Construction of User Interfaces (SE/ComS 319)

Ali Jannesari

Department of Computer Science

Iowa State University, Spring 2022

INTRODUCTION TO REACT FRAMEWORK

Libraries and frameworks for Node.js & JavaScript (1)

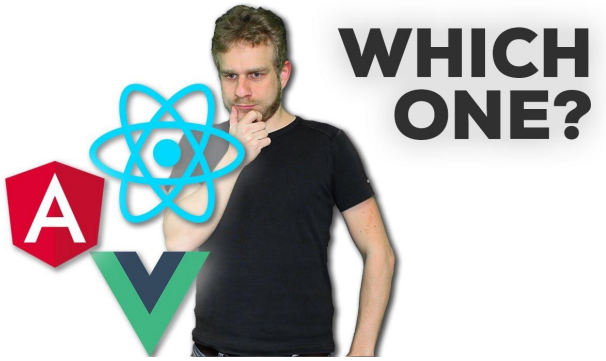
- **Chrome DevTools**

  - Debugging JavaScript
  - Performance analysis
  - Chromium project: open-source projects behind the Google Chrome browser and Google Chrome OS
- **NPM** is a package manager for Node.js packages, or modules

  - npm is the package manager for JavaScript
  - world's largest software registry: https://www.npmjs.com/

Libraries and frameworks for Node.js & JavaScript (2)

- Libraries/frameworks:
  - **React.js**: JavaScript library for building user interfaces by Facebook
  - **Angular/Angular.js**: TypeScript-based/Javascript framework by Google
  - **Vue.js**: rapidly growing JS framework



See this video for an answer: https://www.youtube.com/watch?v=KMX1mFEmM3E&app=desktop

JavaScript framework

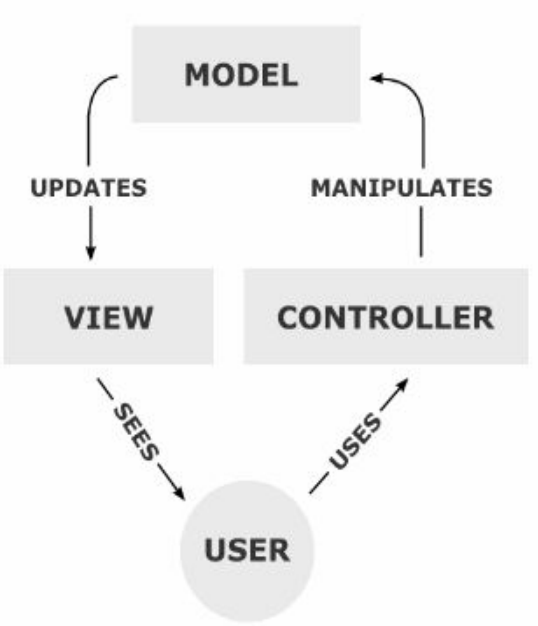
React

JavaScript framework – React

- **React** is a **JavaScript** library for building fast and interactive user interfaces for the web
- It allows developers to write highly efficient JavaScript for the purpose of rendering a UI
- It is an **open source**, reusable **component-based** front-end library
- Traditional JavaScript will re-render the entire DOM during a state change
  - But React will only render the parts of the DOM that have changed
- In a model-view-controller architecture, React is the ‘view’ which is responsible for how the app looks

Model – View – Controller (MVC) – Web UI

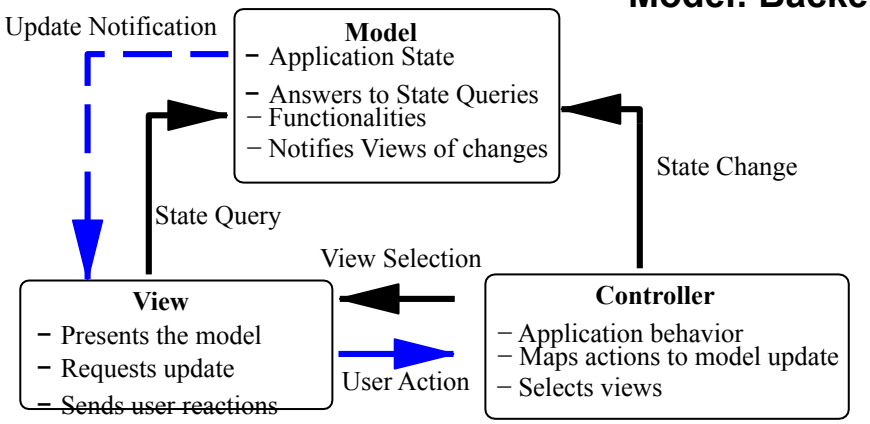
- **MVC in Web UI :**
  - **View**: Browser presentation (HTML)
  - **Model**: Data (Backend Database or (simple) embedded)
  - **Controller**:
    - Client scripts/programs, e.g. JavaScript
    - Server scripts/programs, e.g. Node.js



MVC architecture

- Model-View-Controller architecture:

Model: Backend Database



View: HTML

Controller: JavaScript, Node.js code

- Advantages:
  - Loosely coupled, modular
  - Model with different views
  - Controller decides when/how to update the model and/or the view
  - Model can change the view

A bit of history

Who?

- React is created by Jordan Walke at Facebook.

When?

- First deployed on Facebook's newsfeed in 2011 and became an open-source project in 2013 under the MIT License.

Additional Projects

- React Native for mobile was introduced in 2015.

Who uses React?

Why should I use React?

- Easy to read and understand views
- Components are the future of web development
- If your page uses a lot of fast updating data or real time data - React is the way to go
- Once you are over the React's learning curve, developing an app will become a lot faster

React fundamentals

- Components
- Props
- State
- JSX
- Virtual DOM

React – “Hello World!”

- Create a component called 'Test’, which is rendered in a container called 'root’:

index.js

```
import React from 'react';
import ReactDOM from 'react-dom';


class Test extends React.Component {
  render() {
    return <h1>Hello World!</h1>;
  }
}


ReactDOM.render(<Test />,
document.getElementById('root’));
```

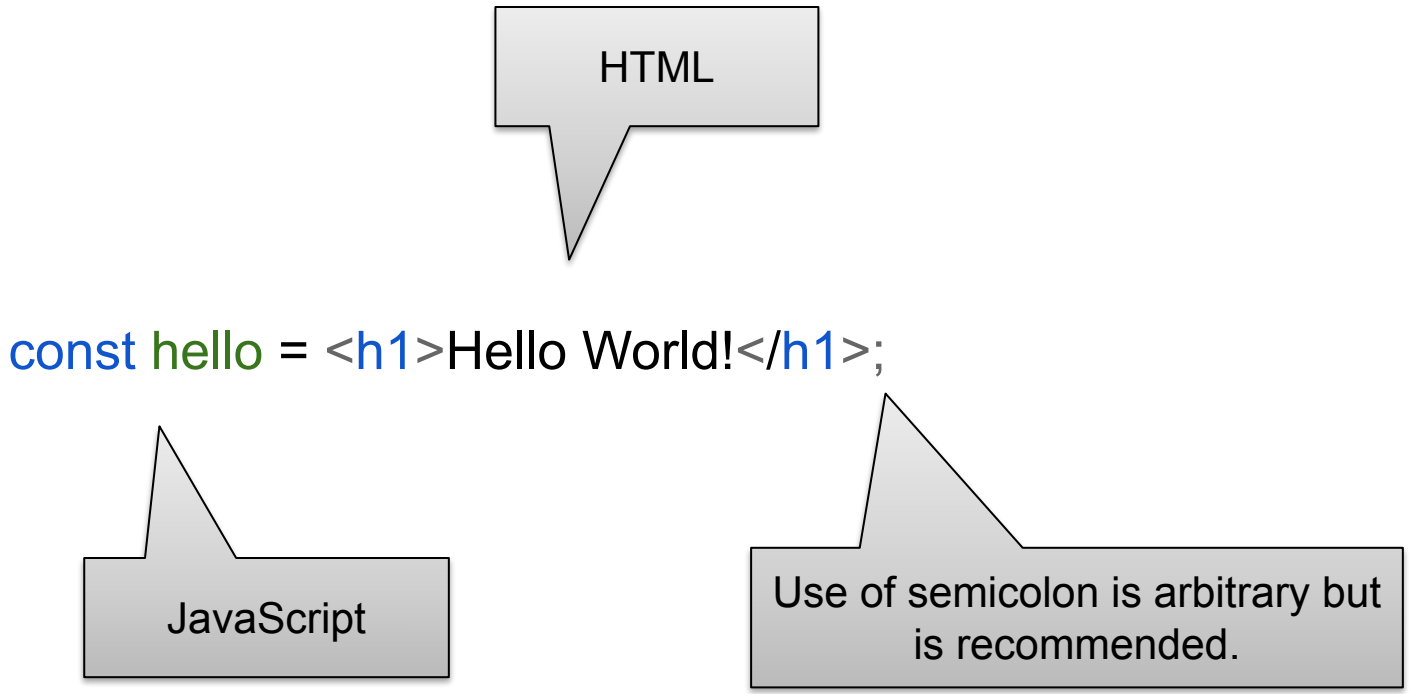
index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport"
      content="width=device-width,
initial-scale=1" />
    <title>React App</title>
  </head>
  <body>


    <div id="root"></div>


  </body>
</html>
```

JSX (JavaScript XML)

- JSX is a syntax extension to JavaScript.
- JSX allows you to write HTML structures in the same file that contains JavaScript code.
- JSX helps in making the code easier to understand and debug as it avoids usage of JS DOM structures which are rather complex.

JSX (JavaScript XML)



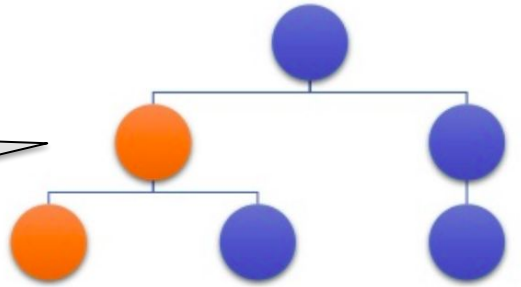
Virtual DOM

- A virtual DOM is used for efficient re-rendering of the DOM.

Document Object Model treats XML or HTML document as a tree structure.

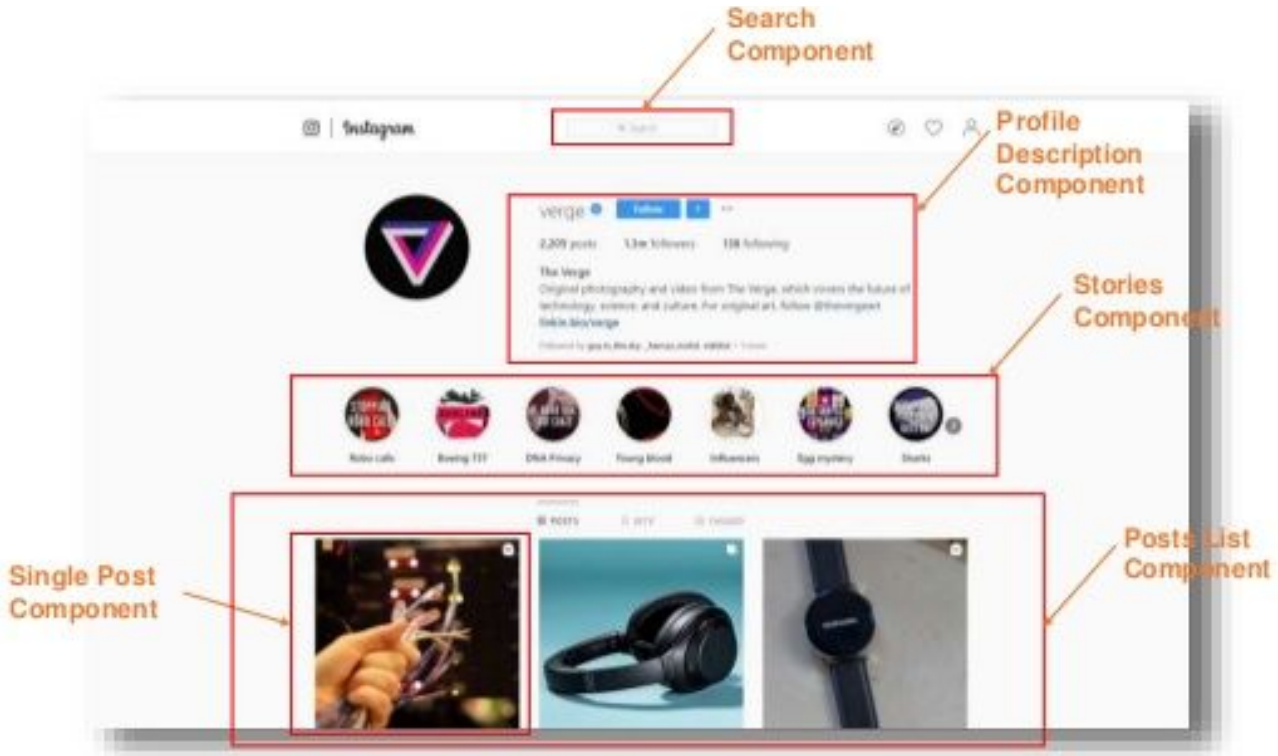
- React aims to re-render the virtual tree only when the state changes.

The state of these two objects has changed so they need to be re-rendered.



Components

- Components are self-contained reusable building blocks of web applications
- They describe pieces of your UI



Components

- Two way of declaring a component:

1. Function Component
2. Class Component

Components

- Function Component:

```
function Title() {
    return (
        <div>
            <h1>This is a function component.</h1>
        </div>
    );
}
```

Components

- Class Components:
- You can create more complex components by taking advantage of JavaScript classes.

```
class Title extends React.Component {
    render() {
        return(
        <div>
            <h1>This is a class component.</h1>
        </div>
        );
    }
}
```

Required method render() - returns React element tree of the Component's view.

State

- State of a component is an object that holds some data.
- This data influences the output of the component.

```
class App extends React.Component {
    constructor(){
        this.state = {
            car: "Toyota",
            bicycle: "Trek"
        }
    }
    render(){
        return(
            <div>
                <h1>{this.state.car}</h1>
                <h1>{this.state.bicycle}</h1>
            ...
```

This is how we storage some data in state.

This is how we access the state properties.

Props

- Props is short for properties, that allow us to **pass arguments** or data to components
- Props are passed to components in the way similar to that of HTML-tag attributes
- React Props are like function arguments in JavaScript *and* attributes in HTML

Props

Main.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App.js';

ReactDOM.render(
   <App carProp="Toyota" bikeProp="Trek"
/>,
    document.getElementById("App"));
```

This is how we pass the properties to a component.

App.js

```
class App extends React.Component {
   render(){
       return(
           <div>

<h1>{this.props.carProp}</h1>

<h1>{this.props.bikeProps}</h1>
           </div>
       );
   }
}
```

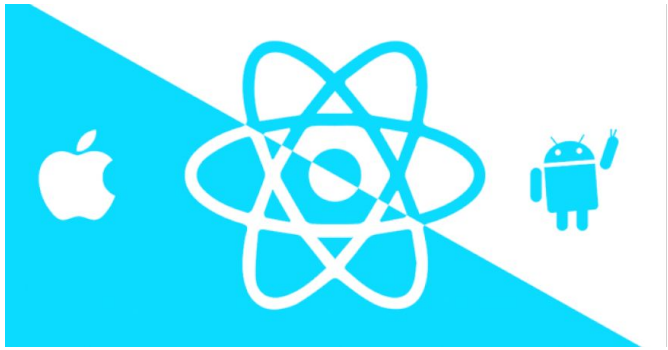
This is how we access the properties passed to the compoent

Beyond ReactJS

- React goes beyond simple UI and has many extensions for complete application support
- It provides **server-side rendering**
- Supports mobile app development
  - **React Native** lets your build mobile apps using only JavaScript.
- Extended with Flux and Redux
  - Flux is the application architecture that Facebook uses for building web applications
  - Redux is a popular JavaScript library for managing the state of your application

React Native

- React Native lets your build mobile apps using only JavaScript.
- A React Native app is a REAL mobile app and not just a web app running on mobile.



Literature – React

- https://www.w3schools.com/react/default.asp
- https://reactjs.org/docs/getting-started.html
- https://react-tutorial.app/
- https://www.tutorialspoint.com/reactjs/index.htm