

# React With Grails 3

GR8Conf US 2017

Zak Klein  
OCI Grails Team  
@ zacharyaklein

---

Visit us at [ocitraining.com](http://ocitraining.com)

---



OCI | TRAINING

# Using React with Grails 3

---

Zachary Klein  
Software Engineer, OCI



OCI | HOME TO GRAILS



Slides/Sample Code:  
<http://bit.ly/2fD91Qf>



# About Me

Grails & web developer for 5 years

Joined OCI Grails team in 2015

Using React since 2015

Author of the React profile/s for Grails 3



@zacharyaklein

On the Grails Team at @ObjectComputing.  
Christ-follower, husband (of Elizabeth), dad (of John & Timmy) programmer, guitarist. Needing to be more like Jesus.




Grails Learn Guides Documentation Download Plugins Community Support Socialize Q

# Grails Guides

Search...


Showing 43 of 43

**Grails Apprentice**



Using Command Objects  
To Handle Form Data


command-object binding validation



A GRAILS QUICKCAST  
FROM OCI


JSON Views  
Grails Quickcast #2

quickcast json-view




How to upload a file with  
Grails 3

command-object aws s3



Creating your first Grails  
Application


mysql gsp



A GRAILS QUICKCAST  
FROM OCI


Developing Grails 3  
Applications with IntelliJ  
IDEA  
Grails Quickcast #8

quickcast IntelliJ-Idea ide



How to change languages  
in a Grails app?

i18n



Create and test a Grails 3  
TagLib; integrate the Trix

<http://guides.grails.org>



start.grails.org/#/index

Grails Learn Documentation Support Download Plugins Community Ecosystem Socialize

## Welcome To The Grails Application Forge by OCI

Project Type:  
Application

Name Your Application:  
LUDWIG

Version:  
3.2.4

Profile:  
react

Pick Your Features

- ☐ asset-pipeline
- ☐ hibernate4
- ☒ hibernate5
- ☒ json-views
- ☐ less-asset-pipeline
- ☐ markup-views
- ☐ mongodb
- ☐ nvc4j
- ☐ rx-mongodb
- ☐ security

Generate Project

OR

Execute the following command

```
curl -O start.grails.org/evan.zip -d profile=react
```

<http://start.grails.org>



# Where we're going today

- What is React?
  - Brief overview of React and friends (npm, webpack, babel)
  - Only a primer - there's lots more!
- How do I use React with Grails?
  - Asset Pipeline plugin/s
  - React profile - monolith
  - React profile 2.x - separate client & server
  - Isomorphic React - server-side rendering w/Nashorn
- Resources & Q/A



# What is React?

A Javascript library for building user interfaces

Key Features:

- Virtual DOM
- JSX
- Components
- Single Page Apps
- Functional emphasis
- Beyond the browser

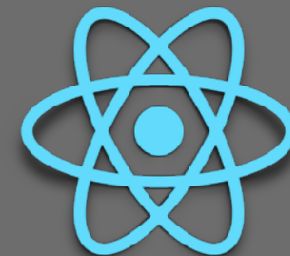


**"React is a JavaScript library,** and so it assumes you have a basic understanding of the JavaScript language."





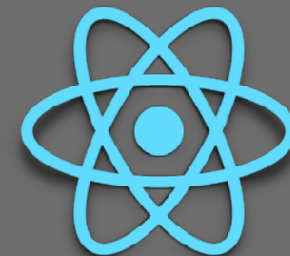
# What is React?



```
import React from 'react'  
import ReactDOM from 'react-dom'  
  
ReactDOM.render(  
  <h1>Hello, world!</h1>,  
  document.getElementById('root')  
)
```



# What is React?

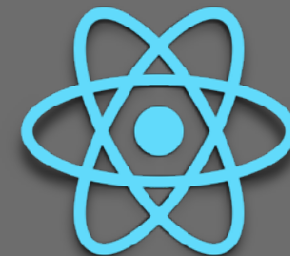


```
import React from 'react'
import ReactDOM from 'react-dom'

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



# What is React?



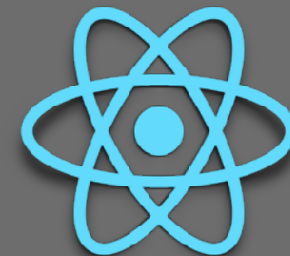
```
import React from 'react'
import ReactDOM from 'react-dom'

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

ReactDOM.render(
  <Greeting name='G3 Summit' />,
  document.getElementById('root')
);
```



# What is React?



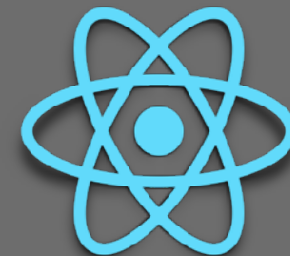
```
var React = require('react');
var ReactDOM = require('react-dom');

var Greeting = React.createClass({
  render: function() {
    return <h1>Hello, {this.props.name}</h1>;
  }
});

ReactDOM.render(
  <Greeting name='G3 Summit' />,
  document.getElementById('root')
);
```



# What is React?



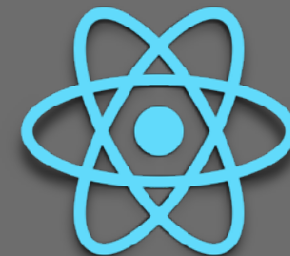
```
import React from 'react'
import ReactDOM from 'react-dom'

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

ReactDOM.render(
  <Greeting name='G3 Summit' />,
  document.getElementById('root')
);
```



# What is React?



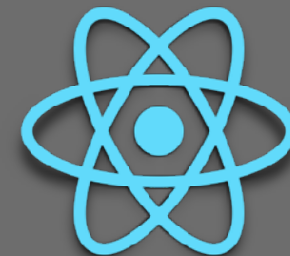
```
var React = require('react');
var ReactDOM = require('react-dom');

var Greeting = React.createClass({
  render: function() {
    return React.createElement(
      'h1', null, `Hello, ${this.props.name}!`
    );
  }
});

ReactDOM.render(
  React.createElement(Greeting, {name: 'G3 Summit'}, null),
  document.getElementById('root')
);
```



# What is React?

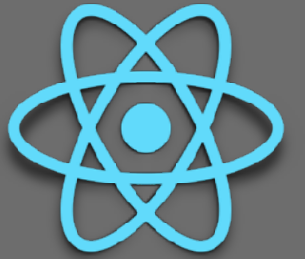


```
var React = require('react');
var ReactDOM = require('react-dom');

var Greeting = React.createClass({
  render: function() {
    return React.createElement(
      'h1', null, 'Hello, ${this.props.name}!'`
    ); name props body
  }
});

ReactDOM.render(
  React.createElement(Greeting, {name: 'G3 Summit'}, null),
  document.getElementById('root') name props body
);
```





# What is React?

```
import React from 'react'
import ReactDOM from 'react-dom'

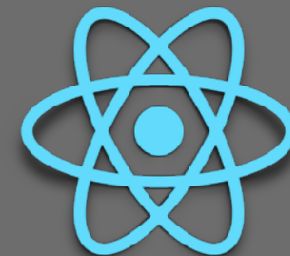
class Greeting extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}!</h1>;
  }
}

ReactDOM.render(
  <Greeting name='G3 Summit' />,
  document.getElementById('root')
);
```





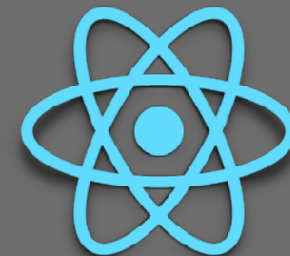
# What is React?



```
class Greeting extends React.Component {  
  render() {  
    return(  
      React.createElement('div', {},  
        React.createElement('h1', {}, "Greetings, ${this.props.name}"),  
        React.createElement('ul', {},  
          React.createElement('li', {},  
            React.createElement('a', {href: 'edit'},  
              'Edit this greeting')  
          ),  
          React.createElement('li', {},  
            React.createElement('a', {href: 'reset'},  
              'Reset this greeting')  
          )  
        )  
    );  
  }  
}  
  
ReactDOM.render(  
  React.createElement(Greeting, {name: 'G3 Summit'}, null),  
  document.getElementById('root')  
);
```

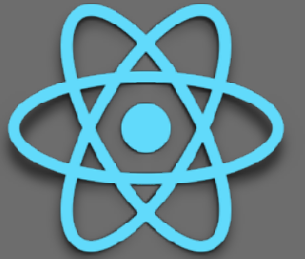


# What is React?



```
class Greeting extends React.Component {  
  render() {  
    return (<div>  
      <h1>Hello, {this.props.name}!</h1>  
      <ul>  
        <li><a href='edit'>Edit this greeting</a></li>  
        <li><a href='reset'>Reset this greeting</a></li>  
      </ul>  
    </div>);  
  }  
}  
  
ReactDOM.render(  
  <Greeting name='G3 Summit' />,  
  document.getElementById('root')  
>);
```





# What is React?

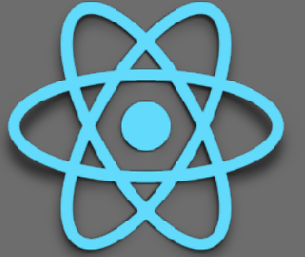
**"React is only the view layer.**

**We're only in one concern.** React only knows how to render markup. It doesn't know where your data came from, how it's stored, or how to manipulate it. What concerns are being violated?"

*Andrew Ray, via <http://blog.andrewray.me/youre-missing-the-point-of-jsx/>*



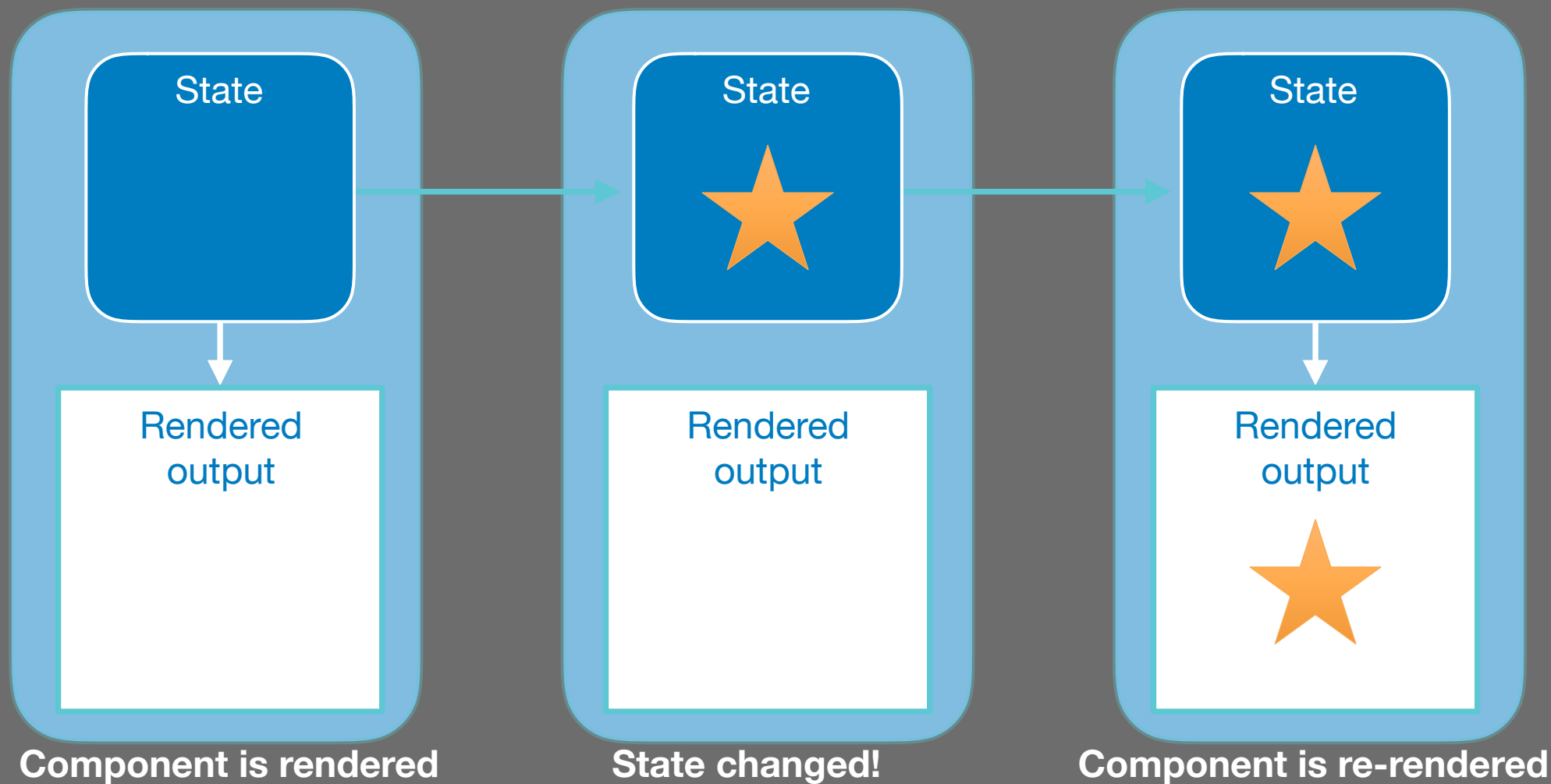
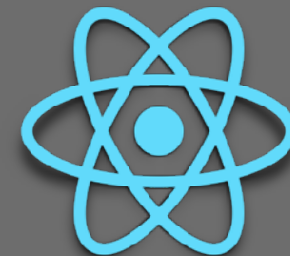
# Component State



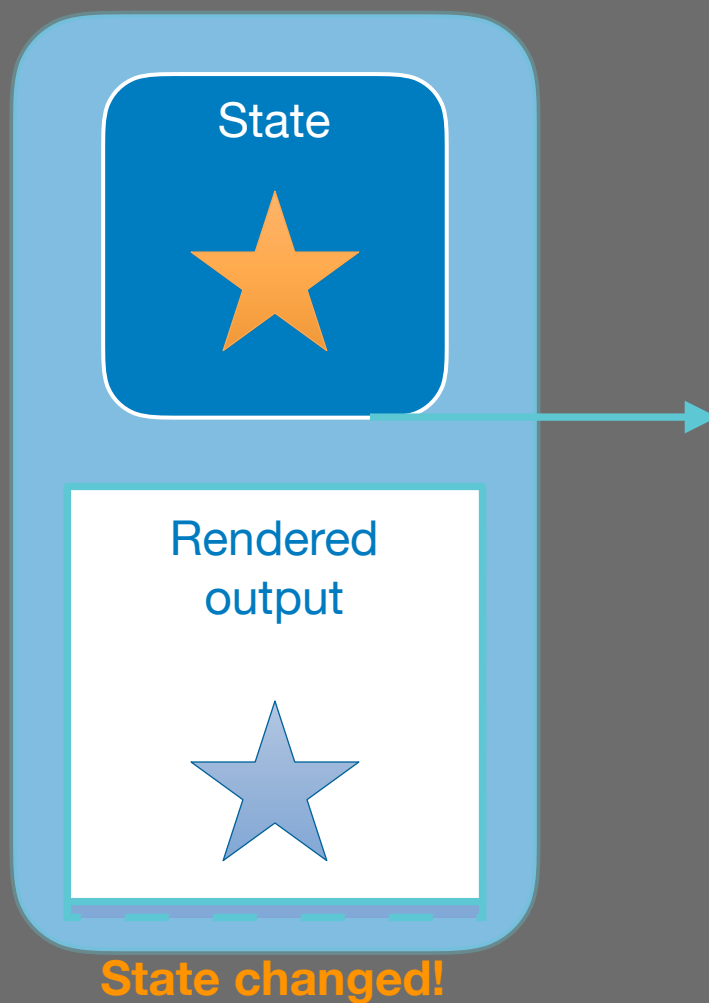
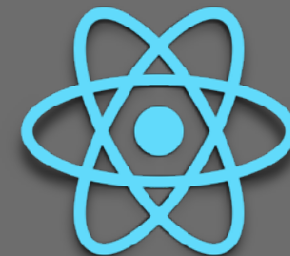
```
class Greeting extends React.Component {  
  constructor() {  
    super();  
  
    this.state = {  
      greetings: ['Hello', 'Salutations', 'Ho there']  
    }  
  }  
  
  render() {  
    const greetings = this.state.greetings;  
    const randomGreeting = greetings[Math.floor(Math.random() * greetings.length)];  
  
    return(  
      <h1>{randomGreeting}, {this.props.name}</h1>  
    );  
  }  
}  
  
ReactDOM.render(  
  React.createElement(<Greeting name='G3 Summit' />, document.getElementById('root'))  
);
```



# Component State

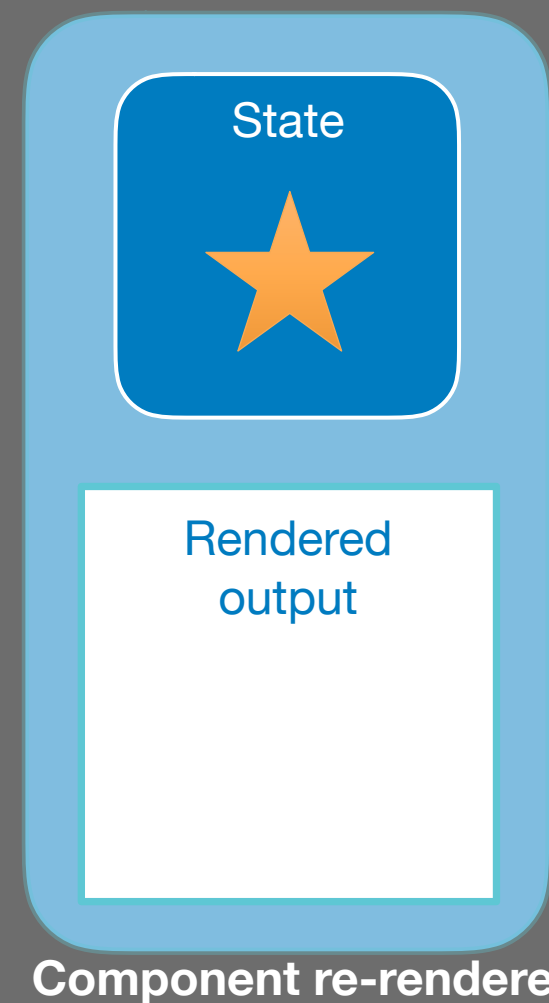


# Virtual DOM



Virtual DOM

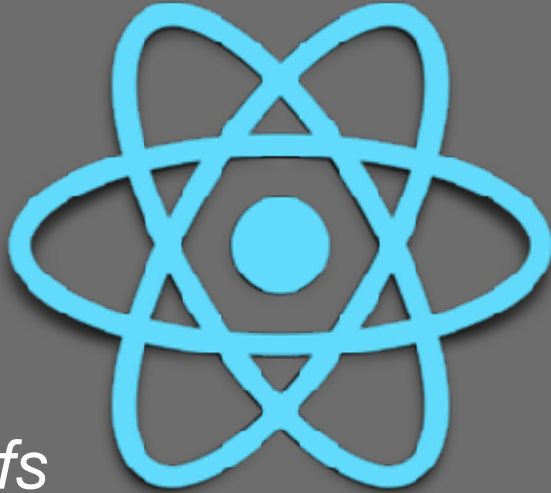
Rendered  
output



# What is React?

Other features:

- Lifecycle methods
- Event-handling
- State management
- PropTypes checking
- DOM access via *refs*
- SPA or partials



**"React is, in our opinion,** the premier way to build big, fast Web apps with JavaScript. It has scaled very well for us at Facebook and Instagram."



# What is React?

React is a small, focused library by design, but there's plenty of options for augmentation.



**Redux**



React Bootstrap



**IMMUTABLE**

**fetch**

**axios**

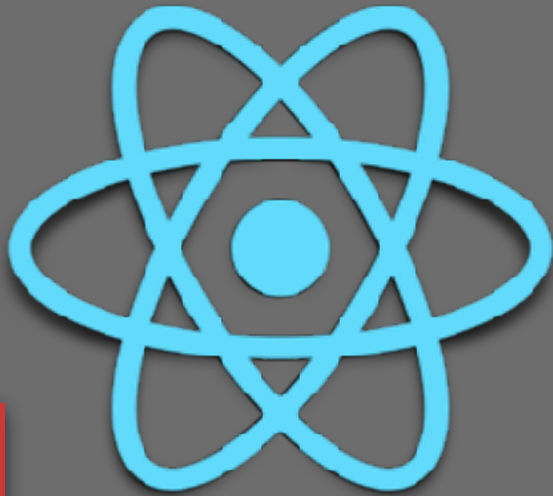
<https://www.toptal.com/react/navigating-the-react-ecosystem>



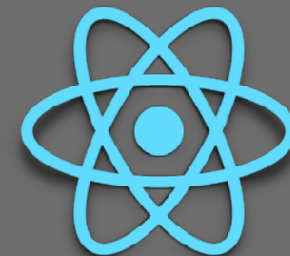


# What is React?

React can be used standalone, but is more frequently used with **node** & friends.



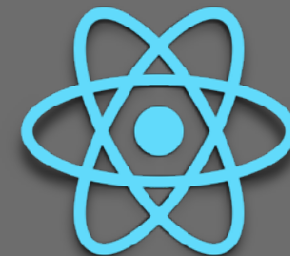
# What is React?



“npm is the default package manager for... Node.js.” (*Wikipedia*)



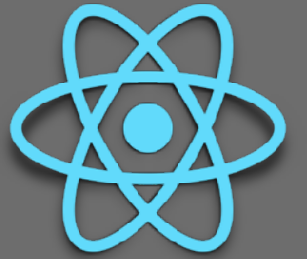
# What is React?



Yarn: “Fast, reliable, and secure dependency management.” *(Official Website)*



# What is React?



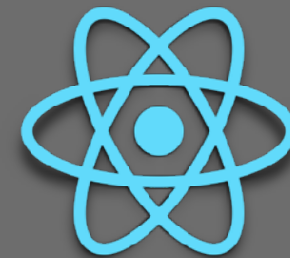
The 6th edition of ECMAScript, officially titled ECMAScript 2015.

A yellow square containing the text "ES6" in a bold, dark grey sans-serif font.

**ES6**



# What is React?

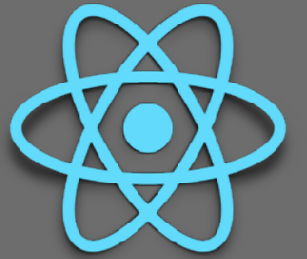


Babel - a Javascript “transpiler”

*BABEL*



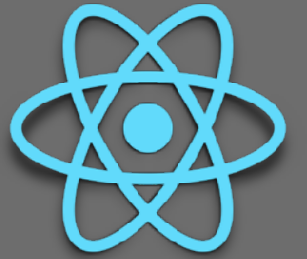
# What is React?



Webpack is an open-source Javascript module bundler. (*official website*)



# Why use React with Grails?



# Why use React with Grails?

- GORM
- Convention/Config
- Spring Boot
- Profiles
- URL mappings
- JSON views
- WebSockets
- Gradle

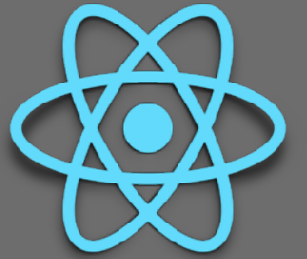


+





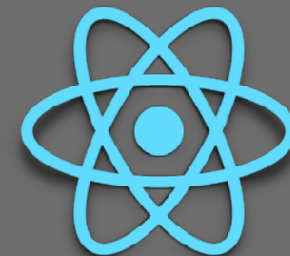
# How do I use React with Grails?



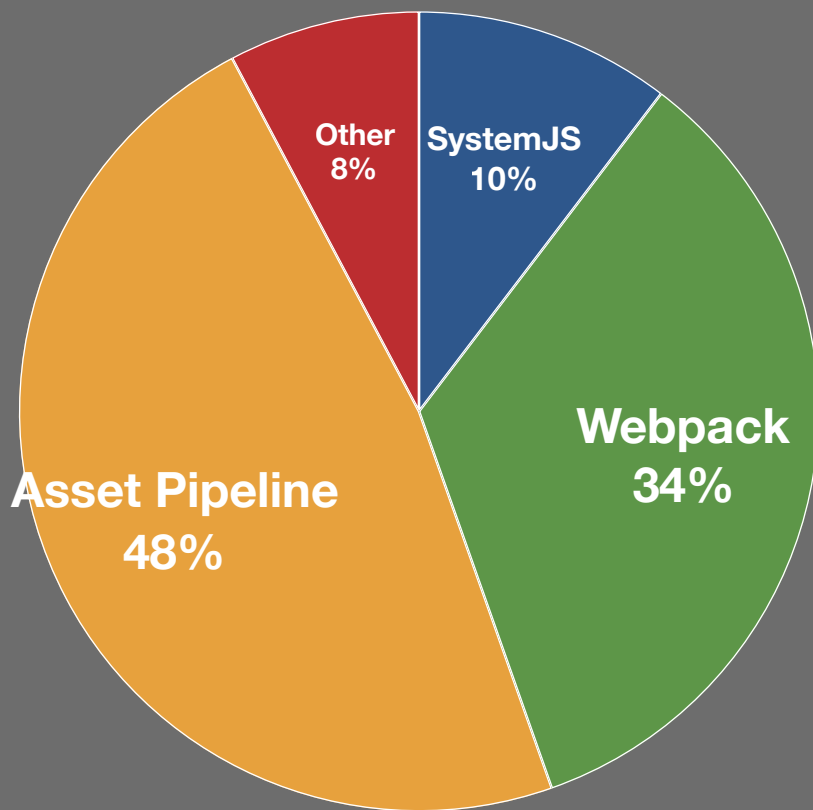
- Most React community resources & documentation assume a node/Javascript-based environment.
- Typical project structure is not immediately transferable to a Grails project.
- Use of **npm** is ubiquitous (but not required) and brings both advantages & challenges.
- Potential tension between front-end and back-end developer workflows
- Wild west - the trail is still being blazed!



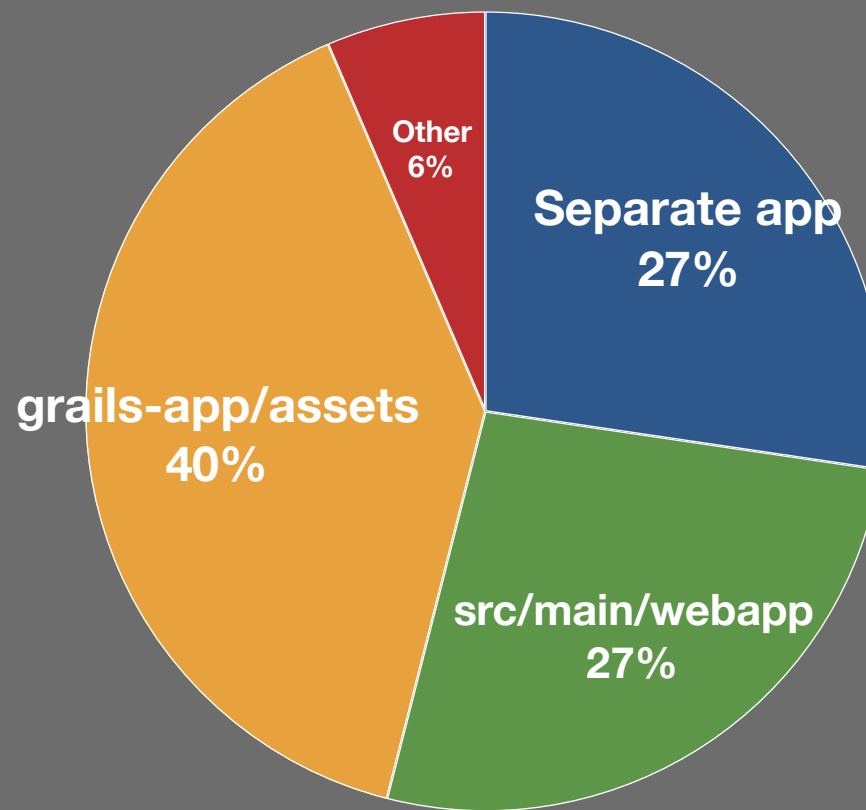
# How do I use React with Grails?



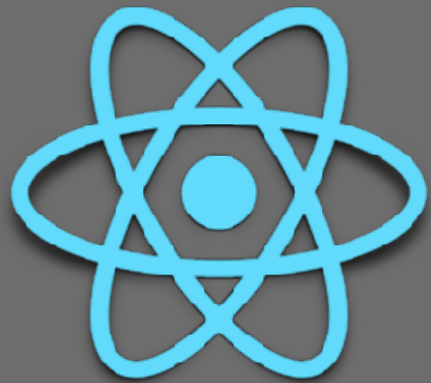
What build system  
would you like to see?



Where should the client  
app live?



# React & the Asset Pipeline



<ASSET PIPELINE/>



# React & the Asset Pipeline



- Front-end asset management/processing in Grails is typically handled via the **Asset Pipeline**.
- Recently-released plugins now support React with AP.
- React/JSX files in *grails-app/assets/javascripts*.
- Take advantage of other front-end Grails/Gradle plugins, such as the excellent **client-dependencies** plugin (<https://github.com/craigburke/client-dependencies-gradle>)
- Performant, extensible, familiar to Grails developers.



# React & the Asset Pipeline



- **Babel Asset-Pipeline Plugin**
  - Makes use of the Babel transpiler to transform ES6 code to ES5
  - Supports Webpack bundler, hot-reloading
  - Promising, but tricky to configure

```
compile(':babel-asset-pipeline:2.1.0')
```



# React & the Asset Pipeline



- **JSX Asset-Pipeline Plugin**
  - Natively parses React/JSX code w/o Babel
  - Follows Asset Pipeline conventions
  - Works well with **client-dependencies** plugin
  - Documentation is lacking
  - Excellent option for teams experienced/invested in AP

```
assets "com.bertramlabs.plugins:jsx-asset-pipeline:2.14.3"
```



# React & the Asset Pipeline



# DEMO



# React & the Asset Pipeline



- “Asset pipeline” seems to be out of favor among front-end developers (perhaps unfairly).
- May be difficult to follow along with **community documentation/resources**.
- AP support for new asset processing tools tends to lag behind Node-based tooling.
- Not compatible with popular front-end toolchains.
- **Grails-centric** - may be confusing/unfamiliar to a dedicated front-end team.



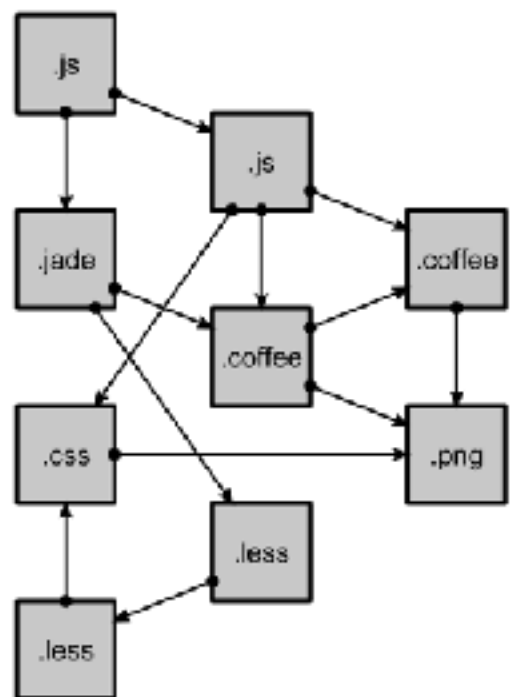


# Webpack

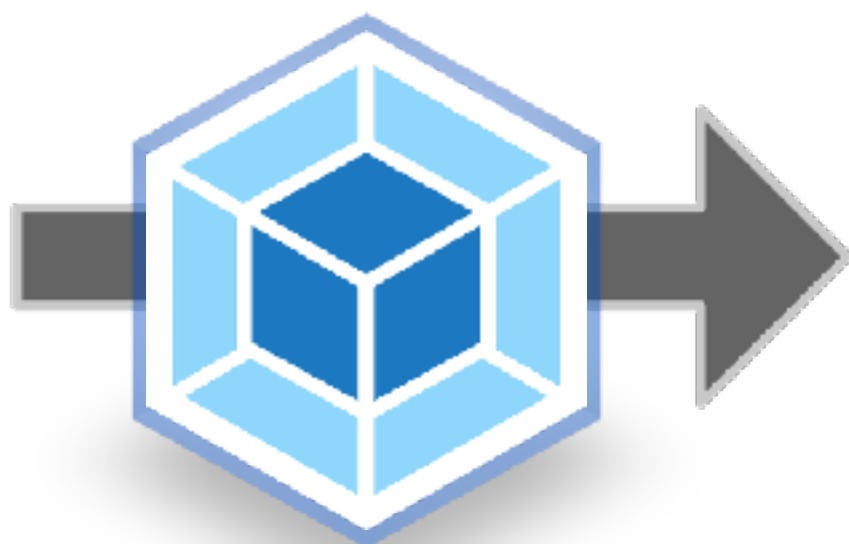


- Webpack is a Module bundler.
- Typically installed via **npm**, run via scripts in *package.json*
- Designed for Single Page Apps
- Supports a huge variety of asset types via configurable “loaders” (processors)
- Supports code-splitting/chunking
- Supports hashed/version assets for cache control
- Outputs a single bundle (by default), containing all required Javascript/CSS to render the target (i.e, your React app)

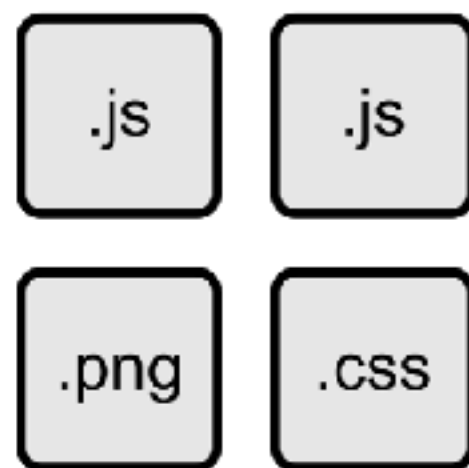




modules  
with dependencies



**webpack**  
MODULE BUNDLER

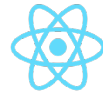


static  
assets

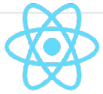


**ES6**

```
import 'text-label'
```

**hello.js****ES6**

```
import 'lodash'
```

**text-label.js****ES6**

```
import 'hello'
```

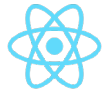
**app.js****npm****ES6**

```
//from node_modules
```

**lodash.js**

```
_react2.default.createElement(  
  _reactBootstrap.Modal.Footer,  
  null,  
  _react2.default.createElement(  
    'span',  
    { className: 'copyright' },  
    'Built with React 15.3.2, webpack  
1.13.1, Grails 3'  
  ),  
  _react2.default.createElement(  
    _reactBootstrap.Button,  
    { onClick: this.toggleModal },  
    'Close'  
  )  
)
```

**ES5  
classic****bundle.js**



ES6

```
function hello() {  
  return <div>Hello world!</div>;  
}
```

hello.js



*BABEL* - loader

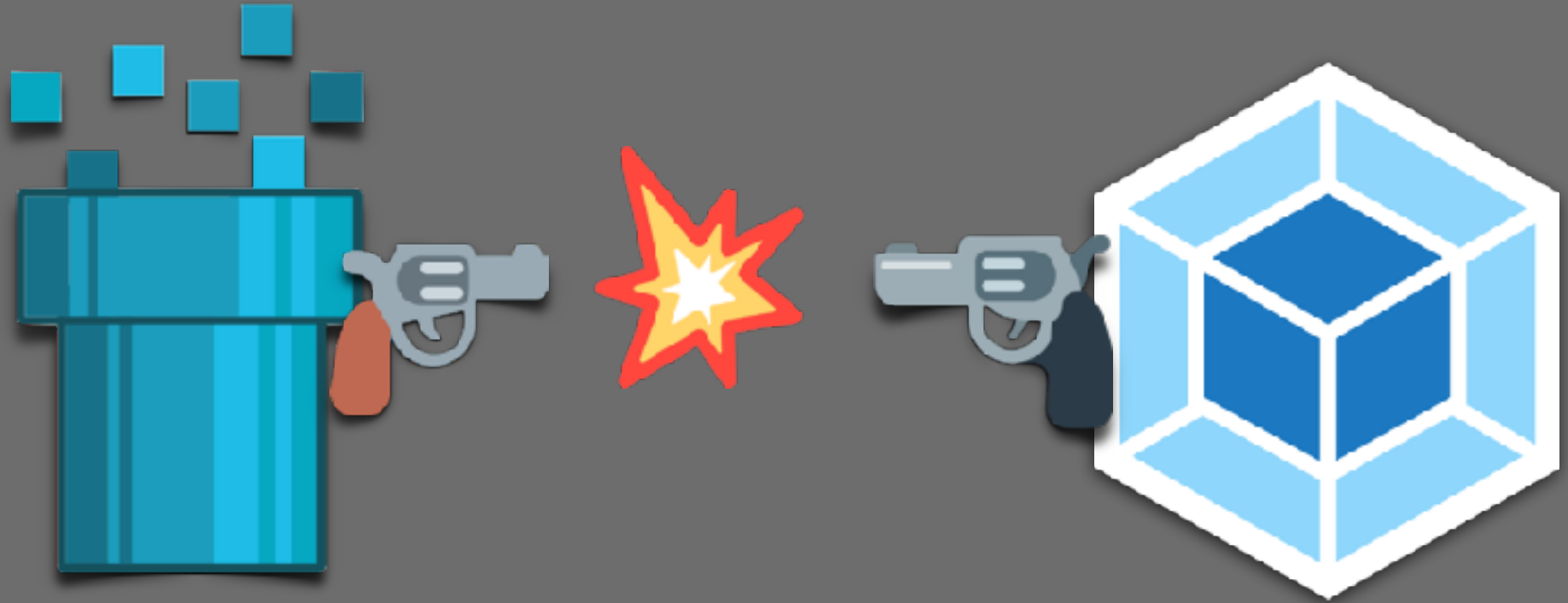
```
"use strict";  
  
function hello() {  
  return React.createElement(  
    "div",  
    null,  
    "Hello world!"  
  );  
}
```

ES5  
classic

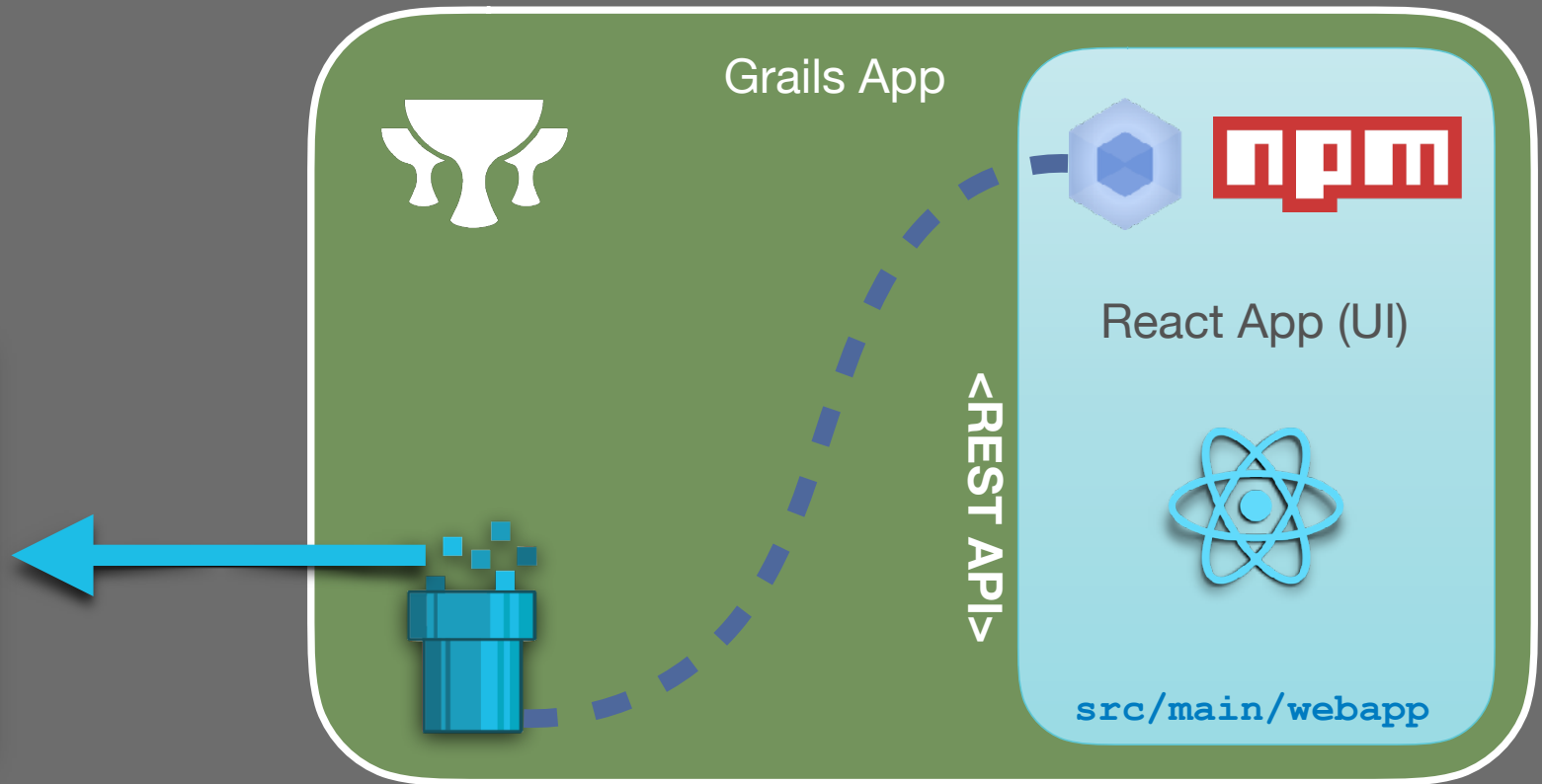
bundle.js



# Asset Pipeline vs Webpack



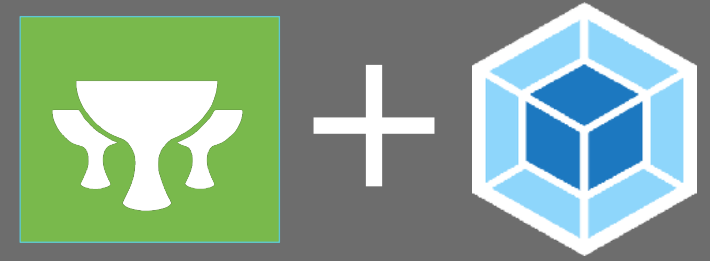
# Webpack & Grails



# Webpack & Grails



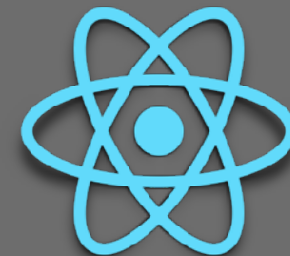
# Webpack & Grails



- Compromise:
  - Configure Webpack to output bundle/s into *grails-app/assets/javascripts*
  - Keep React source code in a separate directory tree (e.g, *src/main/webapp*)
  - Rely on Webpack for processing our React/ES6 code
  - Continue to use AP for non-React assets (jQuery, bootstrap, scaffolded pages, etc), as well as to serve the webpack bundle to the browser.
  - Use Gradle to automate running webpack via **npm** scripts
- All this can be tricky to configure - if only there was an easier way...







# React Profile for Grails 3

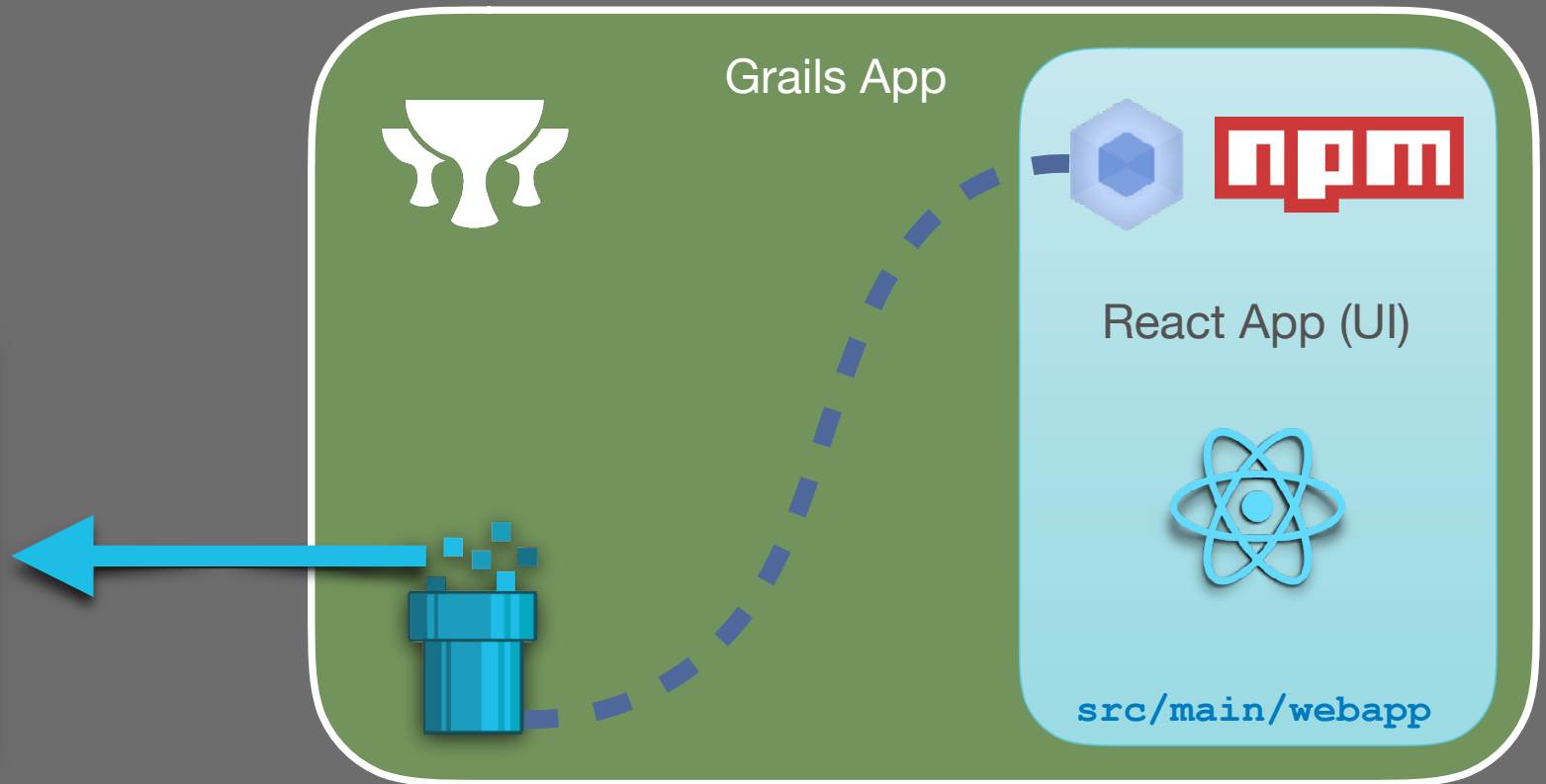
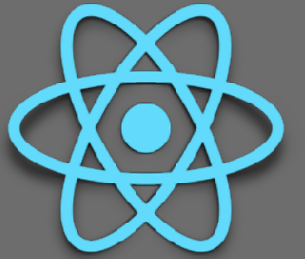
React 1.x Profile generates a Grails project with the following:

- React, ReactDOM etc., installed via **npm**
- Webpack configured to process React code and output to *grails-app/assets/javascripts*
- **gradle-node** plugin installed, custom tasks to run webpack on app startup/packaging
- Sample React code & unit tests

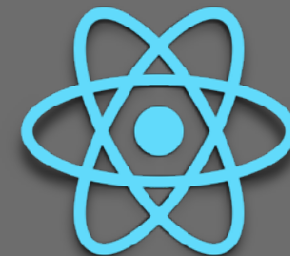
```
grails create-app myReactApp --profile org.grails.profiles:react:1.0.2
```



# React Profile for Grails 3



# React Profile for Grails 3



webpack bundle

npm project file

webpack configuration file

```
$ ls -l

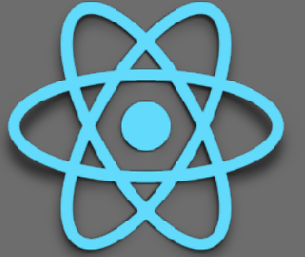
grails-app/
- assets/
- - javascripts/
- - - bundle.js
node_modules/
package.json
src/
- main/
- - webapp/
- - - app/
- - - - about.js
- - test/
- - - js/
- - - - about.spec.js
webpack.config.js
```

React source code

Unit test

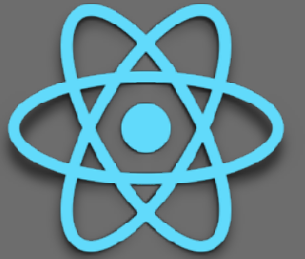


# React Profile for Grails 3



# DEMO





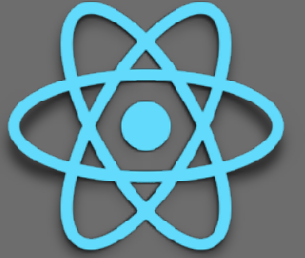
# React Profile for Grails 3

- The React 1.x profile simplifies the setup process for using React & Webpack with Grails
- Designed for monolithic applications
- Gradle-node plugin ties the two “worlds” together
- Single development/test/deployment path
- Gradle wrapper allows front-end devs to run the Grails app w/o installing Grails
- Gradle-node tasks allow Grails devs to run webpack w/o installing npm

```
./gradlew bootRun
```

```
./gradlew webpack
```



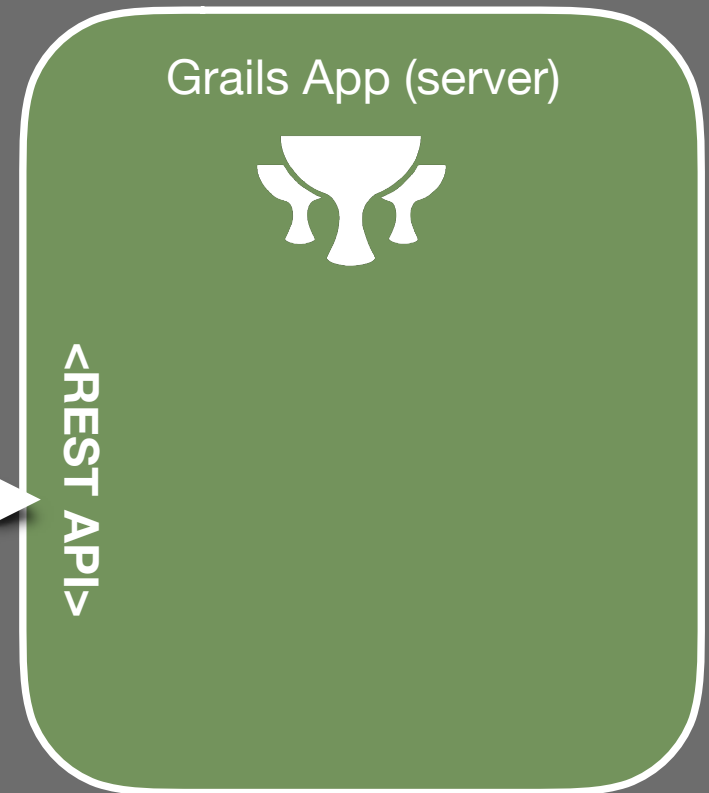
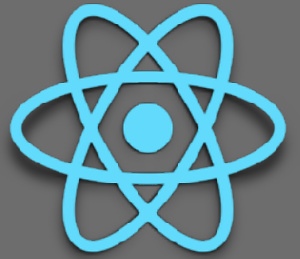


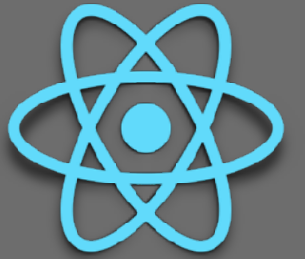
# Separate Client & Server

- Microservice-friendly
- Deploy/update front-end and back-end separately
- Take advantage of Grails' RESTful features
  - Domain resources
  - JSON Views
- Gradle multi-project build for client & server apps
- Requires fully standalone React app, including:
  - webpack, babel & dev-server
  - CORS configuration



# Separate Client & Server





# create-react-app

“**Create React App** is a new officially supported way to create single-page React applications. It offers a modern build setup with no configuration.”

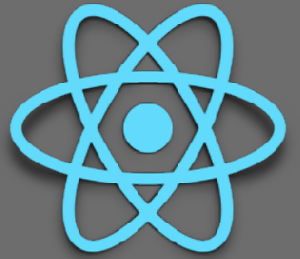
<https://facebook.github.io/react/blog/2016/07/22/create-apps-with-no-configuration.html>

- Generates fully standalone React app
- Provides working webpack & Babel config
- Provides scripts for starting app, building public bundle, running tests
- Simplified development & an easy “exit strategy”





# React Profile for Grails 3



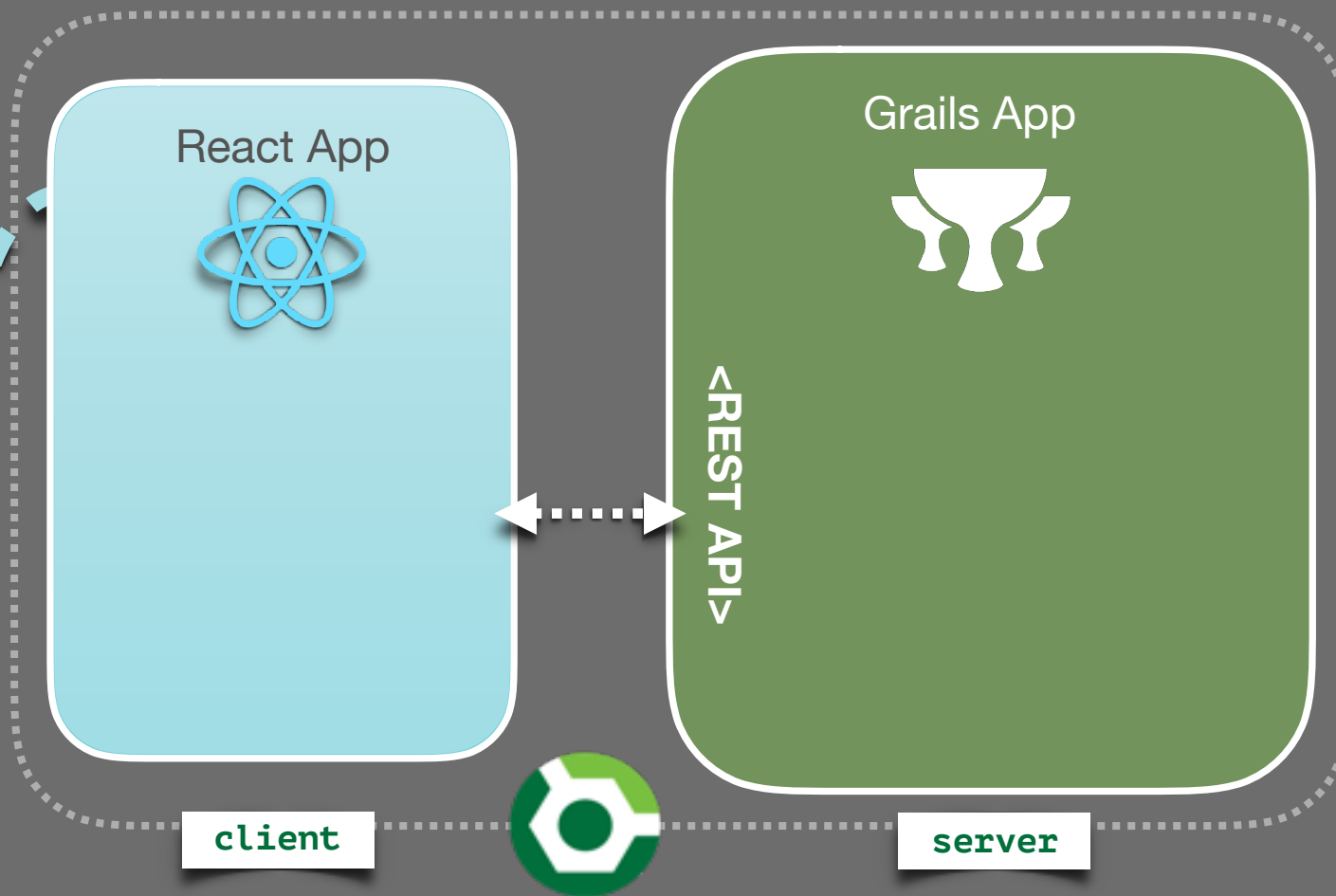
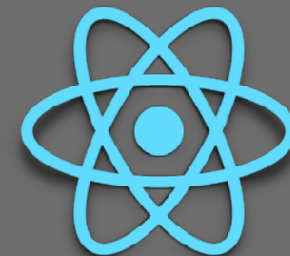
React 2.x Profile generates a multi-project Gradle build:

- React app (generated via **create-react-app**) as *client* project
- Grails 3 app (**rest-api** profile) as *server* project
- Gradle-node tasks defined within *client* project to run npm scripts (*start*, *build*, *test*, & *eject*)
- Grails index page built with *react-bootstrap*, with app data populated via REST call

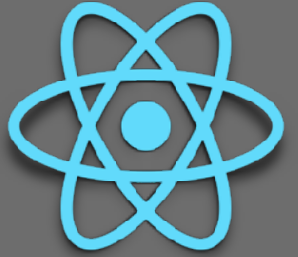
```
grails create-app myReactApp --profile react
```



# React Profile for Grails 3



# React Profile for Grails 3



React app

React source code

Grails 3 app

```
$ ls -l

client/
- build.gradle
- node_modules/
- package.json
- public/
- src/
- - App.js
- - App.test.js
server/
settings.gradle
```

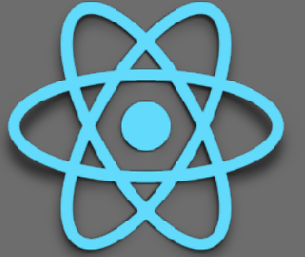
npm project file

Unit test

Gradle project file



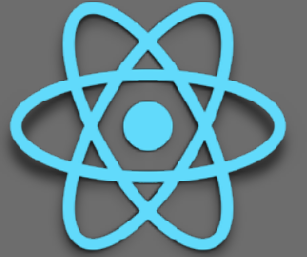
# React Profile for Grails 3



# DEMO



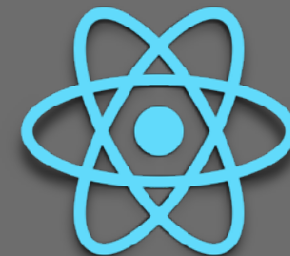
# Isomorphic React



- Isomorphic - same code on client & server (aka “universal”)
- React can be rendered server-side
- Can improve initial page load time
- Can improve SEO performance
- Java 8 introduced a new JavaScript engine, **Nashorn**



# Isomorphic React



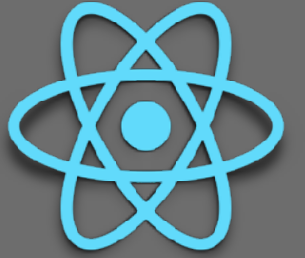
```
function myFunction(data) {  
  render("Here's some data we got from the server: " + data.arg);  
}
```

ES6

```
NashornScriptEngine nashornScriptEngine =  
    new ScriptEngineManager().getEngineByName("nashorn")  
  
nashornScriptEngine.eval(readResource("myscript.js"))  
  
nashornScriptEngine.invokeFunction('myFunction',  
    "${new JsonBuilder([arg:'some arg'])}")
```

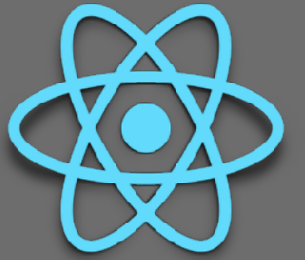


# Isomorphic React



# DEMO





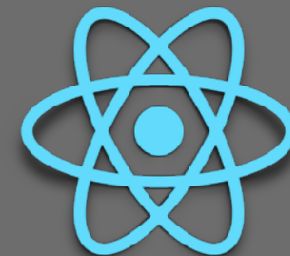
# Isomorphic React

- Tightly couples React source code with Grails app
- Adds additional complexity
- Requires polyfill to work with React
- Nashorn's CSS/LESS support is very poor
- Perhaps useful for React scaffolding?
- Performance?





# Resources



## React Ecosystem

- <https://www.toptal.com/react/navigating-the-react-ecosystem>

## JSX

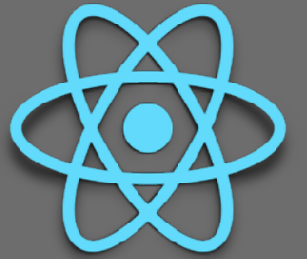
- <http://jamesknelson.com/learned-stop-worrying-love-jsx/>
- <http://blog.andrewray.me/youre-missing-the-point-of-jsx/>

## React Architecture

- <https://facebook.github.io/react/docs/thinking-in-react.html>
- <https://discuss.reactjs.org/t/best-practices-for-extending-subclassing-components/1820>



# Resources



## Grails Plugins & Profiles

- <https://grails.org/plugin/babel-asset-pipeline>
- <https://github.com/bertramdev/asset-pipeline/tree/master/jsx-asset-pipeline>
- <https://github.com/ZacharyKlein/grails-isomorphic>
- <https://github.com/grails-profiles/webpack>
- <https://github.com/grails-profiles/react>

## Using React & Grails

- <http://grailsblog.objectcomputing.com/posts/2016/05/28/using-react-with-grails.html>
- <http://grailsblog.objectcomputing.com/posts/2016/11/14/introducing-the-react-profile-for-grails.html>
- <http://guides.grails.org/using-the-react-profile/guide/index.html>



# Thank you!

twitter: @zacharyaklein  
mailto: [kleinz@ociweb.com](mailto:kleinz@ociweb.com)



OCI | HOME TO GRAILS