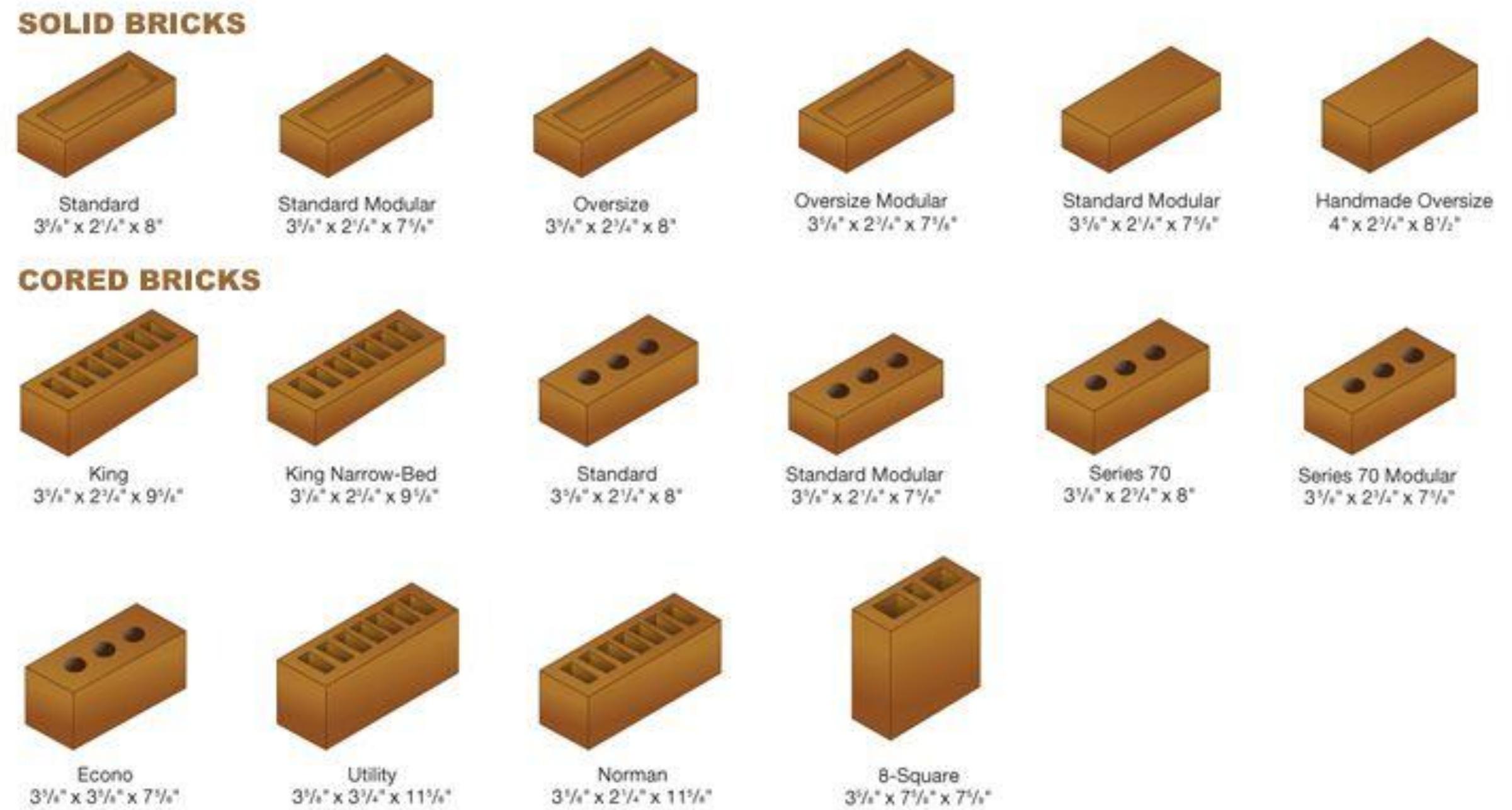
A Dart template generator which helps teams generate files quickly and consistently.

Repository (GitHub)

[View/report issues](#)

@mkobuolys

Mason: Concept



Mason: Expected result

```
1 import 'package:flutter/material.dart';
2
3 class CustomPage extends StatelessWidget {
4   static PageRoute route() {
5     return MaterialPageRoute(builder: (context) => const CustomPage());
6   }
7
8   const CustomPage();
9
10  @override
11  Widget build(BuildContext context) {
12    return const _CustomPageView();
13  }
14}
15
16 class _CustomPageView extends StatelessWidget {
17   const _CustomPageView();
18
19   @override
20   Widget build(BuildContext context) {
21     return Container();
22   }
23 }
```

Mason: Expected result

```
1 import 'package:flutter/material.dart';
2
3 class CustomPage extends StatelessWidget {
4     static PageRoute route() {
5         return MaterialPageRoute(builder: (context) => const CustomPage());
6     }
7
8     const CustomPage();
9
10    @override
11    Widget build(BuildContext context) {
12        return const _CustomPageView();
13    }
14}
15
16 class _CustomPageView extends StatelessWidget {
17     const _CustomPageView();
18
19    @override
20    Widget build(BuildContext context) {
21        return Container();
22    }
23}
```

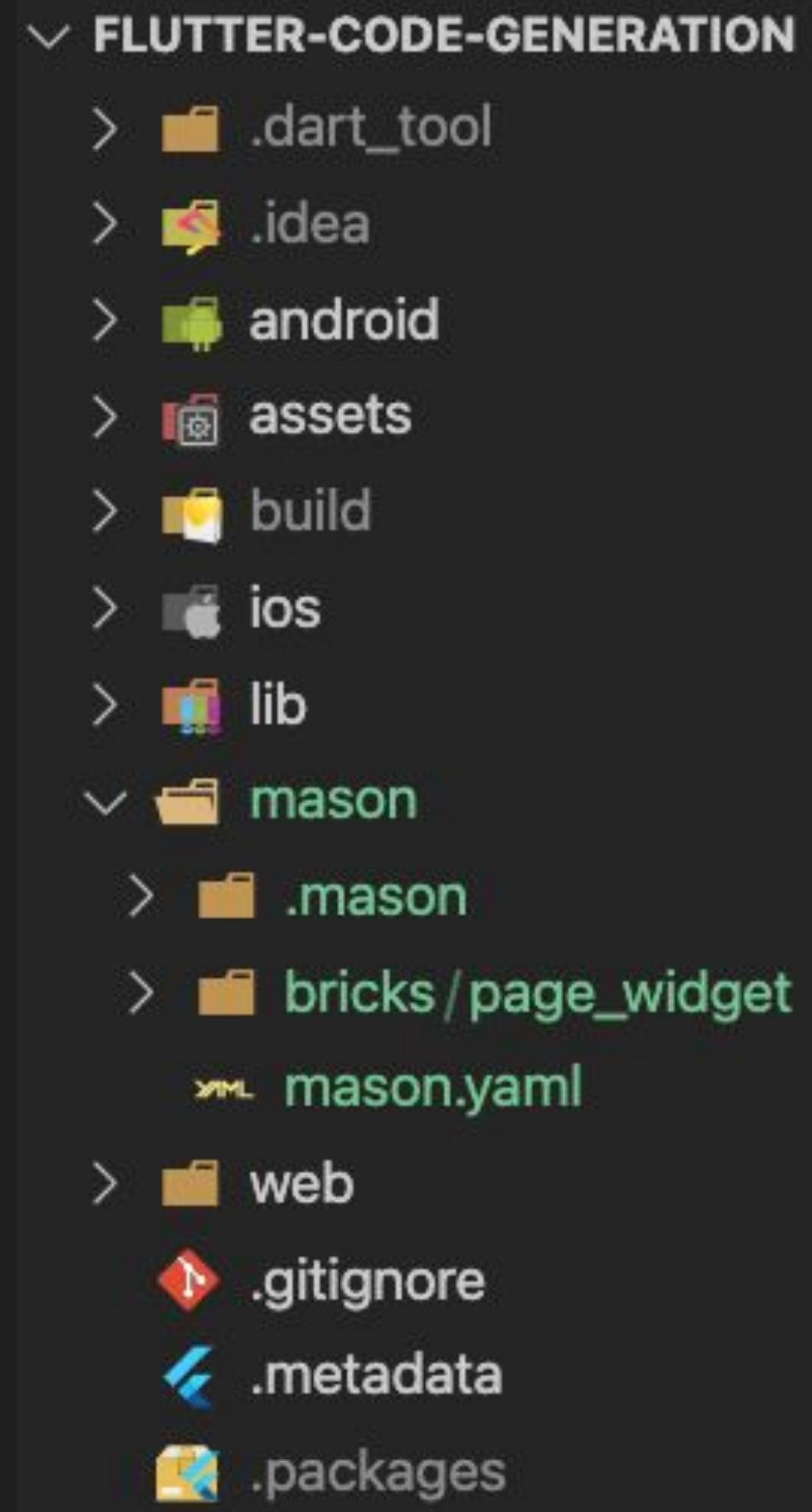
Any name

Optional

Mason: Initial configuration

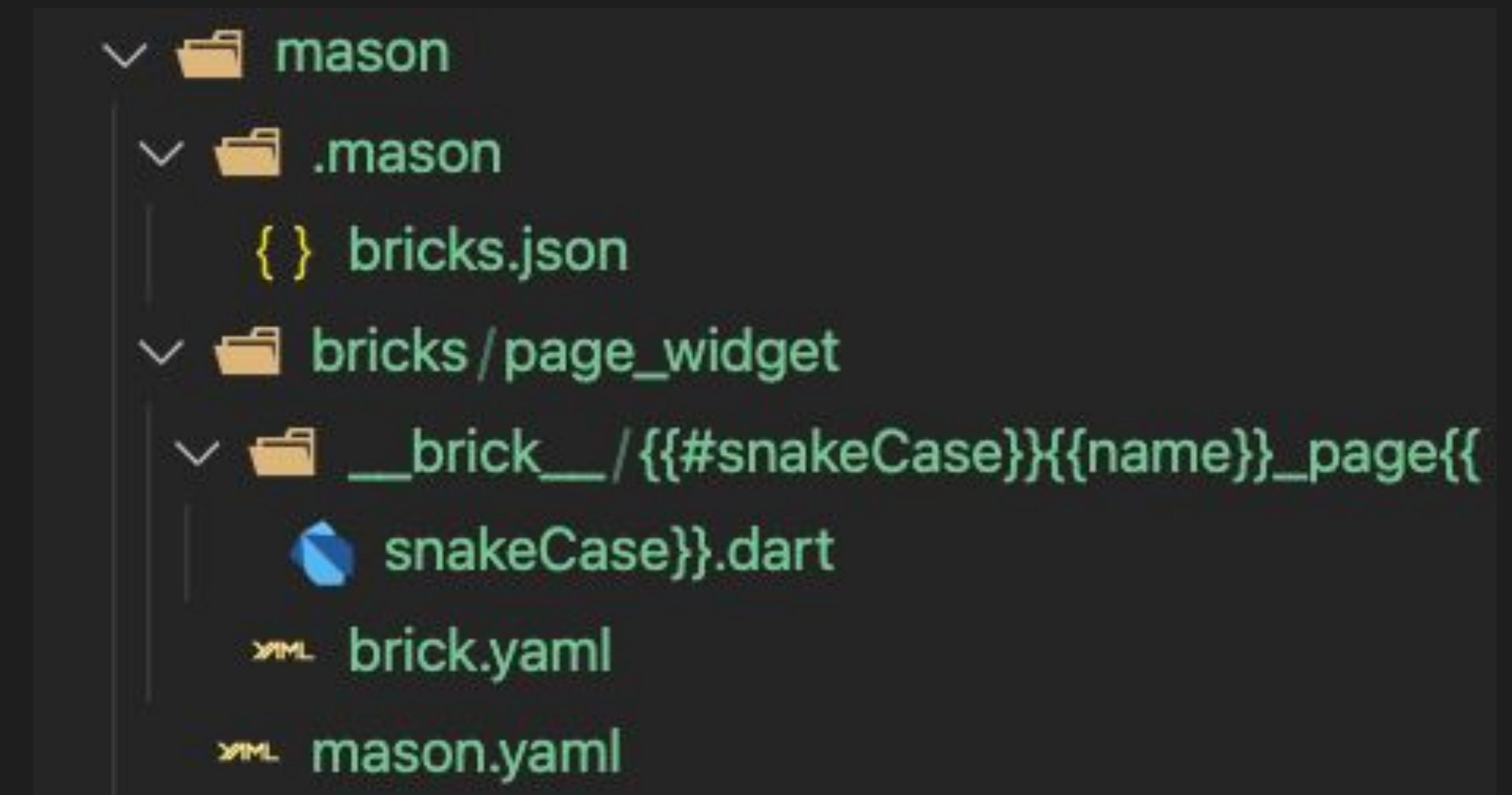
```
> mason init  
> mason new page_widget  
> mason get
```

```
mason.yaml U X  
mason > mason.yaml > ...  
1   bricks:  
2     page_widget:  
3       path: bricks/page_widget
```



Mason: Brick structure/dynamic name

```
brick.yaml U X  
mason > bricks > page_widget > brick.yaml > [ ] vars  
1   name: page_widget  
2   description: A custom page widget brick.  
3   vars:  
4     - name  
5     - routable
```



`__brick_/{#snakeCase}{{name}}_page{{/snakeCase}}.dart`

Mason: Brick

Checks routable flag

```
1 import 'package:flutter/material.dart';
2
3 class {{#pascalCase}}{{name}} page {{/pascalCase}} extends StatelessWidget {
4   {{#routable}}
5     static PageRoute route() {
6       return MaterialPageRoute(builder: (context) => const {{#pascalCase}}{{name}} page {{/pascalCase}}());
7     }
8   {{/routable}}
9
10 const {{#pascalCase}}{{name}} page {{/pascalCase}}();
11
12 @override
13 Widget build(BuildContext context) {
14   return const _{{#pascalCase}} {{name}} page view {{/pascalCase}}();
15 }
16 }
17
18 class _{{#pascalCase}}{{name}} page view {{/pascalCase}} extends StatelessWidget {
19   const _{{#pascalCase}}{{name}} page view {{/pascalCase}}();
20
21   @override
22   Widget build(BuildContext context) {
23     return Container();
24   }
25 }
```

Dynamic name in PascalCase

Mason: Generating code

```
> mason make page_widget -o ...
```

```
Mangirdass-MBP:mason mkobuolys$ mason make page_widget -o ../lib/  
name: Custom  
routable: true  
✓ Made brick page_widget (7.7ms)  
✓ Generated 1 file(s):  
  /Users/mkobuolys/dev/Flutter_projects/flutter-code-generation/lib/custom_page.dart (new)
```

Mason: Expected vs Actual result

```
1 import 'package:flutter/material.dart';
2
3 class CustomPage extends StatelessWidget {
4     static PageRoute route() {
5         return MaterialPageRoute(builder: (context) => const CustomPage());
6     }
7
8     const CustomPage();
9
10    @override
11    Widget build(BuildContext context) {
12        return const _CustomPageView();
13    }
14}
15
16 class _CustomPageView extends StatelessWidget {
17    const _CustomPageView();
18
19    @override
20    Widget build(BuildContext context) {
21        return Container();
22    }
23}
```

```
1 import 'package:flutter/material.dart';
2
3 class CustomPage extends StatelessWidget {
4     static PageRoute route() {
5         return MaterialPageRoute(builder: (context) => const CustomPage());
6     }
7
8     const CustomPage();
9
10    @override
11    Widget build(BuildContext context) {
12        return const _CustomPageView();
13    }
14}
15
16 class _CustomPageView extends StatelessWidget {
17    const _CustomPageView();
18
19    @override
20    Widget build(BuildContext context) {
21        return Container();
22    }
23}
24
25}
```

[mkobuolys/flutter-code-generation](https://github.com/mkobuolys/flutter-code-generation) Public

Code Issues Pull requests Actions Projects Wiki Security Insights

master 1 branch 0 tags Go to file Add file Code

mkobuolys Add Rive be94af0 32 seconds ago 24 commits

android	Init project	6 months ago
assets	Add Rive	32 seconds ago
ios	Update codebase	6 hours ago
lib	Add Rive	32 seconds ago
mason	Add Mason example	6 months ago
web	Init project	6 months ago
.gitignore	Add Mason example	6 months ago
.metadata	Add .gitignore and .metadata files	6 months ago
README.md	Update slides/readme	6 months ago
analysis_options.yaml	Update project dependencies.	3 months ago
header.png	Update header image	6 months ago
i10n.yaml	Add localization files and dependencies	6 months ago
presentation-slides.pdf	Update presentation slides	3 months ago
pubspec.lock	Add Rive	32 seconds ago
pubspec.yaml	Add Rive	32 seconds ago

README.md

Flutter Code Generation Examples

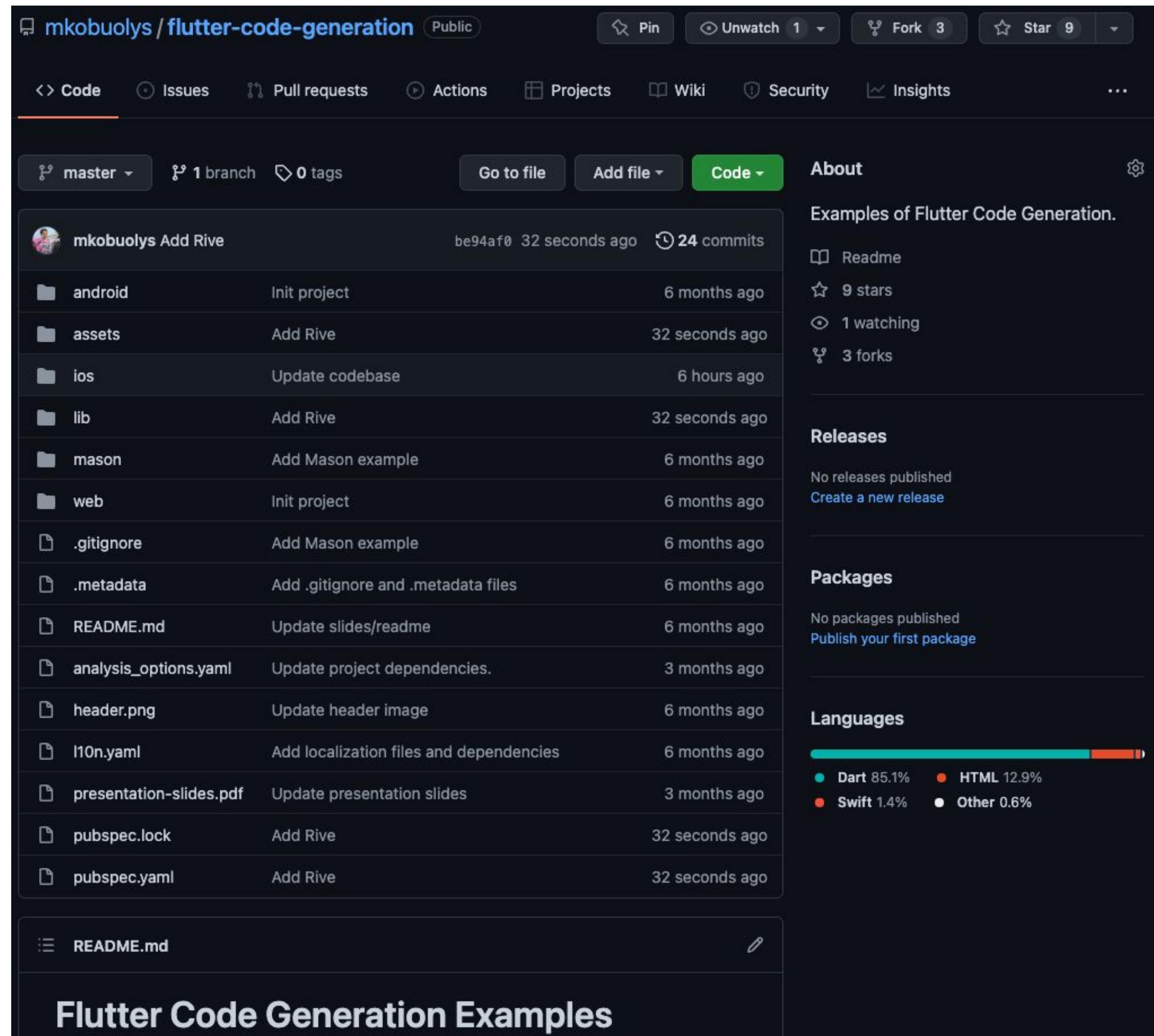
About Examples of Flutter Code Generation.

Readme 9 stars 1 watching 3 forks

Releases No releases published Create a new release

Packages No packages published Publish your first package

Languages Dart 85.1% HTML 12.9% Swift 1.4% Other 0.6%



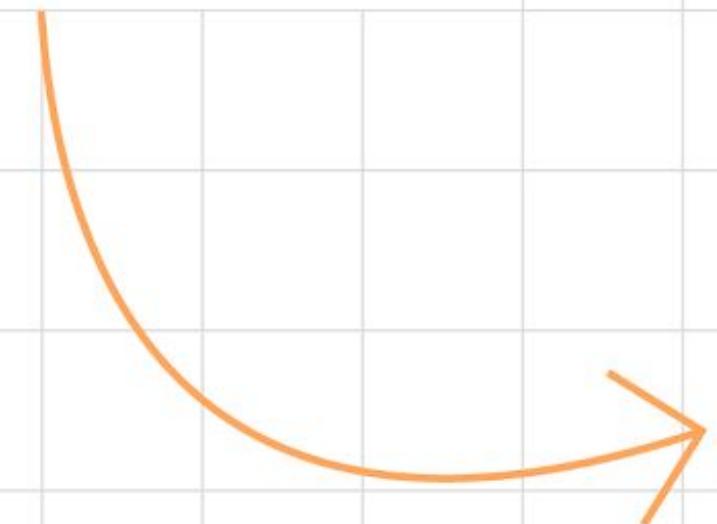
<https://github.com/mkobuolys/flutter-code-generation>

Resources

- Localization
 - Internationalizing Flutter apps – <https://bit.ly/3z945Xu>
 - Internationalization User Guide - <https://bit.ly/3iHdOOv>
 - intl - <https://pub.dev/packages/intl>
- Assets
 - build_runner - https://pub.dev/packages/build_runner
 - flutter_gen - https://pub.dev/packages/flutter_gen

Resources

- Objects & State
 - freezed – <https://pub.dev/packages/freezed>
 - freezed_annotation - https://pub.dev/packages/freezed_annotation
 - json_serializable - https://pub.dev/packages/json_serializable
- Widgets
 - GitHub Copilot - <https://copilot.github.com/>
 - mason - <https://pub.dev/packages/mason>



Save trees. Stay SOLID.
Thank you!



Mangirdas Kazlauskas
GDE for Flutter & Dart
@mkobuolys

