TW-010 TEAM-LEAD VERSION (Sprint-6 Week-2)

TEAMWORK

CLARUSWAY
WAY TO REINVENT YOURSELF

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. You are rendering a list with React when this warning appears in the console: "Warning: Each child in a list should have a unique 'key' prop." How do you fix this issue?

A. Pass the name of each item as its key.
B. Add a key prop with the same value to each item the list.
C. Clear the console warnings.
D. When iterating over the list items, add a unique property to each list item.

Answer: D

2. In this component, how do you display whether the user was logged in or not?

```
render() {
  const isLoggedIn = this.state.isLoggedIn;
  return (
    <div>
      The user is:
    </div>
  );
}
```

A. The user is loggedIn ? logged in : not logged in.
B. Write a function to check the login status.
C. The user is {isLoggedIn = "no"}.
D. The user is {isLoggedIn ? "logged in." : "not logged in"}.

Answer: D

3. Filter, Map and Reduce operation is way to _____ .

A. Un-compress and expand data.
B. Create new immutable dataset.
C. Crunch and analyze the data
D. All of them

Answer: C

4. What is the first file loaded by the browser in a basic React project?

A. src/App.js
B. src/index.js
C. public/index.html
D. public/manifest.json

Answer: C

5. What is the "key" prop?

A. "Key" prop is just there to look pretty and there is no benefit whatsoever
B. "Key" prop is a way for React to identify a newly added item in a list and compare during the "diffing" algorithm
C. It is one of the attributes in Javascript and HTML
D. All I know is that it is NOT commonly used in array

Answer: B

6. How do you handle passing through the component tree without having to pass props down manually at every level?

A. React Send
B. React Pinpoint
C. React Router
D. React Context

Answer:D

7. Why might you use useReducer over useState in a React component?

A. when you want to replace Redux
B. when you need to manage more complex state in an app
C. when you want to improve performance
D. when you want to break your production app

Answer: B

8. Which props from the props object is available to the component with the following syntax?

```
<Message {...props}/>
```

A. any that have not changed
B. child props
C. any that have changed
D. all of them

Answer: D

9. What is the difference between the click behaviors of these two buttons (assuming that this.handleClick is bound correctly)?

```
X. <button onClick="{this.handleClick}>Click Me</button>"
Y. <button onClick="{event => this.handleClick(event)}}>Click Me</button>"
```

A. Button X will not have access to the event object on click of the button
B. Button X will not fire the handler this.handleClick successfully
C. Button Y will not fire the handler this.handleClick successfully
D. There is no difference

Answer: C

10. How do you destructure the properties that are sent to the Dish component?

```
function Dish(props) {
  return (
    <h1>
      {props.name} {props.cookingTime}
    </h1>
  );
}
```

A. `function Dish([name, cookingTime]) { return <h1>{name} {cookingTime}</h1>; }`
B. `function Dish(...props) { return <h1>{name} {cookingTime}</h1>; }`
C. `function Dish(props) { return <h1>{name} {cookingTime}</h1>; }`
D. `function Dish({name, cookingTime}) { return <h1>{name} {cookingTime}</h1>; }`

Answer: D

11. Which attribute do you use to replace innerHTML in the browser DOM?

A. injectHTML
B. dangerouslySetInnerHTML
C. weirdSetInnerHTML
D. strangeHTML

Answer:B

12. What is this pattern called?

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

A. object destructuring
B. spread operating
C. array destructuring
D. code pushing

Answer:C

13. What is wrong with this code?

```
const MyComponent = ({ names }) => (
  <h1>Hello</h1>
  <p>Hello again</p>
);
```

A. React does not allow components to return more than one element.
B. React components cannot be defined using functions.
C. The component needs to use the return keyword.
D. String literals must be surrounded by quotes.

Answer:A

Interview Questions 15m

1. What are React Hooks?

Answer: Hooks are a new addition in React 16.8. They let you use state and other React features without writing a class. With Hooks, you can extract stateful logic from a component so it can be tested independently and reused. Hooks allow you to reuse stateful logic without changing your component hierarchy. This makes it easy to share Hooks among many components or with the community.

2. Why should component names start with capital letter?

Answer: If you are rendering your component using JSX, the name of that component has to begin with a capital letter otherwise React will throw an error as unrecognized tag. This convention is because only HTML elements and SVG tags can begin with a lowercase letter.

```
class OneComponent extends Component {
// ...
}
```

You can define component class which name starts with lowercase letter, but when it's imported it should have capital letter. Here lowercase is fine:

```
class myComponent extends Component {
   render() {
     return <div />;
   }
}

export default myComponent;
```

While when imported in another file it should start with capital letter:

```
import MyComponent from './MyComponent';
```

3. What is Suspense component?

Answer: If the module containing the dynamic import is not yet loaded by the time parent component renders, you must show some fallback content while you're waiting for it to load using a loading indicator. This can be done using Suspense component.

```
const OneComponent = React.lazy(() => import('./OneComponent'));

function MyComponent() {
  return (
    <div>
      <Suspense fallback={<div>Loading...</div>}>
        <OneComponent />
      </Suspense>
    </div>
  );
}
```

As mentioned in the above code, Suspense is wrapped above the lazy component.

4. What is React context vs React redux?

Answer: You can use Context in your application directly and is going to be great for passing down data to deeply nested components which what it was designed for. Whereas Redux is much more powerful and provides a large number of features that the Context Api doesn't provide.

Also, React Redux uses context internally but it doesn’t expose this fact in the public API. So you should feel much safer using Context via React Redux than directly because if it changes, the burden of updating the code will be on React Redux instead developer responsibility.

5. What are controlled components?

Answer: In HTML, form elements such as "input", "textarea", and "select" typically maintain their own state and update it based on user input. When a user submits a form the values from the aforementioned elements are sent with the form. With React it works differently. The component containing the form will keep track of the value of the input in it's state and will re-render the component each time the callback function e.g. onChange is fired as the state will be updated. A form element whose value is controlled by React in this way is called a "controlled component". With a controlled component, every state mutation will have an associated handler function. This makes it straightforward to modify or validate user input.

6. How Virtual-DOM is more efficient than Dirty checking?

Answer: In React, each of our components have a state. This state is like an observable. Essentially, React knows when to re-render the scene because it is able to observe when this data changes. Dirty checking is slower than observables because we must poll the data at a regular interval and check all of the values in the data structure recursively. By comparison, setting a value on the state will signal to a listener that some state has changed, so

React can simply listen for change events on the state and queue up re-rendering. The virtual DOM is used for efficient re-rendering of the DOM. This isn't really related to dirty checking your data. We could re-render using a virtual DOM with or without dirty checking. In fact, the diff algorithm is a dirty checker itself. We aim to re-render the virtual tree only when the state changes. So using an observable to check if the state has changed is an efficient way to prevent unnecessary re-renders, which would cause lots of unnecessary tree diffs. If nothing has changed, we do nothing.

Coding Challenge 20m

- Coding Challenge: JS-CC-025 Roman Numerals

Coffee Break 10m

Video of the Week 5m

- What NOT to do in an Interview

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.

- Random User App (RP-04)

Closing 5m

-Next week's plan

-QA Session
