

```
1
2 Dev Mobile 'Flutter' {
3
```

```
4 [Workshop]
5
```

```
6 <
```



```
7 I'm Aymene KRIOUDJ (aymen19s)
```



```
8 - Computer Science Engineer
```



```
9 - Mobile Developer | Freelancer
```



```
10 - Audit, Consulting, Project Management,
11 SI Engineering, Big Data, ML, BI ...
```

```
12 >
```

```
13 }
14
```

Table Of 'Contents' {

01 Theory Lesson

< Introducing Mobile Dev by
targeting Flutter ... >

02 Theory Lesson

< Learning Flutter ... >

03 Practical Lesson

< Practicing Flutter concepts ... >

04 Practical Lesson

< Coding Flutter ... >

}

```
1 01 {  
2  
3
```

```
4  
5 [Theory Lesson]  
6  
7
```

```
8 < Overview : Let's introduce Mobile Dev  
9 by targeting Flutter ... >  
10
```

```
11 }  
12  
13  
14
```

First 'Step' {

[Most important Skill to have as a Developer]



Googling

"Yes, read it right and again ..."

<p Knowing what to Google is one of the most important skills you can have as a developer.> </p>

}

Current 'Challenges' {

[Challenges of mobile development today]

To "the Native" approaches		To "Cross platform" approaches	
✓	High-quality apps Platform and system integrations	✓	Fast development Quick iterations, hot reload
✓	High-performance UIs Native code, GPU accelerated	✓	Portability, reach Single codebase
✗	Must fund two apps Two teams, codebases, & investments	✗	Poor Performance Slow, jerky, unpredictable
✗	Inconsistent brand, features Different across devices & OEMs	✗	Non-Native Look/Feel Users can tell the difference

}

```
1 Flutter '2.0' {  
2  
3     [Flutter offers the best of both worlds]  
4  
5     < "Flutter combines  
6       native performance and  
7       quality with high-velocity  
8       development and multi-  
9       platform reach." >  
10  
11  
12  
13  
14 }
```

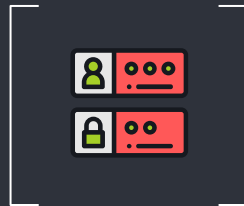
What makes 'Flutter' special {



< high-velocity
development >



< Expressive and
Flexible Toolkit >



< Native IOS and
Android App >

}

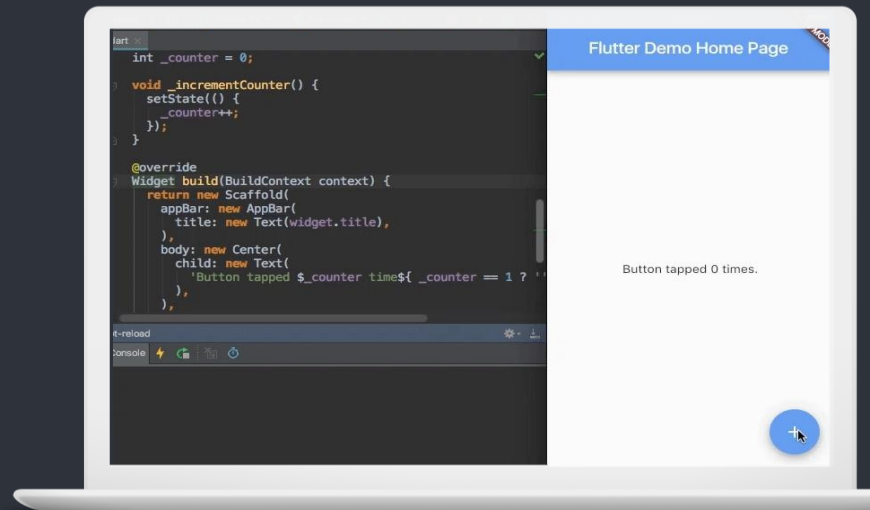
high-velocity development {

<

- Sub-second reload times
- Paint your app to life
- Iterate rapidly on features
- Test hypotheses quicker than ever
- More time to experiment & test features
- Single-codebase for faster collab
- 3X Productivity Gains

>

}

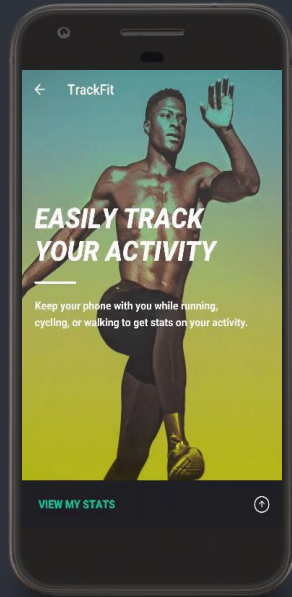


Flexibility and Control for beautiful UI's {

- < - Control every pixel on the screen
- Make your brand come to life
- Never say "no" to your designer
- Stand out in the marketplace
- Win awards with beautiful UI

>

}

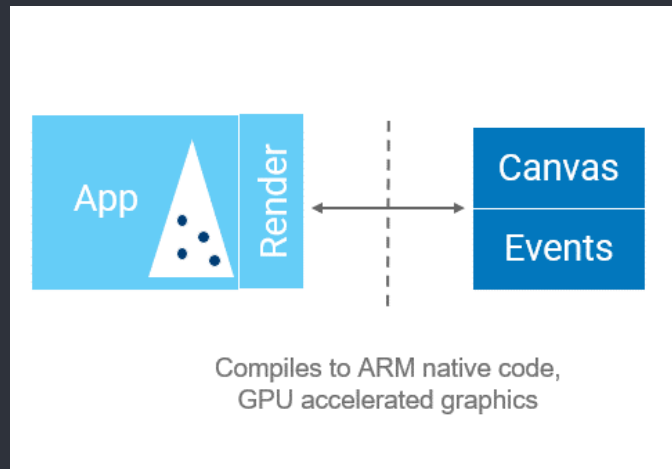


Natively- Apps for iOS and Android {

- < - Compiles directly to native ARM code Does not use a JavaScript bridge
- 60fps, GPU accelerated
- Smooth animations
- Deep platform integrations
- Natural look and feel
- Critical platform differences (scrolling, navigation, fonts)

>

}



4 ways to use Flutter today {

01 : Start a new
app from scratch

Build your new idea in Flutter and reach both
iOS and Android at the same time.

02 : Prototype
a new app idea

Use Flutter to test out an app concept or
idea in record time.

03 : Bring your
app to the the
other platform

You already have an iOS or Android app? Use
Flutter to build for the other platform.
Combine codebases when you've proven your
Flutter app.

04 : Use Flutter
for a part of
your app

Test Flutter in production with one or two
screens in your existing app.

}

Top Companies who built their Apps in Flutter {

" Flutter is one of the
top technologies right
now. It's used by Google,
developers and companies
around the world "



Summary; {

"Flutter is an open-source UI software development kit created by Google."

<p It is used to develop cross platform applications for **Android**, **iOS**, **Linux**, **Mac**, **Windows**, **Google Fuchsia** (operating systems such as Chrome OS), and the **Web** from a single codebase.> </p>

<p On **December 4th, 2018**, **Flutter 1.0** was released at the Flutter Live event, denoting the first "stable" version of the Framework.> </p>

<p On **March 3rd, 2021**, Google released **Flutter 2.0** during an online Flutter Engage event.> </p>

<p on **December 8th, 2021**, Google released the final stable release of Flutter. "**Hello and welcome to Flutter 2.8!**>"> </p>

}

```
1 02 {  
2  
3
```

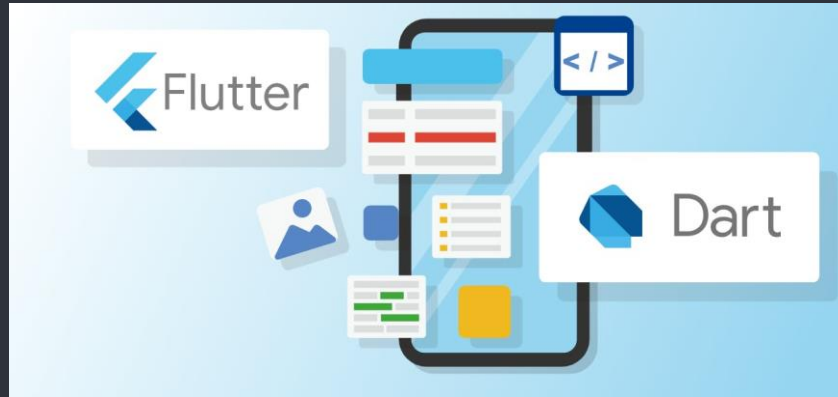
```
4  
5 [Theory Lesson]  
6
```

```
7  
8 < Let's learn Flutter ... >  
9
```

```
10  
11  
12 }  
13  
14
```

```
1 Language written for 'Flutter' {  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14 }
```

[What language is flutter built with ?]



Why 'Flutter' use 'Dart' {



< Fast development >



< Multi-plateform >



< Single codebase >

}

Aymen's Flutter 'RoadMap' {

Step 01 : Dart Language

The initial plan should be deeply focused over the language in which you will be building your app. So focus the mainly over **DART LANGUAGE**

Step 02 : Flutter Ecosystem

Explore the **Flutter Ecosystem** : yaml file, main method, runApp method, MaterialApp, StatelessWidget vs StatefulWidget ...

Step 03 : Flutter Widgets

You should be knowing the **mostly** used **widgets** in Flutter. Everything in Flutter is a **widget**.

Step 04 : Build your own creativity

Build your own App by linking Flutter concepts: **screen navigation**, **code management**, **App design**.

}

Aymen's Flutter 'RoadMap' {

Step 01 : Dart Language

The initial plan should be deeply focused over the language in which you will be building your app. So focus the mainly over **DART LANGUAGE**

Step 02 : Flutter Ecosystem

Explore the **Flutter Ecosystem** : yaml file, main method, runApp method, MaterialApp, StatelessWidget vs StatefulWidget ...

Step 03 : Flutter Widgets

You should be knowing the **mostly** used **widgets** in Flutter. Everything in Flutter is a **widget**.

Step 04 : Build your own creativity

Build your own App by linking Flutter concepts: **screen navigation**, **code management**, **App design**.

}

Dart 'language' {

[Variables and Types]

- `var bool` pressed = `false`;
- `const double` pi = 3,141592653589793;
- `int` count = 34;
- `double` price = 21.75;
- `String` name = "Aymene";
- `List` numbers = [12, 16, 21, 39];
- `List<int>` numbers = [12, 16, 21, 39];
- `Map` sounds= {"cat": "Meow", "dog": "Woof"};
- `DateTime` date = `DateTime`,`now()`;

}

Dart 'language' {

[Control flow statements]

```
1  if (year >= 2001) {  
2      print('21st century');  
3  } else if (year >= 1901) {  
4      print('20th century');  
5  }
```

```
6  for (var object in flybyObjects) {  
7      print(object);  
8  }
```

```
9  }
```

```
for (int month = 1; month <= 12;  
month++) {  
    print(month);  
}
```

```
while (year < 2016) {  
    year += 1;  
}
```

```
1  Dart 'language' {
2
3      [ Functions ]
4
5
6      int fibonacci(int n) {
7          if (n == 0 || n == 1) return n;
8          return fibonacci(n - 1) + fibonacci(n - 2);
9      }
10
11     var result = fibonacci(20);
12
13 }
14
```

Dart 'language' {

[Classes]

```
class User {  
  User(this.name, this.username,  
    this.age);  
  String name;  
  String username;  
  int age;  
}
```

```
user1 = User("user", "user1", 32);
```

```
}
```

```
class User {  
  User( { this.name, this.username,  
    this.age } );  
  String name;  
  String username;  
  int age;  
}
```

```
user2 = User(name: "user",  
  username: "user2");
```

```
1  Dart 'language' {  
2  
3      [ Tyranny Operator and Arrow syntax ]  
4  
5      • Pressed ? go : stop  
6  
7          bool apple = true;  
8              //apple red => true  
9              //apple yellow => false  
10         Container(  
11             color: apple ? Colors.red : Colors.yellow ;  
12         );  
13     }  
14 }
```

```
1 Dart 'language' {
```

```
2     [ Tyranny Operator and Arrow syntax ]
```

- **Function => one instruction**

```
6         bool hasEmpty = aListOfStrings.any((s){
7             return s.isEmpty;
8         });
```

Becomes:

```
12         bool hasEmpty = aListOfStrings.any((s) => s.isEmpty);
```

```
13     }
14 }
```


Aymen's Flutter 'RoadMap' {

Step 01 : Dart Language

The initial plan should be deeply focused over the language in which you will be building your app. So focus the mainly over **DART LANGUAGE**

Step 02 : Flutter Ecosystem

Explore the **Flutter Ecosystem** : yaml file, main method, runApp method, MaterialApp, StatelessWidget vs StatefulWidget ...

Step 03 : Flutter Widgets

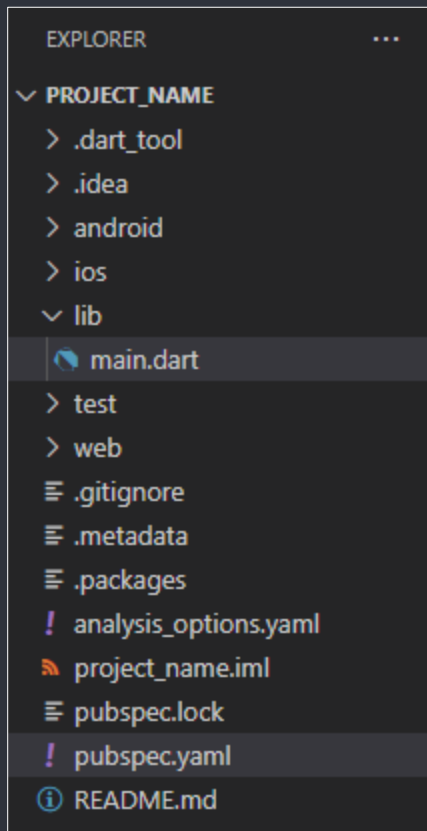
You should be knowing the **mostly** used **widgets** in Flutter. Everything in Flutter is a **widget**.

Step 04 : Build your own creativity

Build your own App by linking Flutter concepts: **screen navigation**, **code management**, **App design**.

}

```
1  Flutter 'Ecosystem' {
2
3      [ Flutter Project ]
4
5
6      • Create new project (command) :
7
8          flutter create project_name
9
10
11     • we are mostly interested in the 2
12       files :
13
14         main.dart & pubspec.yaml
15 }
```



Flutter 'Ecosystem' {

[pubspec.yaml]

}

```
! pubspec.yaml •
! pubspec.yaml
1  name: project_name
2  description: A new Flutter project.
3
4  publish_to: 'none' # Remove this line if you wish to publish to pub.dev
5
6
7  version: 1.0.0+1
8
9  environment:
10 |   sdk: ">=2.15.0 <3.0.0"
11
12 dependencies:
13 |   flutter:
14 |     sdk: flutter
15
16 # The following adds the Cupertino Icons font to your application.
17 # Use with the CupertinoIcons class for iOS style icons.
18 cupertino_icons: ^1.0.2
19
20 dev_dependencies:
21 |   flutter_test:
22 |     sdk: flutter
```

Flutter 'Ecosystem' {

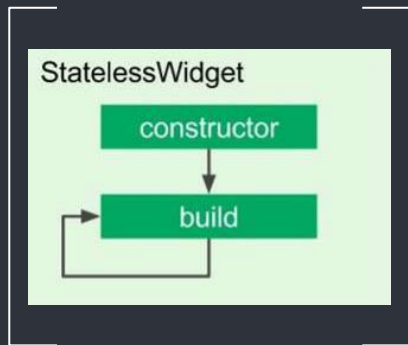
[main.dart]

}

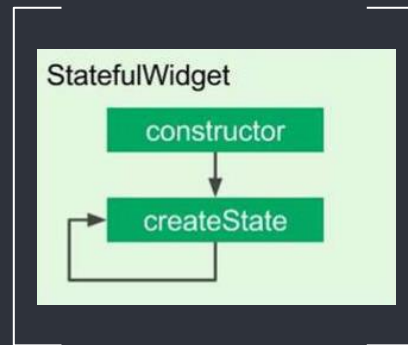
```
main.dart •
lib > main.dart > _MyHomePageState > build
1  import 'package:flutter/material.dart';
2
3  Run | Debug | Profile
4  void main() {
5    runApp(const MyApp());
6  }
7
8  class MyApp extends StatelessWidget {
9    const MyApp({Key? key}) : super(key: key);
10
11    // This widget is the root of your application.
12    @override
13    Widget build(BuildContext context) {
14      return MaterialApp(
15        title: 'Flutter Demo',
16        theme: ThemeData(
17          primarySwatch: Colors.blue,
18        ), // ThemeData
19        home: const MyHomePage(title: 'Flutter Demo Home Page'),
20      ); // MaterialApp
21    }
22  }
```

Flutter 'Ecosystem' {

[StatelessWidget vs StatefulWidget]



< A single StatelessWidget
can build in many different
Build Contexts :
AssetImage, Text ... >



< A StatefulWidget creates
anew State object for each
Build Contexts : Scrollable,
Animatable ... >

Flutter 'Ecosystem' {

[StatelessWidget vs StatefulWidget]

```
class MyHomePage extends StatelessWidget {  
  const MyHomePage({ Key? key }) : super(key: key);  
  
  @override  
  Widget build(BuildContext context) {  
    return Container(  
      );  
  }  
}
```

< StatelessWidget >

```
class MyHomePage extends StatefulWidget {  
  const MyHomePage({ Key? key }) : super(key: key);  
  
  @override  
  _MyHomePageState createState() => _MyHomePageState();  
}  
  
class _MyHomePageState extends State<MyHomePage> {  
  @override  
  Widget build(BuildContext context) {  
    return Container(  
      );  
  }  
}
```

< StatefulWidget >

}

Aymen's Flutter 'RoadMap' {

Step 01 : Dart Language

The initial plan should be deeply focused over the language in which you will be building your app. So focus the mainly over **DART LANGUAGE**

Step 02 : Flutter Ecosystem

Explore the **Flutter Ecosystem** : yaml file, main method, runApp method, MaterialApp, StatelessWidget vs StatefulWidget ...

Step 03 : Flutter Widgets

You should be knowing the **mostly** used **widgets** in Flutter. Everything in Flutter is a **widget**.

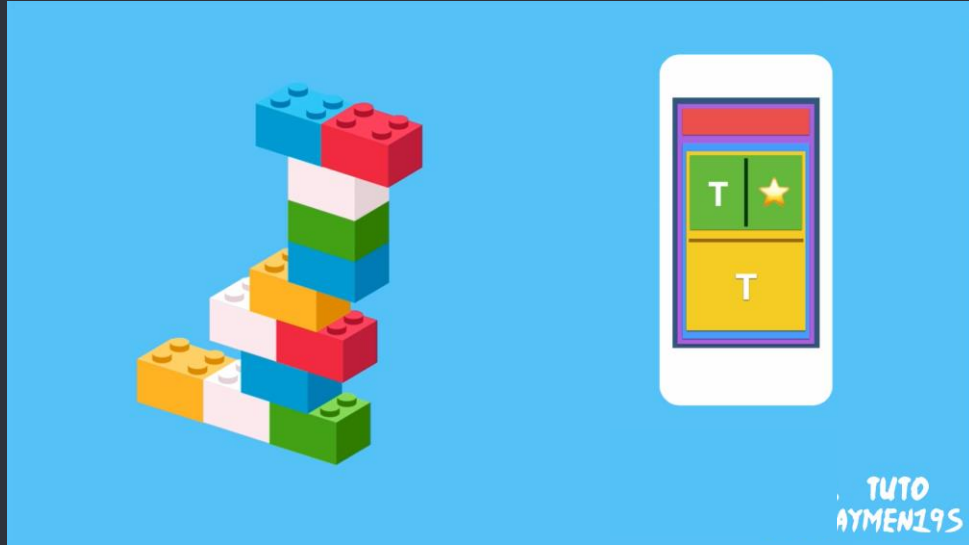
Step 04 : Build your own creativity

Build your own App by linking Flutter concepts: **screen navigation**, **code management**, **App design**.

}

Flutter 'Widgets' {

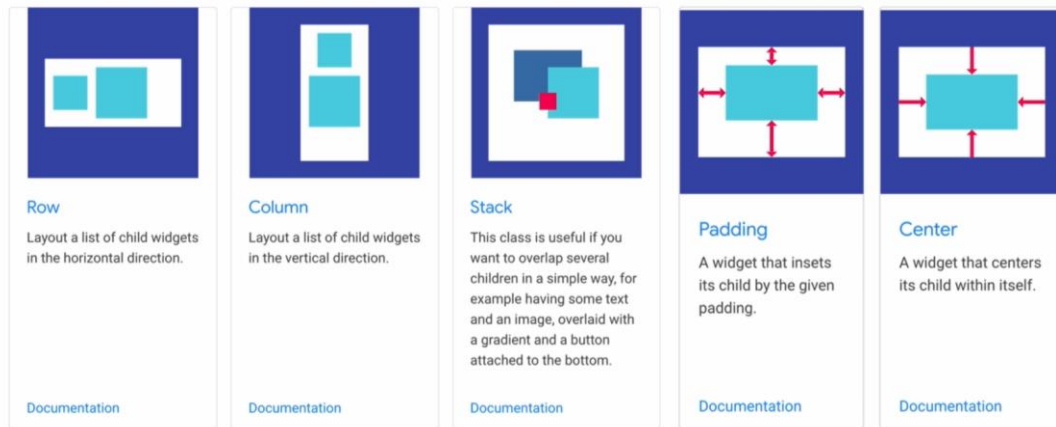
[It's all about widgets in flutter]



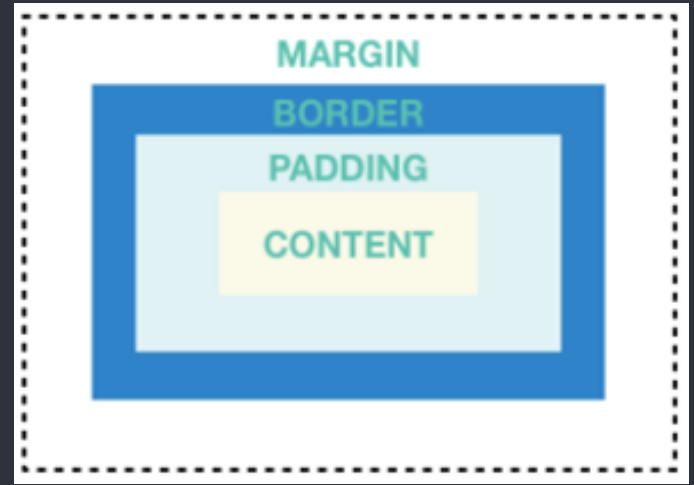
}

Flutter 'Widgets' {

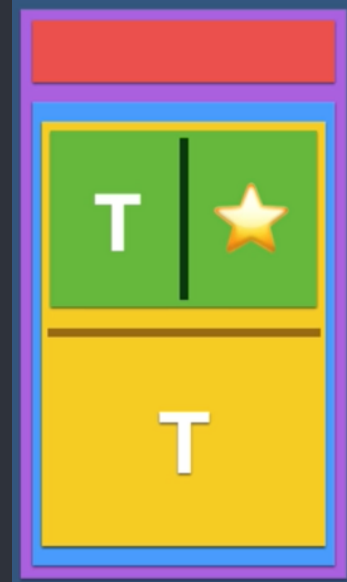
[Widget catalog]



```
1  Flutter 'Widgets' {  
2  
3  
4  
5  
6  
7  
8      [ Container ]  
9  
10  
11  
12  
13  
14 }
```



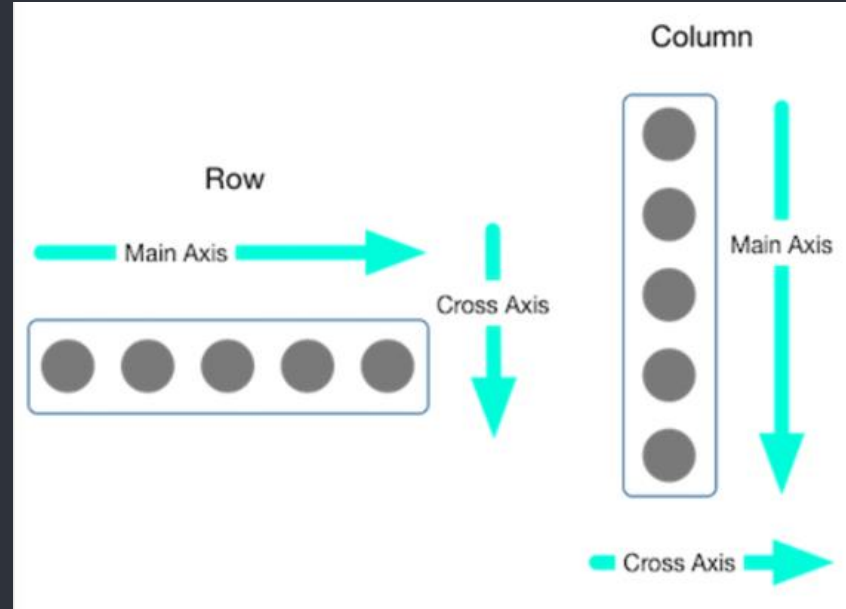
```
1  Flutter 'Widgets' {  
2  
3  
4  
5  
6  
7  
8      [ Text, Image, Icon ]  
9  
10  
11  
12  
13  
14 }
```



Flutter 'Widgets' {

[Rows & Columns]

}



main.dart

learning.dart

```
1 Flutter 'Widgets' {
```

```
2  
3  
4  
5  
6  
7  
8 [ Rows & Columns ]  
9  
10  
11  
12  
13  
14 }
```

justify-content

flex-start



flex-end



center



space-between



space-around



Half Equal Equal Half

space-evenly

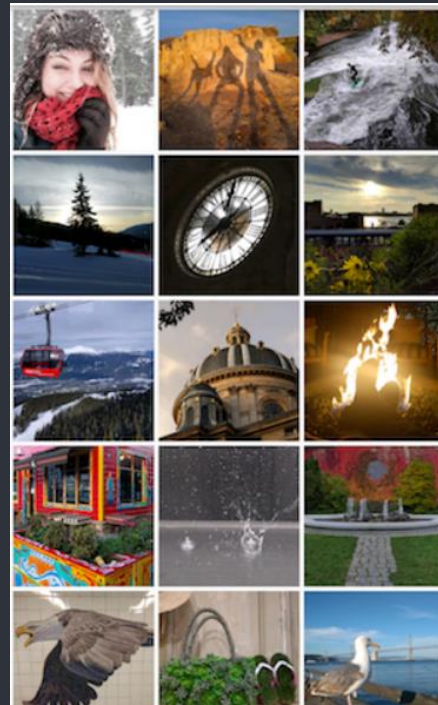


Equal Equal Equal Equal

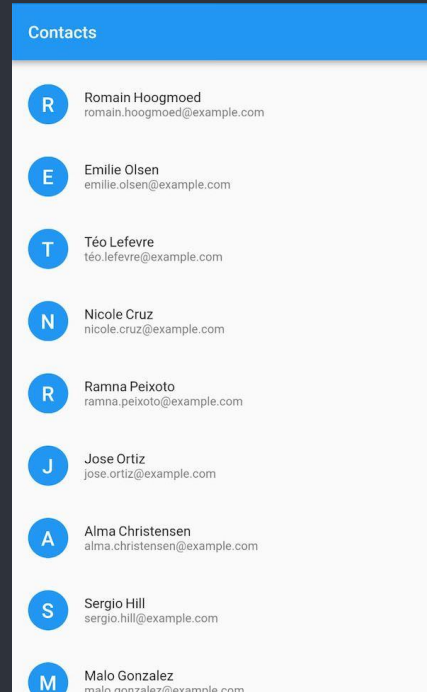
main.dart

learning.dart

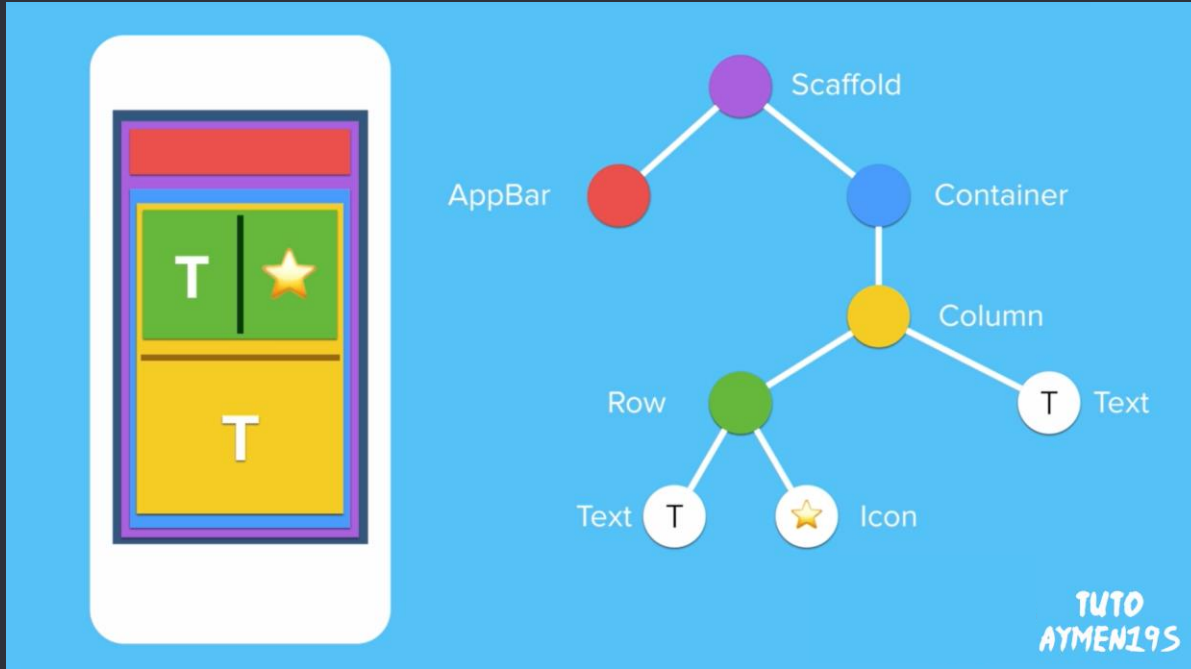
```
1  Flutter 'Widgets' {  
2  
3  
4  
5  
6  
7  
8      [ GridView ]  
9  
10  
11  
12  
13  
14 }
```



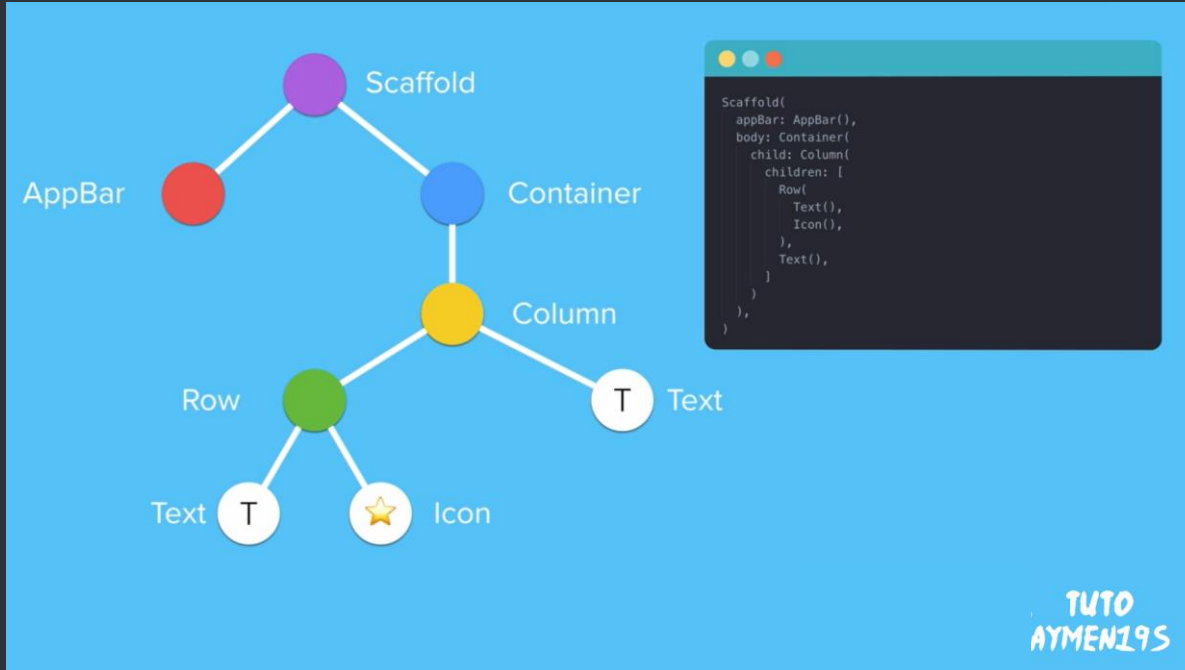
```
1 Flutter 'Widgets' {
2
3
4
5
6
7
8     [ ListView ]
9
10
11
12
13
14 }
```



Flutter 'Widgets' {



Flutter 'Widgets' {



Aymen's Flutter 'RoadMap' {

Step 01 : Dart Language

The initial plan should be deeply focused over the language in which you will be building your app. So focus the mainly over **DART LANGUAGE**

Step 02 : Flutter Ecosystem

Explore the **Flutter Ecosystem** : yaml file, main method, runApp method, MaterialApp, StatelessWidget vs StatefulWidget ...

Step 03 : Flutter Widgets

You should be knowing the **mostly** used **widgets** in Flutter. Everything in Flutter is a **widget**.

Step 04 : Build your own creativity

Build your own App by linking Flutter concepts: **screen navigation**, **code management**, **App design**.

}

```
1 Build your own 'App' {
```

```
2   'Some Mobile App  
3   Design concepts'
```

```
4  
5  
6  
7  
8   [ Minimize Cognitive Load ]  
9  
10  
11  
12  
13  
14 }
```



Number of Choices I Have



Amount of Thought Required



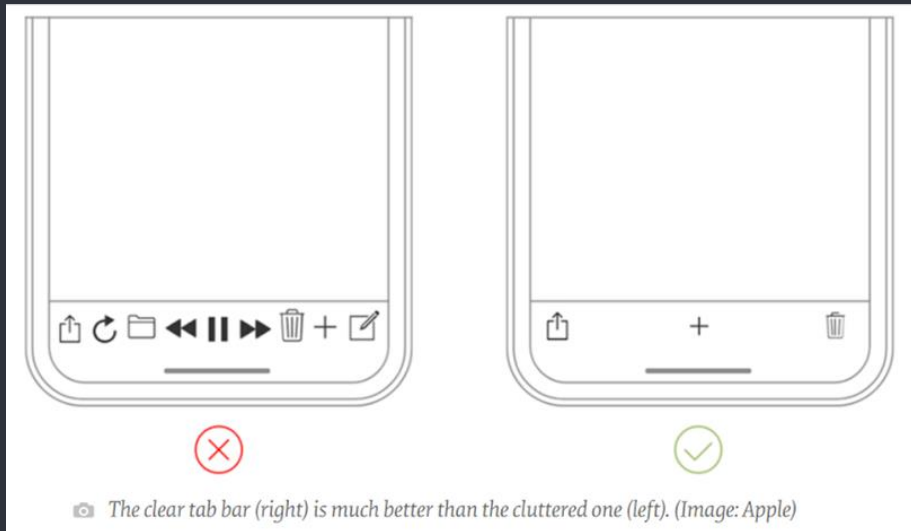
Confusion & Choice

```
1 Build your own 'App' {
```

```
2   'Some Mobile App  
3   Design concepts'
```

```
4  
5  
6  
7  
8   [ Decluttering ]
```

```
9  
10  
11  
12  
13  
14 }
```

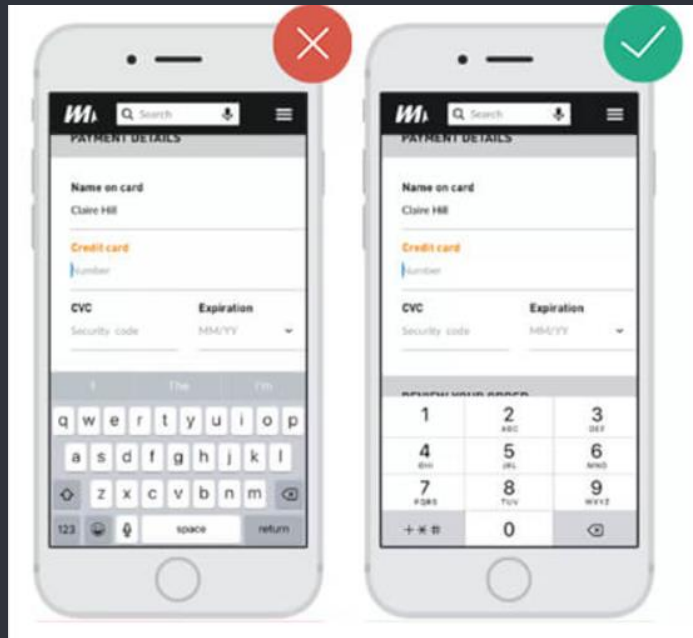


1 Build your own 'App' {

2 'Some Mobile App
3 Design concepts'

4 [Offload tasks & Minimize
5 efforts]

6
7
8
9
10
11
12
13 }
14

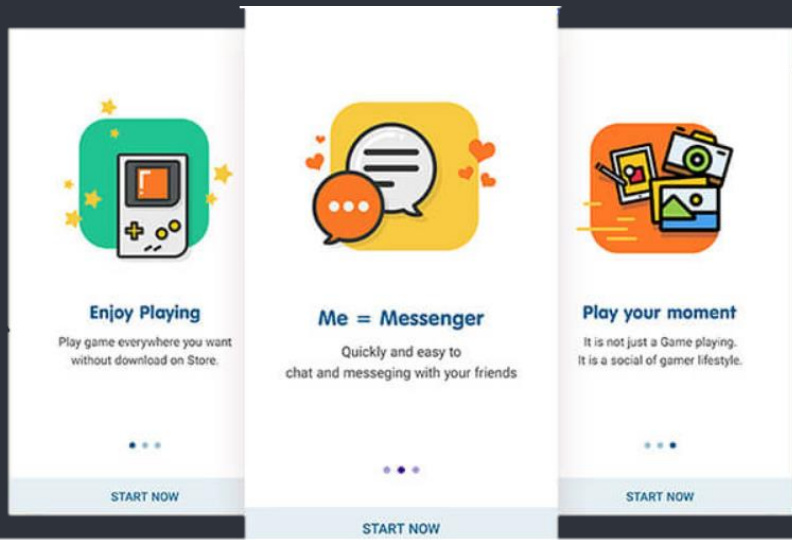


Build your own 'App' {

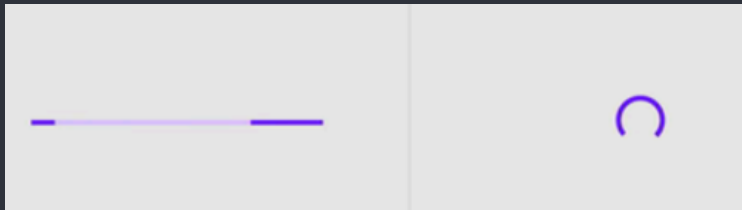
'Some Mobile App
Design concepts'

[Use familiar
screens]

}



```
1 Build your own 'App' {  
2  
3   'Some Mobile App  
4   Design concepts'  
5  
6  
7   [ Make it clear  
8   when loading is  
9   occurring ]  
10  
11  
12  
13 }  
14
```



```
1 03 {
2
3
```

```
4
5 [Practical Lesson]
6
```

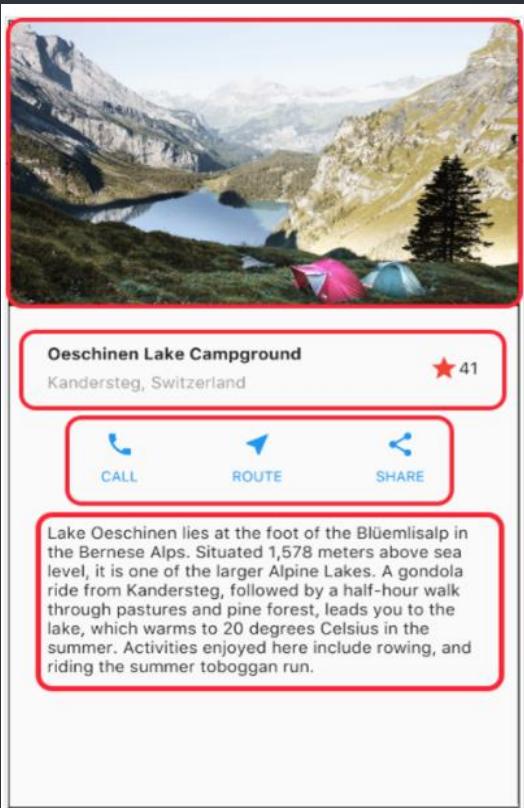
```
7
8 < Let's practice Flutter concepts ... >
9
```

```
10
11 }
12
13
14
```


Exercise < /1 > {

- Which widgets are used in this screen?
- Draw the widget tree.

}



Exercise < /2 > {

- Which widgets are used in this screen?
- Draw the widget tree.

}



1 04 {
2
3

4
5 [Practical Lesson]
6

7
8 < Let's start coding Flutter ... >
9

10
11
12 }
13
14

Exercise < /1 > {



- Open DartPad in your browser.
- Draw the widget tree of this screen.
- Build the screen in DartPad.

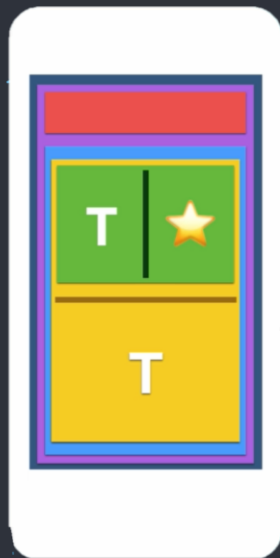
}






Exercise < /2 > {



- Create new project in your desktop.
- Open your code editor (preferably VsCode).
- Build “_____” App together.

}



```
1 Thanks; {
2
3     'Do you have any questions?'
4
5     ha_krioudj@esi.dz
6     +213 656 12 37 58
7
8         
9     linkedin.com/in/aymene-krioudj-685883162/
10    instagram.com/aymen.krdj/
11    facebook.com/aymene19s/ (AymEn Krdj)
12    twitter.com/aymen\_krdj/
13    github.com/aymenkrioudj/
14 }
```

Alternative 'Resources' {

Here is an assortment of alternative resources to help you learn more about Flutter

- * The official website "flutter.dev".
- * [Flutter](#) Medium articles.
- * YouTube playlist "[Widget of the week](#)".
- * The Complete 2021 Flutter Development Bootcamp with Dart by [The App Brewery](#).
- * Stack Overflow "Flutter questions & answers".

}

Acknowledgements & 'Rating' }

Thank you for participating in my session with Openminds. I hope you had as much fun attending as we had delivering them.

I invite you to fill this [anonymous form](#) and be as honest as possible 🙏

Your opinion matters to me, some constructive criticism is always welcomed so we can be the best version of ourselves 🙏

Thanks in advance and until the next adventure! 🙏

