UNIVERSIDAD DE GUADALAJARA

CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍAS

Laboratorio de Inventores

Ejemplos sencillos: desde el botón hasta el contador.

Documento complementario para el libro y código React para Humanos: Aprende a Crear Apps sin ser Programador.

Ejemplos sencillos: desde el botón hasta el contador.

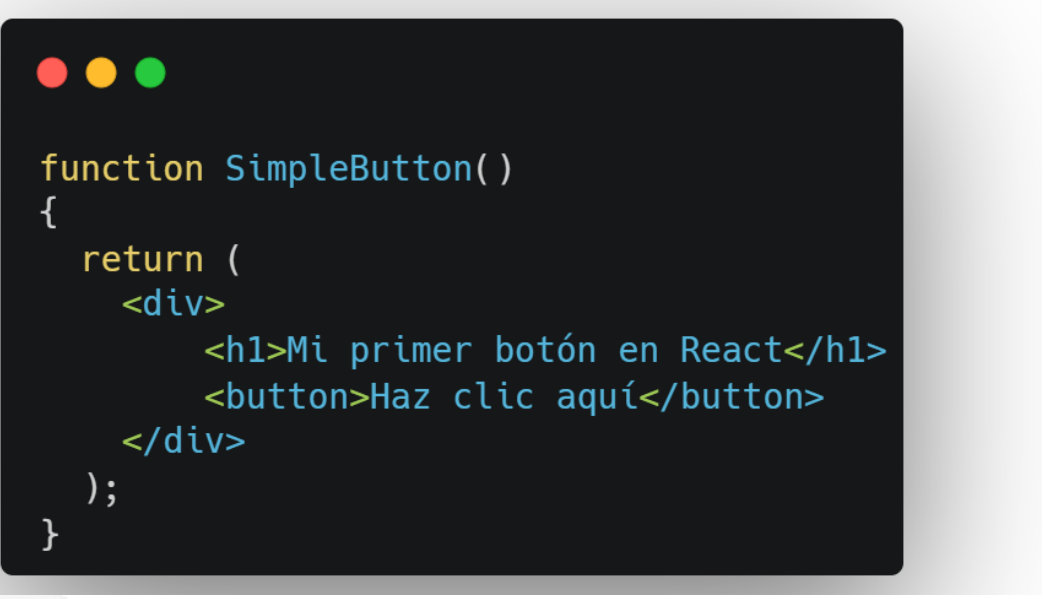
1. **Botón básico en React**

El primer paso es crear un botón simple que muestre texto en la pantalla.

Usando como base el siguiente ejemplo:

```
function SimpleButton()
{
      return <button>Click Me</button>;
}
```

Lo modificamos y tenemos:



```
function SimpleButton()
{
  return (
    <div>
        <h1>Mi primer botón en React</h1>
        <button>Haz clic aquí</button>
    </div>
  );
}
```

Visualizando lo siguiente:

Este componente muestra un botón con el texto “Haz clic aquí”.

Referencia para creación de ejemplo (https://wpwebinfotech.com/blog/how-to-create-react-button-component/).

2. Botón con evento de click.

Para hacer el botón interactivo, agrega un evento onClick el cual mostrará una alerta al hacer clic.

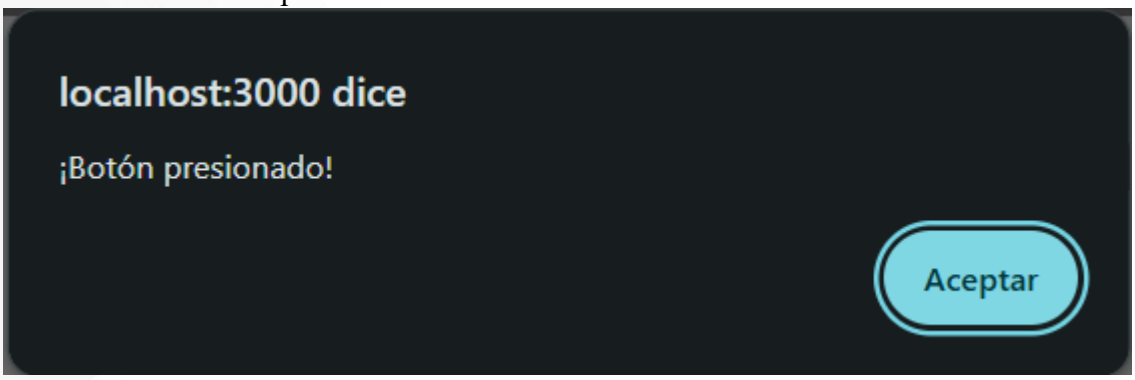
Usando como base el siguiente ejemplo:

```
function AlertButton()
{
  const handleClick = () =>
  {
    alert('Button clicked!');
  };
  return <button onClick={handleClick}>Click Me</button>;
}
```

Lo modificamos y tenemos:

```
function App() {
  const handleClick = () => {
    alert("¡Botón presionado!");
  };

  return (
    <div>
      <h1>Botón con evento</h1>
      <button onClick={handleClick}>Haz clic aquí</button>
    </div>
  );
}
```

Visualizando lo siguiente:
Muestra una alerta al presionar el botón.



Referencia para creación de ejemplo (https://wpwebinfotech.com/blog/how-to-create-react-button-component/).

3. Componente botón reutilizable.

Se puede crear un componente de botón que reciba texto y la función a ejecutar como propiedades (props).

Usando como base el siguiente ejemplo:
Creamos un archivo nuevo en la raíz del proyecto llamado “ReusableBotton” y se agrega la función de botón reutilizable.

Función para botón reutilizable:

```
import React from 'react';
function ReusableButton({ onClick, children })
{
  return (
    <button onClick={onClick}>
       {children}
    </button>
  );
}
export default ReusableButton;
```

Después la agregamos a la función principal importandolo y enseñándolo en la interfaz.

```
import React, { useState } from 'react';
function ConditionalButton()
{
  const [showButton, setShowButton] = useState(true);
  return (
    <div>
      {showButton && (
        <button onClick={() => alert('Button clicked!')}>
          Click Me
        </button>
      )}
      <button onClick={() => setShowButton(!showButton)}>
          Toggle Button
      </button>
    </div>
  );
}
export default ConditionalButton;
```

Lo modificamos y tenemos:

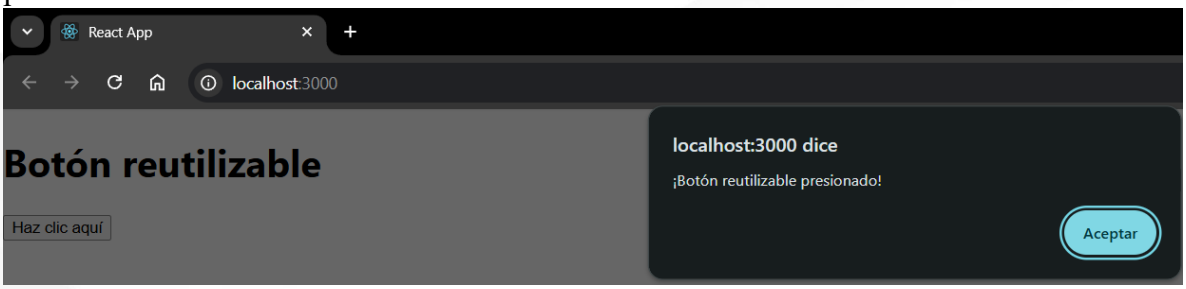
Función para botón reutilizable:

```
import React from 'react';

function Button({ label, onClick })
{
  return (
    <button onClick={onClick}>
      {label}
    </button>
  );
}

export default Button;
```

Luego a se importa a la función principal:

```
import React from 'react';
import Button from './Button';

function App() {
  const handleClick = () => {
    alert("¡Botón reutilizable presionado!");
  };

  return (
    <div>
      <h1>Botón reutilizable</h1>
      <Button label="Haz clic aquí" onClick={handleClick} />
    </div>
  );
}

export default App;
```

Visualizando lo siguiente:
Muestra que el botón declarado fuera del proyecto principal y llamado después fue presionado.



Referencia para creación de ejemplo (https://wpwebinfotech.com/blog/how-to-create-react-button-component/).

4. Contador simple usando useState.

Un ejemplo básico y clásico para principiantes es el contador de clicks usando el hook useState:

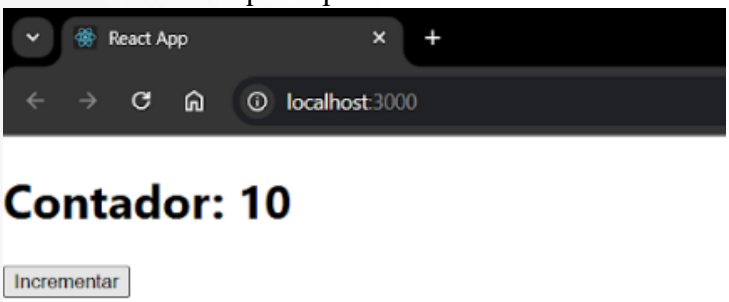
```
import React, {useState} from "react";
const [count, setCount] = useState(0);

const increment = (event) =>
{
  event.preventDefault();
  setCount((prevCount) => prevCount + 1);
};
```

Lo modificamos y tenemos:

Contador con un simple botón, el cual incrementa de 1 en uno con presionar el botón.

```
import React, { useState } from 'react';

function Counter()
{
  const [count, setCount] = useState(0);

  return (
  <div>
    <h1>Contador: {count}</h1>
    <button onClick={() => setCount(count + 1)}>
      Incrementar
    </button>
  </div>
  );
}

export default Counter;
```

Visualizando lo siguiente:
Muestra un botón que al presionarlo aumenta el contador.



Referencia para creación de ejemplo (https://dev.to/ijayyyy/a-simple-react-counter-with-input-value-using-react-hooks-usestate-and-usereducer-4aog).

5. Contador con incremento, decremento y reset.

Se puede ampliar el contador con más botones con el fin de añadir las funciones de incremento, decremento y reinicio del valor.
Usamos lo anterior y con base en la siguiente función de decremento y reinicio creamos nuestro contador:

Decremento:

```
const decrement = (event) =>
{
  event.preventDefault();
  setCount((prevCount) => prevCount - 1);
};
```

Reinicio:

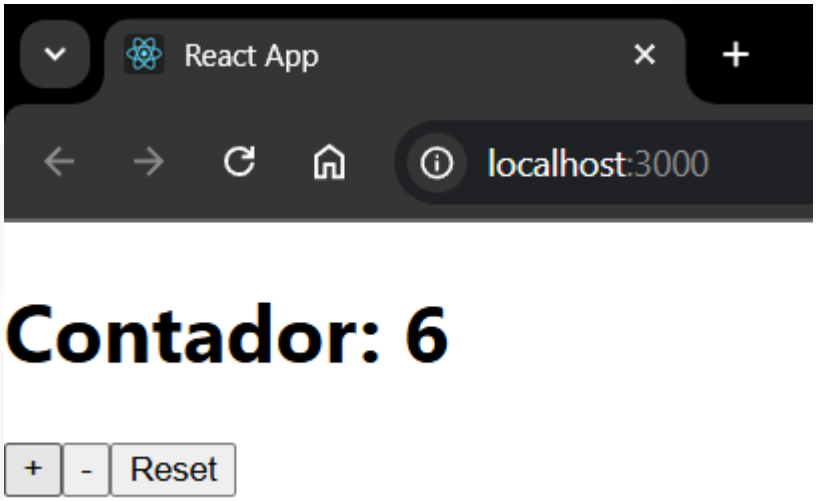
```
const reset = (event) =>
{
  event.preventDefault();
  setCount(0);
};
```

Lo modificamos y tenemos:

```jsx
import React, { useState } from 'react';

function Counter() {
  const [count, setCount] = useState(0);

  return (
    <div>
      <h1>Contador: {count}</h1>
      <button onClick={() => setCount(count + 1)}>+</button>
      <button onClick={() => setCount(count - 1)}>-</button>
      <button onClick={() => setCount(0)}>Reset</button>
    </div>
  );
}

export default Counter;
```

Visualizando lo siguiente:

Un contador con incremento, decremento y rinicio, con un contador que cambia de forma automática.



Referencia para creación de ejemplo (https://dev.to/ijayyyy/a-simple-react-counter-with-input-value-using-react-hooks-usestate-and-usereducer-4aog).

6. Contador reutilizable con props.

Finalmente se puede crear un contador el cual reciba el valor inicial y el incremento en props, haciendo aún más reutilizable.

Usando como base el siguiente ejemplo:

Creamos un archivo nuevo a raíz del proyecto llamado “CountBotton.jsx” y se le agrega el código para botón contador. Usando un prop llamado initialValue y otro increment a la hora de crear este componente de CountBotton.

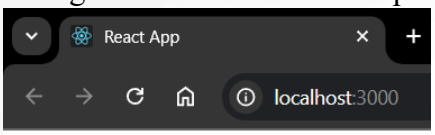
Función para botón contador:

```jsx
import React, { useState } from 'react';

function CountButton({ initialValue = 0, increment = 1 })
{
  const [count, setCount] = useState(initialValue);

  return (
    <div>
      <h1>Contador: {count}</h1>
      <button onClick={() => setCount(count + increment)}>
        +{increment}
      </button>
      <button onClick={() => setCount(initialValue)}>
        Reset
      </button>
    </div>
  );
}

export default CountButton;
```

Luego a se importa a la función principal:
Agregando los valores que se desean en los contadores, tales como valores iniciales y el incremento que se desea. Y llamando a nuestro componente ya creado anteriormente.

```
// App.js
import React from 'react';
import CountButton from './CountButton';

function App()
{
    return (
    <div>
      <h1>Varios contadores</h1>
      <CountButton initialValue={0} increment={1} />
      <CountButton initialValue={10} increment={5} />
    </div>
  );
}

export default App;
```

Visualizando lo siguiente:
Visualizamos dos contadores en los cuales se buscó tenerlos independientes con diferentes configuraciones en la misma aplicación.



Varios contadores

Contador: 7

+1 Reset

Contador: 25

+5 Reset

Referencia para creación de ejemplo (https://www.tutkit.com/es/tutoriales-de-texto/4945-crear-un-componente-de-contador-con-props-en-react).