

## Contenido

Parte A: onClick y el objeto event .....	2
Parte B: Cambiar la UI con estado + evento .....	2
Parte C: onChange (entrada controlada) .....	3
Parte D: onSubmit (formulario) .....	3
Miniretillo (pero esta vez sin ayuda).....	4
Preguntas .....	5

## Parte A: onClick y el objeto event

```
1 function App() {
2   const handleClick = (event) => {
3     console.log('Hola desde React!')
4     console.log(event)
5   }
6   return (
7     <button onClick={handleClick}>
8       Click me
9     </button>
10  )
11 }
12 export default App
```

Ilustración 1. onClick y el event. Elaboración propia

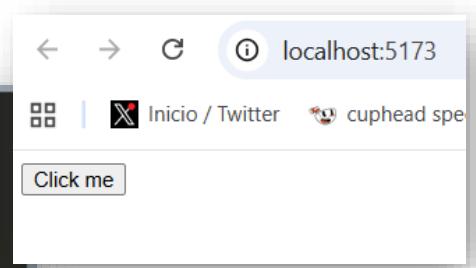


Ilustración 2. Visualización del resultado.  
Elaboración propia

```
Elementos Consola >>
top ▾ 🔍 Filtrar
Niveles predeterminados ▾ 1 problema: 1 | 2 ocultos
HOLA DESDE REACT! App.jsx:3
App.jsx:4
```

Ilustración 3. Visualización del resultado en la consola.  
Elaboración propia

## Parte B: Cambiar la UI con estado + evento

```
1 import { useState } from 'react'
2 function App() {
3   const [isParagraphVisible, setIsParagraphVisible] = useState(true)
4   const toggleStatus = () => {
5     setIsParagraphVisible(!isParagraphVisible)
6   }
7   return (
8     <>
9       <h1>Change UI based on click</h1>
10      {isParagraphVisible && (
11        <p>This paragraph will be shown/hidden on click</p>
12      )}
13      <button onClick={toggleStatus}>
14        {isParagraphVisible ? 'Hide' : 'Show'} Paragraph
15      </button>
16    </>
17  )
18 }
19 export default App
```

Ilustración 4. Cambiar la UI. Elaboración propia

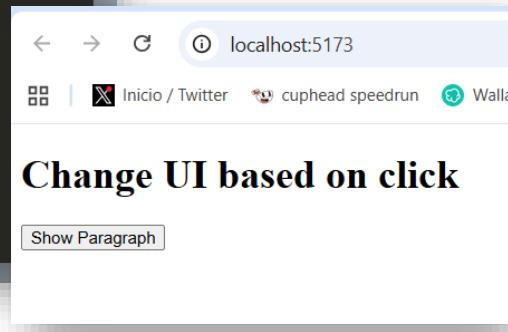


Ilustración 5. Visualización del resultado.  
Elaboración propia

## Parte C: onChange (entrada controlada)

The screenshot shows a code editor on the left and a browser window on the right. The code editor contains the following React component:

```
1 import { useState } from 'react'
2 function App() {
3   const [nickname, setNickname] = useState('')
4
5   return (
6     <>
7       <h1>Alias del jugador</h1>
8       <input
9         type="text"
10        placeholder="Escribe tu alias"
11        value={nickname}
12        onChange={(e) => setNickname(e.target.value)}
13      />
14      <p>Tu alias es: <strong>{nickname || '...'} </strong></p>
15    </>
16  )
17 }
18 export default App
```

The browser window displays the title "Alias del jugador". Below it is an input field containing "Antonio". Underneath the input field, the text "Tu alias es: Antonio" is displayed.

Ilustración 6. onChange. Elaboración propia

Ilustración 7. Visualización del resultado. Elaboración propia

## Parte D: onSubmit (formulario)

The screenshot shows a code editor on the left and a browser window on the right. The code editor contains the following React component:

```
1 import { useState } from 'react'
2 function App() {
3   const [email, setEmail] = useState('')
4   const [msg, setMsg] = useState('')
5   const handleSubmit = (e) => {
6     e.preventDefault()
7     setMsg(`Inscripción enviada para: ${email}`)
8   }
9   return (
10     <>
11       <h1>Registro</h1>
12       <form onSubmit={handleSubmit}>
13         <input
14           type="email"
15           placeholder="tu@email.com"
16           value={email}
17           onChange={(e) => setEmail(e.target.value)}
18           required
19         />
20         <button type="submit">Enviar</button>
21       </form>
22       {msg && <p>{msg}</p>}
23     </>
24   )
25 }
26 export default App
```

The browser window displays the title "Registro". Below it is an input field containing "daw@daw.es" and a button labeled "Enviar". Underneath the input field, the text "Inscripción enviada para: daw@daw.es" is displayed.

Ilustración 8. onSubmit. Elaboración propia

Ilustración 9. Visualización del resultado. Elaboración propia

## Miniretillo (pero esta vez sin ayuda)

The terminal window shows the code for `App.jsx`:

```
1 import Panel from './Panel'
2 import Contador from './Contador'
3
4 function App() {
5   return (
6     <div style={{ padding: '20px' }}>
7       <h1>Mi Panel de Control</h1>
8
9      /* Llamamos al componente Panel (Input + Mostrar/Ocultar) */
10     <Panel />
11
12      /* Línea divisoria para separar las secciones */
13      <hr />
14
15      /* Llamamos al componente Contador (+1 / -1) */
16      <Contador />
17    </div>
18  )
19}
20
21 export default App
```

The browser window shows the application running at `localhost:5173`. The title is "Mi Panel de Control". It contains a text input field with the value "cool", a button labeled "Ocultar sección", and a note "Esta es la sección que se oculta.". Below this is a section titled "Contador: 0" with two buttons "+1" and "-1".

Ilustración 10. `App.jsx`. Elaboración propia

The terminal window shows the code for `Contador.jsx`:

```
1 import { useState } from 'react'
2
3 function Contador() {
4   // Definimos el estado 'count' empezando en 0
5   const [count, setCount] = useState(0)
6
7   // Función para aumentar el contador
8   const sumar = () => setCount(count + 1)
9
10  // Función para restar con la condición de no bajar de 0
11  const restar = () => {
12    if (count > 0) {
13      setCount(count - 1)
14    }
15  }
16
17  return (
18    <div style={{ marginTop: '20px' }}>
19      /* Mostramos el valor del estado en la interfaz */
20      <h3>Contador: {count}</h3>
21
22      /* onClick para ejecutar las funciones de arriba */
23      <button onClick={sumar}>+1</button>
24      <button onClick={restar}>-1</button>
25    </div>
26  )
27}
28
29 export default Contador
```

Ilustración 12. `Contador.jsx`. Elaboración propia

Ilustración 11. Visualización del resultado. Elaboración propia

```

1 import { useState } from 'react'
2
3 function Panel() {
4     // Estado para guardar el texto del input
5     const [modo, setModo] = useState('chill')
6
7     // Estado booleano para mostrar u ocultar
8     const [mostrar, setMostrar] = useState(true)
9
10    return (
11        <div>
12            {/* Input controlado: onChange actualiza el estado 'modo' al escribir */}
13            <input
14                type="text"
15                value={modo}
16                onChange={(e) => setModo(e.target.value)}
17            />
18            {/* Mostramos el texto del estado en tiempo real */}
19            <p>Modo: <strong>{modo}</strong></p>
20
21            {/* Botón con toggle: cambia el estado 'mostrar' al contrario de lo que tenga */}
22            <button onClick={() => setMostrar(!mostrar)}>
23                {/* El texto del botón cambia según el estado (Punto opcional) */}
24                {mostrar ? 'Ocultar' : 'Mostrar'} sección
25            </button>
26
27            {/* Renderizado condicional: si 'mostrar' es true, enseña el párrafo */}
28            {mostrar && <p>Esta es la sección que se oculta.</p>}
29        </div>
30    )
31 }
32
33 export default Panel

```

## Panel.jsx

Ilustración 13. Panel.jsx. Elaboración propia

## Preguntas

### 1. ¿Qué diferencia hay entre onClick y onSubmit?

onClick se activa al pulsar cualquier elemento (como un botón), mientras que onSubmit se activa específicamente al enviar un formulario (al pulsar "Enter" o un botón de tipo submit).

### 2. ¿Por qué usamos e.preventDefault() en un formulario?

Se usa para evitar que la página se recargue automáticamente al enviar el formulario, permitiendo que React maneje los datos sin perder el estado actual.

### 3. ¿Qué es una "entrada controlada" y por qué usamos value + onChange?

Es un input cuyo valor lo manda React a través del estado. Usamos value para mostrar el estado actual y onChange para actualizar dicho estado cada vez que el usuario escribe.

### 4. En tu mini-reto, que estado(s) manejas y que evento(s) los actualizan?

Manejo tres estados: count (número), modo (texto) y mostrar (booleano). Los actualizo con eventos onClick en los botones y onChange en el input de texto.