ReactJS Cheatsheet

MuleSoft®
A Salesforce Company

Hello World!

```
ReactDOM.render(
  <h1>Hello, world!</h1>,
  document.getElementById(‘root’)
);
```

React Mounting Lifecycle

`constructor(props)`

Called before the component is mounted. Needs calling super (props) before any other statement

`render()`

Required method in a class component, exists across multiple life cycle phases. Should return a DOM node or another user-defined component

`componentDidMount()`

Invoked immediately after a component is mounted

`componentWillUnmount()`

Invoked immediately before a component is unmounted and destroyed

`componentDidCatch()`

Invoked after an error has been thrown by a descendant component

React Updating Lifecycle

`componentDidUpdate(prevProps, prevState, snapshot)`

Invoked immediately after updating occurs, not called for the initial render

`shouldComponentUpdate(newProps, newState)`

By returning false, the Component exits the Update life cycle and will not execute componentDidUpdate

`render()`

Used to update the elements that are being rendered by the Component

Other APIs

`setState(updater, callback)`

Enqueues changes to the component state and tells React that this component and its children need to be re-rendered with the updated state

`forceUpdate(callback)`

Causes render() to be called on the component, skipping shouldComponentUpdate(). This will trigger the normal lifecycle methods for child components

Styling elements

```
<div className=”class anotherClass” />
```

To specify a CSS class, use the className attribute. This applies to all regular DOM and SVG elements like <div>, <a>, and others

```
<div
  style={{
    margin: 0,
    marginLeft: 0
  }}
/>
```

Style accepts a JavaScript object with camelCased properties rather than a CSS string

Default Props

```
static defaultProps = { name: ‘Max’ };
```

Defined as a property on the component itself, allows to set default values for props

Visit us at **mulesoft.com/careers**

Other Features

```
return (
  <ul>
    {items.map(item => (
      <TodoItem
        item={item}
        key={item.key}
      />
    ))}
  </ul>
);
```

Returns a list of TodoItem components. Always supply a key in order to help React identify which items have changed, are added, or are removed

```
render() {
  return (
    <Fragment>
      <ChildA />
      <ChildB />
    </Fragment>
  );
}
```

Allows to return multiple elements from render

```
<Fragment> {condition ? <ComponentA /> : <ComponentB />} </Fragment>
```

The conditional evaluation allows returning one of two components depending on a given boolean condition

```
<Fragment> {condition && <ComponentA />}
```

The short-circuit evaluation allows returning a component depending on a given boolean condition

```
render() {
  return “No need for component”;
}
```

Renders as a text node in the DOM

Our mission is to help organizations change and innovate faster by making it easy to connect the world’s applications, data, and devices. With our API-led approach to connectivity, **MuleSoft’s** market-leading **Anypoint Platform™** is enabling Spotify, Coca-Cola, Unilever, and over 1,200 organizations in 60 countries to build application networks. Now as part of **Salesforce**, Fortune’s #1 Best Company to Work for in 2018, we have the rocketfuel to dramatically increase our ability to scale and deliver value to our customers, partners, and employees. We are always looking for exceptional people to help us fulfill our mission and make an impact on how the world connects—join us!