

**FACULTAD DE CIENCIAS E INGENIERÍA**

**ESCUELA PROFESIONAL DE INGENIERÍA**

**DE SISTEMAS E INFORMÁTICA**

**PROYECTO DE INVESTIGACIÓN**

REACT.JS

**PRESENTADO POR:**

Chanamoth Oversluijs Hillary

Soto Soto Yohn

Valverde Valverde Frank

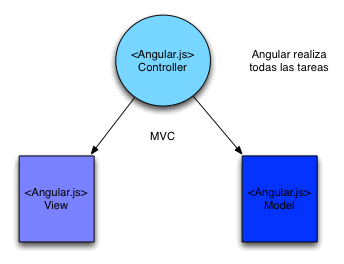
Soriano Miguel

**LIMA – PERÚ**

**2017**

INTRODUCCIÓN

El presente trabajo tiene como tema principal explicar la librería React.js, una herramienta estupenda para el desarrollo de interfaces de usuario y aplicaciones Javascript del lado del cliente. React un framework creado por los ingenieros de Facebook que nos ayudara a crear aplicaciones SPA (single page application), centrándose únicamente en vistas. [React.js](https://facebook.github.io/react/) es una librería de JavaScript que se encuentra en un momento de ebullición. El año pasado ocurrió con[Angular.js](https://angularjs.org/) y este año le toca a React.js.



¿QUÉ ES REACT Y POR QUÉ INTERESA USARLO?

Qué es React y los motivos por los que es una estupenda alternativa para el desarrollo de interfaces de usuario o aplicaciones web del lado del cliente.

React es una librería Javascript focalizada en el desarrollo de interfaces de usuario. Esa es su principal área de trabajo, pero lo cierto es que con todo el ecosistema de aplicaciones y herramientas y componentes, con React encontramos un excelente aliado para hacer todo tipo de aplicaciones web, SPA (Single Page Application) o incluso aplicaciones para móviles.

Es por tanto una base sobre la cual se puede construir casi cualquier cosa con Javascript y que nos facilita mucho el desarrollo, ya que nos ofrece muchas cosas ya listas, en las que no necesitamos invertir tiempo para desarrollar. En este artículo te ampliaremos esta información, aportando además diversos motivos por los que elegir React como librería del lado del cliente.

HISTORIA

React es una librería que proviene de Facebook. Es software libre y a partir de su liberación una creciente comunidad de desarrolladores la está usando. Se crea en base a unas necesidades, generadas por el propio desarrollo de la web de la popular red social. En Facebook necesitaban herramientas para un desarrollo rápido pero focalizado en un mayor rendimiento que otras alternativas existentes en el mercado. Detectaron que el típico marco de binding y doble binding ralentizaba un poco su aplicación, debido a la cantidad de conexiones entre las vistas y los datos. Como respuesta crearon una nueva dinámica de funcionamiento, en la que optimizaron la forma como las vistas se rende rizaban frente al cambio en los datos de la aplicación. A partir de ahí la probaron en su red social con resultados positivos y luego en Instagram, también propiedad de Facebook, con éxito. Más adelante, alentados por los positivos resultados en el rendimiento de React, muchas otras aplicaciones web de primer nivel la fueron adoptando. BBC, Airbnb, Netflix, Dropbox y un largo etc.

OBJETIVO

Sirve para desarrollar aplicaciones web de una manera más ordenada y con menos código que si usas Javascript puro o librerías como jQuery centradas en la manipulación del DOM. Permite que las vistas se asocien con los datos, de modo que si cambian los datos, también cambian las vistas.

El código spaguetti que se suele producir mediante librerías como jQuery se pretende arquitecturizar y el modo de conseguirlo es a través de componentes. Una interfaz de usuario es básicamente creada a partir de un componente, el cual encapsula el funcionamiento y la presentación. Unos compoentes se basan además en otros para solucionar necesidades más complejas en aplicaciones. También permite crear otras piezas de aplicación cómodamente, como los test.

COMPARACION DE REACT CON OTRAS LIBRERIAS O FRAMEWORKS

React aporta una serie de posibilidades muy importante. Al tener las vistas asociadas a los datos, no necesitamos escribir código para manipular la página cuando los datos cambian. Esta parte en librerías sencillas es muy laboriosa de conseguir y es algo que React hace automáticamente.

También en comparación con jQuery nos permite una arquitectura de desarrollo más avanzada, con diversos beneficios como la encapsulación del código en componentes, que nos ofrecen una serie de ventajas más importantes que los plugin, como la posibilidad de que esos componentes conversen e interaccionen entre si, algo que que sería muy difícil de conseguir con Plugins.

ReactJS solapa por completo las funcionalidades de jQuery, por lo que resulta una evolución natural para todos los sitios que usan esa librería. Podrían convivir pero no es algo que realmente sea necesario y recargaría un poco la página, por lo que tampoco sería muy recomendable.

Ya luego en comparación con frameworks como es el caso de Angular o Ember, React se queda a mitad de camino. Pero ojo, a partir de todo el ecosistema de React se llega más o menos a las mismas funcionalidades, así que es una alternativa perfecta. muchos desarrolladores lo entienden como una ventaja con respecto a frameworks completos, ya que tú puedes desarrollar con React a tu gusto, aplicando aquellas herramientas y librerías adicionales que hacen las cosas como mejor se adapte al proyecto.

CARACTERÍSTICAS DE REACT

Composición de componentes

Así como en programación funcional se pasan funciones como parámetros para resolver problemas más complejos, creando lo que se conoce como composición funcional, en ReactJS podemos aplicar este mismo patrón mediante la composición de componentes

Las aplicaciones se realizan con la composición de varios componentes. Estos componentes encapsulan un comportamiento, una vista y un estado.

Flujo de datos unidireccional

Ésta es otra de las cosas que facilita React, aunque no es exclusivo. En este modelo de funcionamiento, los componentes de orden superior propagan datos a los componentes de orden inferior. Los de orden inferior trabajarán con esos datos y cuando cambia su estado podrán propagar eventos hacia los componentes de orden superior para actualizar sus estados.

Performance gracias al DOM Virtual

El desempeño de React es muy alto, gracias a su funcionamiento. Nos referimos al desempeño a la hora del renderizado de la aplicación. Esto se consigue por medio del DOM Virtual. No es que React no opere con el DOM real del navegador, pero sus operaciones las realiza antes sobre el DOM Virtual, que es mucho más rápido. El DOM Virtual está cargado en memoria y gracias a la herramienta que diferenciación entre él y el real, el DOM del navegador se actualiza. El resultado es que estas operaciones permiten actualizaciones de hasta 60 frames por segundo, lo que producen aplicaciones muy fluidas, con movimientos suavizados.

EL ESTADO EN LOS COMPONENTES REACT

Qué son los componentes, con estado y sin estado, en React y cómo gestionar el estado de los componentes adecuadamente. En este artículo vamos a adentrarnos un poco más en el desarrollo de componentes React, dedicando nuestro estudio al estado de los componentes. Existen dos tipos de componentes, con estado (statefull components) y sin estado (stateless components). Los veremos en contraposición.

Todavía no hemos visto demasiados ejemplos, así que aprovecharemos también para ver algunos otros y así ir familiarizándonos con el desarrollo usando esta librería Javascript.

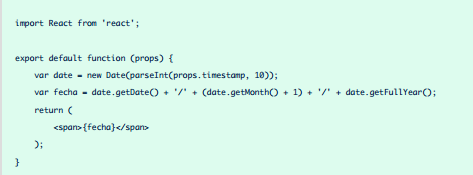
COMPONENTES CON ESTADO / SIN ESTADO

Para comenzar queremos analizar estos dos conceptos en contraposición. Cuando empezamos con React pueden provocar dudas, pero realmente es sencillo. Básicamente, la diferencia entre componentes con o sin estado estriba en que los componentes con estado permiten mantener datos propios a lo largo del tiempo e implementar comportamientos en sus diferentes métodos del ciclo de vida.

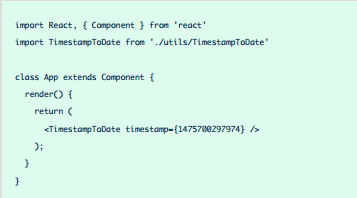
Los componentes sin estado no guardan ninguna información y por ello no necesitan de datos locales. Todos los componentes implementados hasta el momento eran stateless, sin estado. Eso no significa que no puedan personalizar su comportamiento, lo que se consigue con la ayuda de las propiedades de los componentes. Estas propiedades que nos pasan se podrán incluso transformar al producir una salida, de modo que sea acorde con las necesidades, pero no se guardará ningún valor y el componente no tendrá un ciclo de vida.

EJEMPLO DE COMPONENTE SIN ESTADO

Para que quede más claro, esto sería una implementación de componente sin estado que recibe un timestamp como propiedad y genera una vista donde ese instante se muestra convertido en una fecha en español.

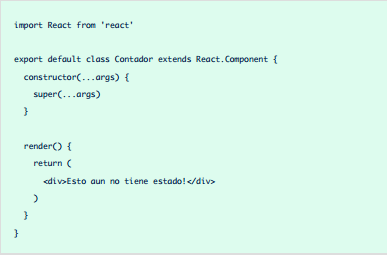


Este componente se podría usar importándolo y luego colocando la etiqueta correspondiente. Me parece interesante mostrar cómo se usaría porque el componente en sí, tal como se ha definido, no tiene nombre. El nombre se lo asignamos a la hora de importarlo.

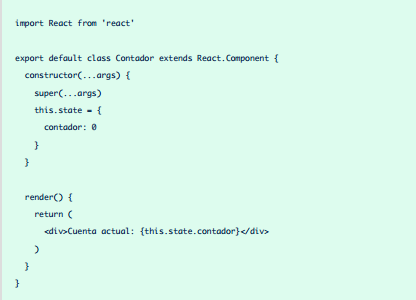


EJEMPLO DE COMPONENTE CON ESTADO

Seguro que estarás deseando ver ya un componente con estado. Enseguida nos ponemos con ello, pero quiero que veamos antes el código de "boilerplate" para crear un componente con estado.



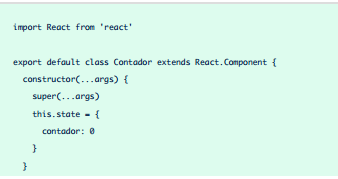
En nuestro ejemplo vamos a implementar el típico del contador. El valor actual del contador será nuestro estado y lo tendremos que inicializar en el constructor.

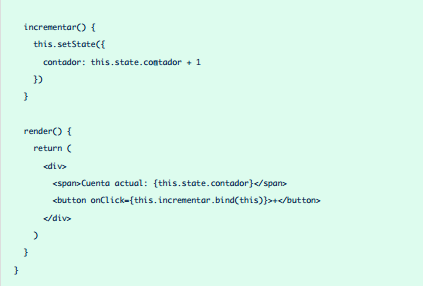


Ahora nuestro constructor ya tiene sentido, porque está realizando la inicialización de la propiedad "state" del componente. Como puedes ver el estado es un objeto, en el que ponemos tantos atributos como sea necesarios guardar como estado.

A la hora de renderizar el componente, por supuesto, podremos usar el estado para mostrar la salida. En este caso puedes ver cómo se vuelca el estado en la vista, con la expresión {this.state.contador}. Algo muy parecido a lo que hacíamos con las propiedades, solo que ahora los datos nos llegan del estado.

Solo nos falta implementar un botón para incrementar ese contador para ello tenemos que entrar en un tema nuevo, que son los eventos. Veamos el siguiente código

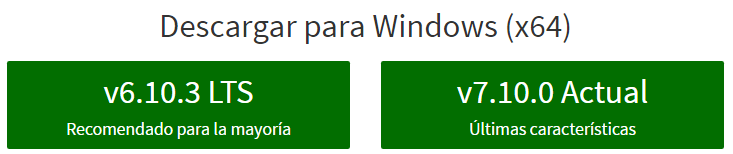




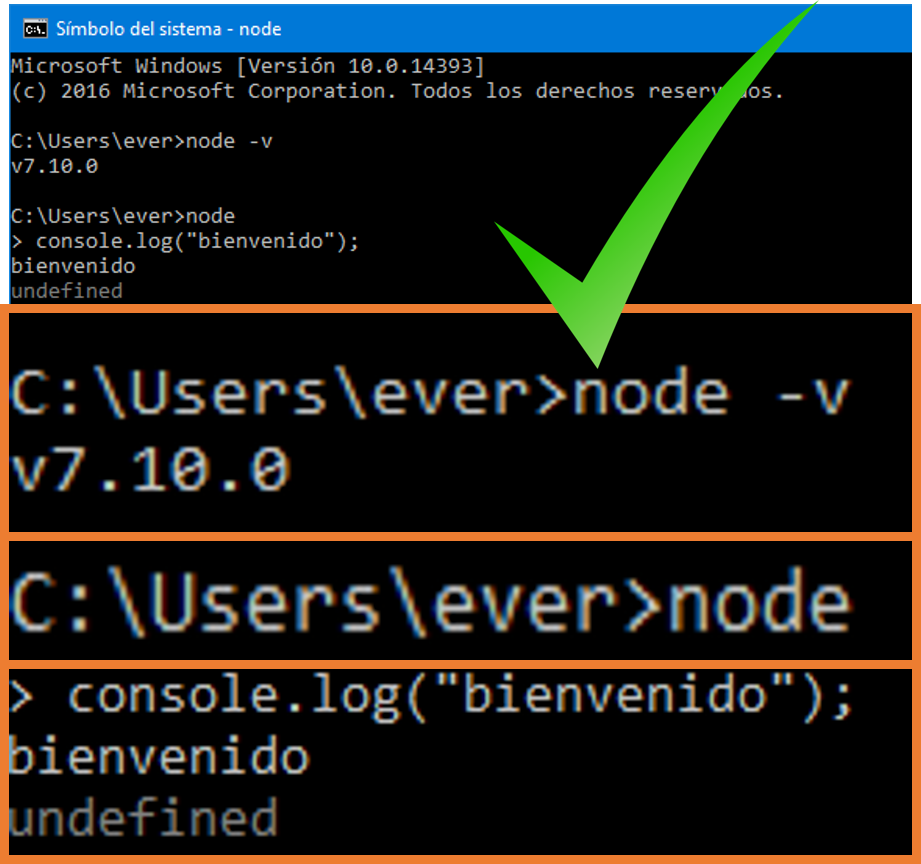
INSTALACION: PASO N°1 INSTALACIÓN DE NODE.JS

Node.js es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome. Node.js usa un modelo de operaciones E/S sin bloqueo y orientado a eventos, que lo hace liviano y eficiente. El ecosistema de paquetes de Node.js, npm, es el ecosistema más grande de librerías de código abierto en el mundo.





INSTALACION: PASO N°2 CONSOLA DE COMANDOS



**1\_**Abrimos la consola de node.

cmd

**2\_**Verificamos la versión de node.

node -v

**3\_**Ingresamos a node.

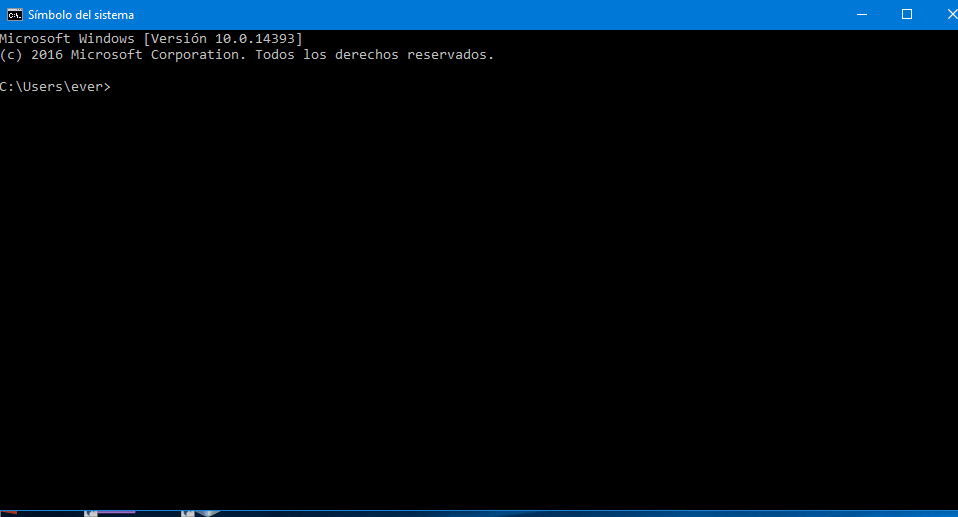
node

**4\_**Probamos los comandos de node.

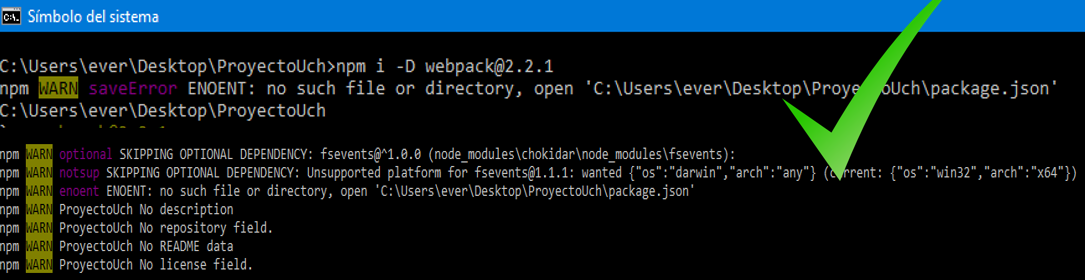
console.log(“Bienvenido UCH”);



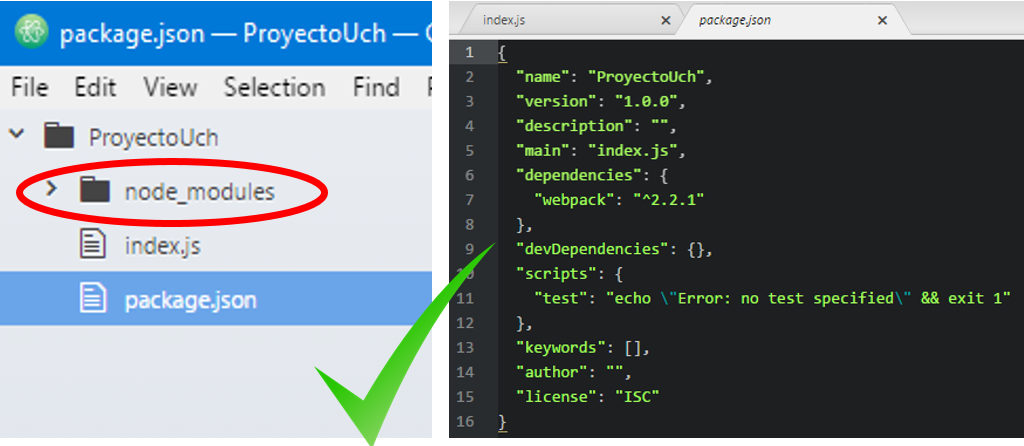
INSTALACION: WEBPACK

**Paso 1**: Ingresamos a la consola de comandos  
**cmd**

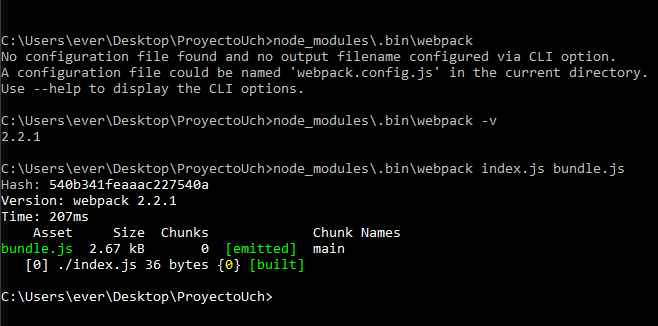
**Paso 2:** Ubicamos la ruta de nuestro proyecto  
y escribimos el siguiente código:  
**npm init -y**

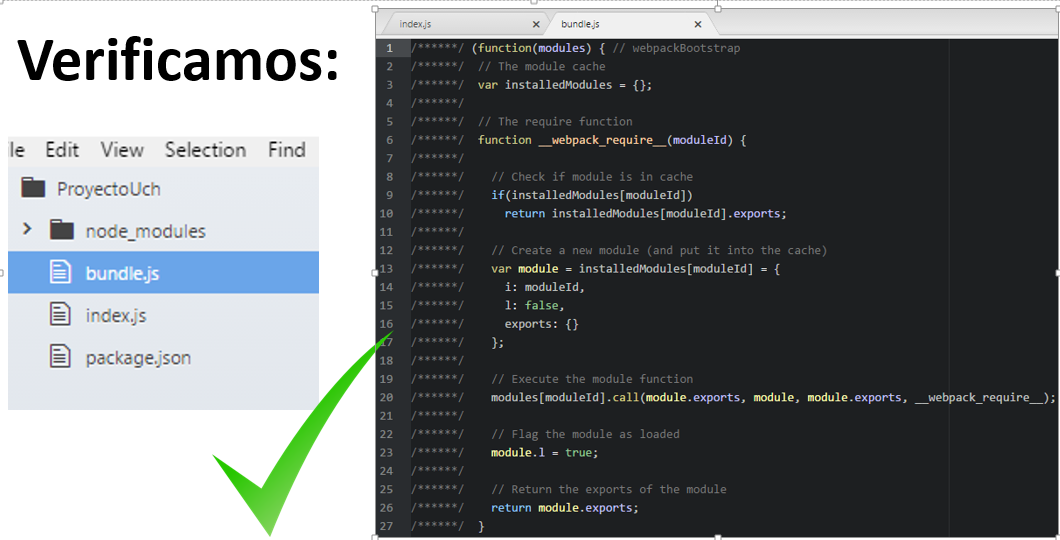


**VERIFICAMOS:** EN ESTE CASO ESTAMOS UTILIZAMOS EL EDITOR DE TEXTO **ATOM**



Paso 3: Accedemos al bin de node\_modules  
node\_modules\.bin\webpack  
Verificamos  
node\_modules\.bin\webpack –v  
Compilamos nuestro archivo  
node\_modules\.bin\webpack index.js bundle.js





CONCLUSIÓN

De momento hemos dado bastante información sobre el estado de los componentes, hemos podido distinguir componentes con estado y componentes sin estado, y aunque todavía nos quedan cosas que aprender, hemos podido ver ejemplos para hacernos una idea de cada tipo de componente.

En futuros artículos tenemos que abordar otros asuntos clave, como la inicialización de propiedades y estado ante diversos estilos de codificación, así como el ciclo de vida de los componentes, que es necesario conocer a fondo para implementar muchas de las necesidades de desarrollo en React

Con React se puede construir interfaces web de usuario,Es una librería de Javascript de código abierto, Fue creado por Facebook y está en constante actualización.Trabaja en la parte de la Vista.Facilidad de abstraer los componentes del DOM de todo el HTML, utilizando Virtual DOM.Librería ligera Multiplataforma.

REFERENCIAS

# <file:///C:/Users/Alumno/Downloads/manual-de-react.pdf>

<http://reactkungfu.com/2015/10/the-difference-between-virtual-dom-and-dom/>