

W4-S1 PRACTICE

ASSETS & STATELESS WIDGETS

Learning objectives

- ✓ Manipulate **Column** layout with **stretch** alignment
- ✓ Use **Icons**, **Image** and **Card** widgets
- ✓ Use **Enums** with attributes to specify a data model
- ✓ Create re-usable **StatelessWidget**
- ✓ Manage **required** and **optional** widget properties



No AI tools allowed to solve this practice



How to submit?

- ✓ **Push** your final code on **your GitHub repository**
- ✓ Then **attach the GitHub path** to the MS Team assignment and **turn it in**

Before practice, to be prepared!

Read the following documentation to be ready for this practice:

<https://www.classcentral.com/classroom/youtube-flutter-tutorial-for-beginners-45851/60c82bddaba3e>

<https://www.classcentral.com/classroom/youtube-flutter-tutorial-for-beginners-45851/60c82bddaba15>

<https://www.youtube.com/watch?v=GPoRjSjd1cl>

<https://api.flutter.dev/flutter/material/Card-class.html>

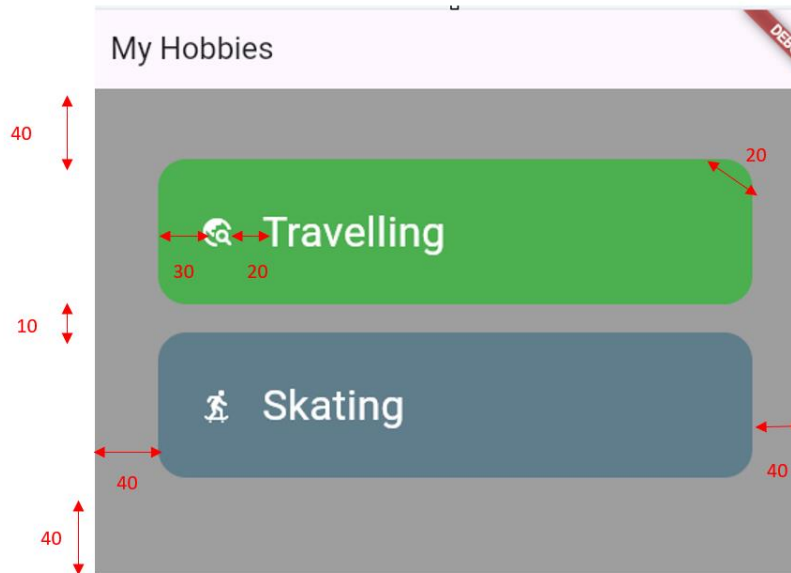
<https://www.classcentral.com/classroom/youtube-flutter-tutorial-for-beginners-45851/60c82bddaba0a>

<https://api.flutter.dev/flutter/widgets/Image-class.html>



EX 1 – The hobbies

In this exercise, you need to arrange **hobbies cards** vertically, each containing a hobby with an icon and text.

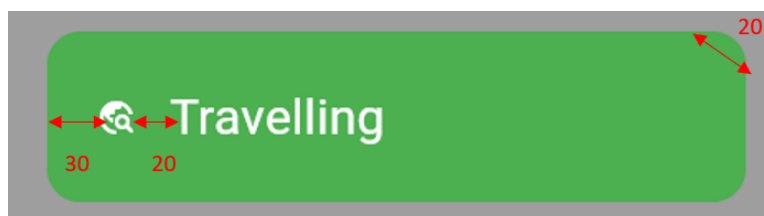


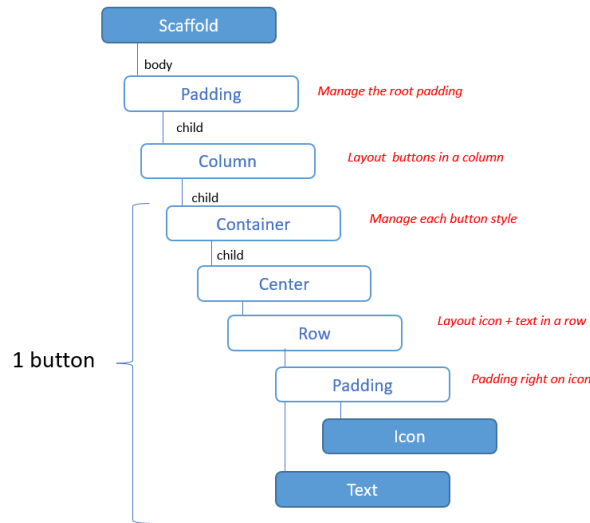
This exercise has 2 parts:

- **Part 1:** create a single hobby card
- **Part 2:** extract the card into a stateless widget

PART 1 – Build the UI

Start by create a single hobby card





- We use **CrossAxisAlignment.stretch** on Column, so that the children to take up the entire width of the parent. [More about Colum here](#)

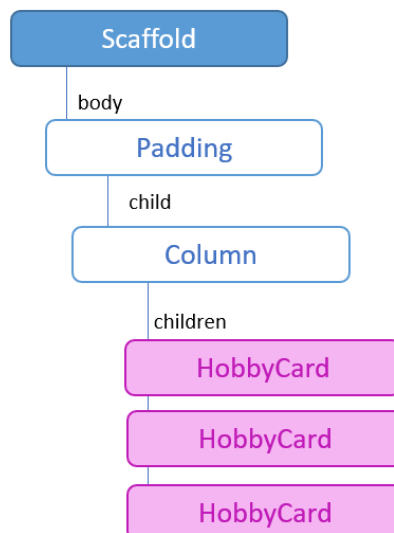
What is the difference between `crossAxisAlignment` and `mainAxisAlignment` in Column widget? Try it out to understand.

- We use predefined icons from the **Icons class**. [More about icons here](#)
- The button radius is performed using a **BoxDecoration** on the container. [More about box decoration here](#)

PART 2 – Create a HobbyCard widget

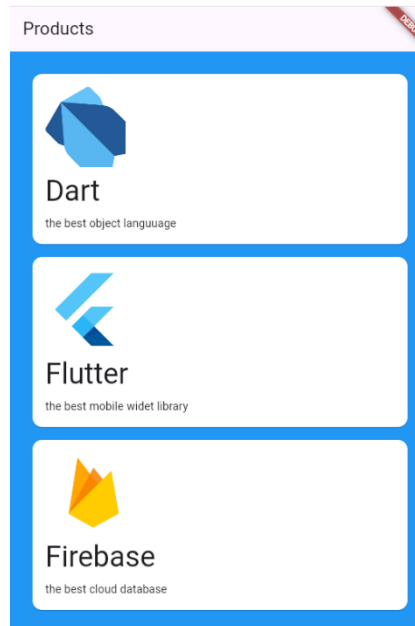
Extract the hobby card into a stateless widget called **HobbyCard** that takes the following parameters:

- The hobby title (String, *required*)
- The hobby icon (IconData, *required*)
- The card background color (Color, *optional*, default value = BLUE)



EX 2 – The products

In this exercise, you need to arrange **product cards** vertically, each containing a product image, title and description.

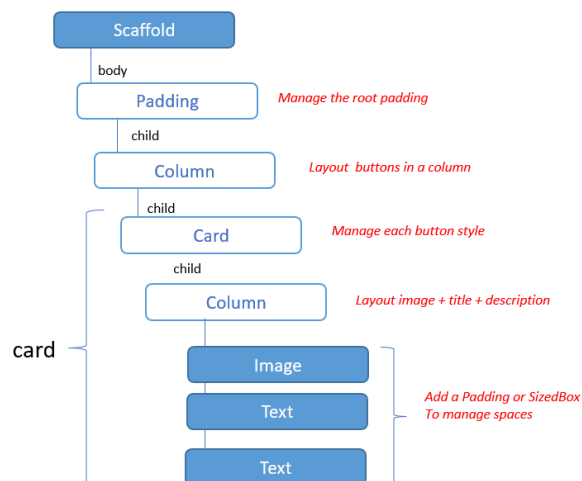


This exercise has 3 parts:

- **Part 1:** create a single hobby card
- **Part 2:** create a **Enum** gathering each product information
- **Part 2:** extract the card into a stateless widget

PART 1 – Build the UI

Start by create a single card



- We use Card widget, to style the card. [More information about card here](#)

- We use Image widget, to display images. More information [here](#) and [here](#).
 - o We will store images in asset folder
 - o The 3 images are provided in the ZIP folder

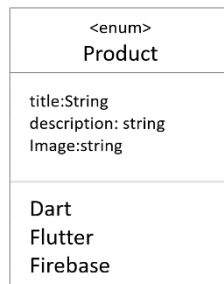
PART 2 – Create a Enum product

To store our product, we are using enum. TO learn more about enum: [here](#)

Enums are a **special kind of class** used to represent **fixed number of constant values**.

Why enum? Because our products are always the same and will not change over time

Enum can also have attributes and constructors, and this is what we are doing right now.

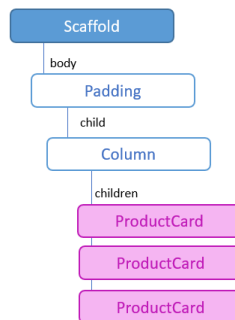


- Start to create a **Product enum**, following the above UML.
- The enum has 3 values: dart, flutter, firebase.

PART 3 – Create a Product widget

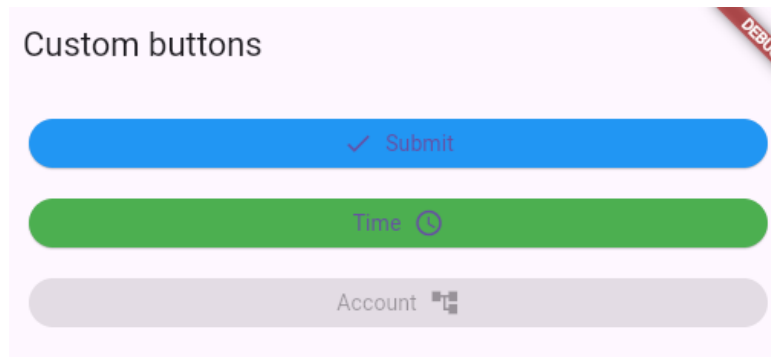
Extract the card into a stateless widget called **ProductCard** that takes the following parameters (named):

- The product (type: Product, **required**)



EX 3– The buttons

In this exercise, you need to build a Custom Button

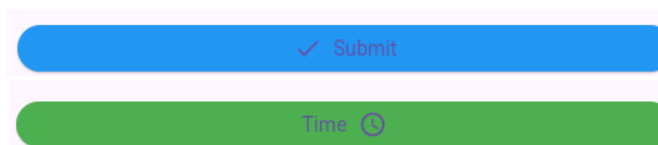


The button can have 3 **types**. Each type has a **specific color**:

Button type	Color
Primary	Blue
Secondary	green
Disabled	grey

Note: we recommend managing this requirement with a dedicated enum!

The button **icon** can be **either** before or after the button **label**:



Note: we recommend managing this requirement with a dedicated enum!

The custom button must be a stateless widget called **CustomButton** that takes the following parameters:

- The button **label** (String, *required*)
- The button **icon** (IconData, *required*)
- The icon position (*left or right, optional, by default left*)
- The button type (*primary, secondary, disabled, optional, by primary*)

EX 4 – The weather Forecast (BONUS)

FREE STYLE EXERCICE

In this exercise, the design, Flutter components and data structure are up to you.

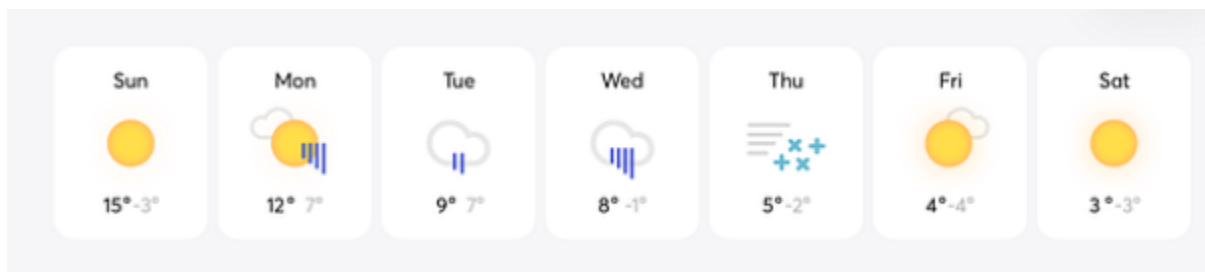
Important: you must use the widgets *we have already learnt*.

Create a stateless widget called WeatherForecast that takes the following parameters:

Required

- Weather condition (Sunny, Rainy, Cloudy, Snowy),)
- Temperature (min , max)
- Day Of Week (Mon, Tue...)

Depending on the weather condition, temperature, you need to compute the widget **colors** and **icons**.



Just ideas, you are free to design your card