Introduction to ReactJs

TAs Lucas and JongHo

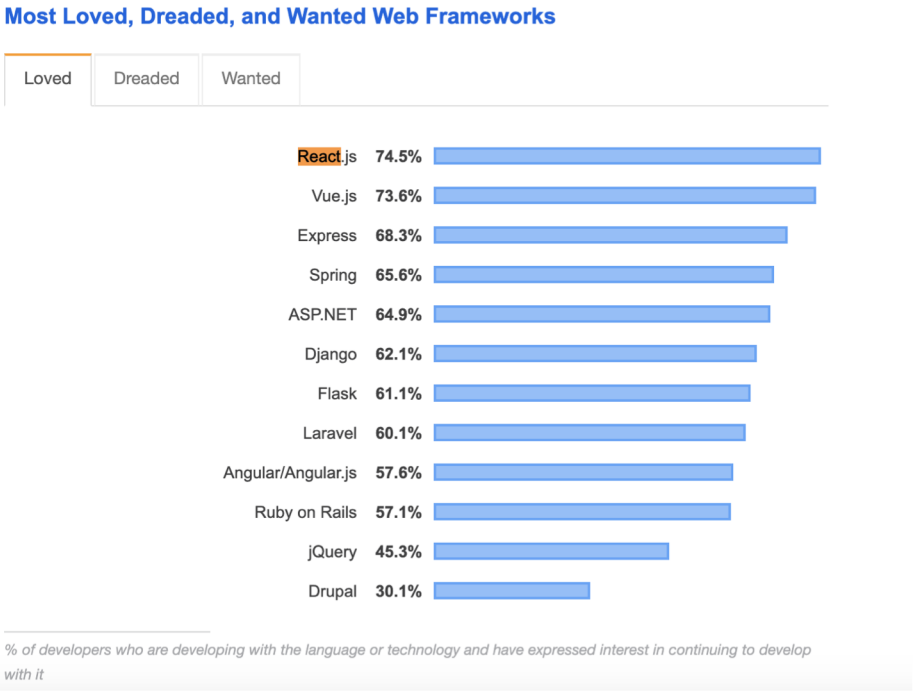
Today’s goals

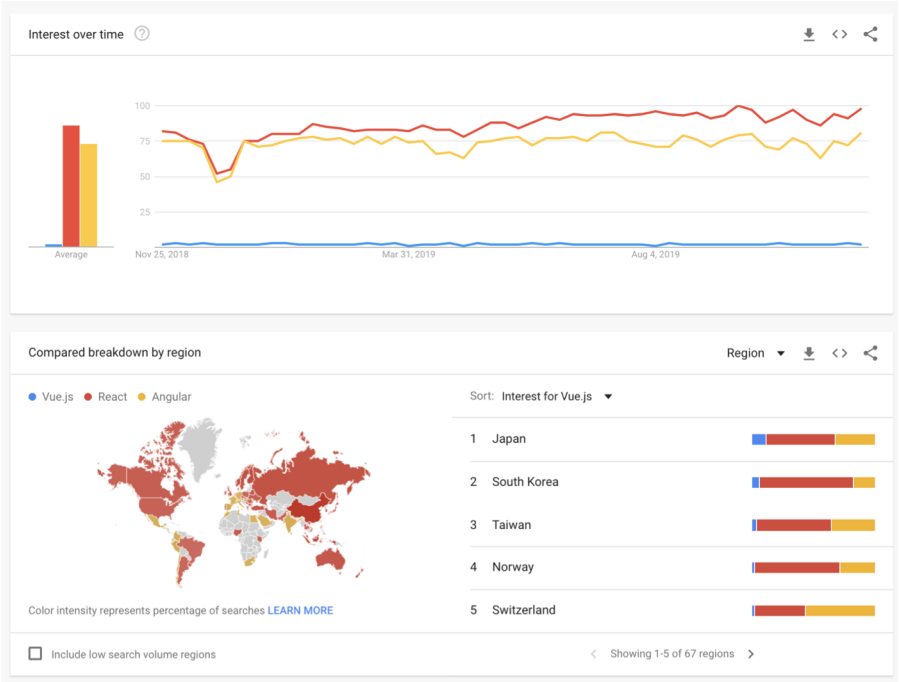
- Understand Reactjs/React fundamentals
- Overview of possible Toolchains and “ecosystem” around React
- How to get started with React dev

What is React?

- A JavaScript library for building user interfaces
- Component focused
  - A component is a mixture of HTML and JavaScript that captures all of the logic required to display a small section of a larger UI.
- Huge ecosystem of plugins, scripts, open source libs, etc.

Why React?





https://insights.stackoverflow.com/survey/2019

https://trends.google.com/trends/explore?cat=31&q=Vue.js,React,Angular

Why React?

- The “good”:
  - Open sourced
  - Huge community
  - Huge amount of libraries for almost anything
  - Highly flexible project structure
- The “bad”:
  - Highly flexible project structure
  - Maybe higher learning curve than Angular (?)
  - In practice you will be using many toolchains (other packages) to achieve different things (more on that later on). It is good to have such a big field to pick from, but demands more searching and learning.

React Fundamentals

- Components have a mix of HTML and JS in a single file: JSX
- CSS can be in a separate file (e.g.: CSS, SASS)
- Can use typescript, but it doesn’t seem common and most resources use JS
- CSS can be in the javascript (CSS-in-JS / JSS): componentization of CSS
  - https://cssinjs.org/?v=v10.0.0 CSS in JS
  - https://medium.com/dailyjs/what-is-actually-css-in-js-f2f529a2757

React Fundamentals: JSX

- JSX is a syntax extension to JavaScript
- It can only have one root element:
  - A single with everything else in it
  - A react fragment with syntax “<>”
- It has HTML tags (lowercase)
- It has components (has to start with capital letter)
- JS is used between { } nested between the HTML

```
return (
  <div className="App">
    <header className="App-header">
      <img src={logo} className="App-logo" alt="logo" />
      <p>
        Edit <code>src/App.js</code> and save to reload.
      </p>
      <HelloMessage name='class 133'/>
  </div>
 );
}
```

React Fundamentals: Components

- Components can be a class or a function.
- Functions are now more popular, but you could use both for different components in the same project
- Both class and function need to return JSX objects.
- A component can use several other components
- A component can pass objects to other components (much like Angular's bindings).

React Fundamentals: Components props

- Props (== Properties) are like inputs for Components
- Props are Read-Only
  - A component should never modify its own props
  - function withdraw(account, amount) { account.total -= amount; } // Should not do this
- A component can access a passed object with “`this.props.name`”

React Fundamentals: Components

Function

```
import React from 'react';
import logo from './logo.svg';
import './App.css';
import HelloMessage from './HelloMessage';

const App = () => {
 return (
   <div className="App">
     <header className="App-header">
       <img src={logo} className="App-logo" alt="logo" />
       <p>
         Edit <code>src/App.js</code> and save to reload.
       </p>
       <HelloMessage name='class 133'/>
   </div>
 );
}

export default App;
```

Import

Calls component and defines a prop value

Class

```
import React from 'react';
class HelloMessage extends React.Component {
 render() {
   return (
     <div>
       Hello {this.props.name}
       <h2>
         It is {new Date().toLocaleTimeString()}.
       </h2>
     </div>
   );}
}
export default HelloMessage;
```

React Fundamentals: Class

- Classes can have state, similar to Angular class attributes
  - state is a single attribute, usually a json object that can have other objects
- Classes can have many methods (e.g.: to handle button click)
- There are lifecycle methods
  - componentDidMount()
  - componentDidUpdate()
  - componentWillUnmount()

React Fundamentals: Class State

Defining State

```
import React from 'react';

class Counter extends React.Component {
   constructor(props) {
      super(props);
      this.addOne = this.addOne.bind(this);
      this.state = {counter: 0};
   }
   addOne(){
      this.setState({counter: this.state.counter+1});
   }
render() {
  return (
    <div>
      <p>{this.state.counter}</p>
      <button onClick={this.addOne}>+</button>
    </div>
  );
}
}
export default Counter;
```

Changing State

Defining method use in JSX

React Fundamentals: Function Hooks

- Functions can’t have a local state like a class does
- There is one type of React Hooks to solve this issue: State Hook
- There are many other types of hooks, including some with same effect as lifecycle methods in classes.
- You can even create your own hooks

React Fundamentals: Function Hooks

- First parameter: the state itself
- Second parameter: the setter
- useState receives an initial value
- You can use many hooks in a function (e.g: create another state hook)

```
import React, { useState } from 'react';
import './App.css';

const CounterFunc = () => {

const [count, setCount] = useState(0);

const add = () => {
  setCount(count + 1);
}

 return (
   <div>
     <p>You clicked {count} times</p>
     <button onClick={() => add()}>
       Click me
     </button>
   </div>
 );
}

export default CounterFunc;
```

Creating a state hook

Live Demo: Setting up

- Create a react app using Create-react-app tool:
  - npx create-react-app my-groceries
  - cd my-app
  - npm start /or/ yarn start
- Add material-ui for styling:
  - npm install @material-ui/core
  - yarn add @material-ui/core
- App has a “index.js” that renders a root component App.js

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

Next steps in learning React

- Go through https://reactjs.org/ walkthrough and tutorials. It has intuitive and easily understandable lessons.
- Redux: A predictable state container for JavaScript apps. Helps manage state that can be used across many components. Has a learning curve, but is extremely useful.
- React router: rendering specific components according to URL.
- Server rendering: Gatsby or NextJS.

Next steps in learning React

- React Native.
- Styling components can be with plain CSS and/or using libs:
    1. Material-UI (https://material-ui.com/)
    2. React Bootstrap (https://react-bootstrap.github.io/)
    3. SASS
    4. Many others
- Hosting with Heroku, Netlify, or many others.