

REACT+ WORDPRESS

a match made in heaven



JOSH BROTON

@joshbroton || joshbroton.com

HilariousGifs.com





OWNER/FOUNDER:
BROTON DIGITAL CONSULTING



FOUNDER:
PRESTIGE CONFERENCE



PRESTIGECONF.COM

(it's this week - you should come)

PRESTIGE CONFERENCE

A PREMIUM BUSINESS AND
CAREER DEVELOPMENT CONFERENCE

MAY 24-25, 2016

REGISTER NOW



OWNER/FOUNDER:
BROTON DIGITAL CONSULTING



FOUNDER:
PRESTIGE CONFERENCE



JAVASCRIPT/FRONTEND:
**SLICEENGINE, VIDSCRIP,
OTHER STARTUPS**

If I had a dollar for everytime I got
distracted, I wish I had some ice
cream.

somee cards
user card





I don't have attention
deficit disorder...

I have this isn't
interesting enough to
pay attention to disorder.



som**ee**cards
user card

**BROTY
+ DECONGESTANTS
+ COFFEE
+ NERVES**



WHAT THIS TALK SHOULD BE



CONFERENCE SESSIONS SHOULD NEVER
AIM TO TEACH THE AUDIENCE SOMETHING,
BUT RATHER INSPIRE THEM TO WANT TO
LEARN.

- TED NEWARD

WHAT WE'RE COVERING

(I hear Mendel &
Travis both have
awesome
sessions)

01

WTF IS A SPA? WHY A SPA?

02

WHY / WHY NOT REACT?

03

REACT + WORDPRESS

01

WHAT IS A SPA?

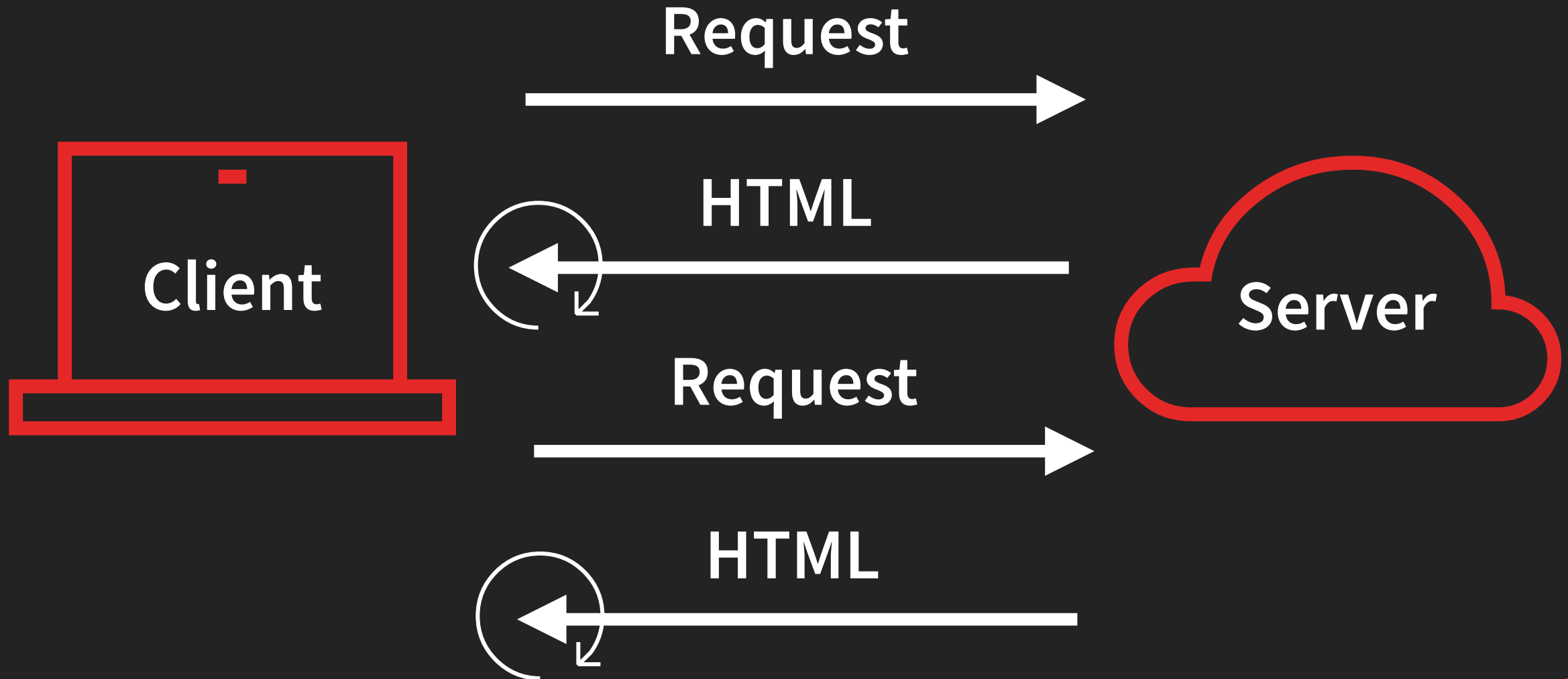


01

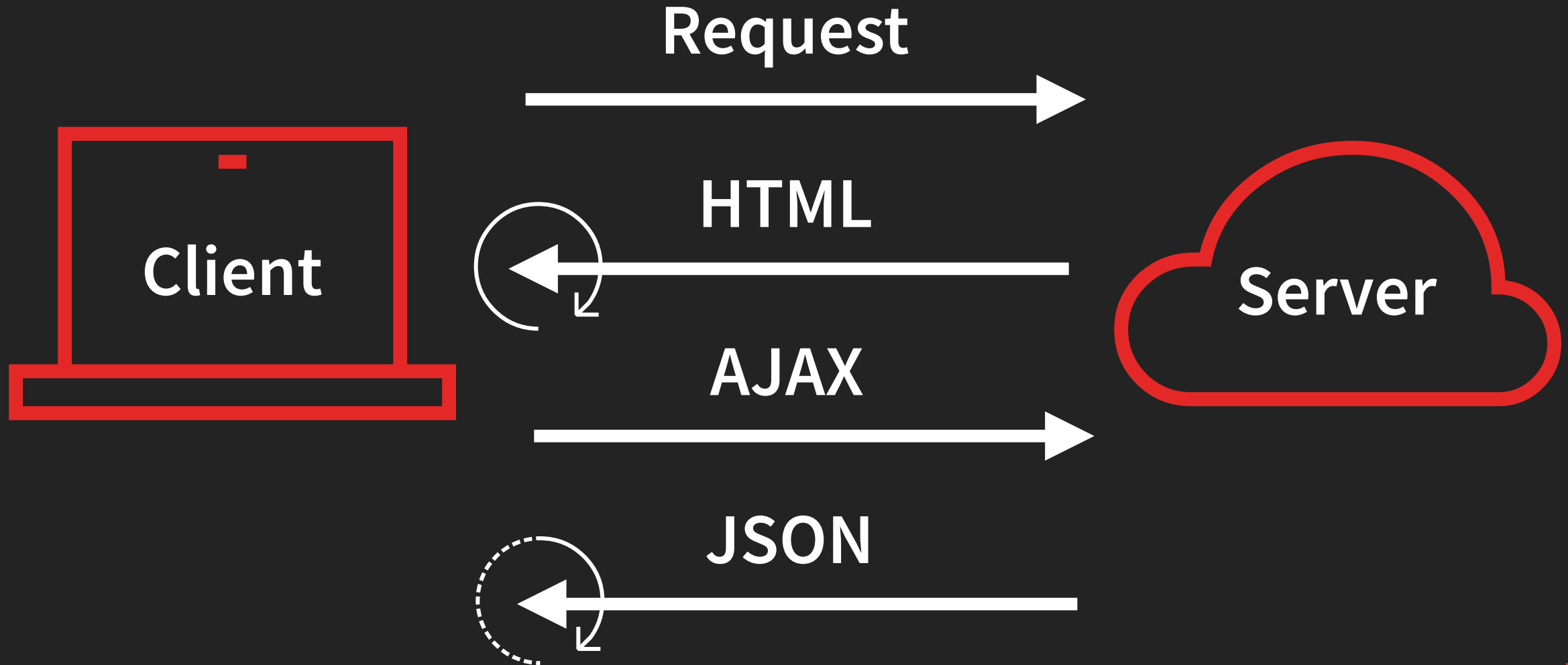
WHAT IS A SPA?

Single Page App

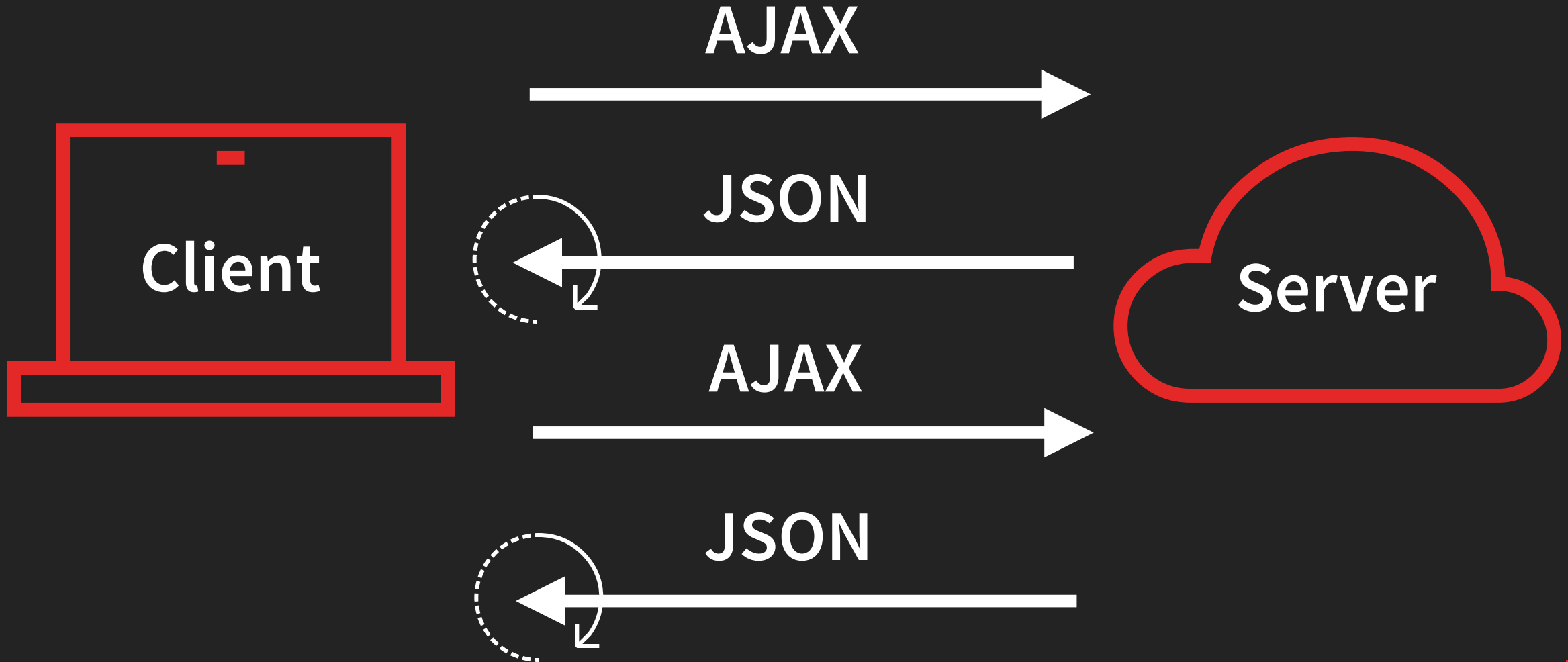
TYPICAL WEB SITE:



SINGLE PAGE APP:



SINGLE PAGE APP:



THE SPA ADVANTAGE



OFFLOADS PROCESSING TO
CLIENTS



FRONT-END CODE IS
EASIER TO TEST



FASTER ON SUBSEQUENT
PAGE LOADS



SCALES WITH LARGE
DATA SETS



02

WHY REACT?

#JOSHAWA

02 WHY REACT?



02

WHY REACT?

2014: THE YEAR OF VANILLA JAVASCRIPT



02

WHY REACT?

2016: THE YEAR OF THE FRAMEWORK

02

WHY REACT?



02 WHY REACT?



02

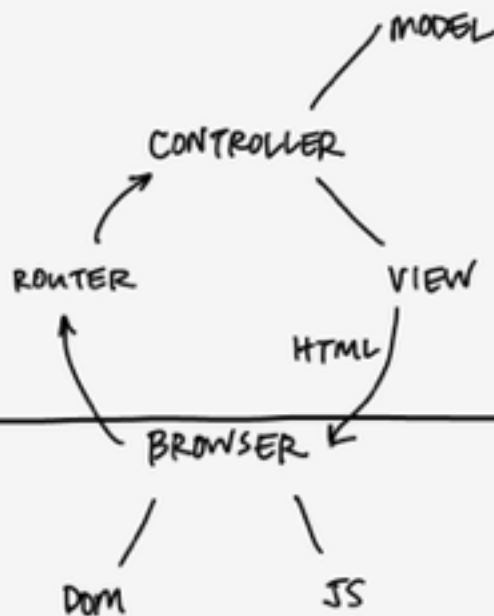
WHY REACT?



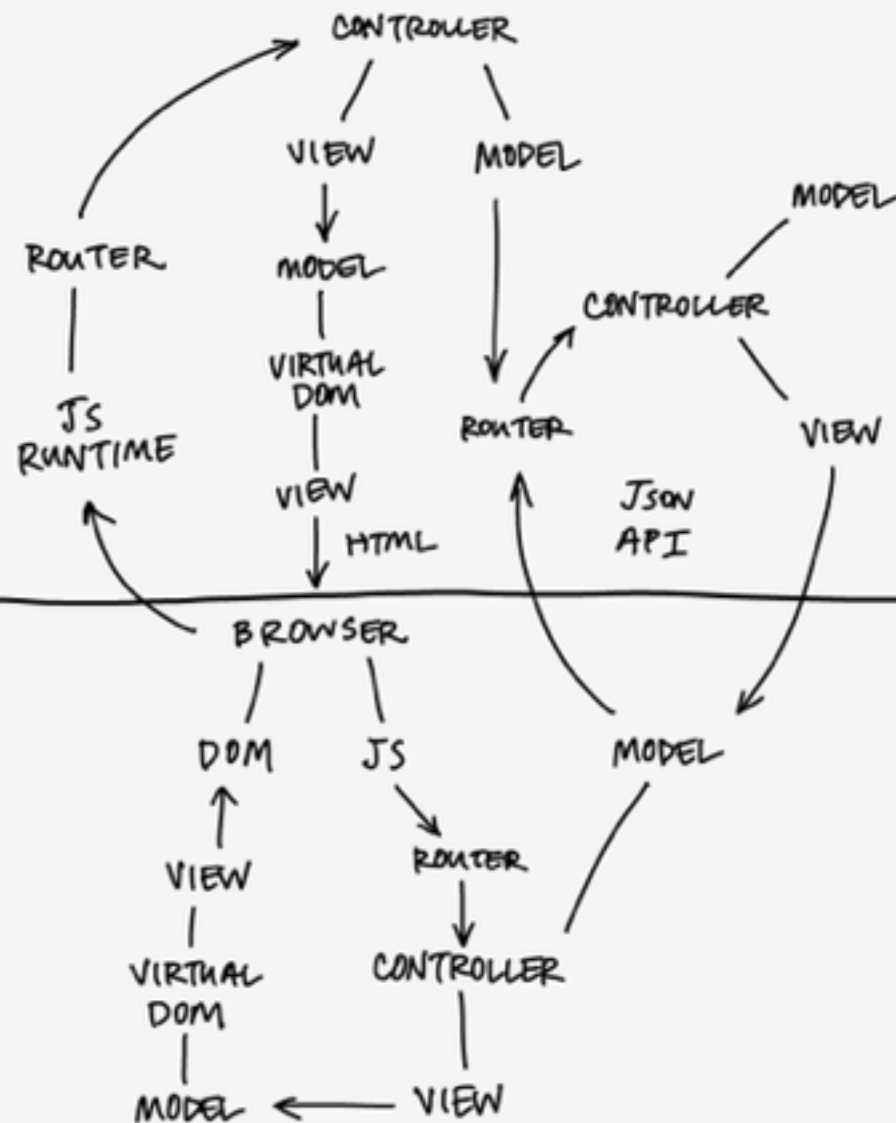
02 WHY REACT?



2004



2016





Ada Rose Edwards

@Lady_Ada_King



Follow

The nice thing about the web is the diagram on the left hand side still applies if you want it to.



Sam Stephenson @sstephenson

12 years of progress?

RETWEETS

20

LIKES

28



10:55 AM - 10 May 2016





02

WHY REACT?



JSX IS OUTSTANDING

02

WHY REACT?

```
(  
  <article className="some-blog-post">  
    <header className="post__header">  
      <h1>  
        {post.title.replace('"', ' ')}  
      </h1>  
      <aside className="byline">  
        {generatePostAuthorLink(post.author)}  
      </aside>  
    </header>  
    <div className="post__content">  
      {post.content}  
    </div>  
    ...  
  </article>  
)
```

02

WHY REACT?

```
(  
  <article className="some-blog-post">  
    <PostHeader title={post.title} author={post.author} />  
    <PostContent content={post.content} />  
    <PostFooter categories={post.categories} />  
  </article>  
)
```

02

WHY REACT?

```
(
  this.state.posts.map((post) => {
    return (
      <article className="some-blog-post">
        <PostHeader title={post.title} author={post.author} />
        <PostContent content={post.content} />
        <PostFooter categories={post.categories} />
      </article>
    )
  })
)
```

02

WHY REACT?

```
(
  {this.state.showPost == true ? (
    <article className="some-blog-post">
      <PostHeader title={post.title} author={post.author} />
      <PostContent content={post.content} />
      <PostFooter categories={post.categories} />
    </article>
  ) : null }
)
```


02

WHY REACT?



JSX IS OUTSTANDING



REACT SYNTAX IS SIMPLE

02

WHY REACT?

```
(  
  <article className="some-blog-post">  
    <PostHeader title={post.title} author={post.author} />  
    <PostContent content={post.content} />  
    <PostFooter categories={post.categories} />  
  </article>  
)
```

02

WHY REACT?

```
class PostHeader extends Component {  
  componentWillMount() { }
```

```
}
```

02

WHY REACT?

```
class PostHeader extends Component {  
  componentWillMount() { }  
  componentDidMount() { }  
  
}
```

02

WHY REACT?

```
class PostHeader extends Component {  
  componentWillMount() { }  
  componentDidMount() { }  
  componentWillReceiveProps(nextProps) { }  
  
}
```


02

WHY REACT?

```
class PostHeader extends Component {  
  componentWillMount() { }  
  componentDidMount() { }  
  componentWillReceiveProps(nextProps) { }  
  shouldComponentUpdate(nextProps, nextState) { //can return false }  
  
}
```

02

WHY REACT?

```
class PostHeader extends Component {  
  componentWillMount() { }  
  componentDidMount() { }  
  componentWillReceiveProps(nextProps) { }  
  shouldComponentUpdate(nextProps, nextState) { //can return false }  
  componentWillUpdate(nextProps, nextState) { }  
  
}
```

02

WHY REACT?

```
class PostHeader extends Component {  
  componentWillMount() { }  
  componentDidMount() { }  
  componentWillReceiveProps(nextProps) { }  
  shouldComponentUpdate(nextProps, nextState) { //can return false }  
  componentWillUpdate(nextProps, nextState) { }  
  componentDidUpdate(prevProps, prevState)  
  
}
```

02

WHY REACT?

```
class PostHeader extends Component {  
  componentWillMount() { }  
  componentDidMount() { }  
  componentWillReceiveProps(nextProps) { }  
  shouldComponentUpdate(nextProps, nextState) { //can return false }  
  componentWillUpdate(nextProps, nextState) { }  
  componentDidUpdate(prevProps, prevState)  
  componentWillUnmount() { }  
  
}
```

02

WHY REACT?

```
class PostHeader extends Component {  
  componentWillMount() { }  
  componentDidMount() { }  
  componentWillReceiveProps(nextProps) { }  
  shouldComponentUpdate(nextProps, nextState) { //can return false }  
  componentWillUpdate(nextProps, nextState) { }  
  componentDidUpdate(prevProps, prevState)  
  componentWillUnmount() { }  
  
  render() { }  
}
```


02

WHY REACT?

```
class PostHeader extends Component {  
  render() {  
    return (  
      <header className="post__header">  
        <a href={this.props.permalink} target="_blank">  
          <h1>  
            {this.props.title}  
          </h1>  
        </a>  
        <aside className="byline">  
          {this.props.author}  
        </aside>  
      </header>  
    )  
  }  
}
```

02

WHY REACT?

```
class PostHeader extends Component {
  render() {
    return (
      <header className="post__header">
        <a href={this.props.permalink} target="_blank">
          <h1>
            {this.props.title}
          </h1>
        </a>
        <aside className="byline"
          onClick={this.showAuthorModal.bind(this)}>
          {this.props.author}
        </aside>
      </header>
    )
  }
}
```

02

WHY REACT?

```
class PostHeader extends Component {
  render() {
    return (
      <header className="post__header">
        <a href={this.props.permalink} target="_blank">
          <h1>
            {this.props.title}
          </h1>
        </a>
        {this.props.author !== this.state.user ? (
          <aside className="byline"
            onClick={this.showAuthorModal.bind(this)}>
            {this.props.author}
          </aside>
        ) : null }
      </header>
    )
  }
}
```

02

WHY REACT?

```
class PostHeader extends Component {
  render() {
    return (
      <header className="post__header">
        <a onClick={this.props.openPostIniFrame.bind(this)}>
          <h1>
            {this.props.title}
          </h1>
        </a>
        {this.props.author !== this.state.user ? (
          <aside className="byline"
            onClick={this.showAuthorModal.bind(this)}>
            {this.props.author}
          </aside>
        ) : null }
      </header>
    )
  }
}
```

02

WHY REACT?

```
class PostHeader extends Component {
  render() {
    return (
      <header className="post__header">
        <h1 role="button"
          onClick={this.props.openPostInIFrame.bind(this)}>
          {this.props.title}
        </h1>
        {this.props.author !== this.state.user ? (
          <aside className="byline"
            onClick={this.showAuthorModal.bind(this)}>
            {this.props.author}
          </aside>
        ) : null }
      </header>
    )
  }
}
```

02

WHY REACT?

```
class PostHeader extends Component {
  render() {
    return (
      <header className="post__header">
        <h1 role="button"
          onClick={this.props.openPostInIFrame.bind(this)}
          className={classNames({'read-title': this.hasRead()})}>
          {this.props.title}
        </h1>
        {this.props.author !== this.state.user ? (
          <aside className="byline"
            onClick={this.showAuthorModal.bind(this)}>
            {this.props.author}
          </aside>
        ) : null }
      </header>
    )
  }
}
```


02

WHY REACT?

```
class PostHeader extends Component {
  render() {
    if(this.props.isFetchingPosts == true) {
      return <LoadingSpinner />
    } else {
      return (
        <header className="post__header">
          <h1 role="button"
            onClick={this.props.openPostInIFrame.bind(this)}
            className={classNames({'read-title': this.hasRead()})}>
            {this.props.title}
          </h1>
          {this.props.author !== this.state.user ? (
            <aside className="byline"
              onClick={this.showAuthorModal.bind(this)}>
              {this.props.author}
            </aside>
          )}
        </header>
      )
    }
  }
}
```

02

WHY REACT?

```
class Post extends Component {
  render() {
    if(this.props.isFetchingPosts == true) {
      return <LoadingSpinner />
    } else {
      return (
        <article className="some-blog-post">
          <PostHeader title={post.title} author={post.author} />
          <PostContent content={post.content} />
          <PostFooter categories={post.categories} />
        </article>
      )
    }
  }
}
```

02

WHY REACT?



JSX IS OUTSTANDING



REACT SYNTAX IS SIMPLE



REACT FAILS FAST AT COMPILE TIME

02

WHY REACT?

```
class PostHeader extends Component {
  render() {
    return (
      <header className="post__header">
        <a href={this.props.permalink} target="_blank">
          <h1>
            {this.props.title}
          </h1>
        </a>
        <aside className="byline">
          {this.props.author}
        </aside>
      </header>
    )
  }
}
```

02

WHY REACT?

```
ERROR in ./src/components/content/PostHeader/PostHeader.jsx
Module build failed: SyntaxError: /Users/joshbroton/dev/git/vidscrip/site/src/components/content/PostHeader/PostHeader.js
x: Unexpected token (13:46)
  11 |         return (
  12 |             <header className="post__header">
> 13 |                 <a href={this.props.permalink} target="_blank">
    |                                     ^
  14 |                     <h1>
  15 |                         {this.props.title}
  16 |                     </h1>
```

02

WHY REACT?

```
client:47
./src/components/content/PostHeader/PostHeader.jsx
Module build failed: SyntaxError:
/Users/joshbroton/dev/git/vidscrip/site/src/components/content/PostHeader/PostHeader.jsx: Unexpected token (13:46)
   11 |         return (
   12 |             <header className="post__header">
>  13 |                 <a href={this.props.permalink} target="_blank">
      |                                     ^
   14 |                     <h1>
   15 |                         {this.props.title}
   16 |                     </h1>
      |
      |_ at Parser.pp.raise
```

02

WHY REACT?



02 WHY REACT?



02

WHY REACT?



JSX IS OUTSTANDING



REACT SYNTAX IS SIMPLE



REACT FAILS FAST AT COMPILE TIME



THE DEVELOPER TOOLS ARE INCREDIBLE

REACT DEVELOPER EXPERIENCE

Hot reloading, Chrome
React Extension, JSX
compile-time checks,
verbose error messages

02

WHY REACT?



JSX IS OUTSTANDING



REACT SYNTAX IS SIMPLE



REACT FAILS FAST AT COMPILE TIME



THE DEVELOPER TOOLS ARE INCREDIBLE



SPEED. GLORIOUS, GLORIOUS SPEED

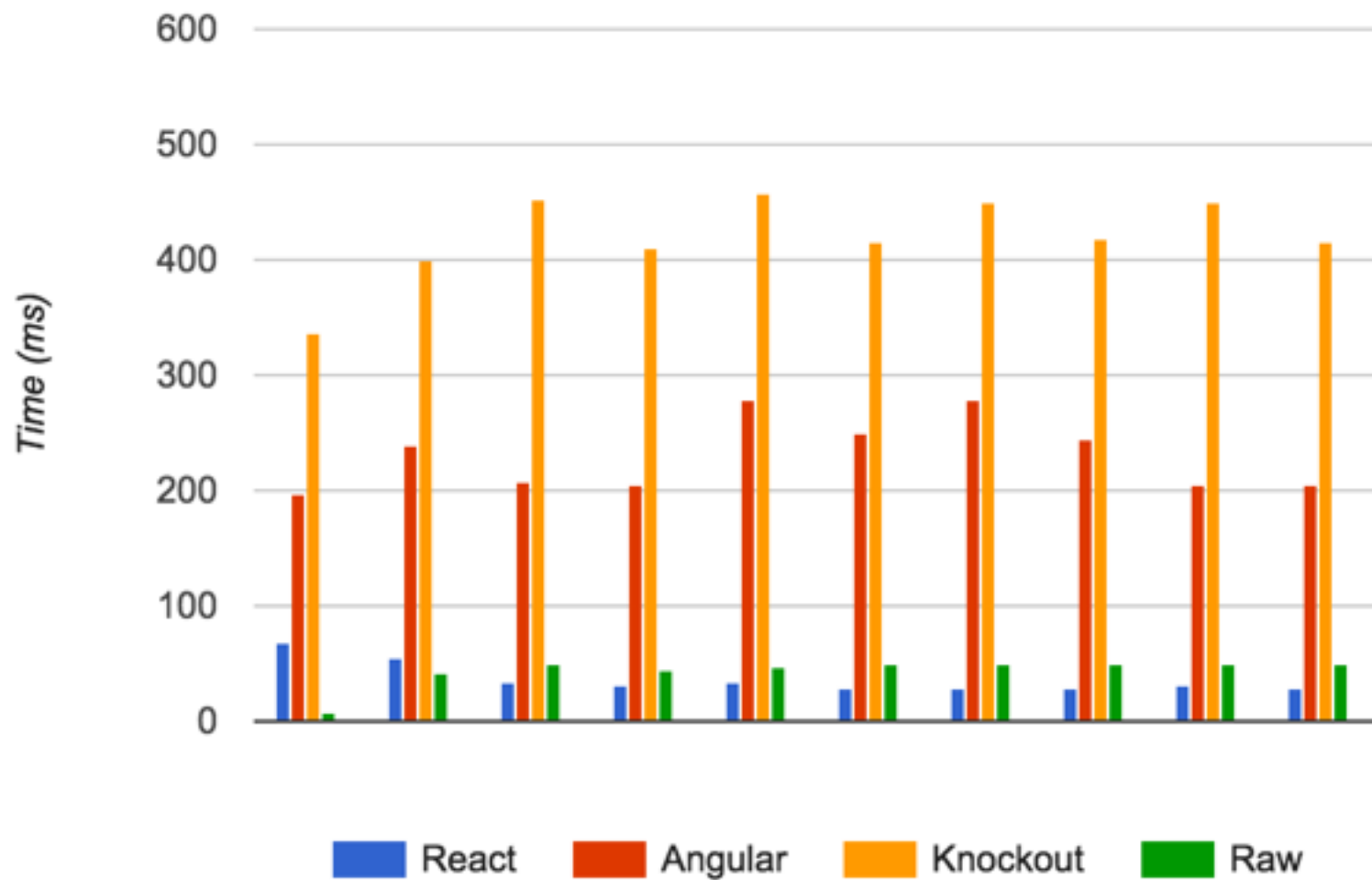
HOW FAST IS REACT?



HOW FAST IS REACT?

10x faster than Angular
18x faster than Knockout
3x faster than DOM
manipulation with jQuery

HOW FAST IS REACT?



HOW FAST IS REACT?

LIBRARY/FRAMEWORK SIZE:

Angular 2: 566k (766k with RxJS)

Ember: 435k

Angular 1: 143k

React + Redux: 139k

03

REACT + WORDPRESS



03

REACT + WORDPRESS

THE WP REST API:
THE FUTURE OF WORDPRESS AND THE WEB

03

REACT + WORDPRESS



WP REST API

[Deactivate](#) | [Edit](#)

JSON-based REST API for WordPress, originally developed as part of GSoC 2013.

Version 2.0-beta13 | By [WP REST API Team](#) | [View details](#)


```
[
- {
  id: 1628,
  date: "2016-04-01T02:32:21",
  date_gmt: "2016-04-01T02:32:21",
- guid: {
    rendered: "http://joshbroton.com/?p=1628"
  },
  modified: "2016-04-01T02:51:13",
  modified_gmt: "2016-04-01T02:51:13",
  slug: "so-you-want-to-change-the-gravity-forms-form-tag",
  type: "post",
  link: "http://joshbroton.com/so-you-want-to-change-the-gravity-forms-form-tag/",
- title: {
    rendered: "So You Want to Change the Gravity Forms Form Tag&#8230;"
  },
- content: {
    rendered: "<p>I had an interesting request from a client today. They use <a href='\"http://www.pardot.com/\"' onclick='\"__gaTracker('send', 'event', 'outbound-article', 'http://www.pardot.com/', 'Pardot');\" target='\"_blank\"'>Pardot</a> to integrate with the forms on their old website, and wanted to do the same with their new WordPress theme. We&#8217;re using the always spectacular <a href='\"http://www.gravityforms.com/\"' onclick='\"__gaTracker('send', 'event', 'outbound-article', 'http://www.gravityforms.com/', 'Gravity Forms');\">Gravity Forms</a> for the forms on the site. </p><p>I did the integrations, and we got an error.</p><pre>The form enc type appears to be incompatible with our pardot form handler according to this article: Wrong Enctype Pardot form handlers can only be integrated with forms using an empty enctype attribute or an enctype of application/x-www-form-urlencoded. Pardot doesn't accept an enctype of multipart/form-data. </pre><p>I didn&#8217;t realize (but now it makes perfect sense) that Gravity Forms defaults to enctype=&#8221;multipart/form-data&#8221;. I scoured the Gravity Forms settings, looking for a way to switch that to the correct enctype without any luck. I hit up a good friend of mine, <a href='\"https://twitter.com/projectgoboy\"' onclick='\"__gaTracker('send', 'event', 'outbound-article', 'https://twitter.com/projectgoboy', 'Rob Harrell');\" target='\"_blank\"'>Rob Harrell</a>, on Twitter, asking if there was a way to edit the form. Thankfully, there is. He sent me <a href='\"https://www.gravityhelp.com/documentation/article/gform_form_tag/\"' onclick='\"__gaTracker('send', 'event', 'outbound-article', 'https://www.gravityhelp.com/documentation/article/gform_form_tag/', 'this page about a filter a developer can use the edit the form tag output by Gravity Forms');\" target='\"_blank\"'>this page about a filter a developer can use the edit the form tag output by Gravity Forms</a>. </p><p>I added this code to the functions.php file to change the enctype of the form:</p><pre>add_filter( 'gform_form_tag', 'form_tag', 10, 2 ); function form_tag( $form_tag, $form ) { if ( $form['id'] != 1 ) { //not the form whose tag you want to change, return the unchanged tag return $form_tag; } $form_tag = preg_replace( '|enctype='(.*)'|", "enctype='application/x-www-form-urlencoded'", $form_tag ); return $form_tag; } </pre><p>Breaking down that code:</p><pre>//Enter the ID of the form you want to alter here: if ( $form['id'] != 1 ) { //Alter the <form> tag however you like here: $form_tag = preg_replace( '|enctype='(.*)'|", "enctype='application/x-www-form-urlencoded'", $form_tag ); //This can be just about any replacement you want, or you can just return a new string that is the new form tag return '<form class='\"gravity-form-or-some-other-class\" action='\"post.php\" enctype='\"application/x-www-form-urlencoded\"'> </pre><p>Pardot is picky about the form they integrate with, and thankfully the Gravity Forms developers made it possible to alter the &lt;form&gt; tag however we need to.</p> "
  },
- excerpt: {
    rendered: "<p>I had an interesting request from a client today. They use Pardot to integrate with the forms on their old website, and wanted to do the same with their new WordPress theme. We&#8217;re using the always spectacular Gravity Forms for &hellip; <a href='\"http://joshbroton.com/so-you-want-to-change-the-gravity-forms-form-tag/\"'>Continued</a></p> "
  },
}
```

03

REACT + WORDPRESS

```
<article id="post-<?php the_ID(); ?>" <?php post_class('post'); ?>
  role="article" itemscope itemtype="http://schema.org/BlogPosting">
  <header class="content__header post__header">
    <h1 class="content__title post__title"
      title="<?php the_title_attribute(); ?>" itemprop="headline">
      <?php the_title(); ?>
    </h1>
    <aside class="byline vcard">
      <time class="byline__date updated" datetime="<?php echo
the_time('Y-m-j'); ?>" pubdate>
        <?php the_time(get_option('date_format')); ?>
      </time>
      by
      <span class="byline__author"><?php the_author_posts_link(); ?></
span>
    </aside>
  </header>
```


03

REACT + WORDPRESS

```
class PostView extends Component {
  componentWillMount() {
    document.body.classList.add('post-list', 'blog');
  }
  componentWillUnmount() {
    document.body.classList.remove('post-list', 'blog');
  }
  render() {
    return (
      //Your JSX goes here
    )
  }
}
```

03

REACT + WORDPRESS

```
<article id={'post-' + post.id} className="post"
  role="article" itemscope itemtype="http://schema.org/BlogPosting">
  <header class="content__header post__header">
    <h1 class="content__title post__title" itemprop="headline">
      {post.title}
    </h1>
    <aside class="byline vcard">
      <time class="byline__date updated">
        {moment(post.date).format()}
      </time>
      by <span class="byline__author">{post.author}</span>
    </aside>
  </header>
  <section class="post__content" itemprop="articleBody">
    <PostThumbnail post={post} />
    <div class="content--inner post__content--inner">(); ?>
      {post.excerpt? post.excerpt : post.content}
```

03

REACT + WORDPRESS

```
<article id={'post-' + post.id} className="post"
  role="article" itemscope itemtype="http://schema.org/BlogPosting">
  <PostHeaderView title={post.title}
    date={post.date}
    author={post.author} />
  <PostContentView thumbnail={post.thumbnail}
    content={post.content} />
  <PostFooterView categories={post.categories}
    tags={post.tags} />
</article>
```

03

REACT + WORDPRESS

```
<PostView post={post} />
```

```
<section className="posts--list">  
  { posts.map((post) => { return <PostView post={post} /> }) }  
</section>
```

03

HEADLESS WORDPRESS

1. index.php file

2. change

```
define('WP_USE_THEMES', true);
```

to

```
define('WP_USE_THEMES', false);
```

3. You're done!

03

HEADLESS WORDPRESS

1. front-page.php file
2. ©2016 `api.joshbroton.com`
3. You're done!

MORE RESOURCES

- <https://facebook.github.io/react/docs/getting-started.html>
- <https://frontendmasters.com/courses/>
- <http://tri.be/blog/redux-react-and-the-wordpress-rest-api-v2/>
- <https://camjackson.net/post/9-things-every-reactjs-beginner-should-know>
- <https://github.com/reactjs/react-a11y>

JOSH BROTON

@joshbroton || joshbroton.com

