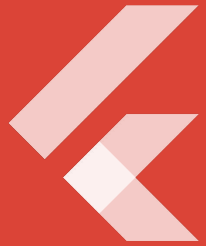
institute of CODING

Mobile Application Development



Flutter

Hero animations and map function

Northumbria University NEWCASTLE

Office for Students OfS

Previously,

We created two screens for our pseudo chat application. We have covered navigation between pages. Now, we will add some functionality.

Profile Picture widget

In chat applications, you can tap to profile picture and navigate to that contact's profile screen. In order to add that functionality, let's create a new widget for displaying profile picture. To do that, we will create `profile_picture.dart` file and create a new widget using the `CircleAvatar` widget.

profile_picture.dart

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

class ProfilePicture extends StatelessWidget {
  final String name;
  final String url;

  ProfilePicture(this.name, this.url);

  @override
  Widget build(BuildContext context) {
    return CircleAvatar(
      backgroundImage: NetworkImage(this.url),
    );
  }
}
```

profile_picture.dart

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

class ProfilePicture extends StatelessWidget {
  final String name;
  final String url;

  ProfilePicture(this.name, this.url);

  @override
  Widget build(BuildContext context) {
    return CircleAvatar(
      backgroundImage: NetworkImage(this.url),
    );
  }
}
```

Passing and storing url and name to widget.

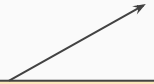
conversations.dart

```
ListTile(
  title: Text("Dennis Doe"),
  subtitle: Text("See you 👋"),
  leading: ProfilePicture("Dennis Doe", "https://i.pravatar.cc/600?u=1"),
  trailing: Text("09:30", style: Theme.of(context).textTheme.caption),
),
```

conversations.dart

```
ListTile(
  title: Text("Dennis Doe"),
  subtitle: Text("See you 👋"),
  leading: ProfilePicture("Dennis Doe", "https://i.pravatar.cc/600?u=1"),
  trailing: Text("09:30", style: Theme.of(context).textTheme.caption),
),
```

Usage of ProfilePicture.



Profile page

Now that we have profile picture widget is ready, we need a profile page to navigate to.

profile_picture.dart

```dart
class Profile extends StatelessWidget {
  final String name;
  final String pictureUrl;

  Profile(this.name, this.pictureUrl);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(backgroundColor: Colors.teal[700], title: Text(name)),
      body: Column(children: <Widget>[Image.network(pictureUrl)]),
    );
  }
}
```

Profile page

We need to navigate to that profile page after tapping the profile picture. But, profile picture does not contain `onTap` or `onClickled` property. How can we define such functionality? Using a `GestureDetector`.

profile_picture.dart

```
GestureDetector(
  onTap: () {
    Navigator.push(
        context,
        MaterialPageRoute(
          builder: (context) => Profile(name, url),
        ));
  },
  child: CircleAvatar(
    backgroundImage: NetworkImage(this.url),
  ),
);
```

profile_picture.dart

```
GestureDetector(
  onTap: () {
    Navigator.push(
        context,
        MaterialPageRoute(
          builder: (context) => Profile(name, url),
        ));
  },
  child: CircleAvatar(
    backgroundImage: NetworkImage(this.url),
  ),
);
```

You can wrap any widget to GestureDetector.

Hero Animations!

Isn’t it nice to animate profile picture when navigation to user profile? Flutter enables these inter-screen animations with an easy-to-use `Hero` widget. All you need to do is giving a unique `tag` to `Hero` widget on both screens.

profile_picture.dart

```
Hero(
  tag: "ProfilePicture",
  child: CircleAvatar(
    backgroundImage: NetworkImage(this.url),
  ),
)
```

profile_picture.dart

```
Hero(
  tag: "ProfilePicture",
   child: CircleAvatar(
     backgroundImage: NetworkImage(this.url),
   ),
),
```

Specify the same tag (unique in screen)

profile_picture.dart

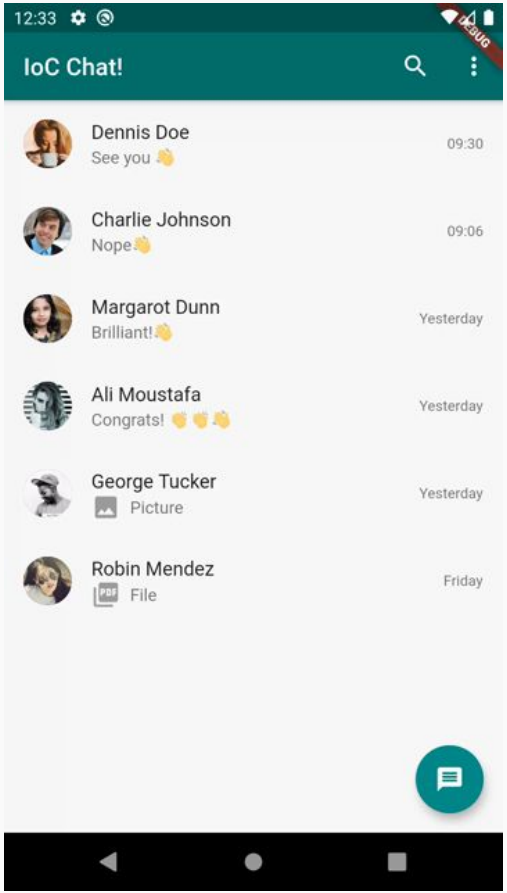
```
Hero(
  tag: "ProfilePicture",
  child: Image.network(pictureUrl),
)
```

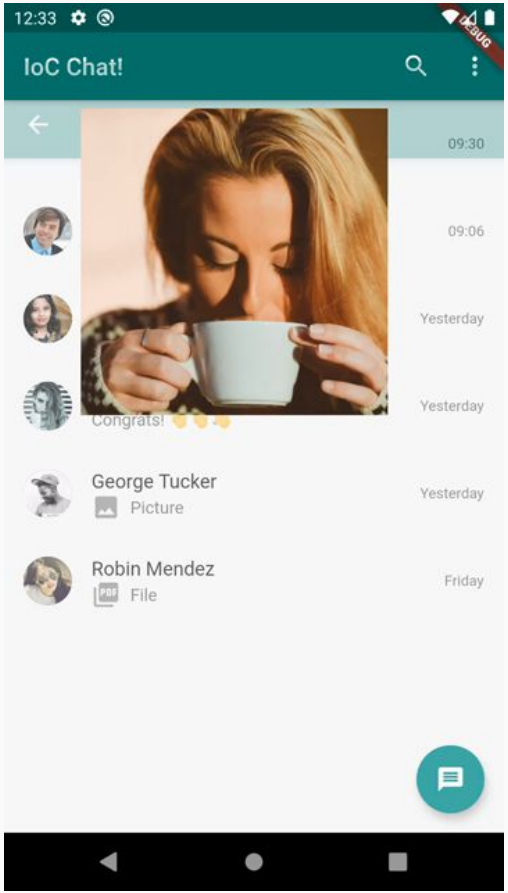
profile_picture.dart

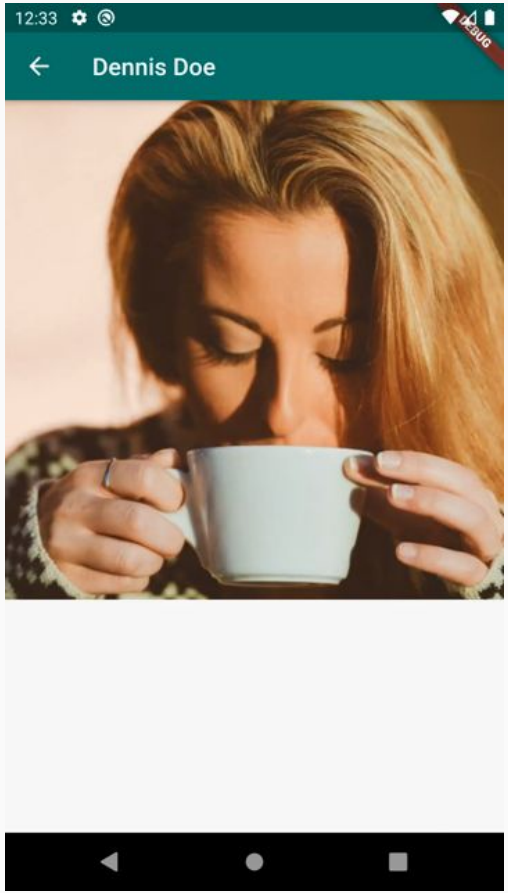
```
Hero(
  tag: "ProfilePicture",
  child: Image.network(pictureUrl),
)
```

Use the same tag

Hero animation preview







Refactoring the current infrastructure

To refactor and clean the current code infrastructure, first create `conversation.dart` class to hold all the necessary information for a conversation list. This will keep contact `name`, `profile picture` url, `when` the last message was sent and the content of `last sent message`.

conversation.dart

```
class Conversation {
  String name;
  String lastMessage;
  String profilePicURL;
  String when;

  Conversation(this.name, this.lastMessage, this.when, this.profilePicURL);
}
```

Refactoring the current infrastructure

Then put all the information on the main screen as a list of `Conversation` objects to be returned from a function.

conversation.dart

```
List<Conversation> getLastConversations() {
  return [
    Conversation("Dennis Doe", "See you 👋", "11:36", "https://i.pravatar.cc/500?u=1"),
    Conversation("Charlie Johnson", "Nope", "10:20", "https://i.pravatar.cc/500?u=2"),
    Conversation("Margarot Dunn", "Brilliant!", "08:00", "https://i.pravatar.cc/500?u=3"),
    Conversation("Ali Moustafa", "Congrats! 👋", "Saturday", "https://i.pravatar.cc/500?u=4"),
    Conversation(
        "George Tucker", "Check out this tweet", "Friday", "https://i.pravatar.cc/500?u=5"),
    Conversation("Robin Mendez", "Saw it", "Thursday", "https://i.pravatar.cc/500?u=6"),
  ];
}
```

Refactoring the current infrastructure

Then using the list we generated, we will revise the main screen's conversation list view.

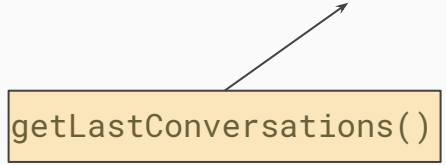
Dart language provides broad range of functions which makes it easier to refactor our current code base.

conversation.dart

```
ListView(
  children: getLastConversations()
      .map((conversation) => ListTile(
            title: Text(conversation.name),
            subtitle: Text(conversation.lastMessage),
            leading: ProfilePicture(conversation.name, conversation.profilePicURL),
            trailing: Text(conversation.when, style:
Theme.of(context).textTheme.caption),
          ))
      .toList(),
)
```

conversation.dart

List of `Conversation` object



```
ListView(
  children: getLastConversations()
      .map((conversation) => ListTile(
            title: Text(conversation.name),
            subtitle: Text(conversation.lastMessage),
            leading: ProfilePicture(conversation.name, conversation.profilePicURL),
            trailing: Text(conversation.when, style:
Theme.of(context).textTheme.caption),
          ))
      .toList(),
)
```

conversation.dart

List of `Conversation` object

```
ListView(
  children: getLastConversations()
      .map((conversation) => ListTile(
            title: Text(conversation.name),
            subtitle: Text(conversation.lastMessage),
            leading: ProfilePicture(conversation.name, conversation.profilePicURL),
            trailing: Text(conversation.when, style:
Theme.of(context).textTheme.caption),
          ))
      .toList(),
)
```

Map each `Conversation` object (given as `conversation`) inside the list of conversations to a `ListTile` object. Remember, arrow (`=>`) operator is actually a function which returns the `ListTile` object in this case.

conversation.dart