

# Cheat Sheet: Working with Flutter

## Interaction and forms in Flutter

Type	Description	Code example
ElevatedButton	Triggers actions on user interaction	<pre>ElevatedButton(   onPressed: () { print('Button Pressed'); },   child: Text('Press Me') );</pre>
TextField	Collects textual input from users	<pre>TextField(   decoration: InputDecoration(hintText: 'Enter your username') );</pre>
Form validation	Ensures input meets certain criteria	<pre>Form(   child: TextFormField(     validator: (value) {       if (value.isEmpty) return 'Cannot be empty';     }   ) );</pre>
GestureDetector	Handles more complex gestures like swipes	<pre>GestureDetector(   onTap: () { print('Tapped!'); },   child: Container(color: Colors.blue, width: 100, height: 100) );</pre>

## Routing in Flutter

Type	Description	Code example
Basic navigation	Manages screen transitions within an app	<pre>Navigator.push(   context,   MaterialPageRoute(builder: (context) =&gt; NewScreen()) );</pre>

Named routes	Uses named routes for navigation, which simplifies route management	<pre>Navigator.pushNamed(context, '/details');</pre>
Passing data between routes	Allows data to be sent between screens	<pre>Navigator.push(   context,   MaterialPageRoute(     builder: (context) =&gt; DetailScreen(data: 'Data from previous screen')   ) );</pre>

Implementing styles in Flutter

Type	Description	Code example
Text widget styling	Customizes the appearance of text elements	<pre>Text(   'Hello Flutter',   style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold, color: Colors.blue) );</pre>
Styling buttons	Modifies the look of buttons to align with the app design	<pre>ElevatedButton(   style: ElevatedButton.styleFrom(primary: Colors.green),   onPressed: () {},   child: Text('Submit') );</pre>

Navigation in Flutter

Type	Description	Code example
Stack navigation	Manages a stack of screens	<pre>Navigator.push(   context,   MaterialPageRoute(builder: (context) =&gt; SecondScreen()) );</pre>

Tab navigation	Organizes content across different tabs	<pre>TabBar(   tabs: [Tab(icon: Icon(Icons.directions_car)), Tab(icon: Icon(Icons.directions_transit))] );</pre>
Bottom navigation bar	Provides navigation between top-level views of an app through a series of tabs at the bottom of the screen	<pre>BottomNavigationBar(   items: &lt;BottomNavigationBarItem&gt;[     BottomNavigationBarItem(icon: Icon(Icons.home), label: 'Home'),     BottomNavigationBarItem(icon: Icon(Icons.business), label: 'Business'),   ],   onTap: (index) { /* Handle tap */ }, );</pre>
Drawer navigation	Offers a slide-in menu from the edge of the screen for navigation purposes	<pre>Drawer(   child: ListView(     children: &lt;Widget&gt;[       DrawerHeader(child: Text('Header')),       ListTile(title: Text('Item 1'), onTap: () {}),     ],   ), );</pre>

Flutter widgets

Type	Description	Code example
DropDownButton	Allows users to select from a list of items in a dropdown menu	<pre>DropDownButton&lt;String&gt;(   value: dropdownValue,   onChanged: (String newValue) {     setState() {       dropdownValue = newValue;     }   },   items: &lt;String&gt;['One', 'Two', 'Three', 'Four']     .map&lt;DropDownMenuItem&lt;String&gt;&gt;((String value) {       return DropDownMenuItem&lt;String&gt;(         value: value,         child: Text(value),       );     }).toList(), )</pre>
Flexible widget	Allows child widgets to flex within available space in a row or column	<pre>Row(children: [   Flexible(child: Container(color: Colors.red, height: 50)),   Flexible(child: Container(color: Colors.blue, height: 50)) )</pre>

<b>ListView.builder</b>	Constructs a list view dynamically with an item builder function	<pre> ListView.builder(   itemCount: items.length,   itemBuilder: (context, index) {     return ListTile(title: Text(items[index]));   }, ) </pre>
<b>SnackBar</b>	Provides lightweight feedback about an operation by showing a brief message	<pre> final snackBar = SnackBar(content: Text('Yay! A SnackBar!')); ScaffoldMessenger.of(context).showSnackBar(snackBar); </pre>
<b>Dialogs</b>	Displays material design dialogs	<pre> showDialog(   context: context,   builder: (BuildContext context) {     return AlertDialog(       title: Text('Alert!'),       content: Text('You have been alerted.'),       actions: &lt;Widget&gt;[         TextButton(           onPressed: () {             Navigator.of(context).pop();           },           child: Text('Close'),         ),       ],     );   }, ); </pre>
<b>Padding widget</b>	Applies padding to its child widget	<pre> Padding(   padding: EdgeInsets.all(8.0),   child: Text('Hello World!'), ) </pre>
<b>GridView</b>	Creates a scrollable grid of widgets arranged in rows and columns	<pre> GridView.count(   crossAxisCount: 2,   children: List.generate(20, (index) {     return Center(       child: Text('Item \$index'),     );   })), ) </pre>

<b>ExpansionPanel</b>	Provides a way to expand and collapse content	<pre> ExpansionPanellist(   expansionCallback: (int index, bool isExpanded) {     setState(() {       items[index].isExpanded = !isExpanded;     });   },   children: items.map&lt;ExpansionPanel&gt;((Item item) {     return ExpansionPanel(       headerBuilder: (BuildContext context, bool isExpanded) {         return ListTile(title: Text(item.headerValue));       },       body: ListTile(title: Text(item.expandedValue)),       isExpanded: item.isExpanded,     );   }).toList(), ) </pre>
<b>Custom paint</b>	Provides a canvas on which to draw during the paint phase	<pre> CustomPaint(   painter: MyPainter(), ) </pre>
<b>Theme data</b>	Defines the color and typography values for a Material app	<pre> ThemeData(   primarySwatch: Colors.blue,   visualDensity: VisualDensity.adaptivePlatformDensity, ) </pre>



# Skills Network