Intro to React.js

Data: 11/3/2019

Follow us on social media!

Website: webdvt.org
Instagram: @WebDVT
Facebook: @WebDVT

What is React?

React is a Javascript library created by Facebook and is used for building user interfaces (UIs) and front-end applications

React is often called a framework because of its behavior and capabilities

React is the most popular framework in the industry (for now)

Why use it?

- Makes **front-end** JavaScript much easier
- Uses self contained, independent **components** with their own **state**
- Much more interactive UIs
- Virtual DOM
- **JSX** - Easily incorporate JS in markup
- Easy to work with teams

What should you know before learning React?

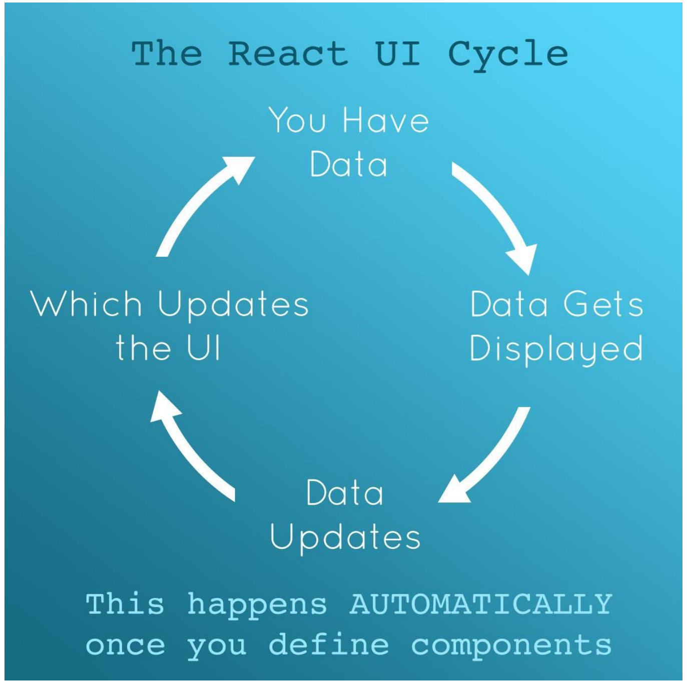
JavaScript Fundamentals (Objects, Arrays, Conditionals, etc)

It may help to learn these first

- Functions
- Destructuring
- High Order Array Methods - forEach, map, filter
- Arrow Functions
- JavaScript Promises

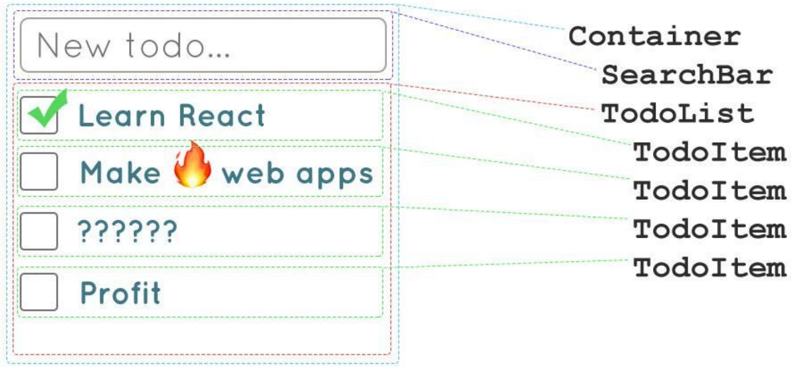
How React Works

1. You display data on a webpage
2. A user interacts with it
3. Now the data changes..
4. ..and you want the webpage to look different

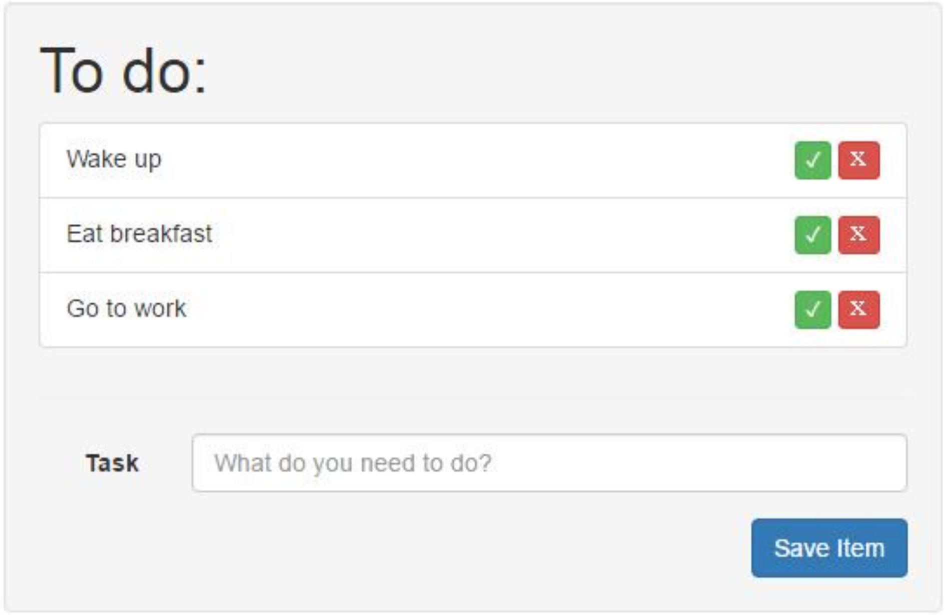


To think in React

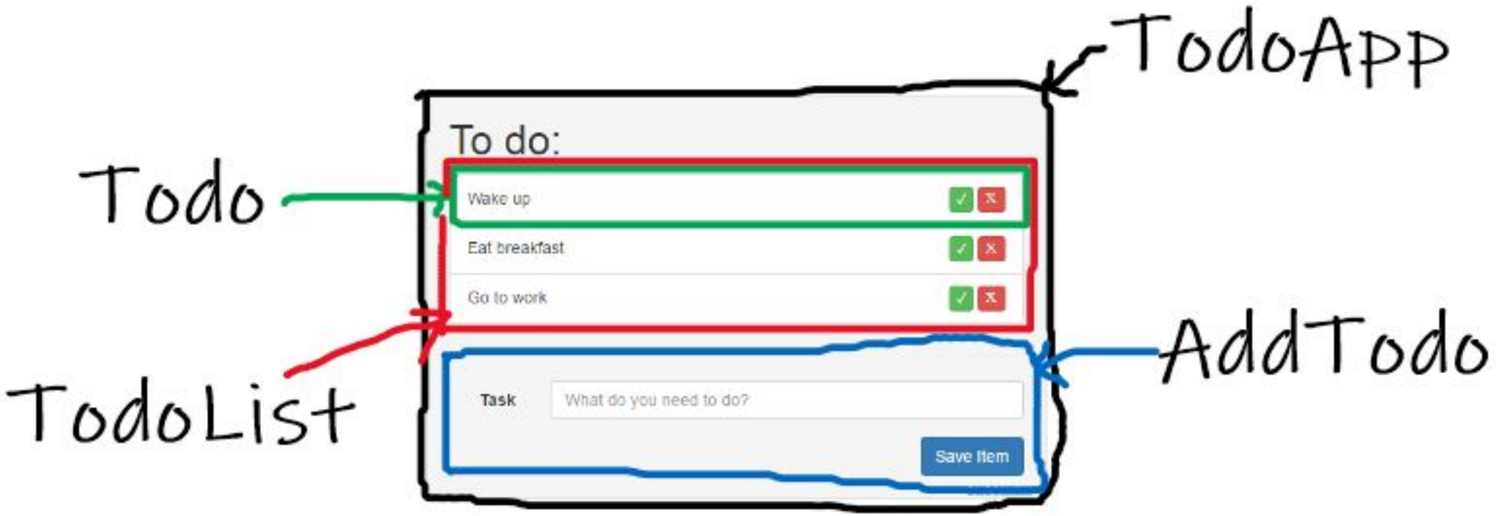
1. Break your **UI** into custom **components**
2. Each component is responsible for displaying itself, based on the external and the internal data available.
3. Build **trees** of these **components** for a full UI



Simple Todo-List..

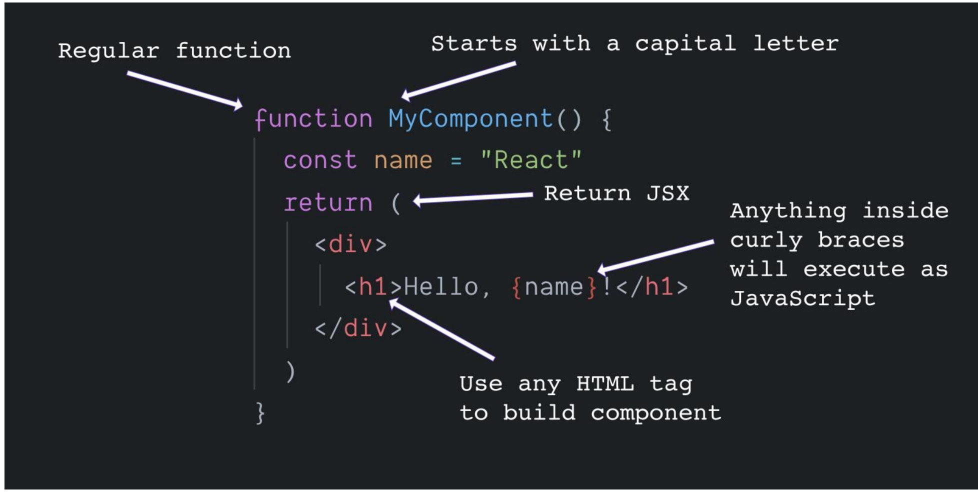


Thinking in Components



React Component

- **Components** are functions that return **JSX**
- JSX looks like HTML but is actually JavaScript
- Inside of JSX use curly braces to contain JS

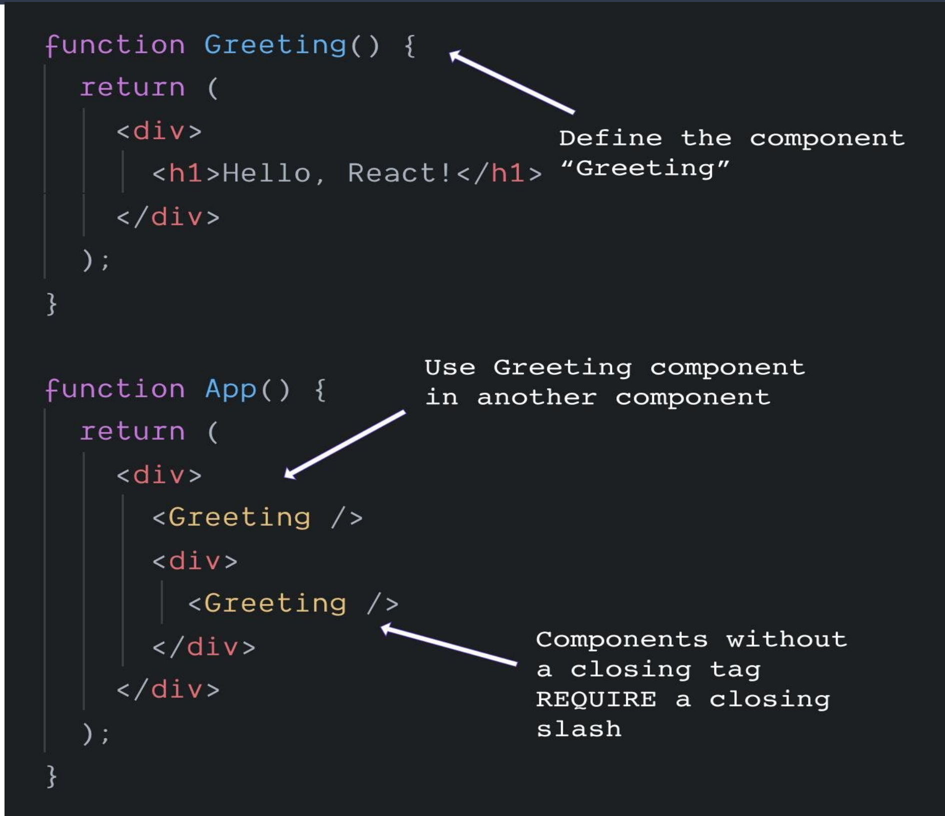


```
function MyComponent() {
  const name = "React"
  return (
    <div>
      <h1>Hello, {name}!</h1>
    </div>
  )
}
```

Regular function
Starts with a capital letter
Return JSX
Anything inside curly braces will execute as JavaScript
Use any HTML tag to build component

Continue

- After defining a **function component**, you can use it within another component
- Use this method to build a tree of components that defines your entire UI



```
function Greeting() {
  return (
    <div>
      <h1>Hello, React!</h1>
    </div>
  );
}

function App() {
  return (
    <div>
      <Greeting />
      <div>
        <Greeting />
      </div>
    </div>
  );
}
```

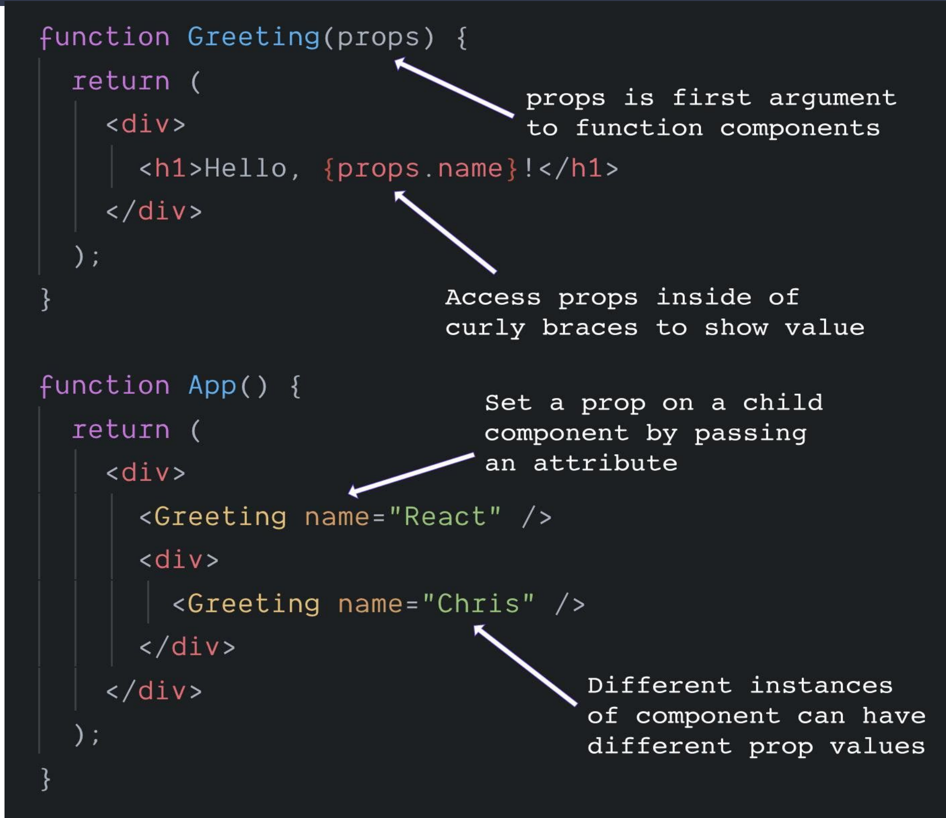
Define the component "Greeting"

Use Greeting component in another component

Components without a closing tag REQUIRE a closing slash

What is Props?

- Data that comes in from outside of a component is called **props** (properties)
- This can be **passed** from a parent to a child through JSX attribute
- Props come into function components as the first **argument** to the function



```
function Greeting(props) {
  return (
    <div>
      <h1>Hello, {props.name}!</h1>
    </div>
  );
}

function App() {
  return (
    <div>
      <Greeting name="React" />
      <div>
        <Greeting name="Chris" />
      </div>
    </div>
  );
}
```

props is first argument to function components

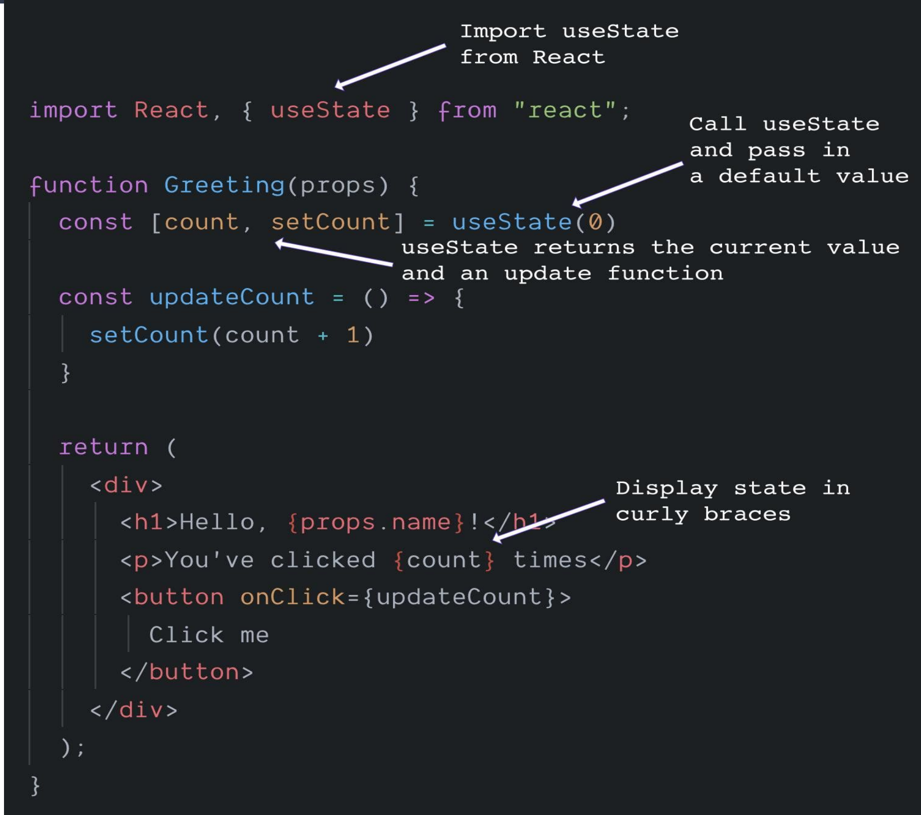
Access props inside of curly braces to show value

Set a prop on a child component by passing an attribute

Different instances of component can have different prop values

What is State?

- Internal, changeable data is called “**state**”
- State is defined by the useState() function, which returns the data, and a function to change that data (in an array).
- NEVER set the state variable **directly** - always use that function (Why? → Next slide..)

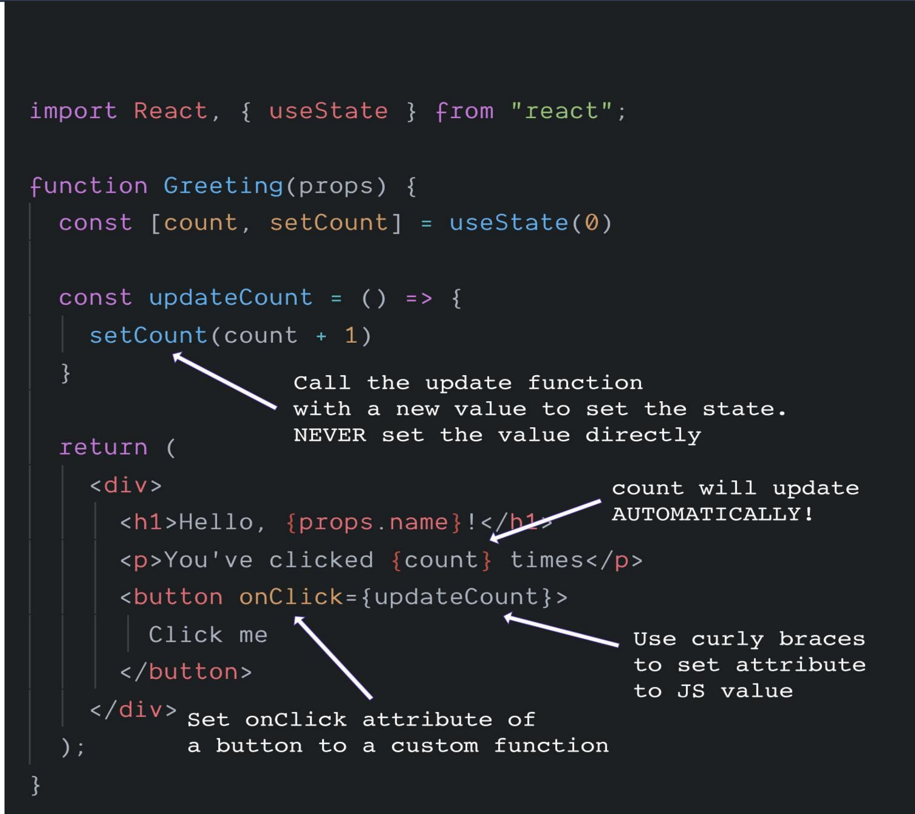


```
import React, { useState } from "react";

function Greeting(props) {
  const [count, setCount] = useState(0)

  const updateCount = () => {
    setCount(count + 1)
  }

  return (
    <div>
      <h1>Hello, {props.name}!</h1>
      <p>You've clicked {count} times</p>
      <button onClick={updateCount}>
        Click me
      </button>
    </div>
  );
}
```

Import useState from React
Call useState and pass in a default value
useState returns the current value and an update function
Display state in curly braces

How React Works

- When state or props **change**, your component updates AUTOMATICALLY 🎉
- ⭐This is the magic of React! ⭐



```jsx
import React, { useState } from "react";

function Greeting(props) {
  const [count, setCount] = useState(0)

  const updateCount = () => {
    setCount(count + 1)
  }

  return (
    <div>
      <h1>Hello, {props.name}!</h1>
      <p>You've clicked {count} times</p>
      <button onClick={updateCount}>
        Click me
      </button>
    </div>
  );
}
```

Call the update function
with a new value to set the state.
NEVER set the value directly

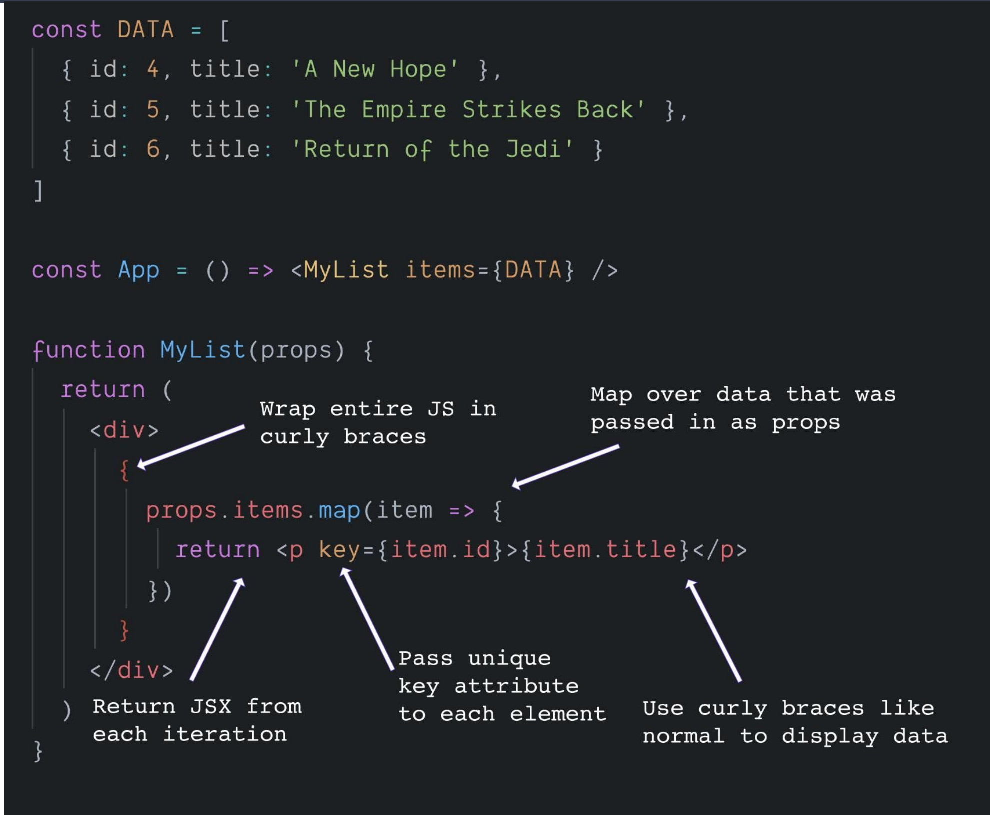
count will update
AUTOMATICALLY!

Use curly braces
to set attribute
to JS value

Set onClick attribute of
a button to a custom function

Iterating List

- Make **list** of things by looping over an array of data with **map()** function
- Return an element from each loop iteration
- Provide a unique key to each element in the list to ensure best performance



```
const DATA = [
  { id: 4, title: 'A New Hope' },
  { id: 5, title: 'The Empire Strikes Back' },
  { id: 6, title: 'Return of the Jedi' }
]

const App = () => <MyList items={DATA} />

function MyList(props) {
  return (
    <div>
      {
        props.items.map(item => {
          return <p key={item.id}>{item.title}</p>
        })
      }
    </div>
  )
}
```

Wrap entire JS in curly braces

Map over data that was passed in as props

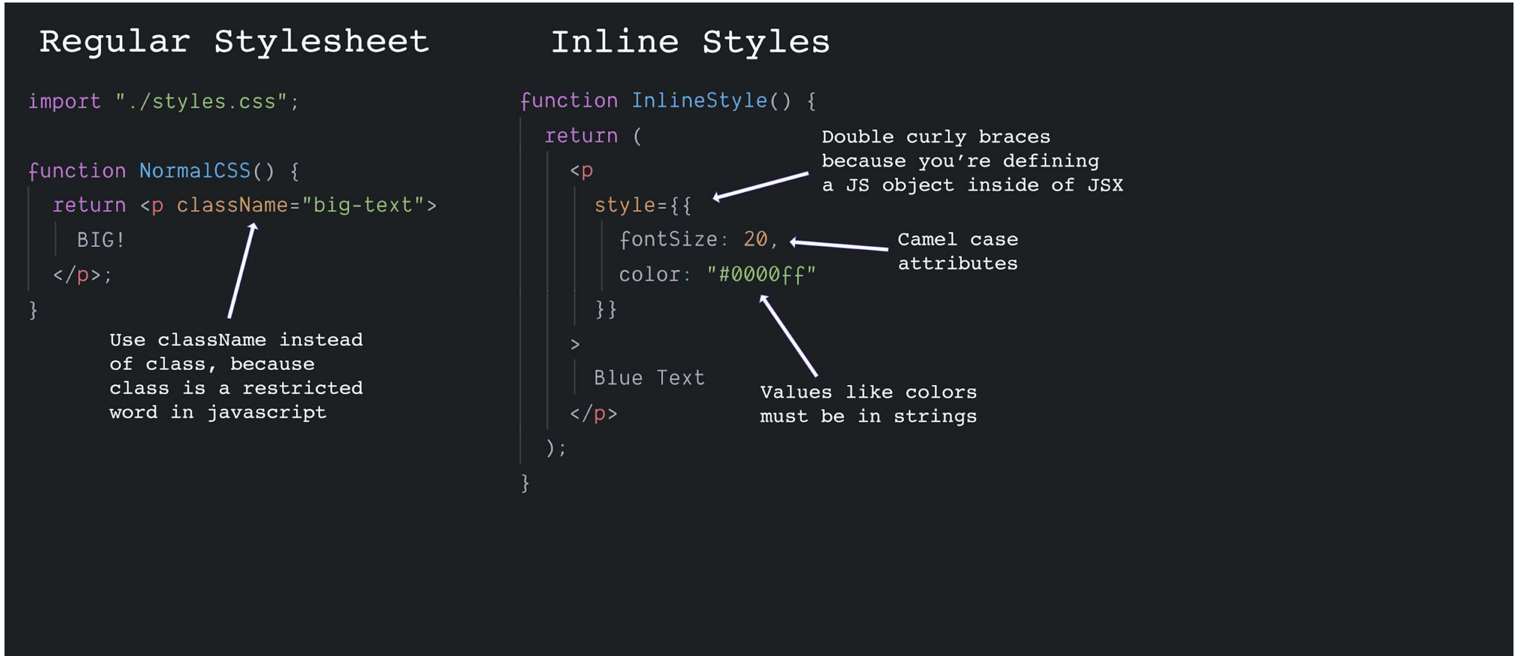
Return JSX from each iteration

Pass unique key attribute to each element

Use curly braces like normal to display data

How to add CSS

- Two build-in ways to **style** components



Regular Stylesheet

```
import "./styles.css";

function NormalCSS() {
  return <p className="big-text">
    BIG!
  </p>;
}
```

Use className instead of class, because class is a restricted word in javascript

Inline Styles

```
function InlineStyle() {
  return (
    <p
      style={{
        fontSize: 20,
        color: "#0000ff"
      }}
    >
      Blue Text
    </p>
  );
}
```

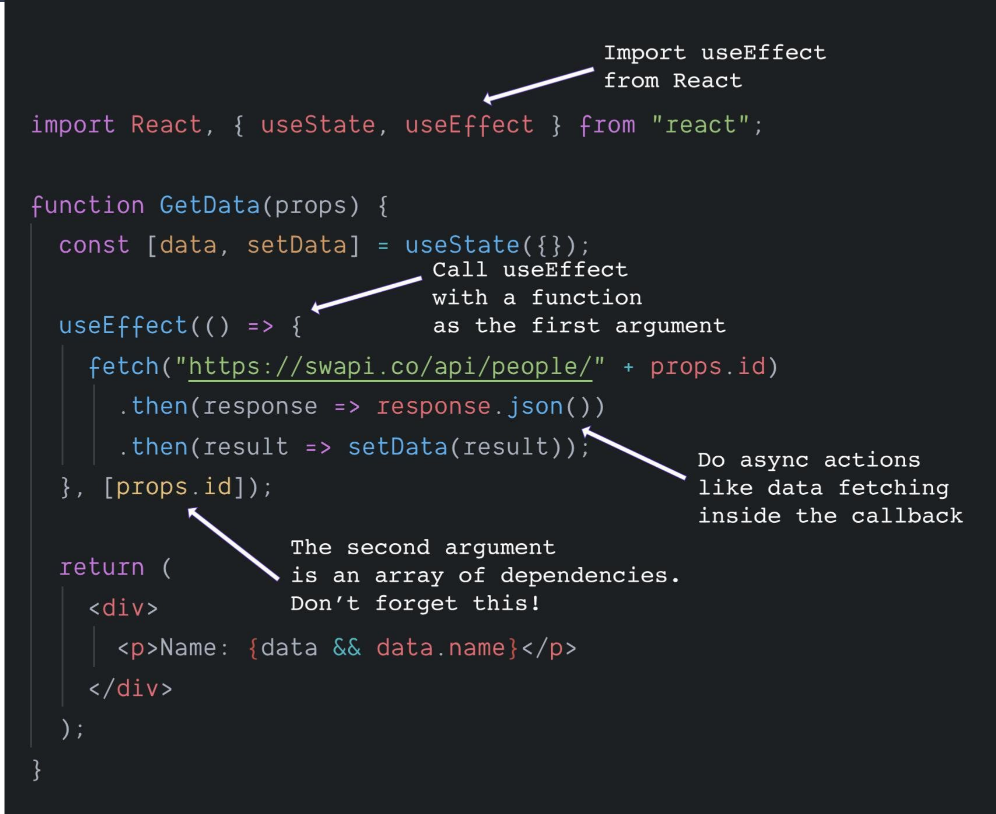
Double curly braces because you're defining a JS object inside of JSX

Camel case attributes

Values like colors must be in strings

Async in React

- Perform **Async** functions and side effects inside of useEffect() (takes a callback)
- The second argument is an array of dependencies
- **Include** any variable the useEffect() uses that might change, or an empty array if there are none



```
import React, { useState, useEffect } from "react";

function GetData(props) {
  const [data, setData] = useState({});

  useEffect(() => {
    fetch("https://swapi.co/api/people/" + props.id)
      .then(response => response.json())
      .then(result => setData(result));
  }, [props.id]);

  return (
    <div>
      <p>Name: {data && data.name}</p>
    </div>
  );
}
```

Import useEffect from React
Call useEffect with a function as the first argument
Do async actions like data fetching inside the callback
The second argument is an array of dependencies. Don't forget this!

How to get started with a React App?

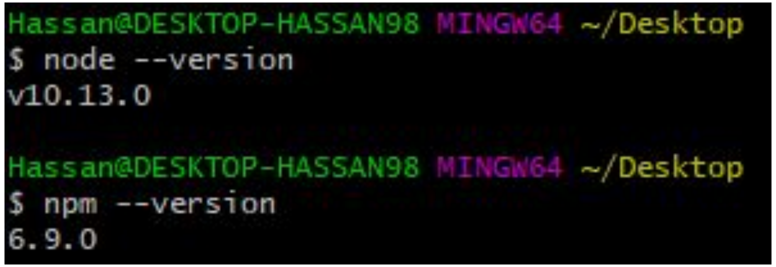
- **Create React App** CLI (Command Line Interface)
- Comes bundled with a **dev server** with auto reload

```
npx create-react-app my-app
cd my-app
npm start
```

Before we get started

- Install **Node.js**

- Check that you have **Node.js** and **NPM** installed



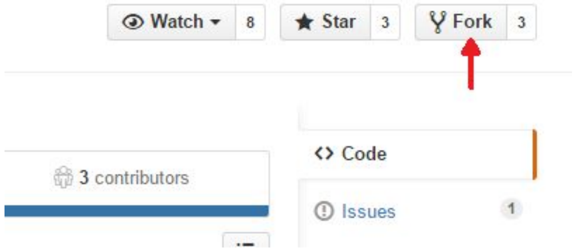
```
Hassan@DESKTOP-HASSAN98 MINGW64 ~/Desktop
$ node --version
v10.13.0

Hassan@DESKTOP-HASSAN98 MINGW64 ~/Desktop
$ npm --version
6.9.0
```

To get started..

1) **Visit** this Github repos:

https://github.com/webdvt/react-todolist

2) **Fork** this repository:



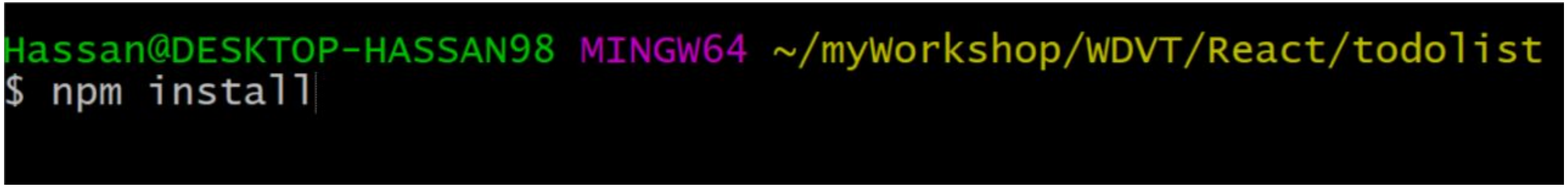
3) **Clone** (your version of the) repos by typing **"git clone** https://github.com/<YOUR_USERNAME>/react-todolist" in your terminal

4) Go into your cloned directory with **"cd react-todolist"**

5) Check out the "**starter**" branch by tying **"git checkout starter"** in your terminal

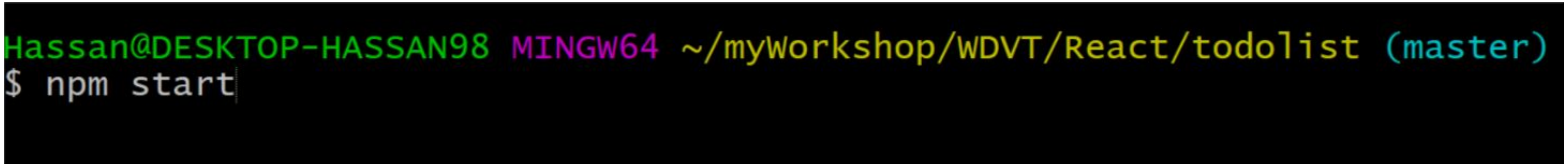
Continue

Install the npm packages listed in “**package.json**” file by executing this command:



```
Hassan@DESKTOP-HASSAN98 MINGW64 ~/myWorkshop/WDVT/React/todolist
$ npm install
```

After that, let’s try running our application:



```
Hassan@DESKTOP-HASSAN98 MINGW64 ~/myWorkshop/WDVT/React/todolist (master)
$ npm start
```

Reference

@chrisachard “Learn React in 10 tweets”:

https://twitter.com/chrisachard/status/1175022111758442497?s=20