Deserializing JSON Data

Douglas Starnes
AUTHOR / SPEAKER

@poweredbyaltnet douglasstarnes.com

Overview

Deserialization is the process of generating Dart types from JSON

Manual and automatic deserialization

- Overlap with manual and automatic serialization
- Manual deserialization depends on Maps
- Automatic deserialization generates code to handle JSON
- Watch the module on serialization first!

Manual Serialization

Use jsonDecode to generate a Map from a JSON string

Use the values in the Map to populate a new type in Dart

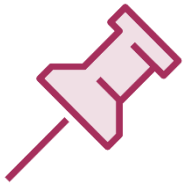
The values in the Map are still dynamic

Cast the values in the Map to the values for the Dart properties

Declare default values

Automatic Deserialization

Saves you from writing the code to cast values and default types



Annotating the type will add code to cast values

The JsonKey annotation has a property to add default value

Which should you use?

main.dart

```
import ‘dart:convert’;
var userMap = jsonDecode(‘{“lastName”: “Jones”, “loyaltyPoints”: 100}’);
var userProfile = UserProfile(
    userMap[‘lastName’] as String,
    userMap[‘loyaltyPoints’] as int
);
```

userprofile.dart

```
import ‘dart:convert’;
class UserProfile {
  UserProfile fromJson(String json) {
    var userMap = jsonDecode(json);
     return UserProfile(
         userMap[‘lastName’] as String,
         userMap[‘loyaltyPoints’] as int
    );
  }
}
```

```
var user = UserProfile(...);
var deserializedUser = user.fromJson(json);
```

userprofile.dart

```
import ‘dart:convert’;
class UserProfile {
  factory UserProfile.fromJson(String json) {
    var userMap = jsonDecode(json);
     return UserProfile(
         userMap[‘lastName’] as String,
         userMap[‘loyaltyPoints’] as int
    );
  }
}
```

```
var deserializedUser =
    UserProfile.fromJson(json);
```

pubspec.yaml

```
dependencies:
  json_annotation: ^3.1.0

dev_dependencies:
  json_serializable: ^3.5.0
  build_runner: ^1.10.1
```

userprofile.dart

```
import ‘package:json_annotation/json_annotation/dart’;

part ‘userprofile.g.dart’;

@JsonSerializable()

class UserProfile {

  String firstName;

  String lastName;

  bool isActive;

  int loyaltyPoints;

  int fitnessGoal;

}
```

userprofile.g.dart

```
part of ‘userprofile.dart’;

UserProfile _$UserProfileFromJson(Map<String, dynamic> json) {
  return UserProfile(
    json[‘firstName'] as String,
    json[‘lastName'] as String,
    json[‘isActive'] as bool,
    json[‘loyaltyPoints'] as int,
    json[‘fitnessGoal'] as int,
  );
}
```

userprofile.dart

```
import ‘package:json_annotation/json_annotation/dart’;

part ‘userprofile.g.dart’;

@JsonSerializable()

class UserProfile {

  String firstName;

  String lastName;

  bool isActive;

  @JsonKey(defaultValue: 100)

  int loyaltyPoints;

  int fitnessGoal;

}
```

userprofile.g.dart

```
part of ‘userprofile.dart’;

UserProfile _$UserProfileFromJson(Map<String, dynamic> json) {
  return UserProfile(
    json[‘firstName'] as String,
    json[‘lastName'] as String,
    json[‘isActive'] as bool,
    json[‘loyaltyPoints'] as int ?? 100,
    json[‘fitnessGoal'] as int,
  );
}
```

userprofile.dart

```
import ‘dart:convert’;
import ‘package:json_annotation/json_annotation/dart’;
part ‘userprofile.g.dart’;
@JsonSerializable()
class UserProfile {
  factory UserProfile.fromJson(String json) {
    var userMap = jsonDecode(json);
    return _$UserProfileFromJson(userMap);
  }
}
```

Summary

Deserialization

Overlap with serialization

Manual or automatic

`FutureBuilder`

`jsonDecode()`

- Takes a JSON string and returns a Map

Factory constructors

Flutter app considerations