GDG

#DevFest

05-10-2019

Flutter
After Hello World

Jeroen Meijer (Jay)

https://jfk.dev

@JFKDEV

/jeroen-meijer

Who am I

- Jeroen Meijer (Jay)
- Dart & Flutter GDE
- Flutter freelancer
- Admin and Head of GitHub for the Flutter Community 💙
- In love with Flutter for 2 years.
- Writer of the VS Code extension Pubspec Assist

Flutter

- Portable UI toolkit
- Open source
- Native apps on almost any platform (iOS, Android, Web)
- High performance
- Uses custom rendering engine and UI elements
- Complete control over the screen

IDE

- Choose your editor (VS Code, IntelliJ/Android Studio, Other)
- Learn your editor (extensions, breakpoints, hotkeys)
- Use linters (clean code, consistency, readability)

Tooling

- `flutter run -d DEVICE_ID`
- `flutter packages get` | `flutter pub get`
- `flutter devices`
- `flutter doctor`
- `flutter analyze`
- `flutter clean`

Widgets

- Immutable Classes, PODOs (Plain Old Dart Objects)
- 2 types: `StatelessWidget` and `StatefulWidget`
- Composed together to create UI
- Can do anything other classes can
    - Parameters `Button(title: 'Submit', color: Colors.blue)`
    - Functions `_onTap() { … }`
    - Getters `get _displayedNames => names.take(3);`

Widgets

- Built-in widget packages are material, cupertino and widgets
- ”There’s a widget for that!”
- Almost all UI is composable from existing widgets
- If something’s missing, look for well-maintained packages
  (Badly maintained packages can cause trouble.)
- Multiple types of widgets (visual, layout, functional)

State Management

Provider
Get It
Redux
MobX
Bloc
ChangeNotifier

- Which one is the best?
- None.
- How to choose the best one for your project:
  - Does my app need to be online all the time?
  - What state management pattern does my team know?
  - How complex is my app and will it grow in the future?

Backend

- Dart supports many protocols, such as `http` and `grpc`.
- Two `http` clients
  - `package:http` for simple APIs (GET, POST, DELETE, etc.)
  - `HttpClient` for fine-grain control (streams, cookies, etc.)
- Use existing packages (for example, `firebase`)
- Use whatever pattern works for you
- Recommended: abstract API class + repository pattern

Packages

- Extend your app’s functionality by importing existing code
- Can be found on pub.dev
- Are imported using the pubspec.yaml file

```
dependencies:
  camera: ^0.5.4+3
```

- Tip: Use Pubspec Assist to easily add and update dependencies.

Plugins

- Most packages are plugins
- Allows two-way communication between Dart and the native platform code
- Call native APIs from Flutter
- Wrap your existing Java/Kotlin and Objective C/Swift code.
- `camera`, `bluetooth`, `connectivity`, `firebase`

Go explore!
Ask questions!

DEVFEST 2019

Flutter
After Hello World

Thank you for your attention.

Jeroen Meijer (Jay)

https://jfk.dev

@JFKDEV

/jeroen-meijer

Slides and code available at: https://jfk.dev/talks

GDG