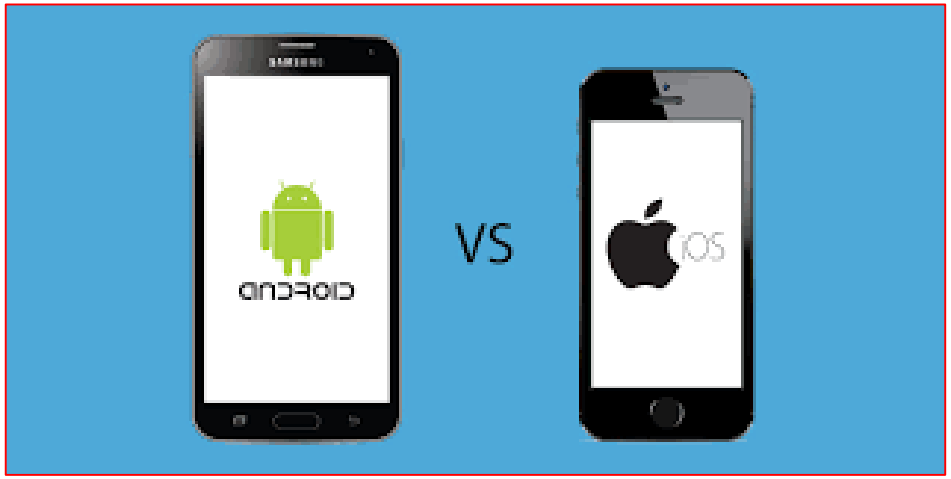
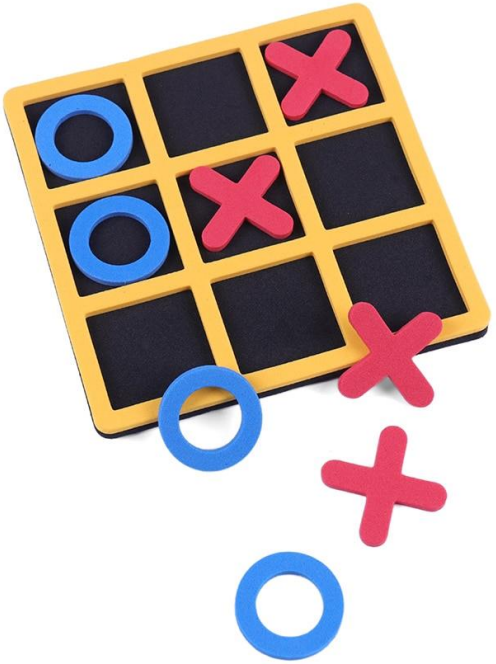
GEB208

Writing Apps for both Android and iOS Mobile Phones



Project Part Two

Game Board



Widgets Used

- TicTacToeGamePage
  - createState()
- GamePageState
  - **build(BuildContext context)** → Widget
    - Scaffold
      - AppBar
        - Text ''
      - GridView

main Function

In the main function, change the home page to TicTacToeGamePage


```
void main() {
  runApp(
    MaterialApp(
      home: TicTacToeGamePage(),
      debugShowCheckedModeBanner: false,
    ),  // MaterialApp
  );
}
```

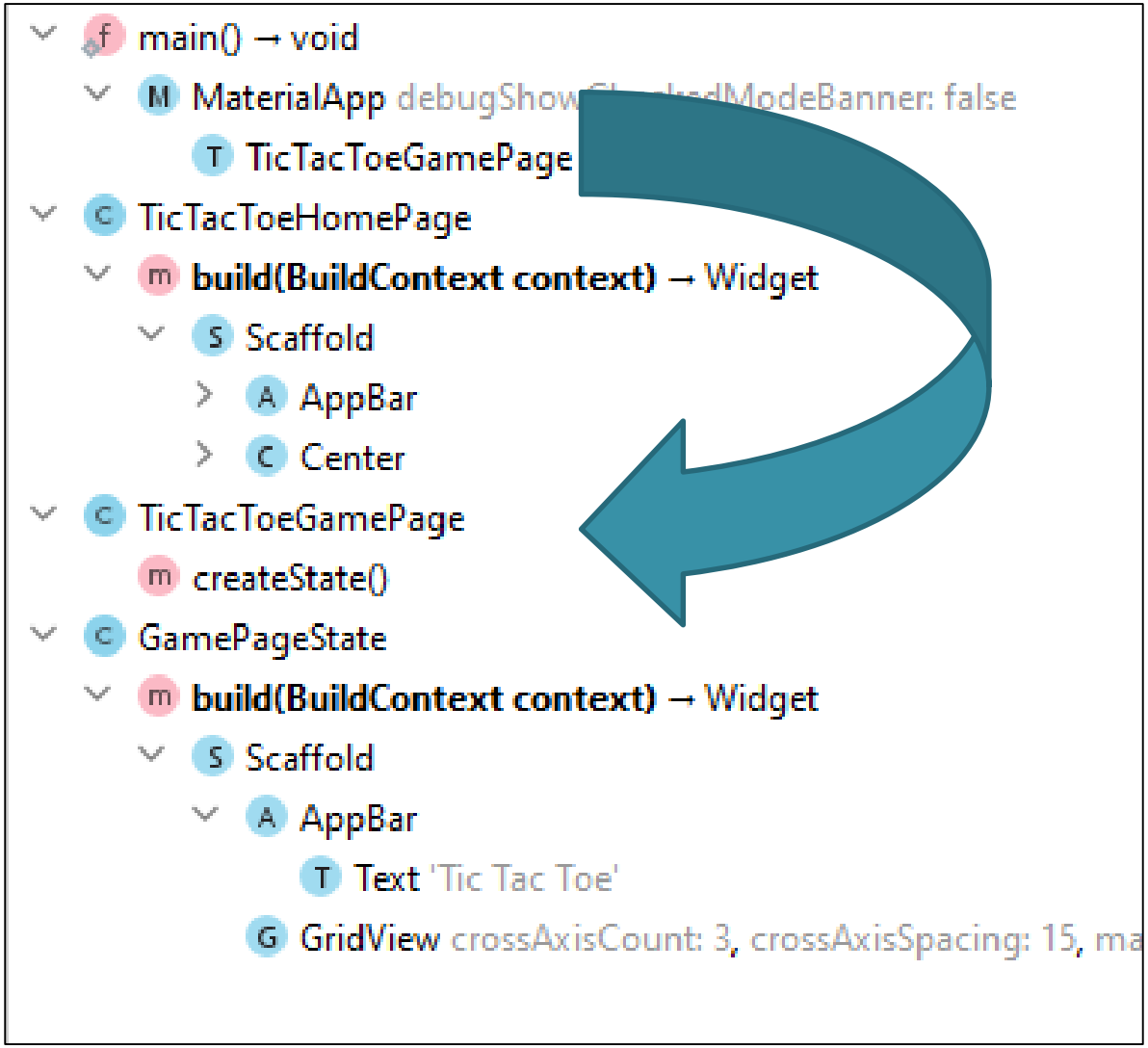
That means the class 'TicTacToeHomePage' is skip temporarily

MaterialApp Widget

- In the main function, use the MaterialApp widget
- In the class TicTacToeHomePage, return the Scaffold widget in the build widget

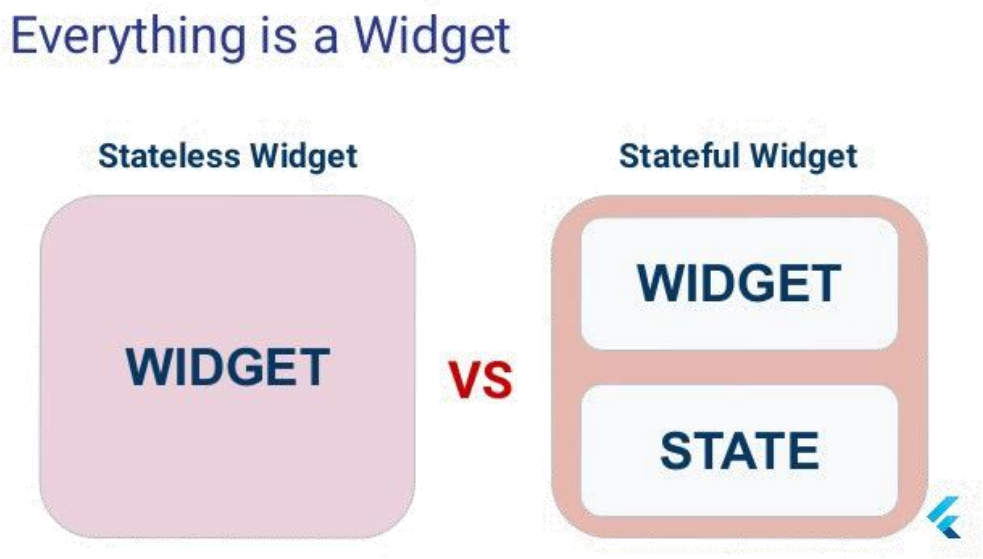
```
void main() {
  runApp(
    MaterialApp(
      home: TicTacToeGamePage(),
      debugShowCheckedModeBanner: false,
    ),  // MaterialApp
  );
}

class TicTacToeHomePage extends StatelessWidget {
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('223999 Chan Tai Man'),
```

Project Part 2 builds on Part 1



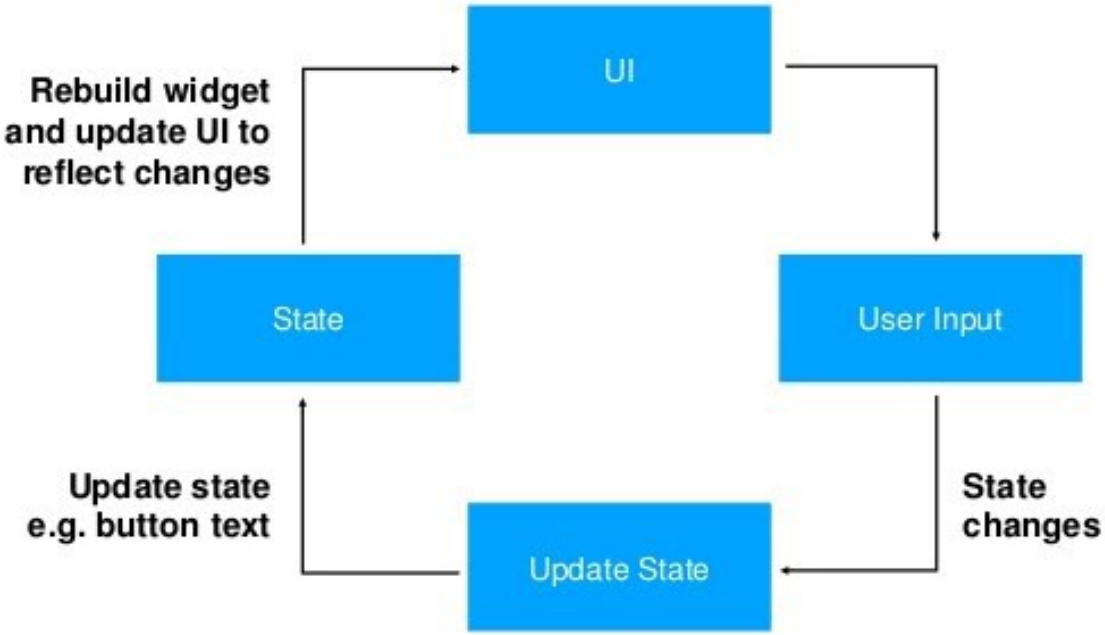
Stateful vs Stateless Widget

- A widget is either stateful or stateless
- If a widget can change — when a user interacts with it — it's stateful
- A stateless widget never changes, e.g. icon, image and text



Stateful Widget

- A stateful widget is dynamic
- It can change its appearance in response to events triggered by user interactions or when it receives data

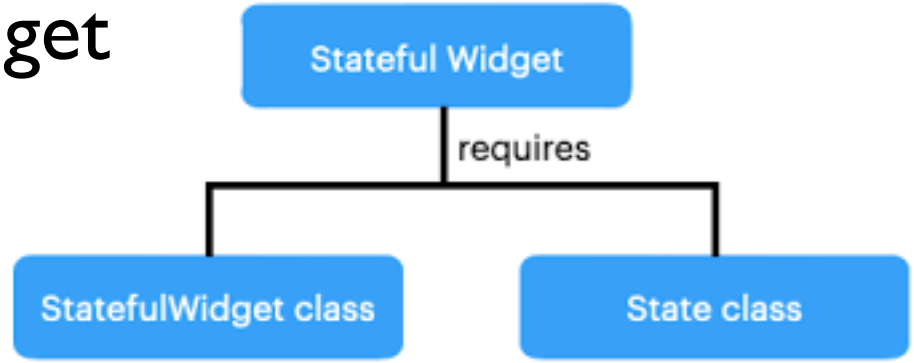


State Object

- A widget's state is stored in a State object, separating the widget's state from its appearance
- The state consists of values that can change, like a slider's current value or whether a checkbox is checked

Implement Stateful Widget

- A stateful widget is implemented by two classes: a subclass of StatefulWidget and a subclass of State
- The state class contains the widget's mutable state and the widget's build() method
- When the widget's state changes, the state object calls setState(), telling the framework to redraw the widget



TicTacToeGamePage

Extends the StatefulWidget

```
class TicTacToeGamePage extends StatefulWidget {
  createState() => GamePageState();
}
```

GamePageState

- Extends the State
- Build method returns the Scaffold

```
class TicTacToeGamePage extends StatefulWidget {
  createState() => GamePageState();
}

class GamePageState extends State {

  Widget build(BuildContext context) {
    return Scaffold(
```

Scaffold

Contains two sections:

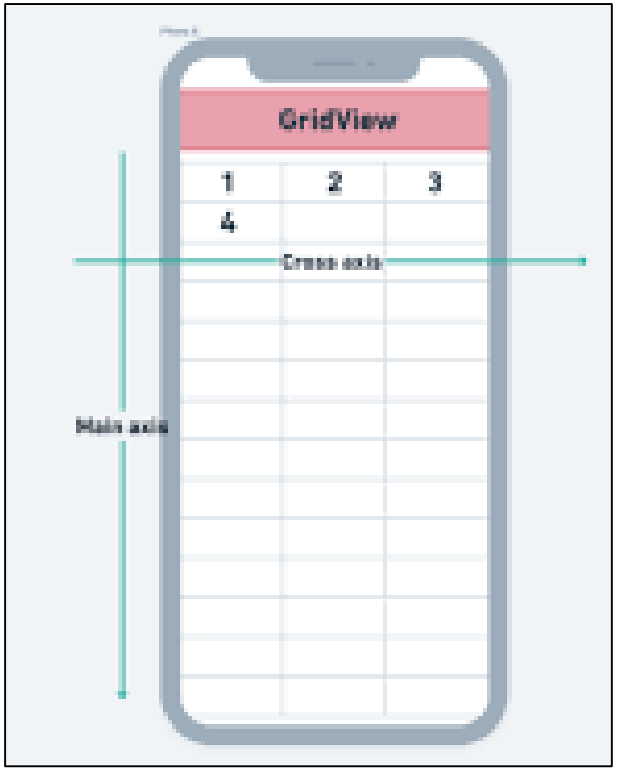
1. AppBar
2. body

AppBar

- Text widget
  - Content ‘Tic Tac Toe’
  - Apply at least two text styles

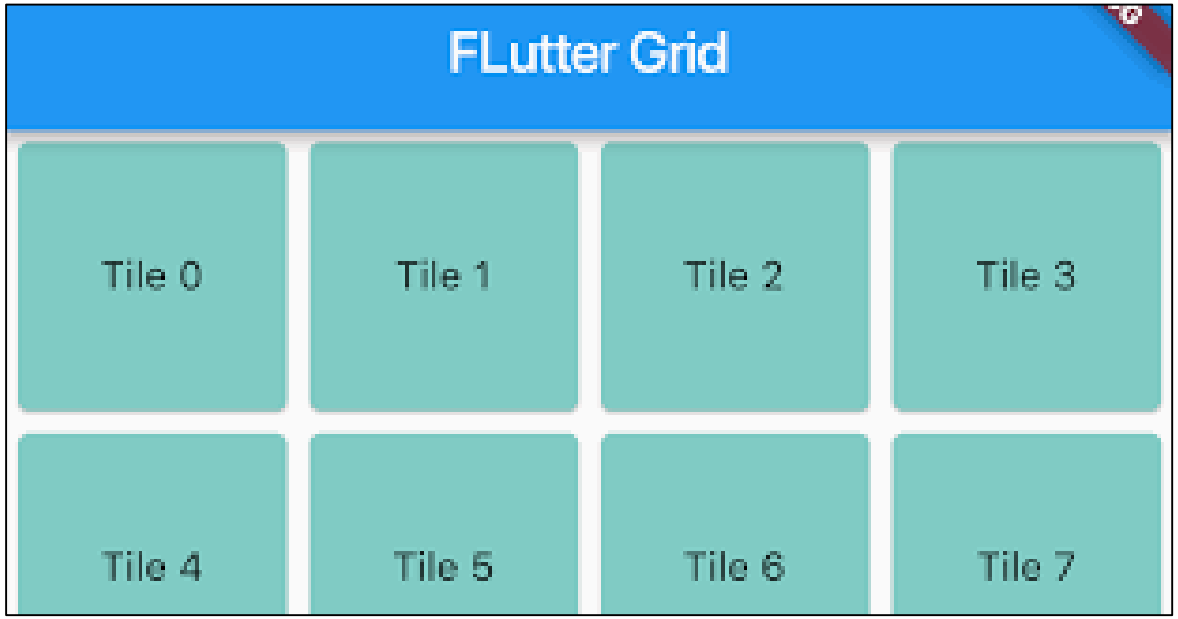
Body

- A GridView widget will be used
- GridView widget displays a list of items as a 2D array, i.e., in a table format



GridView.count (I)

The most commonly used grid layouts are GridView.count, which creates a layout with a fixed number of tiles in the cross axis



GridView.count (II)

The crossAxisCount property defines the number of item horizontally

```
body: GridView.count(
  crossAxisCount: 3,
```

GridView.count (III)

- The children property contains a List widget
- The generate method of the List widget will create a list of items

```
body: GridView.count(
  crossAxisCount: 3,
  children: List.generate(
```

List.generate (I)

- The first parameter in the List.generate is the total number of items generated
- The second parameter is a generator function that produces list values from 0 to 8, these values can be accessed by the variable index

```
body: GridView.count(
  crossAxisCount: 3,
  children: List.generate(
    9,
    (index) {
```

List.generate (II)

Inside the function, return statement is used to create the Outlined button

```
body: GridView.count(
  crossAxisCount: 3,
  children: List.generate(
    9,
    (index) {
      return OutlinedButton(
```

OutlinedButton (I)

- A Text widget is used in the OutlinedButton
- The content of the Text widget is to show the value of the variable index (i.e., 0 to 8)
- The value of the variable index is converted to a string by using string interpolation ($)

```
return OutlinedButton(
  child: Text('$index'),
```

OutlinedButton (II)

- The OnPressed parameter of the OutlinedButton will use a print function
- The print function will display the text 'Key Pressed' with the value of the index

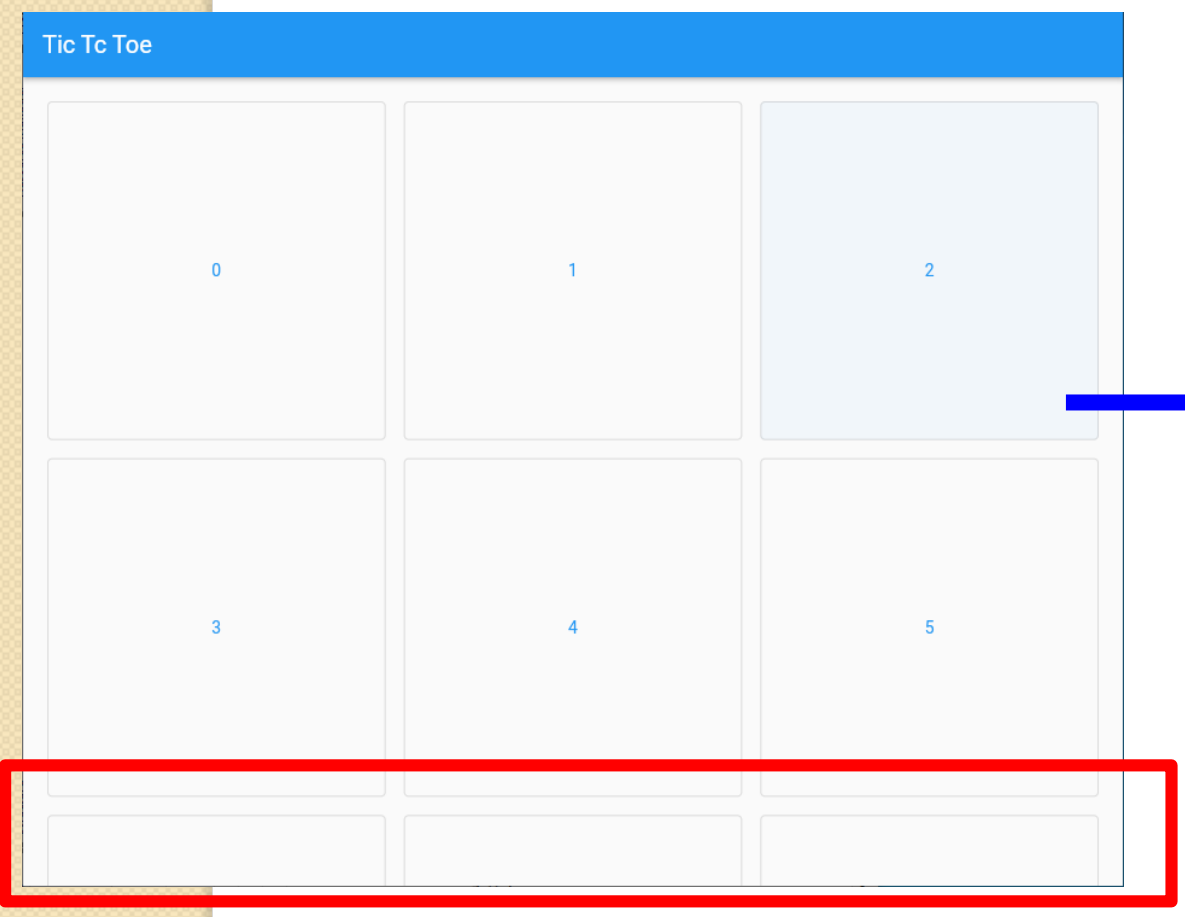
```
return OutlinedButton(
  child: Text('$index'),
  onPressed: () {
    print('Key Pressed $index');
  },
);  // OutlinedButton
```

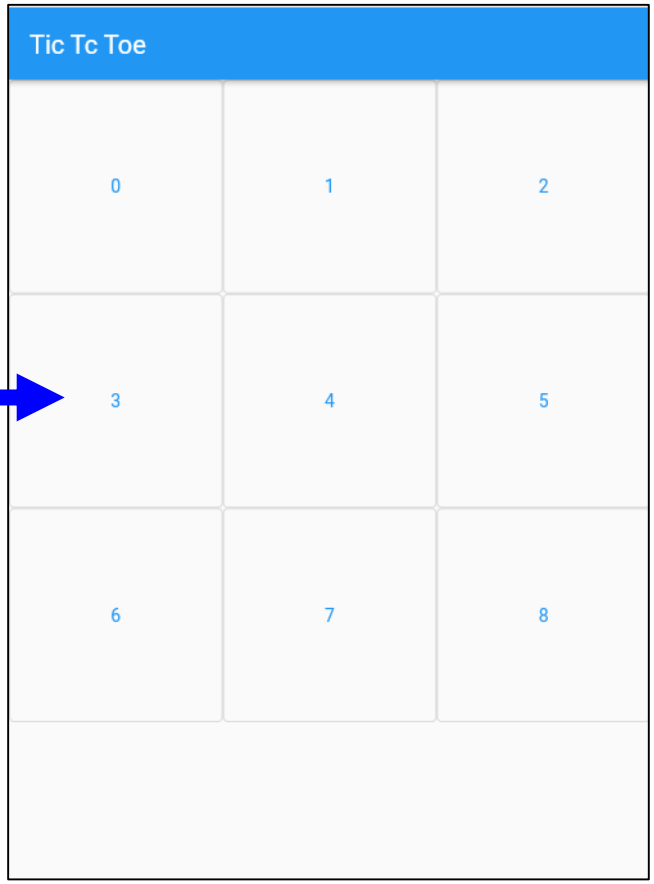
GamePageState Class

```
class GamePageState extends State {
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Tic Tc Toe'),
      ),  // AppBar
      body: GridView.count(
        crossAxisCount: 3,
        children: List.generate(
          9,
              (index) {
            return OutlinedButton(
              child: Text('$index'),
              onPressed: () {
                print('Key Pressed $index');
              },
            );  // OutlinedButton
          },
        ),  // List.generate
      ),  // GridView.count
    );  // Scaffold
  }
}
```

Run the App (I)

Reduce the width of the browser to minimum to avoid overflow





Run the App (II)

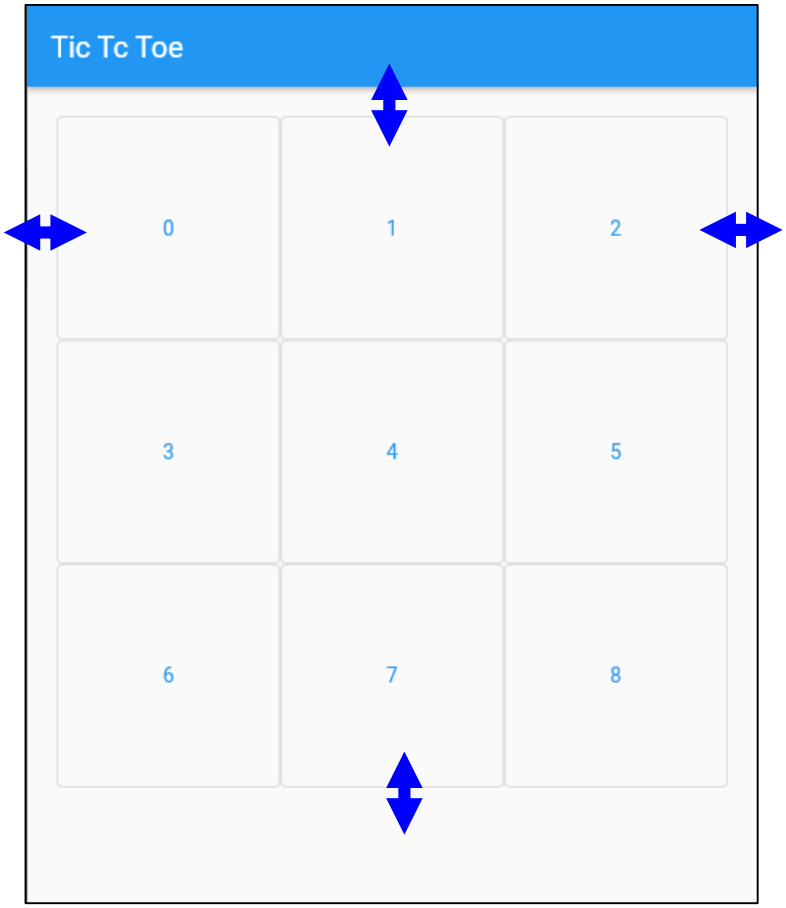
When the buttons are pressed, the console shows the message

```
Console
Debug service listening on ws://127.0.0.1:64757/N1z3iKZQ0Vs=/ws

 Running with sound null safety
Debug service listening on ws://127.0.0.1:64757/N1z3iKZQ0Vs=/ws
Key Pressed 0
Key Pressed 1
Key Pressed 2
Key Pressed 3
Key Pressed 4
Key Pressed 5
Key Pressed 6
Key Pressed 7
Key Pressed 8
```

Styling the GridView (I)

Apply padding to the GridView

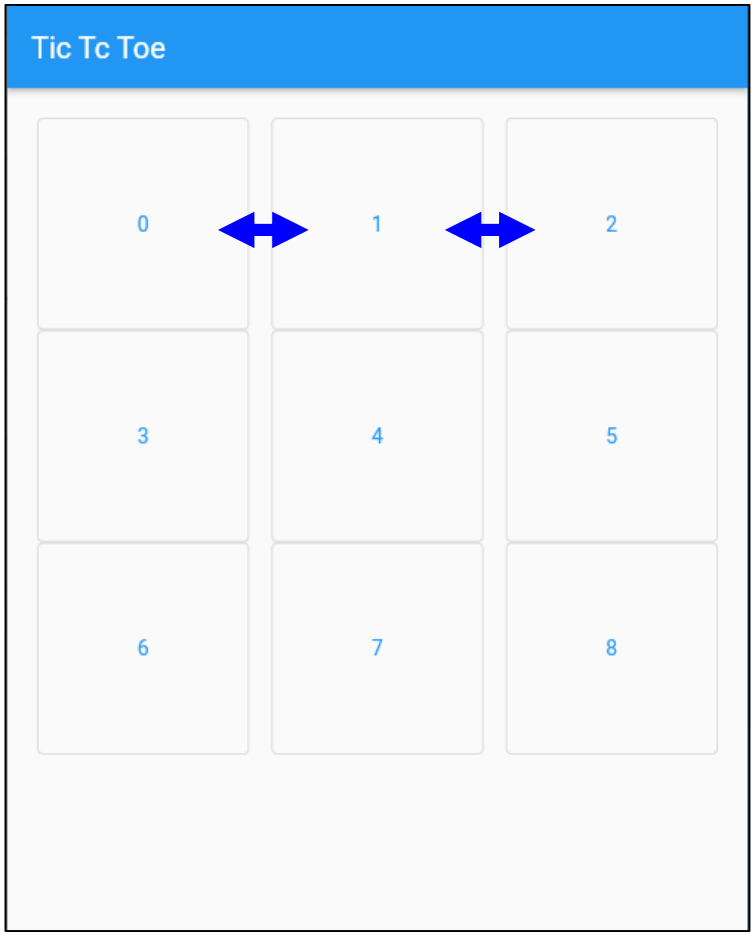
```
body: GridView.count(
  crossAxisCount: 3,
  padding: EdgeInsets.all(20),
  children: List.generate(
```



Styling the GridView (II)

Apply crossAxisSpacing to the GridView

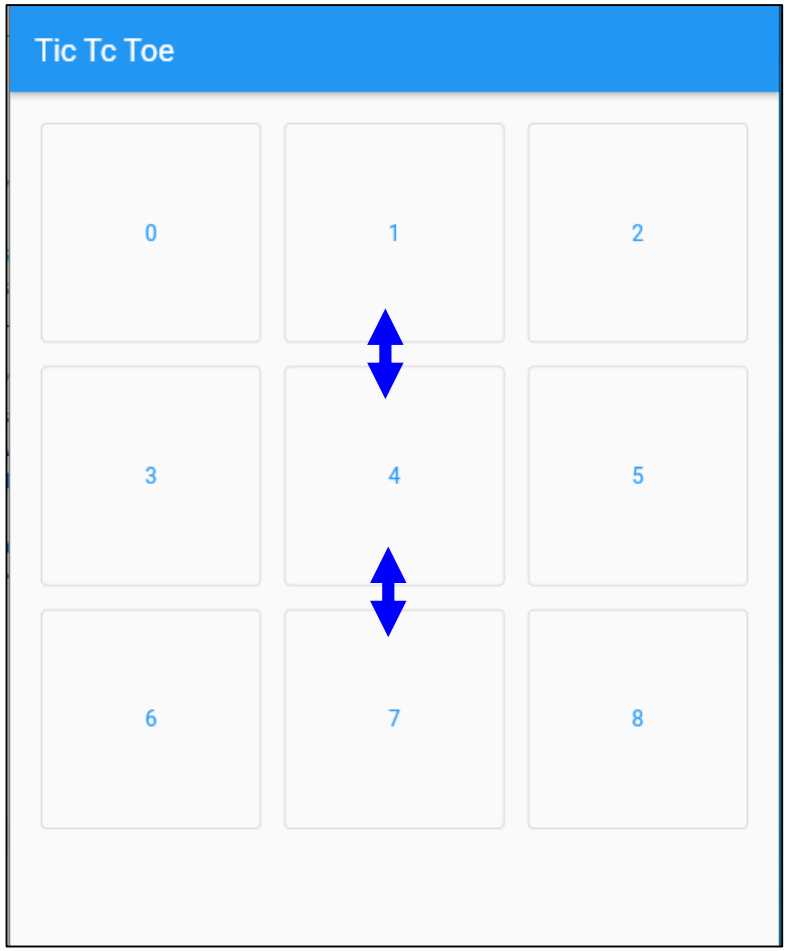
```
body: GridView.count(
  crossAxisCount: 3,
  padding: EdgeInsets.all(20),
  crossAxisSpacing: 15,
  children: List.generate(
```



Styling the GridView (III)

Apply mainAxisSpacing to the GridView

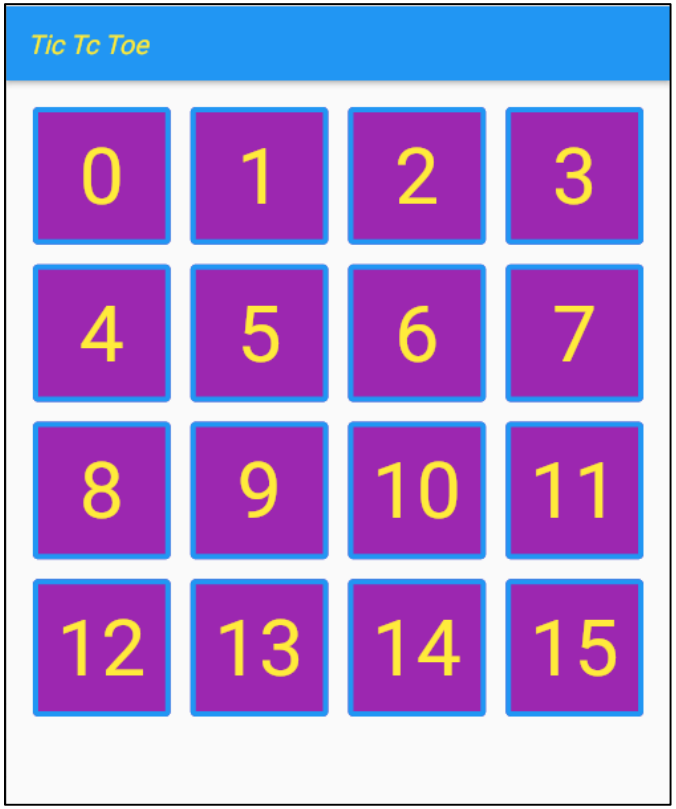
```
body: GridView.count(
  crossAxisCount: 3,
  padding: EdgeInsets.all(20),
  crossAxisSpacing: 15,
  mainAxisSpacing: 15,
  children: List.generate(
```



Sample Screen (Project)

Apply at least:

- two text styles to the AppBar
- one text style to the text in the buttons
- two styles to the buttons



END