Custom Widget in Flutter
Math library
Random class

Custom Widget in Flutter

- A Custom Widget is a customizable widget that encapsulate unique and reusable UI components that aren't provided by Flutter.
- It allows to reuse the widget throughout your app.
- It provides flexibility to design your own widgets with your specific needs.

Why using Custom Widgets

- Reusable UI components.
- Extend existing Flutter widgets.
- Flexibility and customization.
- Improve code organization.
- Easier to manage and update.
- Enhance application performance.

Creating a Custom Widget

1. Define a new class by extending a base widget.
2. Define properties and constructors.
3. Override the `build()` method.

Example

```
class NumberButton extends StatelessWidget {
 final String number;
 const NumberButton({super.key, required this.number});
 @override
 Widget build(BuildContext context) {
   return Container(
     decoration: BoxDecoration(
         color: Colors.grey.shade400,
         borderRadius: BorderRadius.all(Radius.circular(50))),
     child: Center(
       child: Text(
         number,
         style: TextStyle(fontSize: 30, fontWeight: FontWeight.bold),
       ),
     ),
     height: 85,
     width: 85,
     padding: EdgeInsets.all(10),
     margin: EdgeInsets.all(10),
   );
 }
}
```

Using a Custom Widget

- You can Call the `NumberButton` custom class to create a new widget :

```
NumberButton(number: '1')
```

- You can move your Custom Widget to a separate dart file and import it to any other dart classes in the project.

Custom StatefulWidget

Stateful widgets:

- Manage their own state.
- Update state using `setState` to trigger rebuilds the widget as usual.

For `StatelessWidget`, all variables should be `final`.

Math library

- A built-in library in Flutter providing mathematical functions and constants.
- Essential for various calculations and operations.
- To import the Math class: `import 'dart:math';`

Math library

Common mathematical operations:

- sqrt, pow, log
- floor, ceil, round
- min, max,
- sin, cos, tan

Mathematical constants:

- pi, e

Random class

- The **`Random`** class in Dart is used to generate random numbers.
- It comes with the `dart:math` library. Import it!
- The `Random()` constructor creates a new random number generator.

Generating Random Integers

- To generate random integer between 0 and 99:

```
var randomInt = Random().nextInt(100);
```

- To generate a random integer with different range, you can use the `nextInt()` method and shift the range:
- Example, to generate a random number between 10 and 20:

```
var randomInt = 10 + Random().nextInt(11);
```

Generating Random Doubles

- To generate random double use the method:

```
var rand = Random().nextDouble();
```

- The Method `nextDouble()` generates a random floating-point number between 0.0 and 1.0.

Generating Random Booleans

- To generate random boolean use the method:

```
var rand = Random().nextBool();
```

- The Method `nextBool()` generates True/False random boolean values.

Practice

- Using Custom Widget, make the following layout.
- When a number is pressed, it calculates its square root. Use (`print`).
- When * is pressed, it generate a random integer. Use (`print`).