Props

Props, short for properties, is an empty *object* that is passed to every React component by *default*. To pass props down to our child components, we simply need to add an html *attribute* to our component. Each *attribute* will become a *key* in the props object and each *value* given to that *attribute* will become the *value* of that *key*.

To access props in a class component, we will need to precede the *props with the keyword this. Because our class components extend React.Component, we already have access to the props variable that React has defined in the Component class.*

class Header extends React.Component {

    render() {

        return (

            <div>

                // By inheriting from React.Component all we need is the "this" keyword in front of props.

         <h1>My name is { this.props.firstName } { this.props.lastName }</h1>

        </div>

        );

    }

}

We could also destructure our class props this way

class Header extends React.Component {

    render() {

*const* { firstName, lastName } = this.props;

        return (

            <div>

       // Destructuring allows us to use them like variables. This is just a small amount of syntactical sugar.

         <h1>My name is { firstName} { lastName }</h1>

        </div>

        );

    }

}

Flow of Data

Because the data flow in React flows downward, we use props all the time to pass data down from component to component. We are not limited in the amount of stuff we can pass down in props. We can pass down anything we want including functions.

**A Note on Curly Braces: In JSX, we use curly braces to denote a Javascript expression. Typically, you can only send down Strings in props. However, with curly braces, we can send Javascript expressions (assuming they are valid). This includes numbers, strings, functions, objects, etc. Let's look at valid ways to pass down props:**

<SomeComponent someProp="test" someOtherProp={ 67 }/> //Valid. We can send normal strings, but in numbers need curly braces

<SomeComponent someProp={ "test" } someOtherProp={ 67 }/> //Valid. A String is still a Javascript expression

<SomeComponent someProp="test" someOtherProp=67/> //Invalid. Numbers need curly braces

When in doubt, use curly braces.