CMSC335

Web Application Development with JavaScript

React Introduction

Department of Computer Science

University of MD, College Park

Slides material developed by Ilchul Yoon, Nelson Padua-Perez

React Intro

- **React** - **Library** for implementing user interfaces
  - https://react.dev/
- Created by Facebook
- Used to implement single-page applications
  - Single-page app – the page does not need to be completely reloaded when a new state is displayed
- React follows a component approach to build an interface
  - The application is composed of building blocks
- A component is defined using a function that returns the structure of the component
- React has two major APIs
  - React Component API - Parts of the page rendered by ReactDOM
  - ReactDOM - API that renders on a web page
- React can target more than the browser as it abstract the target rendering environment from the components
  - React Component → React DOM → Browser
  - React Component → React Native→ Mobile App
  - React employs a virtual DOM
- Reach employs a virtual DOM

JSX

- JSX is a syntax extension for JavaScript that allows you to create components and elements relatively easily. A component is defined using a function
- Example of a component named **Hello** (using JSX):

```
function Hello({ name }) {
  return <h2>Hola {name}!</h2>;
}
```

- **Example:** HelloComponent
  - To run open **index.html** using Live Server
  - Not using Node.js
- **JSX**
  - JSX (JavaScript XML) - XML/HTML markup used to create React components
  - Language used to describe user interfaces that rely on React
  - Tools like Vite and Babel transform JSX into JavaScript
- React
  - Supports standard HTML tags – When writing JSX, you treat them as regular HTML elements
  - Supports all standard events (onClick, onChange, onSubmit, etc.)
  - **Example:** StandardComponents

Using Vite

- You can use Vite to create an environment for developing a React App
- **npm i vite**
- **npm init vite folderName** (e.g., npm init vite CarDealership)
  - Select React
- **cd CarDealership**
- **npm install**
- **npm run dev**
  - Starts the server, and changes you make will be reflected automatically
  - Select the local link (e.g., http://localhost:5173/)
- **File Organization**
  - **index.html -** defines the root element (div id="root") where the app will be rendered
  - **src** folder - has your components and support files for those components
    - » App.jsx - the component you would like to display
    - » App.css - style information for the component
    - » index.css - included by main.jsx
    - » main.jsx - calls the render() method that will display the app
      - You could remove the <StrictMode></StrictMode> to see better the role of render()
- The displayed page will be refreshed automatically if you edit the source files
- **npm run build**
  - To deploy (dist folder will be created)

App: University

- **Example:** University (notice we removed some of the default files/folders created by Vite)
  - To run app
    - » **cd university**
    - » **npm i**
    - » **npm run dev**
- We use functions to define components. To explore components, let's update the root.render() method in in main.jsx to include components one at a time. Notice how we use import to import the components
- **Example:** Campus.jsx, CampusCity.jsx
- Parentheses in the **return** statement
  - Groups multiple JSX elements into a single expression to return
  - Avoid issues with JavaScript automatic semicolon insertion
- **The return structure of a component can only have one root element**
- **Example:** Student.jsx - We can pass parameters (props, from properties) to a component. We use { } to enclose JavaScript expressions in JSX. The parameter will be defined using { }, and when we need access to a property in JSX, we will use { }

App: University

- **Example:** Faculty.js - JSX can render arrays we create using JavaScript (e.g., using map). Also, we can add an event handler to a button. To add comments in a component, include the comments in { /* a comment */ }. Any variable referenced in JSX (e.g., a local variable) must also be included in { }
- **Example:** University.js
  - Remember that tags (lowercase) in JSX are not HTML elements; they are React elements
  - React fragment (<></>) is like an invisible container that groups a list of child elements without creating a DOM node (e.g., we can avoid an unnecessary)
- **Deployment**
  - **npm run build**
    - » **Creates a dist folder**
    - » You will need to adjust path (e.g., /assets to ./assets in index.html)

React Components vs React Elements

- React Components - start with an uppercase
- React Elements - (e.g., <h1>) start with a lowercase

Create React App Command Line Tool

- **Create React App** is a command-line tool that sets up a basic React project structure, a development server, and required dependencies. It is no longer recommended as there are better alternatives (e.g., vite)
  - See https://react.dev/blog/2025/02/14/sunsetting-create-react-app

Using Vite

- You can use Vite to create an environment for developing with vanilla JavaScript
  - **npm init vite VanillaExample**
    - » Select Vanilla
    - » Select JavaScript
  - **npm run dev**
- **File Organization**
  - **src** – folder with your js and css files
    - » main.js has your entry point to your code
  - index.html - includes src/main.js
- Deployment
  - **npm run build**
    - » **Creates a dist folder**
    - » You will need to adjust path (e.g., /assets to ./assets in index.html and vite.svg in .js file)
      - VanillaExampleDist has the paths updated
    - » Due to CORS, you need to run using a web server (e.g., Live Server)

Reference

- React, The Comprehensive Guide (https://www.sap-press.com/react_5705/)
- React and React native, Fifth Edition, Mikhail Sakhniuk, Adam Boduch (https://www.packtpub.com/en-ic/product/react-and-react-native-9781805127307)