Research Notebook

Joshua Markle

Contents

Timesheet

Date	**Time Completed**
3 October 2023	2h 30m
4 October 2023	1h 30m
5 October 2023	2h

App Setup

In this section, I will research different methods of app development with the goal of deciding how to build my app and what I will use to make it.

3 October 2023 - App Frameworks

In order to make an app, some sort of framework that provides a standard way to build and deploy applications.

What I am looking for in a framework:

- Cross platform: Can build to both iOS and Android
- Established community: Pre-existing libraries and a place to go if I run into errors

In my search for suitable framworks, I have found that the two most popular ones are probably going to give me the best success in this project. These two frameworks are called React and Flutter, and dominate most of the app building options.

	React Native	Flutter
General	Made by facebook, React uses JavaScript to make it's components. It offers a variety of UI options and more native experience	Flutter was made by Google and has it's own take on the app development process. It uses Dart to render widgets (display components)
Pros	• Easy to learn and master (JavaScript) • Native UI unique to iOS and Android • Best for complex animations	• Dart is similar to other languages I know • Excelent documentation • Standard UI across both iOS and Android • Highly performant
Cons	• Poor documentation • Constant updates (many things get deprecated) • Overwelhming number of options/solutions	• Harder to master • Limited libraries by 3rd parties • Large app sizes

Sources

- React Native vs Flutter, cross-platform mobile application frameworks (Wenhao Wu)
- Flutter vs. React Native: Which One to Choose in 2023? (Chinmayee Deshpande)

- Flutter vs React: A Comparison

4 October 2023 - Project Setup

After conducting the research on these two popular frameworks, I have decided to go with the Flutter framework ultimately because of the development process. It has better documentation that is simple and straight to the point and can make working on apps quicker and cause less suffering.

I have setup my development environment using Neovim, a popular text editor that I have been using for a while. It is highly configurable and I plan of leveraging exactally that to hasten development.

Neovim packages installed:

- flutter-tools.nvim by akinsho
- vim-lsc and vim-lsc-dart by natebosch

(Packages can be found on GitHub)

Both of these packages allowed me to setup my project and provide helpful code completion. I also installed the Android operating system onto my laptop inorder for me to run a virtual phone on my computer to test out my app in realtime.

Currently, I have created a blank project and am fully preped for development.

5 October 2023 - Common Knowledge & Best Practices

Before work can be started on any large project, it is important that there is a plan to guide development and ensure that time is not wasted on extra tasks.

Notes pertaining to building a flutter app:

- Widgets: Building blocks of UI
- Stateless Widgets:
  - Don't store mutable state
  - Rebuild only when parent changes
- Stateful Widgets:
  - Maintain mutable state
  - Can rebuild when internal state changes
- Packages & Plugins:
  - Reusable components
  - Find at `https://pub.dev/flutter`
- Basic Flutter App Structure
  - `main.dart` as the entry point
  - Define `main()` function
  - Define at least one `Widget`
- Creating a New App
  - Use terminal or command prompt
  - `flutter create [app_name]`
  - `cd [app_name]`
  - `flutter run` to start the app
- Additional Resources
  - Flutter Documentation: `https://flutter.dev/docs`
  - Flutter Community: `https://flutter.dev/community`

Below is a sample project that builds necessary components for the app and centers text within the center of the screen.

```
import 'package:flutter/material.dart';

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Stateless Widget Example'),
        ),
        body: Center(
          child: Text('Hello, Flutter!'),
        ),
      ),
    );
  }
}
```

This code demonstrates a stateless widget and how Flutter structures their app components. Each class can contain Widget that returns a page to the user's screen when called. Stateful widgets on the other hand update in realtime on the user's screen unlike a stateless widget. This is the core of flutter, everything is a widget either stateless or stateful.

Sources Reviewed:

- Test Drive (Flutter Documentation)
- How To Build Mobile Apps With Flutter (FreeCodeCamp)
- GeeksForGeeks Flutter Documentation