TW-010 TEAM LEAD VERSION (Sprint-7 Week-1)

Meeting Agenda

▶ Icebreaking

▶ Questions

▶ Interview Questions

▶ Coding Challenge

▶ Video of the week

▶ Retro meeting

▶ Case study / project

Teamwork Schedule

Ice-breaking 5m

- Personal Questions (Study Environment, Kids etc.)
- Any challenges (Classes, Coding, studying, etc.)
- Ask how they're studying, give personal advice.
- Remind that practice makes perfect.

Team work 5m

- Ask what exactly each student does for the team, if they know each other, if they care for each other, if they follow and talk with each other etc.

Ask Questions 15m

1. Consider the following code from React Router. What do you call :id in the path prop?

```
<Route path="/:id" />
```

A. This is a route modal
B. This is a route parameter
C. This is a route splitter
D. This is a route link

Answer: B

2. If a function component should always render the same way given the same props, what is a simple performance optimization available for it?

A. Wrap it in the `React.memo` higher-order component.
B. Implement the `useReducer` Hook.
C. Implement the `useRouter` Hook.
D. Implement the `shouldComponentUpdate` lifecycle method.

Answer: A

3. What can you use to handle code splitting?

A. `React.memo`
B. `React.split`
C. `React.lazy`
D. `React.fallback`

Answer: C

4. When might you use `React.PureComponent`?

A. when you do not want your component to have props
B. when you have sibling components that need to be compared
C. when you want a default implementation of `shouldComponentUpdate()
D. when you do not want your component to have state

Answer: C

5. You have written the following code but nothing is rendering. How do you fix this problem?

```
const Heading = () => {
  <h1>Hello!</h1>;
};
```

A. Add a render function.
B. Change the curly braces to parentheses or add a return statement before the `h1` tag.
C. Move the `h1` to another component.
D. Surround the `h1` in a `div`.

Answer: B

6. Which option is correct for State vs Props

A. Props is something that the parent doesn't need and decide to throw around among other parents; State is the parent's favorite child and something the component wants to nurture.
B. Props is a copy of real DOM; State is the definition of the real DOM.
C. Props get passed to the component using naming conventions, like a function parameter; State is managed within the component and holds some information that may change over the lifetime of the component.
D. Prop is managed within the component and holds some information that may change over the lifetime of the component; State gets passed to the component, like a function parameter

Answer: C

Explanation: We can update state within the component using setState() , then we can pass the state to its children. Its children would use this.props to take whatever state from its parents.

7. What is styled-component styling in React?

A. These styles are written as attributes and are passed to the element.
B. It is a JavaScript library for styling React applications. It removes the mapping between styles and components, and lets you write actual CSS augmented with JavaScript.

C. It is basically a .css file that is compiled. When compiled, it produces two outputs. One is CSS that is a modified version of input CSS with the renamed class names. The other is a JavaScript object that maps the original CSS name with the renamed name.
D. It offers a different approach in which no CSS needs to be written to style an application. Instead, It uses utility classes for each CSS property that you can use directly in your HTML or JSX.

Answer: B

8. What is `[e.target.id]` called in the following code snippet?

```
handleChange(e) {
  this.setState({ [e.target.id]: e.target.value })
}
```

A.a computed property name
B.a set value
C.a dynamic key
D.a JSX code string

Answer: C

9. What is sent to an `Array.map()` function?

A.a callback function that is called once for each element in the array
B.the name of another array to iterate over
C.the number of times you want to call the function
D.a string describing what the function should do

Answer: A

10. What package contains the render() function that renders a React element tree to the DOM?

A.`React`
B.`ReactDOM`
C.`Render`
D.`DOM`

Answer: B

11. What do you need to change about this code to get it to run?

```
class clock extends React.Component {
  render() {
    return <h1>Look at the time: {this.props.time}</h1>;
  }
}
```

A.Add quotes around the return value
B.Remove `this`

C.Remove the render method
D.Capitalize `clock`

Answer: D **Explanation:** In JSX, lower-case tag names are considered to be HTML tags. Read this article

12. How do you invoke setDone only when component mounts, using hooks?

```
function MyComponent(props) {
  const [done, setDone] = useState(false);

  return <h1>Done: {done}</h1>;
}
```

A. `useEffect(() => { setDone(true); });`
B. `useEffect(() => { setDone(true); }, []);`
C. `useEffect(() => { setDone(true); }, [setDone]);`
D. `useEffect(() => { setDone(true); }, [done, setDone]);`

Answer: B

13. Which of the following click event handlers will allow you to pass the name of the person to be hugged?

```
class Huggable extends React.Component {
  hug(id) {
    console.log("hugging " + id);
  }

  render() {
    let name = "kitteh";
    let button = // Missing Code
    return button;
  }
}
```

A. `<button onClick={(name) => this.hug(name)}>Hug Button</button>`
B. `<button onClick={this.hug(e, name)}>Hug Button</button>`
C. `<button onClick={(e) => hug(e, name)}>Hug Button</button>`
D. `<button onClick={(e) => this.hug(name,e)}>Hug Button</button>`

Answer: D

14. Which answer best describes a function component?

A. A function component is the same as a class component.
B. A function component accepts a single props object and returns a React element.
C. A function component is the only way to create a component.
D. A function component is required to create a React component.

Answer: B

15. You have created a new method in a class component called handleClick, but it is not working. Which code is missing?

```
class Button extends React.Component{

  constructor(props) {
    super(props);
    // Missing line
  }

  handleClick() {...}
}
```

A. `this.handleClick.bind(this);`
B. `props.bind(handleClick);`
C. `this.handleClick.bind();`
D. `this.handleClick = this.handleClick.bind(this);`

Answer: D

Interview Questions 15m

1. What is the difference between Element and Component?

Answer: An Element is a plain object describing what you want to appear on the screen in terms of the DOM nodes or other components. Elements can contain other Elements in their props. Creating a React element is cheap. Once an element is created, it is never mutated. Whereas a component can be declared in several different ways. It can be a class with a render() method. Alternatively, in simple cases, it can be defined as a function.

2. What is the difference between state and props?

Answer: Both props and state are plain JavaScript objects. While both of them hold information that influences the output of render, they are different in their functionality with respect to component. Props get passed to the component similar to function parameters whereas state is managed within the component similar to variables declared within a function.

3. What is the difference between Shadow DOM and Virtual DOM?

Answer: The Shadow DOM is a browser technology designed primarily for scoping variables and CSS in web components. The Virtual DOM is a concept implemented by libraries in JavaScript on top of browser APIs.

4. What are the lifecycle methods of React? Answer:

Before React 16.3
componentWillMount: Executed before rendering and is used for App level configuration in your root component.
componentDidMount: Executed after first rendering and here all AJAX requests, DOM or state updates, and set up event listeners should occur.
componentWillReceiveProps: Executed when particular prop updates to trigger state transitions.
shouldComponentUpdate: Determines if the component will be updated or not. By default it returns true. If you are sure that the component doesn't need to render after state or props are updated, you can return false value. It is a great place to improve performance as it allows you to prevent a re-render if component receives new prop.
componentWillUpdate: Executed before re-rendering the component when there are props & state changes confirmed by shouldComponentUpdate() which returns true.
componentDidUpdate: Mostly it is used to update the DOM in response to prop or state changes.
componentWillUnmount: It will be used to cancel any outgoing network requests, or remove all event listeners associated with the component.

React 16.3+
getDerivedStateFromProps: Invoked right before calling render() and is invoked on every render. This exists for rare use cases where you need derived state. Worth reading if you need derived state.
componentDidMount: Executed after first rendering and here all AJAX requests, DOM or state updates, and set up event listeners should occur.
shouldComponentUpdate: Determines if the component will be updated or not. By default it returns true. If you are sure that the component doesn't need to render after state or props are updated, you can return false value. It is a great place to improve performance as it allows you to prevent a re-render if component receives new prop.
getSnapshotBeforeUpdate: Executed right before rendered output is committed to the DOM. Any value returned by this will be passed into componentDidUpdate(). This is useful to capture information from the DOM i.e. scroll position.
componentDidUpdate: Mostly it is used to update the DOM in response to prop or state changes. This will not fire if shouldComponentUpdate() returns false.
componentWillUnmount: It will be used to cancel any outgoing network requests, or remove all event listeners associated with the component.

5. What is React Router?

Answer: A tool that allows you to handle routes in a web app, using dynamic routing. Dynamic routing takes place as the app is rendering on your machine, unlike the old routing architecture where the routing is handled in a configuration outside of a running app. React router implements a component-based approach to routing. It provides different routing components according to the needs of the application and platform.

React Router, and dynamic, client-side routing, allows us to build a single-page web application with navigation without the page refreshing as the user navigates. React Router uses component structure to call components, which display the appropriate information.

Most of the applications built with React.js are SPA (single page application), but it doesn't mean your app will have only one view. It means your app doesn't need to reload to another view, but how can we change views and go into the next page? We can use a react router for that! React router is the official and standard routing package that we use in React.js to change views, and move between pages.

With the React router, we can specify the whole routing for our modules that will decide what view should be visible when we enter the specified URL but not only. React router gives us the possibility to create protected views like, for example, the view that we need to be logged in or have any special requirements to visit.

One more useful feature of the React Router is the routing history, that can keep all of the history of our views, and come back to the specified step when needed.

The id of element, to render the route that can show specified elements, and give you access to that param. We can use routing navigation in a few ways. The most popular is to type URL, visit URL by a link inside the menu, or add data to the routing history.

On the example below, you can simple routing:

```
<Switch>
  <Route path="/about">
    <About />
  </Route>
  <Route path="/contact/:id">
    <Contact />
  </Route>
  <Route path="/contact">
    <AllContacts />
  </Route>
  <Route path="/">
    <Home />
  </Route>
</Switch>
```

Coding Challenge 20m

- Coding Challenge: JS-CC-015: Sliding Window Maximum

☕

Coffee Break 10m

☕

Videos of the Week 5m

- What to do in an interview
- React Router Setup in 5 minutes

Retro Meeting on a personal and team level 5m

Ask the questions below:

- What went well?
- What went wrong?
- What is the improvement areas?

Case study/Project 15m

Case study should be explained to the students during the weekly meeting and has to be completed in one week by the students. Students should work in small teams to complete the case study.

- RC-004 Random User App

Closing 5m

-Next week’s plan

-QA Session
