CSI App - dev Bootcamp!

Welcome everyone

Bonjour, everyone 👋

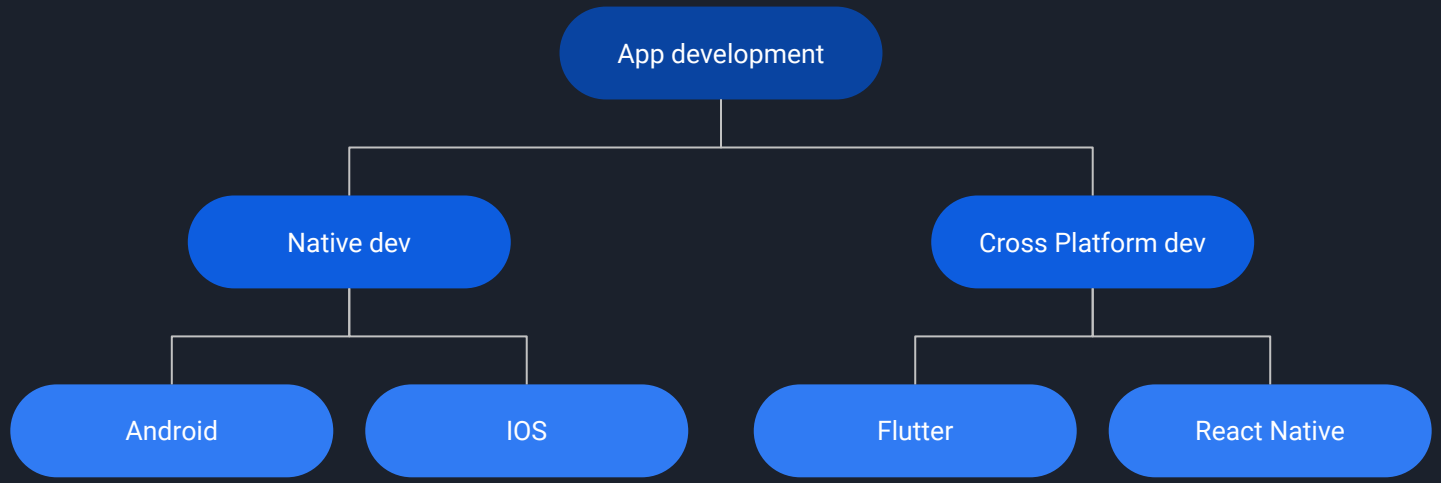
- Tech Captain, CSI
- Beta Microsoft Learn Student Ambassador, Microsoft
- GitHub Campus Expert, GitHub

Brief overview

- Flutter installation
- Building basic UI's
- Introduction to OOP!
- Packages, widgets……..
- A taste of Git and GitHub

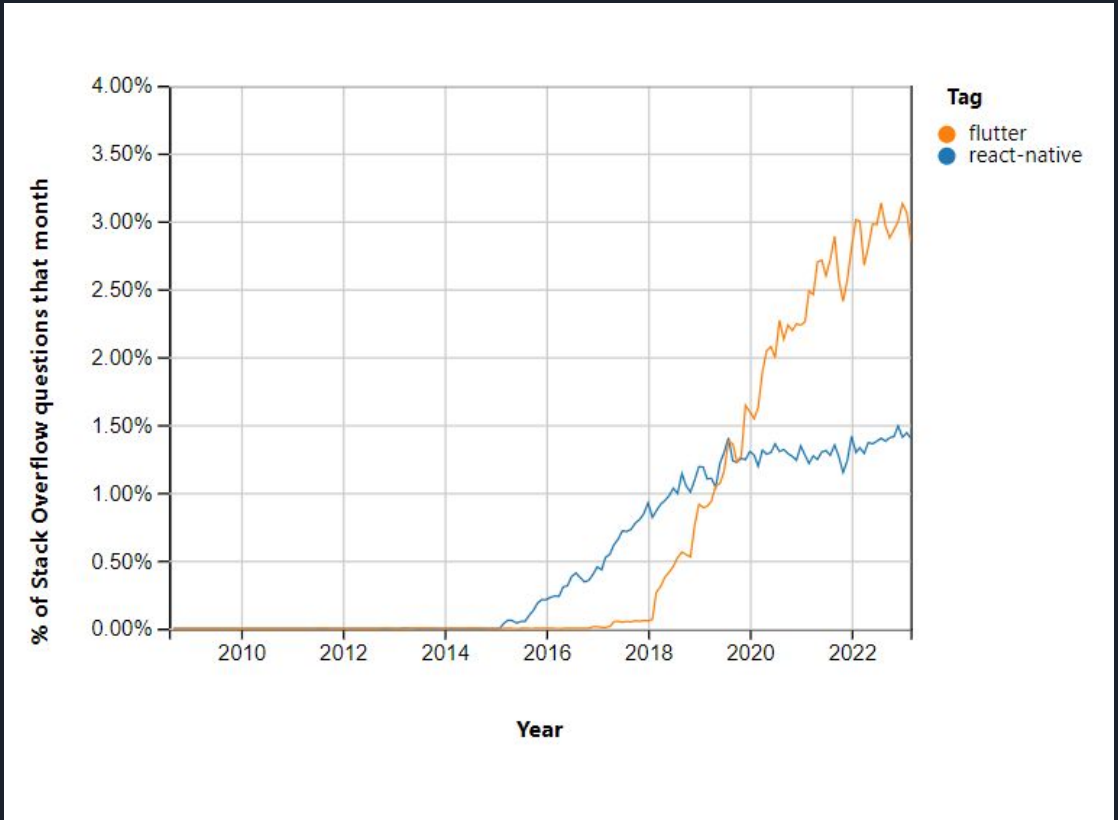
What should you expect? (Realistically)

Let’s start from the absolute basics



Say hello to Flutter!

Flutter V/S React native



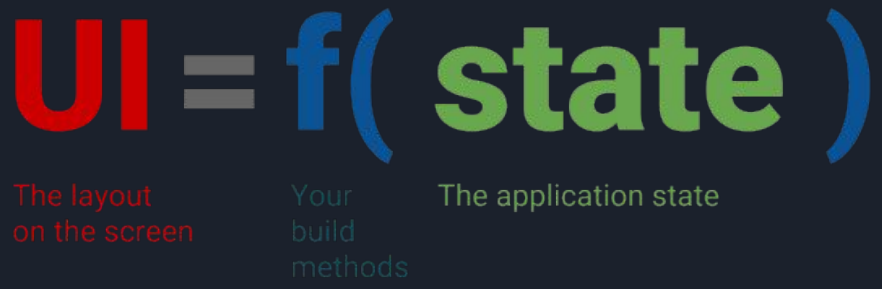
What are Widgets?

A widget is the most basic UI component/element in Flutter

The idea is you build your UI around Flutter

State in Flutter

In Flutter, "state" refers to data that can change and affect the appearance of the app's user interface.



Stateful V/S Stateless Widgets

State”less” widgets: Do Not maintain state

Eg: Static apps such as Calculator, Gallery, Contacts

State”ful” widgets: Maintain the state of the app

Eg: Dynamic apps which change UI periodically such as news app, Netflix, Weather app, Todo app

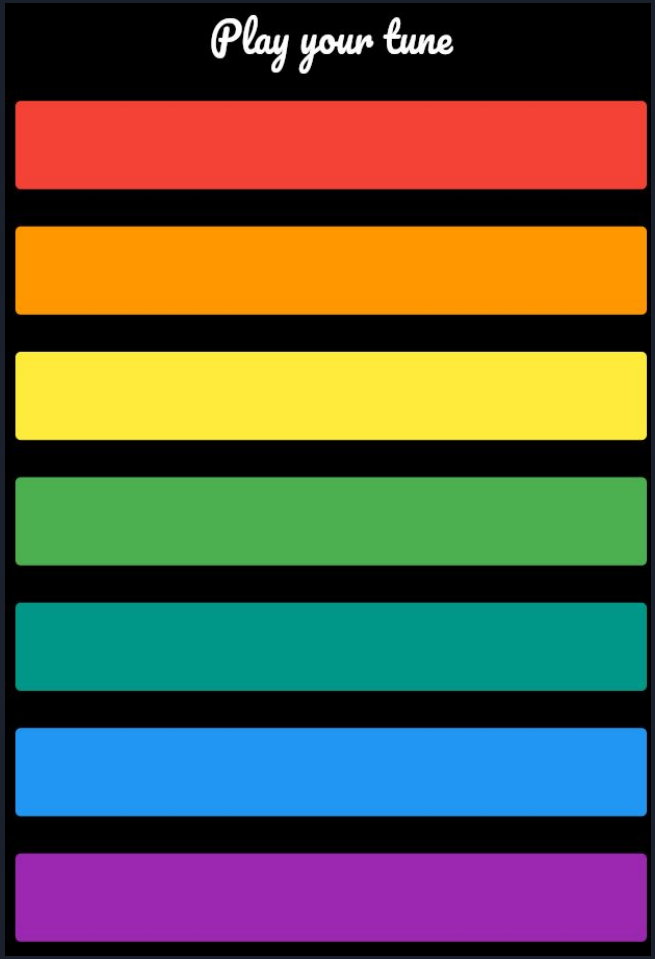
Layout Building

A brief list of UI widgets we will be using today:

1. Column
2. Padding
3. Container
4. Sized Box
5. Elevated Button
6. Scaffold

App - 1: Xylophone App

Play multiple sounds when clicked



Scaffold widget

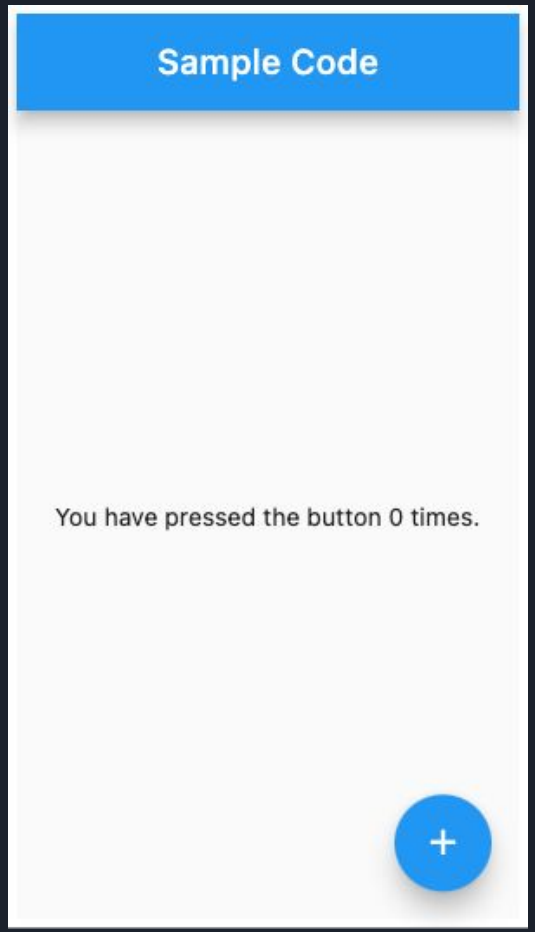
The idea is whenever you make a new widget, return a scaffold!

Scaffold is a widget that provides a basic structure for building a Material Design app.

Scaffold provides the following:

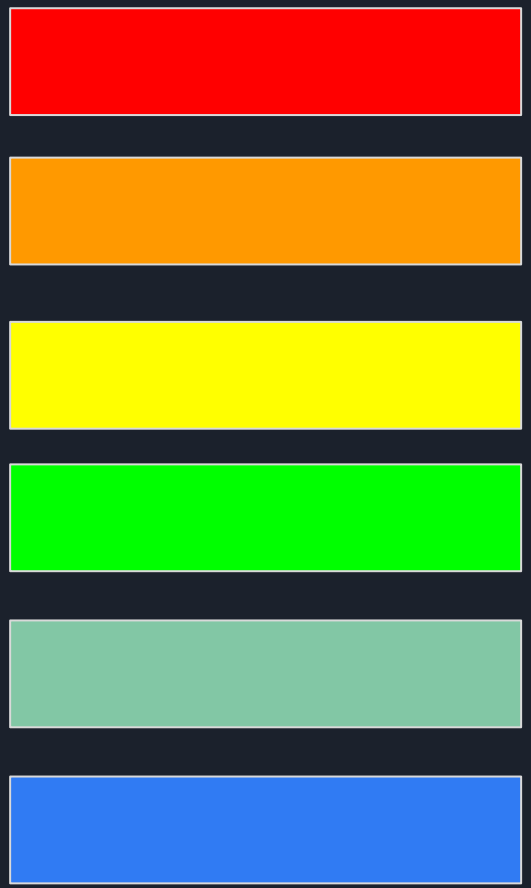
1. AppBar
2. Bottom Navigation Bar
3. The body
4. A side drawer
5. Floating Action Button

Scaffold with some of its properties



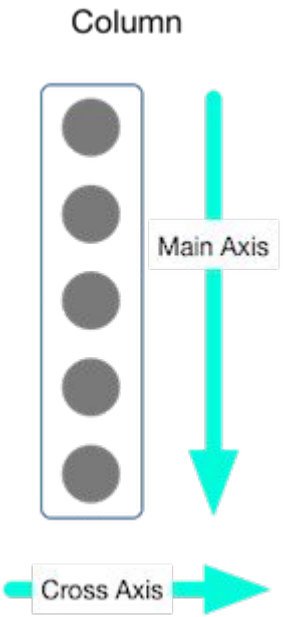
Let’s start building

We have to place the rectangles such that they stack on top of each other



Column widget

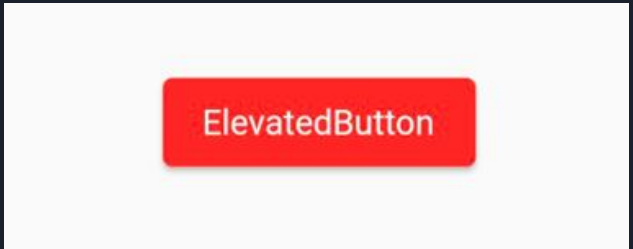
The Column widget is commonly used to create vertical arrangements of widgets, such as a list of items, a form, or a group of related information.



Elevated button widget

It’s one of the many buttons available in flutter

The idea is you press on it and it triggers something



The text and the center widget

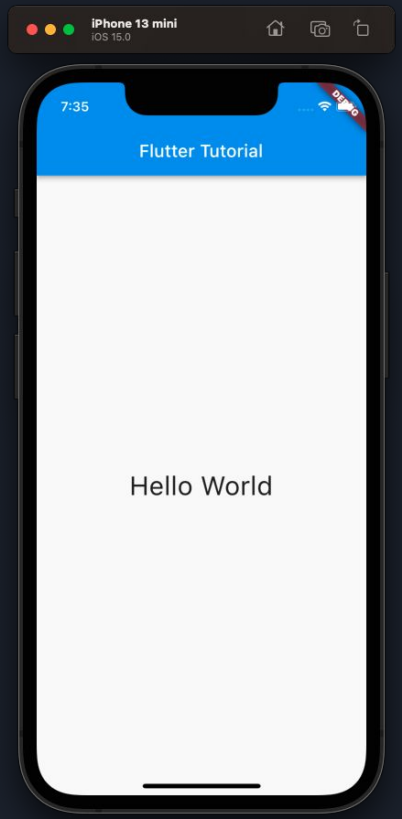
Text: Use this to add text (simple!)

Center: Use this to center anything

How do we achieve something like this?

There is text and its centered

What if there was a way to overlap them?



Child of a widget

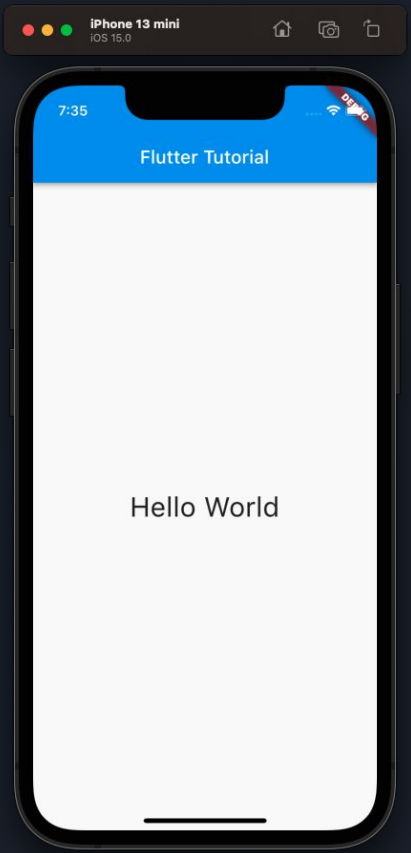
Some widgets have a property called child which allows you overlap it with another widget of desire

dart

```dart
Center(
  child: // Child widget goes here
)
```

dart

```dart
Center(
  child: Text('Hello World'),
)
```



Inheritance in OOP:

Keep in mind two things:

1. Who's the parent?
2. Who's the child?

The crux is the child inherits the properties of its parent. (It's that simple)

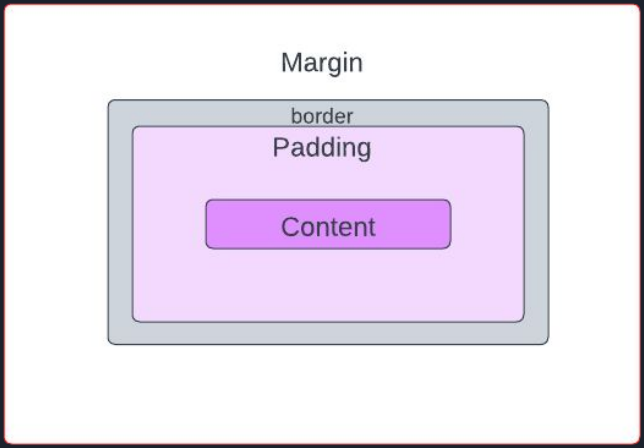
Padding Widget

What is Padding?

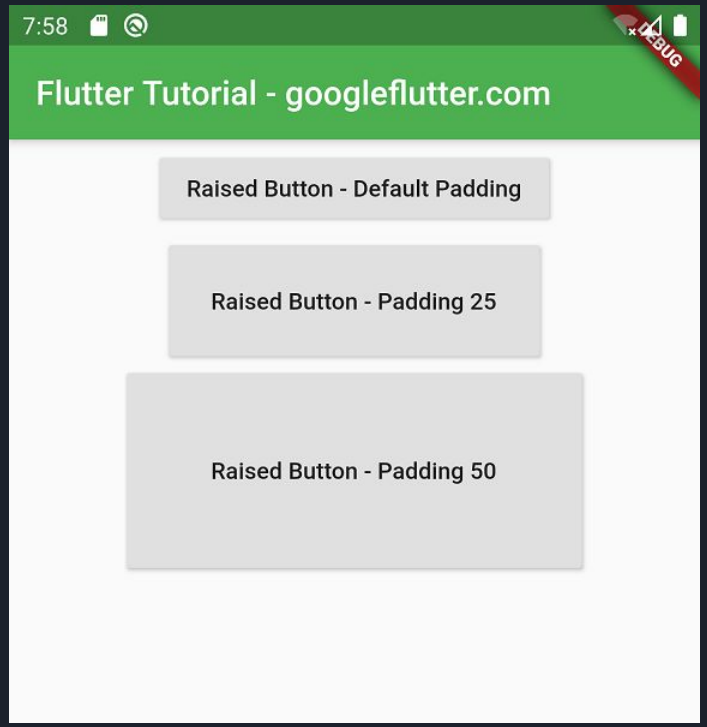
padding refers to the space between the content (text, images, buttons, etc.) and the edge of a container (such as a box, frame or window).

less

```
Padding(
  padding: EdgeInsets.all(16.0),
  child: // child widget goes here
)
```



A simple example of Padding



Sized Box widget

This widget is usually used to size any widget that we cannot directly size.

less

```
SizedBox(
  width: widthValue,
  height: heightValue,
  child: childWidget,
)
```

Understanding packages:

In programming, a package is a collection of related code files and resources that are designed to provide specific functionality.

Packages are a way of organizing code in a structured and modular way, which makes it easier to manage and reuse code across different projects.

Using packages can save a lot of time and effort for developers, as they can leverage pre-existing code to accomplish a specific task rather than starting from scratch.

This makes the development process faster and more efficient, as developers can focus on solving the unique challenges of their project rather than writing code that has already been written and tested.

Pub.dev for more flutter and dart packages

Functions in dart

In a nutshell

arduino

Copy code

```arduino
return_type function_name(parameter1, parameter2, ...) {
  // Function body
  // Code to be executed
  return value;
}
```

A break-down of functions:

1. Return Type: This is the data type of the value that the function will return after it is executed.
2. Function Name
3. Parameters: List of things that the function accepts
4. Function Body: This is where you write code that gets executed when the function is called

arduino

```
return_type function_name(parameter1, parameter2, ...) {
  // Function body
  // Code to be executed
  return value;
}
```

A simple example on functions:

arduino

```
double calculateArea(double length, double width) {
  double area = length * width;
  return area;
}
```

Abstraction in OOP

Abstraction is a fundamental concept in computer science and software engineering that refers to the process of simplifying complex systems by focusing on the essential details and hiding unnecessary complexity.

Abstraction is the process of hiding the implementation details of a system and only showing the necessary information to the user.

List in programming

In a nutshell

In programming, a list is a data structure that represents an ordered sequence of elements. It allows you to store multiple values of the same or different data types in a single variable, making it easier to work with large amounts of data.

bash

```
List<type> listName = [element1, element2, ..., elementN];
```

An example on Lists:

arduino

```
List<int> numbers = [1, 2, 3, 4, 5];
```

Git (A version control system)

In a nutshell

Allows developers to keep track of changes made to code over time.
Git is used to:

1. Collaborate on code with others
2. Track changes to code
3. Revert changes
4. Backing up code

We made it!