HARE KRISHNA

WORKSHOP SCHEDULE

15 SEPTEMBER, 2023

08:00 - 08:30 AM Introduction to Flutter & Dart

08:30 - 09:00 AM Creating & Running Your First Flutter App and Understanding the Directory & Files Architecture

09:00 - 12:00 PM Writing Your Own Flutter Code

- Import Statements
- void main()
- runApp()
- MaterialApp(), CupertinoApp(), WidgetsApp()
- Stateful & Stateless Widgets (stf & stl)
- Widget & Property Structure
- Life Cycle of A Stateful Widget
- Media Query - For Sizes
- Basic UI Components -
  - Text, TextStyle()
  - Scaffold()
    - AppBar()
    - FloatingActionButton()
    - Drawer()
    - BottomNavigationBar()
  - Card(), SizedBox(), Container()
  - Padding(), EdgeInsets(), Center(), Align()
  - *** Row(), Column(), Stack() -> Multi Child Widgets
  - Buttons -> Elevated, Outlined, Material

SingleChildScrollView()

ListView(), GridView()

*** Images() -> assets, memory, network, file

Button OnClick

Add Toast

Adding A Dependency

Adding Font, Adding Images & Assets to Project

---------- BREAK ----------

01:00 - 02:00 PM

UNDERSTANDING REST API & REST API ARCHITECTURE

HOW TO SIMPLY MAKE AN API CALL IN FLUTTER

NAVIGATION & ROUTING IN FLUTTER

02:00 - 04:00 PM

SIMPLE POKEMON APP IN FLUTTER DEMONSTRATING ABOVE TOPICS

---------- DAY 01 ENDS HERE ----------

16 SEPTEMBER, 2023

07:30 - 08:00 AM - REVISON OF TOPICS OF 15 SEPTEMBER, 2023

08:00 - 09:30 AM - DART BASICS

- VARIABLES
- DATA TYPES
- RULES FOR DECLARING VARIABLES
- OPERATORS
  - ARITHMATIC
  - LOGICAL
  - RELATIONSHIP
- CONDITIONAL STATEMENTS
  - IF ELSE ELSE IF
  - SWITCH CASE
  - BREAK, CONTINUE, RETURN STATEMENT
- LOOPS
  - FOR LOOP, FOR EACH
  - WHILE LOOP, DO .. WHILE LOOP
- FUNCTIONS
  - FUNCTIONS
    - 4 TYPES OF FUNCTIONS
    - RECURSIVE FUNCTION
- TRY CATCH EXCEPT

09:30 - 10:00 AM - PRACTICE TEST FOR DART BASICS

10:00 - 10:30 AM - BREAK

10:30 - 11:30 AM - OOPS CONCEPTS

- CLASS
- OBJECTS
- INHERITENCE
- FUNCTION OVERLOADING
- FUNCTION OVERRIDING

CONSTRUCTOR

11:30 AM - 12:00 PM - PRACTICE TEST FOR DART OOPS

12:00 PM - 01:00 PM - LUNCH BREAK

01:00 PM - 02:30 PM - PORTFOLIO APP MAKING

UNDERSTANDING GIT

GIT COMMANDS

CLONING A PROJECT

MAKING CHANGES IN EXISTING PROJECT

DEPLOYING A PROJECT

02:30 - 03:15 PM - BASICS OF FIREBASE & DEPLOYMENT OF PROJECT

03:15 - 03:30 PM - BUFFER TIME & BREAK

03:30 - 04:30 PM - QUESTION & ANSWERS FROM ME

HOW TO APPLY FOR FLUTTER INTERSHIP

HOW TO APPLY FOR FLUTTER JOB

HOW TO PRACTICE

WHERE TO APPLY

ETC ETC ETC

04:30 - 05:00 PM - EXTRA TIME - BUFFER TIME (WORKSHOP ENDS)

THANKING YOU

HARE KRISHNA

DAY 02: 16 SEPETEMBER 2023

Life Cycle of a Stateful Widget

Media Query - For Sizes

Basic UI Components –

1. Text(), TextStyle()

Text widget is used to display a piece of string. The style of the string can be set by using style property and TextStyle class. The sample code for this purpose is as follows:

```
Text('Hello World!', style: TextStyle(fontWeight: FontWeight.bold))
```

2. Image

Image widget is used to display an image in the application. Image widget provides different methods to load images from multiple sources and they are as follows:

Image.asset - Load image from flutter project's assets

```
Image.asset('assets/smiley.png', height: 200, width: 200)
```

Image.file - Load image from system folder

```
Image.file(C:/Users/Manish/Desktop/test.png', height: 200, width: 200)
```

Image.Network - Load image from network

```
Image.network('https://manishtalreja.com/test.png', height: 200, width: 200)
```

Image.memory - Load image from memory

3. Icon

Icon widget is used to display a glyph from a font described in IconData class. The code to load a simple email icon is as follows:

```
Icon(Icons.email)
```

4. Buttons

Old Widget	Old Theme	New Widget	New Theme
FlatButton	ButtonTheme	TextButton	TextButtonTheme
RaisedButton	ButtonTheme	ElevatedButton	ElevatedButtonTheme
OutlineButton	ButtonTheme	OutlinedButton	OutlinedButtonTheme

A. TextButton()

```
TextButton(
  style: TextButton.styleFrom(
    primary: Colors.red, // foreground
  ),
  onPressed: () { },
  child: Text('TextButton with custom foreground'),
)
```

B. Elevated Button

```
ElevatedButton(
  style: ElevatedButton.styleFrom(
    primary: Colors.red, // background
    onPrimary: Colors.white, // foreground
  ),
  onPressed: () { },
  child: Text('ElevatedButton with custom foreground/background'),
)
```

C. Outlined Button

```
OutlinedButton(
  style: OutlinedButton.styleFrom(
    shape: StadiumBorder(),
    side: BorderSide(
      width: 2,
      color: Colors.red
```

```
    ),
  ),
  onPressed: () { },
  child: Text('OutlinedButton with custom shape and border'),
)
```

FloatingActionButton()

1. Small

```
FloatingActionButton.small(
            onPressed: () {
              // Add your onPressed code here!
            },
            child: const Icon(Icons.add),
          ),
```

2. Regular

```
FloatingActionButton(
            onPressed: () {
              // Add your onPressed code here!
            },
            child: const Icon(Icons.add),
          ),
```

3. Large

```
FloatingActionButton.large(
            onPressed: () {
              // Add your onPressed code here!
            },
            child: const Icon(Icons.add),
          ),
```

4. Extended

```
FloatingActionButton.extended(
            onPressed: () {
              // Add your onPressed code here!
            },
            label: const Text('Add'),
            icon: const Icon(Icons.add),
          ),
```

5. TextField()

```
SizedBox(
    width: 250,
    child: TextField(
      obscureText: true,
      keyboardType: TextInputType.number,
      decoration: InputDecoration(
        border: OutlineInputBorder(),
        labelText: 'Password',
      ),
    ),
  )
```

1. Keyboard Type

A TextField allows you to customise the type of keyboard that shows up when the TextField is brought into focus. We change the keyboardType property for this.

The types are:

- TextInputType.text (Normal complete keyboard)
- TextInputType.number (A numerical keyboard)
- TextInputType.emailAddress (Normal keyboard with an “@”)
- TextInputType.datetime (Numerical keyboard with a “/” and “:”)
- TextInputType.numberWithOptions (Numerical keyboard with options to enabled signed and decimal mode)
- TextInputType.multiline (Optimises for multi-line information)

Full TextField Document:

https://medium.com/flutter-community/a-deep-dive-into-flutter-textfields-f0e676aaab7a

6. ListTile

```
ListTile(
        title: const Text('ListTile with Hero'),
        leading: CircleAvatar(child: Text('C')),
        trailing: Icon(Icons.favorite_rounded),
        subtitle: const Text('Tap here for Hero transition'),
        tileColor: Colors.cyan,
        onTap: () {

        },
      ),
```

In Flutter, ListView is a scrollable list of widgets arranged linearly. It displays its children one after another in the scroll direction i.e, vertical or horizontal.

There are different types of ListViews :

ListView

ListView.builder

Properties of ListView Widget:

scrollDirection: This property takes in Axis enum as the object to decide the direction of the scroll on the ListView.

shrinkWrap: This property takes in a boolean value as the object to decide whether the size of the scrollable area will be determined by the contents inside the ListView.

A . ListView.builder()

The builder() constructor constructs a repeating list of widgets. The constructor takes two main parameters:

An itemCount for the number of repetitions for the widget to be constructed (not compulsory).

An itemBuilder for constructing the widget which will be generated ‘itemCount‘ times (compulsory).

```
ListView.builder(
        itemCount: 20,
        itemBuilder: (context, position) {
            return Card(
            child: Padding(
                padding: const EdgeInsets.all(20.0),
                child: Text(
                position.toString(),
                style: TextStyle(fontSize: 22.0),
                ),
            ),
            );
        },
        ),
```

Flutter GridView is a widget that is similar to a 2-D Array in any programming language. As the name suggests, a GridView Widget is used when we have to display something on a Grid. We can display images, text, icons, etc on GridView. We can implement GridView in various ways in Flutter :

GridView.count()

GridView.builder()

GridView.custom()

GridView.extent()

```
GridView.count(
        crossAxisCount: 2,
        crossAxisSpacing: 10.0,
        mainAxisSpacing: 10.0,
        shrinkWrap: true,
        children: List.generate(20, (index) {
          return Padding(
            padding: const EdgeInsets.all(10.0),
            child: Container(
              decoration: BoxDecoration(
                image: DecorationImage(
                  image: NetworkImage('img.png'),
                  fit: BoxFit.cover,
                ),
                borderRadius:
                BorderRadius.all(Radius.circular(20.0),),
              ),
            ),
          );
        },),
      ),
    ),
```

The Stack widget has two types of child one is positioned which are wrapped in the Positioned widget and the other one is non–positioned which is not wrapped in the Positioned widget. For all the non-positioned widgets the alignment property is set to the top-left corner.

Properties of Stack Widget:

alignment: This property takes a parameter of Alignment Geometry, and controls how a child widget which is non-positioned or partially-positioned will be aligned in the Stack.

clipBehaviour: This property decided whether the content will be clipped or not.

fit: This property decided how the non-positioned children in the Stack will fill the space available to it.

overflow: This property controls whether the overflow part of the content will be visible or not,

textDirection: With this property, we can choose the text direction from right to left. or left to right.

```
Stack(
            children: <Widget>[
              Container(
                width: 300,
                height: 300,
                color: Colors.red,
              ), //Container
              Container(
                width: 250,
                height: 250,
                color: Colors.black,
              ), //Container
              Container(
                height: 200,
                width: 200,
                color: Colors.purple,
              ), //Container
            ], //<Widget>[]
          ),
```