Herramienta para validación de parámetros en javascript propTypes

**Yarn add prop-types**

Configuración de visual studio code instalación de extensiones y plugisns

* Vscode-icon
* Reactjs code snnipets
* ESlint (encargada de avisar sobre los problemas de código en javascript)

Plugin para el debug de react en Chrome

React developer tools

**Librería para los estilos**

**Material-ui.com**

**Comandos**

Comando para configurar el proyecto con las mejores prácticas de babel, yarn entre otras

**npx create-react-app nombreproyecto**

luego ubicarnos en la carpeta del proyecto y ejecutar

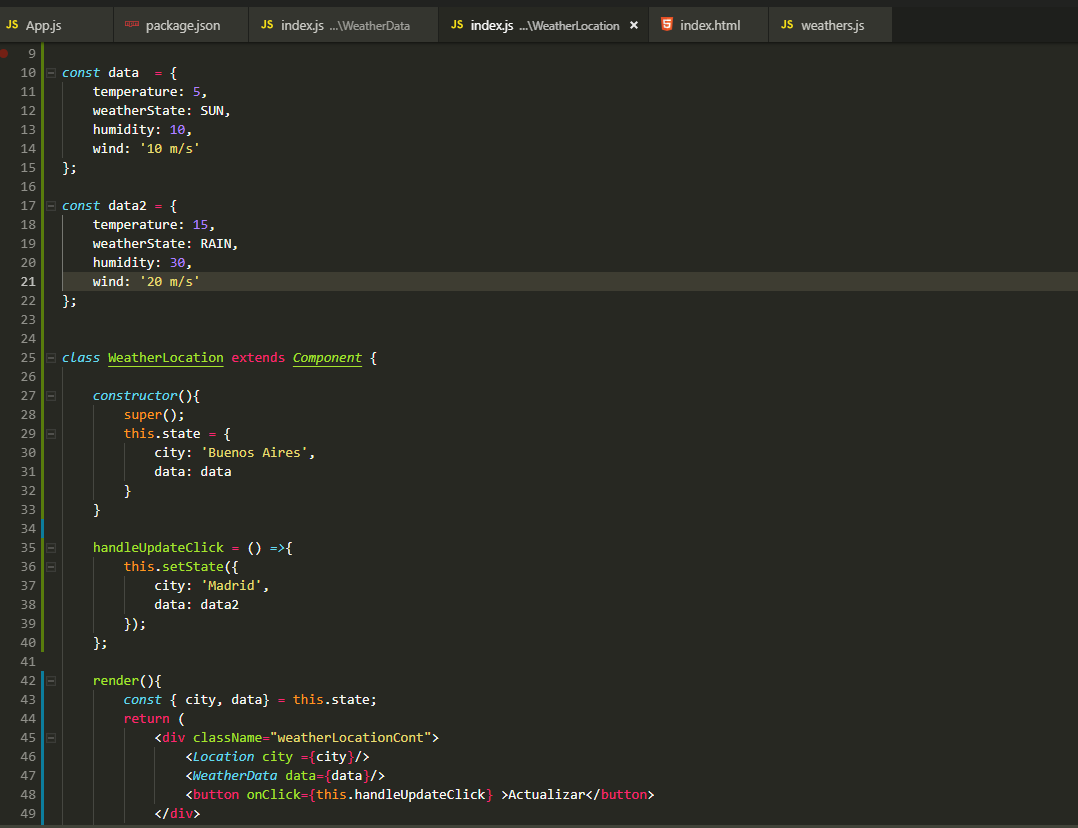
verificar la ubicación ya que para inicial el proyecto debemos estar dentro de la carpeta del proyecto

**yarn start**

**DESTRUCTURING**

Es una técnica que nos permite asignar valores a variables desde un objeto más complejo, o desde arrays con varios elementos, mediante la técnica “object pattern”

**CONCEPTOS REACTJS**

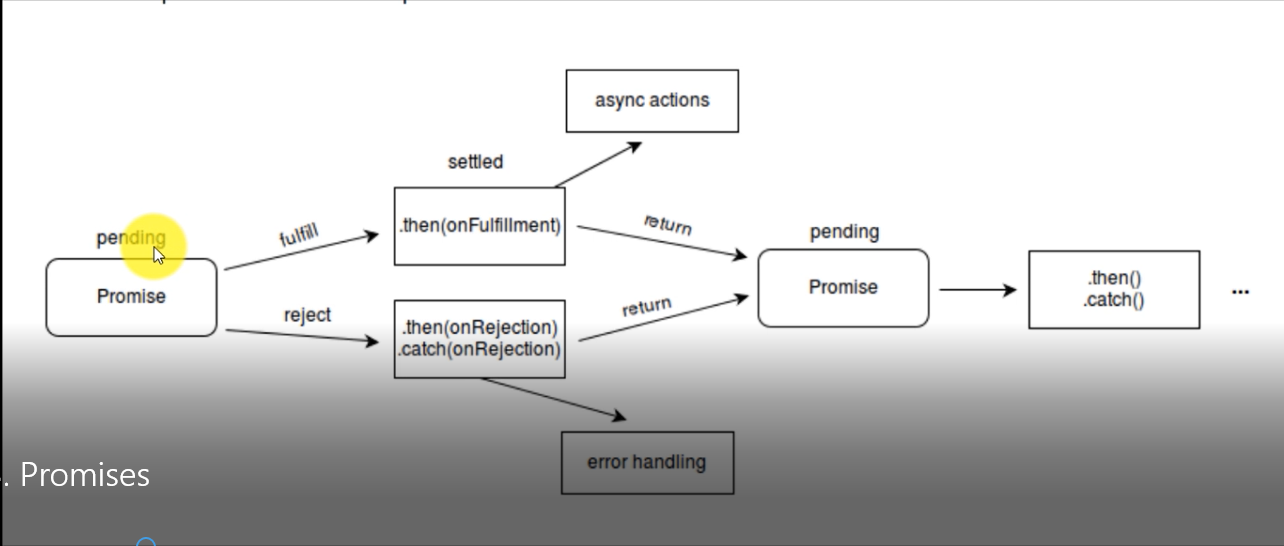


Esta imagen refleja uno de los conceptos importantes de react cuando tenemos un component class y queremos actualizar el state de los parámetros, muy importante tener en cuenta que siempres debemos tenemos un constructor y dentro del mismo es indispensable colocar el super, otro importante concepto es el manejo del this para todo lo relacionado con el componente que estamos modificando actualmente.

**PROMISE**

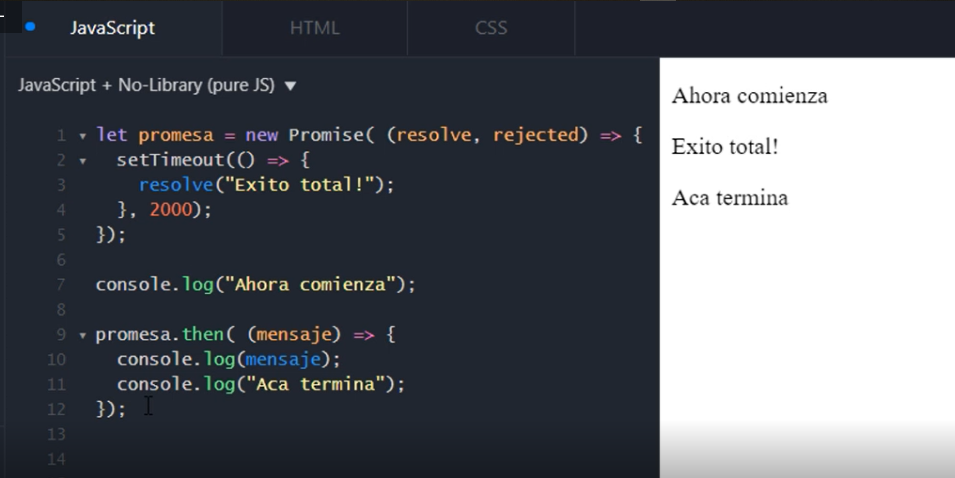
Estados de una promesa pendiente, cumplida, rechazada.

Flujo que puede seguir una promise

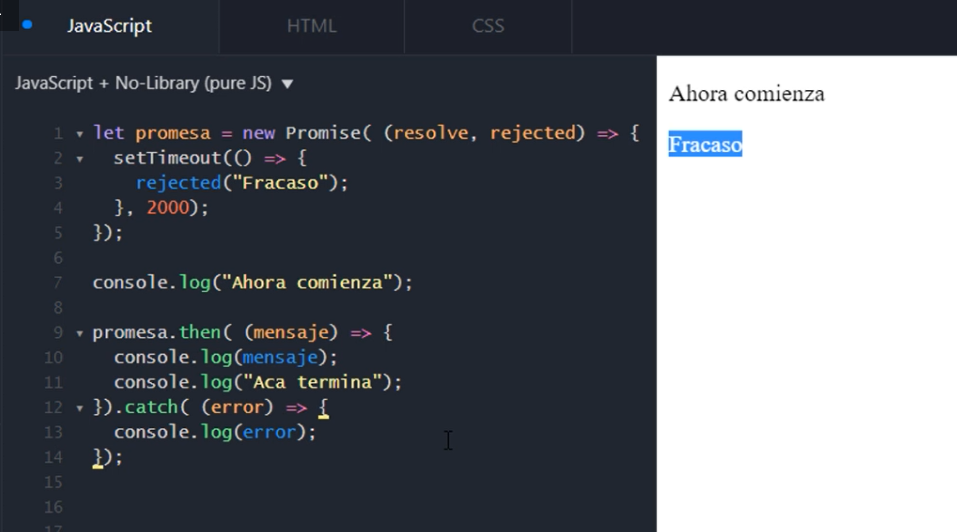


Definir y utilizar promises

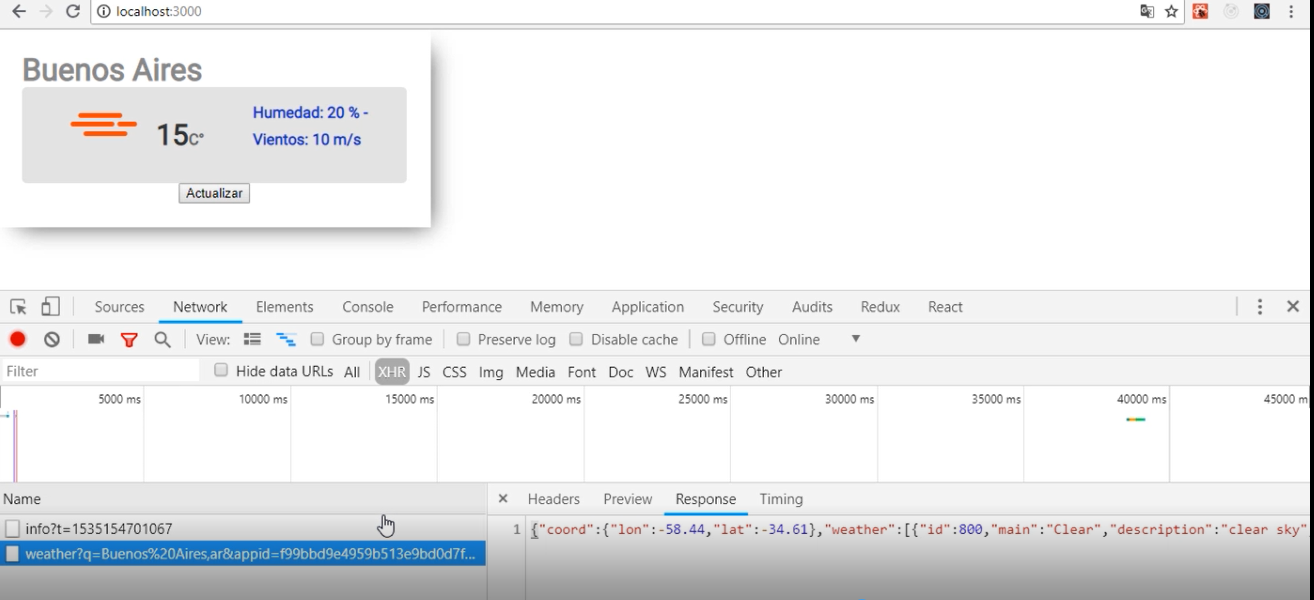
Promise aceptada



Promise rechazada

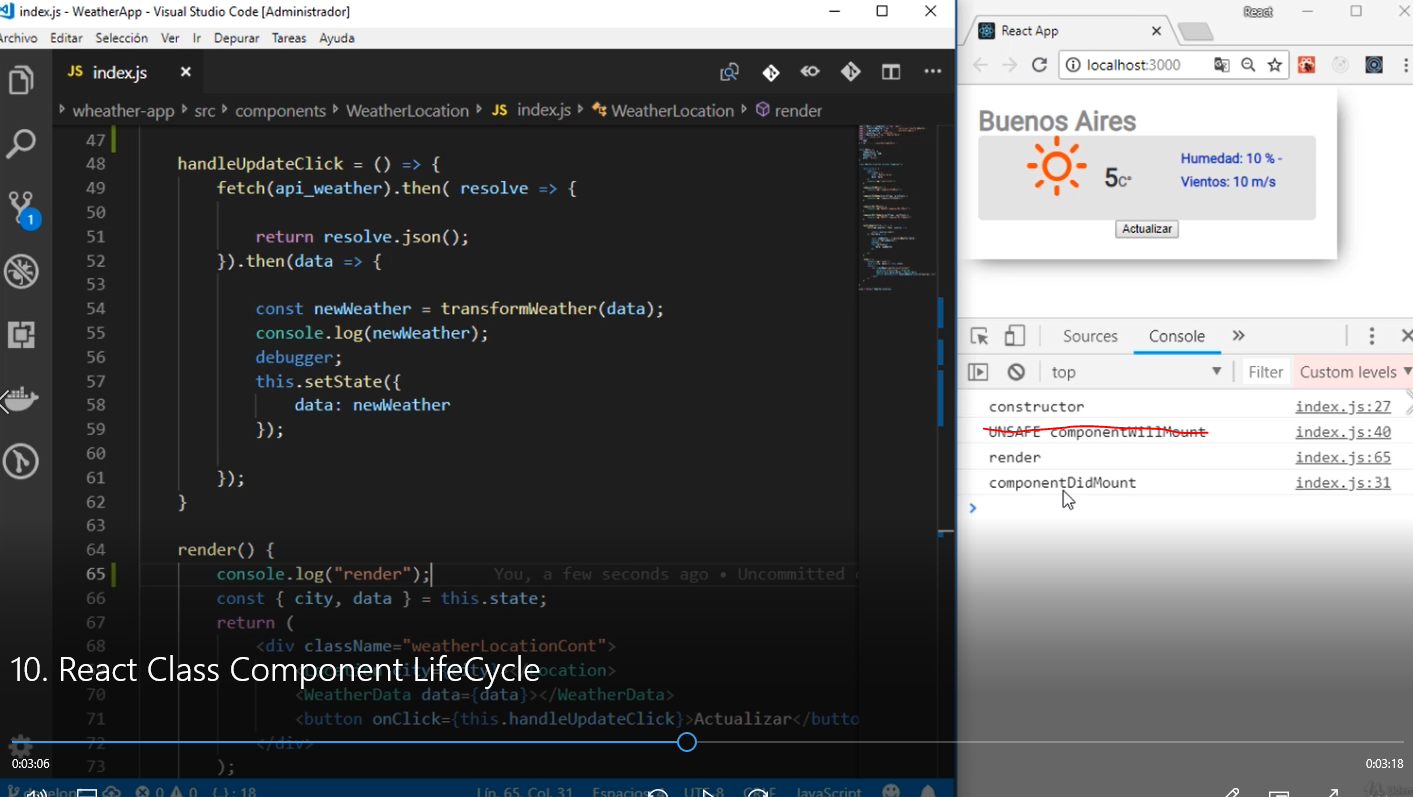


**HERRAMIENTAS UTILES**

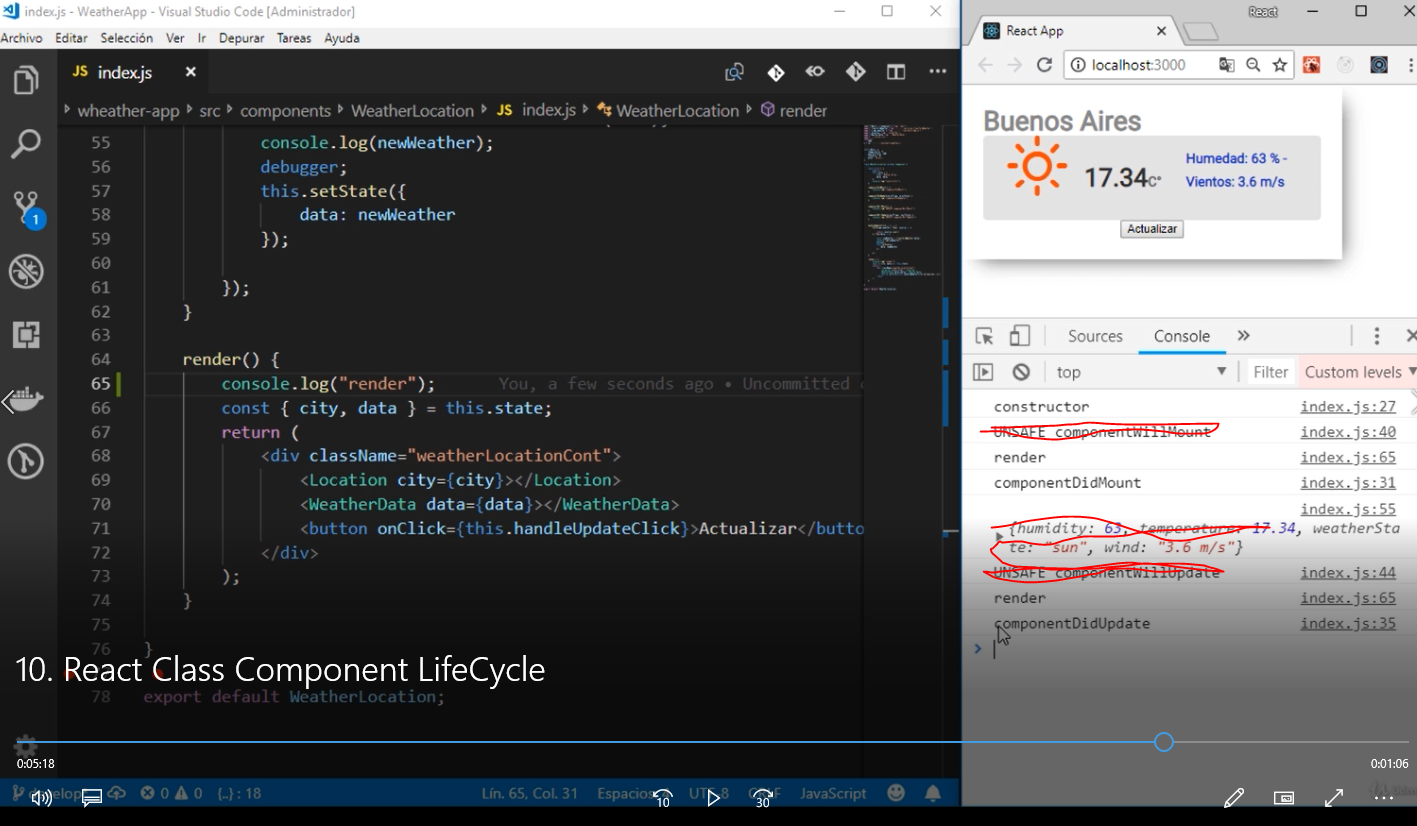


Al momento de realizar un fecth() (consumos de apis o servicios), en el navegador podemos encontrar en la parte de network y filtrado por XHR los datos que nos retorna la api anteriormente consumida y poder así tener claridad y tener un mejor manejo de los datos, igualmente podemos hacer uso de las pestañas header, preview and timing.

**CICLO DE VIDA DE RENDERIZACIÓN DE LOS COMPONENTES**



Esto sucede cuando la aplicación se inicia por primera vez

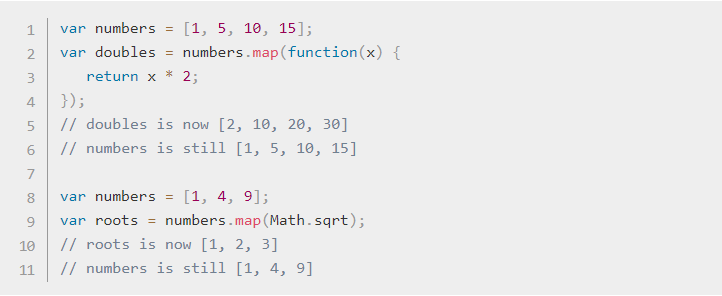


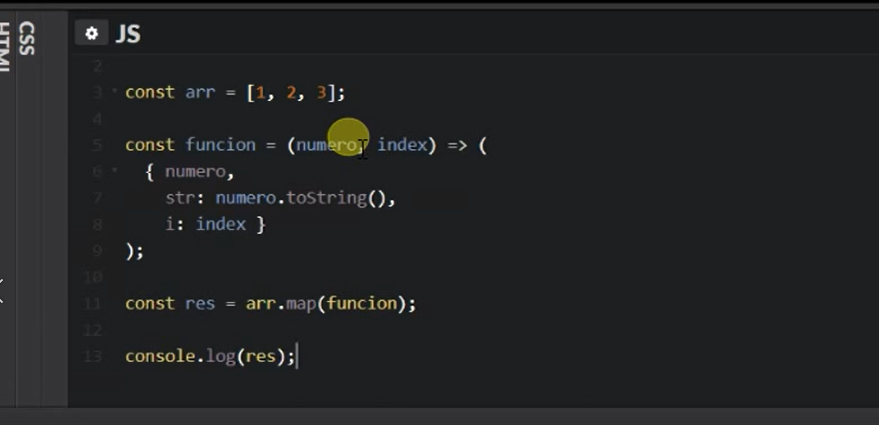
Esto es lo que sucede cuando se renderiza por segunda vez

**FUNTIONS DE COLLECCTIONS MAP**

[**https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos\_globales/Array/map**](https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos_globales/Array/map)

El método **map()** crea un nuevo array con los resultados de la llamada a la función indicada aplicados a cada uno de sus elementos.





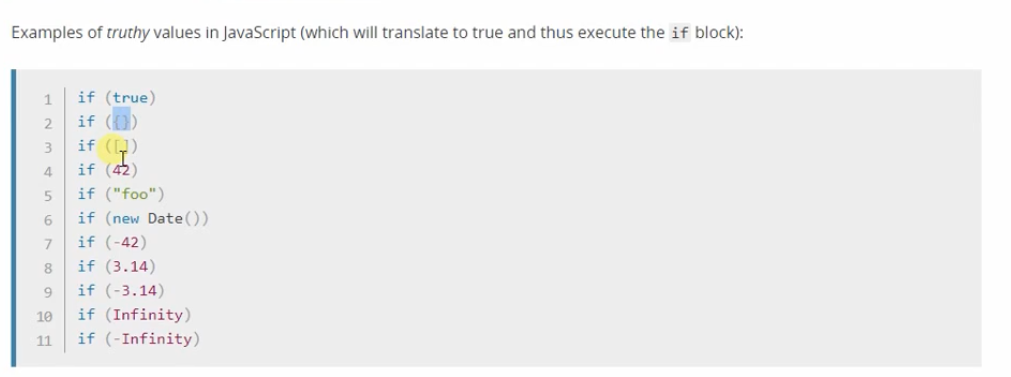
Este código retorna un objeto creado a parte de un array.

**PARA TENER EN CUENTA**

Es importante tener en cuenta que la inicialización de los estados (this.state = { data: valor}) solo se puede realizar en el constructor del componentes, a partir de allí si se desea actualizar algún estado solo se puede utilizar la función this.setState({ data:nuevoValor }).

**THURTY VALUES**

Esto hace parte de los valores que javascript toma como valores verdaderos.



**FALSY VALUES**

Hace referencia a los valores que javascript toma como falsos

