W2-S9 PRACTICE

Event & States

At the end of his practice, you should be able to…

- ✓ Handle UI **events** (button click, text entered.)
- ✓ Use states hook to handle different variables (Boolean, string, number...) as **states**
- ✓ Using state hook to **bind textfield**
- ✓ Use state hook to **display color conditionally**

How to work?

- ✓ Download **the start code** from the Google classroom
- ✓ For each exercise you can either:
  - Run `npm install`
  - Or move an existing `node_modules` to the exercise folder *(fastest option!)*

How to submit?

- ✓ **Create a repository on GitHub** with the name of this practice:

  Ex: C2-S6 PRACTICE
- ✓ **Push your final code** on this GitHub repository (if you are lost, follow this tutorial)
- ✓ Finally submit on **Google classroom** your GitHub repository URL

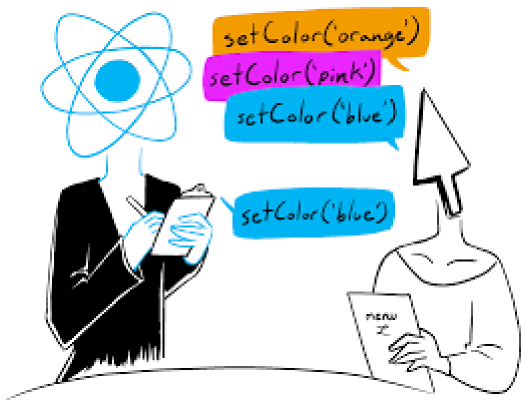
  Ex: `https://github.com/thebest/ C2-S6 PRACTICE.git`

Are you lost?

You can read the following documentation to be ready for this practice:

https://www.joshwcomeau.com/react/data-binding/

https://react.dev/learn/responding-to-events



EXERCISE 1

The goal of this app is to manage the weather, which can be either sunny or raining!
As you can see in the picture, you need to change both:

- The **main** element **background** color (*yellow or blue)*
- The **h1** element **title** *(sun time, rain time)*

Step 1: you need **a state hook** with a Boolean value: `isRaining`

- If this state is **true,** then the weather is **raining**
- If this state is **false,** then the weather is **not raining** *(it's sunny)*

Step 2: Then you need to handle the 2 buttons events

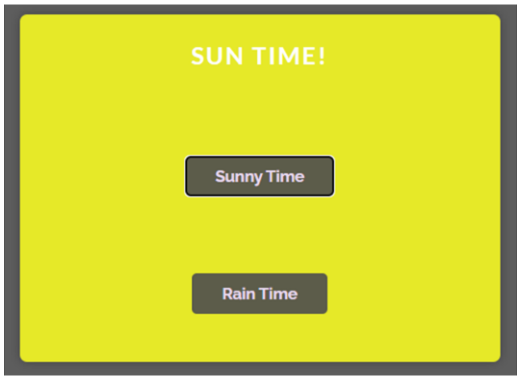
- Click on **Sunny time** -> change the state to **false** ***(****it's sunny)*
- Click on **Rain time** -> change the state to **true** *(it's raining)*

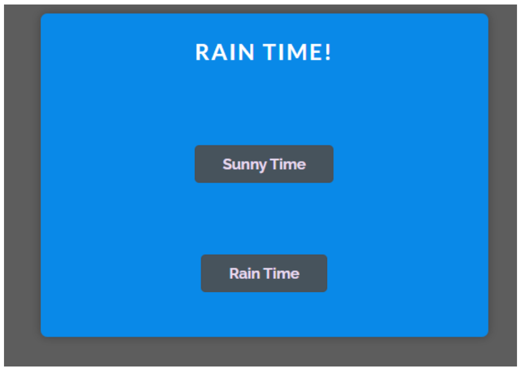
  Note: use the 2 functions already created to handle this.

Step 3: The last step if to update the <main> background and the <h1> text according to the `isRaining` state

- For the background color, you can use the (index.css) style classes: sunny and rainy
- You can to put the logic of your code in **functions** and bind your **JSX** properties with them

The finished app will look like this:





EXERCISE 2

The goal of this app is to **convert** some text entered in the textfield to **upper case.**

Step 1: Handle the textfield event when a key is pressed

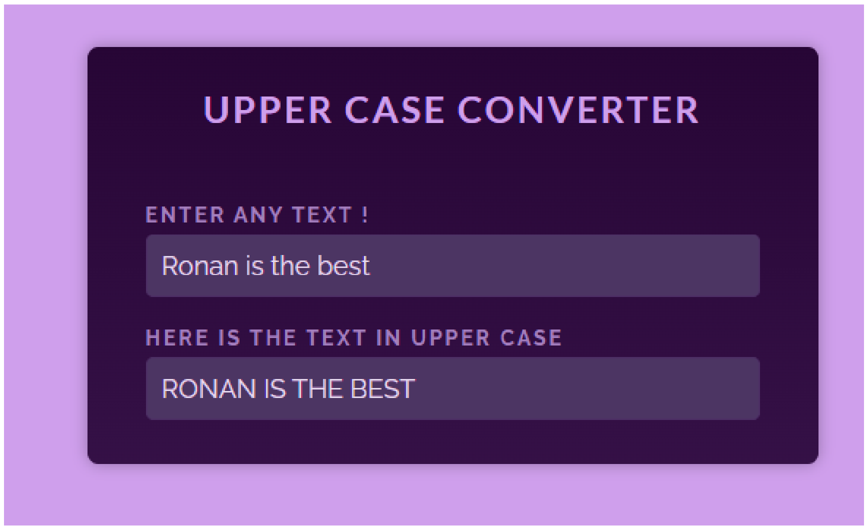
- How to **handle a key pressed** event?
- Create a function to handle this event and write something on the console to test it.

Step 2: You need to **use a state** to keep the value of the text entered

- What will be the type of this state? What will be the default value?
- How to update this state with the current textfield value? And when?

Step 3: Finally display the text in lower case in the second textfield

The finished app will look like this:



UPPER CASE CONVERTER

ENTER ANY TEXT !

Ronan is the best

HERE IS THE TEXT IN UPPER CASE

RONAN IS THE BEST

EXERCISE 3

The goal of this app is to **add 2 numbers** when the user clicks on the COMPUTE button.

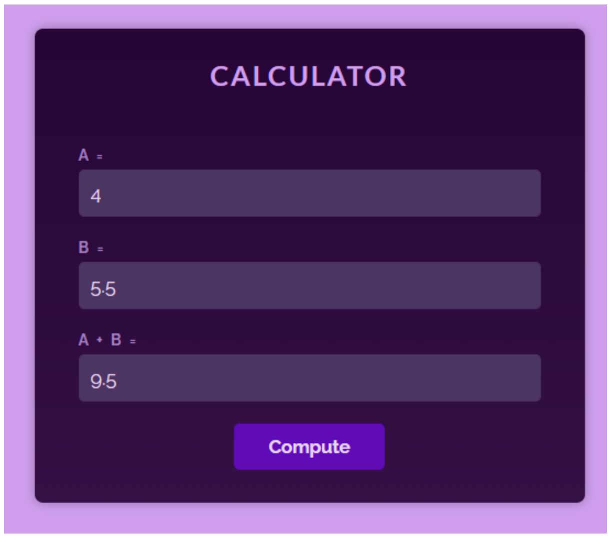
- If the textfields do not contain numbers, you need to display a red warning (see the pictures).

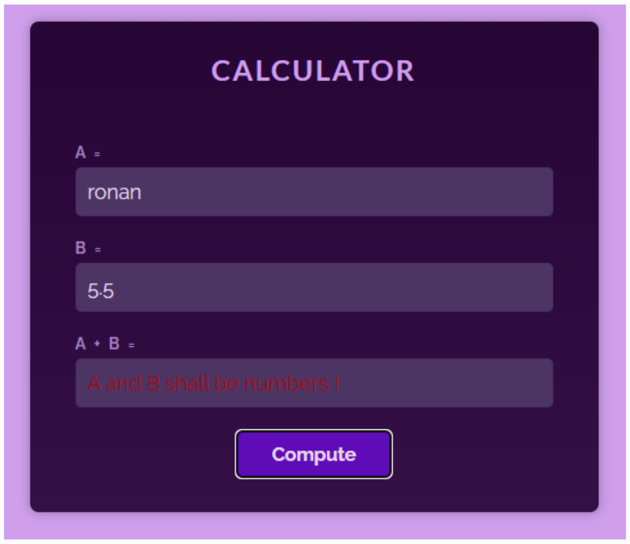
<u>Warning</u>: you should have succeeded exercise 2 before starting this one!

<u>You should think about the below question before starting your code:</u>

- How many **states** do you need?
- How to check if a text **is a number**?
- How to manage the **red color** conditionally? *(when the result is a warning)*
- When you click on compute button,
  - What will you check?
  - How many states will you update?

The finished app will look like this:





EXERCISE 4

The goal of this app is to manage a **carousel of images**

- Images shall change when clicking on left or right buttons
- The carrousel must loop:
    - Clicking right on last image, will go to the first image
    - Clicking left on first image, will go to the last image

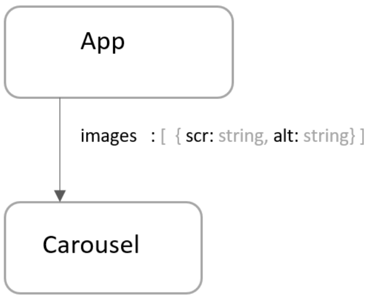
Images

All images are in `/asset` folder

We provide `imagesData.js` with the list already defined.

Components

The component Carousel gets as prop the list of images to display. Your work will be on this component!

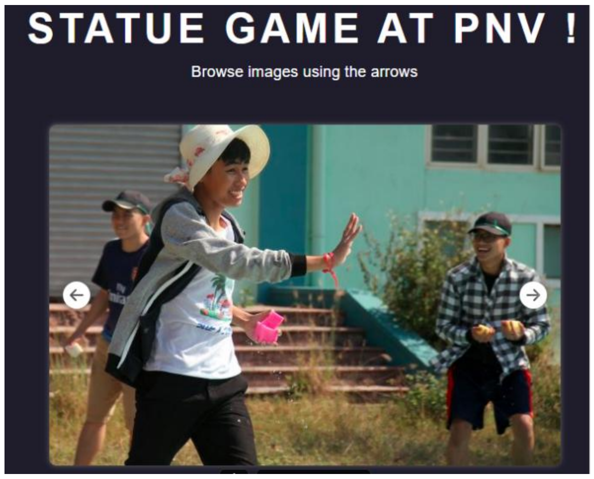


<u>You should think about the below question before starting your code:</u>

- How many **states** do you need?
- How to manage the current image?
- How to manage the **cases** when we are on the last image or first image?

<u>Explore React Icon</u>

- This project is using an extra library: `react-icons`: don't forget to run `npm i` to install this library
- If you want to know how to use icons: https://www.geeksforgeeks.org/reactjs-icons/


STATUE GAME AT PNV !
Browse images using the arrows