LAB-10

SMART DEVICE PROGRAMMING

Name: Shrey K. Naik

Roll No.: CE073

Batch: A4

ID: 20CEUOG036

GIT REPO:

https://github.com/naikshrey2308/CE073_SDP_Labs

TUTORIAL-1

Stateful Widget:

When a Flutter builds a Stateful Widget, it creates a State object. This object is where all the mutable state for that widget is held.

The concept of state is defined by two things:

1. The data used by the widget might change.
2. The data can't be read synchronously when the widget is built. (All state must be established by the time the build method is called).

The lifecycle of stateful widget has the following simplified steps:

- createState()
- mounted == true
- initState()
- didChangeDependencies()
- build()
- didUpdateWidget()
- setState()
- deactivate()
- dispose()
- mounted == false

Code Test 1:

home.dart

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

class Home extends StatefulWidget {
  const Home({Key? key}) : super(key: key);

  @override
  State<Home> createState() => _HomeState();
}

class _HomeState extends State<Home> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SafeArea(
        child: Container(
          padding: EdgeInsets.all(8.0),
          child: TextButton.icon(
            onPressed: () {
              Navigator.pushNamed(context, "/location");
            },
            icon: Icon(Icons.edit_location,
              color: Colors.redAccent,
            ),
            label: Text("Edit Location",
              style: TextStyle(
                fontSize: 18.0,
                color: Colors.redAccent,
```

```
              ),
            ),
          ),
        ),
      ),
    );
  }
}
```

loading.dart

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

class Loading extends StatefulWidget {
  const Loading({Key? key}) : super(key: key);

  @override
  State<Loading> createState() => _LoadingState();
}

class _LoadingState extends State<Loading> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Text("LOADING..."),
    );
  }
}
```

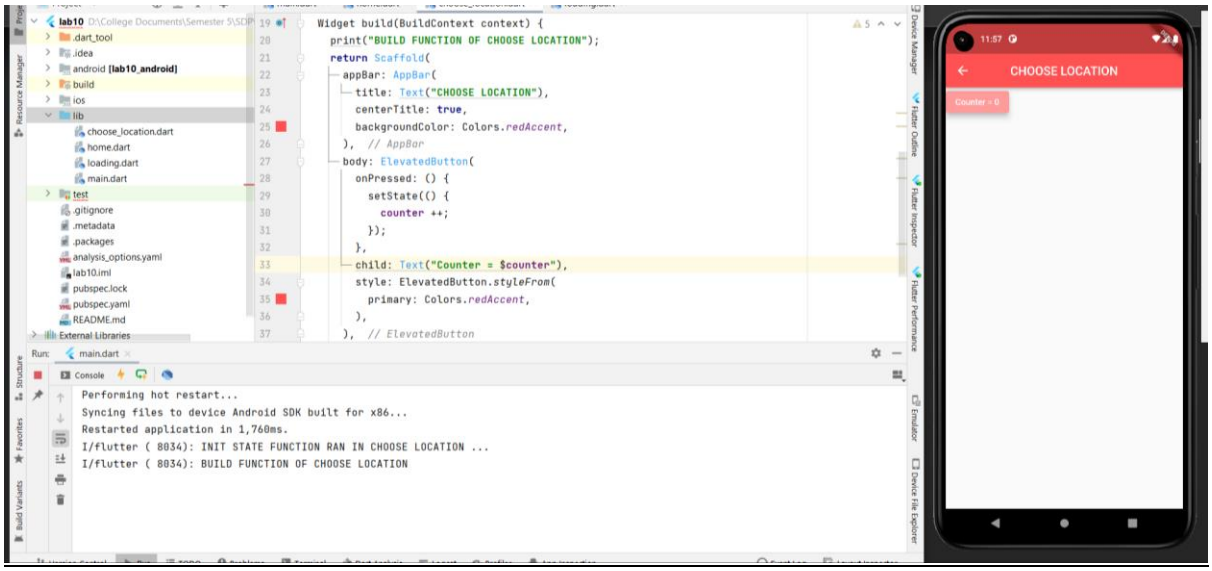
choose_location.dart

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

class ChooseLocation extends StatefulWidget {
  const ChooseLocation({Key? key}) : super(key: key);

  @override
  State<ChooseLocation> createState() => _ChooseLocationState();
}

class _ChooseLocationState extends State<ChooseLocation> {
  int counter = 0;

  void initState() {
    super.initState();
    print("INIT STATE FUNCTION RAN IN CHOOSE LOCATION ... ");
  }

  @override
  Widget build(BuildContext context) {
    print("BUILD FUNCTION OF CHOOSE LOCATION");
    return Scaffold(
      appBar: AppBar(
        title: Text("CHOOSE LOCATION"),
        centerTitle: true,
        backgroundColor: Colors.redAccent,
      ),
      body: ElevatedButton(
        onPressed: () {
```

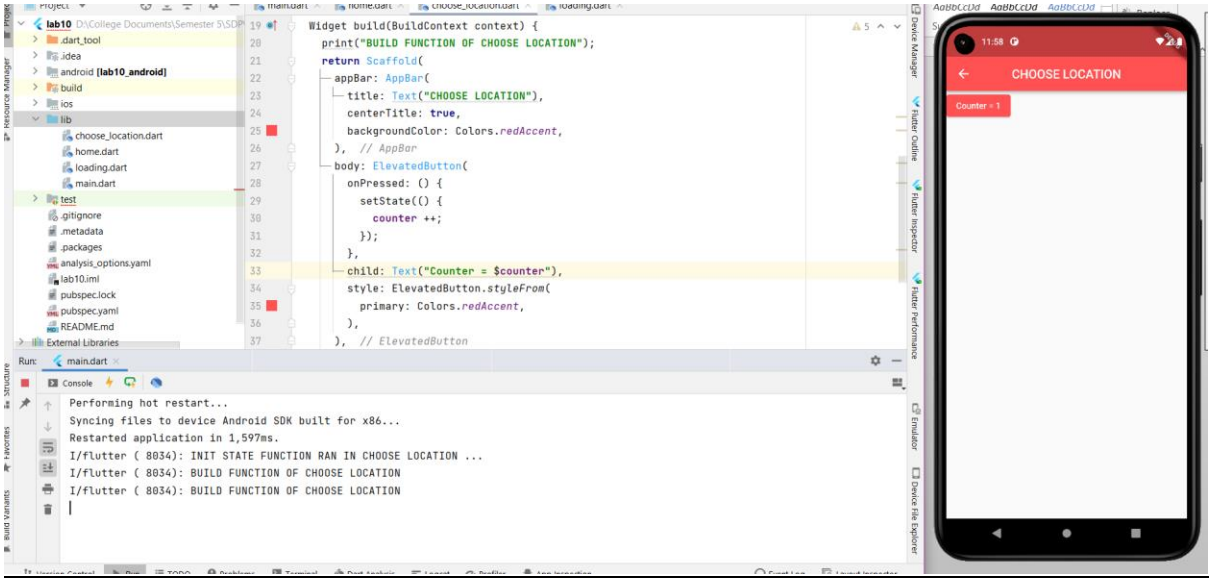
```
            setState(() {
              counter ++;
            });
          },
          child: Text("Counter = $counter"),
          style: ElevatedButton.styleFrom(
            primary: Colors.redAccent,
          ),
        ),
      );
    }
}
```

main.dart

```
import "package:flutter/material.dart";
import 'package:lab10/loading.dart';
import 'package:lab10/choose_location.dart';
import 'package:lab10/home.dart';

void main() {
  runApp(MaterialApp(
    initialRoute: "/home",
    routes: {
      "/": (context) => Loading(),
      "/home": (context) => Home(),
      "/location": (context) => ChooseLocation(),
    },
  ));/**/
}
```



After clicking the button,



Code Test 2:

Async (Asynchronous function):

- When an async function is called, a Future is immediately returned and the body of the function is executed later.
- As the body of the async function is executed, the Future returned by the function call will be completed along with its result.

Await:

- In async function we can use await keyword which will wait for the result.

Futures:

- Dart is a single-threaded programming language.
- Future<T> object represents the result of an asynchronous operation which produces a result of type T.
- If the result is not usable value, then the future's type is Future<void>.
- A Future represents a single value either a data or an error asynchronously

There are 2 ways to handle Futures:

- Using the Future API
- Using the async and await operation.

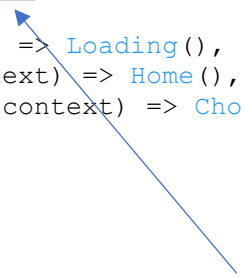
location.dart

```
import "package:flutter/material.dart";
import 'dart:convert';
import 'package:http/http.dart';

class Loading extends StatefulWidget {
  const Loading({Key? key}) : super(key: key);
```

```
  @override
  State<Loading> createState() => _LoadingState();
}

class _LoadingState extends State<Loading> {

  void getData() async {
    String url = 'https://jsonplaceholder.typicode.com/albums/1';

    final res = await get(Uri.parse(url));

    Map json = jsonDecode(res.body);

    print(json);

  }

  void initState() {
    super.initState();
    getData();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Loading Screen"),
        centerTitle: true,
        backgroundColor: Colors.redAccent,
      ),
      body: Text("LOADING..."),
    );
  }
}
```

main.dart

```
import "package:flutter/material.dart";
import 'package:lab10/loading.dart';
import 'package:lab10/choose_location.dart';
import 'package:lab10/home.dart';

void main() {
  runApp(MaterialApp(
    initialRoute: "/",
    routes: {
      "/": (context) => Loading(),
      "/home": (context) => Home(),
      "/location": (context) => ChooseLocation(),
    },
  ));/**/
}
```



TUTORIAL-2

Code Test 1:

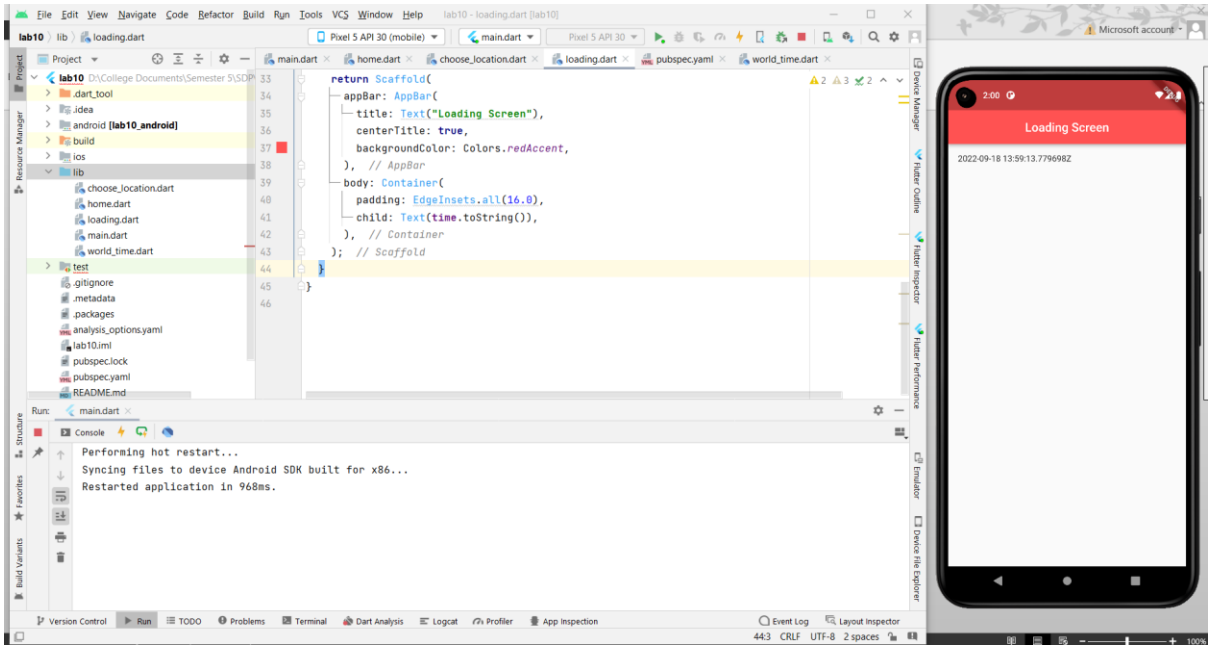
world_time.dart

```
import 'package:http/http.dart';
import 'dart:convert';

class WorldTime {

  String? location;
  String? time;
  String? flag;
  String? url;

  WorldTime({ this.location,this.flag,this.url });

  Future<void> getTime() async {
    Response response = await
get(Uri.parse('http://worldtimeapi.org/api/timezone/$url'));
    Map timeData = jsonDecode(response.body);
    String dateTime = timeData['datetime'];
    String offset = timeData['utc_offset'];
    String offsetHours = offset.substring(1,3);
    String offsetMinutes = offset.substring(4,6);
    DateTime currenttime = DateTime.parse(dateTime);
    currenttime = currenttime.add(
        Duration(minutes:
        int.parse(offsetMinutes),hours:int.parse(offsetHours)));
    time = currenttime.toString();
  }
}
```

loading.dart

```dart
import "package:flutter/material.dart";
import 'dart:convert';
import 'package:http/http.dart';
import 'package:lab10/world_time.dart';

class Loading extends StatefulWidget {
  const Loading({Key? key}) : super(key: key);

  @override
  State<Loading> createState() => _LoadingState();
}

class _LoadingState extends State<Loading> {
  String? time = 'LOADING.........';

  void setWorldTime() async {
    WorldTime timeinstance = WorldTime(location: 'kolkata', flag:
'india.png', url: 'Asia/Kolkata');
    await timeinstance.getTime();
    // print(timeinstance.time);
    setState(() {
      time = timeinstance.time;
    });
  }

  void initState() {
    super.initState();
    // getData();
    setWorldTime();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Loading Screen"),
        centerTitle: true,
        backgroundColor: Colors.redAccent,
      ),
      body: Container(
        padding: EdgeInsets.all(16.0),
        child: Text(time.toString()),
      ),
    );
  }
}
```

Here, we are updating the state based on the data received from the get request, and eventually the UI gets re-rendered with the updated values.


return Scaffold(
appBar: AppBar(
title: Text("Loading Screen"),
centerTitle: true,
backgroundColor: Colors.redAccent,
), // AppBar
body: Container(
padding: EdgeInsets.all(16.0),
child: Text(time.toString()),
), // Container
); // Scaffold
}
}
Performing hot restart...
Syncing files to device Android SDK built for x86...
Restarted application in 968ms.
Loading Screen
2022-09-18 13:59:13.779698Z