Flutter如何JSON转Model

原创 coderwhy coderwhy

Flutter如何JSON转Model

> 在开发中，服务端通常给我们返回的是JSON数据，我们需要将JSON数据转成我们的模型对象来使用。
>
> 在Flutter中，有几种JSON转模型的方式，我们还是以豆瓣为例，来进行一个演练；

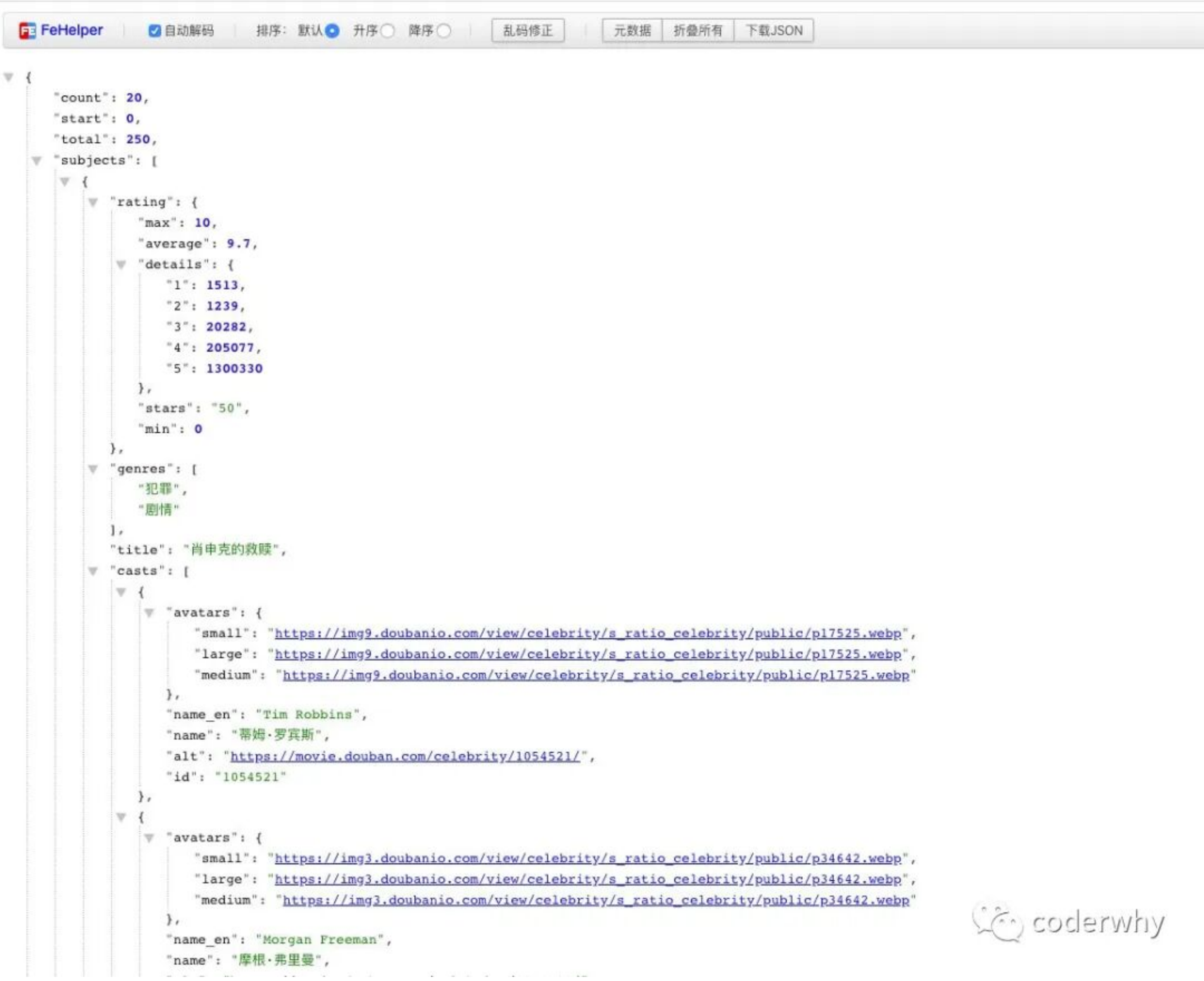
一. 豆瓣数据

这里我们使用豆瓣的请求地址：

- https://douban.uieee.com/v2/movie/top250?start=0&count=20

在浏览器中请求，获取到的数据如下：

- 注意：这里我使用了一个格式化插件：FeHelper，所以结构看起来很清晰



```
{
    "count": 20,
    "start": 0,
    "total": 250,
    "subjects": [
        {
            "rating": {
                "max": 10,
                "average": 9.7,
                "details": {
                    "1": 1513,
                    "2": 1239,
                    "3": 20282,
                    "4": 205077,
                    "5": 1300330
                },
                "stars": "50",
                "min": 0
            },
            "genres": [
                "犯罪",
                "剧情"
            ],
            "title": "肖申克的救赎",
            "casts": [
                {
                    "avatars": {
                        "small": "https://img9.doubanio.com/view/celebrity/s_ratio_celebrity/public/p17525.webp",
                        "large": "https://img9.doubanio.com/view/celebrity/s_ratio_celebrity/public/p17525.webp",
                        "medium": "https://img9.doubanio.com/view/celebrity/s_ratio_celebrity/public/p17525.webp"
                    },
                    "name_en": "Tim Robbins",
                    "name": "蒂姆·罗宾斯",
                    "alt": "https://movie.douban.com/celebrity/1054521/",
                    "id": "1054521"
                },
                {
                    "avatars": {
                        "small": "https://img3.doubanio.com/view/celebrity/s_ratio_celebrity/public/p34642.webp",
                        "large": "https://img3.doubanio.com/view/celebrity/s_ratio_celebrity/public/p34642.webp",
                        "medium": "https://img3.doubanio.com/view/celebrity/s_ratio_celebrity/public/p34642.webp"
                    },
                    "name_en": "Morgan Freeman",
                    "name": "摩根·弗里曼",
```

image-20200318110639561

这个数据还是比较复杂的：

- 如果我们希望在Flutter代码中使用，直接将JSON转成Map来使用也可以，但是非常麻烦，而且类型会不容易确定，并且不安全；
- 所以对于面向对象开发的语言，我们通常都会将它转成模型对象，之后使用一个个模型对象；

我们一起来探究一下，目前Flutter中比较常见的将JSON转成模型的方式。

二. 手动转化

JSON转模型，必然可以通过手动来进行转化：

- 优点：完全是自己可控的，并且需要哪些字段就转化哪些字段，对于不需要的，忽略即可；并且继承关系也会一目了然
- 缺点：麻烦，并且容易出错；

下面是我之前针对上面的数据，写的JSON转Model的模型类：

```
class Person {
  String name;
  String avatarURL;

  Person.fromMap(Map<String, dynamic> json) {
    this.name = json["name"];
    this.avatarURL = json["avatars"]["medium"];
  }
}

class Actor extends Person {
  Actor.fromMap(Map<String, dynamic> json): super.fromMap(json);
}

class Director extends Person {
  Director.fromMap(Map<String, dynamic> json): super.fromMap(json);
}

int counter = 1;

class MovieItem {
  int rank;
  String imageURL;
  String title;
  String playDate;
  double rating;
  List<String> genres;
  List<Actor> casts;
  Director director;
  String originalTitle;

  MovieItem.fromMap(Map<String, dynamic> json) {
    this.rank = counter++;
    this.imageURL = json["images"]["medium"];
    this.title = json["title"];
    this.playDate = json["year"];
    this.rating = json["rating"]["average"];
    this.genres = json["genres"].cast<String>();
    this.casts = (json["casts"] as List<dynamic>).map((item) {
      return Actor.fromMap(item);
    }).toList();
    this.director = Director.fromMap(json["directors"][0]);
    this.originalTitle = json["original_title"];
  }
}
```

三. json_serializable

json_serializable是dart官方推荐和提供的JSON转Model的方式：

- 一个自动化源代码生成器来为你生成 JSON 序列化数据模板；
- 由于序列化数据代码不再需要手动编写或者维护，你可以将序列化 JSON 数据在运行时的异常风险降到最低；

第一步：添加相关的依赖

依赖分为项目依赖（dependencies），开发依赖（dev_dependencies）：

- 注意：需要执行flutter pub get确保我们的项目中有这些依赖

```
dependencies:
  json_annotation: ^3.0.1

dev_dependencies:
  json_serializable: ^3.2.5
  build_runner: ^1.8.0
```

第二步：以json_serializable 的方式创建模型类

这里不以豆瓣数据为例，以一个简单的Json数据作为例子

```
  final jsonInfo = {
    "nickname": "coderwhy",
    "level": 18,
    "courses": ["语文", "数学", "英语"]
,
    "register_date": "2222-2-22",
    "computer": {
      "brand": "MackBook",
      "price": 1000
    }
  };
```

创建对应的模型（以json_serializable 的方式，创建完成后代码是 **报错的** ）

- 1.part 'user.g.dart'
  - 这个是之后json_serializable会自动帮助我们生成的文件
- 2.JsonSerializable()
  - 注解：告诉json_serializable哪一个类需要进行转换
- 3.JsonKey
  - 当映射关系不一样时，可以指定映射关系
- 4.另外，这里必须有我们的构造方法
- 5.需要有对应的工厂构造器
  - _ [redacted]_UserToJson(this)调用的该方法目前会报错，需要json_serializable来生成
- 6.toString方法不是必须的，是待会儿进行测试的

User类的代码：

```
import 'package:json_annotation/json_annotation.dart';
import 'model/computer.dart';

part 'user.g.dart';

@JsonSerializable()
class User {
  String name;
  String email;
  @JsonKey(name: "register_date")
  String registerDate;
  List<String> courses;
  Computer computer;

  User(this.name, this.email, this.registerDate, this.courses, this.computer);

  factory User.fromJson(Map<String, dynamic> json) => _$UserFromJson(json);
  Map<String, dynamic> toJson() => _$UserToJson(this);

  @override
  String toString() {
    return 'User{name: $name, email: $email, registerDate: $registerDate, courses: $courses, computer: $computer}'
;
  }

}
```

Computer类的代码：

```
import 'package:json_annotation/json_annotation.dart';

part 'computer.g.dart';

@JsonSerializable()
class Computer {
  String brand;
  double price;

  Computer(this.brand, this.price);

  factory Computer.fromJson(Map<String, dynamic> json) => _$ComputerFromJson(json);
  Map<String, dynamic> toJson() => _$ComputerToJson(this);

  @override
  String toString() {
    return 'Computer{brand: $brand, price: $price}';
  }
}
```

第三步：生成JSON序列化代码

在项目终端运行下面的指令：

- 该指令是生成一次JSON序列化的代码

```
flutter pub run build_runner build
```

或运行下面的指令：

- 会监听文件的改变，重新生成JSON序列化的代码

```
flutter pub run build_runner watch
```

第四步：测试代码

```
final jsonInfo = {
  "nickname": "coderwhy",
  "level": 18,
  "courses": ["语文", "数学", "英语"],
  "register_date": "2222-2-22",
  "computer": {
    "brand": "MackBook",
    "price": 1000
  }
};

final user = User.fromJson(jsonInfo);
print(user);
```

更多资料，请查看下面的资源：

- `dart:convert` 和 `JsonCodec` 文档
- Pub 中的 json_serializable package
- GitHub 中的 json_serializable 例子

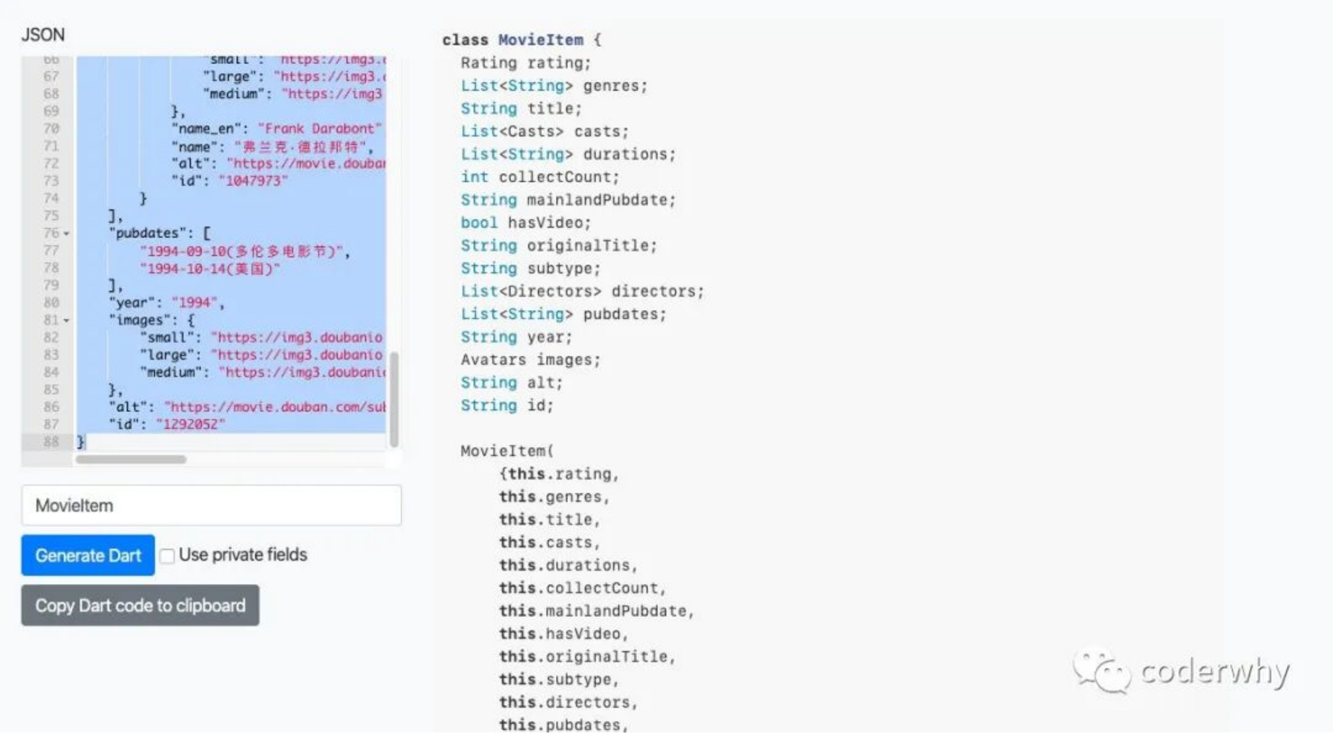
四. 网页转换

目前有一些网页，可以直接将JSON转成Model

- 网页推荐：https://javiercbk.github.io/json_to_dart/

我们这里以网页版本为例，非常简单：

- 注意：可能因为豆瓣的数据过于复杂，所以在生成的时候发现少了一个Directors类
- 这里我重新复制对应的JSON，再次生成了一下



JSON

```
                "small": "https://img3.
                "large": "https://img3.
                "medium": "https://img3
            },
            "name_en": "Frank Darabont"
            "name": "弗兰克·德拉邦特",
            "alt": "https://movie.douba
            "id": "1047973"
        }
    ],
    "pubdates": [
        "1994-09-10(多伦多电影节)",
        "1994-10-14(美国)"
    ],
    "year": "1994",
    "images": {
        "small": "https://img3.doubanio
        "large": "https://img3.doubanio
        "medium": "https://img3.doubani
    },
    "alt": "https://movie.douban.com/sub
    "id": "1292052"
}
```

MovieItem

□ Use private fields

```
class MovieItem {
  Rating rating;
  List<String> genres;
  String title;
  List<Casts> casts;
  List<String> durations;
  int collectCount;
  String mainlandPubdate;
  bool hasVideo;
  String originalTitle;
  String subtype;
  List<Directors> directors;
  List<String> pubdates;
  String year;
  Avatars images;
  String alt;
  String id;

  MovieItem(
      {this.rating,
      this.genres,
      this.title,
      this.casts,
      this.durations,
      this.collectCount,
      this.mainlandPubdate,
      this.hasVideo,
      this.originalTitle,
      this.subtype,
      this.directors,
      this.pubdates,
```

image-20200318155605684

```
class MovieItem {
  Rating rating;
  List<String> genres;
  String title;
  List<Casts> casts;
  List<String> durations;
  int collectCount;
  String mainlandPubdate;
  bool hasVideo;
  String originalTitle;
  String subtype;
  List<Directors> directors;
  List<String> pubdates;
  String year;
  Avatars images;
  String alt;
  String id;

  MovieItem(
      {this.rating,
        this.genres,
        this.title,
        this.casts,
        this.durations,
        this.collectCount,
        this.mainlandPubdate,
        this.hasVideo,
        this.originalTitle,
        this.subtype,
        this.directors,
        this.pubdates,
        this.year,
        this.images,
        this.alt,
        this.id});

  MovieItem.fromJson(Map<String, dynamic> json) {
    rating =
    json['rating'] != null ? new Rating.fromJson(json['rating']) : null;
    genres = json['genres'].cast<String>();
    title = json['title'];
    if (json['casts'] != null) {
      casts = new List<Casts>();
      json['casts'].forEach((v) {
        casts.add(new Casts.fromJson(v));
      });
    }
    durations = json['durations'].cast<String>();
    collectCount = json['collect_count'];
    mainlandPubdate = json['mainland_pubdate'];
    hasVideo = json['has_video'];
    originalTitle = json['original_title'];
    subtype = json['subtype'];
    if (json['directors'] != null) {
      directors = new List<Directors>();
      json['directors'].forEach((v) {
        directors.add(new Directors.fromJson(v));
      });
    }
    pubdates = json['pubdates'].cast<String>();
    year = json['year'];
    images =
    json['images'] != null ? new Avatars.fromJson(json['images']) : null;
    alt = json['alt'];
    id = json['id'];
  }

  Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = new Map<String, dynamic>();
    if (this.rating != null) {
      data['rating'] = this.rating.toJson();
    }
    data['genres'] = this.genres;
    data['title'] = this.title;
    if (this.casts != null) {
      data['casts'] = this.casts.map((v) => v.toJson()).toList();
    }
    data['durations'] = this.durations;
    data['collect_count'] = this.collectCount;
    data['mainland_pubdate'] = this.mainlandPubdate;
    data['has_video'] = this.hasVideo;
    data['original_title'] = this.originalTitle;
    data['subtype'] = this.subtype;
    if (this.directors != null) {
      data['directors'] = this.directors.map((v) => v.toJson()).toList();
    }
    data['pubdates'] = this.pubdates;
    data['year'] = this.year;
    if (this.images != null) {
      data['images'] = this.images.toJson();
    }
    data['alt'] = this.alt;
    data['id'] = this.id;
    return data;
  }
}

class Rating {
  int max;
  double average;
  Details details;
  String stars;
  int min;

  Rating({this.max, this.average, this.details, this.stars, this.min});

  Rating.fromJson(Map<String, dynamic> json) {
    max = json['max'];
    average = json['average'];
    details =
    json['details'] != null ? new Details.fromJson(json['details']) : null
;
```

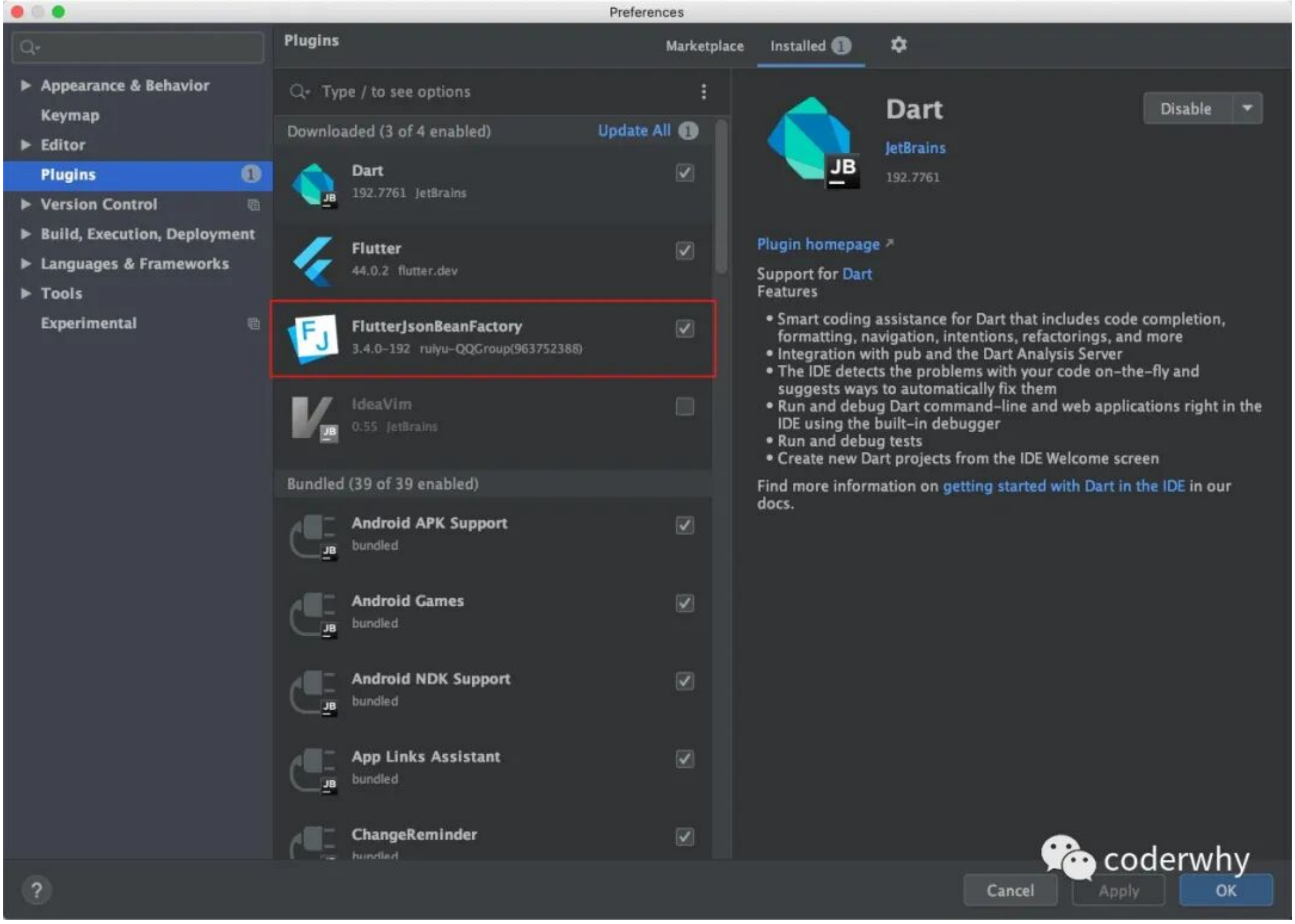
```
    stars = json['stars'];
    min = json['min'];
  }

  Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = new Map<String, dynamic>();
    data['max'] = this.max;
    data['average'] = this.average;
    if (this.details != null) {
      data['details'] = this.details.toJson();
    }
    data['stars'] = this.stars;
    data['min'] = this.min;
    return data;
  }
}

class Details {
  int i1;
  int i2;
  int i3;
  int i4;
  int i5;

  Details({this.i1, this.i2, this.i3, this.i4, this.i5});

  Details.fromJson(Map<String, dynamic> json) {
    i1 = json['1'];
    i2 = json['2'];
    i3 = json['3'];
    i4 = json['4'];
    i5 = json['5'];
  }

  Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = new Map<String, dynamic>();
    data['1'] = this.i1;
    data['2'] = this.i2;
    data['3'] = this.i3;
    data['4'] = this.i4;
    data['5'] = this.i5;
    return data;
  }
}

class Casts {
  Avatars avatars;
  String nameEn;
  String name;
  String alt;
  String id;

  Casts({this.avatars, this.nameEn, this.name, this.alt, this.id});

  Casts.fromJson(Map<String, dynamic> json) {
    avatars =
    json['avatars'] != null ? new Avatars.fromJson(json['avatars']) : null
;
    nameEn = json['name_en'];
    name = json['name'];
    alt = json['alt'];
    id = json['id'];
  }

  Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = new Map<String, dynamic>();
    if (this.avatars != null) {
      data['avatars'] = this.avatars.toJson();
    }
    data['name_en'] = this.nameEn;
    data['name'] = this.name;
    data['alt'] = this.alt;
    data['id'] = this.id;
    return data;
  }
}

class Directors {
  Avatars avatars;
  String nameEn;
  String name;
  String alt;
  String id;

  Directors({this.avatars, this.nameEn, this.name, this.alt, this.id});

  Directors.fromJson(Map<String, dynamic> json) {
    avatars =
    json['avatars'] != null ? new Avatars.fromJson(json['avatars']) : null
;
    nameEn = json['name_en'];
    name = json['name'];
    alt = json['alt'];
    id = json['id'];
  }

  Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = new Map<String, dynamic>();
    if (this.avatars != null) {
      data['avatars'] = this.avatars.toJson();
    }
    data['name_en'] = this.nameEn;
    data['name'] = this.name;
    data['alt'] = this.alt;
    data['id'] = this.id;
    return data;
  }
}

class Avatars {
  String small;
  String large;
  String medium;

  Avatars({this.small, this.large, this.medium});

  Avatars.fromJson(Map<String, dynamic> json) {
    small = json['small'];
    large = json['large'];
    medium = json['medium'];
  }

  Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = new Map<String, dynamic>();
    data['small'] = this.small;
    data['large'] = this.large;
    data['medium'] = this.medium;
    return data;
  }
}
```

五. 编辑器插件

目前也有一些AndroidStudio或者VSCode的插件，来帮助我们直接将JSON生成对应的Model

- VSCode目前没有找到比较好用的插件推荐
- Android Studio推荐 **FlutterJsonBeanFactory**

第一步：安装插件



Support for Dart
Features

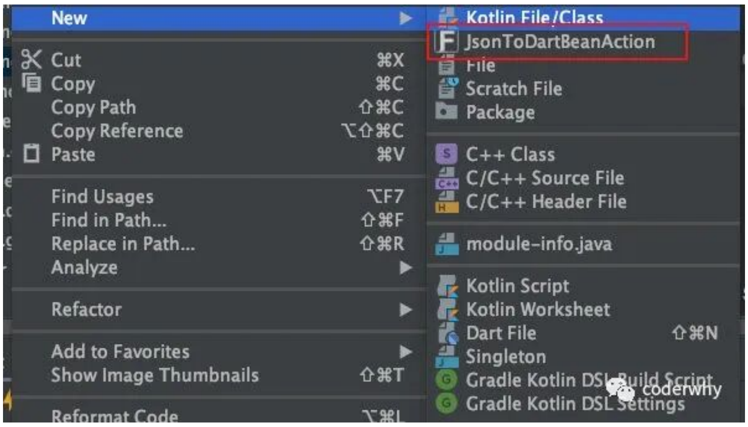
- Smart coding assistance for Dart that includes code completion, formatting, navigation, intentions, refactorings, and more
- Integration with pub and the Dart Analysis Server
- The IDE detects the problems with your code on-the-fly and suggests ways to automatically fix them
- Run and debug Dart command-line and web applications right in the IDE using the built-in debugger
- Run and debug tests
- Create new Dart projects from the IDE Welcome screen

Find more information on getting started with Dart in the IDE in our docs.

安装插件

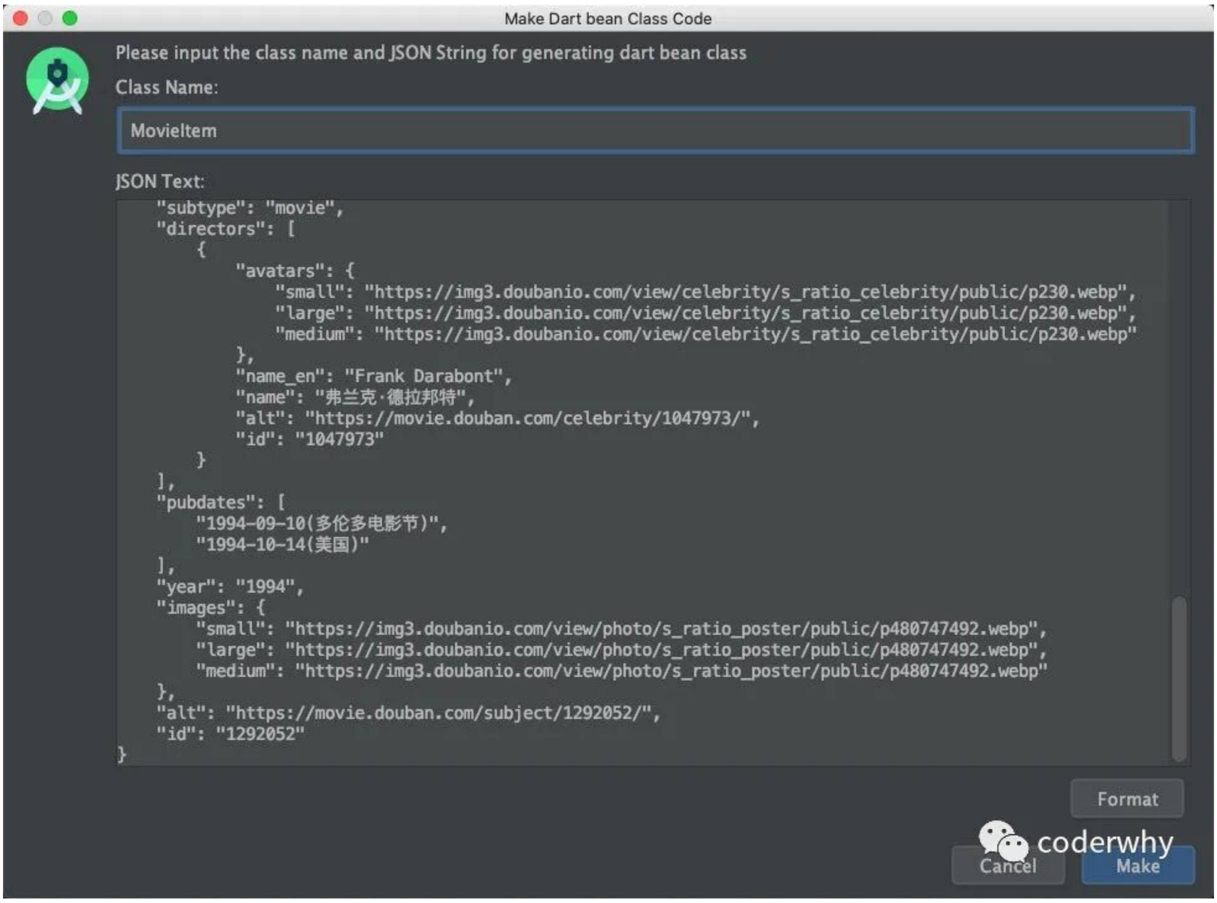
第二步：创建模型

右键新建文件：



新建文件

给类起一个名字，并且将JSON复制过去



Please input the class name and JSON String for generating dart bean class

Class Name: MovieItem

JSON Text:

```
    "subtype": "movie",
    "directors": [
        {
            "avatars": {
                "small": "https://img3.doubanio.com/view/celebrity/s_ratio_celebrity/public/p230.webp",
                "large": "https://img3.doubanio.com/view/celebrity/s_ratio_celebrity/public/p230.webp",
                "medium": "https://img3.doubanio.com/view/celebrity/s_ratio_celebrity/public/p230.webp"
            },
            "name_en": "Frank Darabont",
            "name": "弗兰克·德拉邦特",
            "alt": "https://movie.douban.com/celebrity/1047973/",
            "id": "1047973"
        }
    ],
    "pubdates": [
        "1994-09-10(多伦多电影节)",
        "1994-10-14(美国)"
    ],
    "year": "1994",
    "images": {
        "small": "https://img3.doubanio.com/view/photo/s_ratio_poster/public/p480747492.webp",
        "large": "https://img3.doubanio.com/view/photo/s_ratio_poster/public/p480747492.webp",
        "medium": "https://img3.doubanio.com/view/photo/s_ratio_poster/public/p480747492.webp"
    },
    "alt": "https://movie.douban.com/subject/1292052/",
    "id": "1292052"
}
```

转换界面

第三步：使用生成的模型

创建完成后会生成对应的模型，并且还会生成一个文件夹，里面有生成模型过程的代码

- 这里不再给出，代码都是相似的