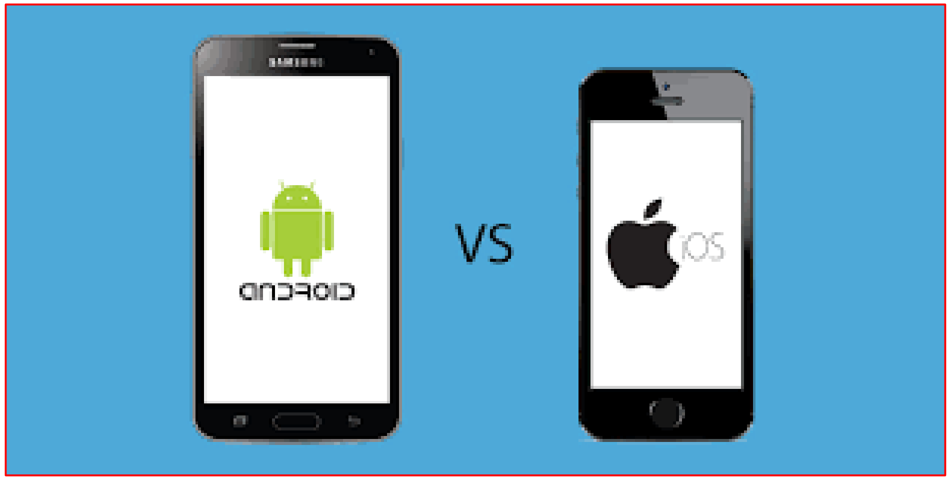
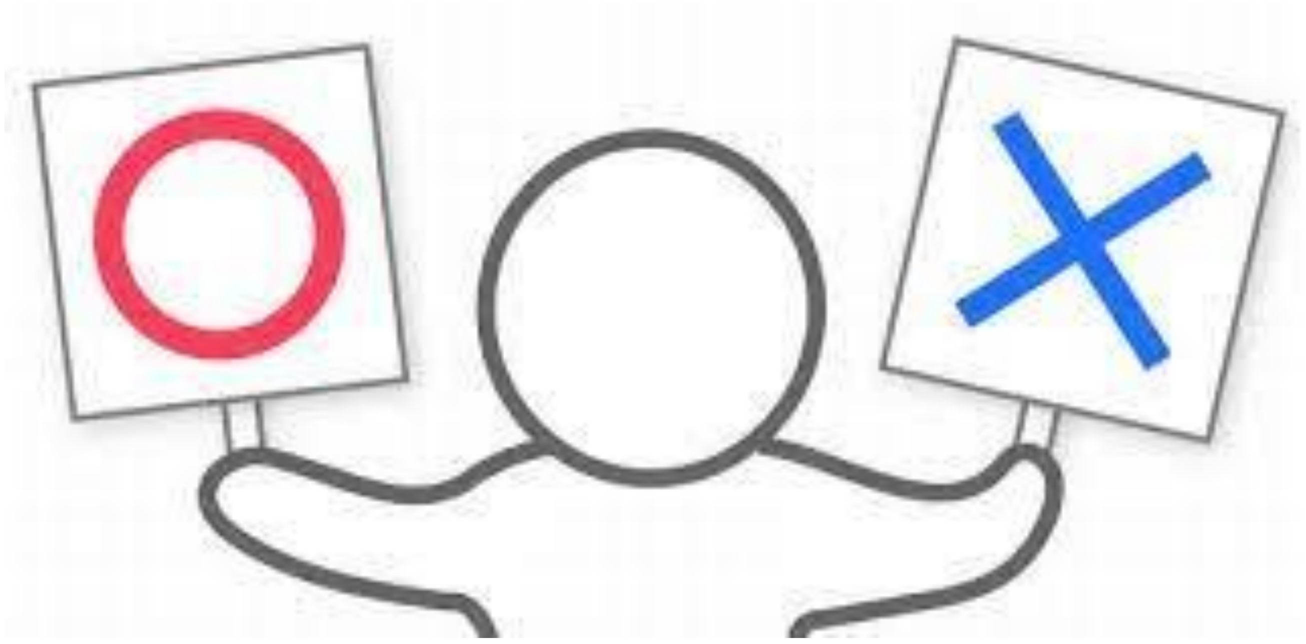
GEB208
Writing Apps for both Android and iOS Mobile Phones

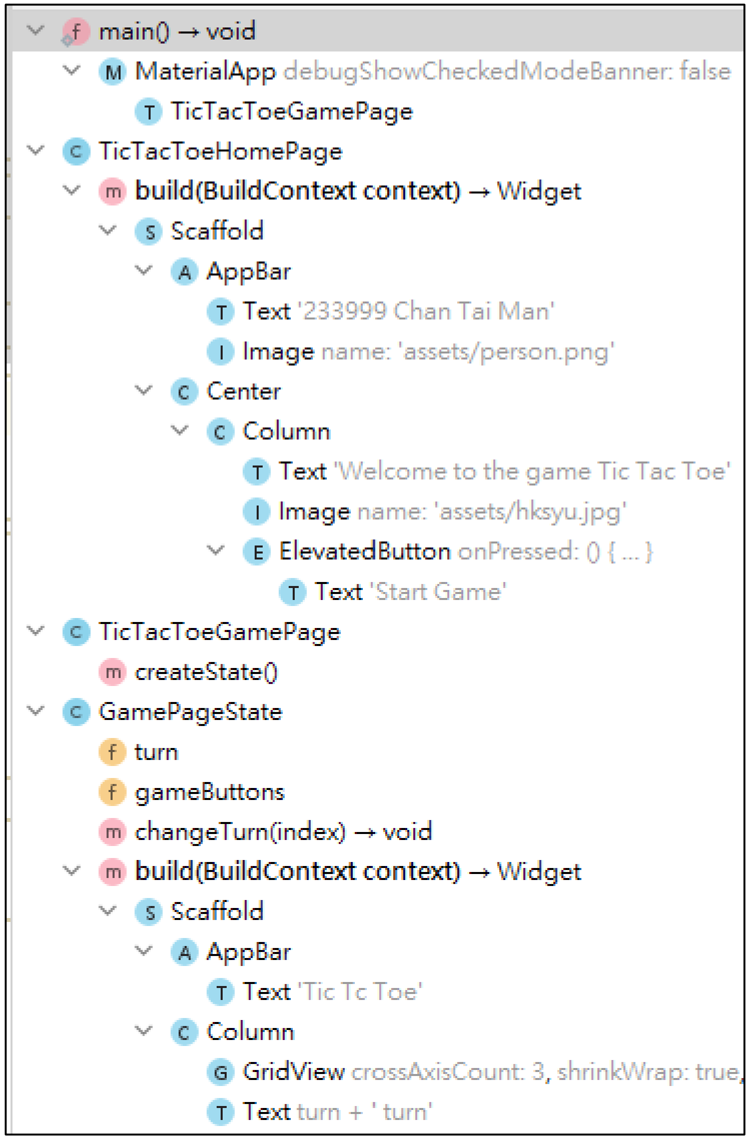


Project Part Three

Showing X or O



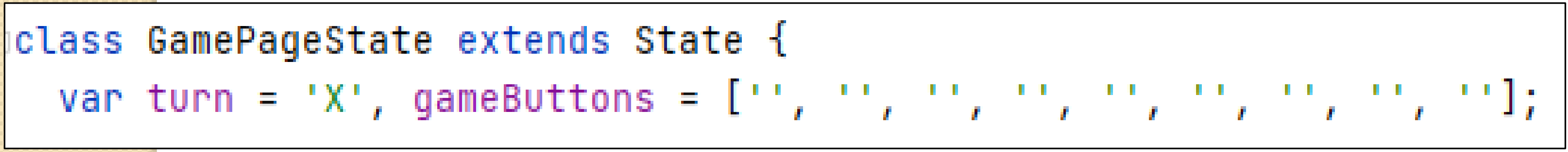
Widgets Used



```
main() → void
  MaterialApp debugShowCheckedModeBanner: false
    TicTacToeGamePage
TicTacToeHomePage
  build(BuildContext context) → Widget
    Scaffold
      AppBar
        Text '233999 Chan Tai Man'
        Image name: 'assets/person.png'
      Center
        Column
          Text 'Welcome to the game Tic Tac Toe'
          Image name: 'assets/hksyu.jpg'
          ElevatedButton onPressed: () { ... }
            Text 'Start Game'
TicTacToeGamePage
  createState()
GamePageState
  turn
  gameButtons
  changeTurn(index) → void
  build(BuildContext context) → Widget
    Scaffold
      AppBar
        Text 'Tic Tc Toe'
      Column
        GridView crossAxisCount: 3, shrinkWrap: true,
        Text turn + ' turn'
```

Variables Used

- At the beginning of the class GamePageState, two variables will be declared
- Variable ‘turn’ represents the current player, the content of it will be either ‘X’ or ‘O’
- List ‘gameButtons’ has 9 elements which represent the 9 cells on the game board, their initial value are nil string i.e.“

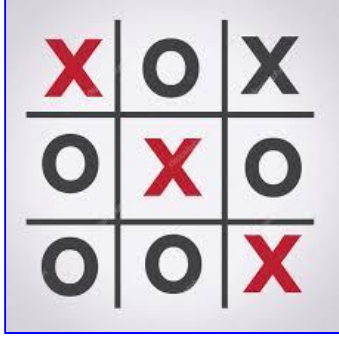


```
class GamePageState extends State {
  var turn = 'X', gameButtons = ['', '', '', '', '', '', '', '', ''];
```

Variable ‘turn’

- The initial value of the variable ‘turn’ is ‘X’ because player ‘X’ will start the game first
- After player ‘X’ makes his/her choice on the game board, the value of the variable ‘turn’ will change to ‘O’
- At the same time, the content of the variable ‘turn’ will be assigned to the corresponding element in the list ‘gameButtons’ to record which cell is chosen by the player

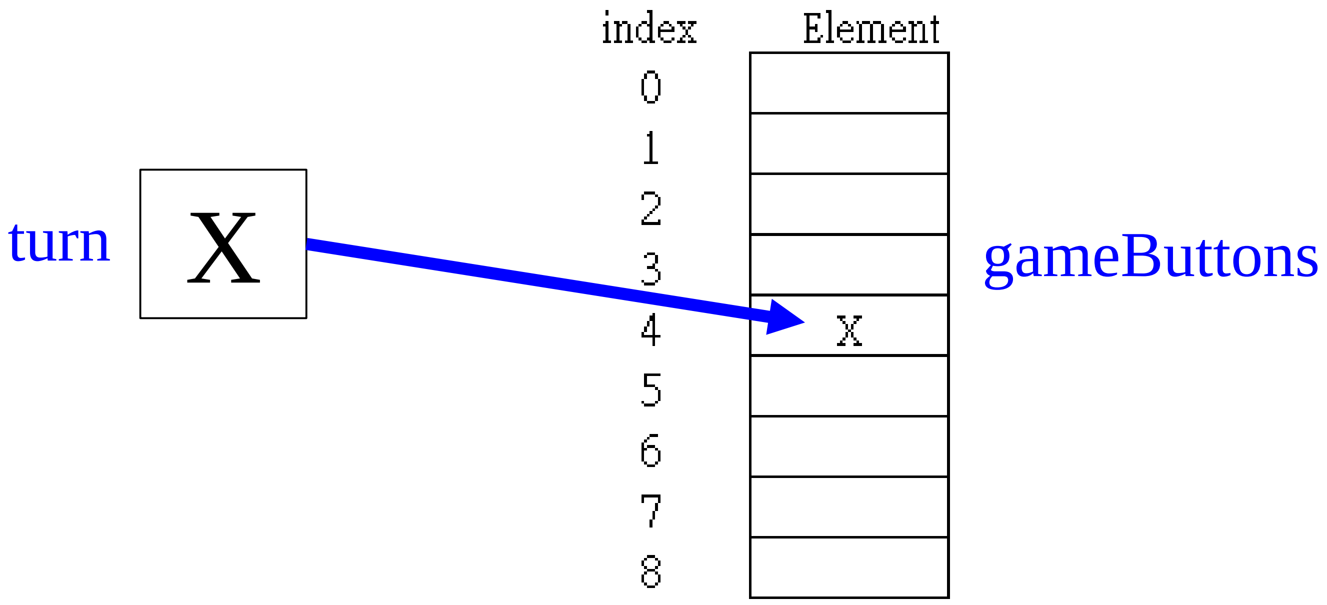
List ‘gameButtons’



- The initial value of all the elements of the list ‘gameButtons’ are nil string, i.e. ''
- After the player makes his/her choice on the game board, the corresponding element in the list will be assigned the content of either ‘X’ or ‘O’
- The content of ‘X’ or ‘O’ is come from the content of the variable ‘turn’, i.e. the current player

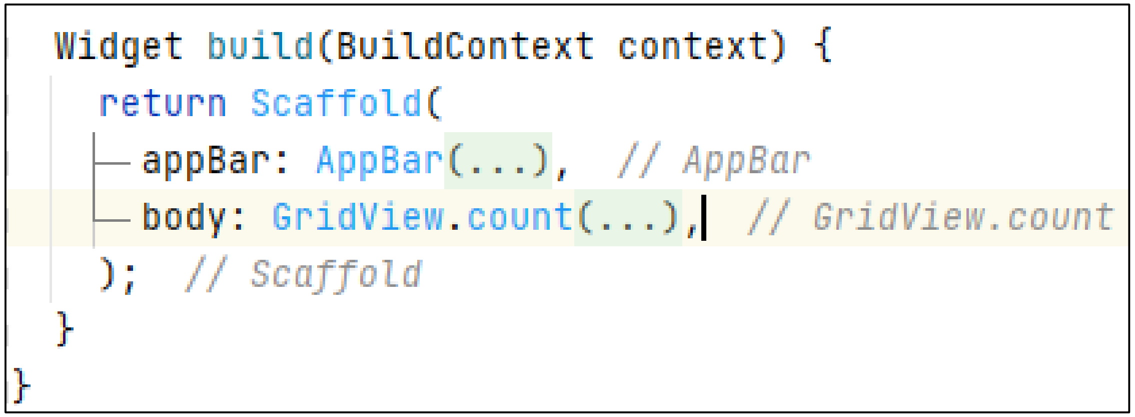
Example

If player X choose the fifth cell (2nd cell on the 2nd row) on the game board, the content of the variable 'turn' will be assigned to the fifth element of the list 'gameButtons' as:



Column Widget (I)

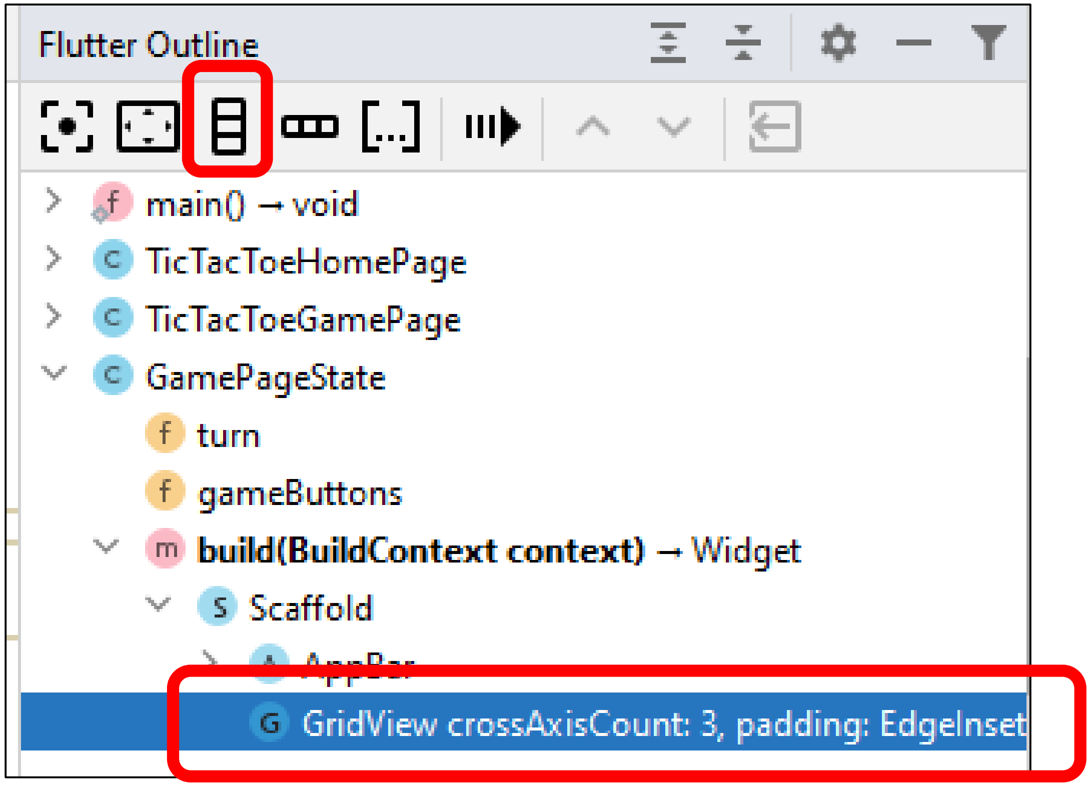
- At the bottom of the game board, a Text widget will be used to show the current player
- As there is already a GridView widget in the body of the Scaffold, a Column widget must be used to accommodate both the GridView widget and the newly created Text widget



```
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(...),  // AppBar
      body: GridView.count(...),  // GridView.count
    );  // Scaffold
  }
}
```

Column Widget (II)

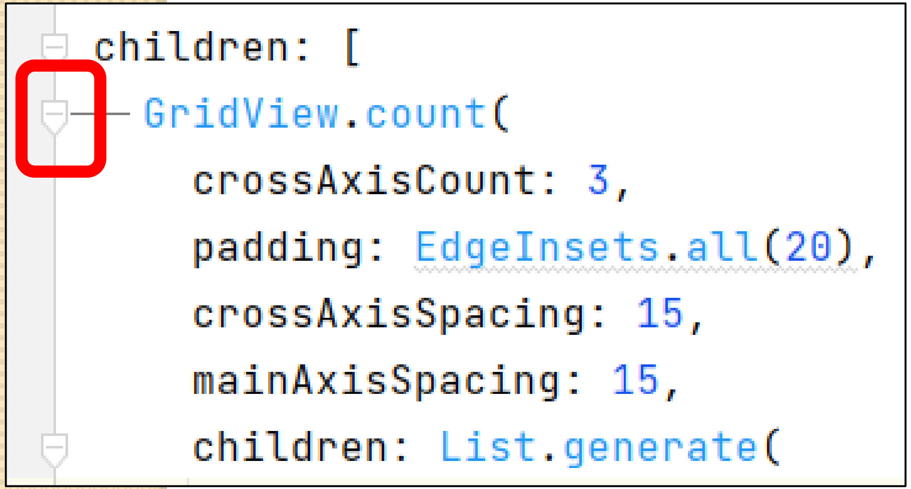
In the Flutter Outline, select the GridView.count widget, click the 'Wrap with Column' icon on the toolbar


Flutter Outline

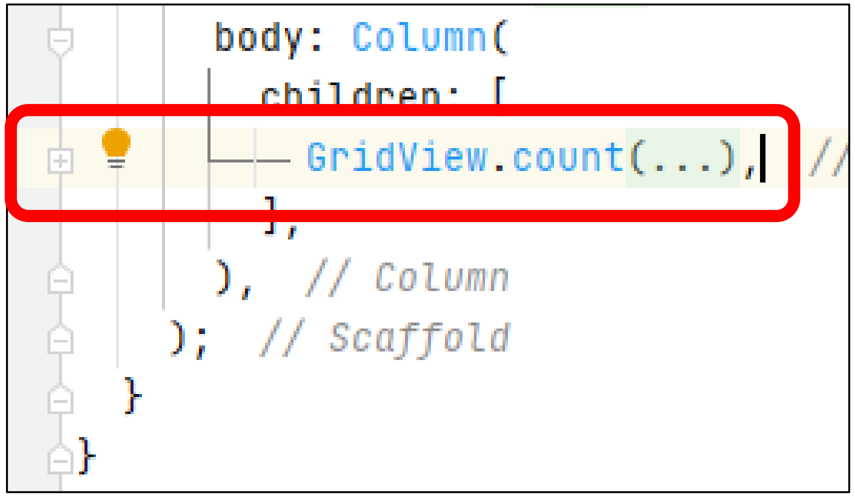
main() → void
TicTacToeHomePage
TicTacToeGamePage
GamePageState
turn
gameButtons
build(BuildContext context) → Widget
Scaffold
AppBar
GridView crossAxisCount: 3, padding: EdgeInset

Text Widget (I)

Click the – sign on the left of the GridView.count widget to collapse all the code of this widget



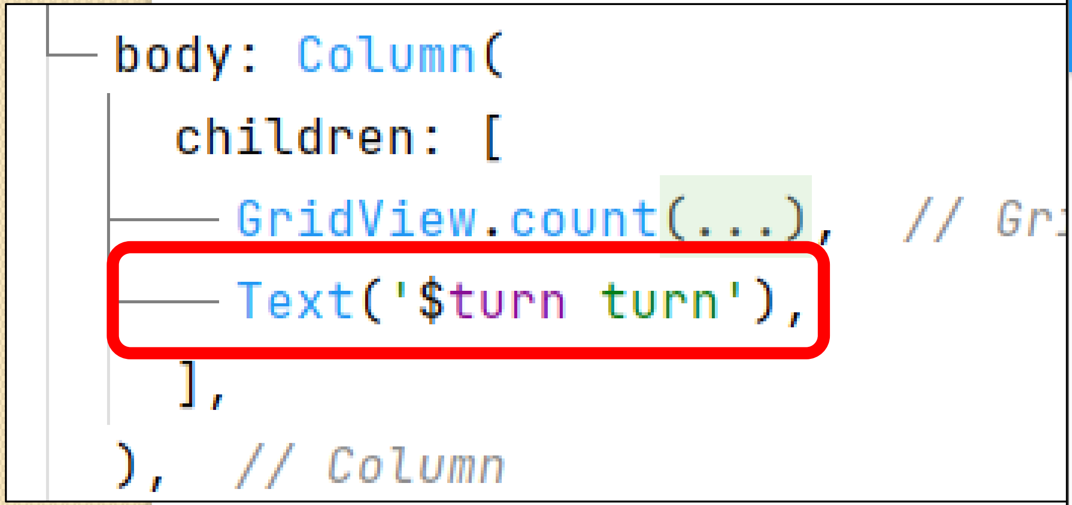
```
children: [
  GridView.count(
    crossAxisCount: 3,
    padding: EdgeInsets.all(20),
    crossAxisSpacing: 15,
    mainAxisSpacing: 15,
    children: List.generate(
```



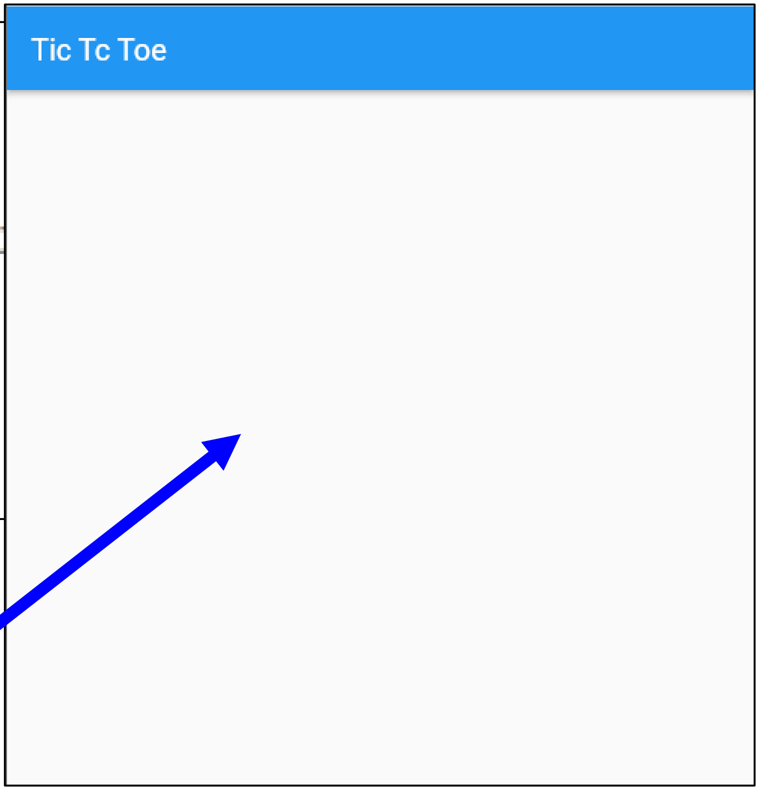
```
        body: Column(
          children: [
            GridView.count(...), //
          ],
        ), // Column
      ); // Scaffold
    }
}
```

Text Widget (II)

Insert the Text widget after the GridView. count widget as shown



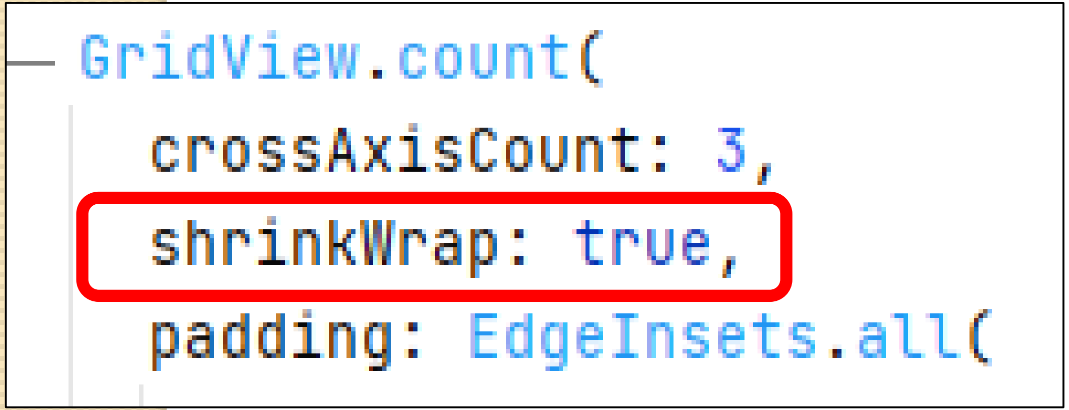
```
body: Column(
  children: [
    GridView.count(...),   // Gr
    Text('$turn turn'),
  ],
),  // Column
```



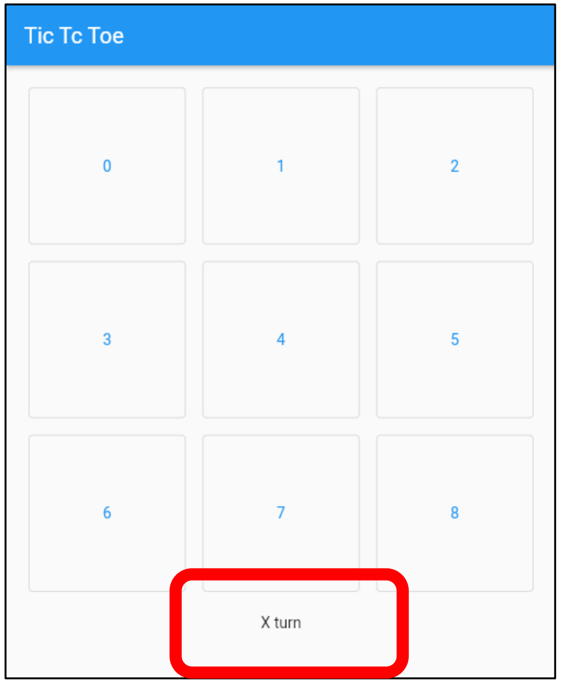
Nothing show up

shrinkWrap Property

- When a Column widget is used, the GridView.count widget will use as much as possible the vertical space of the column
- To prevent this, the shrinkWrap property will be used as shown:

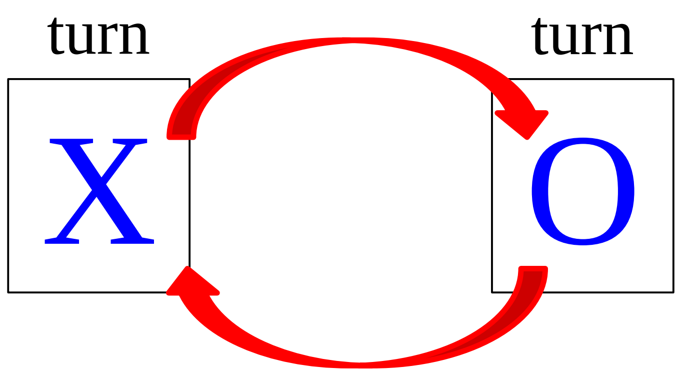


```
GridView.count(
  crossAxisCount: 3,
  shrinkWrap: true,
  padding: EdgeInsets.all(
```



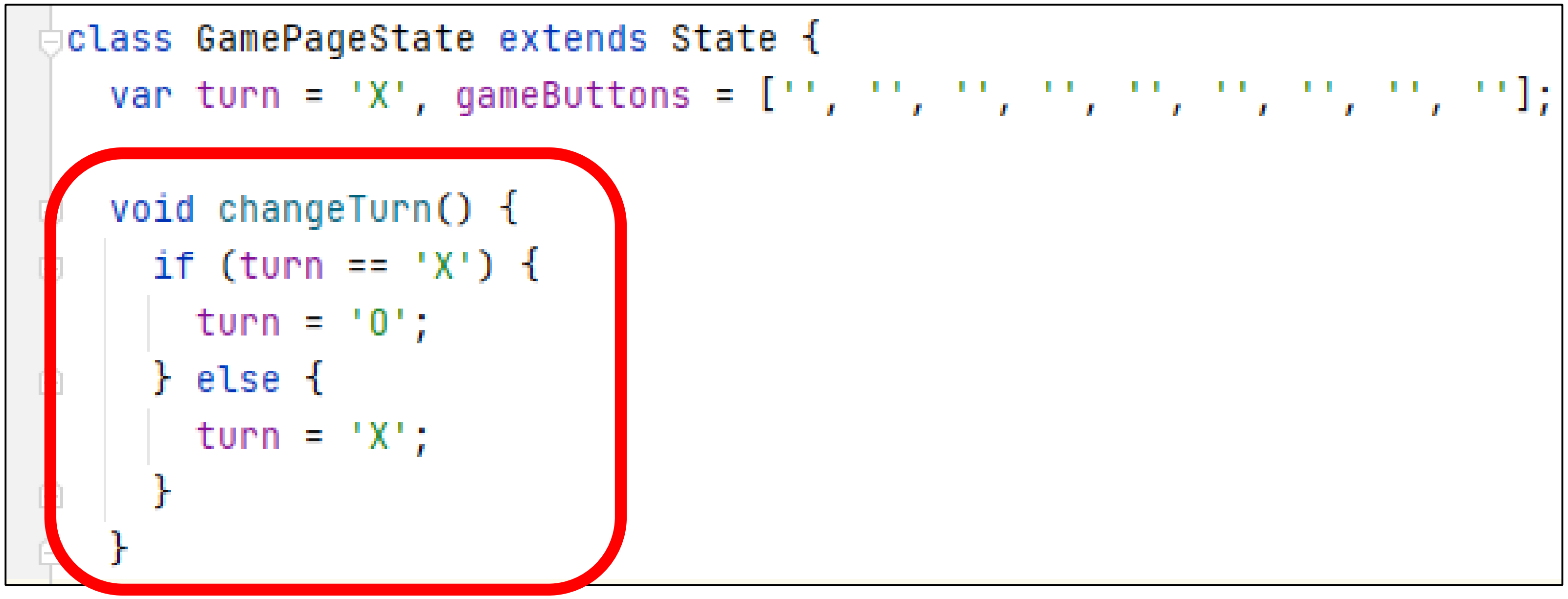
Changing Turn (I)

- A new function 'changeTurn' will be created to change the content of the variable 'turn'
- If the content of 'turn' is 'X', then it will be changed to 'O'
- On the other hand, if the content of 'turn' is 'O', then it will be changed to 'X'



Changing Turn (II)

The function ‘changeTurn’ will be placed inside the class ‘GamePageState’ as

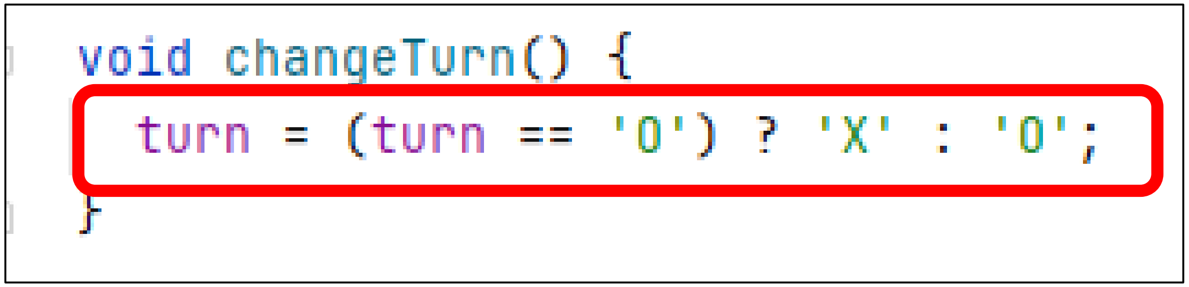


```
class GamePageState extends State {
  var turn = 'X', gameButtons = ['', '', '', '', '', '', '', '', ''];

  void changeTurn() {
    if (turn == 'X') {
      turn = 'O';
    } else {
      turn = 'X';
    }
  }
```

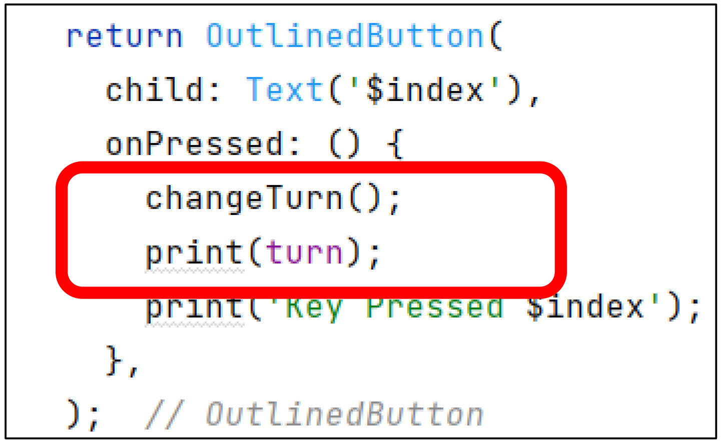
Changing Turn (III)

- Using a short form, the if statement can be re-write as



```
void changeTurn() {
    turn = (turn == 'O') ? 'X' : 'O';
}
```

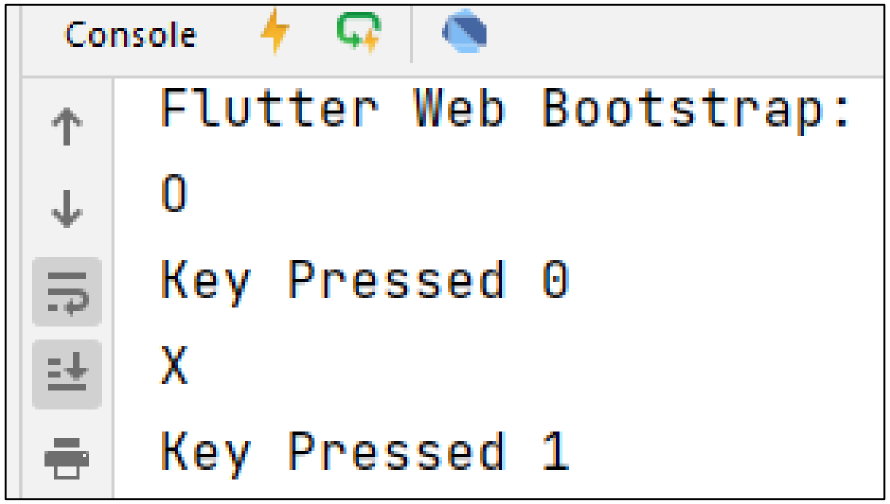
- To execute this function, in the onPressed function, add the following statements



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

Changing Turn (IV)

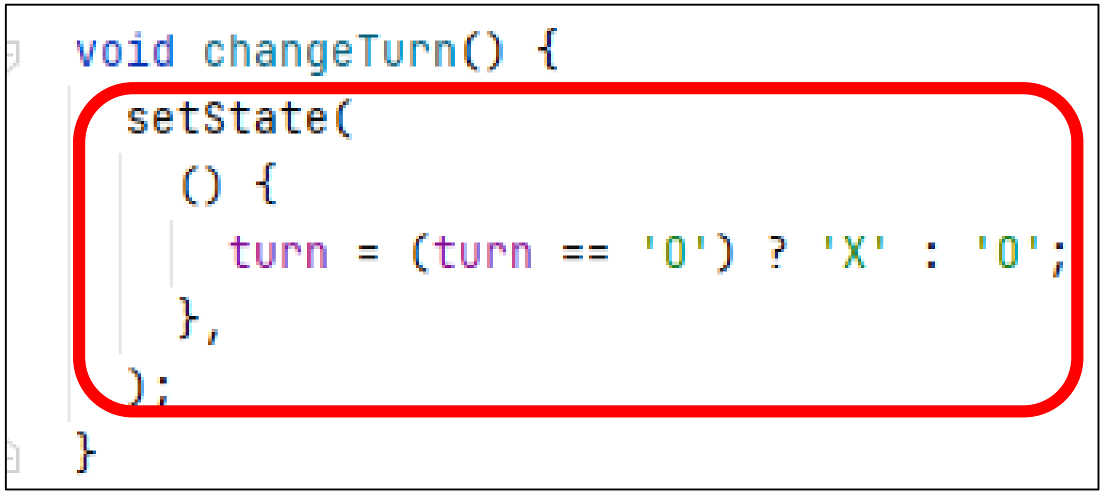
- The first new statement call to execute the function ‘changeTurn’
- The print statement will display the content of variable ‘turn’, which representing the current player, as a checkpoint



```
Flutter Web Bootstrap:
0
Key Pressed 0
X
Key Pressed 1
```

setState Method (I)

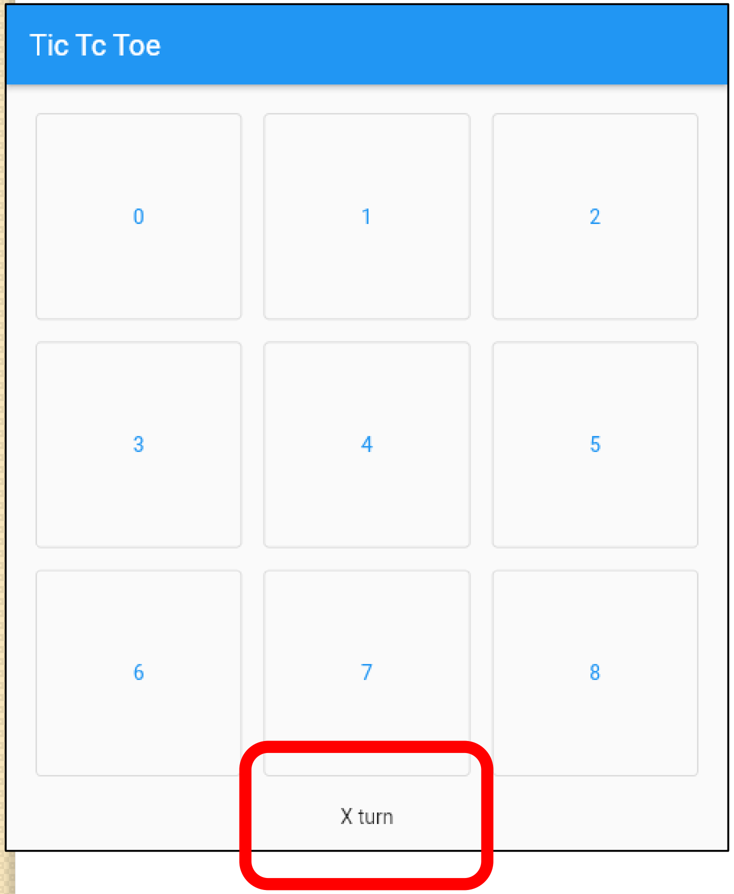
- setState is one of the ways to update the screen’s UI if there is a change of the state of object, e.g. the content of the variable ‘turn’ to be shown in the Text widget
- Re-write the function ‘changeTurn’ as:

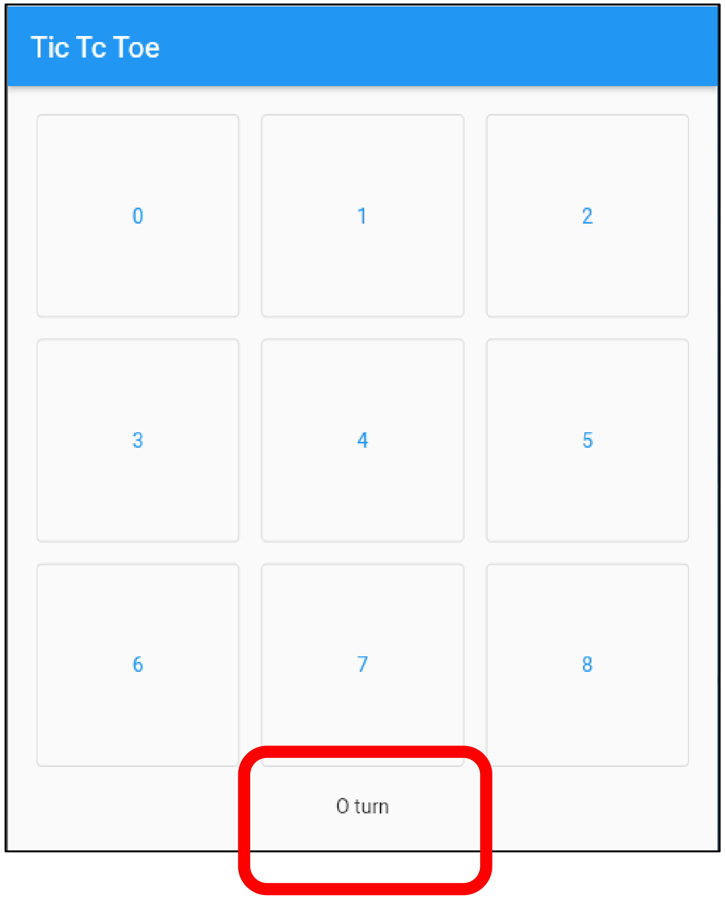


```
void changeTurn() {
  setState(
    () {
      turn = (turn == 'O') ? 'X' : 'O';
    },
  );
}
```

setState Method (II)

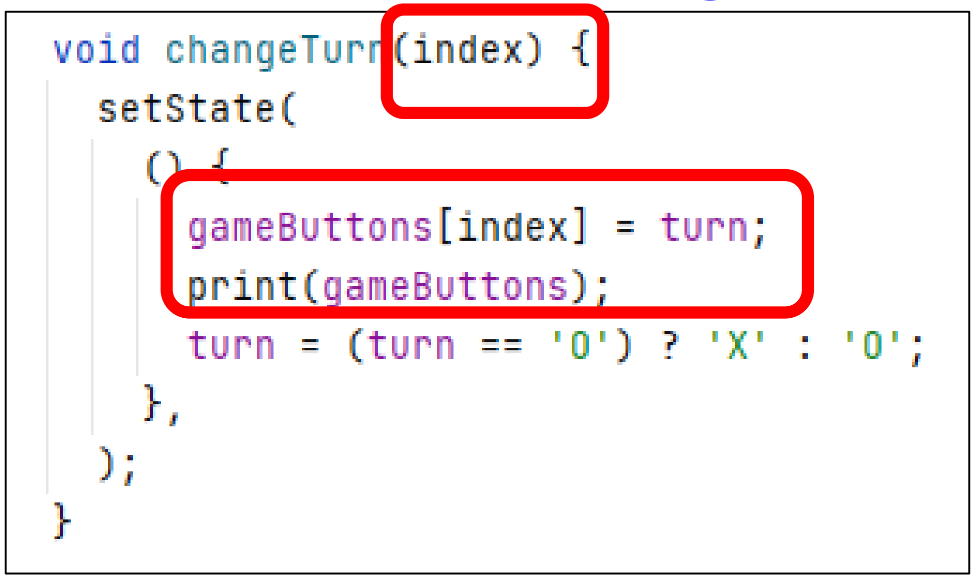
Run the App, the content of the Text widget changes every time a button is pressed





Record Cells Chosen (I)

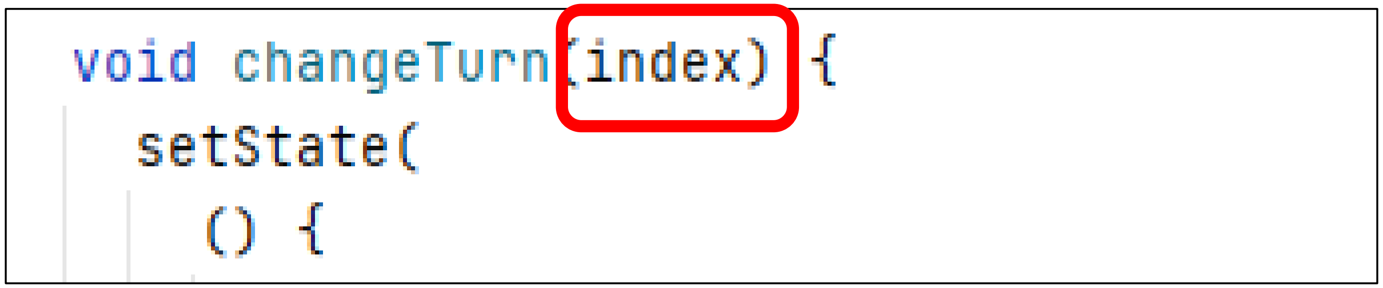
- The content of the variable 'turn' will be assigned to the corresponding element of the list 'gameButtons' in order to record which cell is chosen and by which player
- Re-write the function 'changeTurn' as:



```
void changeTurn(index) {
  setState(
    () {
      gameButtons[index] = turn;
      print(gameButtons);
      turn = (turn == 'O') ? 'X' : 'O';
    },
  );
}
```

Record Cells Chosen (II)

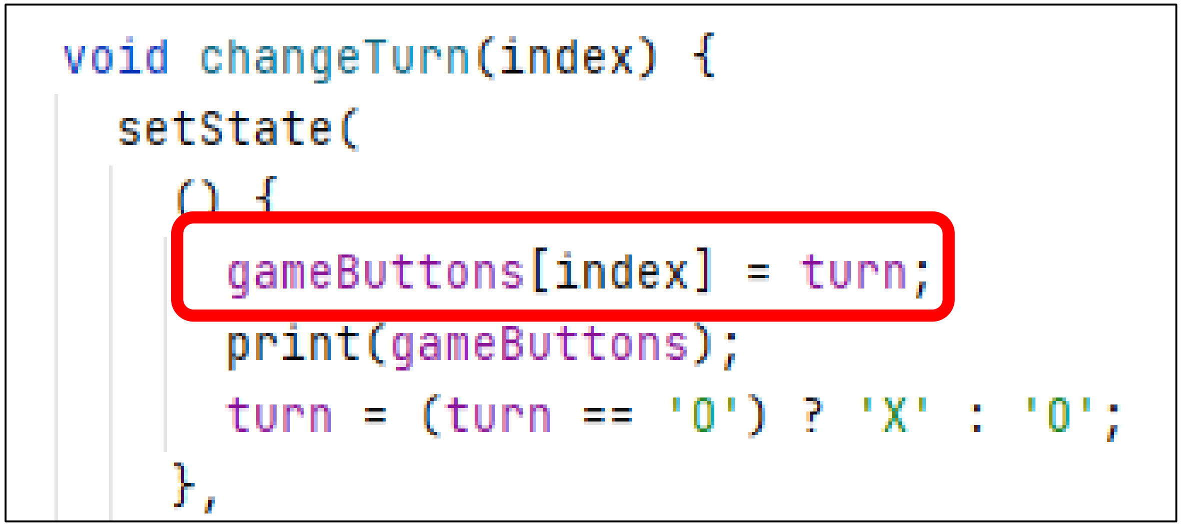
- The function 'changeTurn' will receive an argument from the calling function and store it in the variable 'index'
- The value of 'index' will be used to identify which cell on the game board is being clicked



```
void changeTurn(index) {
  setState(
    () {
```

Record Cells Chosen (III)

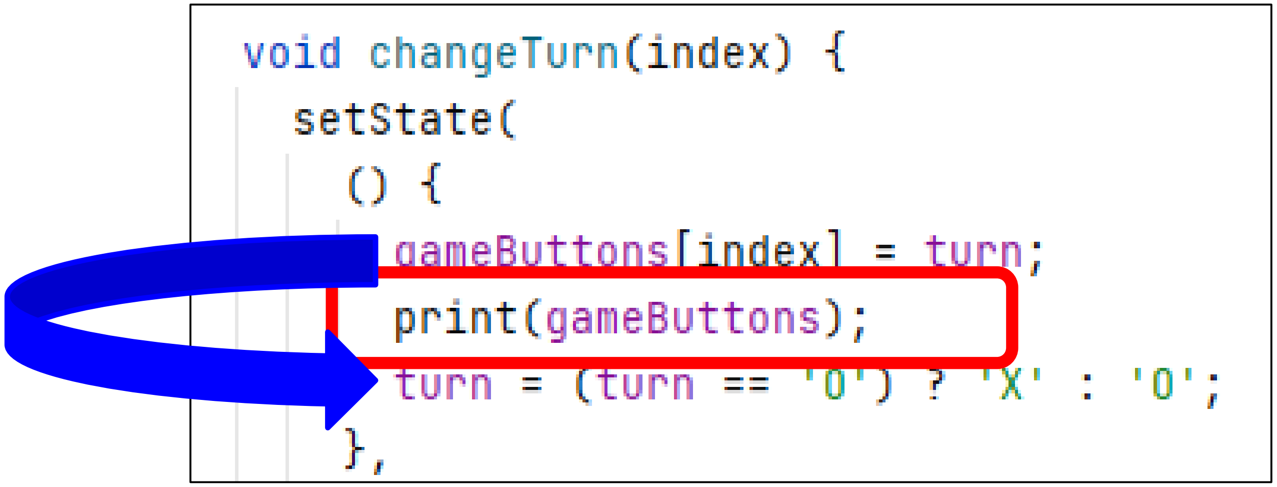
The content of the variable 'turn', i.e. the current player, will be assigned to the corresponding element in the list 'gameButtons' by using the value from the variable 'index'



```
void changeTurn(index) {
  setState(
    () {
      gameButtons[index] = turn;
      print(gameButtons);
      turn = (turn == 'O') ? 'X' : 'O';
    },
```

Record Cells Chosen (IV)

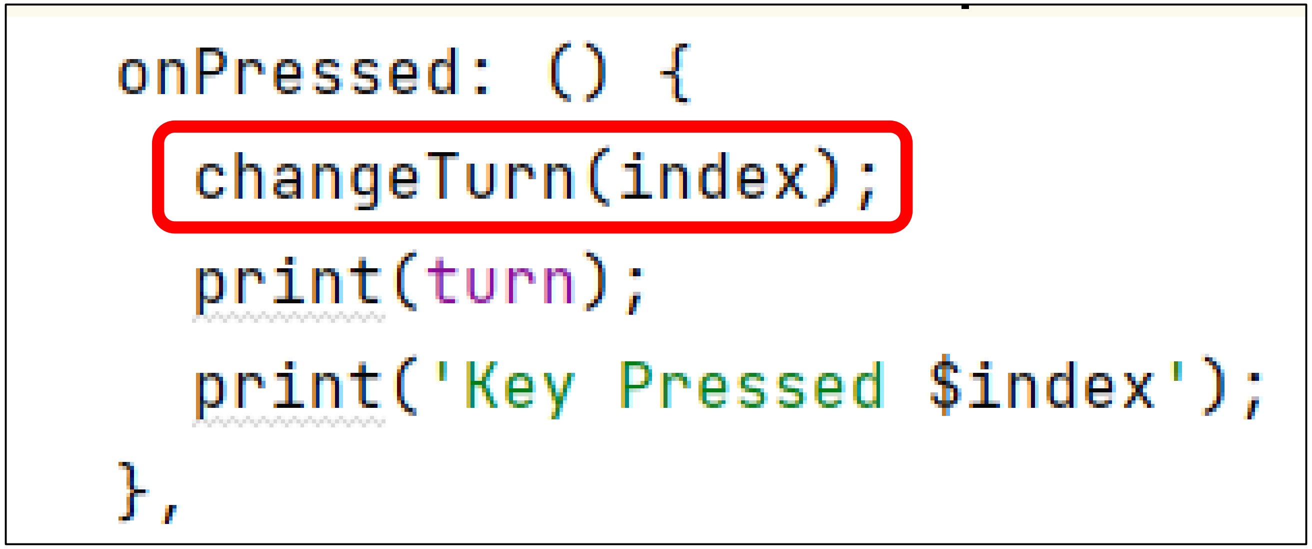
- The print statement shows all the elements of the list 'gameButtons' in the console as a check point
- Pay attention that the assignment statement must be placed before the short form if statement. Why?



```
void changeTurn(index) {
  setState(
    () {
      gameButtons[index] = turn;
      print(gameButtons);
      turn = (turn == 'O') ? 'X' : 'O';
    },
```

Record Cells Chosen (V)

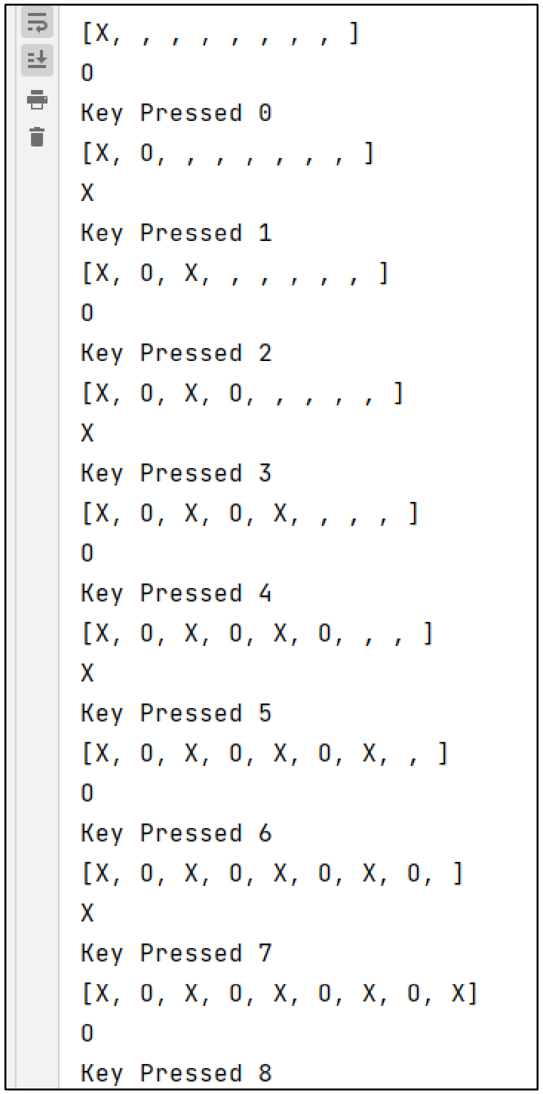
Finally, in the onPressed function, call to execute the function ‘changeTurn’ by providing a parameter ‘index’, which representing the cell being clicked



```
onPressed: () {
  changeTurn(index);
  print(turn);
  print('Key Pressed $index');
},
```

Record Cells Chosen (VI)

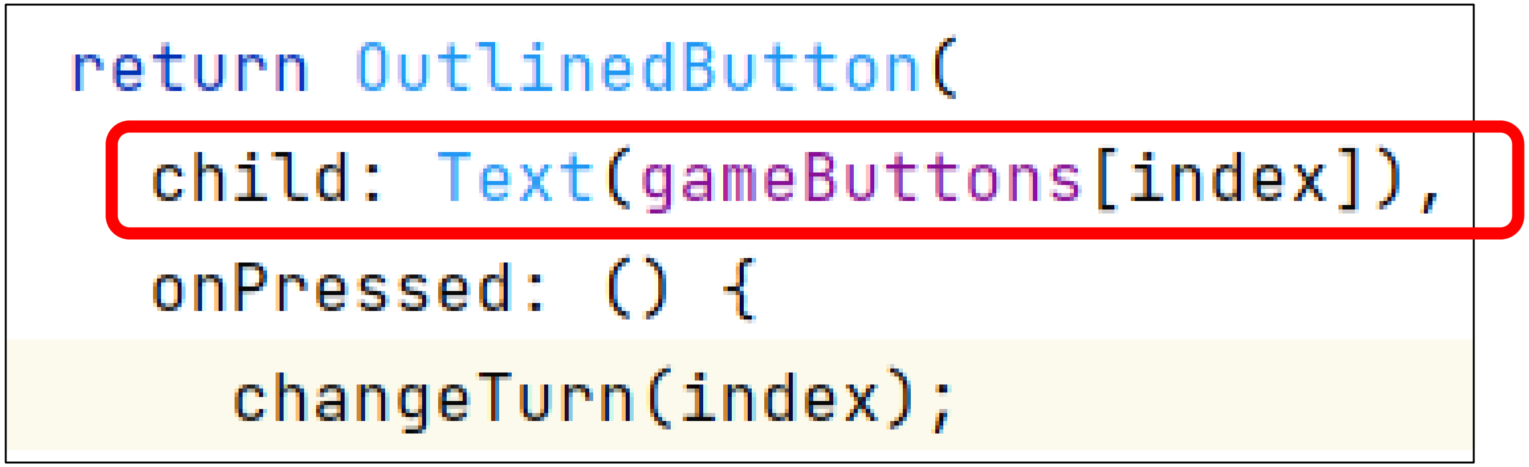
Run the app, click the 9 buttons in sequence, the console should display:



```
[X, , , , , , , , ]
O
Key Pressed 0
[X, O, , , , , , , ]
X
Key Pressed 1
[X, O, X, , , , , , ]
O
Key Pressed 2
[X, O, X, O, , , , , ]
X
Key Pressed 3
[X, O, X, O, X, , , , ]
O
Key Pressed 4
[X, O, X, O, X, O, , , ]
X
Key Pressed 5
[X, O, X, O, X, O, X, , ]
O
Key Pressed 6
[X, O, X, O, X, O, X, O, ]
X
Key Pressed 7
[X, O, X, O, X, O, X, O, X]
O
Key Pressed 8
```

Show on Buttons (I)

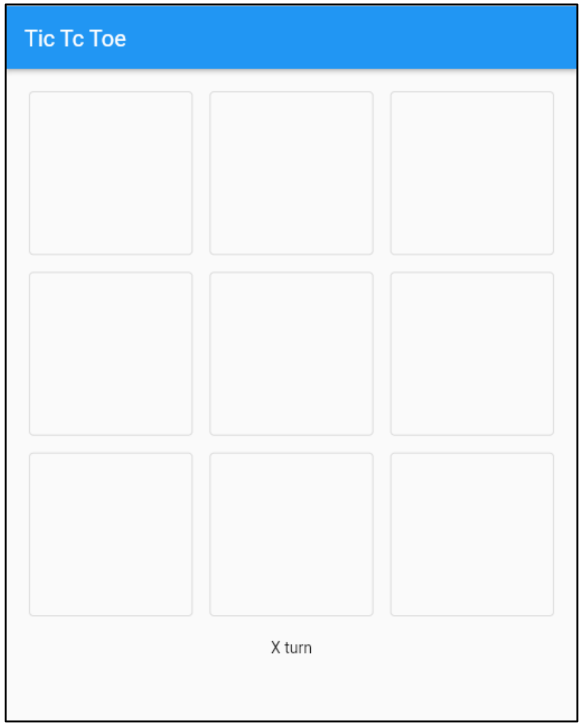
In the OutlinedButton, change the content to be displayed as shown:



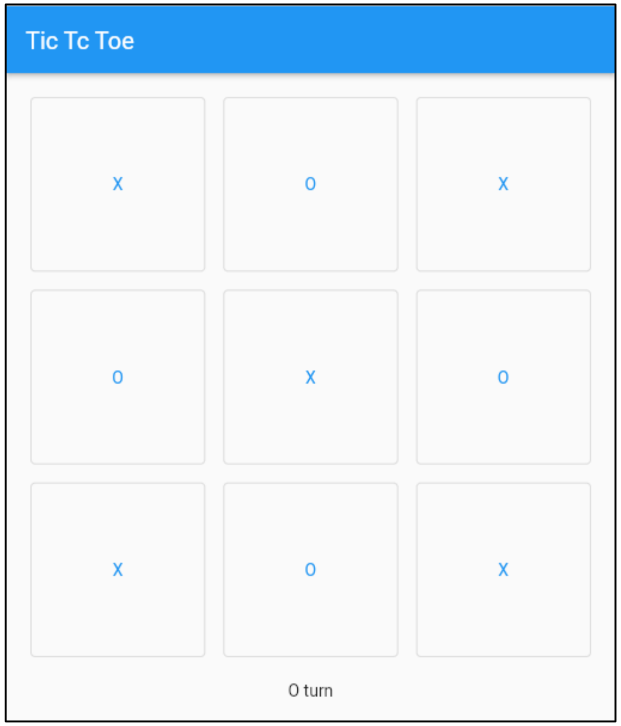
```
return OutlinedButton(
  child: Text(gameButtons[index]),
  onPressed: () {
    changeTurn(index);
```

Show on Buttons (II)

Run the app, all the buttons are blank initially because all the elements of the list 'gameButtons' are nil string at start

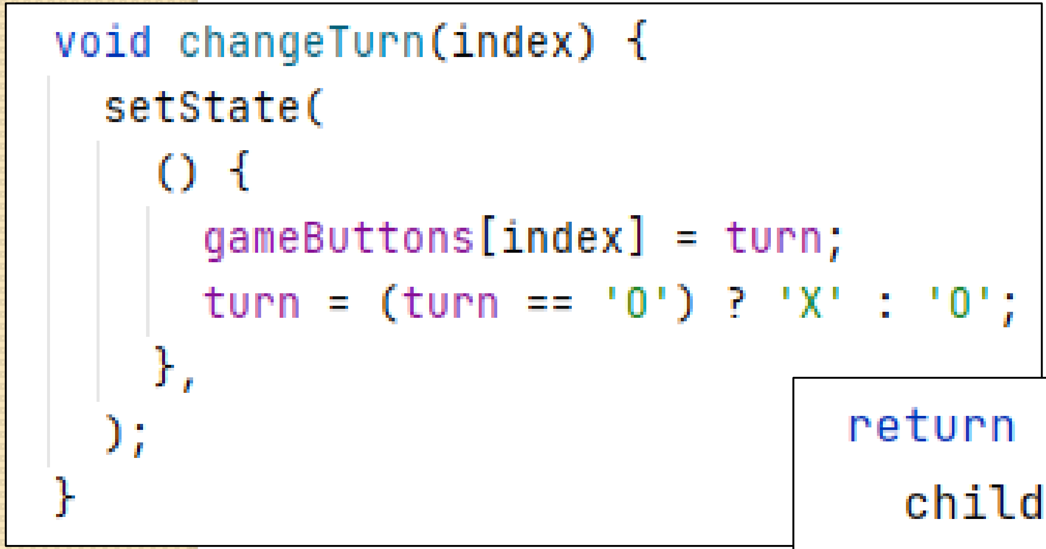


After all the buttons are clicked

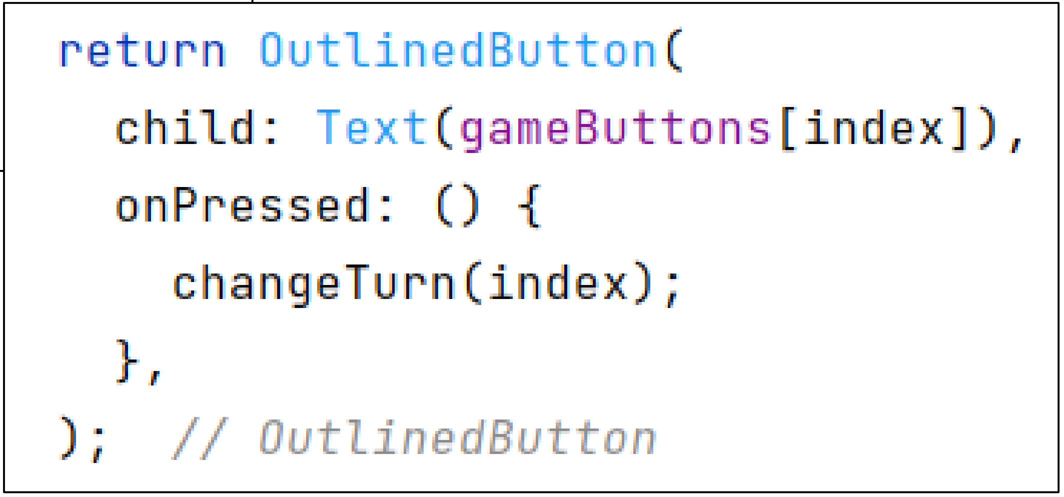


Removing the Check Points

Delete all the print statements as shown:



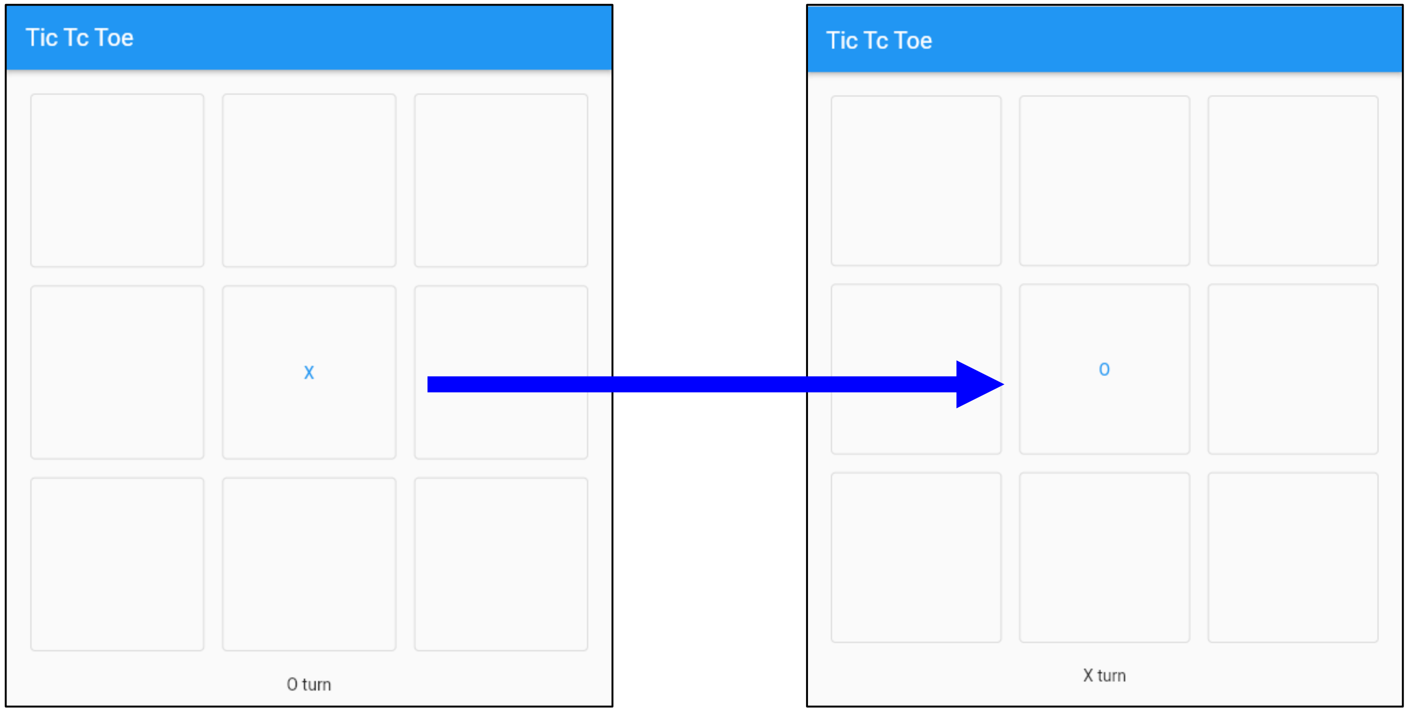
```
void changeTurn(index) {
  setState(
    () {
      gameButtons[index] = turn;
      turn = (turn == 'O') ? 'X' : 'O';
    },
  );
}
```



```
return OutlinedButton(
  child: Text(gameButtons[index]),
  onPressed: () {
    changeTurn(index);
  },
);  // OutlinedButton
```

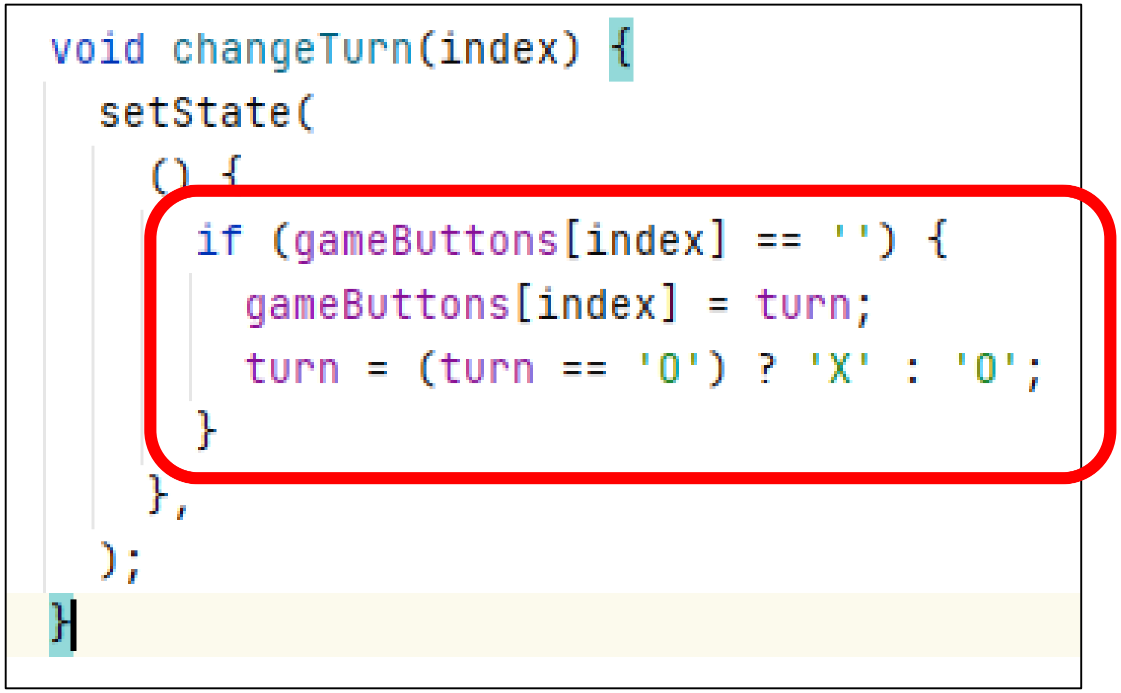
Clicking the Same Button (I)

- If the same button is clicked more than once, its content changes
- For example, the middle cell is clicked twice



Clicking the Same Button (II)

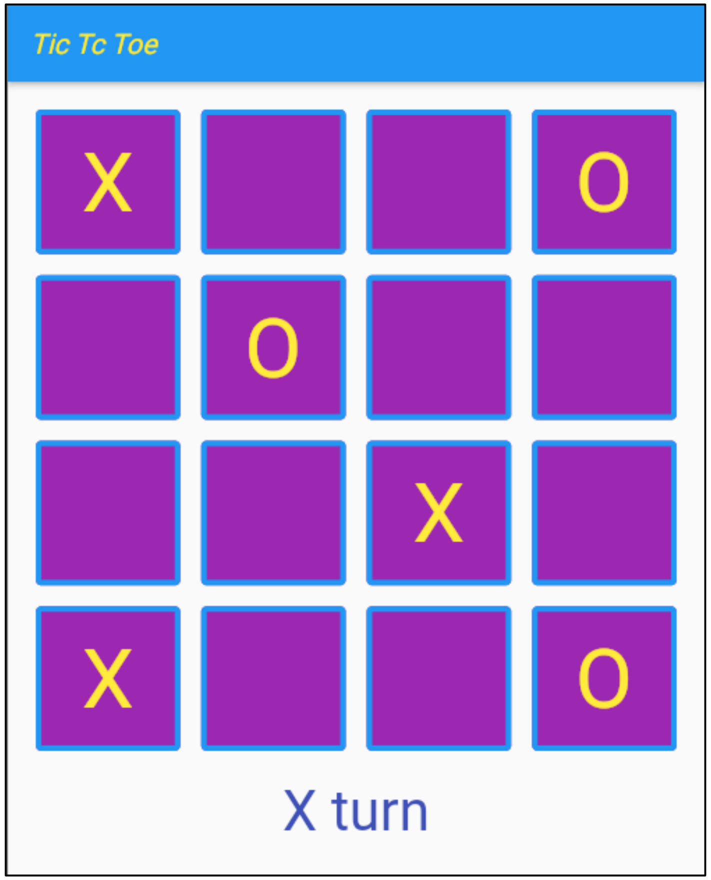
To prevent the error in the previous slide, an if statement will be used in the function ‘changeTurn’ as shown:



```
void changeTurn(index) {
  setState(
    () {
      if (gameButtons[index] == '') {
        gameButtons[index] = turn;
        turn = (turn == 'O') ? 'X' : 'O';
      }
    },
  );
}
```

Sample Screen (Project)

Apply at least two text styles to the Text Widget at the bottom



END