**Hello and welcome!**

My name is Ken, and I am here to talk about the React JS programming language. I’ll give you an introduction, then we will build an application, which is one of the best ways to learn.

Features from ES6 – a recent version of JavaScript will be used for this course.

Hopefully, you have some familiarity with HTML and JavaScript, but it is not required.

Please see other courses available on Skillshare.com for these classes if you can, because the more knowledge and experience you gain, the better. I tend to have lots of analogies that I use because I find visual ideas lead to an stronger understanding of something abstract such a software.

--------------------------

What is React?

React is the visual aspect of websites. Think of it as the interior of a car or house, if you will. The structure is there, but you are changing the look and features.

React JavaScript is a declarative, efficient, and flexible JavaScript library for building user interfaces. It lets you compose complex user interfaces from small and isolated pieces of code called “components”.

In technical terms such as MVC Architecture – Model-View-Controller, React is the View.

Why use React?

When you need to have multiple parts/components such as active/inactive navigation items, accordion sections that expand and collapse, dynamic inputs, buttons that can become active or disabled, user login and permissions, React is a good choice. It helps to manage those changing states and with presenting different or dynamic views to the user based on the state or conditions.

Virtual DOM

The common practice was to directly manipulate the DOM (the Document Object Model). For those who need a refresh it’s the parts that make up a web page.

(show DOM image). React is very efficient because it keeps a virtual copy of the page’s DOM and only changes the parts of the DOM that actually changed and not the entire DOM. Frequent changes and complexity slow down full DOM update. With React it becomes much faster.

Reusable parts/components.

Create components that you can call any time, combine, and nest with other components. This reduces the code you write and allows you to reuse code. It is like having a hammer; you wouldn’t want to get a new one every time you need it. Good coding practices are similar.

Supported by Facebook and a large community of React users.

JSX, React Native and Redux too, but that is for another video.

Set up environment

Check for updated NodeJS or download: <https://nodejs.org/en/>

npm -g install npm

(the g is global so that you can have it on the machine and not the directory)  -needs verification

Now we

npm install -g create-react-app  
create-react-app my-app

react start my-app

Congratulations! You created a React Application! Time to open it up to see what is inside.

What is a component?

Components make it possible to break down the UI into reusable, independent pieces. As a concept, think of components as JavaScript functions. They accept any given input (as “props”) and return React elements that describe what happens visually on the screen.

Now we move on to a more useful app. We will create a To Do List.

Find src and erase its contents?

What about a shopping list?

class ShoppingList extends React.Component {

render() {

return (

<div className="shopping-list">

<h1>Shopping List for {this.props.name}</h1>

<ul>

<li>Instagram</li>

<li>WhatsApp</li>

<li>Oculus</li>

</ul>

</div>

);

}

}