Code generation

Same, but different

Generate your data classes in a fast and fun way

Code generation

Same, but different

Pantelis Tsakoulis

Senior Flutter Engineer
@Kin+Carta

Nikos Portolos

Flutter Engineer
@Kin+Carta

Code generation

Same, but different

Generate your data classes in a fast and fun way

What’s a **Data Class**?

Data Classes
in other languages

Kotlin

Classes whose main purpose is to hold data.

The compiler automatically derives the following members from all properties declared in the primary constructor:

- **equals()/hashCode()** pair
- **toString()** of the form "User(name=John, age=42)"
- **componentN() functions** corresponding to the properties in their order of declaration.
- **copy()** function

https://kotlinlang.org/docs/data-classes.html

python™

A data class is a class typically containing mainly data, although there aren't really any restrictions.
It is created using the @dataclass decorator.

By default, data classes implement:

- **.__init__()** method to instantiate
- **.__repr__()** method to provide a nice string representation
- **.__eq__()** method that can do basic object comparisons

https://realpython.com/python-data-classes/

Data Classes

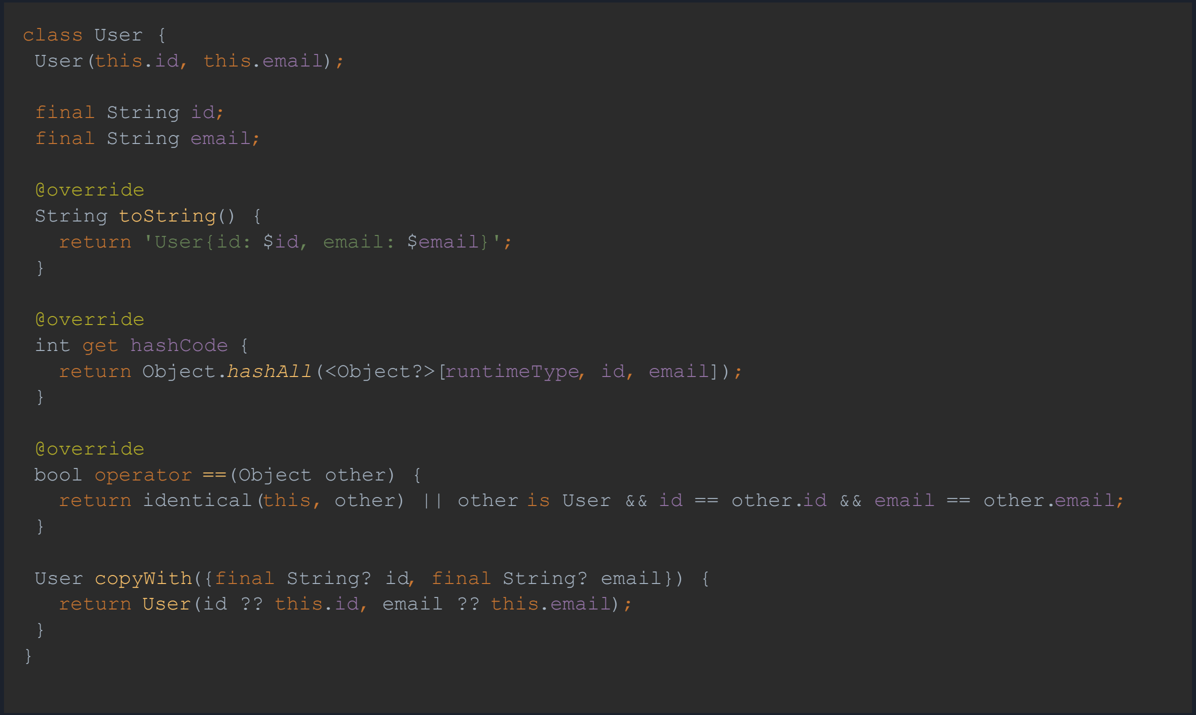
in Dart

A Data Class is essentially a regular Dart class that comes with an automatically provided constructor and implementations of **==**, **hashCode**, and **copyWith()** methods based on the **fields** the user declares in the class.

https://github.com/dart-lang/language/issues/1482

Data Classes

in Dart



```dart
class User {
 User(this.id, this.email);

 final String id;
 final String email;

 @override
 String toString() {
   return 'User{id: $id, email: $email}';
 }

 @override
 int get hashCode {
   return Object.hashAll(<Object?>[runtimeType, id, email]);
 }

 @override
 bool operator ==(Object other) {
   return identical(this, other) || other is User && id == other.id && email == other.email;
 }

 User copyWith({final String? id, final String? email}) {
   return User(id ?? this.id, email ?? this.email);
 }
}
```

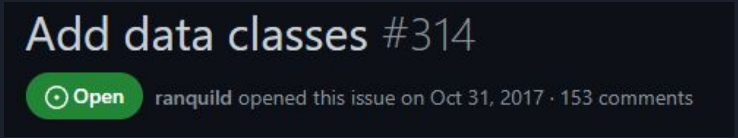
Data Classes
in Dart

Not supported natively by Dart

#1 requested feature in the dart_lang repo

Community packages and plugins

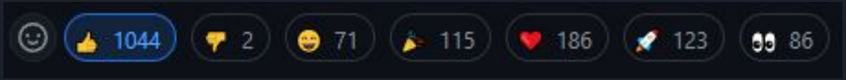
The community requested…



Add data classes #314

⊙ Open ranquild opened this issue on Oct 31, 2017 · 153 comments

… and the community reacted



☺ 👍 1044 👎 2 😄 71 🎉 115 ❤️ 186 🚀 123 👀 86

https://github.com/dart-lang/language/issues/314

Code Generation

in Dart/Flutter

Code generation
in Dart/Flutter

Mainly used for:

- Data Classes
- Json Serializing
- Unions
- Assets
- Routes
- Tests
- BLoC
- Provider/ Riverpod
- Translations
- App Flavors
- Native splash screen
- Code templates

Advantages:

- Save time and effort
- Avoid mistakes
- Code/asset safety
- Requires less testing

Code generation

in Dart/Flutter

📦 Pub packages

```
dart/flutter pub run build_runner build
```

build_runner

A build system for Dart code generation and modular compilation.
Provides a concrete way of generating files using Dart code.
Files are always generated directly on disk.
Rebuilds are incremental.

Code generation

in Dart/Flutter

source_gen

Its main purpose is to expose a developer-friendly API on top of lower-level packages like the analyzer or build.

analyzer

This package provides a library that performs static analysis of Dart code.
It is useful for tool integration and embedding.

build

A package for authoring build_runner compatible code generators.
Defines the basic pieces of how a build happens and how they interact.

Code generation

in Dart/Flutter

📦 Pub packages

freezed

json_ serializable

copy_with_ extension

built_value

flutter_gen

many more...

Data Class Generation

in Dart/Flutter

Data Class Generation

in Dart/Flutter

Manually

✔ Mostly used

- By new starters
- For simple projects
- To avoid extra dependencies

Data Class Generation

in Dart/Flutter

Manually

Pub packages (build_runner)

Most common ways of generating Data Classes

copy_with_extension + equatable
+ json_serializable
(if json support is needed)

freezed
+ json_serializable
(if json support is needed)

Equatable is not a code generator, so manual code is still required

Data Class Generation

in Dart/Flutter

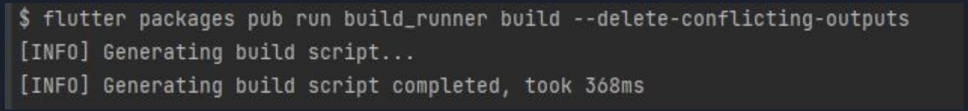
Manually

Pub packages
(build_runner)

⚠️ Issues with build_runner

Might be hard to use

Especially for new starters or
in projects with multiple packages



```
$ flutter packages pub run build_runner build --delete-conflicting-outputs
[INFO] Generating build script...
[INFO] Generating build script completed, took 368ms
```

Data Class Generation

in Dart/Flutter

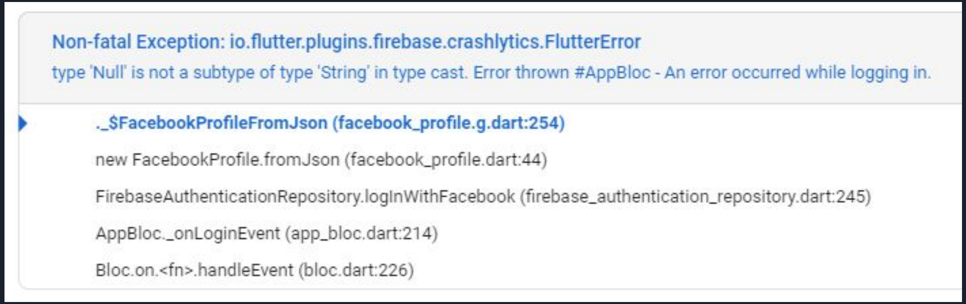
Manually

Pub packages (build_runner)

⚠️ Issues with build_runner

Usually the generated code is complex

It's hard to navigate through the generated code and understand it



Non-fatal Exception: io.flutter.plugins.firebase.crashlytics.FlutterError
type 'Null' is not a subtype of type 'String' in type cast. Error thrown #AppBloc - An error occurred while logging in.

```
._$FacebookProfileFromJson (facebook_profile.g.dart:254)
new FacebookProfile.fromJson (facebook_profile.dart:44)
FirebaseAuthenticationRepository.logInWithFacebook (firebase_authentication_repository.dart:245)
AppBloc._onLoginEvent (app_bloc.dart:214)
Bloc.on.<fn>.handleEvent (bloc.dart:226)
```

Data Class Generation

in Dart/Flutter

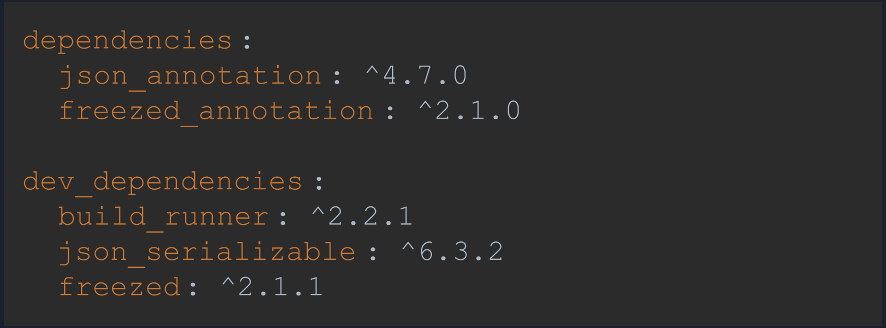
Manually

Pub packages
(build_runner)

⚠️ Issues with build_runner

A lot of dependencies

Common example freezed + json_serializable,
add 5 dependencies in your project



```
dependencies :
  json_annotation : ^4.7.0
  freezed_annotation : ^2.1.0

dev_dependencies :
  build_runner : ^2.2.1
  json_serializable : ^6.3.2
  freezed: ^2.1.1
```

Data Class Generation

in Dart/Flutter

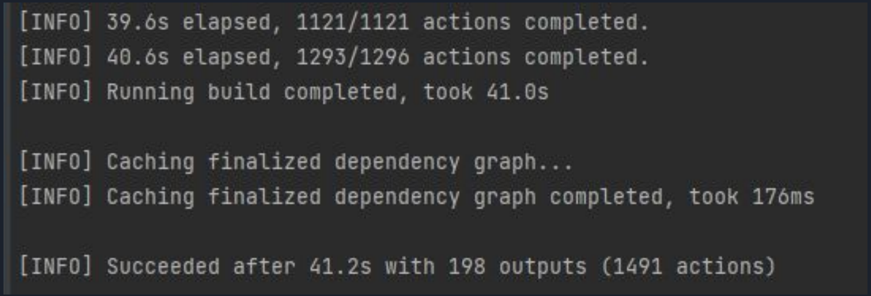
Manually

Pub packages
(build_runner)

⚠️ Issues with build_runner

Becomes slower as the project grows

Consumes more resources as it re-creates generated files when dart files change



```
[INFO] 39.6s elapsed, 1121/1121 actions completed.
[INFO] 40.6s elapsed, 1293/1296 actions completed.
[INFO] Running build completed, took 41.0s

[INFO] Caching finalized dependency graph...
[INFO] Caching finalized dependency graph completed, took 176ms

[INFO] Succeeded after 41.2s with 198 outputs (1491 actions)
```

Data Class Generation

in Dart/Flutter

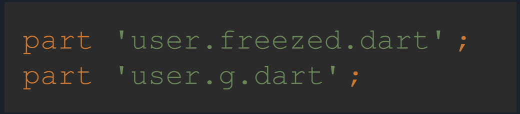
Manually

Pub packages
(build_runner)

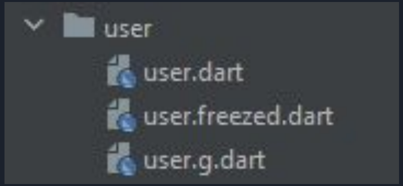
⚠️ Issues with build_runner

**Creates at least one generated file
for each dart file**

Freezed + json_serializable create 2 files



```
part 'user.freezed.dart';
part 'user.g.dart';
```



```
user
    user.dart
    user.freezed.dart
    user.g.dart
```

Data Class Generation
in Dart/Flutter

Manually

Pub packages (build_runner)

IDE Plugins

VS Code

IntelliJ IDEA

android studio

Challenges & risks:

- Requires consistency between different IDEs
- Requires extra installations and updates
- Plugin creators need to maintain compatibility for multiple IDEs

Data Class Generation

in Dart/Flutter

Manually

Pub packages (build_runner)

IDE Plugins

Static Metaprogramming

Metaprogramming refers to code that operates on other code as if it were data.

It can take code in as parameters, reflect over it, inspect it, create it, modify it, and return it.

Static metaprogramming means doing that work at compile-time, and typically modifying or adding to the program based on that work.

https://github.com/dart-lang/language/issues/1482

Data Class Generation
in Dart/Flutter

NEW!

- Manually
- Pub packages (build_runner)
- IDE Plugins
- Static Metaprogramming
- Data Class Plugin

Data Class Plugin

Data Class Plugin

Manually

Pub packages (build_runner)

IDE Plugins

Static Metaprogramming

Data Class Plugin

What's Data Class Plugin?

Data Class Plugin is a tool that uses the **Dart Analysis Server** to generate code on-the-fly.

How does it work?

Data Class Plugin uses the **analyzer** and the **analyzer plugin** to get access on the source code, parse it and provide actions based on that.

Data Class Plugin

Manually

Pub packages (build_runner)

IDE Plugins

Static Metaprogramming

Data Class Plugin

How to install?

1. Add it as a dependency in your project's pubspec.yaml

```yaml
dependencies:
 data_class_plugin: ^0.0.4
```

2. Update your analysis_options.yaml

```yaml
analyzer:
 plugins:
   - data_class_plugin
```

3. Restart the analysis server

Data Class Plugin

Manually

Pub packages (build_runner)

IDE Plugins

Static Metaprogramming

Data Class Plugin

Benefits of **Data Class Plugin**

Single dependency

in pubspec.yaml

Easy to setup and customize

yaml configuration file

Integrates with dart's analysis server

IDE agnostic

Runs per file

On demand

Creates 0 output files

Code is generated directly on the original file

Produces simple code

The generated code is easy to read and understand

Data Class Plugin

Manually

Pub packages (build_runner)

IDE Plugins

Static Metaprogramming

Data Class Plugin

Features of **Data Class Plugin**

Data Classes

constructor
copyWith
hash and equals
toString
from/to json conversion
Generic types
Super classes

Union classes

Data class
from/to json conversion

Json serialization

Customizable name and naming convention
Custom from/to json functions

Enums

constructor
from/to json conversion

Data Class Plugin

Live demo

Data Class Plugin

Pros & Cons

Data Class Plugin

Pros & Cons

Manually

Pub packages (build_runner)

IDE Plugins

Static Metaprogramming

Data Class Plugin

Easy to use

Faster development iteration

Less dependencies

Less produced code

Human readable code

Can require manual re-run on breaking changes

Custom changes in generated code will be overwritten

Cannot generate all at once - manual re-run required

Data Class Plugin

Future Roadmap

Data Class Plugin

Future Roadmap

- Manually
- Pub packages (build_runner)
- IDE Plugins
- Static Metaprogramming
- **Data Class Plugin**

Speed up the code generation process for the whole project

Potential Bug fixes - Please provide feedback!

Enhance configuration for more options on the generated code

Reach production ready levels

Thank you!

Pantelis Tsakoulis

Senior Flutter Engineer
@Kin+Carta

Nikos Portolos

Flutter Engineer
@Kin+Carta

Code generation

Same, but different

We really appreciate
your feedback!