React States and Hooks - Practice Exercises

Answer the following practice questions, by creating a simple click counter app, with React States for questions 1 & 2 and React Hooks for questions 3 & 4, respectively.

Questions on React State

1. Create a class component called **"MyCounter.js"** that depends on another class component, **"CounterDisplayer.js"**. **"MyCounter.js"** keeps track of the click count value in the state, and this is where you initialize your state, which holds the click counts.

   Create a function named **"allClicksCounter()"** in **"MyCounter.js"** that will update or increase your state by one anytime the button in the **"MyCounter.js"** component is clicked.

   Hint:
   - The primary function of **"CounterDisplayer.js"** is to show how many times the button in the **"MyCounter.js"** component has been clicked. That is, you need to use props to pass the updated click data from **"MyCounter.js"** to **"CounterDisplayer.js"**.
   - Display the number of clicks exactly below the button.

2. Create another class component called **"EvenCounterDisplayer.js"**, on which the **"MyCounter.js"** component depends. This component keeps track of and displays **ONLY** the number of **even** click counts when the button inside **"MyCounter.js"** is clicked.

   Hint:
   - When the button is clicked for the first time, your **"EvenCounterDisplayer.js"** component should display **"Clicked 0 times"**. However, when the button is clicked for the second time, **"Clicked 2 times"** should appear. When the button is clicked for the third time, it should show **"Clicked 2 times"**. When clicked the fourth time, it should show **"Clicked 4 times"** and so forth.

- Display the “EvenCounterDisplayer.js” component directly below the "CounterDisplayer.js" component so that you can see both counters together.

Watch this demo clip for questions 1 and 2, which shows how your app should look.

Question on React Hooks

3. Create a functional component called "**IncreaseDecreaseCount.js**". There will be three buttons; a button to increase, a button to decrease, and a button to reset the click count values, and you will need to implement the useState() Hook to update the clicks.

   Hint:

   Right above the buttons, there will be a count displayer with an initial click value of 0. When the increase button is clicked, the value increases by one, when the decrease button is clicked, the value will decrease by one and when a button to reset is clicked, the value resets to the initial value, which is 0. Whenever any button is clicked, the change in the click value will be displayed on the count displayer.

   ** Watch this demo clip for question 3, which shows how your app should look. **

4. Create a functional component called "UseEffectForTitle.js". There will be a button and a count displayer, right below the button, with an initial click value of 0. When the button is clicked, the value should increase by one and the displayer and the document's title should display the changed count value. Basically, the change in the document’s title is a side-effect to the change in the count value due to the button’s click.

   When the component is rendered for the first time (or when it is mounted), display an alert text that says “*Component is mounted*”, on the screen.

Hint:

- Implement the useState() Hook to update the click values.
- Use the useEffect() Hook to write the logic/ function behind changing the document's title to the current value of the click count and also to show the alert message.
- The alert message should only display when the component is mounted, not every time the button is clicked.
- Click here to watch the demo for question 4, which shows how your app should look.

Steps to follow for each question

Steps to follow for Question 1

- In your **"Evangadi > phase 3 > Week-6"** folder create a folder called "**StatesAndHooksProject**";

- Create two class-based components called **"MyCounter.js"** and **"CounterDisplayer.js"** in the **"Components"** folder of your react project located in **"Evangadi > phase 3 > Week-6 > StatesAndHooksProject"**;

- Don't forget to do the following in your "**MyCounter.js**" component:
  - to call a constructor() function;
  - to call the super() method in your constructor;
  - to Initialize your state that will count the clicks to 0 in the constructor () function;
  - to Create "**allClicksCounter()**" function that will update/increase your state by one. You will need to use the **setState()** method in your **"allClicksCounter()"** function to update your state;
  - Render a button that calls the method "**allClicksCounter()**" when clicked;
  - to pass the updated count value to the **"CounterDisplayer.js"** component to display the number of clicks counted; and,
  - to import the **"CounterDisplayer.js"** in the **"MyCounter.js"** component.

Hint:

- The "**MyCounter.js**" component is going to be dependent on the "**CounterDisplayer.js**";
- The "**MyCounter.js**" component should be the one to be called on the **App.js** component, not the "**CounterDisplayer.js**"
- Do not forget to import "**MyCounter.js**" in "**App.js**"; and,
- You might need to use both states and props for this question.

Steps to follow for Question 2

- Go to the "**Components**" folder and create a component called "**EvenCounterDisplayer.js**";
- Don't forget to do the following in your "**MyCounter.js**" component:
  - to create the "**evenClicksCounter()**" function that will return the click counts that are only even. You will need to use the setState() method in your "**allClicksCounter()**" function to update your state to only contain the even counts.
  - to pass the updated counter value to the "**EvenCounterDisplayer.js**"' component to display the even number of clicks counted
  - to import the "**EvenCounterDisplayer.js**" in your "**MyCounter.js**"

Hint:

- The "**MyCounter.js**" component is going to be dependent on the "**EvenCounterDisplayer.js**"
- Do not forget to import "**MyCounter.js**" in "**App.js**";
- The "**MyCounter.js**" component should be the one to be called on the App.js component, not the "**EvenCounterDisplayer.js**"
- In **"MyCounter.js"**, there will be only one clickable button, and when clicked, both the **"allClicksCounter()"** and **"evenClicksCounter()"** functions will be executed, and the corresponding results will be displayed right below it.
- For this question, you would need to use both states and props.

Steps to follow for Question 3

- Go to the "**Components**" folder and create a functional component called "**IncreaseDecreaseCount.js**";
- Don't forget to do the following in your "**IncreaseDecreaseCounter.js**" component:
    - to import useState Hook from React;
    - to declare the state variable for the component to hold the increased/decreased click value into your component;
    - to declare an updater function that will update the current click count value;
    - to Pass the initial state value as an argument to the useState() Hook;
    - to write the logic/functions that will increase, decrease and reset your click count values;
    - to attach each click event with the respective event handler function;
    - to import "**IncreaseDecreaseCount.js**" in your App.js; and,
    - to call the "**IncreaseDecreaseCounter.js**" component in your App.js.

Steps to follow for Question 4

- Go to the "**Components**" folder and create a functional component called "**UseEffectForTitle.js**";
- Don't forget to do the following in your "**UseEffectForTitle.js**" component:
    - to import useState and UseEffect Hooks from React;
    - to declare the state variable for the component to hold the click value into your component;
    - to declare an updater function that will update the current click count value;
    - to pass the initial state value as an argument to the useState() Hook;
    - to write the logic/function that will increase the click count by 1 whenever there is a button click;
    - to attach the click event with the event handler function;
    - to write a function that will alert the ***"Component is mounted"*** message when the component is mounted (rendered for the first time). You will need to use useEffect() Hook to show this effect;

- to write a function that will change the document's title to the current count value whenever there is a button click. You will need to use useEffect() Hook to show this effect;
- to import "**UseEffectForTitle.js**" in your App.js; and,
- to call the "**UseEffectForTitle.js**" component in your App.js

Happy coding 😀