

Flutter & Dart Complete Guide

Prepared by Eng. Yousef Salah Kassem

DAFL 2
Flutter UI Part 3

Lecture 3



- Introduction
- Tab Navigation
- Drawer Navigation
- Bottom NavBar Navigation
- Alert Dialog
- CheckBox
- RadioButton
- Indicators



Types of navigation you must know

When you develop apps, you must notice the basic three types of navigation in detail. However, developers are keen on guiding the use of the navigation types accordingly.

- Stack Navigation
- Tab Navigation
- Drawer Navigation

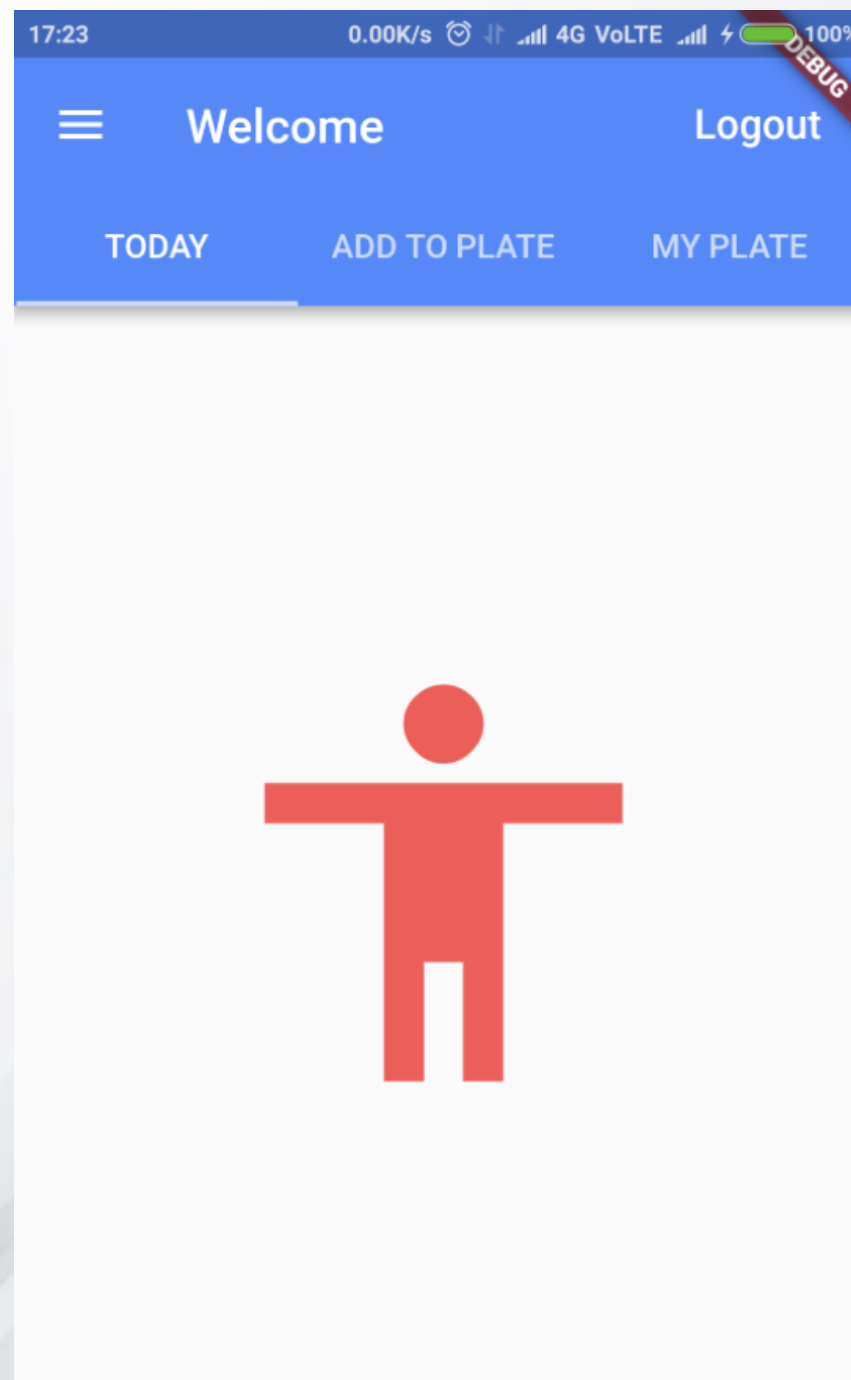


Tab Navigation

Tab Navigation

- tabs are a staple in the design or development of mobile app navigation. Hence, it allows users to find out switches and enable them to focus on the current context. The flutter makes it easy to find tabbed
- navigation within built-in widgets such as TabBar, and TabBarView. They are integrating into the beautiful and functional tab navigation
- experience. It is mainly perfect for organizing content into logical sections.



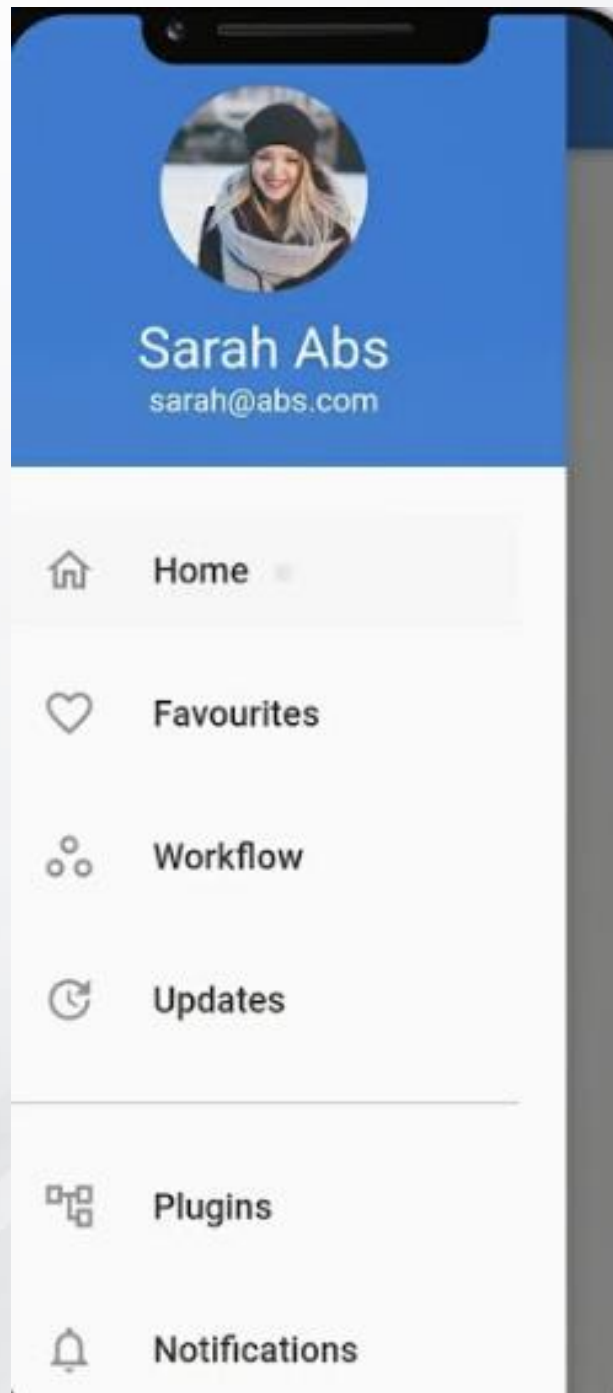


Drawer Navigation

Drawer Navigation

- The drawer navigation is nothing but a pattern which includes a hamburger menu or side menu. It is a familiar navigation style which includes the
- design and development of mobile apps. It consists of hidden panels which slide out with the screen or reveal the menu with dynamic navigation options. Hence, it consists of a space-saving
- technique which is visible and provides easy access.



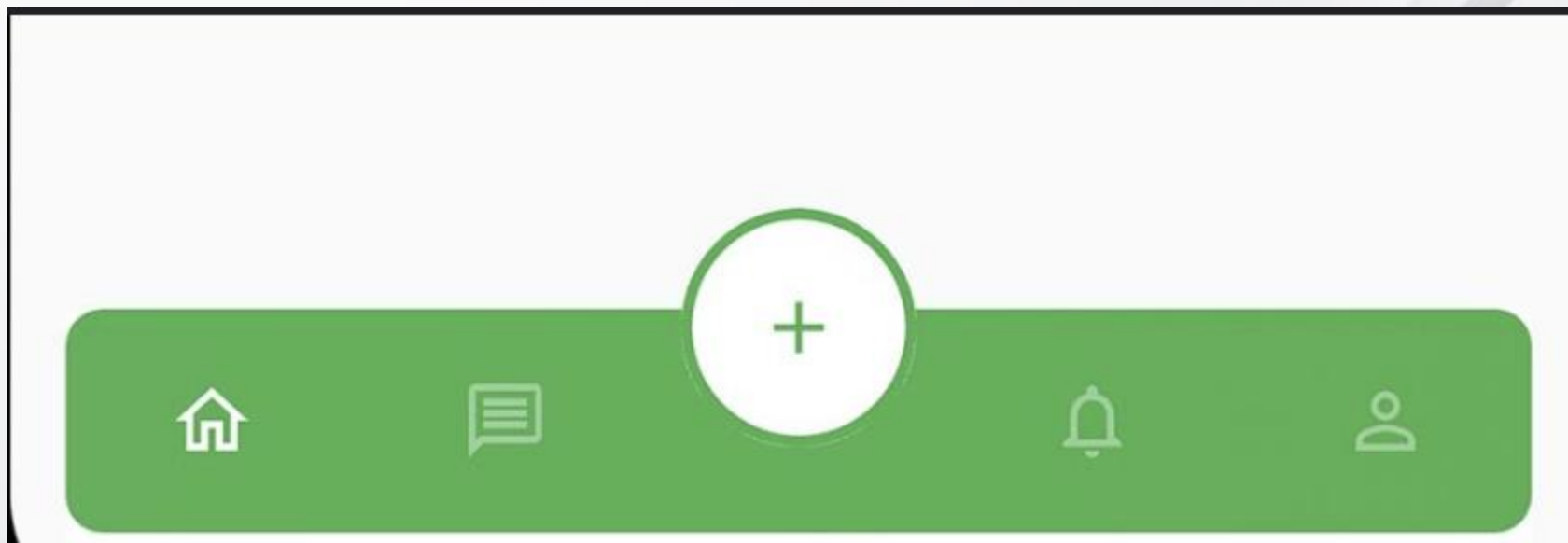


Bottom NavBar Navigation

Bottom NavBar Navigation

- Bottom Navigation Bar is a navigation component displaying three to five destinations at the bottom of a screen. Each destination is usually represented by an icon and an optional text label. When a bottom navigation icon is tapped, the user is taken to the top-level navigation destination associated with that icon.





Types of navigation you must know

When you develop apps, you must notice the basic three types of navigation in detail. However, developers are keen on guiding the use of the navigation types accordingly.

- Stack Navigation
- Tab Navigation
- Drawer Navigation



Alert Dialog

Alert Dialog

- An alert dialog is a useful feature that notifies the user with important information to make a decision or provide the ability to choose a specific action or list of actions.
- It is a pop-up box that appears at the top of the app content and the middle of the screen.
- It can be dismissed manually by the user before resuming interaction with the app.



Alert Dialog

- An alert can be thought of as a floating modal which should be used for a quick response such as password verification, small app notifications, and many more.
- The alerts are very flexible and can be customized very easily.
- In Flutter, the AlertDialog is a widget, which informs the user about the situations that need acknowledgment.
- The Flutter alert dialog contains an optional title that displayed above the content and list of actions displayed below the content.



- Title: This property gives the title to an AlertDialog box that occupies at the top of the AlertDialog.
- It is always good to keep the title as short as possible so that the user knows about its use very easily. We can write the title in AlertDialog as below:

```
AlertDialog(title: Text("Sample Alert Dialog"),
```



- Action: It displays below the content. For example, if there is a need to create a button to choose yes or no, then it is defined in the action property only.
- We can write an action attribute in AlertDialog as below:

```
actions: <Widget>[ FlatButton(child: Text("Yes"),),  
FlatButton(child: Text("No"),)  
,)
```



Content: This property defines the body of the AlertDialog widget. It is a type of text, but it can also hold any kind of layout widgets.

We can use the Content attribute in AlertDialog as below:

```
actions: <Widget>[ FlatButton(child: Text("Yes"),), FlatButton(child:  
Text("No"),)
```

```
],)
```

```
content: Text("It is the body of the alert Dialog"),
```



ContentPadding: It gives the padding required for the content inside the **AlertDialog** widget.

We can use Content Padding attribute in **AlertDialog** as below:
`contentPadding: EdgeInsets.all(32.0),`



Shape: This attribute provides the shape to the alert dialog box, such as curve, circle, or any other different shape.

shape: CircleBorder(),

shape: CurveBorder(),



We can categories the alert dialog into multiple types, which are given below:

- Basic AlertDialog
- Confirmation AlertDialog
- Select AlertDialog
- TextField AlertDialog



CheckBox

Alert Dialog

- A checkbox is a type of input component which holds the Boolean value. It is a GUI element that allows the user to choose multiple options from several selections.
- Here, a user can answer only in yes or no value. A marked/checked checkbox means yes, and an unmarked/unchecked checkbox means no value.
- Typically, we can see the checkboxes on the screen as a square box with white space or a tick mark.
- A label or caption corresponding to each checkbox described the meaning of the checkboxes.



- In Flutter, we can have two types of checkboxes: a compact version of the Checkbox named "checkbox" and the "CheckboxList Tile" checkbox, which comes with header and subtitle.



Attributes	Descriptions
value	It is used whether the checkbox is checked or not.
onChanged	It will be called when the value is changed.
Tristate	It is false, by default. Its value can also be true, false, or null.
activeColor	It specified the color of the selected checkbox.
checkColor	It specified the color of the check icon when they are selected.
materialTapTargetSize	It is used to configure the size of the tap target.

Radio Button

Radio Button

- A radio button is also known as the options button which holds the Boolean value.
- It allows the user to choose only one option from a predefined set of options.
- This feature makes it different from a checkbox where we can select more than one option and the unselected state to be restored.
- We can arrange the radio button in a group of two or more and displayed on the screen as circular holes with white space (for unselected) or a dot (for selected).



Radio Button

- We can also provide a label for each corresponding radio button describing the choice that the radio button represents.
- A radio button can be selected by clicking the mouse on the circular hole or using a keyboard shortcut.



Example

```
ListTile(  
  title: const Text('www.google.com'), leading: Radio(  
    value: BestSearchEngine.google, groupValue: _site,  
    onChanged: (BestSearchEngine value) {  
      setState(() { _site = value;  
    }); },  
  ), ),
```



Indicators

A progress bar is a graphical control element used to show the progress of a task such as downloading, uploading, installation, file transfer, etc.

- Flutter can display the progress bar with the help of two widgets, which are given below:
 - LinearProgressIndicator
 - CircularProgressIndicator



Flutter provides mainly two types of linear progress indicators:

- **Determinate:**
 - Determinate progress bar indicates the actual amount of progress at each point in making the task.
 - Its value will increase monotonically from 0.0 to 1.0 to show the amount of task completed at that time.
 - We need to use a non-null value from 0.0 to 1.0 for creating a determinate progress indicator.



Indeterminate:

- Indeterminate progress bar does not indicate the amount of progress in completing the task.
- It means we do not know when the task is finished. It makes progress without indicating how much progress remains.
- We can make an indeterminate progress indicator by using a null value.



A slider in Flutter is a material design widget used for selecting a range of values.

- It is an input widget where we can set a range of values by dragging or pressing on the desired position.
- Usually, we use the slider widget for changing a value. So, it is required to store the value in a variable.
- This widget has a slider class that requires the `onChanged()` function. This function will be called whenever we change the slider position.



A slider can be used for selecting a value from a continuous or discrete set of values. By default, it uses a continuous range of values.

- If we want to use discrete values, we must have to use a non-null value for divisions. This discrete division indicates the number of discrete intervals.
- Before getting the value, we have to set the minimum and maximum value. Slider provides min and max arguments to set the minimum and maximum limitations.
- For example, we have a set of values from 0.0 to 50.0, and divisions are 10, the slider would take values like 0.0, 10.0, 20.0, 30.0, 40.0, and 50.0.

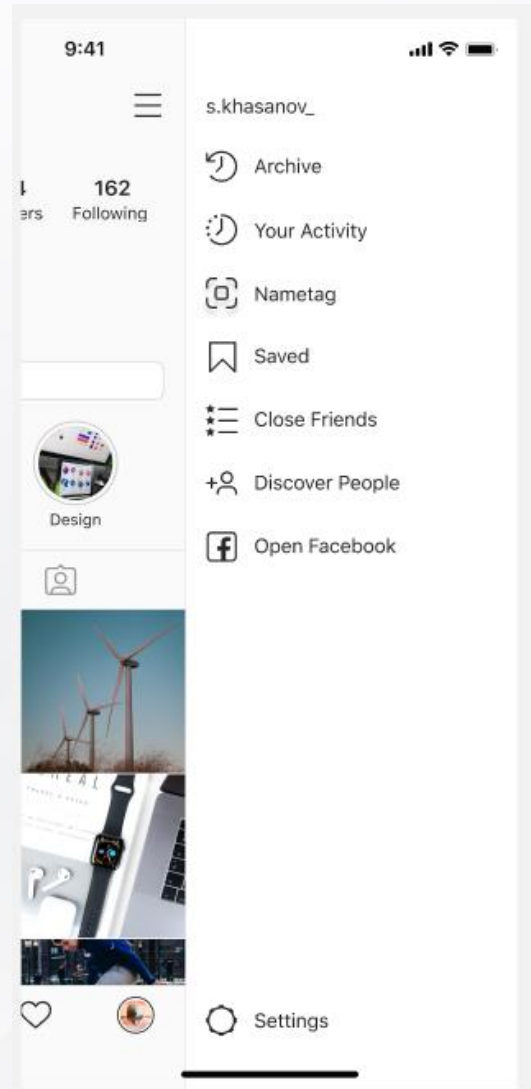


Slider

```
Slider(  
  min: 0,  
  max: 100,  
  value: _value,  
  onChanged: (value) {  
    setState(() {  
      _value = value;  
    });  
  },  
),
```



Live Coding



What is Next ?



Thank You