Web Based Technologies 2020/2021

REACT

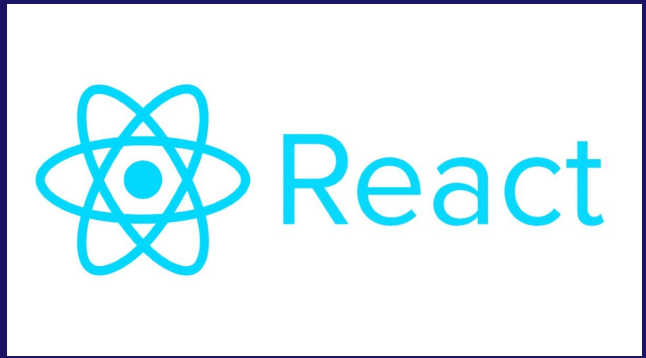
Manuel Germán Morales

TABLE OF CONTENTS

Hello react!

Hello react!

- React is a JS library for building user interfaces.
- React is a component-based library.
- You don't need to worry about re-rendering components, React does that for you.
- You can use JSX to describe your components.



1

React components

What is a component?

- A component is a small part of our interface.
- Components can be defined as classes or functions.
- Every react component has a lifecycle defined by “lifecycle methods”.
- React components can have **prop**erties and also a **state**.

Class component definition

```
class Component extends React.Component {
      ...
}
```

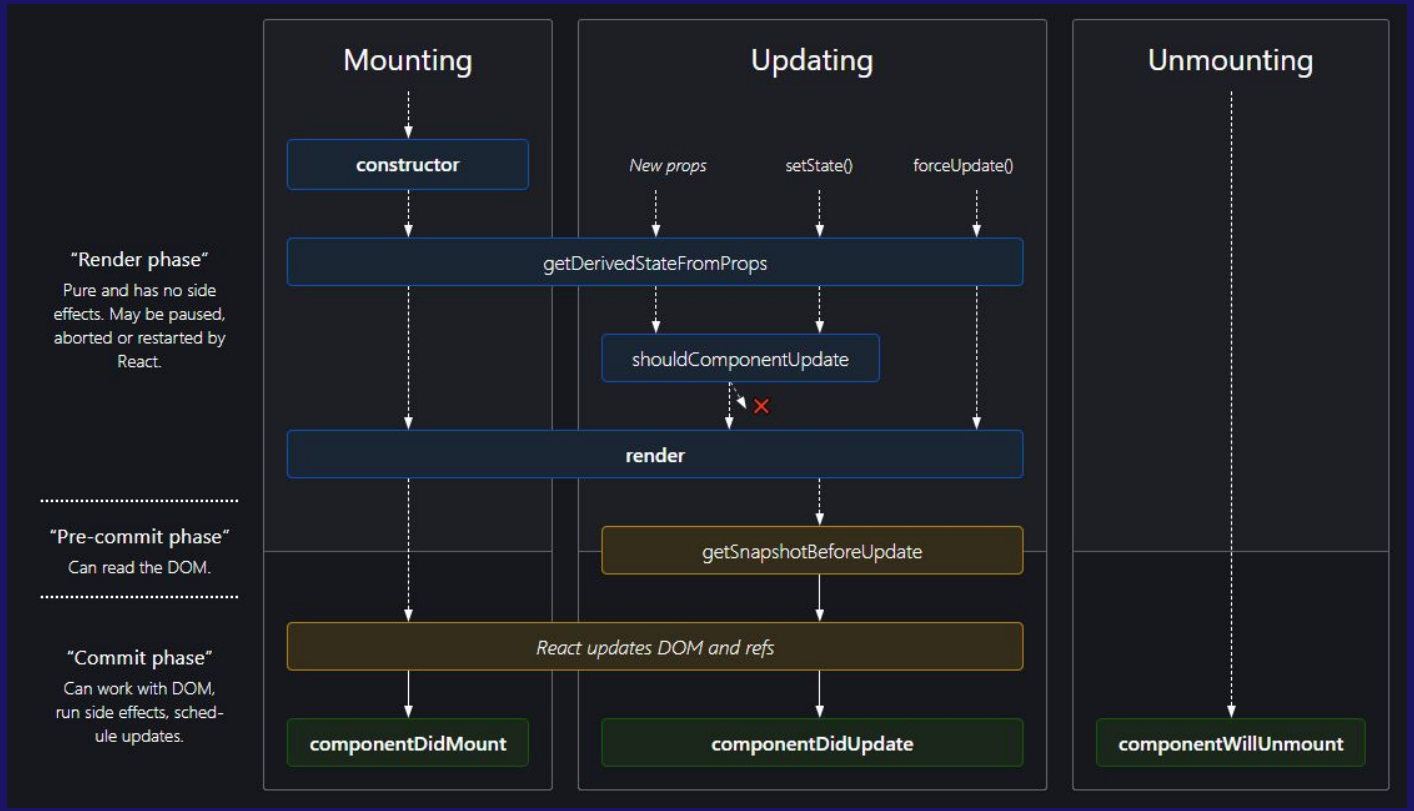
Function component definition

```
function Component() {...}

const Component = () => {...}
```

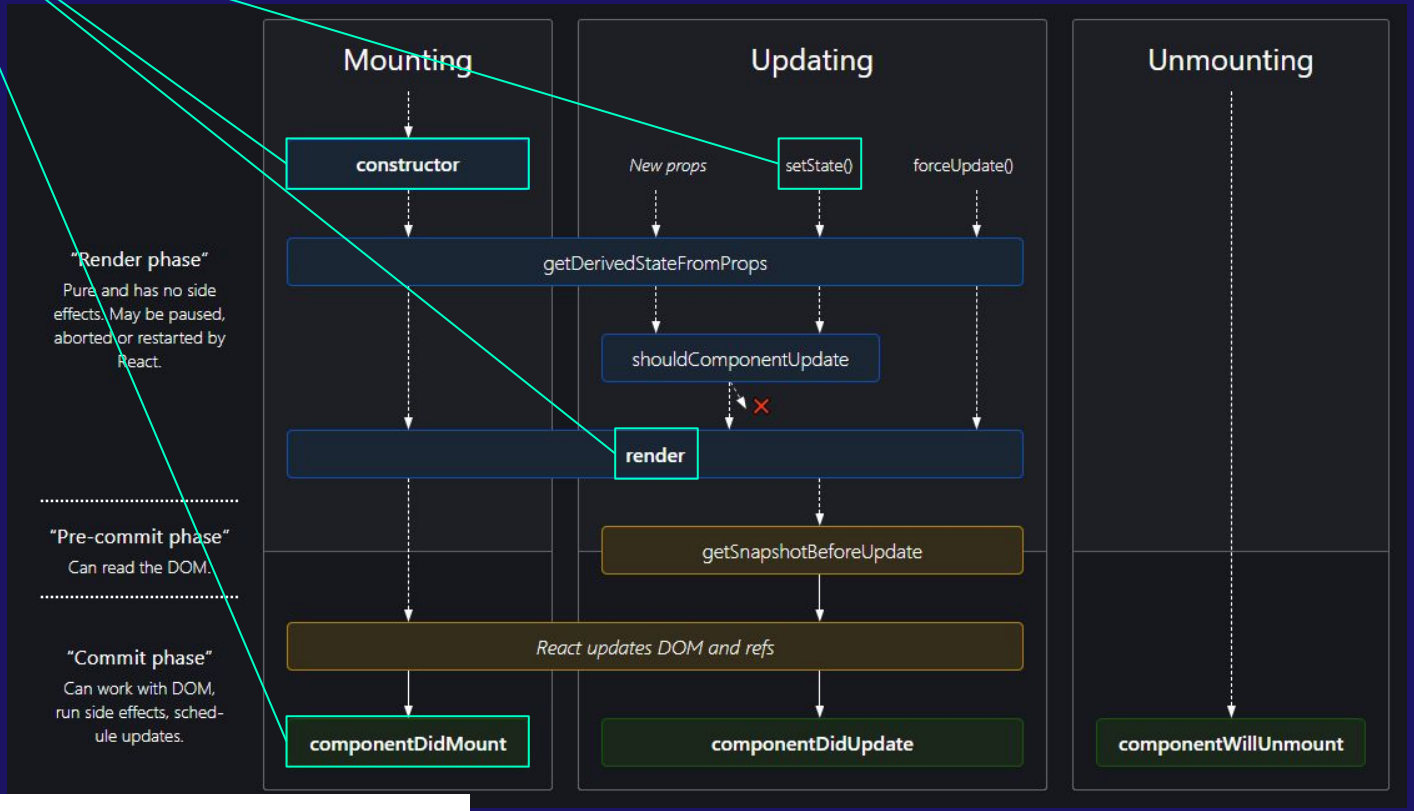
Don’t forget to export it!

Lifecycle



Source: https://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/

Lifecycle



More info about these methods at:
https://reactjs.org/docs/react-component.html

Source: https://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/

JSX vs Non-JSX component

Non-JSX

```
class Component extends React.Component {
  render() {
    return React.createElement(
      "div",
      null,
      "Hello React!"
    );
  }
}

ReactDOM.render(React.createElement(Component),
document.getElementById('html-element'));
```

JSX

```
class TheCoolerComponent extends React.Component {
  render() {
    return (
      <div>
        Hello React!
      </div>
    );
  }
}

ReactDOM.render(
  <TheCoolerComponent />,
  document.getElementById('html-element')
);
```

Output of both components:

What do you prefer? I hope the JSX one.

JSX vs Non-JSX component

Non-JSX

```
class Component extends React.Component {
  render() {
    return React.createElement(
      "div",
      null,
      "Hello React!"
    );
  }
}

ReactDOM.render(React.createElement(Component),
document.getElementById('html-element'));
```

Seems more complex & ugly ;(

Output of both components: Hello react!

JSX

```
class TheCoolerComponent extends React.Component {
  render() {
    return (
      <div>
        Hello React!
      </div>
    );
  }
}

ReactDOM.render(
  <TheCoolerComponent />,
  document.getElementById('html-element')
);
```

Simple and easier! :D
It's pretty similar to HTML

What do you prefer? I hope the JSX one.

2

Props & State

Props

- Props is short for properties.
- Props are passed into the component. However, components can also have default props.
- A component should not change its props.
- We can use the **state** of a component to define the **props** of another.

Class component with props

```
class Component extends React.Component {
       render() {
          return (
            <div>
              Hello {this.props.msg}
            </div>
          );
        }
}
```

Function component with props

```
const Component = (props) => {
      return <div>Hello {props.msg}</div>;
}
```

Default props definition

```
Component.defaultProps = {
      msg: “React”;
};
```

State

- If we want interactivity we will need to use the **state** of the component.
- The state of a component is initialized in his **constructor**.
- If we want to change the state of a component, we HAVE to use **setState()** function. (Do you remember the component lifecycle “updating” phase?)
- State changes are asynchronous!!!!

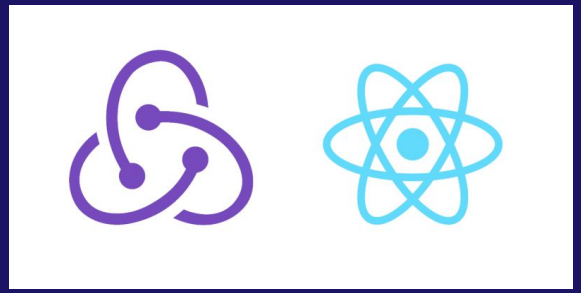
It’s better to explain all these concepts with an example!

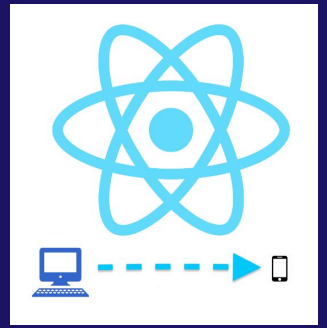
3

“To infinity and beyond”

What to do now?

- React docs are really good.
- Android/iOS apps: Research about react-native.
- Server side rendering: Redux, Next.js, Next.js+Express (NodeJS).
- Find libraries/frameworks that have react components, for example:
  - https://react-bootstrap.github.io/ (We have seen this in action today!).
  - https://www.react-most-wanted.com/
- And don't stop learning! "To infinity and beyond!".





THANKS !

CREDITS: This presentation template was created by Slidesgo, incluiding icons by Flaticon, and infographics & images by Freepik.