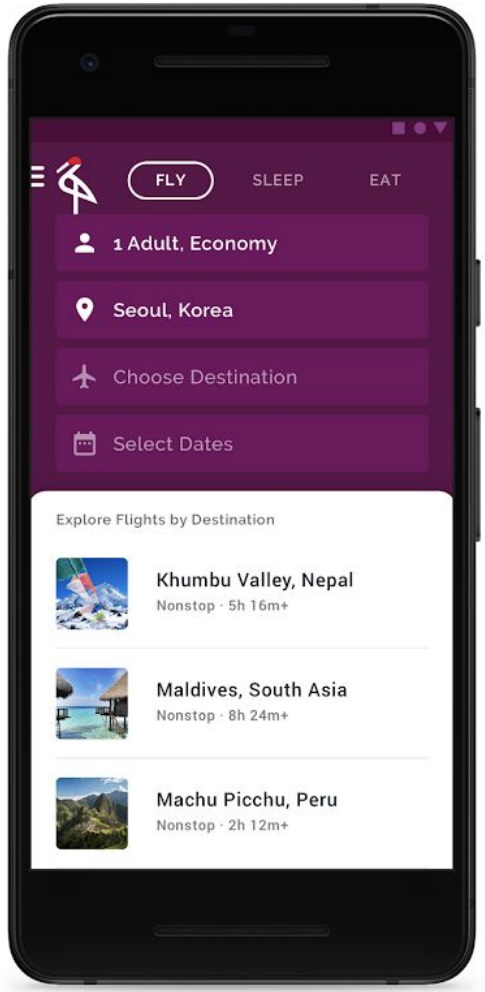
Flutter

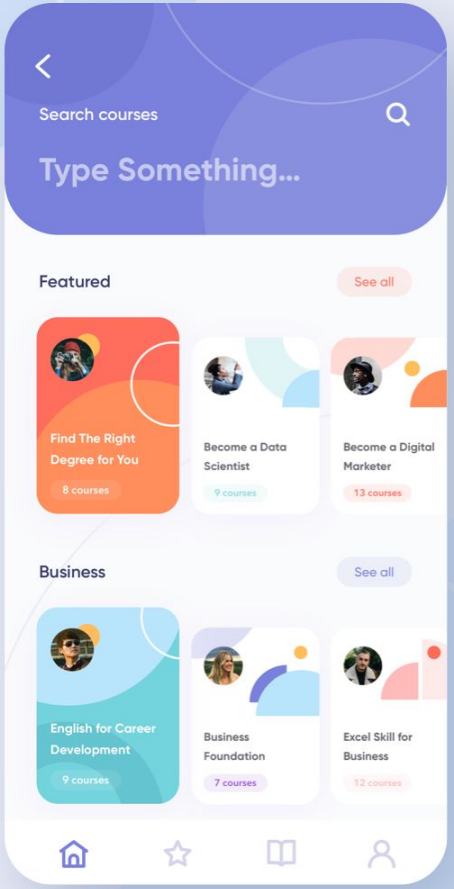
By Michelle Thaung and Eric Lo

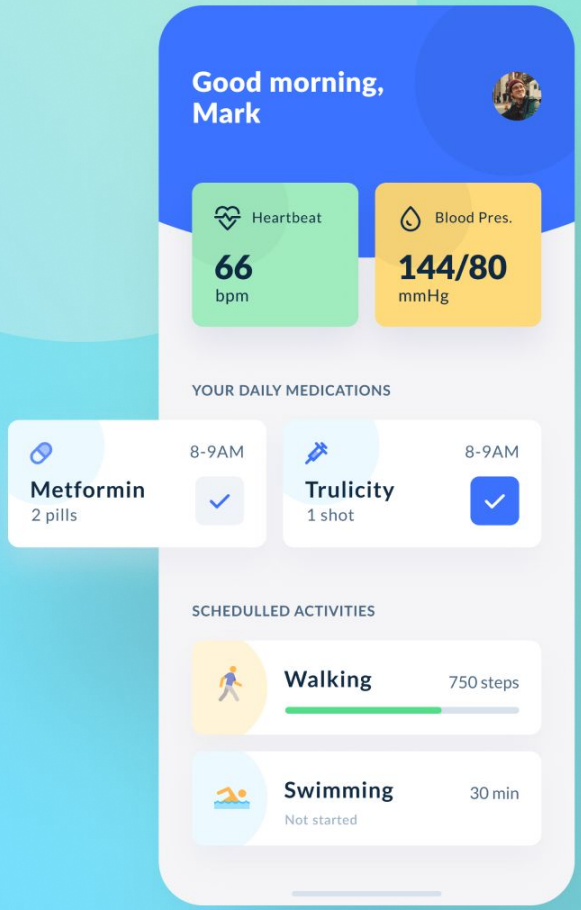
What is Flutter?

- Flutter is an SDK built for mobile app development for iOS and Android
- Created by Google in May 2017
- Open source
- Uses Dart as its programming language




Sourav Suman
App Developer
My Tasks
To Do
5 tasks now, 1 started
In Progress
1 tasks now, 1 started
Done
18 tasks now, 13 started
Active Projects
25%
Medical App
0 hours progress
60%
Making History Notes
20 hours progress


Search courses
Type Something...
Featured
See all
Find The Right Degree for You
8 courses
Become a Data Scientist
9 courses
Become a Digital Marketer
13 courses
Business
See all
English for Career Development
9 courses
Business Foundation
7 courses
Excel Skill for Business
12 courses


Good morning, Mark
Heartbeat
66
bpm
Blood Pres.
144/80
mmHg
YOUR DAILY MEDICATIONS
8-9AM
Metformin
2 pills
8-9AM
Trulicity
1 shot
SCHEDULLED ACTIVITIES
Walking
750 steps
Swimming
30 min
Not started

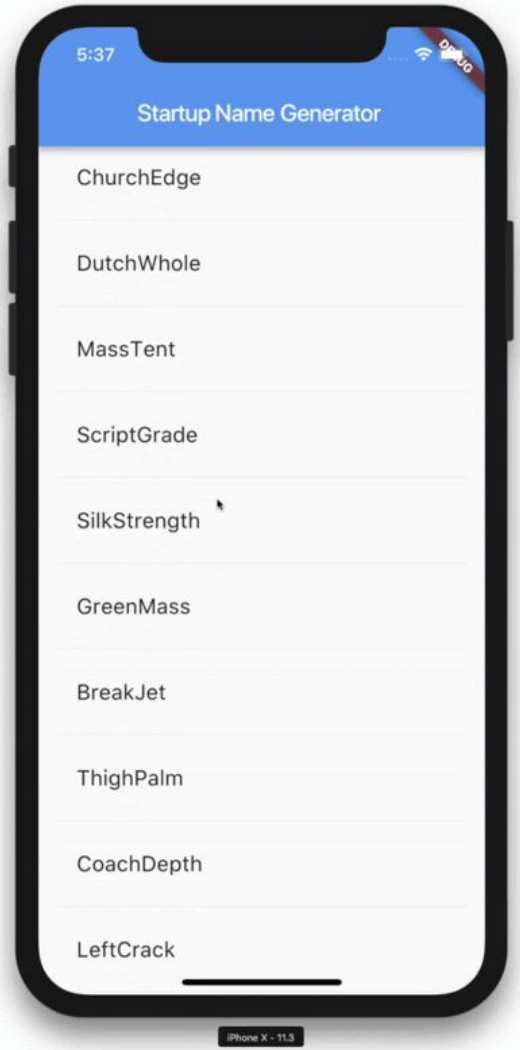
Why should we care?

- **Cross-platform** mobile app development
- Convenient to debug and fast to start up apps
- Growing in popularity

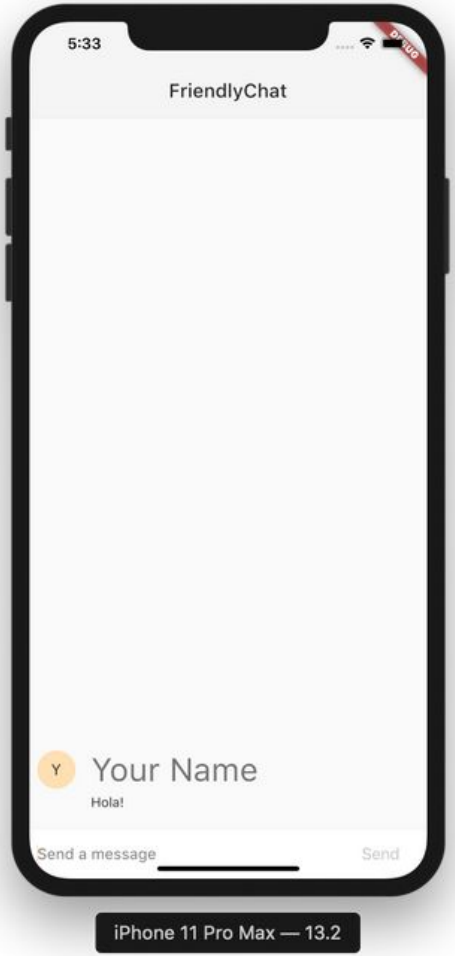
How does this relate to the work we do in this course?

- Useful in app development
- Write one thing, deploy in multiple environments (like Flask)
- Widgets are kind of like HTML classes
- Dart has C-style syntax, like Java

```
class ChatScreen extends StatefulWidget {
  @override
  _ChatScreenState createState() => _ChatScreenState();
}

class _ChatScreenState extends State<ChatScreen> with TickerProviderStateMixin {
  final List<ChatMessage> _messages = [];
  final _textController = TextEditingController();
  final FocusNode _focusNode = FocusNode();
  bool _isComposing = false;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('FriendlyChat'),
        elevation:
            Theme.of(context).platform == TargetPlatform.iOS ? 0.0 : 4.0,
      ), // AppBar
      body: Container(
        child: Column(
          children: [
            Flexible(
              child: ListView.builder(
                padding: EdgeInsets.all(8.0),
                reverse: true,
                itemBuilder: (_, int index) => _messages[index],
                itemCount: _messages.length,
              ), // ListView.builder
            ), // Flexible
            Divider(height: 1.0),
            Container(
              decoration: BoxDecoration(color: Theme.of(context).cardColor),
              child: _buildTextComposer(),
            ), // Container
          ],
        ), // Column
        decoration: Theme.of(context).platform == TargetPlatform.iOS
            ? BoxDecoration(
                border: Border(
                  top: BorderSide(color: Colors.grey[200]),
                ), // Border
              ) // BoxDecoration
            : null), // Container
    ); // Scaffold
  }

  Widget _buildTextComposer() {
    return IconTheme(
```


5:37
Startup Name Generator
ChurchEdge
DutchWhole
MassTent
ScriptGrade
SilkStrength
GreenMass
BreakJet
ThighPalm
CoachDepth
LeftCrack
iPhone X - 11.3

Demo


5:33
FriendlyChat
Y
Your Name
Hola!
Send a message
Send
iPhone 11 Pro Max — 13.2

Want to learn more?

- Great documentation on the official Flutter website, easy to read like DigitalOcean's
- Google every widget that you do not know

Step 3: Add a Stateful widget

State*less* widgets are immutable, meaning that their properties can't change—all values are final.

State*ful* widgets maintain state that might change during the lifetime of the widget. Implementing a stateful widget requires at least two classes: 1) a `StatefulWidget` class that creates an instance of 2) a `State` class. The `StatefulWidget` class is, itself, immutable and can be thrown away and regenerated, but the `State` class persists over the lifetime of the widget.

In this step, you'll add a stateful widget, `RandomWords`, which creates its `State` class, `_RandomWordsState`. You'll then use `RandomWords` as a child inside the existing `MyApp` stateless widget.

1. Create the boilerplate code for a stateful widget.
   In `lib/main.dart`, position your cursor after all of the code, enter **Return** a couple times to start on a fresh line. In your IDE, start typing `stful`. The editor asks if you want to create a `Stateful` widget. Press **Return** to accept. The boilerplate code for two classes appears, and the cursor is positioned for you to enter the name of your stateful widget.

2. Enter `RandomWords` as the name of your widget.
   The `RandomWords` widget does little else beside creating its `State` class.

   Once you've entered `RandomWords` as the name of the stateful widget, the IDE automatically updates the accompanying `State` class, naming it `_RandomWordsState`. By default, the name of the `State` class is prefixed with an underbar. Prefixing an identifier with an underscore enforces privacy in the Dart language and is a recommended best practice for `State` objects.

   The IDE also automatically updates the state class to extend `State<RandomWords>`, indicating that you're using a generic `State` class specialized for use with `RandomWords`. Most of the app's logic resides here—it maintains the state for the `RandomWords` widget. This class saves the list of generated word pairs, which grows infinitely as the user scrolls and, in part 2 of this lab, favorites word pairs as the user adds or removes them from the list by toggling the heart icon.

   Both classes now look as follows:

```
class RandomWords extends StatefulWidget {
  @override
  _RandomWordsState createState() => _RandomWordsState();
}

class _RandomWordsState extends State<RandomWords> {
  @override
  Widget build(BuildContext context) {
    return Container();
  }
}
```

Sources

Pictures:

https://flutter.dev/

https://swansoftwaresolutions.com/a-programming-language-called-dart-what-is-it-and-how-is-it-used/

https://blog.google/technology/developers/build-your-next-ios-and-android-app-flutter/

https://morioh.com/p/a6c11b4745b2

https://codelabs.developers.google.com/codelabs/first-flutter-app-pt1

https://codelabs.developers.google.com/codelabs/flutter/index.html#8

https://morioh.com/p/934139cb3030

https://github.com/TheAlphamerc/flutter_smart_course

Sources:

https://flutter.dev/docs/get-started/codelab

https://codelabs.developers.google.com/codelabs/flutter/index.html#8

https://nordicapis.com/what-is-the-difference-between-an-api-and-an-sdk/

https://www.ericdecanini.com/2020/05/25/flutter-vs-android-ios-hybrid-vs-native-app-development-in-2020/

https://www.futuremind.com/blog/pros-cons-flutter-mobile-development