WEB TECHNOLOGIES

React.JS – Components

Vinay Joshi
Department of
Computer Science and Engineering

React.JS – Components

Introduction

- Need for Components
  - There may be elements which are similar
  - A need to make a change will result in changes in multiple places
  - Similar to functions, if we could write code related to the element in one place, changes can be minimized
- Solution - Reusable piece of JavaScript code that output (via JSX) HTML elements

React.JS – Components
First Component

- Using the ***class*** syntax, create the component as follows

```
class HelloWorld extends React.Component {

 }
```

- Add a render method to the class that returns JSX

```
class HelloWorld extends React.Component {
    render() {
        return <p>Hello, componentized world!</p>;
    }
}
```

React.JS – Components

Adding Component

- Add the JSX in the render method with a element with the tag name as the Component name

```
ReactDOM.render(
    <HelloWorld />,
    document.querySelector("#container")
);
```

React.JS – Components

Parameterized Components

- The component added earlier is still a hardcoded component which returns a paragraph with text “Hello, componentized world!”
- To modify the component to accept ***properties*** (or parameters),
  - we need to accept properties in the component and
  - pass these properties while creating the components

React.JS – Components

Parameterized Components…(cntd.)

- To accept properties, modify the component as follows

```
class HelloWorld extends React.Component {
    render() {
        return <p>Hello, {this.props.greetTarget}!</p>;
    }
}
```

- Here ***greetTarget*** is the property that needs to passed during creation of the component

- Expression within { and } would be evaluated in JSX

React.JS – Components

Creating Parameterized Components

- Add the greetTarget attribute or property while creating the component

```
ReactDOM.render(
    <div>
        <HelloWorld greetTarget="Batman"/>
        <HelloWorld greetTarget="Iron Man"/>
    </div>,
    document.querySelector("#container")
);
```

React.JS – Components

Components with Child Elements

- A Component can have child elements under it like any HTML element.

  For example,

```
<MyComponent foo="bar">
    <p>Something!</p>
</MyComponent>
```

- Like earlier, we can access this.props.foo
- The child elements can be accessed using this.props.children. This contains the text as a single property. (Note: it is not an array)

React.JS – Components

Components with Child Elements…(cntd.)

```
<MyComponent foo="bar">
    <p>Something!</p>
</MyComponent>
```

- Can be translated into a div as follows

```
class MyComponent extends Render.Component {
    render(){
        return (
            <div class = {this.props.foo}>
                {this.props.children}
            </div>
        );
    }
}
```

THANK YOU

Vinay Joshi

Department of Computer Science and Engineering

vinayj@pes.edu

+91 80 2672 6622