MAD LAB EXPERIMENT NO. : 02

NAME : DEEPA BEHRANI

CLASS : D15B

ROLL_NO: 05

Exploring Flutter Widgets

Flutter provides a rich set of widgets that can be categorized into two types:

1. Visible Widgets (Output and Input Widgets)

These widgets are responsible for displaying content and handling user interactions.

a. Text Widget

- The **Text** widget is used to display text on the screen.
- It includes properties such as textAlign for alignment and style for customization.
- Example:
- Text(
- 'Hello, Flutter!',
- textAlign: TextAlign.center,
- style: TextStyle(fontWeight: FontWeight.bold),
-)

b. Button Widgets

- Flutter does not provide a generic Button widget; instead, it has different button types like:
    - **ElevatedButton** (Previously RaisedButton)
    - **TextButton** (Previously FlatButton)
    - **OutlinedButton**
- Example of an **ElevatedButton**:
- ElevatedButton(
- onPressed: () {
- print("Button Clicked!");
- },
- child: Text("Click Me"),
-)

c. Image Widget

- The Image widget allows displaying images from various sources:
    - **Asset images**: Stored in the assets/ folder.
    - **Network images**: Loaded from a URL.
    - **File images**: Loaded from the local storage.
    - **Memory images**: Loaded from memory.
- Example of an **asset image**:
- Image.asset('assets/sample.jpg')

---

2. Invisible Widgets (Layout and Control Widgets)

These widgets help in structuring the UI.

a. Container Widget

- A versatile widget used for styling, padding, and layout customization.
- Example:
- Container(
- padding: EdgeInsets.all(10),
- decoration: BoxDecoration(color: Colors.blue),
- child: Text("Flutter Container"),
-)

b. Row and Column Widgets

- Used for **horizontal (Row)** and **vertical (Column)** alignment of widgets.
- Example:
- Row(
- mainAxisAlignment: MainAxisAlignment.center,
- children: [
- Text("Hello"),
- Text("Flutter"),
-],
-)

c. Scaffold Widget

- Provides the basic layout structure, including an **AppBar, Body, and FloatingActionButton**.
- Example:

- Scaffold(
- appBar: AppBar(title: Text("Flutter App")),
- body: Center(child: Text("Welcome to Flutter!")),
-)

CODE :

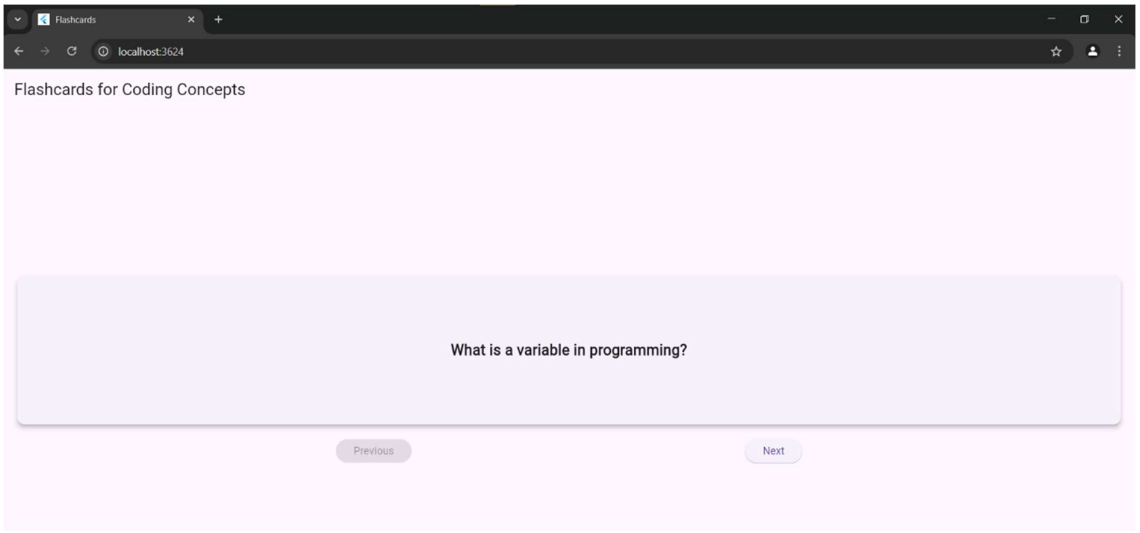
```
import 'package:flutter/material.dart';

void main() {
  runApp(FlashcardApp());
}

class FlashcardApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Flashcards',
      home: FlashcardScreen(),
    );
  }
}

class FlashcardScreen extends StatefulWidget {
  @override
  _FlashcardScreenState createState() => _FlashcardScreenState();
}

class _FlashcardScreenState extends State<FlashcardScreen> {
  int currentIndex = 0;
  bool showAnswer = false;
```

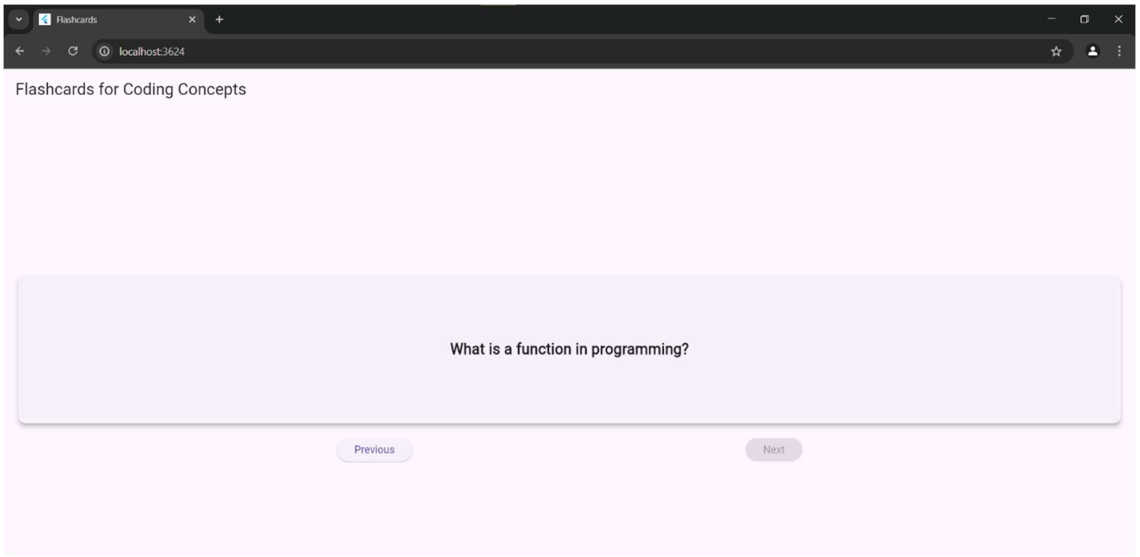
```
final List<Map<String, String>> flashcards = [
  {
    'question': 'What is a variable in programming?',
    'answer': 'A storage location with a name that holds data.'
  },
  {
    'question': 'What does the "const" keyword do in Dart?',
    'answer': 'Declares a compile-time constant.'
  },
  {
    'question': 'What is a function in programming?',
    'answer': 'A reusable block of code that performs a task.'
  },
];

void nextFlashcard() {
  if (currentIndex < flashcards.length - 1) {
    setState(() {
      currentIndex++;
      showAnswer = false;
    });
  }
}

void previousFlashcard() {
  if (currentIndex > 0) {
    setState(() {
      currentIndex--;
      showAnswer = false;
    });
  }
```

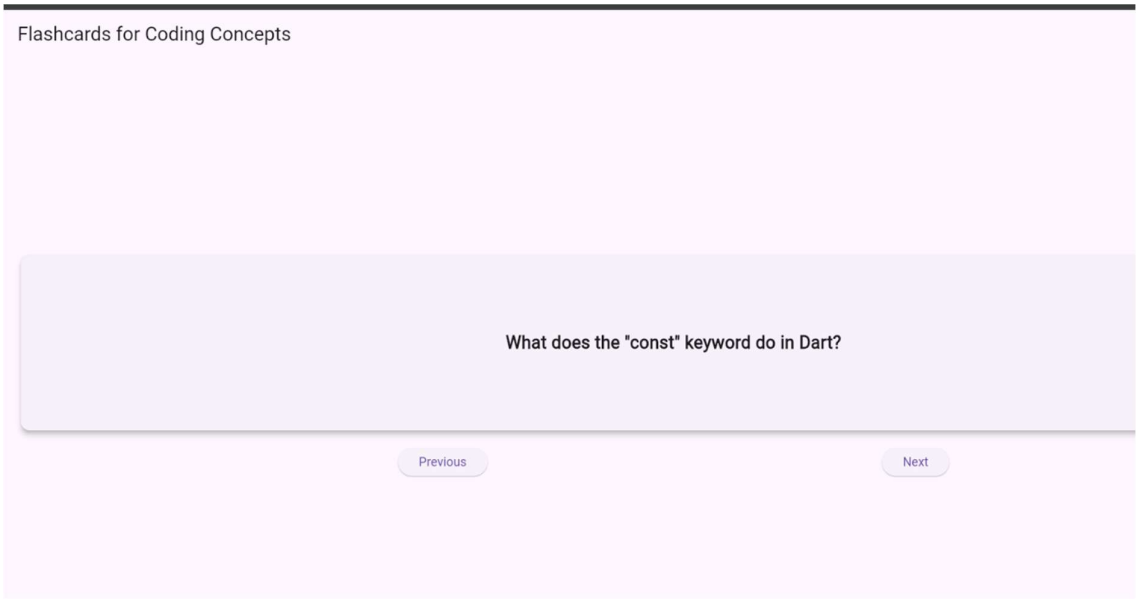
```
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(title: Text('Flashcards for Coding Concepts')),
    body: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        GestureDetector(
          onTap: () {
            setState(() {
              showAnswer = !showAnswer;
            });
          },
          child: Card(
            elevation: 5,
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(10),
            ),
            margin: EdgeInsets.all(20),
            child: Container(
              padding: EdgeInsets.all(20),
              height: 200,
              alignment: Alignment.center,
              child: Text(
                showAnswer
                    ? flashcards[currentIndex]['answer']!
                    : flashcards[currentIndex]['question']!,
                textAlign: TextAlign.center,
                style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold), ), ),),),
```

```
Row(
 mainAxisAlignment: MainAxisAlignment.spaceEvenly,
 children: [
  ElevatedButton(
   onPressed: currentIndex > 0 ? previousFlashcard : null,
   child: Text('Previous'),
  ),
  ElevatedButton(
   onPressed:
   currentIndex < flashcards.length - 1 ? nextFlashcard : null,
   child: Text('Next'),
 ),],),],),); }}
```

OUTPUT:







Conclusion

Flutter provides a robust and flexible widget-based UI framework for developing cross-platform applications. It classifies widgets into **visible (output and input)** and **invisible (layout and control)** categories, enabling a structured and reusable approach to UI design.

- **Visible widgets**, such as Text, Button, and Image, facilitate user interaction and content display.
- **Invisible widgets**, including Container, Row, Column, and Scaffold, contribute to efficient UI structuring and layout management.

With its **real-time code reflection capabilities**, an extensive widget library, and optimized performance, Flutter proves to be a highly effective framework for modern application development.

A comprehensive understanding of these fundamental widgets serves as a strong foundation for building dynamic and interactive applications, such as a **flashcard app for coding concepts**.