Class 13: React States

Announcements

- Stay posted on Piazza for Exercise 3
  - Will be short
- Wednesday’s class will be online as well
- Office hours will not change
- Exercise 4 posted tomorrow

Problems with Props

- You cannot change your props!

- Let’s try to update the date on the screen and see if props will update.
    - Spoiler: props will not automatically trigger changes to the view!

Trying to Update the DOM

```
function tick(props) {
    return (
      <div>
        <h1>Hello, world!</h1>
        <h2>It is {props.date.toLocaleTimeString()}.</h2>
      </div>
    );

  }
  ReactDOM.render(
    <Tick date = {new Date()}/>,
    document.getElementById('root')
  );
  //setInterval(tick, 1000);
```

Try #2

```
function tick() {
  const element = (
    <div>
      <h1>Hello, world!</h1>
      <h2>It is {new Date().toLocaleTimeString()}.</h2>
    </div>
  );
  ReactDOM.render(
    element,
    document.getElementById('root')
  );
}

setInterval(tick, 1000);
```

Example from: ReactJS.org

Props

- Read only
- Used to pass data
- Can be used to define an initial state value
    - As long as the initial state does not require being updated

State

- Data that is created and managed by the component
- Renderings are updated when states are updated
- Can be updated and changed
- Works asynchronously

State

this.state and the setState() method will actually dynamically update the view -- without us having to force the DOM to update.

By updating the view, React will see changes to the virtual DOM and update the actual DOM.

Rules:

- Do not directly modify the state data, use setState()
- Changes to state are merged
- Treat changes asynchronously!

Creating states

In your constructor use the following format:

```
this.state = {
    data: 0
}
```

Should look eerily similar to creating an object in JS.

Creating states

- Make sure you create state data in the constructor.
- This should be the only place you directly modify the state data (more about that later)
- Any changes you make to the state will be merged with original data

Updating State Guidelines

- Use setState() inside of a function

- Use arrow functions to bind to current object
  - () => {}
  - (prevState, props) => {}

- If you are trying to trigger a function (with a button for example) use arrow functions to call.

Let’s make a State example

I would like to create a button that updates its state of a counter….Let’s try it!

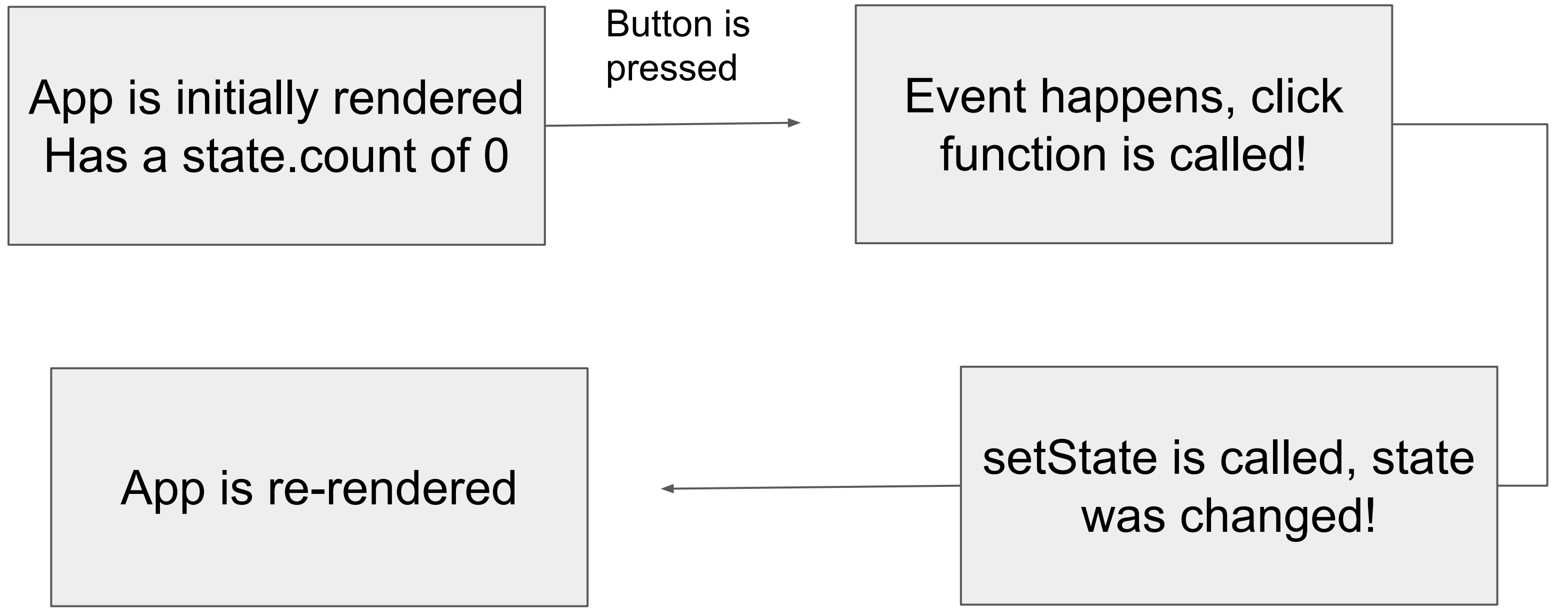
Updating State Guidelines

Either:

- Be consistent about using arrow functions, both in:
  - setState()
  - When the function is called

- Use this.funcName = this.funcName.bind(this)
  - If you don’t want to use () to call the function
  - I generally recommend using arrow functions

What is really happening?


App is initially rendered
Has a state.count of 0
Button is pressed
Event happens, click function is called!
setState is called, state was changed!
App is re-rendered

WTWAW (What To Walk Away With)

- Create a state in a class component
- Know why we use arrow functions
- Create an app where a button triggers a state data update