FLUTTER TRAINING

Outline

- Getting started
- Dart basics
- Flutter widgets
- Material design
- Responsive design
- Network requests (REST APIs, JSON parsing)
- Persistent storage
- Unit testing
- Architectural patterns (provider, scoped model, bloc)
- Animations
- Mobile application security
- .apk and .ipa artifacts generation
- CI/CD (Codemagic)
- Writing platform-specific code
- Developing packages
- Product flavors
- Publishing

Weekly Schedule

Week 1 (Taha Ali) (28th April) **[Presentation Slides]**

Remove fear of learning a new language i.e. Dart.

Setup Flutter and explore syntax of fundamentals in dart and their similarities with other languages that you are familiar with. Practice syntax using DartPad of different data types, variables, operators, loops, conditional statements, lists, maps, functions, classes, interfaces etc.

Assignment: Read text from a file and find words that appear most in a line in the file.

(i) finding the highest frequency word(s) in each line

(ii) finding lines in the file whose "highest frequency words" is the greatest value among all lines.

Print the result in the following format:

> *The following words have the highest word frequency per line:*
> *["word1"] (appears in line #)*
> *["word2", "word3"] (appears in line #)*

Week 2 (Taha Ali) (5th May) **[Presentation Slides]**

Explore Flutter Widget Catalog which is one of the reasons you are able to build apps faster. Understand the difference between stateless and stateful widgets, Make use of basic widgets such as Row, Column, Container, Icon. Differentiate between usage of different types of widgets such as the single child ones like Text and Padding, ones with multiple children like List and Grid and the ones that expose some events like Button and GestureDetector.

Explore in greater depth how widgets that incorporate material design components aid in rapid prototyping. Build beautiful layouts with ease by making use of Scaffold, Appbar, BottomNavigationBar, Drawer, FloatingActionButton, SnackBar, ListTile and many more.

Assignment: Build a simple but complete ToDo list application. Next, reconstruct the UI layouts of everyday apps that are based on material design such as Gmail or WhatsApp in Flutter.

Week 3 (Vinod) (12th May) **[Presentation Slides]**

Learn to write code in the form of reusable components and build custom controls. Explore techniques to make UI responsive to different screen sizes. Main two approaches make use of MediaQuery and LayoutBuilder other than that widgets such as FittedBox, AspectRatio, OrientationBuilder also facilitate in building responsive UI.

Assignment: Building a responsive page using LayoutBuilder that adapts to different screen sizes and orientation. The page will contain a list view that will be hidden inside a drawer in portrait mode and in landscape mode that list will share ⅓ of the screen's width.

Week 4 (Anas)(19th May) **[Presentation Slides]**

Explore http library in dart to make network requests. Fetching data from the internet and posting it back by consuming REST APIs. Deserializing JSON into Dart objects and vice versa. Also explore different ways to parse JSON in Flutter such as manual serialization and by using code generation by help of json_serializable. Learn how to parse this JSON in the background using a separate Isolate.

Assignment: Building a weather app that consumes real data from OpenWeather APIs.

Week 5 (A. Umer/ Taha Ali) (26th May/ 02nd June) **[Presentation Slides]**

Many times an app needs to persist and query larger amounts of data on the local device. Learn different forms of local persistence solutions such as key-value store, local filing and SQLite.

Writing unit tests by the help of flutter_test package and mocking dependencies using mockito.

Assignment: Build an app that generates random startup names and you can save your favorite ones by following Flutter basics tutorial. Add logic to persist this data when the app closes and reopens.

Write unit tests for business logic that we have implemented so far in previous weeks assignments.

Week 6 (Anas)(2nd June/ 9th June) **[Presentation Slides]**

Explore different architectural patterns for better state management and separation of concern. As you build with Flutter, there comes a time when you need to share application state between screens, across your app. There are many approaches you can take, the most popular ones are: provider, scoped model and Bloc.

There are several core concepts in BloC architecture and it's proving to be a better alternative than Redux when taking the reactive programming approach in Flutter. Learn how to manage events, states, transitions, streams and blocs.

Assignment: Build example applications of each approach. A store catalog and shopping cart scenario coded using provider and counter app implemented using scoped_model. Build a ToDo app that demonstrates usage of BloC pattern.

Week 7 (Farhan)(16th June) **[Presentation Slides]**

Working with animations. Learn about implicit animations in Flutter and widgets that facilitate them. Code the working of an animation controller. Explore a third party tool Rive to design and integrate animations into Flutter.

Assignment: Animate a chat app to practice implicit animations.

Week 8 (A. Umer)(23rd June) **[Presentation Slides]**

Mobile apps increasingly deal with sensitive data. Learn different aspects of strengthening the security of Flutter based apps. Root check, SSL pinning, encryption and more.

CodeMagic: Continuous integration and continuous delivery for Flutter. First thing is how artifacts such as .apk and .ipa are generated in Flutter next learn to build, test and deliver using this third party tool that lets you automate these things and direct your focus to things of more concern.

Assignment: Integrate and automate the testing and builds of all the apps built so far in previous weeks using CodeMagic. Also implement applicable aforementioned security mechanisms on the same apps to practice how it's done in Flutter.

Week 9 (Farhan)(30 June) **[Presentation Slides]**

Flutter uses a flexible system that allows you to call platform-specific APIs whether available in Kotlin or Java code on Android, or in Swift or Objective-C code on iOS. This platform-specific API support is based on a flexible message passing style. Learn about Flutter platform-channels.

Assignment: Build an application that uses platform-channels to implement platform specific functionality such as quick settings tile for your app in Android.

Week 10 (Vinod)(7 July) **[Presentation Slides]**

Essential part of being a community is giving back. Packages enable the creation of modular code that can be shared easily. Learn to write Flutter packages and plugins.

Need to set up product flavors for different development environments or release types? Learn about implementing flavors in Flutter, android flavors and iOS schema.

https://flutter.dev/docs/deployment/flavors

Assignment: Design and develop creative and reusable custom controls and expose them as a Flutter package on https://pub.dev/

Reference Links

Book
https://tinyurl.com/wydfyn9
Course
https://www.appbrewery.co/p/flutter-development-bootcamp-with-dart (paid)
https://www.appbrewery.co/p/flutter-development-bootcamp-with-dart1 (free)

Dart language tour
https://dart.dev/guides/language/language-tour
Introduction to widgets
https://flutter.dev/docs/development/ui/widgets-intro
Building layouts
https://flutter.dev/docs/development/ui/layout
https://flutter.dev/docs/development/ui/layout/tutorial
Stateless vs stateful widgets
https://flutter.dev/docs/development/ui/interactive
Responsive design
https://flutter.dev/docs/development/ui/layout/responsive
Network requests
https://flutter.dev/docs/cookbook/networking/fetch-data
JSON serialization
https://flutter.dev/docs/development/data-and-backend/json
Persistence
https://flutter.dev/docs/cookbook/persistence
Testing
https://flutter.dev/docs/testing
State management
https://flutter.dev/docs/development/data-and-backend/state-mgmt/simple
Animations
https://flutter.dev/docs/development/ui/animations
Platform specific code
https://flutter.dev/docs/development/platform-integration/platform-channels
Developing packages
https://flutter.dev/docs/development/packages-and-plugins/developing-packages
Creating flavours
https://flutter.dev/docs/deployment/flavors