TEAM LEAD VERSION (Week-14)

Meeting Agenda

▶ Icebreaking

▶ Questions

▶ Interview Questions

▶ Coding Challenge

▶ Video of the
week

▶ Retro
meeting

▶ Case study /
project

Teamwork Schedule

Ice-breaking	**5m**	

- Personal Questions (Stay at home & Corona, 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. What can you use to handle code splitting?

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

Answer:C

2. When do you use useLayoutEffect?

A. to optimize for all devices
B. to complete the update
C. to change the layout of the screen
D. when you need the browser to paint before the effect runs

Answer:D

3. 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 Y will not fire the handler this.handleClick successfully.
C. Button X will not fire the handler this.handleClick successfully.
D. There is no difference

Answer: B

4. 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({name, cookingTime}) { return <h1>{name} {cookingTime}</h1>; }`

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

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

Answer: B

5. 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

6. Why is it important to avoid copying the values of props into a component's state where possible?

A. because you should never mutate state
B. because getDerivedStateFromProps() is an unsafe method to use
C. because you want to allow a component to update in response to changes in the props
D. because you want to allow data to flow back up to the parent

Answer: C

7. What is the children prop?

A. a property that adds child components to state
B. a property that lets you pass components as data to other components
C. a property that lets you set an array as a property
D. a property that lets you pass data to child elements

Answer:B

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

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

Answer:B

Interview Questions	**15m**	

1. What is a higher order component?

> Answer:

A higher-order component (HOC) is an advanced technique in React for reusing component logic. HOCs are not part of the React API. They are a pattern that emerges from React's compositional nature.

A higher-order component is a function that takes a component and returns a new component.

HOC's allow you to reuse code, logic and bootstrap abstraction. HOCs are common in third-party React libraries. The most common is probably Redux's connect function. Beyond simply sharing utility libraries and simple composition, HOCs are the best way to share behavior between React Components. If you find yourself writing a lot of code in different places that does the same thing, you may be able to refactor that code into a reusable HOC.

2. What is `create-react-app`?

> Answer:

`create-react-app` is the official CLI (Command Line Interface) for React to create React apps with no build configuration.

We don't need to install or configure tools like Webpack or Babel. They are preconfigured and hidden so that we can focus on the code. We can install easily just like any other node modules. Then it is just one command to start the React project.

```
create-react-app my-app
```

It includes everything we need to build a React app:

- React, JSX, ES6, and Flow syntax support.
- Language extras beyond ES6 like the object spread operator.
- Autoprefixed CSS, so you don't need `-webkit-` or other prefixes.
- A fast interactive unit test runner with built-in support for coverage reporting.
- A live development server that warns about common mistakes.
- A build script to bundle JS, CSS, and images for production, with hashes and sourcemaps.

3. What is Redux?

> Answer:

The basic idea of Redux is that the entire application state is kept in a single store. The store is simply a javascript object. The only way to change the state is by firing actions from your application and then writing reducers for these actions that modify the state. The entire state transition is kept inside reducers and should not have any side-effects.

Redux is based on the idea that there should be only a single source of truth for your application state, be it UI state like which tab is active or Data state like the user profile details.

```
{
    first_name: 'John',
    last_name: 'Doe',
    age: 42
}
```

All of these data is retained by redux in a closure that redux calls a store . It also provides us a recipe of creating the said store, namely `createStore(x)`.

The `createStore` function accepts another function, `x` as an argument. The passed in function is responsible for returning the state of the application at that point in time, which is then persisted in the store. This passed in function is known as the `reducer`.

This is a valid example reducer function:

```
export default function reducer(state={}, action) {
    return state;
}
```

This store can only be updated by dispatching an action. Our App dispatches an `action`, it is passed into `reducer`; the reducer returns a fresh instance of the `state`; the store notifies our App and it can begin it's re render as required.

Coding Challenge	**20m**	

- Coding Challenge: JS-CC-009 : Convert Numbers

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**	

This week, there will only be one in-class project.

Closing	**5m**	

-Next week's plan

-QA Session