



React and Redux: Lesson 2

JSX, State, Lifecycle
methods

William Abboud

JSX Deep Dive

- XML-like syntax to create UI elements
- Compiles down to `React.createElement()` functions
- Needs React in scope
- Capitalize custom components
- Pass JS expressions in props
- Pass strings into props directly
- Booleans attributes default to true
- Spread attributes
- Children
- Comments



Displaying JSON data

```
1 [
2   {
3     "artist": "Eminem",
4     "real-name": "Marshall Bruce Mathers III",
5     "aliases": ["Ken Kaniiff", "Marshall Mathers", "Slim Shady"],
6     "albums": [
7       {
8         "name": "Infinite",
9         "year": "1996"
10      },
11      {
12        "name": "The Slim Shady LP",
13        "year": "1999"
14      },
15      {
16        "name": "The Marshall Mathers LP",
17        "year": "2000"
18      },
19      {
20        "name": "The Eminem Show",
21        "year": "2002"
22      },
23      {
24        "name": "Encore",
25        "year": "2004"
26      }
27    ]
28  }
29 ]
```

```
1 import React from 'react';
2
3 export default function ArtistCard(props) {
4   return (
5     <div>
6       <h2>{props.artist}</h2>
7       <img src={props.photo} />
8       <br />
9       {props.aliases.map(alias => <span>{alias}</span>)}
10      <br />
11      {props.albums.map(album => <span>{album.name}</span>)}
12    </div>
13  );
14 }
```

PropTypes

- Used for documentation
- Evaluates props runtime
- Can remove the need to use `typeof` operator
- Can be used as API documentation

<https://reactjs.org/docs/typechecking-with-proptypes.html>



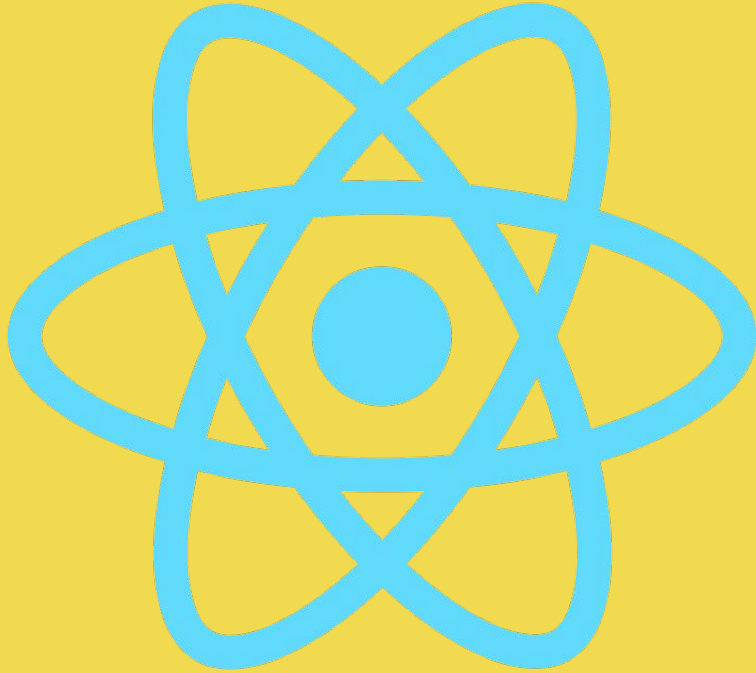
Composition

- **Composition offers the ability to create hierarchies of components**
- **Increases reusability**
- **Easy to test**
- **Easy to isolate bugs or errors**

The background is a high-speed photograph of a blue ocean. In the upper left, a dark, craggy rock face is partially visible. The water is a vibrant blue, with white foam and numerous water droplets captured in mid-air, creating a sense of motion and energy. The lighting is bright, highlighting the texture of the water and the rock.

State

ES6 Classes and React



The image shows the interior of a dome, likely from a classical building. The ceiling is composed of large, light-colored stone or marble blocks. In the center is a circular medallion (oculus) with a sunburst or floral design. Radiating from this center are eight triangular segments, each containing a carved decorative motif. The segments are separated by recessed lines. The dome's structure is further defined by concentric rings of decorative moldings and small, carved figures (putti) at the intersections. The overall appearance is aged and weathered, with some staining and wear visible on the stone surfaces.

Component Lifecycle



Initialization phase

1. **getDefaultProps** - called only once to return default props
2. **getInitialState** - called only once to return an initial state object
3. **ComponentWillMount** - called before render, useless
4. **Render** - returns your markup for rendering, has to be pure
5. **ComponentDidMount** - called last, most frequently used to do AJAX calls, and interacting with the DOM



Update phase - Props

1. **ComponentWillReceiveProps** - used to set state based on upcoming props
2. **ShouldComponentUpdate** - determines renders on props/state update
3. **ComponentWillUpdate** - used to prepare for update (no `setState`) almost useless
4. **ComponentDidUpdate** - basically just like `componentDidMount` but for after update widely used to change the DOM



Update phase - State

1. **ShouldComponentUpdate** - determines renders on props/state update
2. **ComponentWillUpdate** - used to prepare for update (no `setState`) almost useless
3. **ComponentDidUpdate** - basically just like `componentDidMount` but for after update widely used to change the DOM



Unmounting

1. **ComponentWillUnmount** - called before the component is removed from the DOM, used to clear out any side effects such as leftover timeouts, event listeners etc.