**UNIDAD 4: ReactJS**

1. [UNIDAD 4: Introducción](https://learn.nextu.com/mod/page/view.php?id=3608&pid=P_WEBDEV_V2)

**Bienvenido a Herramientas, Build Systems y tecnologías emergentes**

ReactJS es conocida por ser una de las librerías JavaScript con mayor rendimiento gracias a todas las herramientas y/o características que ésta posee, especialmente por su DOM virtual. En esta Unidad veremos cómo es el funcionamiento de ReactJS y por qué su rendimiento es tan alto.

**Puntos de aprendizaje**

Unidad 4: ReactJS

**Los objetivos del Programa que corresponden a esta Unidad son:**

* Crear ambientes de programación y a gestionar recursos.
* Implementar la construcción de sitios web a partir de componentes.

**Los objetivos del Curso 8 que corresponden a esta Unidad son:**

* Identificar y apropiar tendencias web.
* Aplicar el concepto ambiente web.
* Implementar web components.

1. Lección 1: ¿Qué es ReactJS?
   1. [Introducción a ReactJS](https://learn.nextu.com/mod/lesson/view.php?id=3609&pid=P_WEBDEV_V2)

Standard de desarrollo – Web components / Componentes Web, que cuentan con:



Esto con el fin de poder reutilizarlo cuantas veces se desee y dentro de otra aplicación si es necesario.

ReactJS:

* + Es bastante eficiente
  + Es bastante funcional

Librería escrita en JavaScript y desarrollada por Facebook.



Características de ReactJS:

* + Rendimiento y eficiencia
  + Flujo de datos unidireccional – se dice que es una de la forma más eficiente de datos. Más sencillo para encontrar errores.
  + DOM virtual – principal característica, y este se encarga de guardar en memoria un estado inicial de cada uno de los componentes que conforma la app, y de hacerle un seguimiento exhaustivo de cada uno de ellos con el fin de identificar cuales presentan cambios. Una vez lo detecta se encarga de renderizar únicamente los componentes que presentan modificaciones sin la necesidad de renderizar todos los componentes.
  + Compatibilidad: compatibilidad con los diferentes navegadores y compatibilidad con otras librerías y frameworks javascript
    - Compatibilidad con otros navegadores es bastante buena, incluso en versiones más antiguas como en el caso de internet explorer.
    - Compatibilidad con otros Frameworks y/o librerías JavaScript: es muy buena siempre y cuando, estas no tengan un control sobre el DOM ya que React es quien se encarga de controlarlo.
  + Fácil de aprender
  + Orientada a componentes lo que facilita: crear, actualizar, modificar y unir componentes en cualquier momento durante el proceso de desarrollo y soporte

Aplicaciones con funcionalidades desarrolladas con React:

Facebook e Instagram

Netflix

HipChat (Atlassian) comunicaciones en equipo de trabajo

* 1. Actividad 1

Tiempo de pensar

Hemos visto la introducción de la librería de JavaScript llamada ReactJS, sus principales características y algunas de las aplicaciones más usadas en la actualidad en las que se usó esta librería para la creación de ciertos componentes y/o funcionalidades durante su proceso de desarrollo, así que te invitamos a que realices esta actividad para que afiances tus conocimientos.  
  
Instrucciones:  
  
Lee el enunciado con atención y realiza la asociación que consideres correcta.  
  
Enunciado:  
  
ReactJS es una librería escrita en código JavaScript y desarrollada por la reconocida empresa Facebook, con el fin de facilitar el proceso de creación de componentes Web que puedan ser reutilizados. Esta herramienta tiene unas características que la hacen especial, a continuación, verás algunas de ellas listadas y tu misión será asociales la definición que les corresponde. ¡Adelante!



* 1. Lectura: Ventajas y desventajas de ReactJS

Ventajas & desventajas de ReactJS



Como ya lo sabes, ReactJS es una de las librerías Front-end de JavaScript más potentes que existe actualmente y que cuenta con una comunidad de desarrolladores que hacen que cada vez mejore más y tenga nuevas características para ofrecerle a quienes lo usan en la creación de sus aplicaciones Web. ReactJS, como cualquier otra librería JavaScript, tiene sus ventajas y desventajas, así que veamos cuáles son:

##### **Ventajas de ReactJS:**

Es una herramienta fácil de aprender, ya que cuenta con una documentación y una comunidad de desarrolladores muy amplia que siempre está abierta a resolver dudas o inquietudes. Adicionalmente, a nivel de sintaxis es simple, ya que si comprendemos apropiadamente el concepto de Web components y cómo es esta tendencia de trabajo para la creación de aplicaciones Web, será bastante fácil aprender ReactJS.



Al trabajar con componentes independientes, la creación y el mantenimiento de nuestras aplicaciones se convierte en una tarea muy sencilla y rápida. Detectar los errores no toma mucho tiempo y agregar nuevas funcionalidades a la aplicación se convierte en algo bastante simple.



Su capacidad para renderizar rápidamente cada uno de los componentes de la aplicación permite brindar una mejor experiencia de usuario.



Es posible combinar React con otros frameworks o bibliotecas para lograr funcionalidades necesarias con las que este no cuenta pero con los que se está combinando y/o integrando.

##### REACT + JS LIBRARIES

Contar con una comunidad de desarrollo tan amplia y con un grupo de desarrolladores tan expertos y reconocidos detrás de su creación hace pensar que es una librería que estará un buen periodo de tiempo en el mundo del desarrollo Front-end.



ReactJS es de las pocas librerías JavaScript que se integra y soporta versiones antiguas de Internet Explorer, de forma que, si dentro de los requisitos de desarrollo de tu aplicación Web se encuentra el soporte de versiones antiguas de Internet Explorer, ReactJS es la librería que necesitas.

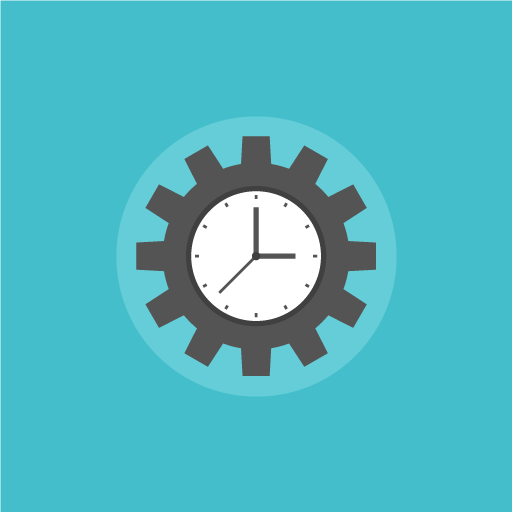


##### **Desventajas de ReactJS:**

React, al tener un DOM virtual que se encarga de actualizar el estado y el contenido de cada componente de manera individual, no se lleva bien con librerías como jQuery que buscan tener una manipulación total del DOM, lo que en ocasiones, al juntarlas, causa que no se muestre o no se renderice la aplicación por la compatibilidad de éstos.



Si se tiene un volumen de datos complejo y bastante amplio, en algunas ocasiones el tiempo de renderizado puede tomar más tiempo de lo esperado.



Aunque mencionamos que React es compatible con otras librerías y frameworks que agregan ciertas funcionalidades a nuestras aplicaciones, al momento de compilarlas y desplegarlas puede tomar un poco más de tiempo, ya que debe instalar dependencia de dichas librerías que se usaron para agregar funciones requeridas y que, como mencionamos, React no posee, por eso lo conocen como el “Frankenstein” de las librerías JavaScript.

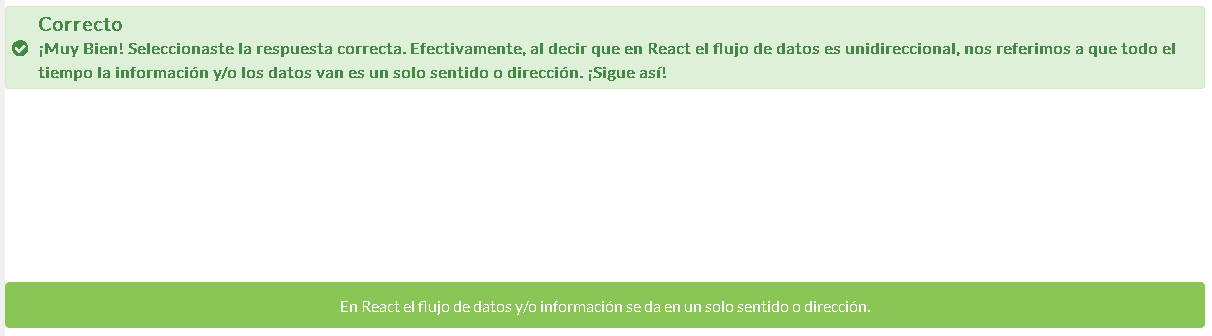


Como podemos ver, ReactJS tiene más ventajas que desventajas. A decir verdad, es una librería muy potente y sencilla de implementar para trabajar la parte visual de nuestra aplicación y con el pasar del tiempo continuará creciendo y mejorando para que todas estas desventajas empiecen a desaparecer.

* 1. Actividad 2

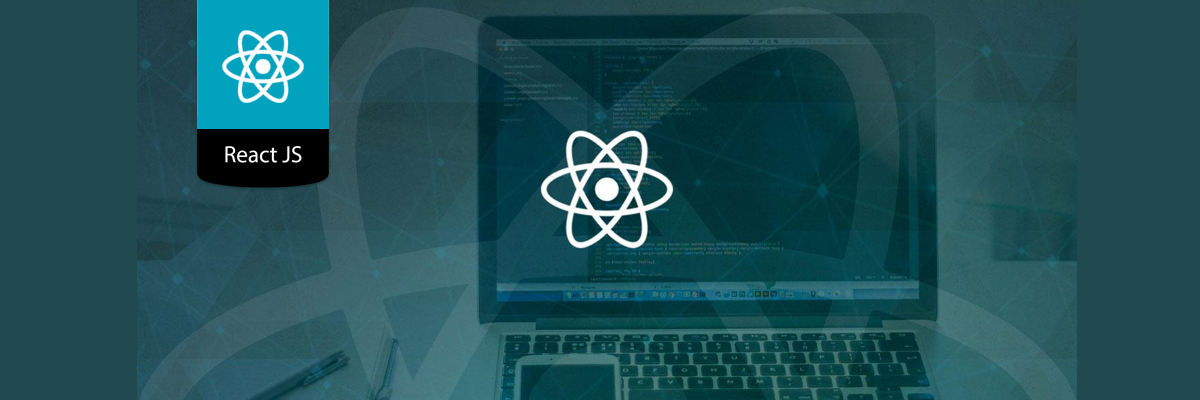
Tiempo de pensar

Hemos estudiado las diferentes ventajas y desventajas que nos ofrece ReactJS al momento de utilizarla para construir las vistas de nuestras aplicaciones Web, así que es hora de afianzar tus conocimientos a través de la siguiente actividad.  
  
Instrucciones:  
  
Lee el siguiente enunciado y selecciona la respuesta correcta.  
  
Enunciado:  
  
Al referirnos a React como una librería cuyo flujo de datos es unidireccional, estamos afirmando que:



* 1. Lectura: ¿Cómo funciona ReactJS?

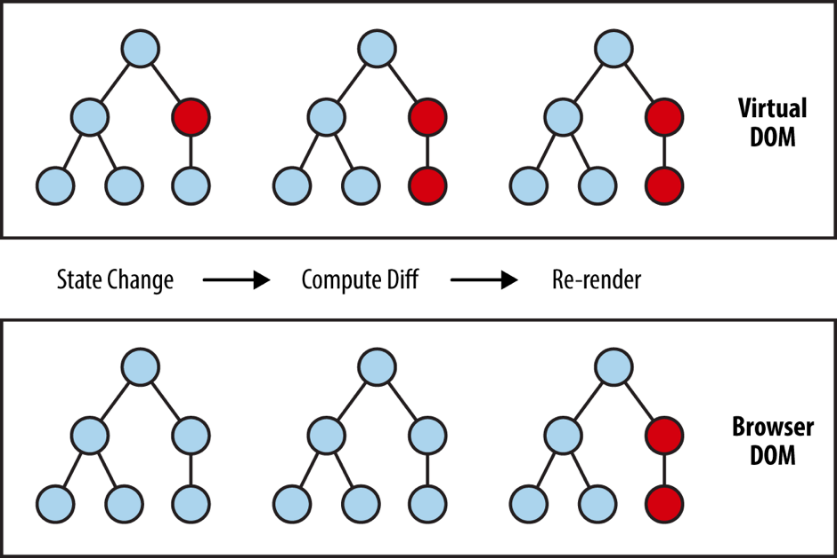
**¿Cómo funciona ReactJS?**



ReactJS es conocida por ser una de las librerías JavaScript con mayor rendimiento gracias a todas la herramientas y/o características que ésta posee, especialmente por su DOM virtual, pero veamos cómo es el funcionamiento de ReactJS que hace que su rendimiento sea tan alto.

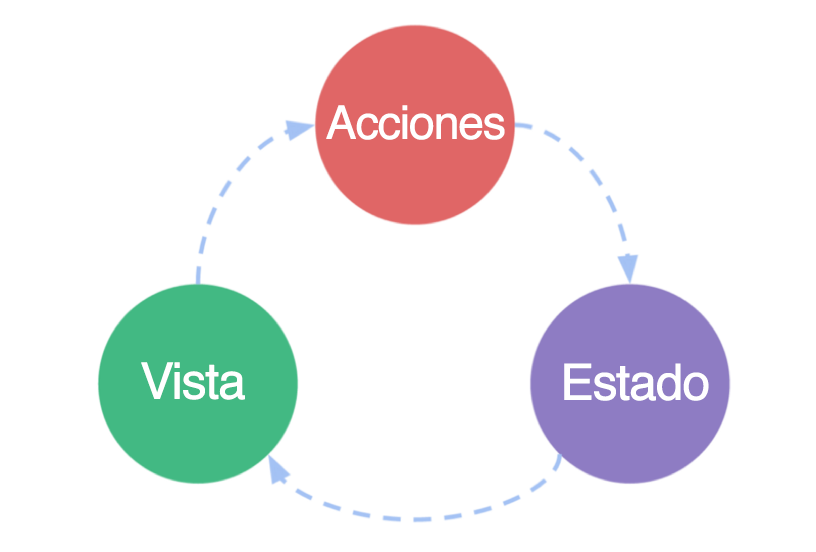
**Funcionamiento de react**

ReactJS está creado con el fin de de manejar funciones que permitan tener un control muy preciso sobre los diferentes componentes que construimos y que conforman nuestra página Web, dicho control se logra a través de un elemento fundamental conocido como el estado o state. La primera vez que nuestra aplicación es renderizada, el DOM virtual de React guarda en memoria el estado inicial de todos lo componentes de nuestra página y los mantiene en constante vigilancia. Una vez ocurre algún cambio en alguno de los estados de los componentes de nuestra aplicación, el ya mencionado DOM virtual se encarga de identificar dichos componentes que sufrieron modificaciones de estado comparándolos con el estado inicial que tiene almacenado en memoria y se da el proceso por el cual se considera que el rendimiento de React es tan alto. El DOM virtual se encarga de renderizar nuevamente los componentes que presentan cambios, guardando este nuevo estado en memoria y sin tener la necesidad de cargar y renderizar nuevamente nuestra aplicación para notar los cambios que presentan nuestros componentes.



**Flujo de datos:**

Dentro del funcionamiento de React, hay una característica y/o punto muy importante que debemos mencionar: el flujo de datos unidireccional, también conocido como flujo de datos en un solo sentido. Éste permite planear la forma de construir las aplicaciones más complejas e identificar los diferentes errores que se pueden presentar en éstas. A diferencia de las demás librerías que manejan un flujo de datos bidireccional, el flujo en un solo sentido permite hacer un seguimiento muy detenido al funcionamiento de nuestra aplicación y transporte de información, y así obtener una respuesta mucho más rápida. Es aquí donde podemos respaldar lo que hablábamos inicialmente cuando dijimos que React es una de las librerías con un alto rendimiento a comparación de las demás.

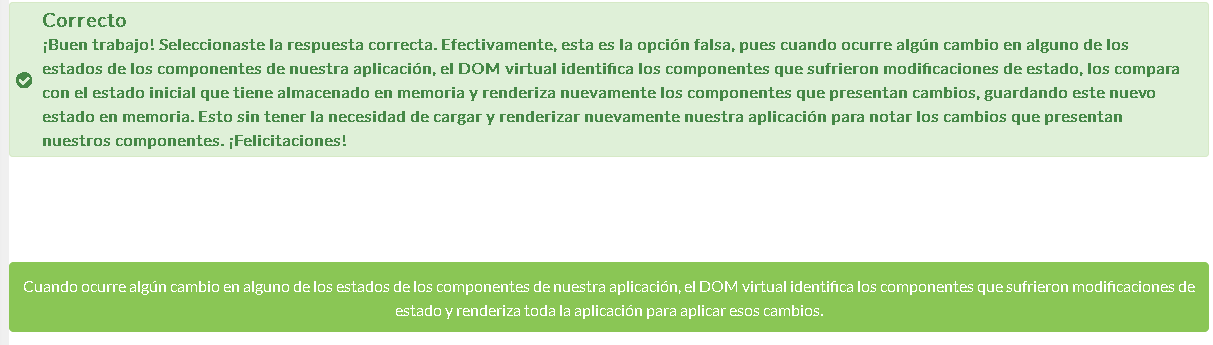


Esta es la razón por la que el funcionamiento de ReactJS es tan alto, pues cada una de las características o herramientas que la componen tiene una tarea en específico que va cumpliendo a medida que los componentes van cambiando y se le van mostrando al usuario, reduciendo los tiempos de carga y permitiendo reutilizar dichos componentes para desarrollar futuras aplicaciones.

* 1. Actividad 3

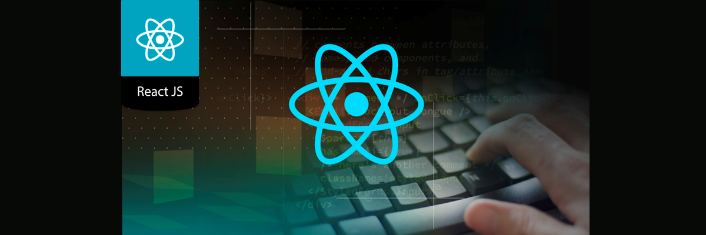
Tiempo de pensar

*Ahora que sabes cómo es el funcionamiento de ReactJS, te invitamos a reforzar tus conocimientos con la siguiente actividad.  
  
Instrucciones:  
  
Lee el siguiente enunciado y selecciona la respuesta correcta.  
  
Enunciado:  
  
A continuación te presentamos algunas afirmaciones sobre el funcionamiento de ReacJS, tu misión será encontrar la opción que contiene una afirmación que no corresponde con lo que has visto hasta ahora. ¡Adelante!*



* 1. Lectura: Por qué trabajar con ReactJS

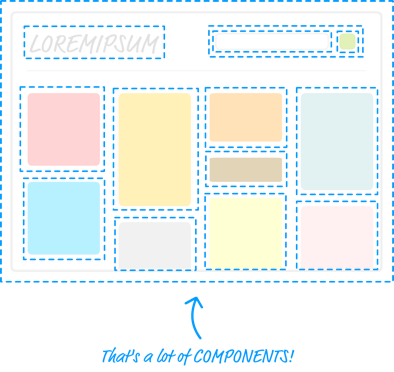
¿Por qué trabajar con ReactJS?



Existen diferentes librerías y frameworks Front-end de JavaScript que nos permiten crear las vistas de nuestras aplicaciones Web de forma rápida y eficiente, lo que dificulta la decisión cuando debemos seleccionar una de estas librerías para implementar en la creación de nuestros proyectos. ReactJS es una de ellas, de la cual hemos venido hablando, ésta nos ofrece muchas características para trabajar, así que te invitamos a explorar algunas de las razones por las cuales deberíamos seleccionarla para crear la interfaz gráfica de nuestras aplicaciones Web:

**WebComponents o Componentes Web**

En la actualidad, el desarrollo de componentes es una tendencia o filosofía de trabajo que se está implementado fuertemente en el proceso de creación de aplicaciones Web y móviles. ReactJs y React Native nos ofrecen la posibilidad de trabajar y crear nuestros propios componentes, los cuales podemos reusar y modificar para construir diferentes aplicaciones en el momento en que sea necesario, estos componentes tienen un ciclo de vida dentro de la aplicación.



**Eficiencia**

El hecho de que React tiene su propio DOM virtual, donde cada uno de los componentes de nuestra aplicación está activo, hace que el rendimiento de la aplicación sea más flexible y amplio, ya que React puede saber qué componente cambió y en qué momento lo hizo sin actualizar toda la aplicación sino solo esa parte que sufrió cambio, lo que hace que el usuario no tenga tiempo de espera mientras se recarga nuevamente la aplicación sino que con ésta mostrándose normalmente en pantalla podrá notar que los componentes se van actualizando uno a uno sin afectar la carga o visualización de los demás.



**Mejoramiento y crecimiento continuo**

Los ingenieros de desarrollo de Facebook fueron los encargados de crear esta librería y una vez fue publicada, la comunidad de desarrolladores empezó a aportar. Esto lo vienen haciendo hasta la fecha para que la librería se mantenga y tenga un constante crecimiento y mejoramiento.



**Compatibilidad con los navegadores**

ReactJS es de las pocas librerías JavaScript que se integra y soporta versiones antiguas de Internet Explorer de manera que, si dentro de los requisitos de desarrollo de tu aplicación Web se encuentra el soporte de versiones antiguas de Internet Explorer, ReactJS será la librería que necesitas. Adicionalmente, se integra a la perfección con los demás navegadores como Safari, Google Chrome, Opera, Firefox, etc.



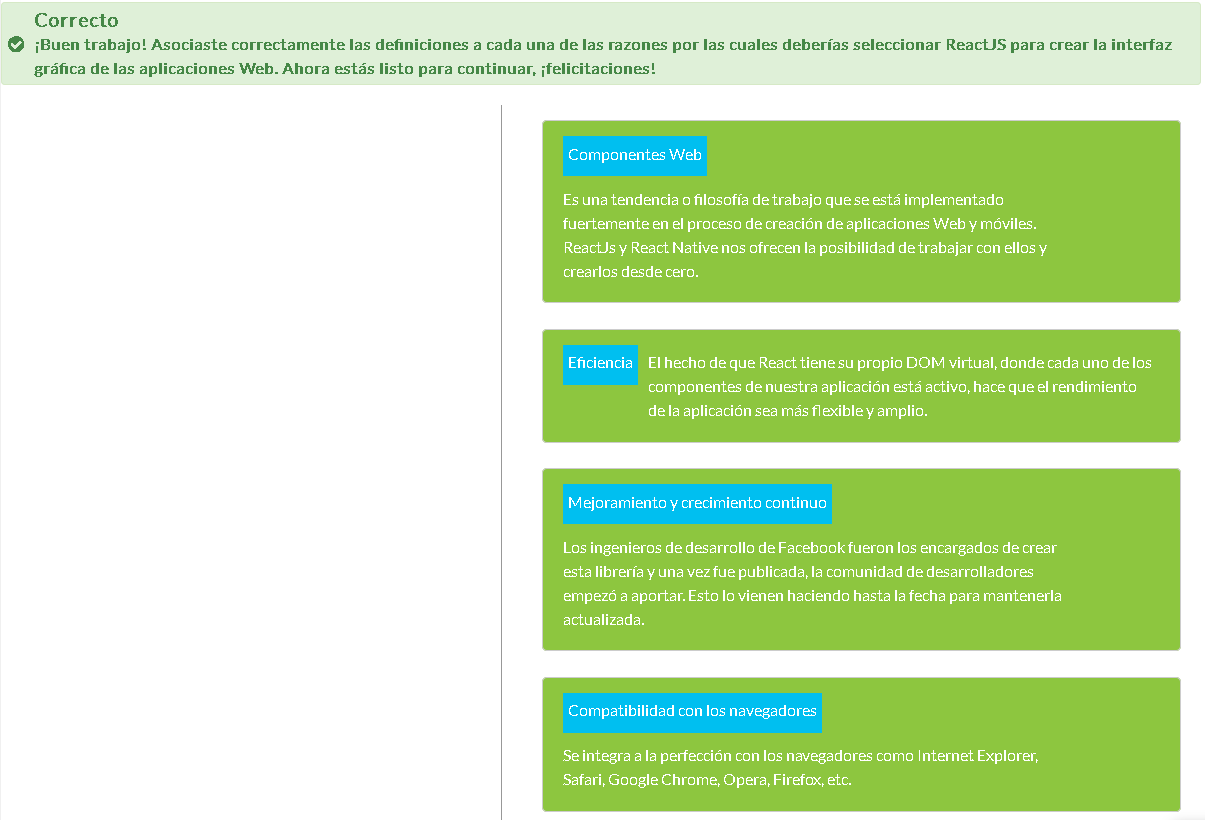
Otras de las razones para elegir ReactJS como la librería Front-end para crear nuestras aplicaciones Web son:

* Navegadores como Google Chrome tienen herramientas y extensiones en la Chrome Store que nos permiten hacerle seguimiento a nuestro código a través del navegador y hacer seguimiento de cómo va funcionando el DOM virtual de ReactJS.
* Cuando se crea una aplicación Web con ReactJS, si se desea crear una aplicación móvil en la parte del Front-end existe React Native, la cual tiene una estructura similar y la transición no es es compleja.
  1. Actividad 4

Tiempo de pensar

Ahora que sabes por qué trabajar con ReactJS, te invitamos a que valides tus conocimientos con la siguiente actividad.  
  
Instrucciones:  
  
Lee atentamente el enunciado y realiza la asociación que consideres correcta.

Enunciado:  
  
ReactJS nos ofrece varias razones por las cuales deberíamos seleccionarla para crear la interfaz gráfica de nuestras aplicaciones Web. A continuación, verás esas razones, las cuales deberás asociar con sus respectivas definiciones. ¡Adelante!



1. Lección 2: Configuración ReactJS
   1. [Lectura: Babel.JS](https://learn.nextu.com/mod/lesson/view.php?id=3610&pid=P_WEBDEV_V2)

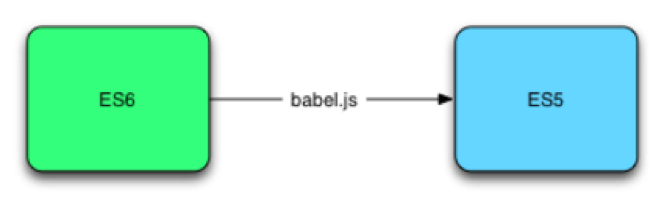
Babel.js



**Babel es una herramienta que nos permitirá implementar las características de JavaScript ECMAScript 6-ES6, convirtiendo el código en ES5 para que sea leído por cualquier navegador. ¿Cómo? Aquí lo verás.**

**La razón:**

ES6 presenta nuevas características que hacen que desarrolladores, como tú, prefieran trabajar con este lenguaje, lo cual es muy crítico pues la mayoría de los navegadores y las plataformas de Javascript tienen poco soporte para él.



Pues bien, Babel.js implementa plugins -conocidos como presets- que realizan dicha transformación para React, ES2015 y npm. Por eso, podemos decir técnicamente que es un transcompilador.  
Es importante tener presente que, si quisieramos hacer uso de Babel, necesitamos tener instalado Node.js pues está escrito en JavaScript.

**Babel.js y Node.js:**

Para usar node.js e instalar el cliente de Babel.js necesitamos:  
  
1. Crear una carpeta para el nuevo proyecto.  
  
2. En la consola de tu sistema operativo navega hasta esa carpeta y ejecuta los comandos:

npm install babel-preset-es2015  
npm install babel-cli -g

Con esto habrás instalado el preset es2015 y también el cliente de babel de manera global, así podrás ejecutar los comandos de Babel en cualquier proyecto.  
  
3. Dentro de la carpeta crea un archivo llamado ".babelrc" y dentro coloca el texto { "presets": ["es2015"] } presta mucha atención a las comillas dobles, deben ser rectas, no arqueadas.  
  
4. En la carpeta del proyecto crea un archivo llamado origen.js y dentro coloca las siguientes líneas de código:

JavaScript

var lista=[2,3,5,7];

lista.map(x=> x\*x).forEach(x=>console.log(x));

Guarda el archivo y cierra el editor de código.  
  
5. En la consola ejecuta el comando:

babel origen.js –o destino.js

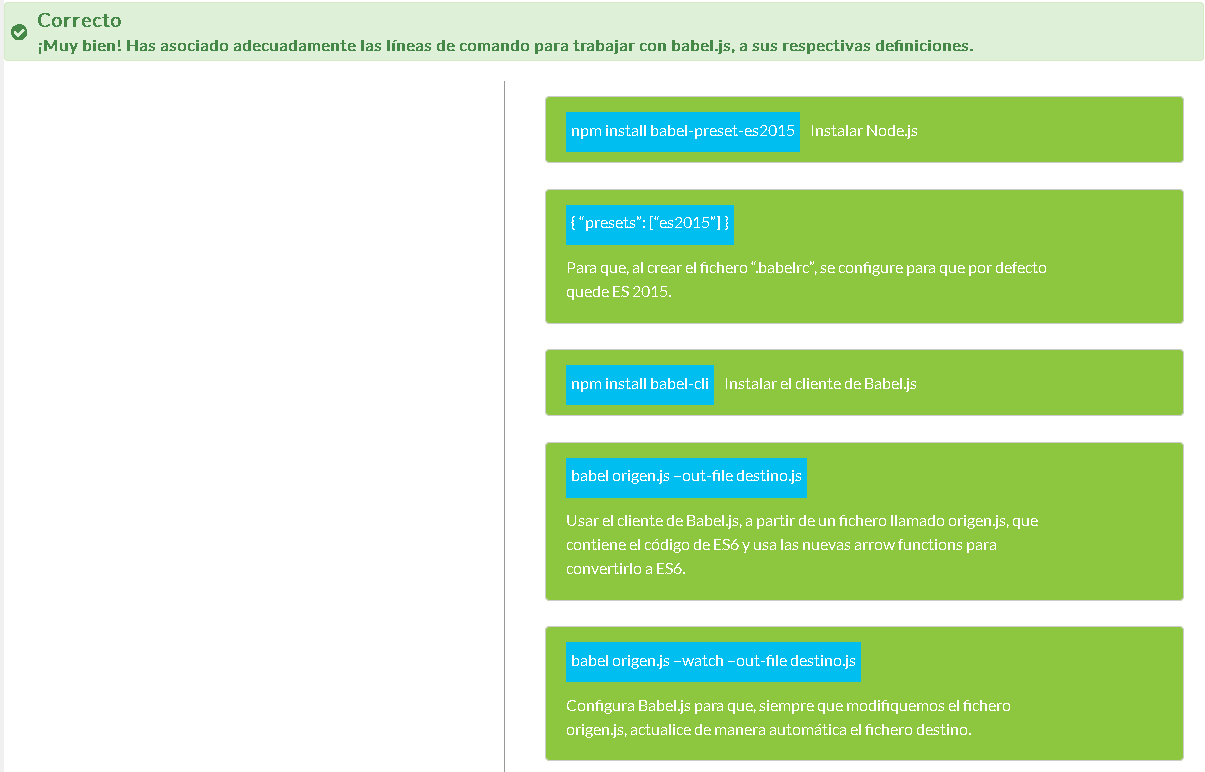
Esto creará un nuevo archivo llamado destino.js, abrelo para que puedas observar cómo se ha transpilado el código fuente de ES6 a ES5 y eso sería todo.

**Verás qué, en tu proyecto de convertirte en un gran desarrollador, usarás con frecuencia este transcompilador para implementar React.js**

* 1. Actividad 1

Tiempo de pensar

Ahora que sabes que Babel es una herramienta que nos permitirá implementar las características de JavaScript ECMAScript 6-ES6, convirtiendo el código en ES5 para que sea leído por cualquier navegador, te invitamos a realizar la siguiente actividad.   
  
Instrucciones:  
  
Asocia las líneas de comando que te permiten trabajar con Babel.js, presentadas en la columna izquierda, a la función que les corresponde de la columna derecha.¡Adelante!



* 1. Configuración del ambiente
  + Verificar Si temenos Node JS instalado, con el comando:

Node -v

* + Instalar babel de manera global con:

Npm install -g babel

* + Luego ejecutamos

Npm install -g babel-cli

* + Crear directorio donde se guarda la app
  + Luego se inicializa el proyecto

Npm init

* + Se confirma que el archivo se ha creado correctamente
  + Instalar webpak

Npm install webpack –save

Npm install webpack-dev-server –save

Npm install webpack-cli

* + Ahora instalar React:

Npm install react –save

Npm install react-dom –save

* + Plugins de Babel:

Npm install babel-core

Npm install babel-loader

Npm install babel-preset-react

Npm install babel-preset-es2015

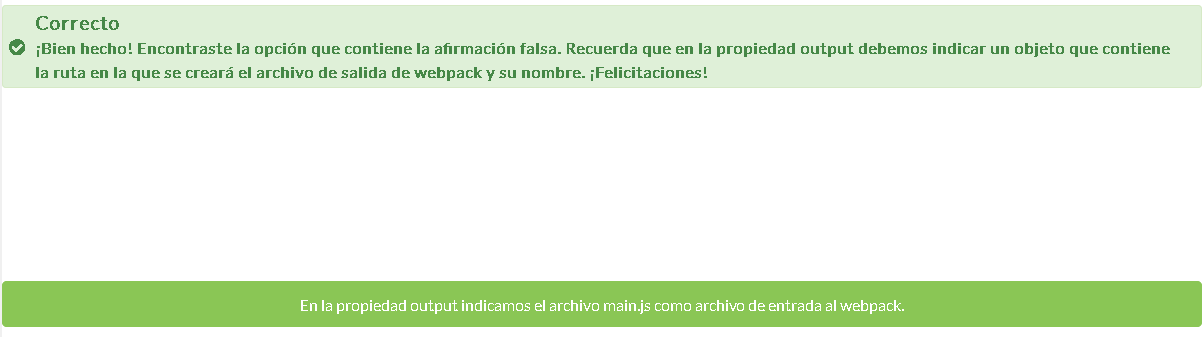
Para versiones nuevas de node y npm ([Node >= 8.10 and npm >= 5.6](https://nodejs.org/en/" \t "_blank)):

npx create-react-app firstapp

* 1. Actividad 2

Tiempo de pensar

Ya que sabes cómo preparar tu máquina para comenzar con la creación de aplicaciones con ReactJS, te invitamos a realizar la siguiente actividad para que pongas a prueba tus conocimientos.  
  
Instrucciones:  
  
Lee atentamente el siguiente enunciado y elige la opción que consideres correcta.  
  
Enunciado:  
  
A continuación encontrarás algunas afirmaciones sobre el proceso de configuración de ReactJS en tu máquina, tu misión será encontrar la afirmación que no corresponde a lo que has visto hasta ahora. ¡Adelante!



* 1. Lectura: WebPack

Webpack



Webpack se define como un bundler, o sistema de empaquetado de módulos, que permite compactar los recursos o assets de una aplicación, mejorando el desempeño en la carga inicial de la misma desde el cliente. Este tipo de empaquetadores se usan en proyectos que involucran gran cantidad de archivos y directorios enlazados mediante importaciones de diferentes tipos. Al usar ReactJS como framework en el desarrollo de una aplicación, encontramos una gran cantidad de archivos e importaciones a paquetes locales y externos. Por tal motivo, Webpack es un aliado necesario en la estructura global de la aplicación.

**Bundle:**

Un bundle, o paquete, es un archivo JavaScript que incluye todos los recursos que requiere la aplicación para funcionar adecuadamente. Dentro de este paquete podemos encontrar archivos JavaScript, CSS, Imágenes, LESS, SASS, entre otros.

**¿Cómo funciona?**

Webpack hace un recorrido por todo el código del proyecto buscando por sentencias en las que se importan archivos. Por ejemplo, en archivos JavaScript, la palabra reservada import es detectada por Webpack, así como en archivos CSS importados. Luego de identificar todas estas importaciones, Webpack realiza un bundle o paquete general donde reúne todos los recursos o assets y los organiza de tal manera que sus importaciones no se vean afectadas, manteniendo el correcto funcionamiento de la aplicación.

**Instalación:**

Para instalar Webpack en un proyecto debes tener instalado en tu máquina node y npm. Ejecuta el siguiente comando para instalar Webpack a nivel global:

$ npm install webpack -g

**Configuración:**

Para configurar Webpack se debe crear un archivo JavaScript con el nombre webpack.config.js, el cual cuenta con la siguiente estructura:

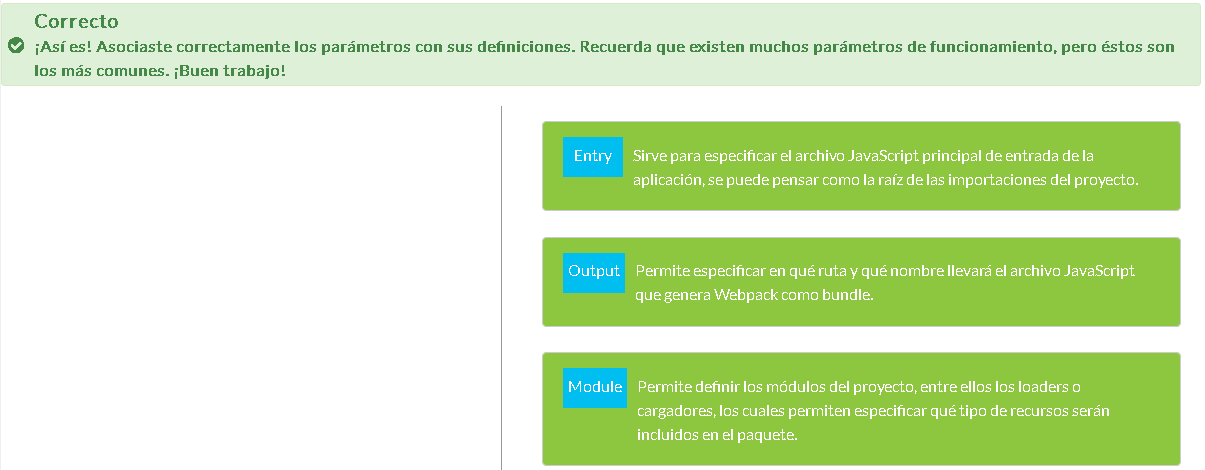


El archivo de configuración consta de un objeto llamado config, el cual cuenta con las propiedades que se definen por el desarrollador para indicarle a Webpack determinados parámetros de funcionamiento. Aunque existen muchos otros parámetros, los más comunes son: *entry, output, y module.* Entry sirve para especificar el archivo JavaScript principal de entrada de la aplicación, se puede pensar como la raíz de las importaciones del proyecto. Output permite especificar en qué ruta y qué nombre llevará el archivo JavaScript que genera Webpack como bundle. Module permite definir los módulos del proyecto, entre ellos los loaders o cargadores, los cuales permiten especificar qué tipo de recursos serán incluidos en el paquete.

* 1. Actividad 3

Tiempo de pensar

Webpack se define como un bundler, o sistema de empaquetado de módulos, que permite compactar los recursos o assets de una aplicación, mejorando el desempeño en la carga inicial de la misma desde el cliente. Ahora te invitamos a que refuerces tus conocimientos con la siguiente actividad  
Instrucciones:  
  
Lee atentamente el siguiente enunciado y realiza la asociación que consideres correcta.  
  
Enunciado:  
  
Como ya sabes, el archivo de configuración consta de un objeto llamado config, el cual cuenta con las propiedades que se definen por el desarrollador para indicarle a Webpack determinados parámetros de funcionamiento. A continuación, encontrarás los parámetros más comunes vistos hasta ahora, tu misión será asociar las definiciones correspondientes a cada uno de ellos. ¡Adelante!



1. Lección 3: Web Components
   1. [Lectura: JSX](https://learn.nextu.com/mod/lesson/view.php?id=3611&pid=P_WEBDEV_V2)

JSX



A continuación, te explicaremos de qué se trata el lenguaje JSX.  
Este lenguaje es una extensión de JavaScript, creado por Facebook, para el uso con la librería React.

También es un preprocesador que transforma el código de JavaScript. Así que no te preocupes si al comienzo pareciera que mezclas código HTML con los ficheros JavaScript porque no es cierto.   
  
Lo cierto es que: aunque parece que estuvieras escribiendo en HTML, en realidad es JavaScript. Observa:

JavaScript

var Icon = (

<div className='icon-container'>

<img

src='icon-react.png'

className='icon-image'

/>

</div>)

ReactDOM.render(Icon, document.getElementById('app'))

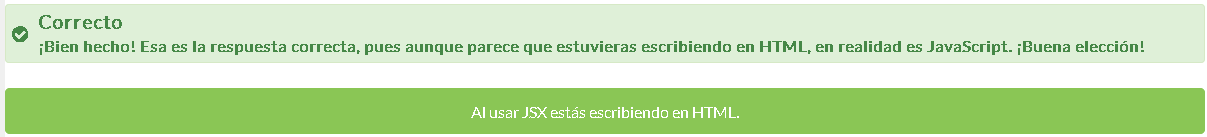
Verás:   
Es muy importante que tengas presente que algunas palabras reservadas se nombran de otra manera. Por ejemplo, Class (para definir clases de CSS) con JSX se escribe: className  
  
Usar JSX, en la medida en que nuestros proyectos vayan creciendo y sumamos más componentes, será la forma de agilizar los desarrollos. Por eso, una recomendación es que utilices un Bundler integrado al entorno de desarrollo, como: WebPack o Browserify, que sea éste el que aplique la transformación antes de publicar en producción.

**Recuerda:  
No es escribir HTML dentro de JS. Es crear JS de una forma más práctica.**

* 1. [Actividad 1](https://learn.nextu.com/mod/lesson/view.php?id=3611&pageid=14548&pid=P_WEBDEV_V2)

Tiempo de pensar

Ya que conoces qué es JSX, te invitamos a realizar la siguiente actividad para que afiances tus conocimientos.  
  
Instrucciones:  
  
Lee el siguiente enunciado y selecciona la respuesta correcta.  
  
Enunciado:  
  
A continuación encontrarás algunas afirmaciones sobre el lenguaje JSX, tu misión será encontrar cuál de esas opciones contiene una afirmación que no corresponde a lo que has visto hasta ahora. ¡Adelante!



* 1. Componentes

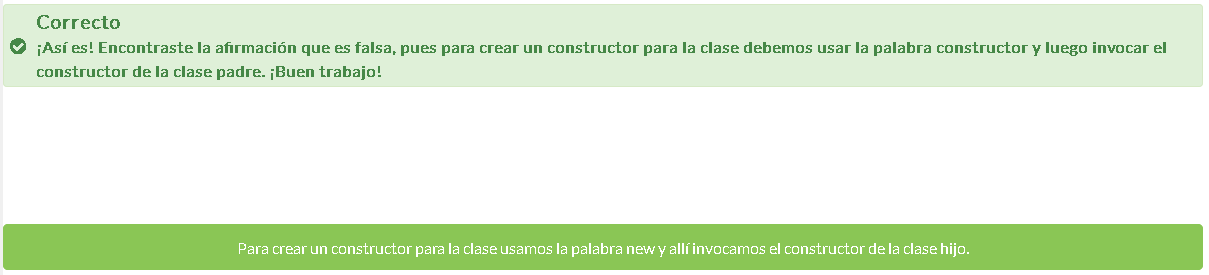
Un componente es la unidad básica de una aplicación en ReactJS, cada una de ellos posee una porción de las vistas de un proyecto.

Un componente es en sí una clase con una serie de métodos

* 1. Actividad 2

Tiempo de pensar

En esta ocasión conociste los componentes en ReactJS: qué son, cómo se definen y para qué sirven en las aplicaciones que vamos a desarrollar. Ahora es tiempo de reforzar tus conocimientos a través de la siguiente actividad.  
  
Instrucciones:  
  
Lee el siguiente enunciado y selecciona la respuesta correcta.  
  
Enunciado:  
  
A continuación encontrarás algunas afirmaciones sobre los componentes en ReactJS, tu misión será encontrar cuál de esas opciones contiene una afirmación que no corresponde a lo que has visto hasta ahora. ¡Adelante!



* 1. Lectura: DOM Virtual

DOM Virtual



**Vamos a tratar de explicarte este concepto de la forma más simple posible.**

El DOM Virtual puede ser similar a una tabla con datos, en una página HTML, que en cada fila tiene un botón de edición para cambiar, de forma rápida, los datos.

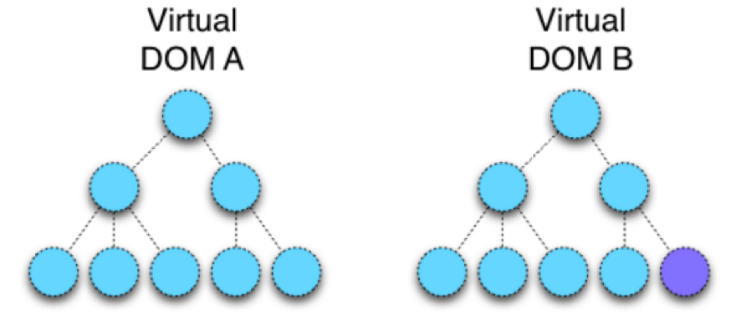
##### **DOM VIRTUAL A:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Dato 1** | **Dato 2** | **Dato 3** | **Edit** |
| Dato 4 | Dato 5 | Dato 6 | Edit |
| Dato 7 | Dato 8 | Dato 9 | Edit |

##### **Por ejemplo, podemos cambiar el Dato 9 por el Dato 10. DOM VIRTUAL B:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Dato 1** | **Dato 2** | **Dato 3** | **Edit** |
| Dato 4 | Dato 5 | Dato 6 | Edit |
| Dato 7 | Dato 8 | **Dato 10** | Edit |

Esto se explica de la siguiente manera:   
  
Como la tabla se genera desde un array de objetos, cuando lo editamos se actualiza y, al terminar, se vuelve a generar.   
  
Pero pon mucha atención porque, a pesar de que la edición es muy sencilla y en cada dato, entre más nodos cambiamos en el DOM, menor es el rendimiento del navegador al hacer la actualización.   
  
¿Por qué?  
  
Volviendo al ejemplo de la tabla, el navegador debe validar, entre los 9 datos que la componen, cuáles cambiaron y después se renderiza.

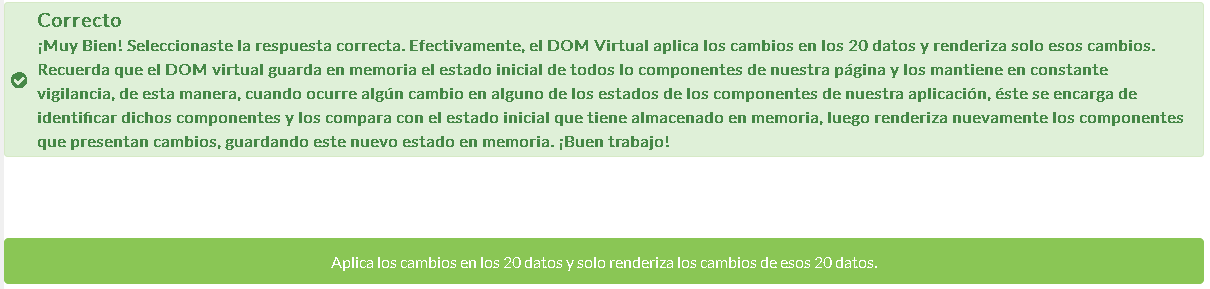


**Aun así, siempre será una mejor opción que actualizar la tabla por completo.**

* 1. Actividad 3

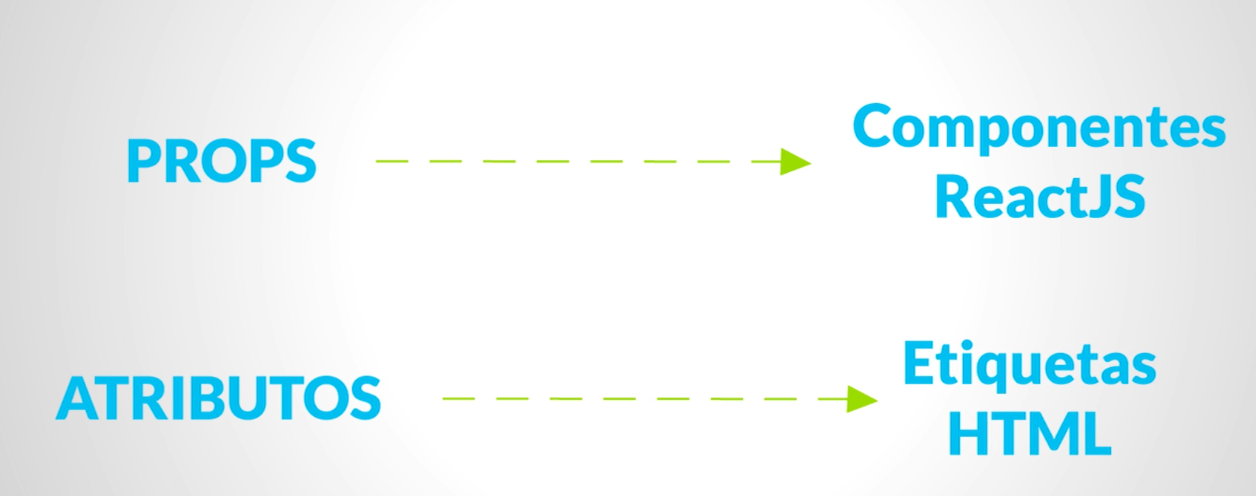
Tiempo de pensar

Ahora que sabes qué es el DOM Virtual, te invitamos a realizar la siguiente actividad para que pongas a prueba tus conocimientos.  
  
Instrucciones:  
  
Lee el siguiente enunciado y selecciona la respuesta correcta.  
  
Enunciado:  
  
Hace poco tiempo creaste una aplicación para un cliente donde tienes una lista con 100 datos. Hace unos días te notifican que 20 de estos elementos tienen cambios después de accionar un botón, así que los debes aplicar. Teniendo en cuenta los conocimientos adquiridos hasta ahora, ¿qué hace el DOM Virtual para aplicar esos cambios y garantizar la actualización correcta de tu aplicación?



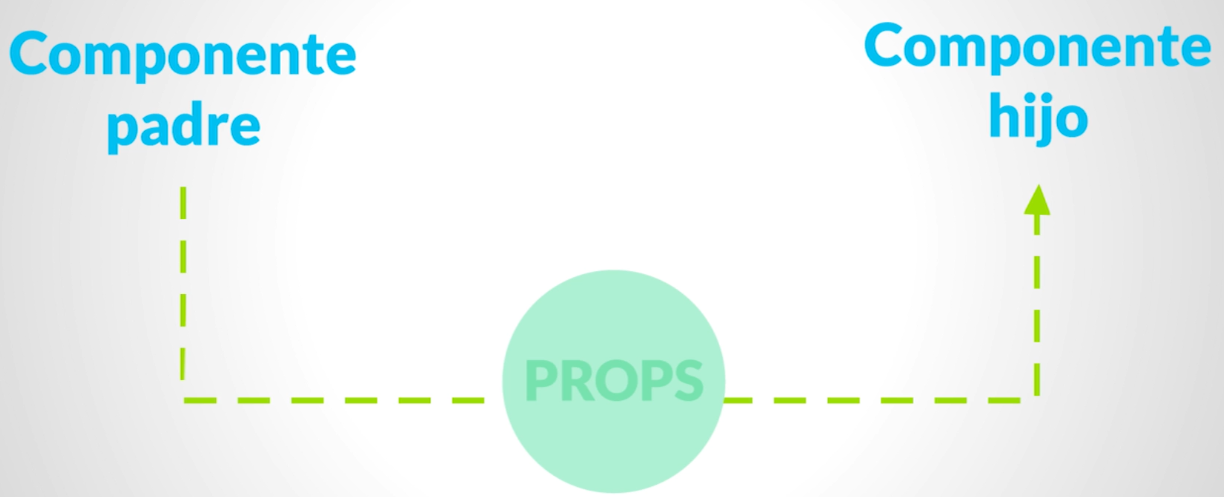
* 1. States & Props

Analogía:



Los props a diferencia de los state son inmutables, es decir no pueden ser cambiados o modificados.

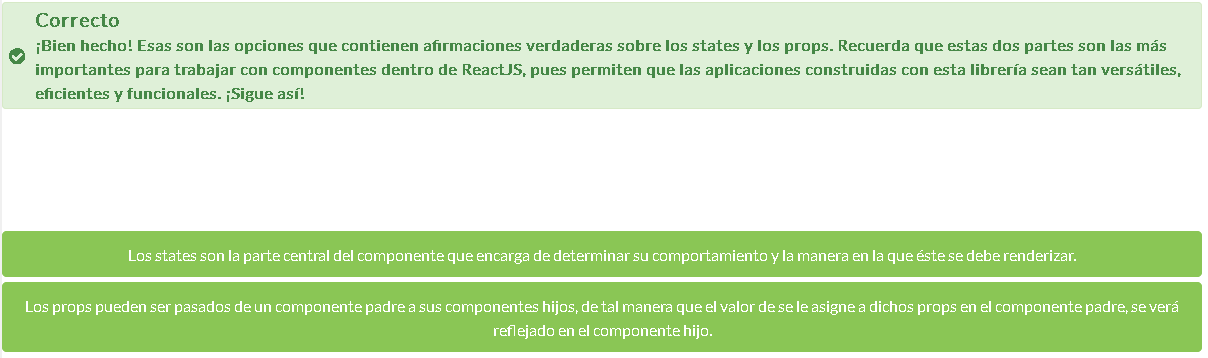
Los props pueden ser pasados de un componente padre a un componente hijo.



* 1. Actividad 4

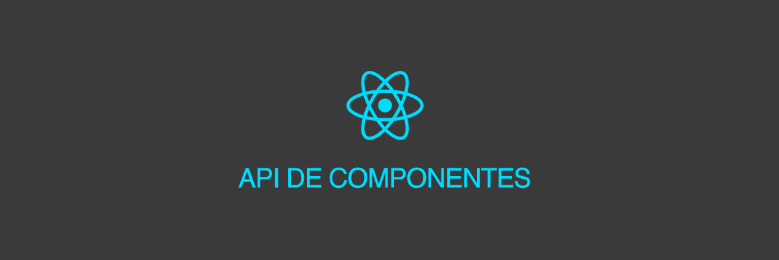
Tiempo de pensar

Ya que conoces dos de las partes más importantes para trabajar con componentes dentro de ReactJS y que permiten que nuestras aplicaciones construidas con esta librería sean tan versátiles, eficientes y funcionales, te invitamos a realizar la siguiente actividad para que pongas a prueba tus conocimientos.  
  
Instrucciones:  
  
Lee el siguiente enunciado y selecciona las opciones que consideres correctas.  
  
Enunciado:  
  
A continuación encontrarás algunas afirmaciones sobre states y props, dos de las partes más importantes para trabajar con componentes dentro de ReactJS, tu misión será encontrar cuáles de esas opciones contienen afirmaciones verdaderas sobre estas partes. ¡Adelante!



1. Lección 4: Ciclo de vida del componente
   1. [Lectura: API Components](https://learn.nextu.com/mod/lesson/view.php?id=3612&pid=P_WEBDEV_V2)

API Components



Dentro del ciclo de vida de un componente ReactJS, nos encontramos con diversos métodos que nos permite leerlo, visualizar su atributos, actualizarlo y modificarlo, con el fin de obtener determinados resultados de ser necesario. En el API de componentes, encontramos 3 métodos bastante interesantes que nos permiten actualizar los atributos como los estados y las características visuales de nuestros componentes, siempre y cuando dichos atributos o características no sean las props o propiedades pues éstas son inmutables, es decir, las propiedades no pueden ser alteradas y/o modificadas. Los métodos de los componentes API a los que nos referimos son **setState, forceUpdate & findDOMNode.**  
  
Veamos cómo funcionan:

**Método setState()**

Este es uno de los métodos más usados en React, ya que a través de él podemos actualizar el estado de un componente. Se debe tener presente que este método no se debe utilizar dentro del método render, ya que puede ocasionar un bucle infinito debido a que al detectarse una modificación de estado se intenta renderizar nuevamente el componente, entraría otra vez al método setState, se volvería a actualizar el estado y se volvería a renderizar ocasionando el ya mencionado bucle infinito.   
  
Veamos cómo podemos utilizar el método **setState():**



**Método forceUpdate()**

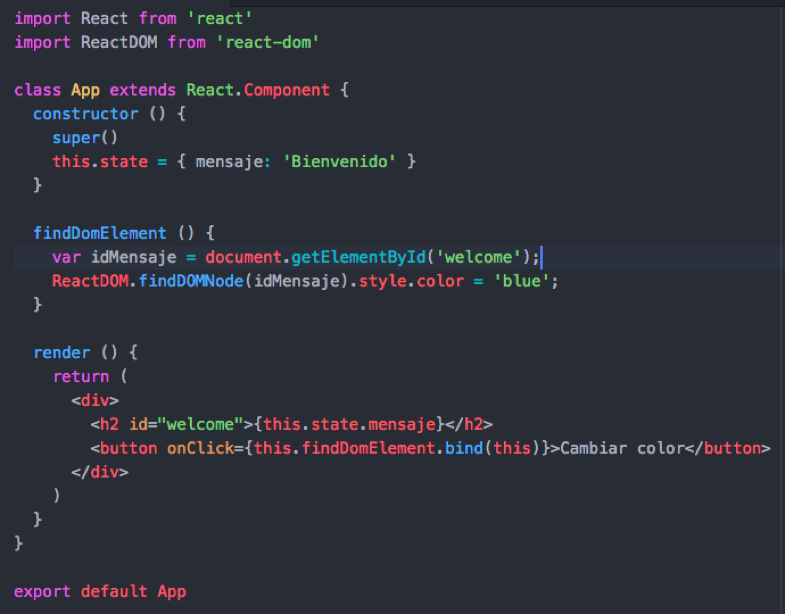
Este método, a diferencia del método anterior, no es usado para actualizar el estado del componente, sino que es usado para forzar la actualización del componente en general en cualquier momento. Es necesario tener presente que al ser un método que modifica un componente no puede ser puesto dentro del render porque nos ocasionará un error, la manera de evitar este error es crear una función por fuera del método render e invocar el atributo onClick de alguno de los elementos HTML o agregarla en alguno de los métodos del ciclo de vida de un componente a excepción del método render.

Veamos cómo utilizar el método **forceUpdate():**



**Método FindDOMNode()**

Este método permite manipular el DOM y todos sus nodos y/o elementos, para ello es necesario importar react-dom para que podamos usar el método ReactDOM.findDOMNode(). Este método, al igual que los anteriores, no puede ser usado dentro del método render, ya que éste busca actualizar las propiedades de los elementos del DOM y en el momento en el que éste se ejecuta, los elementos todavía no se han cargado en el DOM y produce un error.

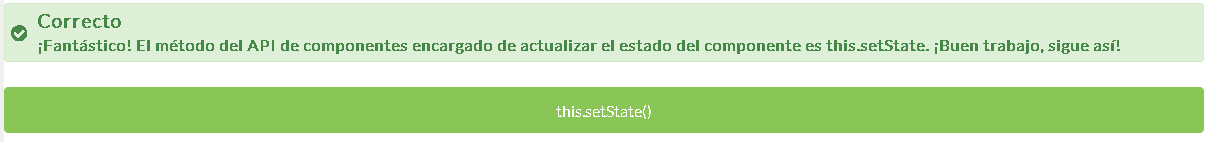


Esto es solo una parte de los métodos del API de componentes, no mencionamos los demás pues la mayoría fueron removidos desde la actualización 0.14 de React, en cambio éstos aún son utilizados y continúan en la versión actual.

* 1. Actividad 1

Tiempo de pensar

Ahora que conoces lo más importante de los métodos del API de componentes, te invitamos a que afiances tus conocimientos realizando la siguiente actividad  
  
Instrucciones:  
  
Lee atentamente el siguiente enunciado y selecciona la respuesta correcta.  
Enunciado:  
  
¿Cuál de las siguientes opciones corresponde al siguiente texto?  
Es conocido por ser el método del API de componentes que permite realizar la actualización del estado de un componente.



* 1. Ciclo de montaje

Ciclo de vida de un componente:

Ciclo de inicialización o montaje 🡪 Es el proceso en el que se encuentra un componente después de que lo hemos instanciado y queremos que se muestre en pantalla, o cuando desde un componente padre se hace un llamado para renderizar al componente hijo.

Preparación, por ejemplo, tener datos precargados de una base de datos, para cuando llegue el momento de renderizarlo no sea necesario que se ejecuten consultas.

4 fases:

* + Constructor: el método constructor, es un componente que se implementa previamente, el cual tiene una misión principal 🡪 inicializar todos los estados que se van a trabajar dentro de dicho componente y este siempre va acompañado por el método super, formato:

Constructor + super()

Si no se necesita inicializar previamente un estado, no hay necesidad de hacer uso de este método

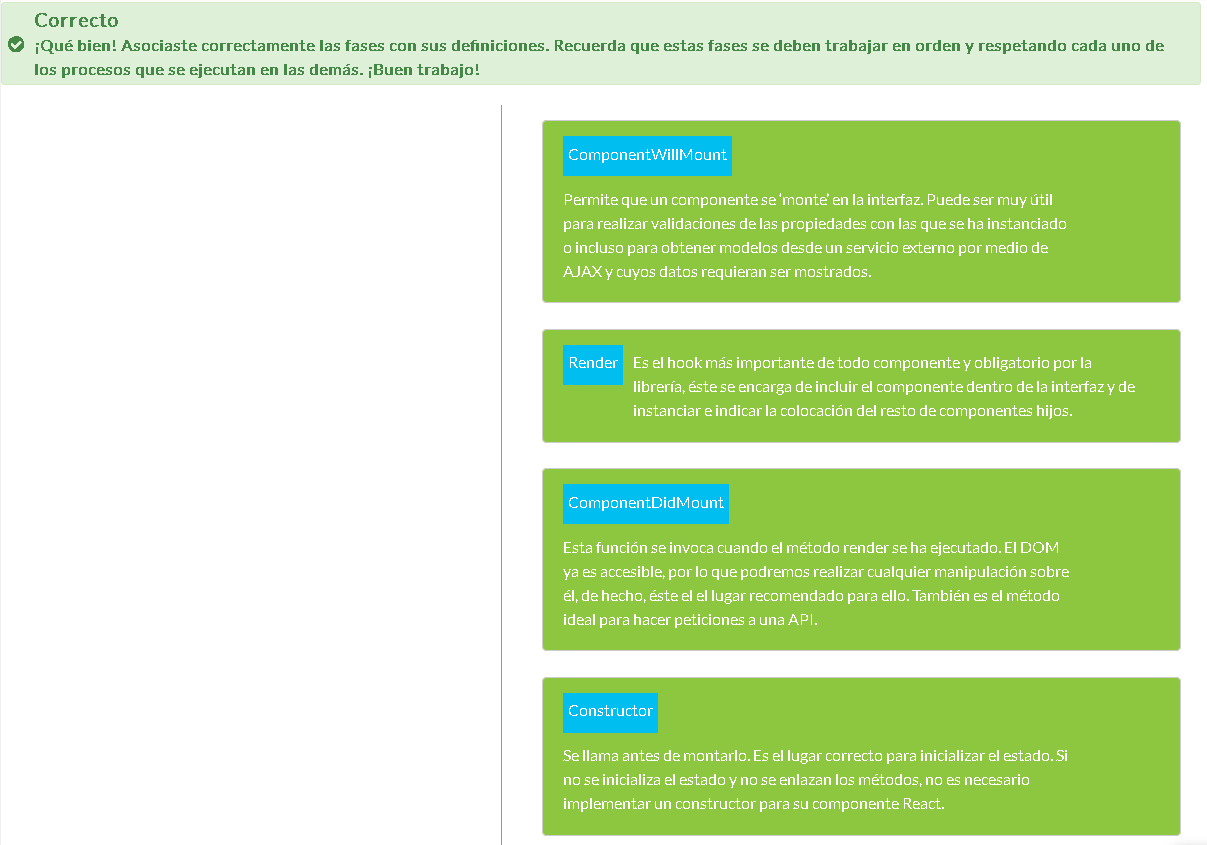
* + ComponentWillMount puede invocar funciones o mostrar datos y sólo es invocado una única vez en el cliente
  + Render, es considerado el método más importante de todo componente y su uso es obligatorio, a diferencia de los otros donde su uso es opcional. Render es el componente principal encargado de agregar el componente principal en la interfaz de usuario y de ir integrando e indicar los componentes hijos.

Dentro del método render es posible crear variables y realizar operaciones, validaciones y ciclos, pero no se recomienda modificar el valor de los estados dentro de este, ya que puede causar errores en el momento de montar nuestro componente.

* + ComponentDidMount, se ubica después que el método render, este debe ser el último método del ciclo de montaje, aquí es donde se puede modificar estados, variables, ejecutar funciones, entre otras. Al igual que ComponentWillMount esta se ejecuta una única vez en el cliente y siempre vendrá del método render
  1. Actividad 2

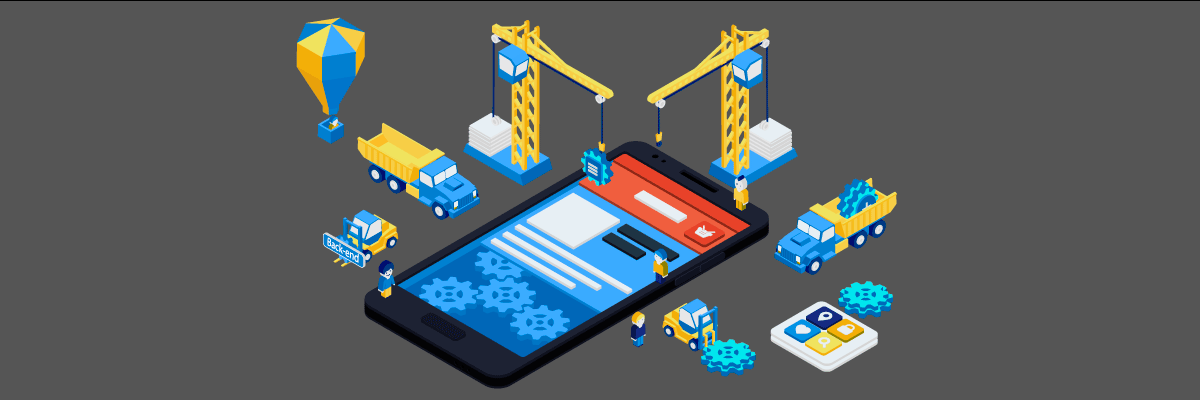
Tiempo de pensar

Ya que conoces todo lo relacionado con la etapa inicial del ciclo de vida de un componente, el ciclo de inicialización o montaje, te invitamos a afianzar tus conocimientos con la siguiente actividad  
  
Instrucciones:  
  
Lee atentamente el siguiente enunciado y realiza la asociación que consideres correcta.  
  
Enunciado:  
  
El ciclo de montaje es el proceso en el que se encuentra un componente después de que lo hemos instanciado y queremos que se muestre en pantalla o cuando desde el componente padre se hace un llamado para renderizar el componente hijo. Este ciclo se compone de 4 fases, las cuales se encuentran a continuación, así que tu misión será asociarles las definiciones correspondientes. ¡Adelante!



* 1. Lectura: Ciclo de actualización y desmontaje

Ciclo de actualización & desmontaje



Los ciclos de actualización y desmontaje son los ciclos que, aunque muchos no implementan durante el trabajo con componentes, tienen una vital importancia en el ciclo de vida de éstos, ya que ambos ciclos tienen una relación directa con los states o estados y con los props o propiedades, pues el proceso de actualización se da por dos motivos: cuando hay nuevos props o propiedades que está recibiendo el componente o cuando en determinado momento se realiza un cambio o modificación sobre el estado del componente. Éste consta de 5 fases o métodos, mientras que el proceso de destrucción o desmontaje consta de un solo solo método, el cual, como su nombre lo dice, se encarga de destruir o demostrar el componente y es considerado como la última fase o fase final.  
Veamos más a fondo estos dos ciclos o procesos:

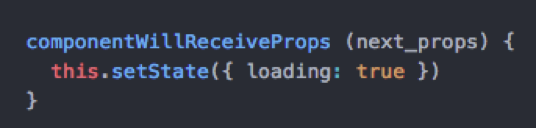
**Ciclo de actualización:**

Como ya se mencionó anteriormente, el ciclo de actualización se da en el momento en que nuestro documento recibe nuevas props o propiedades, o porque de alguna u otra manera el estado del componente cambió. El ciclo o proceso de actualización consta de 5 etapas fundamentales, entre la cuales encontramos:



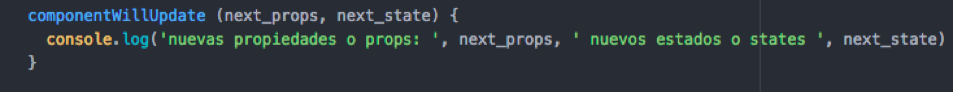
**componentWillReceiveProps:**

este método es ejecutado única y exclusivamente cuando el componente hijo recibe nuevas props o propiedades de parte de su componente padre y tiene como parámetro dichas props que va a recibir de parte de su componente padre.



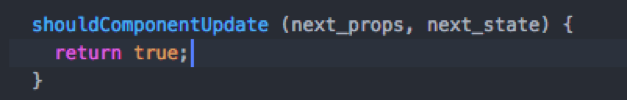
**componentWillUpdate:**

este método debe ser llamado antes del método render, justo en el momento en el que se reciben las nuevas props o propiedades y los nuevos states o estados, ya que aquí es donde podemos hacer los preparativos justo antes de que ocurra la actualización del componente y reciba como parámetros nuevas props o propiedades y nuevos estados o states. Es clave tener siempre presente que dentro de este método por ningún motivo podemos hacer uso de this.setState(), ya que lo más probable es que aparezca un error en nuestra aplicación, causándonos un bucle infinito que no permitirá que nuestro componente se renderice apropiadamente.



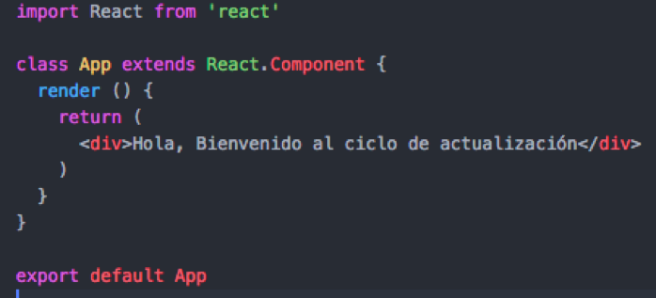
**shouldComponentUpdate:**

este método se ejecuta de la misma manera que el método componentWillUpdate. Con el método shouldComponetUpdate tenemos la posibilidad de mejorar el rendimiento de nuestra aplicación. Se debe tener algo siempre presente y es que este método o función por defecto siempre retorna true, pero si hacemos que se retorne false cancelaría el render hasta que se presente un nuevo cambio de propiedades o de estado. Lo más importante es que si este método retorna false los métodos **componetWillUpdate** y **componentwillReceiveProps** no serían ejecutados, ya que este método es el que valida si se deben ejecutar actualizaciones de componente.



**Render:**

el método render es considerado el método más importante en ReactJS, ya que básicamente es el que nos permite montar nuestros componentes a la interfaz gráfica. En el ciclo de actualización de un componente es necesario tener presente que si el método shouldComponentUpdate retorna false como resultado, el método render se cancelará hasta que se presente un nuevo cambio de atributos, es decir, un nuevo cambio de states o props.



**componentDidUpdate:**

una vez se renderiza nuestro componente, el método invoca el método **componentDidUpdate**, con el fin de realizar determinada operación en el DOM después de que se haya actualizado el componente, éste puede compararse con el método del ciclo de montaje **componentDidMount** en el sentido en el que ambos se dan o se ejecutan posterior a la renderización de nuestro componente.

****

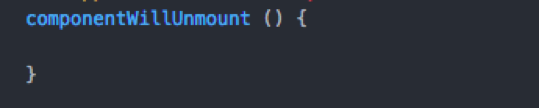
**Ciclo de desmontaje o destrucción:**

El ciclo de desmontaje es la última fase que se da en el ciclo de vida de un componente, ya que es aquí donde el componente se destruye o se desmonta del DOM. Este ciclo consta de un solo método y es el método **componentWillUnmount**.



**componentWillUnmount:**

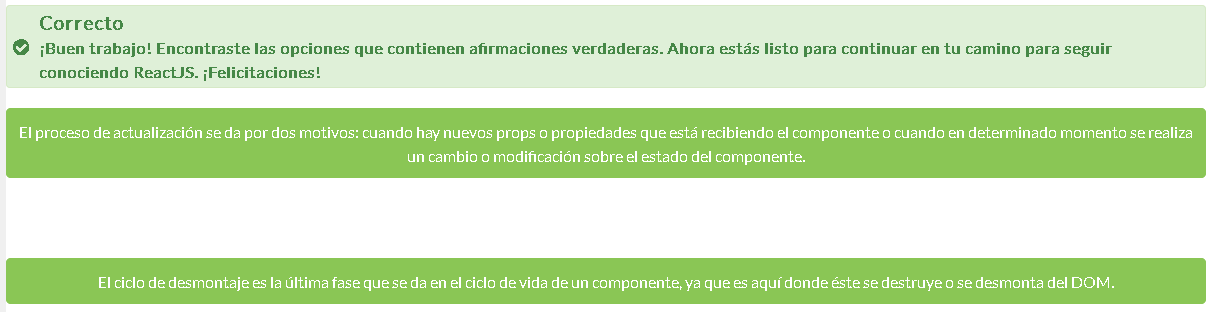
este método es utilizado para realizar diferentes operaciones de limpieza, cómo eliminar elementos que se crearon durante el componentDidMount, timers y otros elementos que puedan influir en el rendimiento de nuestra aplicación. Adicionalmente, ejecuta otras múltiples operaciones de limpieza con el fin de mantener siempre el buen rendimiento que tienen las aplicaciones React.



* 1. Actividad 3

Tiempo de pensar

Ahora que conoces el ciclo de actualización y desmontaje, te invitamos a reforzar tus conocimientos con la siguiente actividad.  
  
Instrucciones:  
  
Lee atentamente el siguiente enunciado y elige las opciones que consideres correctas.  
  
Enunciado:  
  
Los ciclos de actualización y desmontaje son los ciclos que, aunque muchos no implementan durante el trabajo con componentes, tienen una vital importancia en el ciclo de vida de éstos. A continuación encontrarás varias afirmaciones sobre estos ciclos, tu misión será encontrar las opciones que contienen afirmaciones verdaderas sobre el tema. ¡Adelante!



1. Lección 5: Routers & Complementos
   1. [React Router](https://learn.nextu.com/mod/lesson/view.php?id=3613&pid=P_WEBDEV_V2)

React-router es la librería JS que nos permite controlar las rutas a los diferentes componentes en nuestras aplicaciones ReactJS.

Las rutas son creadas con el fin de navegar en nuestras aplicaciones y pasar fácilmente de una vista a otra.

React-router es una librería que nos permite controlar y gestionar dichas rutas para cargar los componentes y/o secciones en aplicaciones reactJS, ya que este último no cuenta con dicha funcionalidad y/o característica.

Para instalar react-router:

Npm install react-router –save

Router – es el encargado de envolver o encapsular todas las rutas que vamos a definir dentro de nuestra aplicación

Routes – Representa cada una de las rutas en las que se mostrarán nuestros componentes, las cuales están definidas por dos atributos:

* + Path: se asigna la ruta en la cual podemos visualizar nuestro componente
  + Component: en este atributo se agrega el nombre que se mostrará cuando la ruta que se solicita por el navegador coincide con el atributo path que agregamos

Historias/ Histories:

React router esta construido a partir de historias, las cuales son las encargadas de estar atentas a escuchar la barra de navegación del navegador, para analizar y cambiar la URL según sea necesario.

En el caso de react router las historias son quienes nos ayudan a renderizar nuestros componentes según coincida la ruta que hay en el navegador con la que le asignamos a cada componente en el atributo path de la etiqueta route.

Existen 3 tipos de historias que se pueden encontrar frecuentemente, las cuales son:

* + BrowserHistory- es el tipo de historia de navegación más utilizada y recomendado para trabajar con react router, pues utiliza la API de historia integrada en el navegador para manipular la URL, acceder a ellas y crear URLs reales para nuestros componentes, en el siguiente formato:

<https://www.ejemplo.com/rutadelcomponente>

* + hashHistory- este tipo de historia de navegación no es tan usada como browserHistory, debido a que cuando se trabaja con renderización desde el lado del servidor en nuestras aplicaciones, este no lo soporta. HashHistory se caracteriza por utilizar el símbolo # en alguna parte de la URL. Ejemplo:

<https://www.ejemplo.com/#/rutadelcomponente>

* + createMemoryHistory – es mas utilizada con React Native

cualquiera de ellas nos permite construir una historia personalizada para trabajar con react router.

Uno de los componentes principales que incluyen react router es el Link, el cual usa una etiqueta ancla de HTML internamente y es usado para acceder a los diferentes componentes o rutas que necesitamos, este se compone principalmente por el atributo “to” en el cual se define la ruta que queremos con solo hacer click.

Nested Routes o Rutas Anidadas: Constan de encapsular o incluir elementos o etiquetas de route dentro de otras en los casos en donde se considere necesario

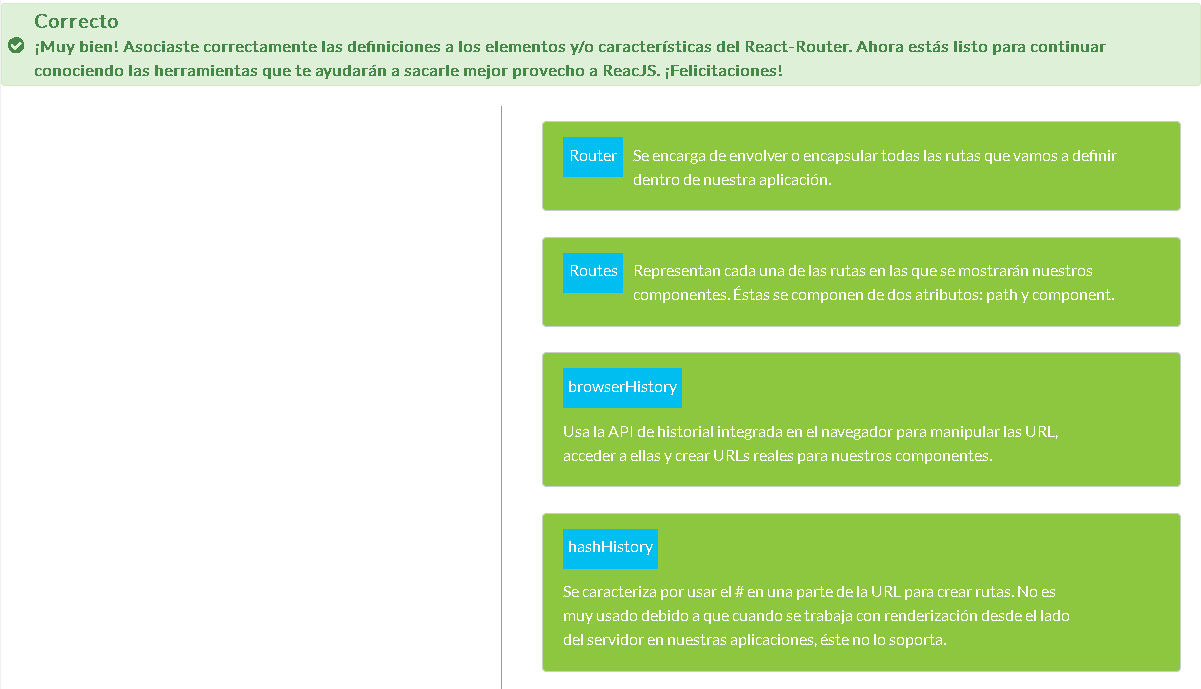
indexRoute: es el elemento de react router que nos permite definir el elemento que se renderizara por defecto una vez que iniciemos nuestra aplicación ReactJS y la abrimos en nuestro navegador, este elemento no cuenta con el elemento path y únicamente se encuentra el atributo component

<indexRoute component={nombreComponente}/>

* 1. Actividad 1

Tiempo de pensar

En esta oportunidad viste todo lo relacionado con React-router, la librería JavaScript que nos permite controlar las rutas a los diferentes componentes en nuestras aplicaciones ReactJS. Ahora es tiempo de que pongas a prueba tus conocimientos realizando la siguiente actividad  
  
Instrucciones:  
  
Lee atentamente el siguiente enunciado y realiza la asociación que consideres correcta.  
  
Enunciado:  
  
React-router es una librería que nos permite controlar y gestionar rutas para cargar los componentes y/o secciones en aplicaciones ReactJS, la cual cuenta con algunos elementos y/o características para su buen funcionamiento. Algunas de éstas las encontrarás listadas a continuación, tu misión es asociarles la definición que les corresponde. ¡Adelante!



* 1. Lectura: Super Agent - cargando y enviando datos al Back-end

**Superagent: cargando & enviando datos al back-end**



La mayoría de librerías y frameworks JavaScript utilizan AJAX como la herramienta para establecer comunicación entre el Front-end y el Back-end, con el fin de enviar y cargar datos desde nuestra base de datos. Pero cuando trabajamos con ReactJS nos encontramos con la que es quizás, una de sus limitaciones más notorias: su incompatibilidad o mala comunicación con otras librerías que tratan de controlar el DOM, como jQuery. Usar AJAX directamente no es lo más óptimo cuando estamos tratando de conectar el Back-end y el Front-end para nuestras aplicaciones React. Por lo que es necesario buscar una herramienta que sea eficiente, óptima, funcional, que permita conectar ambas partes de la aplicación y gestionar nuestros datos.  
Es allí donde aparece el **SuperAgent**.

**SUPERAGENT**

Esta es una librería de peticiones del lado del cliente que posee muchas características HTTP de alto nivel. SuperAgent es una API tipo AJAX diseñada para ofrecer eficiencia, flexibilidad, dinamismo, orden y un aprendizaje rápido. SuperAgent apareció como una solución fácil de usar después de que varios desarrolladores se vieran en algunos aprietos con otras librerías cuando pretendían establecer comunicación entre el Back-end y Front-end por cuestiones de compatibilidad entre librerías.  
  
La manera de incorporar SuperAgent en nuestros proyectos es instalarlo a través del comando npm de la siguiente manera:

npm install superagent --save

Una vez ejecutemos este comando y finalice el proceso de instalación, aparecerá el siguiente resultado en la línea de comandos:



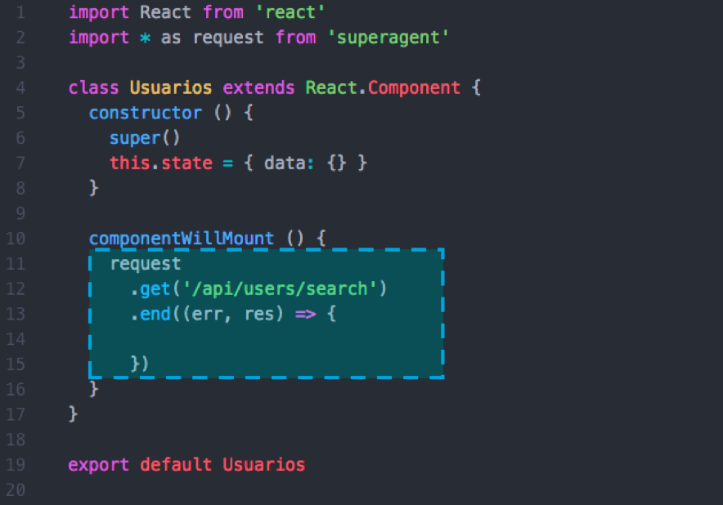
Uno de los temas más importantes que debemos tener presente al momento de trabajar con SuperAgent son los request o peticiones, que través de sus métodos nos permiten comunicar las dos partes de nuestra aplicación y obtener o enviar la información que necesitamos.  
Veamos un poco cómo trabajar con request o peticiones:

**REQUEST / PETICIONES**

Para trabajar apropiadamente con las peticiones o requests lo primero que debemos hacer es elegir e invocar el método apropiado para comunicar nuestro Back-end con nuestro Front-end.  
Adicionalmente, debemos saber qué obtendremos como resultado para saber cómo manejarlo, es decir, si obtenemos un objeto, un array, una ruta, etc. Finalmente, si estamos hablando de carga de información debemos tener claro cómo vamos a procesar dichos datos para entregarlos al usuario, o para almacenarlos en la base de datos, si estamos hablando de envío de información.  
  
Cuando queremos usar SuperAgent en nuestras aplicaciones ReactJS, lo primero que debemos hacer después de instalarlo es importarlo en el componente en que crearemos los métodos de comunicación con el Back-end y lo hacemos de la siguiente manera:

import \* as request from ‘superagent’

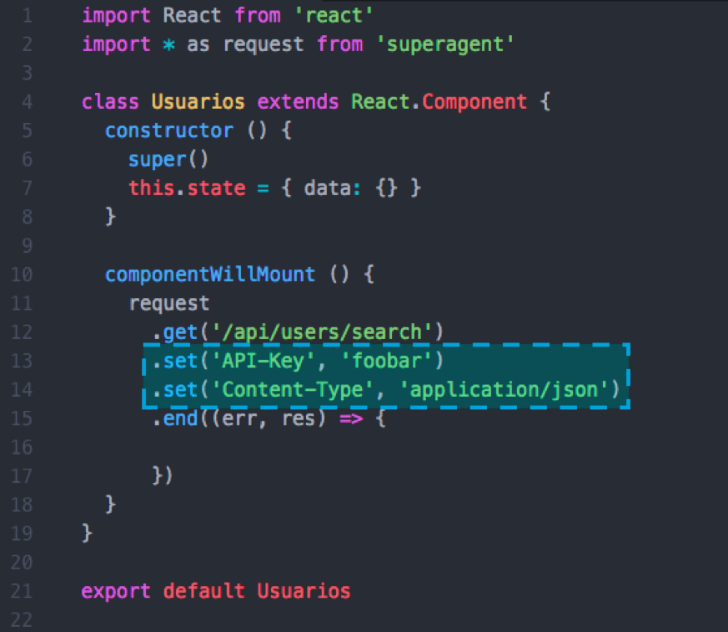
Así ya tendremos disponible todas las características que trae SuperAgent por defecto, de esta manera comenzamos a trabajar SuperAgent y ReactJS:



Cuando trabajamos con request o peticiones en SuperAgent, o en cualquier otra librería, tenemos disponibles una serie de campos y métodos fundamentales que nos permiten construir apropiadamente dichas peticiones, los principales son:

**HEADERS**

Los header o cabeceras son determinados campos que nos permiten determinar ciertas condiciones al momento de crear nuestra petición y que son fundamentales para obtener el resultado que esperamos en ésta. Dentro de dichas condiciones encontramos el tipo de contenido que retornará nuestra petición, si se requiere o no la autenticación para realizar nuestra aplicación, entre otras. Configurar y/o setear los campos en nuestras cabeceras es bastante simple, para ello invocamos .set() con un nombre para el campo y su respectivo valor, por ejemplo, **.set(‘nombreCampo’, ‘Valor’).** Veamos cómo se inician nuestras peticiones y cómo configurar los campos de los headers o cabeceras:



Como podemos ver, tenemos varios métodos set donde estamos pasando los headers dentro de un request, sin embargo, podemos pasar todos estos campos dentro de un solo .set(). Veamos cómo hacerlo:



**PETICIONES GET CON SUPERAGENT**

Para trabajar peticiones **GET** con SuperAgent, lo único que debemos hacer es agregar el método **.get()** debajo del request y dentro de los paréntesis agregar la ruta a la cual nos vamos a conectar para obtener información de nuestro Back-end. Veamos cómo hacerlo:



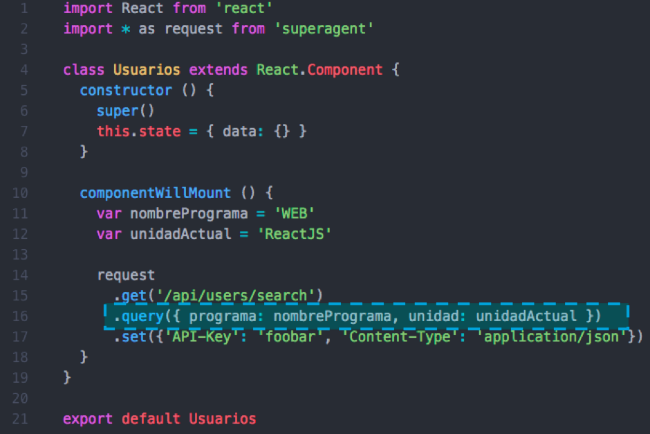
En las peticiones **GET** también podemos pasar parámetros que requiere el Back-end para realizar una consulta y obtener los datos en específico que se mostrarán en Front-end después de enviarlos en la respuesta de la petición. Veamos cómo se hace:



Esta es una forma de enviar parámetros, aunque hay una mucho más elegante y es haciendo uso del método .query(), el cual se encarga de recibir nuestros parámetros como objetos.  
Veamos cómo hacerlo:



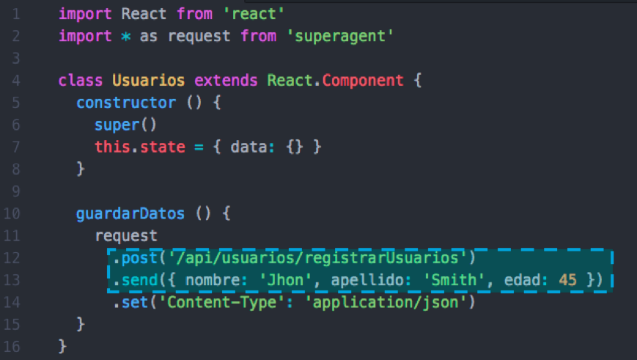
Finalmente, podemos decir que si tenemos múltiples parámetros para pasar en la petición GET, no es necesario tener múltiples métodos .query() sino que podemos hacerlo dentro de uno solo.   
Veamos cómo:



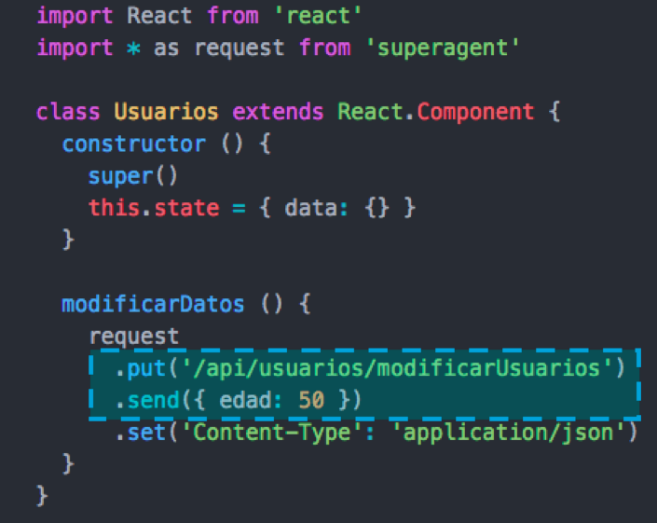
**PETICIONES POST / PUT**

Al igual que se hace con las peticiones GET, para hacer peticiones POST y PUT con SuperAgent, solo hace falta agregar el respectivo método debajo del objeto request con la respectiva ruta entre comillas y debajo de éste los parámetros con los datos que vamos a enviar al Back-end para almacenar en nuestra base de datos.  
Veamos cómo hacerlo:

**POST**



**PUT**

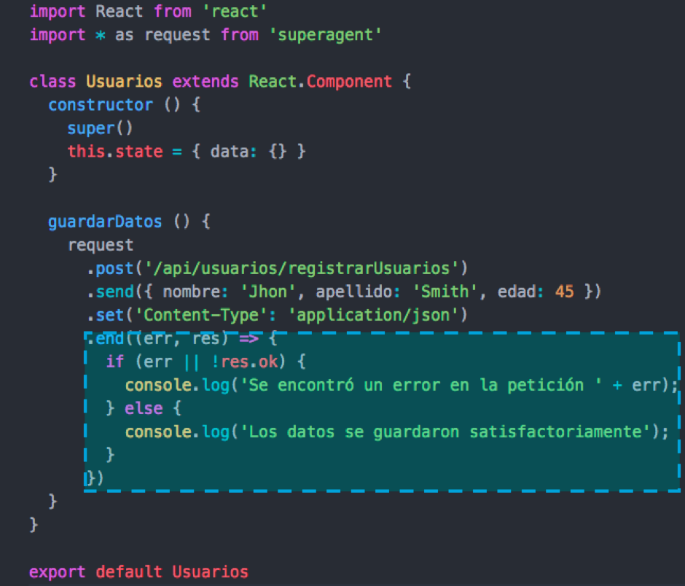


**END()**

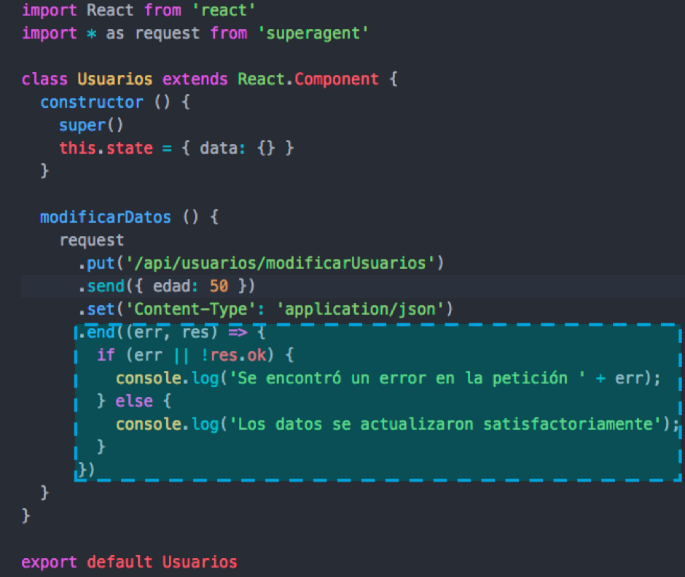
Finalmente llegamos al método en el que ya vamos a encontrar una respuesta y/o resultado al conectar el Front-end con el Back-end: el método .end().  
En éste creamos una función a la cual se le pasan dos parámetros, uno para el error y el otro para la respuesta, dentro de ésta ya procesamos los datos y notificaciones que se le mostrarán al usuario para que éste sepa qué sucedió al conectar el Back-end con el Front-end.  
Veamos cómo se hace:



**END() - PETICIONES GET**



**END() - PETICIONES POST**



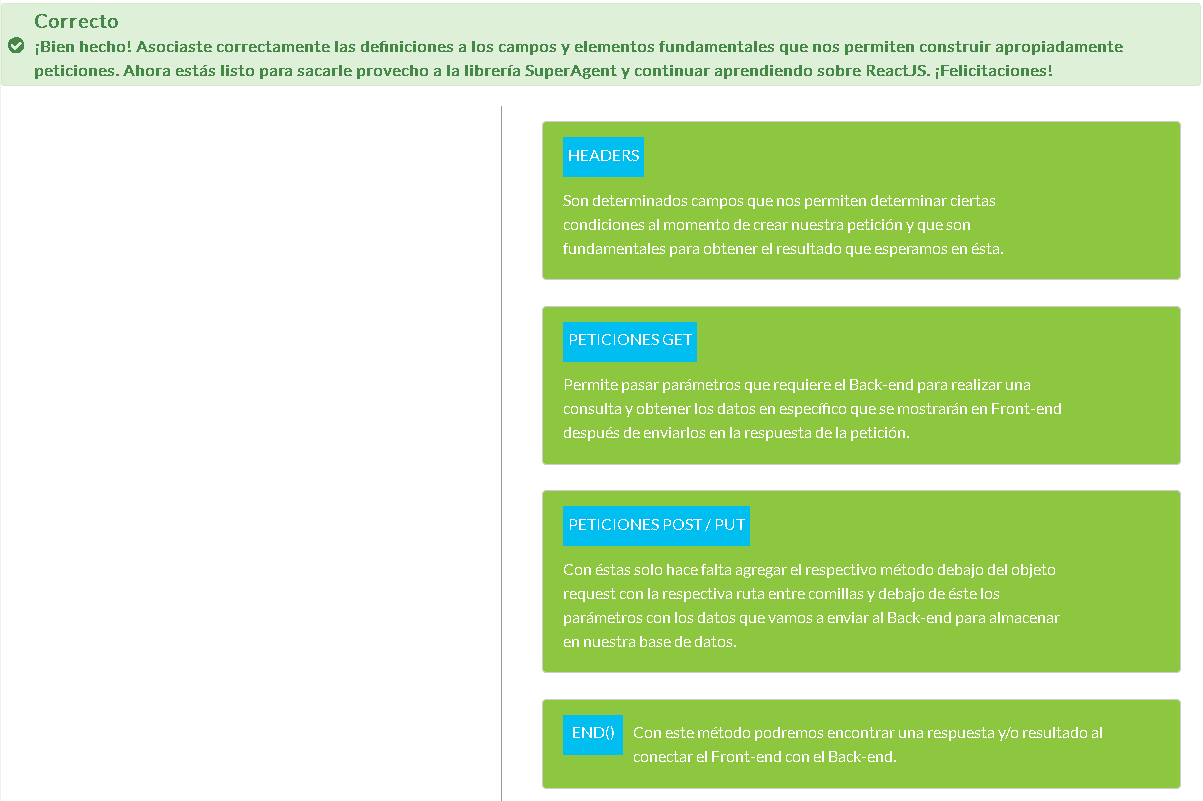
**END() - PETICIONES PUT**

En el método .end() debemos tener presente algo en el momento de crear la función interna de éste, ya que como puedes ver en las imágenes anteriores no la estamos representando con function sino con =>, debido a que cuando usamos function y dentro de la función usamos this encontraremos problemas de contexto, por tal motivo usamos => para evitar dichos problemas y que todo funcione adecuadamente.

* 1. Actividad 2

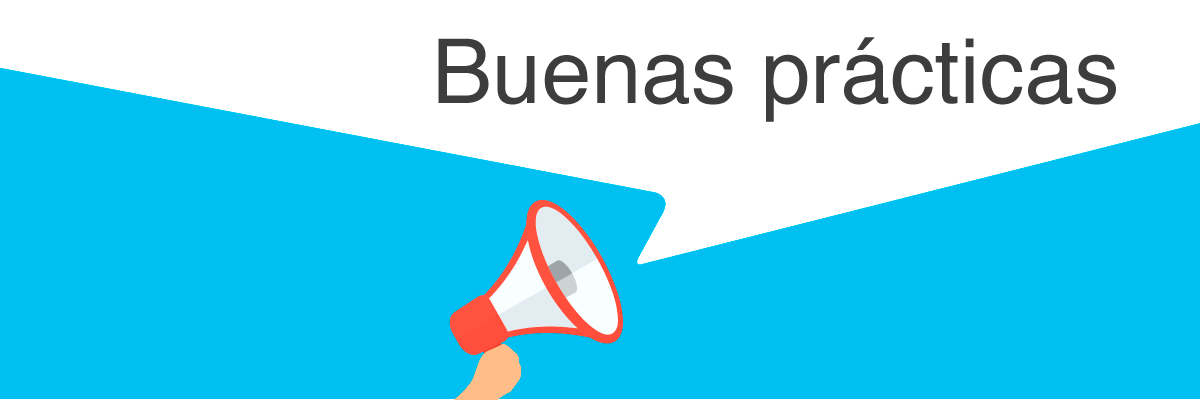
Tiempo de pensar

Ahora que conoces la librería SuperAgent que ha sido diseñada para ofrecer eficiencia, flexibilidad, dinamismo, orden y un aprendizaje rápido, te invitamos a reforzar tus conocimientos realizando la siguiente actividad  
  
Instrucciones:  
  
Lee atentamente el siguiente enunciado y realiza la asociación que consideres correcta.  
  
Enunciado:  
  
Como ya lo sabes, uno de los temas más importantes que debemos tener presente al momento de trabajar con SuperAgent son los request o peticiones, que través de sus métodos nos permiten comunicar las dos partes de nuestra aplicación y obtener o enviar la información que necesitamos. Al trabajar con éstos, tenemos disponibles una serie de campos y métodos fundamentales que nos permiten construir apropiadamente dichas peticiones, así que a continuación verás listados, tu misión será asociarles la definición que les corresponde. ¡Adelante!



* 1. Lectura: Buenas prácticas en ReactJS

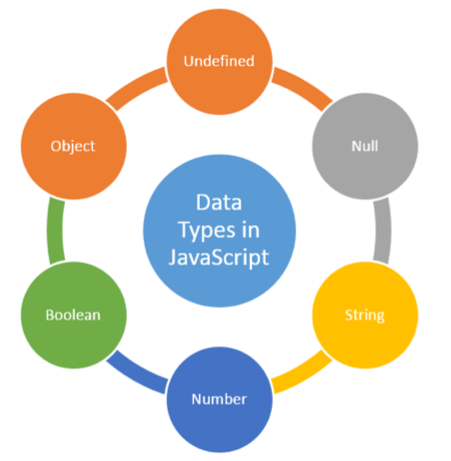
**Reactjs: buenas prácticas de programación**



Así como en todo lenguaje de programación se invita a los desarrolladores a hacer uso de buenas prácticas al momento de escribir el código, las librerías y frameworks que hacen parte de éstos también sugieren lo mismo, ya que hacen que los desarrolladores creen código ordenado, intuitivo, claro y coherente con respecto a la funcionalidad puntual de cada clase, función y/o método que se cree.  
  
ReactJS, al ser una librería JavaScript, sugiere la mayoría de reglas, principios y estándares de buena programación que tiene JavaScript, pero como librería tiene sus propias reglas para escribir código adecuadamente y las veremos a continuación:

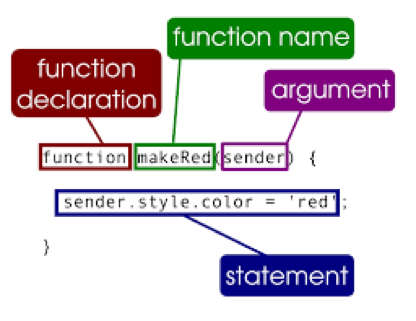
**Definición de variables**

En el momento en el que el programador va a definir una variable, debe tener siempre presente el tipo de datos que almacenará la variable, el nombre que le asignará y el tipo de variable, es decir, si es global o local, por lo que se considera una buena práctica de programación en el lenguaje JavaScript que los programadores en el momento de asignarle nombre a las variables, les asignen uno que tenga que ver con el valor que va a guardar y su tipo de dato. Adicionalmente, si se usara un nombre compuesto, se sugiere usar camelcase, iniciando la primera palabra en minúscula sin dejar espacio y la segunda palabra debe iniciar en mayúscula.



**Creación de funciones**

En el momento de crear funciones se sugiere que éstas tengan un nombre asociado con su funcionalidad, que se puedan diferenciar las variables globales de las locales, que no tengan muchas líneas de código, con el fin de prestarles un soporte que no sea muy complicado y que con solo verla el desarrollador pueda identificar qué hace, es decir, qué rol cumple determinada función dentro de nuestra aplicación.



**Validación de Props**

Una de las principales buenas prácticas de desarrollo de aplicaciones ReactJS es definir el tipo de prop que le está pasando el componente padre al componente hijo, es decir, definir el tipo de dato de éstas. Después de crear el componente hijo y cerrar la clase en la que éste se creó, validamos el tipo de props que le heredó su componente padre, es decir, si son propiedades de tipo numéricas, cadenas de texto, valores booleanos, objetos, arrays y/o funciones.   
  
Dicha validación la hacemos de la siguiente manera:



**Inicialización de estados**

Es vital que el proceso para inicializar estados se dé única y exclusivamente dentro del constructor, ya que éste es el primero de los métodos de la clase del componente que se ejecuta, su valor se almacena en memoria y quedan activos en el DOM virtual para cambios en funciones que se puedan encontrar más adelante. Adicionalmente, se recomienda que tengan un nombre relacionado con el valor que se almacena y un tipo de dato.

**Modificación de estados**

Esta operación solo tiene dos recomendaciones que al implementarlas ya nos llevan al uso de buenas prácticas de codificación. La primera de ellas es la más importante: nunca realizar una modificación de states o estados dentro del método Render, ya que esto ocasiona que se congele nuestro navegador debido a que el evento render se va a accionar cada vez que encuentra el cambio más minucioso en uno de los componentes. Evidentemente no va renderizar nuestra aplicación todo el tiempo, pero se está cambiando un valor de un estado dentro de este método cada vez que se intenta renderizar y/o actualizar un componente. La segunda sugerencia es más simple: evitar la modificación innecesaria y reiterativa de un solo estado, ya que no tiene sentido cambiar el valor de un estado en todas las funciones, es necesario analizar dónde y cuándo se requiere el cambio, así como a qué está sujeto, es decir, de qué depende que éste cambie.

* 1. Actividad 3

Tiempo de pensar

Ahora que conoces las buenas prácticas de ReactJS, te invitamos a validar tus conocimientos realizando la siguiente actividad  
  
Instrucciones:  
  
Lee atentamente el siguiente enunciado y realiza la asociación que consideres correcta.  
  
Enunciado:  
  
Como en todo lenguaje de programación se invita a los desarrolladores a hacer uso de buenas prácticas al momento de escribir el código, así mismo React nos dice cuáles son sus propias reglas para escribir código adecuadamente. A continuación encontrarás algunas de ellas listadas, tu misión será asociarles la definición que les corresponde. ¡Adelante!



1. UNIDAD 4: Prueba

|  |  |
| --- | --- |
| **Comenzado en** | Wednesday, 30 de December de 2020, 13:30 |
| **Estado** | Terminados |
| **Finalizado en** | Wednesday, 30 de December de 2020, 13:37 |
| **Tiempo empleado** | 6 mins 46 segundos |
| **Puntos** | 8/10 |
| **Calificación** | **83** de un total de 100 |
| **Comentario de retroalimentación** | Buen trabajo! |

Top of Form

### Pregunta 1

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

Selecciona las opciones que contienen características sobre JSX

Seleccione una o más de una:

a. Es una extensión de JavaScript, creado por Facebook, para el uso con la librería React.

b. Es un preprocesador que transforma el código de JavaScript.

c. Es una mezcla de código HTML con los ficheros JavaScript.

d. Es la manera más práctica de escribir HTML dentro de JavaScript.

#### Retroalimentación

JSX es una extensión de JavaScript, creado por Facebook, para el uso con la librería React. También es un preprocesador que transforma el código de JavaScript, así que aunque parece que estuvieras escribiendo en HTML, en realidad es JavaScript.

Las respuestas correctas son: Es una extensión de JavaScript, creado por Facebook, para el uso con la librería React., Es un preprocesador que transforma el código de JavaScript.

### Pregunta 2

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

Lee el enunciado y elige la opción que lo completa.

Configurar los campos en nuestras cabeceras es bastante simple, para ello invocamos \_\_\_\_\_\_ con un nombre para el campo y su respectivo valor.

Seleccione una:

a. .query()

b. .end()

c. .get()

d. .set()

#### Retroalimentación

Los header o cabeceras son determinados campos que nos permiten determinar ciertas condiciones al momento de crear nuestra petición y que son fundamentales para obtener el resultado que esperamos en ésta. Configurar y/o setear los campos en nuestras cabeceras es bastante simple, para ello invocamos .set() con un nombre para el campo y su respectivo valor

La respuesta correcta es: .set()

### Pregunta 3

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

Selecciona la opción a la que hace referencia el siguiente enunciado:

Este es uno de los métodos más usados en React, ya que a través de él podemos actualizar el estado de un componente.

Seleccione una:

a. findDOMNode

b. forceUpdate

c. updateState

d. setState

#### Retroalimentación

En el API de componentes, encontramos 3 métodos bastante interesantes que nos permiten actualizar los atributos como los estados y las características visuales de nuestros componentes, siempre y cuando dichos atributos o características no sean las props o propiedades pues éstas son inmutables. El método más usado en React es el setState, pues es a través de él que podemos actualizar el estado de un componente.

La respuesta correcta es: setState

### Pregunta 4

Parcialmente correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

Selecciona las opciones del listado que contienen características sobre el bundle.

Seleccione una o más de una:

a. Es un archivo JavaScript que incluye todos los recursos que requiere la aplicación para funcionar adecuadamente.

b. En él podemos encontrar archivos JavaScript, CSS, Imágenes, LESS, SASS, entre otros.

c. En él se reúnen todos los recursos o assets y éste los organiza de tal manera que sus importaciones no se vean afectadas, manteniendo el correcto funcionamiento de la aplicación.

d. Es un paquete local y externo que debemos importar.

#### Retroalimentación

Un bundle, o paquete, es un archivo JavaScript que incluye todos los recursos que requiere la aplicación para funcionar adecuadamente. Dentro de este paquete podemos encontrar archivos JavaScript, CSS, Imágenes, LESS, SASS, entre otros.

Las respuestas correctas son: Es un archivo JavaScript que incluye todos los recursos que requiere la aplicación para funcionar adecuadamente., En él podemos encontrar archivos JavaScript, CSS, Imágenes, LESS, SASS, entre otros., En él se reúnen todos los recursos o assets y éste los organiza de tal manera que sus importaciones no se vean afectadas, manteniendo el correcto funcionamiento de la aplicación.

### Pregunta 5

Incorrecta

Puntúa 0 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

ReactJS nos ofrece muchas características para trabajar, lee el enunciado con atención y determina cuál es la opción que contiene la característica de la que éste habla:

React tiene su propio DOM virtual, donde cada uno de los componentes de nuestra aplicación está activo, lo que hace que el rendimiento de la aplicación sea más flexible y amplio.

Seleccione una:

a. WebComponents

b. Eficiencia

c. Crecimiento continuo

d. Compatibilidad con los navegadores

#### Retroalimentación

El hecho de que React tiene su propio DOM virtual hace que el rendimiento de la aplicación sea más flexible y amplio, ya que React puede saber qué componente cambió y en qué momento lo hizo sin actualizar toda la aplicación sino solo esa parte que sufrió cambio, lo que hace que el usuario no tenga tiempo de espera mientras se recarga nuevamente la aplicación.

La respuesta correcta es: Eficiencia

### Pregunta 6

Parcialmente correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

Selecciona las opciones que contienen características sobre el SuperAgent

Seleccione una o más de una:

a. Es una librería de peticiones del lado del cliente que posee muchas características HTTP de alto nivel.

b. Es una API tipo AJAX diseñada para ofrecer eficiencia, flexibilidad, dinamismo, orden y un aprendizaje rápido.

c. Después de instalar SuperAgent, lo primero que debemos hacer es importarlo en el componente en que crearemos los métodos de comunicación con el Back-end.

d. Es una herramienta que no permite establecer comunicación entre el Back-end y Front-end de la aplicación ni gestionar nuestros datos.

#### Retroalimentación

SuperAgent es una librería de peticiones del lado del cliente que posee muchas características HTTP de alto nivel. Además, es una API tipo AJAX diseñada para ofrecer eficiencia, flexibilidad, dinamismo, orden y un aprendizaje rápido. SuperAgent apareció como una solución fácil de usar después de que varios desarrolladores se vieran en algunos aprietos con otras librerías cuando pretendían establecer comunicación entre el Back-end y Front-end por cuestiones de compatibilidad entre librerías.

Las respuestas correctas son: Es una librería de peticiones del lado del cliente que posee muchas características HTTP de alto nivel., Es una API tipo AJAX diseñada para ofrecer eficiencia, flexibilidad, dinamismo, orden y un aprendizaje rápido., Después de instalar SuperAgent, lo primero que debemos hacer es importarlo en el componente en que crearemos los métodos de comunicación con el Back-end.

### Pregunta 7

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

Selecciona las opciones que contienen características sobre los Props y States

Seleccione una o más de una:

a. Los props son características y/o datos de los componentes, éstos no pueden ser alterados o modificados.

b. La característica principal de los props es que no pueden ser pasados de un componente padre a sus componentes hijos

c. Los states nos permiten crear componentes interactivos y dinámicos basándose en su comportamiento.

d. Los states o estados no pueden ser alterados o modificados.

#### Retroalimentación

La mayoría de los componentes creados con ReactJS cuentan con una parte central que se encarga de determinar su comportamiento y la manera en la que éste se debe renderizar. A dicha parte se le conoce como States o Estado, mientras que los props, o propiedades, podemos definirlos como la mitad de lo que hace especial trabajar ReactJS con componentes, éstos son para los componentes ReactJS lo mismo que los atributos a los elementos o etiquetas HTML.

Las respuestas correctas son: Los props son características y/o datos de los componentes, éstos no pueden ser alterados o modificados., Los states nos permiten crear componentes interactivos y dinámicos basándose en su comportamiento.

### Pregunta 8

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

Lee el enunciado y elige la opción que lo completa.

El método \_\_\_\_\_\_\_\_\_\_\_\_ es usado para forzar la actualización del componente en general en cualquier momento.

Seleccione una:

a. findDOMNode

b. forceUpdate

c. ReactDOM

d. setState

#### Retroalimentación

Este método, a diferencia del método setState, no es usado para actualizar el estado del componente sino que es usado para forzar la actualización del componente en general en cualquier momento.

La respuesta correcta es: forceUpdate

### Pregunta 9

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

De las opciones que se listan a continuación, elige aquella que no pertenece a las ventajas de ReactJS.

Seleccione una:

a. Es una herramienta fácil de aprender, ya que cuenta con una documentación y una comunidad de desarrolladores muy amplia que siempre está abierta a resolver dudas o inquietudes.

b. Al trabajar con componentes independientes, la creación y el mantenimiento de nuestras aplicaciones se convierte en una tarea muy sencilla y rápida.

c. Si se tiene un volumen de datos complejo y bastante amplio, en algunas ocasiones el tiempo de renderizado puede tomar más tiempo de lo esperado.

d. ReactJS es de las pocas librerías JavaScript que se integra y soporta versiones antiguas de Internet Explorer.

#### Retroalimentación

Si se tiene un volumen de datos complejo y bastante amplio, en algunas ocasiones el tiempo de renderizado puede tomar más tiempo de lo esperado, lo que es una desventaja. Sin embargo, una de sus ventajas también es que renderiza rápidamente cada uno de los componentes de la aplicación, lo que permite brindar una mejor experiencia de usuario.

La respuesta correcta es: Si se tiene un volumen de datos complejo y bastante amplio, en algunas ocasiones el tiempo de renderizado puede tomar más tiempo de lo esperado.

### Pregunta 10

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

Lee el enunciado y elige la opción que lo completa.

Es importante tener presente que, si quisiéramos hacer uso de Babel, necesitamos tener instalado \_\_\_\_\_\_\_\_\_\_\_\_\_.

Seleccione una:

a. Node.js

b. npm

c. React.js

d. React Native

#### Retroalimentación

Para usar Babel debemos tener instalado Node.js pues está escrito en JavaScript.

La respuesta correcta es: Node.js

Bottom of Form

**Conclusión del curso**

¡Felicitaciones!

Has llegado al final del Curso: **Herramientas, Build Systems y tecnologías emergentes**

Has realizado un recorrido por las temáticas avanzadas de desarrollo web: aprendiste las herramientas para tareas conocidas como task runners, diversas técnicas a través de varias herramientas para la gestión de tareas en desarrollo, profundizaste en SPA y dos de los frameworks más populares en la actualidad web: Angular 2 y React Js. Con la última versión actualizada de Angular y los conceptos a la vanguardia de React, este curso encierra un completo ambiente de desarrollo en un nivel muy superior al desarrollo web convencional.

Gracias a este curso y lo que con esfuerzo y dedicación practicaste con los ejercicios, estás en capacidad de reconocer y aplicar conceptos e instalar y configurar el espacio de trabajo, considerando que:

* **Unidad 1: Automatización de tareas**
  + La automatización nos permite ejecutar tareas que por lo general están escritas como plugins.
  + Las tareas más usadas en el desarrollo en Javascript son: Minify, Linting, Transpiling, Licensing and Versioning, Sourcemaps, Rename, delete, create y map functions through files, Documentación, Ejecución y cubrimiento de pruebas, Uso automático de repositorios, Servir aplicaciones, Descargar URL`s, Watchers, Livereload, Procesamiento de imágenes, Servir y subir archivos vía FTP / SSH, Renderizar componentes web en el servidor, Construir múltiples versiones de nuestras aplicaciones, Enviar emails y Gestionar archivos estáticos en CDN’s.
  + Gulp es una herramienta en forma de script, escrita en Nodejs, que ayuda a automatizar tareas comunes en el desarrollo de aplicaciones, lo cual facilita el proceso de desarrollo, gastando menos tiempo en el proceso de escribir código, realizar validaciones y pruebas de nuestro código.
  + Las funciones que trae la librería Gulp y que facilitan la programación son: src(‘url-file’ o [‘url-file’,’url-file’]), dest(‘’), task(‘’, [‘’], fn) y watch(‘url-files or url-file’, opts, [task]/fn).
  + Grunt es una herramienta que nos facilita el desarrollo de aplicaciones web pues optimiza nuestro tiempo mientras realizamos tareas complejas.
  + Existen muchas herramientas para la automatización de tareas pero la principal de todas viene siendo el mismo manejador de paquetes npm de nodejs.
* **Unidad 2: Single Page Apps**
  + El término SPA (Single Page Application) es usado exclusivamente para hacer referencia a las aplicaciones Web, las cuales contienen un único documento HTML y un alto nivel de interacción.
  + El funcionamiento de una SPA está basado principalmente en JavaScript, pero en realidad se necesita más que eso para su correcto funcionamiento, en caso de no usar alguno de los frameworks actuales como Angular, o una librería como React.js, la implementación de una SPA se basa en las siguientes técnicas: AJAX, WebSockets, Arquitecturas de servidor (API REST en la mayoría de los casos) y librerías para el manejo de las rutas del lado del cliente (Page.js).
  + Page.js es una librería que sirve de enrutador en el Front-end. Creada y mantenida por visionmedia, esta librería es muy liviana y fácil de usar, con la cual se realiza enrutamiento de las vistas en JavaScript.
  + Una SPA se caracteriza por eliminar la navegación entre varios archivos HTML para acceder a cada una de las vistas del sistema.
  + Es importante conocer la forma en la que se establece una comunicación entre SPA y Back-end a medida que el usuario navega por la aplicación y requiere enviar o recibir datos después de la petición inicial mencionada.
* **Unidad 3: Angular 2**
  + Implementar MongoDB, una base de datos NoSQL orientada a documentos, y sus servicios (Mongod, Mongo, MongoImport, MongoExport).
  + Angular 2 es un framework JavaScript de código abierto promovido por Google y la comunidad de desarrollo libre. Enfocado en la filosofía mobile-first, es uno de los frameworks para el desarrollo de SPAs con mayor renombre en el mundo.
  + Tiene dos ventajas: su rapidez y desempeño y que es multiplataforma, es decir que posible por medio de la integración con otras herramientas y frameworks, crear aplicaciones móviles nativas o híbridas, así como aplicaciones de escritorio en múltiples sistemas operativos.
  + Databinding puede traducirse como enlace de datos y hace referencia a la comunicación que existe entre la vista y la lógica en una aplicación, es decir, la manera en la que se pasan datos desde las clases TypeScript a las plantillas HTML de los componentes.
  + Angular 2 es un framework que permite estructurar aplicaciones Web tipo SPA y trabaja por medio de la filosofía de componentes. Como toda aplicación debe estar en constante comunicación con un Back-end que contenga la lógica del negocio.
  + Para crear una comunicación adecuada entre una API, o ya bien cualquier tipo de servicio de Back-end, Angular usa el módulo Http, el cual incluye una gran cantidad de herramientas y utilidades que permiten hacer peticiones, recibir respuestas y gestionar dichas comunicaciones.
* **Unidad 4: ReactJS**
  + ReactJS es una de las librerías Front-end de JavaScript más potentes que existe actualmente y que cuenta con una comunidad de desarrolladores que hacen que cada vez mejore más y tenga nuevas características para ofrecerle a quienes lo usan en la creación de sus aplicaciones Web.
  + ReactJS es de las pocas librerías JavaScript que se integra y soporta versiones antiguas de Internet Explorer.
  + React, al tener un DOM virtual que se encarga de actualizar el estado y el contenido de cada componente de manera individual, no se lleva bien con librerías como jQuery que buscan tener una manipulación total del DOM.
  + Babel es una herramienta que nos permitirá implementar las características de JavaScript ECMAScript 6-ES6, convirtiendo el código en ES5 para que sea leído por cualquier navegador.
  + Webpack se define como un bundler, o sistema de empaquetado de módulos, que permite compactar los recursos o assets de una aplicación, mejorando el desempeño en la carga inicial de la misma desde el cliente.
  + JSX es una extensión de JavaScript, un lenguaje creado por Facebook, para el uso con la librería React.
  + El DOM Virtual puede ser similar a una tabla con datos, en una página HTML, que en cada fila tiene un botón de edición para cambiar, de forma rápida, los datos.