institute of CODING

Mobile Application Development

Flutter

Displaying online data in the interface

Following slides contain examples from official Flutter tutorials.

Northumbria University NEWCASTLE

Office for Students OfS

Previously,

Continuing from what we have done so far, we will display the post we fetched on the previous slides.

Before diving into the interface, let’s recap the stateful widgets since we need to use them later.

Stateful Widgets

We use stateful widgets whenever we have something changing throughout the lifetime of the widget. It can be either user interaction (user typing something) or it can be the data fetched from the internet.

Stateful Widgets

If we use stateless widget to fetch and display online data, it need to be fetched everytime when widget is drawn. To avoid downloading the data over and over again, we can put it into a stateful widget and Flutter will preserve its state no matter how many times widget is drawn.

Post.dart

```
class Post {
  int userId;
  int id;
  String title;
  String body;

  Post({this.userId, this.id, this.title, this.body});

  Post.fromJson(Map<String, dynamic> json) {
    userId = json['userId'];
    id = json['id'];
    title = json['title'];
    body = json['body'];
  }
```

Post.dart

```
  // continued

  Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = new Map<String, dynamic>();
    data['userId'] = this.userId;
    data['id'] = this.id;
    data['title'] = this.title;
    data['body'] = this.body;
    return data;
  }
}
```

main.dart

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

Future<Post> fetchPost() async {
  final URL = 'https://jsonplaceholder.typicode.com/posts/1/';
  final response = await http.get(URL);
  if (response.statusCode == 200) {
    return Post.fromJson(json.decode(response.body));
  } else {
    throw Exception('Failed to load post');
  }
}
```

main.dart

```
class HomePage extends StatefulWidget {
  final posts = fetchPost();

  @override
  _HomePageState createState() => _HomePageState();
}
```

main.dart

```
class HomePage extends StatefulWidget {
  final posts = fetchPost();
  @override
  _HomePageState createState() => _HomePageState();
}
```

Fetch the data and preserve it in the state

main.dart

```
class _HomePageState extends State<HomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(title: Text("Institute of Coding")),
        body: FutureBuilder(
            future: widget.posts,
            builder: (BuildContext context, AsyncSnapshot<Post> snapshot) {
              return CircularProgressIndicator();
            }));
  }
}
```

main.dart

```
class _HomePageState extends State<HomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(title: Text("Institute of Coding")),
        body: FutureBuilder(
            future: widget.posts,
            builder: (BuildContext context, AsyncSnapshot<Post> snapshot) {
              return CircularProgressIndicator();
            }));
  }
}
```

Use FutureBuilder widget when there is an async process (i.e. Future).

main.dart

```
class _HomePageState extends State<HomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(title: Text("Institute of Coding")),
        body: FutureBuilder(
            future: widget.posts,
            builder: (BuildContext context, AsyncSnapshot<Post> snapshot) {
              return CircularProgressIndicator();
            }));
  }
}
```

Provide the Future object

main.dart

```
class _HomePageState extends State<HomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(title: Text("Institute of Coding")),
        body: FutureBuilder(
            future: widget.posts,
            builder: (BuildContext context, AsyncSnapshot<Post> snapshot) {
              return CircularProgressIndicator();
            }));
  }
}
```

→ **Access state members using "*widget*"**

main.dart

```
class _HomePageState extends State<HomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(title: Text("Institute of Coding")),
        body: FutureBuilder(
            future: widget.posts,
            builder: (BuildContext context, AsyncSnapshot<Post> snapshot) {
              return CircularProgressIndicator();
            }));
  }
}
```

Provide the Future object

main.dart

```
class _HomePageState extends State<HomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(title: Text("Institute of Coding")),
        body: FutureBuilder(
            future: widget.posts,
            builder: (BuildContext context, AsyncSnapshot<Post> snapshot) {
              return CircularProgressIndicator();
            }));
  }
}
```

builder is to create widgets depending on the status of Future object.

main.dart

```
class _HomePageState extends State<HomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(title: Text("Institute of Coding")),
        body: FutureBuilder(
            future: widget.posts,
            builder: (BuildContext context, AsyncSnapshot<Post> snapshot) {
              return CircularProgressIndicator();
            }));
  }
}
```

CircularProgressIndicator is a spinner, specifying there is an ongoing operation in the background.

main.dart

```
builder: (BuildContext context, AsyncSnapshot<Post> snapshot) {
  if (snapshot.connectionState == ConnectionState.done) {
    return Column(
      children: <Widget>[
        Text(snapshot.data.title),
        SizedBox(height: 16),
        Text(snapshot.data.body),
      ],
    );
  } else {
    return CircularProgressIndicator();
  }
}
```

Snapshot holds the data that will become available later and also contains the current status of the given data.

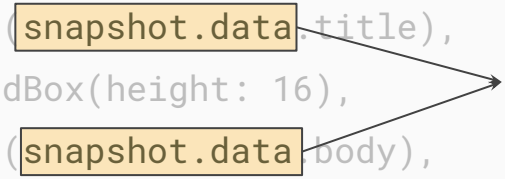
main.dart

```
builder: (BuildContext context, AsyncSnapshot<Post> snapshot) {
  if (snapshot.connectionState == ConnectionState.done) {
    return Column(
      children: <Widget>[
        Text(snapshot.data.title),
        SizedBox(height: 16),
        Text(snapshot.data.body),
      ],
    );
  } else {
    return CircularProgressIndicator();
  }
}
```

Check connectionState to ensure the data is available

main.dart

```
builder: (BuildContext context, AsyncSnapshot<Post> snapshot) {
  if (snapshot.connectionState == ConnectionState.done) {
    return Column(
      children: <Widget>[
        Text(snapshot.data.title),
        SizedBox(height: 16),
        Text(snapshot.data.body),
      ],
    );
  } else {
    return CircularProgressIndicator();
  }
}
```



You can access the data through snapshot

main.dart

```
builder: (BuildContext context, AsyncSnapshot<Post> snapshot) {
  if (snapshot.connectionState == ConnectionState.done) {
    return Column(
      children: <Widget>[
        Text(snapshot.data.title),
        SizedBox(height: 16),
        Text(snapshot.data.body),
      ],
    );
  } else {
    return CircularProgressIndicator();
  }
}
```

Show an indicator that data is loading until it becomes available

More convenient way

A more convenient way, instead of checking connection state, you can check whether there is data exists or has any errors.

main.dart

```
if (snapshot.hasData) {
  return Column(
    children: <Widget>[
      Text(snapshot.data.title),
      SizedBox(height: 16),
      Text(snapshot.data.body),
    ],
  );
} else if (snapshot.hasError) {
  return Text("Error occurred!");
} else {
  return CircularProgressIndicator();
}
```

main.dart

```
if (snapshot.hasData) {
  return Column(
    children: <Widget>[
      Text(snapshot.data.title),
      SizedBox(height: 16),
      Text(snapshot.data.body),
    ],
  );
} else if (snapshot.hasError) {
  return Text("Error occurred!");
} else {
  return CircularProgressIndicator();
}
```