# React Hooks

## Exercise 1

use the useState hook to show a number that is incremented by one when a button is clicked.

#### Solution

function ClickCounter() {  
  
 const [counter, setCounter]=React.useState(0);  
  
 return (  
  
 <>  
  
 <button className="counter" onClick={(e)=>{  
  
 setCounter(()=>counter+1);  
  
 }}>Click Me  
  
 </button>  
  
 <div>{counter}</div></>  
  
 );  
  
}

## Exercise 2

use the useState hook to create a button that restricts clicking on it to 10 clicks. After 10 clicks it will display "too many clicks".

#### Solution

function MaxNum() {  
  
 const [counter, setCounter]=React.useState(0);  
  
 return (  
  
 <>  
  
 <button className="counter" id="myDiv" onClick={()=>{  
  
 setCounter(()=>counter+1);  
  
 counter<10?counterDiv.innerHTML=counter:counterDiv.innerHTML="Too Many Clicks";  
  
 }}>click  
  
 </button>  
  
 <div id="counterDiv"></div></>  
  
 );  
  
}

## Exercise 3

use the useState hook to simulate a light bulb. The light bulb should only have 2 states: "On" and "Off".

#### Solution

function LightBulb(p) {  
  
 const [state, setState] = React.useState(p.state)  
  
 return (  
  
 <button onClick={(e) => e.target.innerHTML == "On" ? setState("Off") : setState("On")}>{state}</button>  
  
 )  
  
}

## Exercise 4

use the useState hook to manage 2 states: "open" and "closed". If the state is "open" the height should be 1rem, else it should be 5 rems.  
  
The state should be initialized via props.

#### Solution

function ToggleDiv(p){  
  
 const [state, setState]=React.useState(p.state)  
  
 return (  
  
 <div onClick={()=>state=="open"?setState("closed"):setState("open")} style={state=="open"?{height:'5rem'}:{height:'1rem'}}>Hello!</div>  
  
 )  
  
}

## Exercise 5

use the useEffect hook to fetch a list from an API.

#### Hints

The useEffect hook should be used when you need to set up the component once, for example getting data from an API.

#### Solution

function ShowAPIList(){  
 const [items, setItems]=React.useState([]);  
 React.useEffect(()=>{axios.get("/api/list").then((response)=>{  
 setItems(response.data)  
 })   
 },[]  
 );  
 return <div>{items.map(i=><div>{i}</div>) }</div>  
}

## Exercise 6

use useEffect and useState to display the current date and time. The display should be refreshed every second.

#### Hints

The useEffect hook should be used when you need to set up the component once, for example getting data from an API or setting an interval.

#### Solution

function Clock(){  
 const [time, setTime]=React.useState(Date());  
 React.useEffect(()=>{  
 setInterval(()=>setTime(Date()), 1000)  
 })  
 return <div>{time}</div>  
}

## Exercise 7

use the useState hook to validate user input. If the input is too short, the components will show "too short" in red.

#### Solution

function ValidateInput(p){  
  
 const [contact, setContact]=React.useState(p.contact);  
  
 const [error, setError]=React.useState("");  
  
 return (   
  
 <>  
  
 <input onInput={(e) => {  
  
 if (e.target.value.length<3 && e.target.value.length!=0){  
  
 setError("too short")  
  
 } else {  
  
 setError("")  
  
 }  
  
 setContact(e.target.value)   
  
 }  
  
 } placeholder="Enter contact to show"/>  
  
 <Message message={`showing contact: ${contact}`} error={error} />  
  
 </>  
  
 )  
  
}

## Exercise 8

use the useState hook to manage 4 lightbulbs. A lightbulb can be either "Off" or "On".

#### Solution

function LightBulb(p) {  
 const [state, setState] = React.useState(p.state)  
 return (  
 <button onClick={(e) => e.target.innerHTML == "On" ? setState("Off") : setState("On")}>{state}</button>  
 )  
}  
  
function LightBulbs(p){  
  
 let bulbArray=[]  
  
 for (let i=0; i<4; i++){  
  
 bulbArray.push(<LightBulb />)  
  
 }   
  
 return(  
  
 <div>  
  
 {bulbArray}  
  
 </div>  
  
 )  
  
}

## Exercise 9

use the useState hook in a component that checks your vision. It shows a symbol, and if you guess it correctly it will display another symbol in a smaller font.

#### Solution

function EyeCheck(props){  
  
 const [num, setNum]=React.useState(12);  
  
 return (   
  
 <>  
  
 <input onInput={(e) => {  
  
 e.target.value == num?setNum(()=>num>5?num-1:setNum(num)):setNum(num)  
  
 e.target.parentElement.children[1].style.fontSize=`${num}px`  
  
 }  
  
 } />  
  
 <Message message={num} />  
  
 </>  
  
 )  
  
}

## Exercise 10

use the useState hook to manage an array of divs which can be either open or closed. The state of the divs is initialized when the page loads.

#### Solution

## Exercise 11

use the useState hook to create a vehicle gearbox. The 4 gear states are drive, neutral, parking and low. Switching between modes is done by buttons. At any given moment only one of the buttons can be pressed. You can't press more than one button at a time.