**React Tutorial: An Overview and Walkthrough**

## **What is React.js?**

React.js is an open-source JavaScript library that is used for building user interfaces specifically for single-page applications. It’s used for handling the view layer for web and mobile apps. React also allows us to create reusable UI components. React was first created by Jordan Walke, a software engineer working for Facebook. React first deployed on Facebook’s newsfeed in 2011 and on Instagram.com in 2012.

React allows developers to create large web applications that can change data, without reloading the page. The main purpose of React is to be fast, scalable, and simple. It works only on user interfaces in the application. This corresponds to the view in the MVC template. It can be used with a combination of other JavaScript libraries or frameworks, such as Angular JS in MVC.

## **What are the React.js Features?**

Let us take a closer look at some important features of React.

React.js properties includes the following

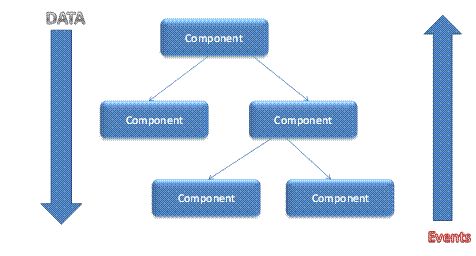
* React.js is declarative
* React.js is simple
* React.js is component based
* React.js supports server side
* React.js is extensive
* React.js is fast
* React.js is easy to learn

**JSX**

In React, instead of using regular JavaScript for templating, it uses JSX. JSX is a simple JavaScript that allows HTML quoting and uses these HTML tag syntax to render subcomponents. HTML syntax is processed into JavaScript calls of React Framework. We can also write in pure old JavaScript.

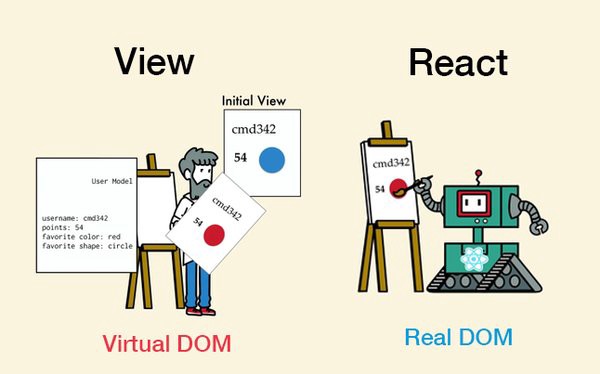
**Single-Way data flow**

In React, a set of immutable values are passed to the components renderer as properties in its HTML tags. The component cannot directly modify any properties but can pass a call back function with the help of which we can do modifications. This complete process is known as “properties flow down; actions flow up”.



**Virtual Document Object Model**

React creates an in-memory data structure cache which computes the changes made and then updates the browser. This allows a special feature that enables the programmer to code as if the whole page is rendered on each change whereas react library only renders components that actually change.



## **Why React?**

Now, the main question arises in front of us is why one should use React. There are so many open-source platforms for making the front-end web application development easier, like Angular. Let us take a quick look on the benefits of React over other competitive technologies or frameworks. With the front-end world-changing on a daily basis, it’s hard to devote time to learning a new framework – especially when that framework could ultimately become a dead end. So, if you're looking for the next best thing but you're feeling a little bit lost in the framework jungle, I suggest checking out React.

1. Simplicity

ReactJS is just simpler to grasp right away. The component-based approach, well-defined lifecycle, and use of just plain JavaScript make React very simple to learn, build a professional web (and mobile applications), and support it. React uses a special syntax called JSX which allows you to mix HTML with JavaScript. This is not a requirement; Developer can still write in plain JavaScript but JSX is much easier to use.

2. Easy to learn

Anyone with a basic previous knowledge in programming can easily understand React while Angular and Ember are referred to as ‘Domain-specific Language’, implying that it is difficult to learn them. To react, you just need basic knowledge of CSS and HTML.

3. Native Approach

React can be used to create mobile applications (React Native). And React is a diehard fan of reusability, meaning extensive code reusability is supported. So at the same time, we can make IOS, Android and Web applications.

4. Data Binding

React uses one-way data binding and an application architecture called Flux controls the flow of data to components through one control point – the dispatcher. It's easier to debug self-contained components of large ReactJS apps.

5. Performance

React does not offer any concept of a built-in container for dependency. You can use Browserify, Require JS, EcmaScript 6 modules which we can use via Babel, ReactJS-di to inject dependencies automatically.

6. Testability

ReactJS applications are super easy to test. React views can be treated as functions of the state, so we can manipulate with the state we pass to the ReactJS view and take a look at the output and triggered actions, events, functions, etc.

## **How to create a React app?**

Hopefully we will learn that tomorrow.