Una exploración profunda de los componentes en React

**1. Introducción a los componentes en React**

En React, un componente es una parte de nuestro código que podemos reutilizar y que puede o no tener interacción con el usuario. Puede ser muy simple o muy complejo y puede contener otros componentes. Con esta descripción podría describir también un "servicio" en Angular, un "partial" en Marionette o una "vista" de Backbone pero veremos que la filosofía de React es diferente, ya lo veréis. Es importante entender y pensar teóricamente en React (y más en sus versiones superiores como React Native) que lo que estamos creando son componentes de un código nativo, algo que no abstrae mucho cuando trabajes con este framework (o librería).

¿Qué es un componente?

✏️ En el pasado ya os he escrito sobre React, un framework o librería para crear interfaces de usuario basados en JavaScript. Pero creo que nunca he explicado la idea de componentes que propone de acuerdo a otro tipo de aproximación, o mejor, lo haré ahora. No quiero que me quede gigante por eso no voy muy profundo por ahora pero sí que quiero hacerlo suficiente como para que tengamos una buena "base teórica" de React.

Introducción a los componentes en React

Y definimos los 2 archivos que servirán de entrada index y app.jsx.

Ya tenemos nuestro proyecto generado, vamos a pensar en la estructura que tendremos. La idea es que tendremos 2 partes fundamentales, el árbol de directorios es el siguiente: . └── src # Rama principal del proyecto ├── index.html # Archivo de entrada principal └── js # Directorio con todo el JavaScript ├── app.jsx # Archivo principal del proyecto └── css # Directorio con todo el CSS └── app.css # Archivo principal del proyecto

Vamos a crear un proyecto con el generador de paquetes para Node.js "npm". Comencemos por abrir un terminal si se encuentran en un sistema Unix o una consola de comandos sobre los sistemas operativos Windows. Una vez en la terminal, escribiremos el siguiente comando, le damos a la tecla "Intro" de nuestro teclado y nuestro proyecto se creará de forma automática. Suponiendo que tienes instalado "gulp" a nivel global, le tienes que dar enter y es importante estar en el directorio que va a ser el nuevo de tu proyecto.

**1.1. ¿Qué son los componentes en React?**

React se basa en el manejo por componentes, Conceptualmente, los componentes son clases JavaScript que hacen uso de una serie de métodos. Cada componente se estructura en torno a tres ideas muy similares a objetos de programación: métodos, render y estado. Este concepto de Presidente para concebir la interfaz de un producto ha sido importado de las teorías de diseño más primitivas que tienen cabida en la disciplina del conocimiento denominada Computación Pública (PC – Public Computing – human-computer interaction in the public space). Conceptualmente, los componentes son clases JavaScript que hacen uso de una serie de métodos, cuyos valores tuvieran que ser calculados muy a menudo para los diferentes posibles valores de sus propiedades (a modo de ejemplo de lo explicado, podemos recordar un sencillo componente como el que se muestra en el siguiente listado, correspondiente a la popular aplicación Facebook).

Algunas de las características que destacan a React es que utiliza JSX para sus componentes, el cual no puede entender un navegador sin ayuda de un transpilador y que su forma de trabajo es más bien “ciero-ahueco”, ya que no ofrece una cadena de herramientas para el desarrollo. Además, el DOM fue pensado para ser manejado desde JavaScript; sin embargo, el desarrollo completo desde cero de una nueva página web o aplicación es un proceso complicado, que requiere de múltiples pasos intermedios, varios de los cuales son hechos innecesariamente cada vez que los usuarios interactúan con una aplicación o página web. A adición a lo mencionado, el éxito de React puede resumirse en un concepto: componentes. Éste proporciona una manera sencilla de concebir y mantener interfaces complejas controlando la información en pequeñas “burbujas de materia”.

**1.2. Ventajas de utiliza**