Introduction to React JS

Steve Pietrek

October 3, 2017

Steve Pietrek

App Dev Practice Manager
Front End Developer
SharePoint Application Architect

@spietrek
spietrek@gmail.com
Cardinal Solutions
Raleigh/Durham

Presentation References
https://github.com/spietrek/triangle-reactjs-intro-to-reactjs

cardinal

Cardinal provides creative technology solutions that transform client visions into compelling business and customer experiences.

1996 EST. IN

7 LOCATIONS

450 CONSULTANTS

65+M REVENUE

NATIONAL FOOTPRINT

Modern Cloud Apps

Mobile / Digital

Internet of Things

Analytics

Modern Data Center

Collaboration Technology

Microsoft

docker

redhat.

Pivotal

Triangle ReactJS

- I fell in love with ReactJS and the community.
- Allows me to stay active in the React ecosystem and give back to the community.
- Allow people to come together, in a safe environment, and learn about all things React.
- Pair up with other groups in the community (Modern Web Triangle, RTP Angular.js, Women in Tech Allies, Triangle Tech Night).
- Potential meetup formats: Presenter, Lightning Talks, Panel Discussion, Workshops.

Triangle ReactJS Code of Conduct

- Harassment includes offensive verbal comments related to gender, sexual orientation, disability, physical appearance, body size, race, religion, sexual images in public spaces, deliberate intimidation, stalking, following, harassing photography or recording, sustained disruption of talks or other events, inappropriate physical contact, and unwelcome sexual attention.
- Participants asked to stop any harassing behavior are expected to comply immediately.
- If a participant engages in harassing behavior, the meetup organizers may take any action they deem appropriate, including warning the offender or expulsion from the meetup.
- If you are being harassed, notice that someone else is being harassed, or have any other concerns, please contact an organizer immediately.
- Organizers will be happy to help participants contact venue security or local law enforcement, provide escorts, or otherwise assist those experiencing harassment to feel safe for the duration of the meetup. We value your attendance.
- We expect participants to follow these rules at meetup and workshop venues and meetup-related social events.

Future Triangle ReactJS Meetup Topics

- Introduction to React (10/3)
- State Management (11/7)
- Lifecycle Methods
- React 16
- React Router
- React Native
- Preact
- Functional React
- Styles & CSS in JS
- Testing
- Best Practices
- Others?

@trianglereactjs

Upcoming Events

NCDevCon	•October 7-8 •North Carolina's Premier Web & Mobile Conference
Triangle Tech Night Meetup	•October 11 •Augmented Virtual Reality for Today & Beyond •J.A.R.V.I.S. - A Virtual Front Desk •GeoIoT and Location Analytics
All Things Open	•October 23-24 •Open Source Conference
Triangle Modern JavaScript	•October 25 •PWAs & Mobile Apps w Vue.js & Framework7, Cross Platform Apps with React
Women in Tech Allies	•October 18 •Site building basics - Start from scratch with HTML, CSS & JavaScript
Triangle ReactJS	•November 7 •State Management in ReactJS

@triangleReactjs

Topics

ES6

React Overview

JSX

Components

Lifecycle Methods

Create React App

ES6

ES6

Aka ECMAScript2015

New JavaScript implementation

Some browsers support some features, some don’t. Need to transpile using Babel.

Important ES6 features:

- Block-Scoped Constructs Let and Const
- Template Literals
- Multi-line Strings
- Arrow Functions
- Rest Parameters/Spread Operator
- Destructuring Assignment

Block-Scoped Constructs Let and Const

- **var** is hoisted and function-scoped
- **let** is not hoisted and limited in scope to the block, statement, or expression where it is used
- **const** is immutable (read-only reference to a value) and limited in scope to the block, statement, or expression where it is used
- Recommendation: always use **const** until you need to mutate

Construct	Scope	Reassignable	Mutable
const	Block	No	Yes
let	Block	Yes	Yes
var	Function	Yes	Yes

Template Literals

- Avoid using concatenation
- Use backtick

- ES5

```
var name = "My name is " + firstName + " " + lastName
```

- ES6

```
const name = `My name is ${firstName} ${lastName}`
```

Multi-line Strings

- Avoid using concatenation
- Use backtick

- ES5

  "Character cannot be developed in ease and quiet. Only through experience " +

  "of trial and suffering can the soul be strengthened, ambition inspired, and success achieved. "

- ES6

  `Character cannot be developed in ease and quiet. Only through experience of trial and suffering can the soul be strengthened, ambition inspired, and success achieved.`

Arrow Functions

- Traditional function with a **.bind(this)**
- Biggest benefit > **this** will have the same value as the context of the function
- No more **that = this** or **self = this**

- ES5

```
var names = ['Steve', 'Laura'];
var messages = names.map(function (name, index) {
  return 'Name of ' + index + ' element is ' + name + ' '
})
```

- ES6

```
var names = ['Steve', 'Laura'];
var messages = names.map((name, index) => `Name of ${index} element is ${name} `)
```

Rest Parameters/Spread Operator

- Allows an iterable to be expanded
- Use “real” array and can use array methods

- ES5

```
function max() {
  return Math.max.apply(null, arguments);
}
max(5,8,1,18);
```

- ES6 (spread)

```
function max() { return Math.max(...arguments); }
```

- ES6 (rest and spread)

```
const max = (...nums) => Math.max(...nums);
```

Destructuring Assignment

- Breaks down objects and arrays into individual variables

Example 1:

```
const parts = ['shoulders', 'knees'];
const lyrics = ['head', ...parts, 'and', 'toes'];
console.log(lyrics); // ['head', 'shoulders', 'knees', 'and', 'toes']
const [first] = lyrics;
console.log(first); // head
```

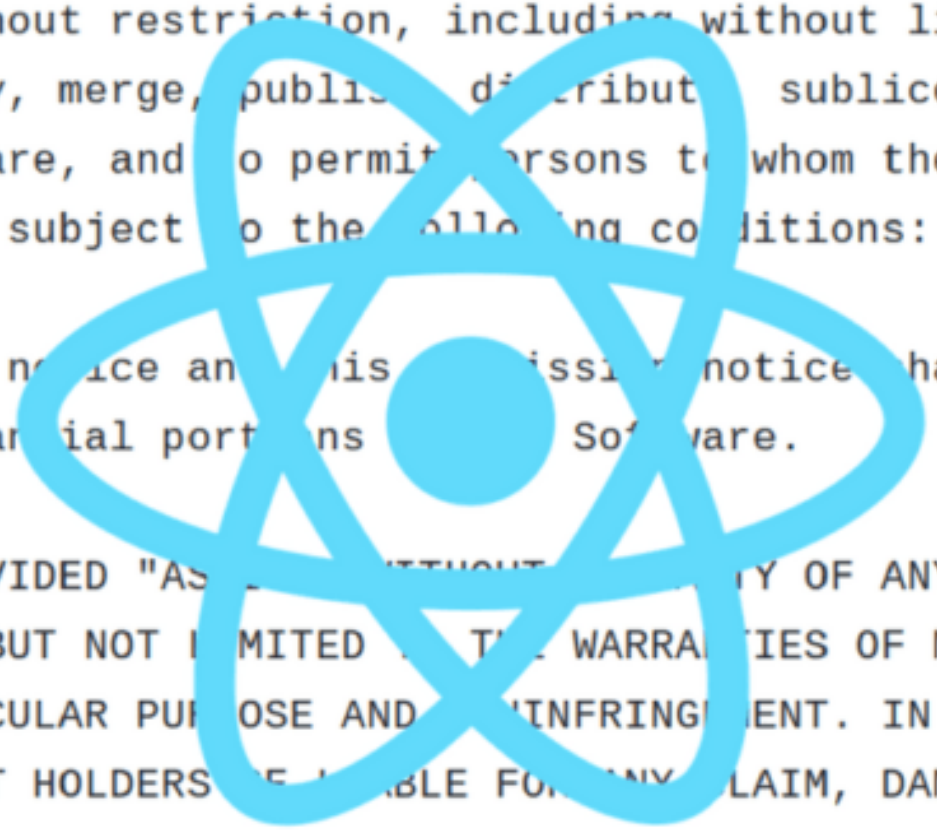
Example 2:

```
const TodoListClearButton = ({ isFetching, todos, onClear }) => (
    isFetching || !todos.length ? null : <button onClick={onClear}>Clear Todos</button>
);
```

React Overview

A JAVASCRIPT LIBRARY FOR BUILDING USER INTERFACES

FOCUSES ON THE “VIEW” IN MODEL VIEW CONTROLLER



Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

MIT License

React Key Concepts

Components

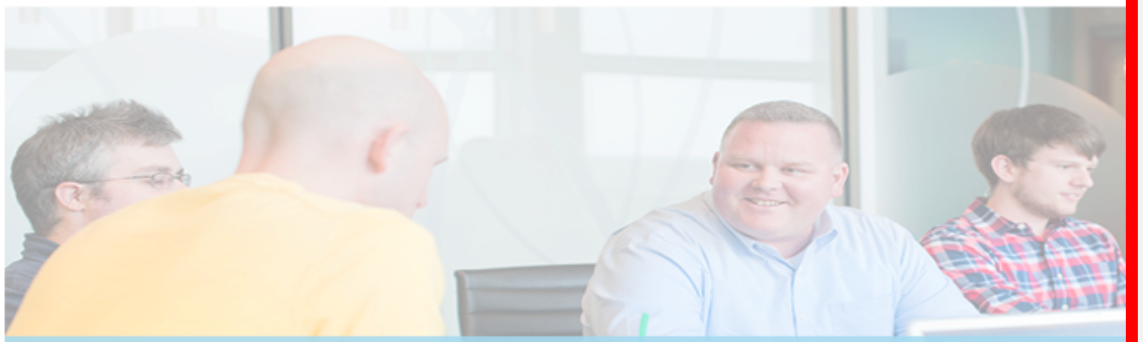
Virtual DOM

EVERYTHING IN REACT IS A COMPONENT

Office 365 | Sites

Drive everyday

Human Resources Dep



Annual Employee Appreciation Celebration Planning i

Click here to see some of the exciting events we have in store for you!

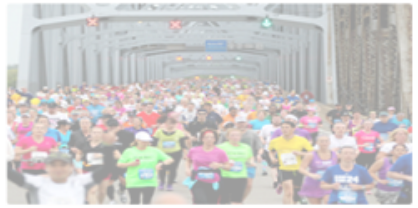
News & Announcements

June 7, 2015

The Benefits of Blogging

Blogging for business is not a new concept, but for many small businesses, especially service businesses, it can be an...

Read More.

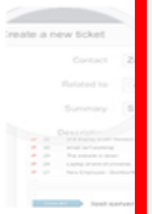


June 17, 2015

Flying Pig 5K Marathon

It's the largest weekend party in town, and you can be part of it! Whether you're a runner or walker, whether you want to...

Read More.



June 23, 20

Automa

We are e nounce a help des that will to your r

Read More.

12 Projects

Red Projects

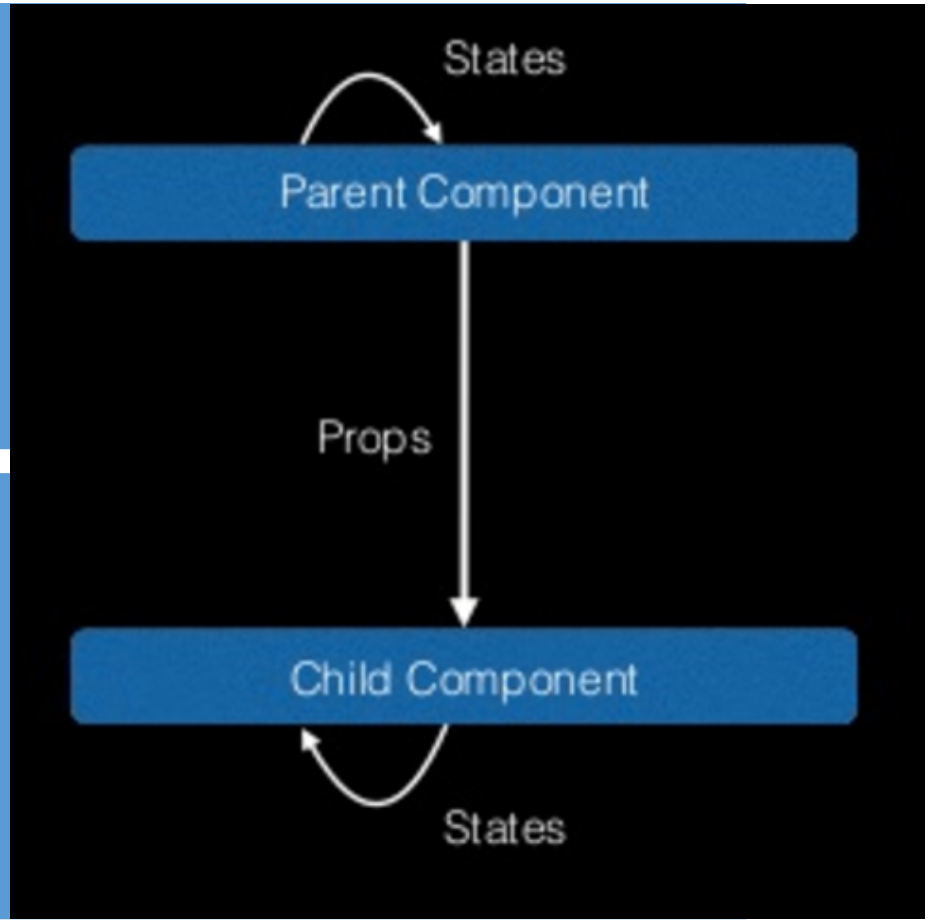
- Firetruck
- Irongate

All Projects

- Bladerunner
- Constantine
- Excalibur
- Firetruck
- Irongate
- Kryptonite
- Mad Hatter
- Moonshine
- Orion
- Revolution
- Sputnik
- Zircon

One Way Data Flow

Data flows down	Props are immutable States are mutable Anything can be passed to children as props
Events flow up	Through parent passed event handler props



Props vs State

	props	**state**
Can use a plain JavaScript object to store data?	Yes	Yes
Can get initial value from parent component?	Yes	Yes
Can be changed by parent component?	Yes	No
Can set default values inside component?	Yes	Yes
Can change inside component?	No	Yes
Can set initial value for child components?	Yes	Yes
Does change trigger render event?	Yes	Yes

Typechecking with PropTypes

TypeScript or Flow can be used to typecheck your **whole** app

PropTypes ensure your props data is valid

A warning will display in the console if an invalid value was provided for a prop

defaultProps is used to set a default prop value when it is not required

```
class Greeting extends React.Component {
  render() {
    return (
      <h1>Hello, {this.props.name}</h1>
    );
  }
}

// Specifies the default values for props:
Greeting.defaultProps = {
  name: 'Stranger'
};

// Renders "Hello, Stranger":
ReactDOM.render(
  <Greeting />,
  document.getElementById('example')
);
```

```
import PropTypes from 'prop-types';

class Greeting extends React.Component {
  render() {
    return (
      <h1>Hello, {this.props.name}</h1>
    );
  }
}

Greeting.propTypes = {
  name: PropTypes.string
};
```

propTypes / defaultProps Example

JSX

JSX

Syntax extension to JavaScript which expresses the Virtual DOM

Declarative description of the UI inline in JavaScript code

Use inside if statements and for loops, assign to variables, accept as arguments, return from functions

Wrap JavaScript expressions in {} to be used within an element

Preprocessor translates syntax into plain JavaScript objects (Babel)

JSX vs. Create Element Example

```
class Hello extends React.Component {
  render () {
    return (
      <div className="container">
        <h1>Getting Started</h1>
      </div>
    );
  }
}

ReactDOM.render(<Hello />, mountNode);
```

```
class Hello extends React.Component {
  render () {
    return (
      React.createElement("div",
        { className: "container"},
         React.createElement("h1", null, "Getting Started")
      )
    );
  }
}
ReactDOM.render(React.createElement(Hello, null), mountNode);
```

Advanced JSX Example

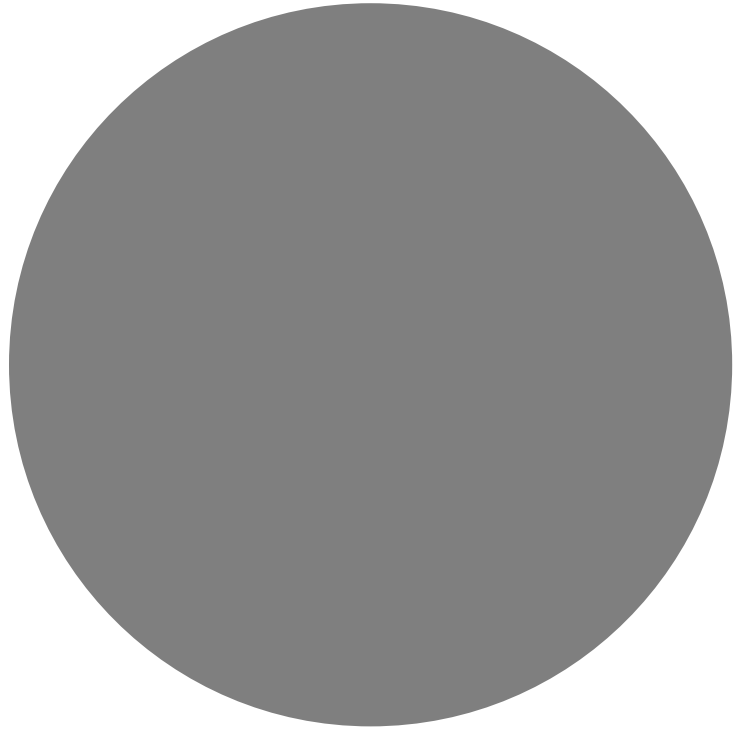
```
render() {
  return (
    <div className="container">
      <Paper zDepth={5}>
        <List>
          <Subheader className="projectSubheading">Red Projects</Subheader>
          {this.props.projects
            .filter( row => row.status === "R")
            .map( (row, index) => (
            <ProjectListItem key={index} project={row} />
          ))}
        </List>
        <Divider />
        <List>
          <Subheader className="projectSubheading">All Projects</Subheader>
          {this.props.projects
            .map( (row, index) => (
            <ProjectListItem key={index} project={row} />
          ))}
        </List>
      </Paper>
    </div>
  );
}
```

Create React App

Create React App

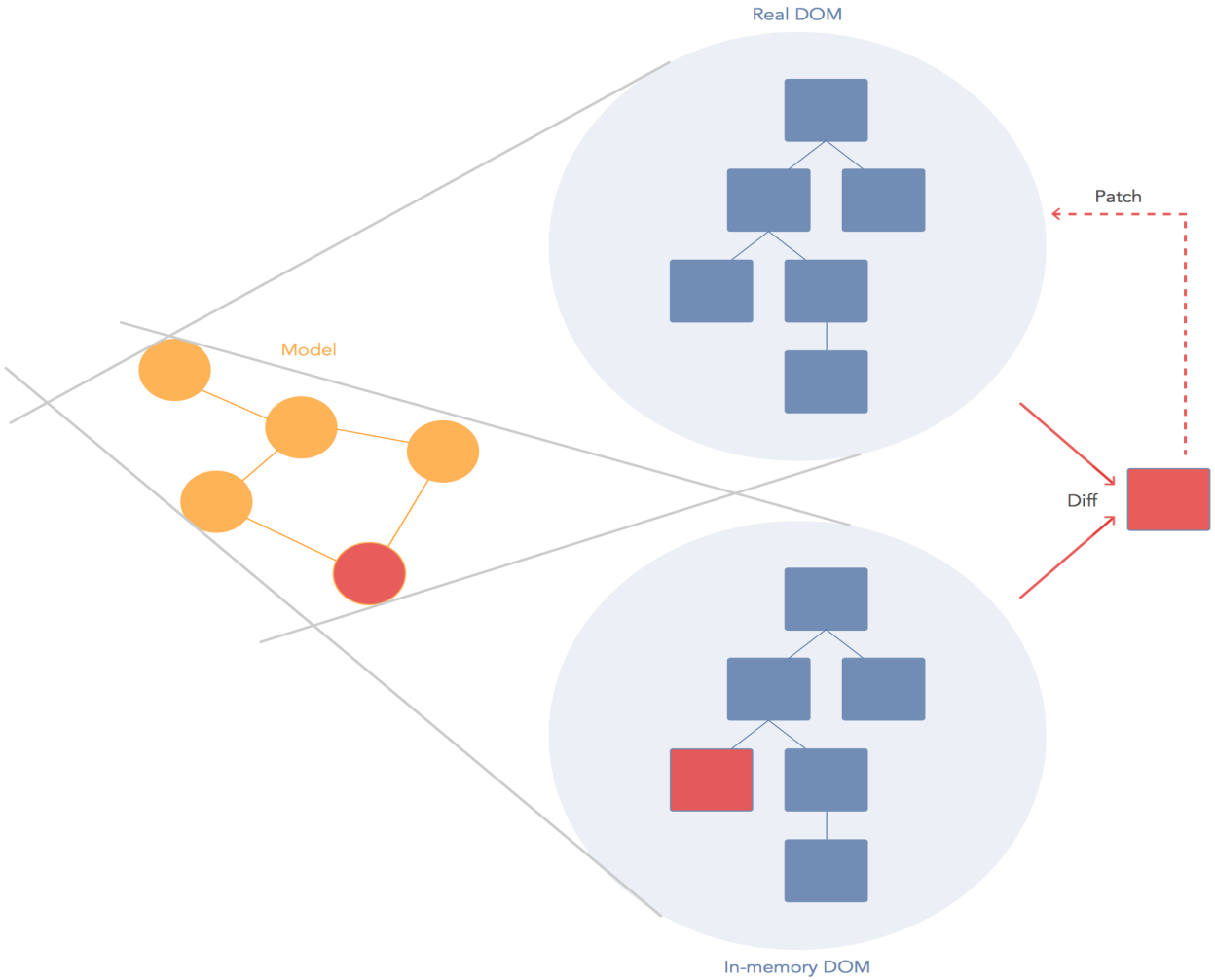
- Create React apps with no build configuration
- Every project built with it has a familiar base, reducing the ramp-up time as you and your teammates switch projects
- Quick start (in Terminal/Console):
  - npm install –g create-react app
  - create-react-app my-app
  - cd my-app
  - npm start or yarn start
- Uses ES6, Webpack, Babel, and ESLint
- Can be upgraded in-place to get the latest config and tooling
- You can “eject” to a custom setup at any time. No going back.

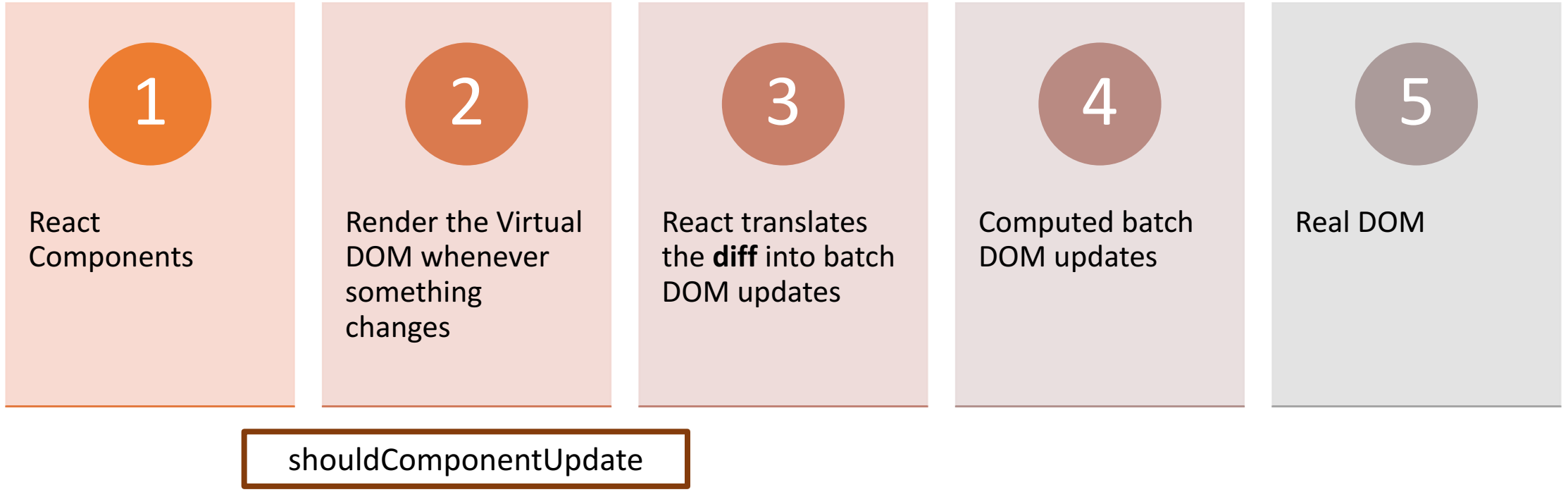
React Demo #1



Virtual DOM

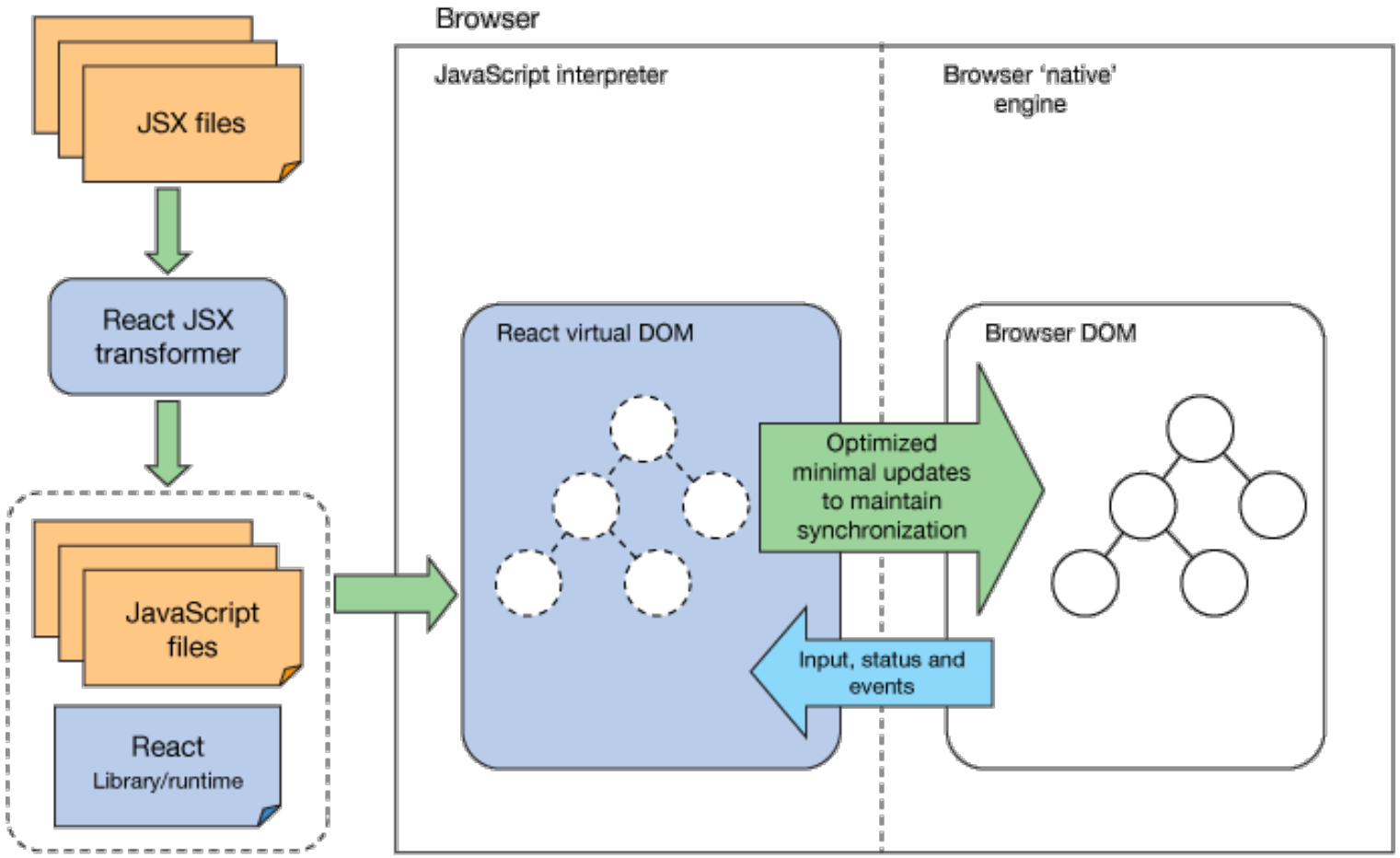
Virtual DOM




1
React Components
2
Render the Virtual DOM whenever something changes
3
React translates the **diff** into batch DOM updates
4
Computed batch DOM updates
5
Real DOM
shouldComponentUpdate

Virtual DOM

Virtual DOM


JSX files
React JSX transformer
JavaScript files
React
Library/runtime
Browser
JavaScript interpreter
Browser 'native' engine
React virtual DOM
Browser DOM
Optimized minimal updates to maintain synchronization
Input, status and events

Immutability

- An **immutable** object is an object whose state cannot be modified after it is created
- General rule: Treat state as if it were immutable
- You are circumventing React's state management > calling setState may replace your mutation
- Mutable array method: push
- Immutable array methods: Map, filter, concat

Example 1 (array):

```
let updatedTodos = this.state.todos.concat(newTodo);
this.setState({todos: updatedTodos});
```

Example 2 (mutate object):

```
let updateItem = { …this.state.todo, isComplete:true };
this.setState({todo: updateItem});
```

- Check out **React Immutability Helpers**

More on Components

Container vs. Presentational Components

- Note: React components do not need to emit DOM
- Container component does data fetching/business logic and renders its corresponding (shares the same name) sub-component
- Benefits:
  - Allow for separation of concerns
  - Allow for reusability of presentational components
  - Allow for better collaboration with designers and other developers

Container vs. Presentational Components

Container Component	Presentational Component
Concerned with how things work	Concerned with how things look
Provide data and behavior to presentational or other container components	Have no dependencies on the rest of the app (e.g. Redux, stores)
Interact with Redux or other state management libraries	Receive data and callbacks only through props
Generally stateful	Only state is UI state (not App state)
Generated using higher order components such as connect() (Redux)	Are generally written as functional components

Anti-Pattern

```
// CommentList.js
import React from "react";

class CommentList extends React.Component {
  constructor() {
    super();
    this.state = { comments: [] }
  }

  componentDidMount() {
    fetch("/my-comments.json")
      .then(res => res.json())
      .then(comments => this.setState({ comments }))
  }

  render() {
    return (
      <ul>
        {this.state.comments.map(({ body, author }) =>
          <li>{body}-{author}</li>
        )}
      </ul>
    );
  }
}
```

Best Practice

Container Component

```
// CommentListContainer.js
import React from "react";
import CommentList from "./CommentList";

class CommentListContainer extends React.Component {
  constructor() {
    super();
    this.state = { comments: [] }
  }

  componentDidMount() {
    fetch("/my-comments.json")
      .then(res => res.json())
      .then(comments => this.setState({ comments }))
  }

  render() {
    return <CommentList comments={this.state.comments} />;
  }
}
```

Presentational Component

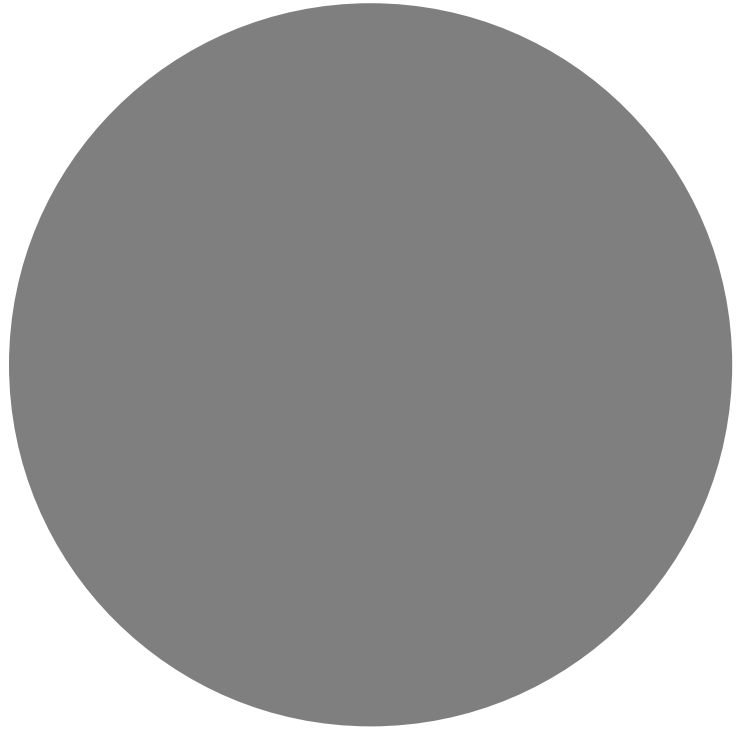
```
// CommentList.js
import React from "react";

const Commentlist = comments => (
  <ul>
    {comments.map(({ body, author }) =>
      <li>{body}-{author}</li>
    )}
  </ul>
)
```

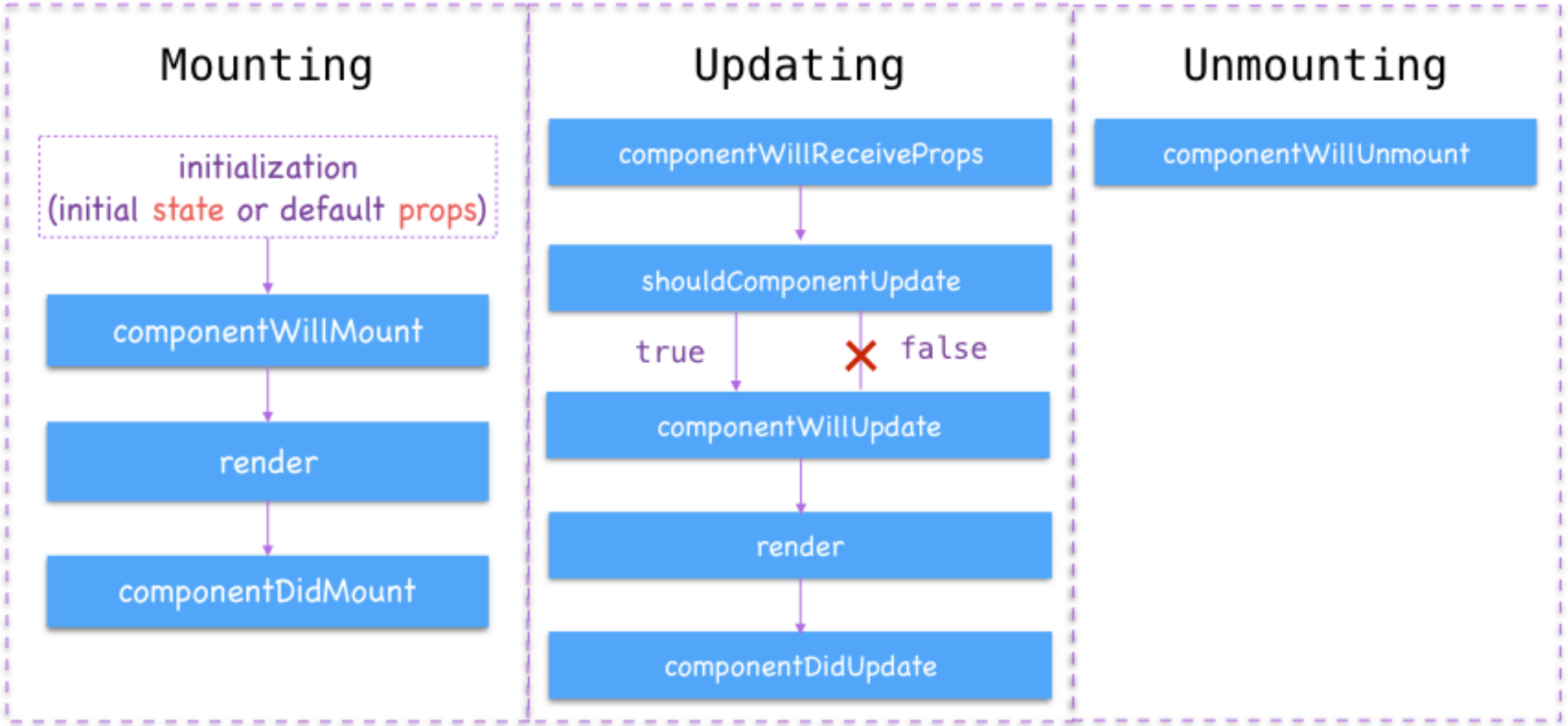
Stateless vs. Component vs. Pure Component

Stateless component	Created using a function No state No lifecycle methods Presentational components
Normal component	Allows state Lifecycle methods Generally Container components
Pure component	Similar to Normal components Implements shouldComponentUpdate automatically Shallow compare of new props and old props Allows state Lifecycle methods Be careful when using

React Demo #2



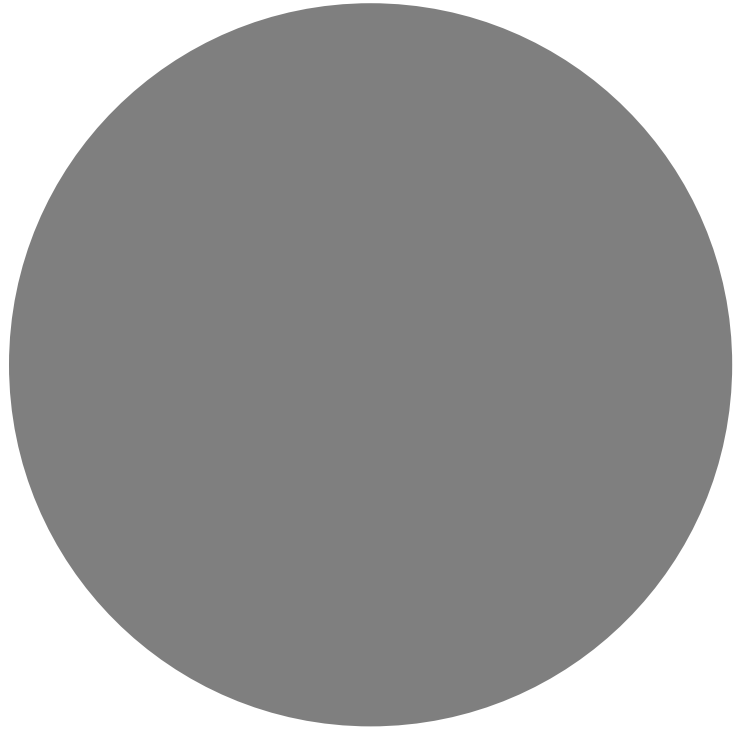
Lifecycle Methods


Mounting
initialization
(initial state or default props)
componentWillMount
render
componentDidMount
Updating
componentWillReceiveProps
shouldComponentUpdate
true
false
componentWillUpdate
render
componentDidUpdate
Unmounting
componentWillUnmount

Lifecycle Methods

Method	Common Uses	Call setState?
componentWillMount	App configuration	No
componentDidMount	Make AJAX calls	Yes*
componentWillReceiveProps	Handle prop changes from parent after construction	Yes
shouldComponentUpdate	Determines when the component will re-render	No
componentWillUpdate	Similar to componentWillReceiveProps but cannot call setState	No
componentDidUpdate	Updating the DOM in response to prop or state changes	Yes
componentWillUnmount	Cleaning up anything	No

React Demo #3



Questions

Resources

- React - https://facebook.github.io/react/
- Create React App - https://github.com/facebookincubator/create-react-app
- axios - https://github.com/mzabriskie/axios