

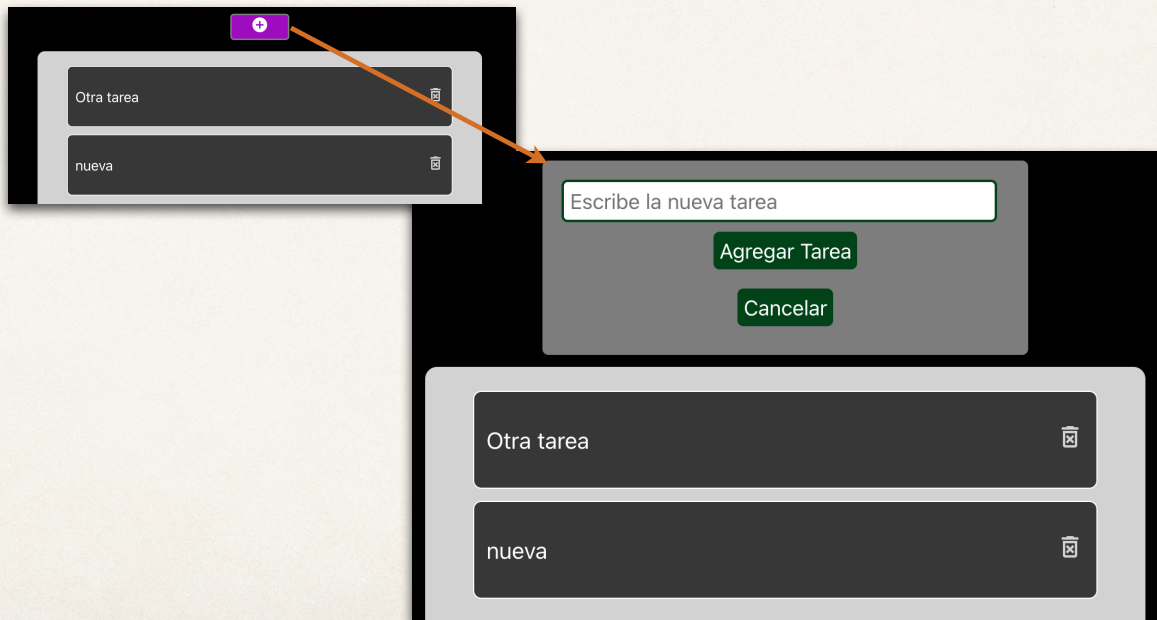
Formas en React

Conceptos básicos. Hooks / Estado

App

- ✧ Vamos a crear una app que permita manejar una lista de tareas.
 - ➡ Los datos que se almacenan son:
 - ✧ Descripción de la tarea.
 - ✧ Indicador del estado: en proceso y completada.

Ejemplos



M. en C. Roberto Martínez Román - rmroman@tec.mx

Proyecto

- ❖ Crea el proyecto administra-tareas.
 - ➔ **`npx create-react-app administra-tareas`**
- ❖ Abre el proyecto en VSC y borra el contenido inicial.

M. en C. Roberto Martínez Román - rmroman@tec.mx

Instalar paquetes de iconos

- ❖ Instala el paquete de iconos (<https://react-icons.github.io/react-icons/>) desde la terminal de VSC.

➔ **npm install react-icons**

- ❖ Para incluir un icono:

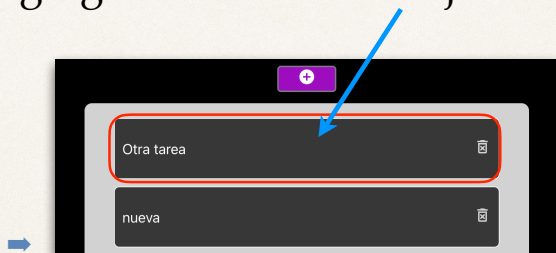
```
import { FaBeer } from 'react-icons/fa';  
  
class Question extends React.Component {  
  render() {  
    return <h3> Lets go for a <FaBeer />? </h3>  
  }  
}
```

```
<h2>Lista de Tareas: <GrTask /> </h2>  
<h2> <FcAlarmClock /> </h2>  
<h1> <FcAlarmClock /> </h1>  
<h4> <FcAlarmClock /> </h4>
```



Componentes

- ❖ Crea la carpeta *components* dentro de *src*.
- ❖ Agrega el archivo *Tarea.js*



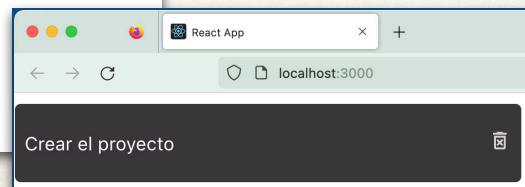
Una Tarea

```
import "../styles/Tarea.css";
import { MdOutlineDeleteForever } from "react-icons/md";

const Tarea = (props) => {
  return (
    <div className="tarea-contenedor">
      <div className="tarea-texto">
        {props.texto}
      </div>
      <div className="tarea-icone">
        <MdOutlineDeleteForever />
      </div>
    </div>
  );
};

export default Tarea;
```

Agrega temporalmente una Tarea en App.js y verifica que aparece correctamente



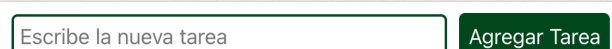
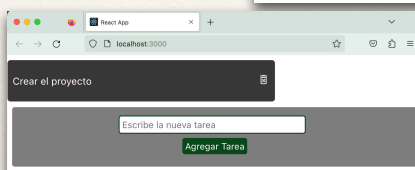
M. en C. Roberto Martínez Román - rmroman@tec.mx

Forma para capturar una nueva Tarea

❖ Agrega el componente **CapturaTarea** al proyecto.

```
const CapturaTarea = (props) => {
  return (
    <form className="tarea-forma">
      <input
        className="tarea-input"
        type="text"
        placeholder="Escribe la nueva tarea"
        name="texto"
      />
      <button className="tarea-boton">Agregar tarea</button>
    </form>
  );
};
```

Agrega temporalmente CapturaTarea en App.js



M. en C. Roberto Martínez Román - rmroman@tec.mx

El componente ListaTareas

- ❖ Vamos a encapsular en un solo componente la forma y las tareas.
- ❖ Crea el componente **ListaTareas** que contenga **CapturaTarea**.
- ❖ Este componente usa el hook de estado para guardar un arreglo con todas las tareas agregadas.
- ❖ Con la función *map* del arreglo, genera componentes **Tarea** para conformar la lista.

M. en C. Roberto Martínez Román - rmroman@tec.mx

ListaTareas

```
const ListaTareas = (props) => {  
  const [arrTareas, setArrTareas] = useState([]);  
  
  return (  
    <Fragment>  
      <CapturaTarea />  
      <div className="lista-tareas">  
        {arrTareas.map((tarea) => {  
          return <Tarea texto={tarea.texto} />  
        })}  
      </div>  
    </Fragment>  
  );  
};  
  
export default ListaTareas;
```

M. en C. Roberto Martínez Román - rmroman@tec.mx

Contenido del input

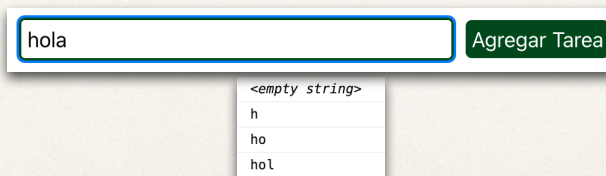
- ❖ Para leer el contenido del input, podemos hacer dos cosas:
 - ➔ Capturar los cambios y guardar el último valor en el estado del componente.
 - ➔ Definir una referencia al input, para después acceder a su valor en cualquier parte de la función.

M. en C. Roberto Martínez Román - rmroman@tec.mx

Valor del input en el estado

```
// Contenido del input
const [descripcionTarea, setDescripcionTarea] = useState("");
// Manejador del evento cuando el input cambia de valor
const cambioEntradaHandler = (event) => {
  setDescripcionTarea(event.target.value);
  console.log(descripcionTarea);
};
```

```
<input
  className="tarea-input"
  type="text"
  placeholder="Escribe la nueva tarea"
  name="texto"
  onChange={cambioEntradaHandler}
/>
```



hola

Agregar Tarea

<empty string>
h
ho
hol

M. en C. Roberto Martínez Román - rmroman@tec.mx

Referencias

- ❖ React posee un hook especial para acceder directamente a nodos del DOM.
- ❖ Una *referencia* es un objeto que almacena valores durante el tiempo de vida de un componente.

```
const refInput = useRef();  
  
return (  
  <form className="tarea-forma" onSubmit={agregarTareaHandler}>  
    <input  
      ref={refInput}  
      className="tarea-input"  
      type="text"  
    />  
  </form>  
);
```

```
const agregarTareaHandler = (event) => {  
  console.log(refInput);  
};
```

M. en C. Roberto Martínez Román - rmroman@tec.mx

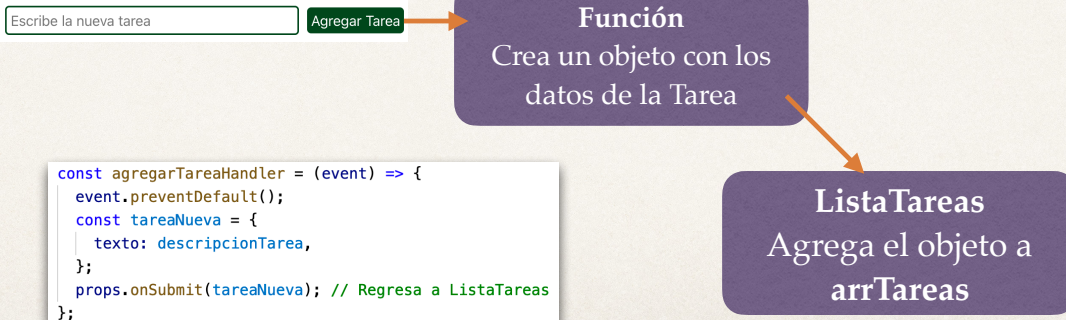
Click

- ❖ ¿Qué sucede cuando das click al botón?

M. en C. Roberto Martínez Román - rmroman@tec.mx

Seguimos en CapturaTarea

- ❖ Capturamos el click del botón *Agregar Tarea*.
- ❖ Llamamos a una función que prepara un objeto *con los datos de la tarea* y lo entrega a **ListaTareas** para que se agregue al arreglo.



M. en C. Roberto Martínez Román - rmroman@tec.mx

Actualiza arrTareas

The diagram shows the code for updating the task list. A code block contains the following code:

```
const ListaTareas = (props) => {  
  // Arreglo de tareas  
  const [arrTareas, setArrTareas] = useState([]);  
  
  const agregarTarea = (tarea) => {  
    const arrTareasNuevo = [tarea, ...arrTareas];  
    setArrTareas(arrTareasNuevo);  
  };  
  
  return (  
    <Fragment>  
      <CapturaTarea onSubmit={agregarTarea}/>  
      <div className="lista-tareas-contenedor">  
        Entender Hook de estado  
        Aprender Hooks  
        Aprender React  
      </div>  
    </Fragment>  
  );  
};
```

An arrow points from the `agregarTarea` function in the code to the `onSubmit={agregarTarea}` prop in the `<CapturaTarea>` component. To the right, a visual representation of the task list is shown, featuring a form with the input 'Entender Hook de estado' and the 'Agregar Tarea' button, and a list of tasks: 'Entender Hook de estado', 'Aprender Hooks', and 'Aprender React', each with a delete icon.

M. en C. Roberto Martínez Román - rmroman@tec.mx

Estado de la Tarea

❖ Los datos completos de la Tarea son:

```
{  
  id: uuidv4(),  
  texto: descripcionTarea,  
  completada: false  
}
```

Instalar el paquete uuid
<https://www.npmjs.com/package/uuid>

```
import { v4 as uuidv4 } from 'uuid';  
uuidv4(); // => '9b1deb4d-3b7d-4bad-9bdd-2b0d7b3dcb6d'
```

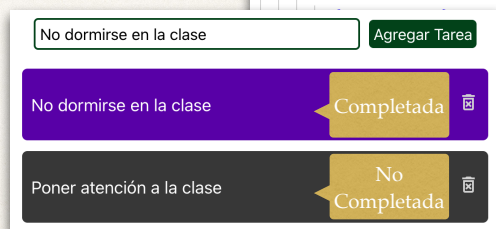
agregarTareaHandler

```
import { v4 as uuidv4 } from "uuid";  
  
const CapturaTarea = (props) => {  
  // Contenido del input  
  const [descripcionTarea, setDescripcionTarea] = useState("");  
  // Manejador del evento cuando el input cambia de valor  
  const cambioEntradaHandler = (event) => { ...  
};  
  
const agregarTareaHandler = (event) => {  
  event.preventDefault();  
  const tareaNueva = {  
    id: uuidv4(),  
    texto: descripcionTarea,  
    completada: false  
  };  
  props.onSubmit(tareaNueva); // Regresa a ListaTareas  
};
```


Tarea completada

- ❖ ¿Cómo formatear el renglón cuando está completa la tarea?

```
const Tarea = (props) => {  
  const estiloTarea = "tarea-contenedor " + (props.completada ? "completada" : "");  
  
  return (  
    <div className={estiloTarea}>  
      <div className="tarea-texto">
```



```
-contenedor-icone">  
  <div>  
    <div>  
      <div>  
        <div>  
          <div>  
            <div>  
              <div>  
                <div>  
                  <div>  
                    <div>  
                      <div>  
                        <div>  
                          <div>  
                        </div>  
                      </div>  
                    </div>  
                  </div>  
                </div>  
              </div>  
            </div>  
          </div>  
        </div>  
      </div>  
    </div>  
  </div>  
</div>
```

Funciones para completar/borrar

- ❖ Completar tarea cuando haces click sobre el texto.

```
const completarTarea = (id) => {  
  const arrTareasNuevo = arrTareas.map((tarea) => {  
    if (tarea.id === id) {  
      tarea.completada = !tarea.completada;  
    }  
    return tarea;  
  });  
  setArrTareas(arrTareasNuevo);  
};
```

- ❖ Eliminar tarea cuando haces click sobre el icono .

```
const eliminarTarea = (id) => {  
  const arrTareasNuevo = arrTareas.filter((tarea) => tarea.id !== id);  
  setArrTareas(arrTareasNuevo);  
};
```


ListaTareas y Tarea

```
return (  
  <Fragment>  
    <CapturaTarea onSubmit={agregarTarea} />  
    <div className="lista-tareas-contenedor">  
      {arrTareas.map((tarea) => (  
        <Tarea  
          texto={tarea.texto}  
          completada={tarea.completada}  
          key={tarea.id}  
          id={tarea.id}  
          completarTarea={completarTarea} /*Declara la tarea completada*/  
          eliminarTarea={eliminarTarea} /*Elimina esta tarea*/  
        />  
      ))}  
    </div>  
  </Fragment>  
)
```

```
const Tarea = (props) => {  
  const estiloTarea =  
    "tarea-contenedor " + (props.completada ? "completada" : "");  
  
  return (  
    <div className={estiloTarea}>  
      <div  
        className="tarea-texto"  
        onClick={() => props.completarTarea(props.id)} >  
        {props.texto}  
      </div>  
      <div  
        className="tarea-contenedor-icone"  
        onClick={() => props.eliminarTarea(props.id)} >  
        <MdOutlineDeleteForever />  
      </div>  
    </div>  
  );  
};
```

M. en C. Roberto Martínez Román - rmroman@tec.mx

Finalmente

❖ ¿Qué sucede si no hay tareas?

➡ `{arrTareas.length === 0 && <h1>No hay tareas</h1>}`

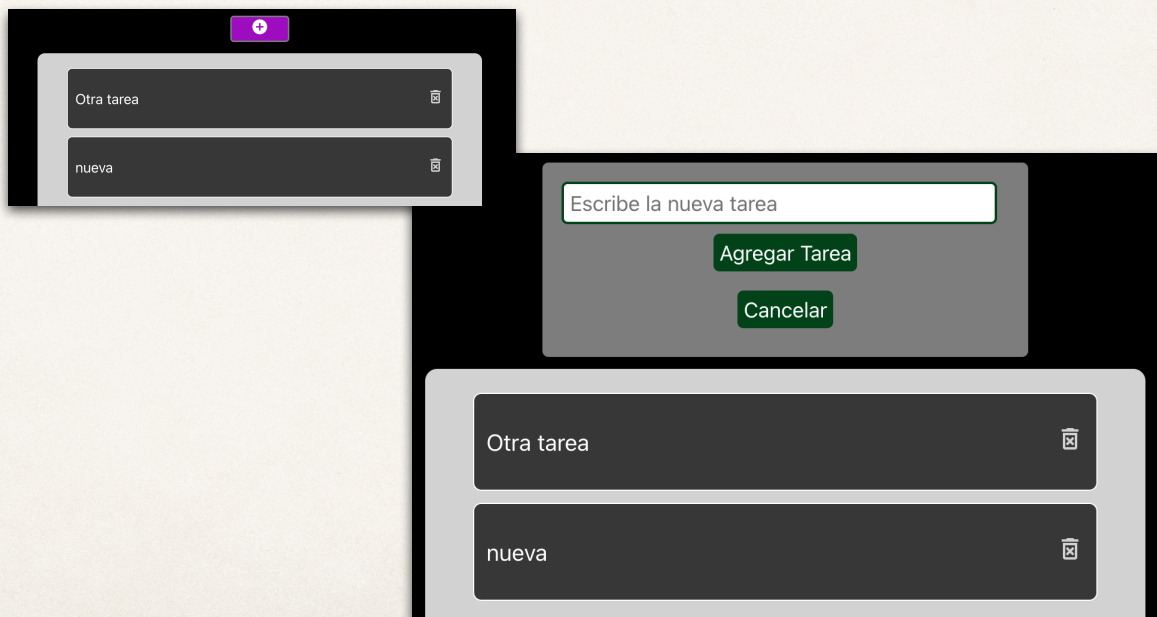
M. en C. Roberto Martínez Román - rmroman@tec.mx

RETO

- ❖ Esconder la forma de captura y en su lugar poner un botón de agregar.
- ❖ Cuando se hace click al botón de agregar, éste se esconde y aparece la forma de captura.
- ❖ Cuando se agrega la tarea, la forma se esconde y reaparece el botón de agregar.
- ❖ Agregar un botón para cancelar la captura.
- ❖ Evitar que se agregue una tarea sin texto.

M. en C. Roberto Martínez Román - rmroman@tec.mx

Ejemplos



M. en C. Roberto Martínez Román - rmroman@tec.mx