WEEK 2
INTRO TO REACT

PROFESSIONAL & CONTINUING EDUCATION
UNIVERSITY *of* WASHINGTON

LEARNING OBJECTIVES

DEVELOPERS WILL BE ABLE TO...

> Compare and contrast single-page applications (SPAs) and multi-page applications

> Create a React app using functional components

> Utilize props to render data in React

> Create event handlers in React

> Use conditionals and loops to render JSX elements

> Design a user interface in the context of React components, state, and props

WEEK 1 ASSIGNMENT TOPICS

- JavaScript modules
  - ECMAScript (import, export) + Node + Babel
- Homework pipeline
  - git, Github, and pull requests (PRs)
  - CI
  - Jest and ESLint
- Q&A

INTRO TO REACT DISCUSSION

- Let’s discuss your experiences with React
  - What did you like?
  - What did you dislike?
  - Did you run into any challenges? How did you overcome them?
  - How does writing React code compare to writing jQuery code?
  - What is state? What are props?

THE SINGLE-PAGE APPLICATION (SPA)

SINGLE-PAGE APPLICATIONS (SPA) WHY?

- Before SPAs, front-end applications mostly consisted of multiple pages and were often rendered server-side
  - Server fetches data, interpolates it into HTML templates, then sends it to the client
- SPAs load a single page + JS + CSS
  - Change content using JavaScript and fetch data as needed from a server (usually JSON/XML)
  - *Can* provide benefits like speed after first load, decoupling from a server, better caching, and fast deployments
  - Drawbacks can include bad SEO, memory leaks, security issues

SINGLE-PAGE APPLICATIONS (SPA)
WHY REACT?

- React is a JavaScript framework for building SPAs
- Created by Facebook
- Used widely by the industry
  - Redfin, SAP Concur, Amazon, PayScale, Microsoft
- Provides the “view” of MVC, allowing it to be composed easily with other libraries
- Large community + library support

10,000 FOOT VIEW OF REACT

- Implements a virtual DOM
- It’s just the UI
  - Each component’s goal is to add some HTML to the DOM
  - We can add additional concepts (event handling, state) to interact with the UI
- Data flows one way
  - All data “comes from the top” and “flows down” into the components.
- Components are declarative
  - A component is rendered by the React API based on its definition, state, and props

REACT COMPONENT

```
// class
import React from ‘react’;
import PropTypes from ‘prop-types’;

function Picture({ src }) {
  return <img src={src} />;
}

Picture.propTypes = {
  src: PropTypes.string.isRequired
};
```

REACT COMPONENT

```
// class
import React from ‘react’;
import PropTypes from ‘prop-types’;

const Picture = ({ src }) => (
  <img src={src} />
);

Picture.propTypes = {
  src: PropTypes.string.isRequired
};
```

CREATE-REACT-APP

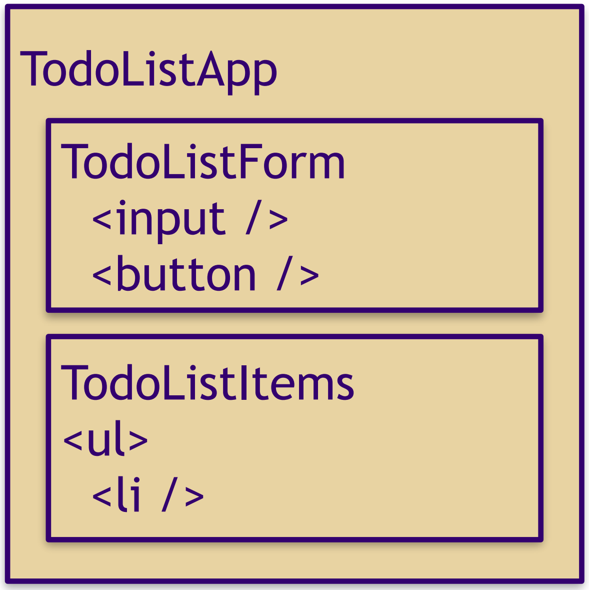
- https://github.com/facebook/create-react-app
- We can use create-react-app to setup an app without having to deal with configuration (much of which is outside the scope of this class)
- Provides support for the build, testing, “hot reloading”, CSS, images, files, and more.
  - https://create-react-app.dev/docs/getting-started

LET'S MAKE A REACT APP TODO-LIST

- We're going to make a todo-list
- Requirements
  - Should display an input field to enter an item
  - Should display a button to add an item
  - Should display items to do

LET'S MAKE A REACT APP TODO-LIST

- Design
  - Components
  - What are the props and state?
  - Where should state live?
- Implementation
  - Build out static UI
  - Fill in data via props
  - Fill in state + event handlers



TRY IT OUT

- Create a component called TodoItems
  - It should have one prop, named “todos”
  - Define propTypes for the component
    - “todos” should be an array of strings
  - It should loop over the todos array and render each todo as a list item
  - The rendered list items should be in an unordered list element

ADDING STATE

```
import React, { useState } from ‘react’;

const TodoList = () => {
  const [itemToAdd, setItemText] = useState(‘’);

  const updateInput = (e) => {
    setItemText(e.target.value);
  };

  return (
    <div>
      <input
        value={itemToAdd}
        onChange={updateInput}
      />
    </div>
  );
}
```

ADDING STATE

```
import React, { useState } from ‘react’;

const TodoList = () => {
  const [itemToAdd, setItemText] = useState(‘’);
  const [todos, setTodos] = useState([]);

  const updateInput = (e) => setItemText(e.target.value);

  const addItem = (e) => {
    e.preventDefault();
    setTodos([…todos, itemToAdd]);
    setItemText(‘’);
  };

  return (
    <div>
      <form onSubmit={addItem}>
        <input
          value={itemToAdd}
          onChange={updateInput}
        />
        <button type="submit">Add Item</button>
      </form>
    </div>
  );
}
```

SUMMARY

- React is a library for making single-page applications
- React is designed around **components**. Components are defined and rendered in order to display data.
  - Components can be functions or classes
    - We'll talk about classes at a later time
  - Components can have static data (props) or dynamic data (state)
- Props are controlled by parent components
- State is internal and controlled by the component itself

FOR NEXT WEEK

Assignment and Research

> Complete Week 2 assignment

> Get familiar with React state via React's documentation and other online sources. Try to answer the questions on the next slide.

> > https://reactjs.org/docs/hooks-overview.html

FOR NEXT WEEK

React State + Hooks - Let’s answer these questions

- What are hooks?
- Why do we lift state “up” to a parent component in React?
- What does the useState hook do?
- What does the useEffect hook do?
- How can state from a useState hook be shared with other components?
  - Can state be shared to a parent component?
  - Can state be shared to a child component?
  - Can state be shared to a sibling component?