

HASTENING REACT SSR WITH COMPONENT MEMOIZATION AND TEMPLATIZATION

@WalmartLabs



Web App Performance

Increasing User Demand

INTERNET USERS ARE INCREASINGLY DEMANDING

In 1999, the average user was willing to wait 8 seconds for a page to load. By 2010, **57% of online shoppers said they would abandon a page after 3 seconds.**



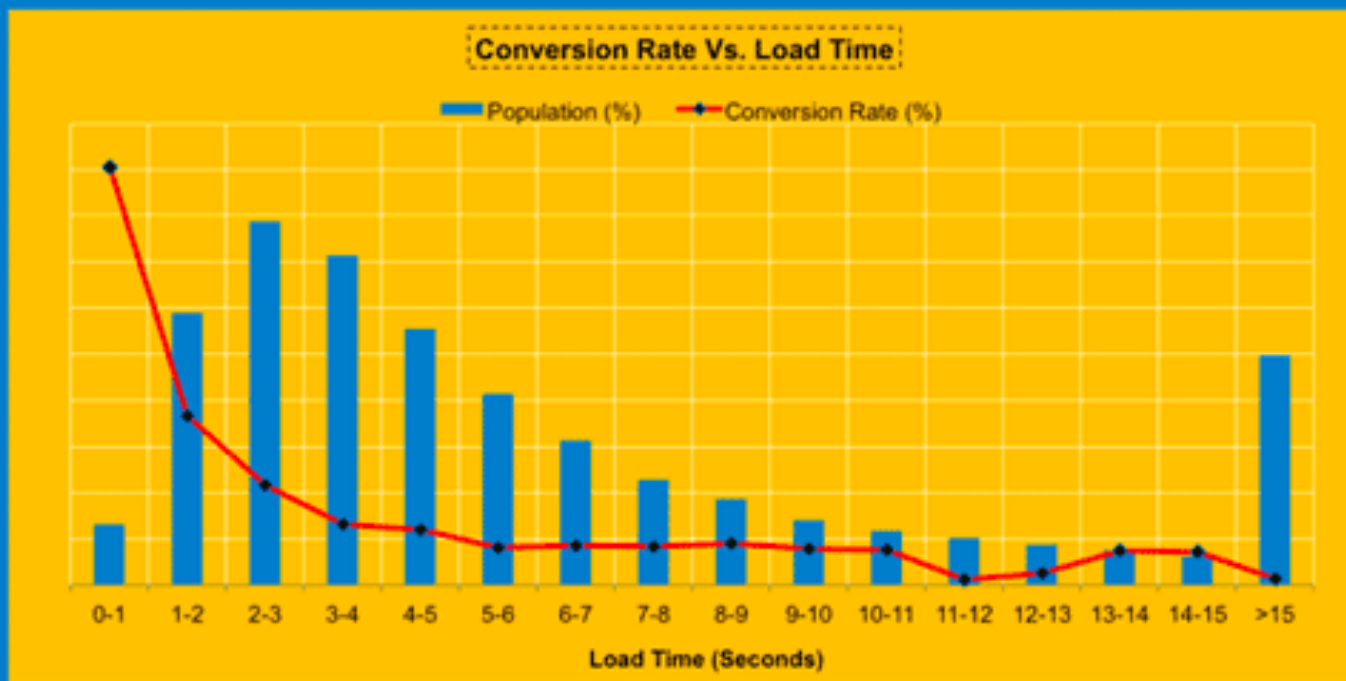
Time is Money

Impact of site performance on overall site conversion rate....

Baseline – 1 in 2 site visits had response time > 4 seconds

* Sharp decline in conversion rate as average site load time increases from 1 to 4 seconds

* Overall average site load time is lower for the converted population (3.22 Seconds) than the non-converted population (6.03 Seconds)



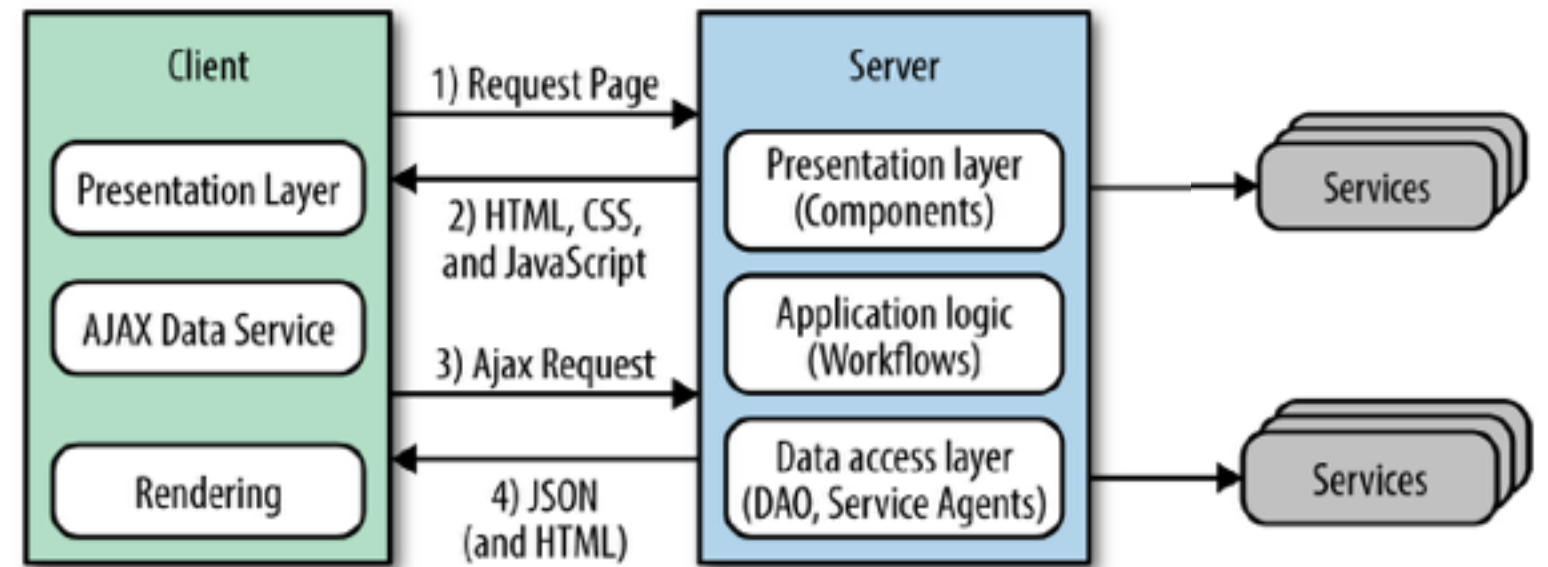
Note: Load Time here is the time taken from head of the page to page ready (T_Page)

- For every 1 second of improvement, experienced up to a 2% increase in conversions
- For every 100 ms of improvement, grew incremental revenue by up to 1%

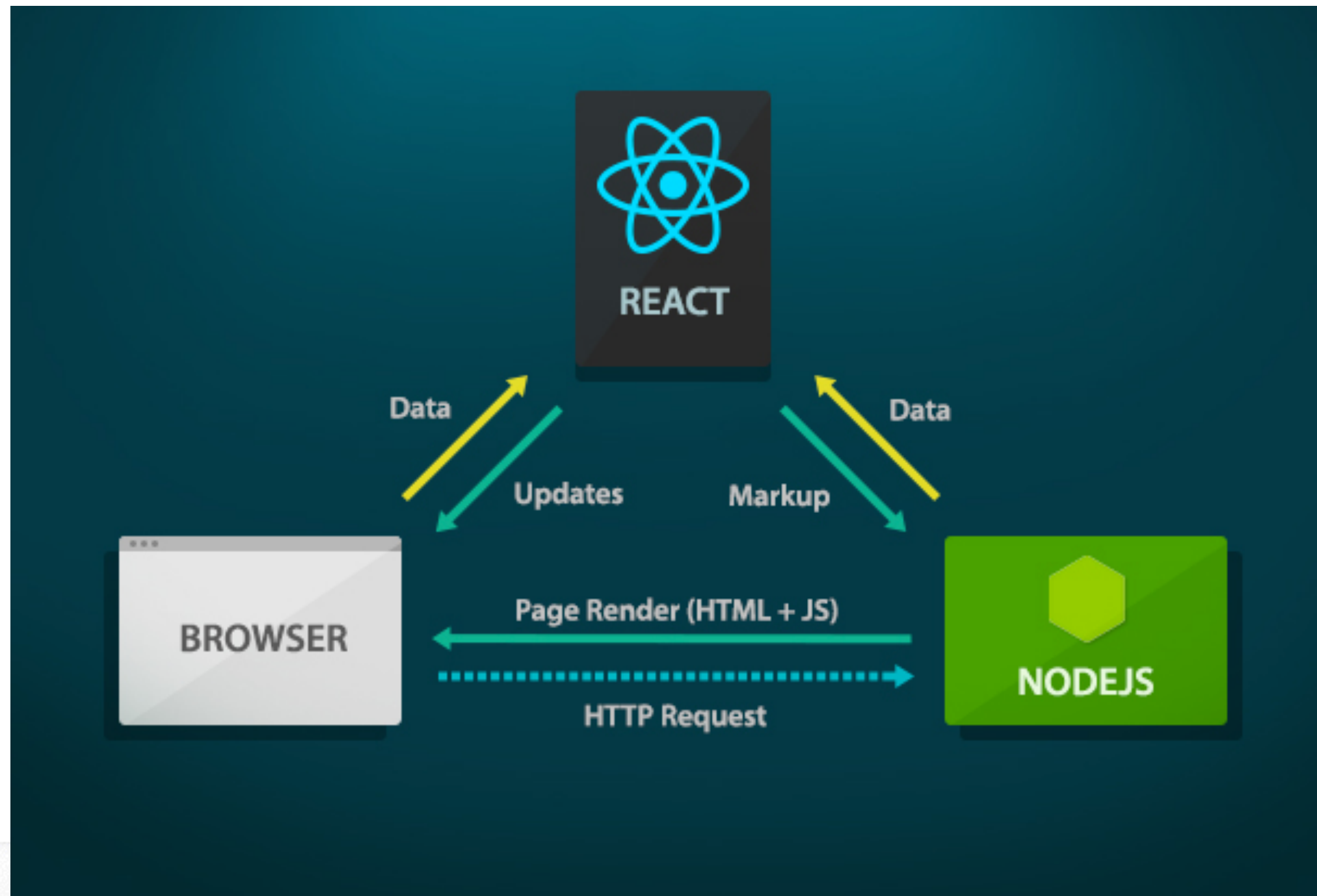
Source: <http://www.globaldots.com/how-website-speed-affects-conversion-rates>

Server-Side Rendering (SSR)

- Better user experience of the initial page load
- Better search engine ranking

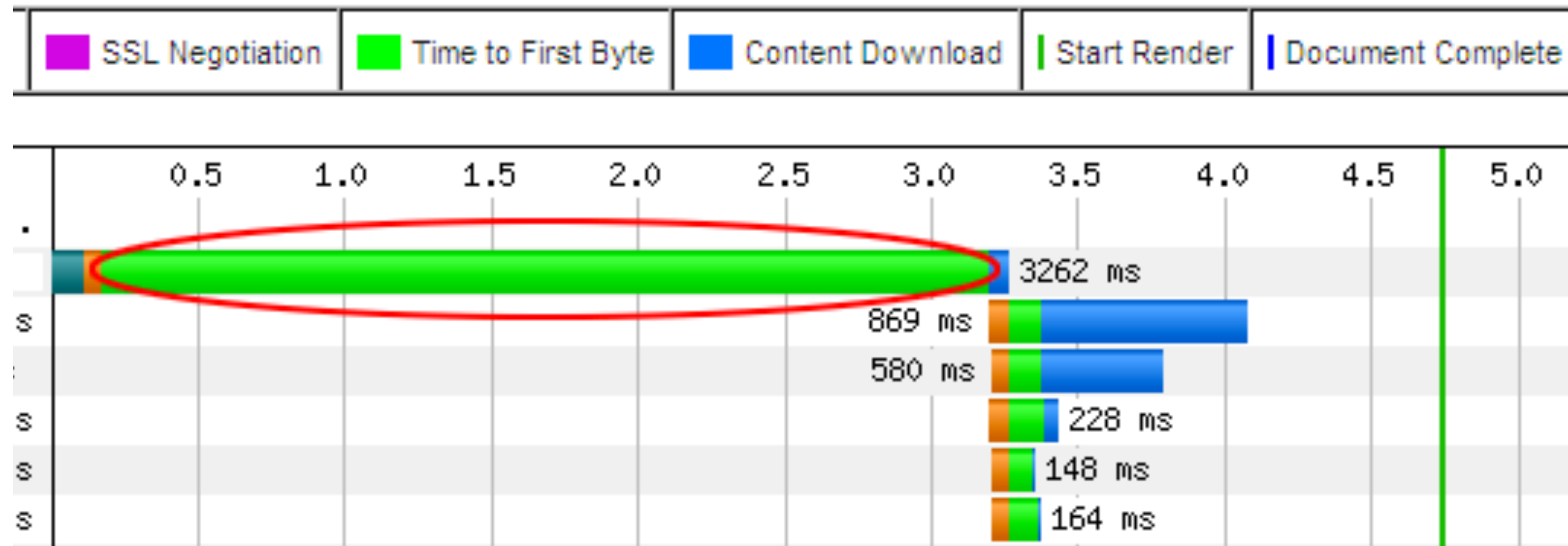


SSR with React



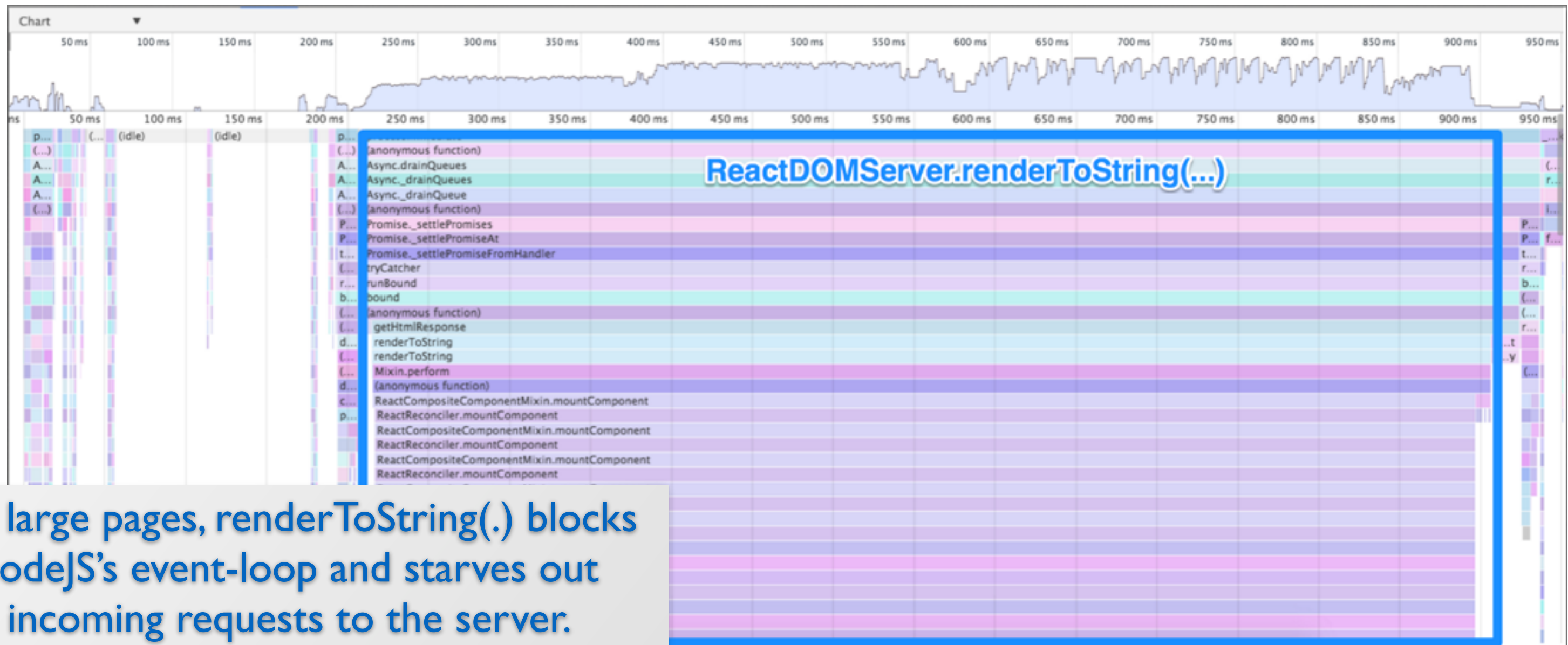
Server-Side React

Waterfall View



Bad User Experience!

Server-Side React



CPU profile for one request

Server-Side **React**

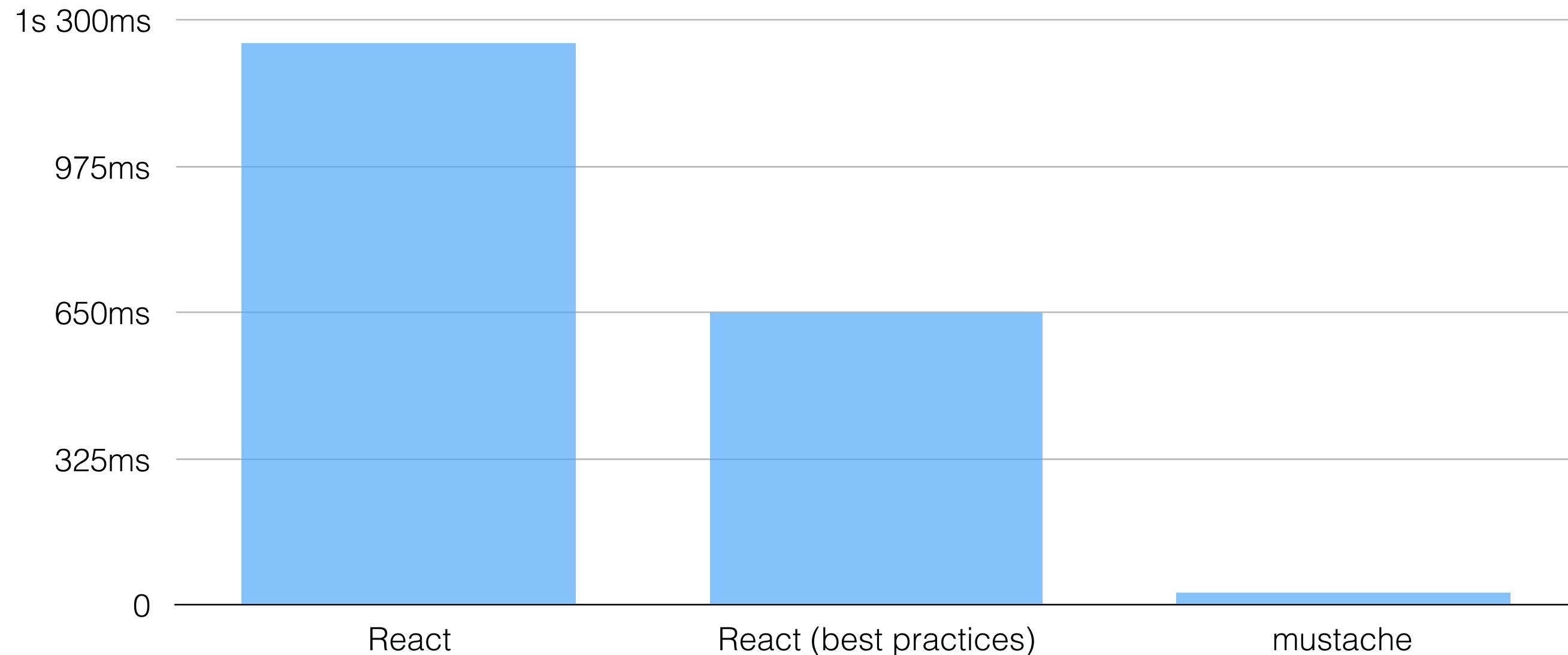


**React.js Conf 2015
Q&A with the team**

- Facebook: “Funny story about Server Rendering - it wasn’t actually designed that way.” (Sebastian Markbåge)
- Facebook: “we don’t use it that heavily, which is why we haven’t really invested strongly in it.” (Sebastian Markbåge)

<https://youtu.be/EPpkboSKvPI?t=7m48s>

Server-Side React



Server-Side **React**

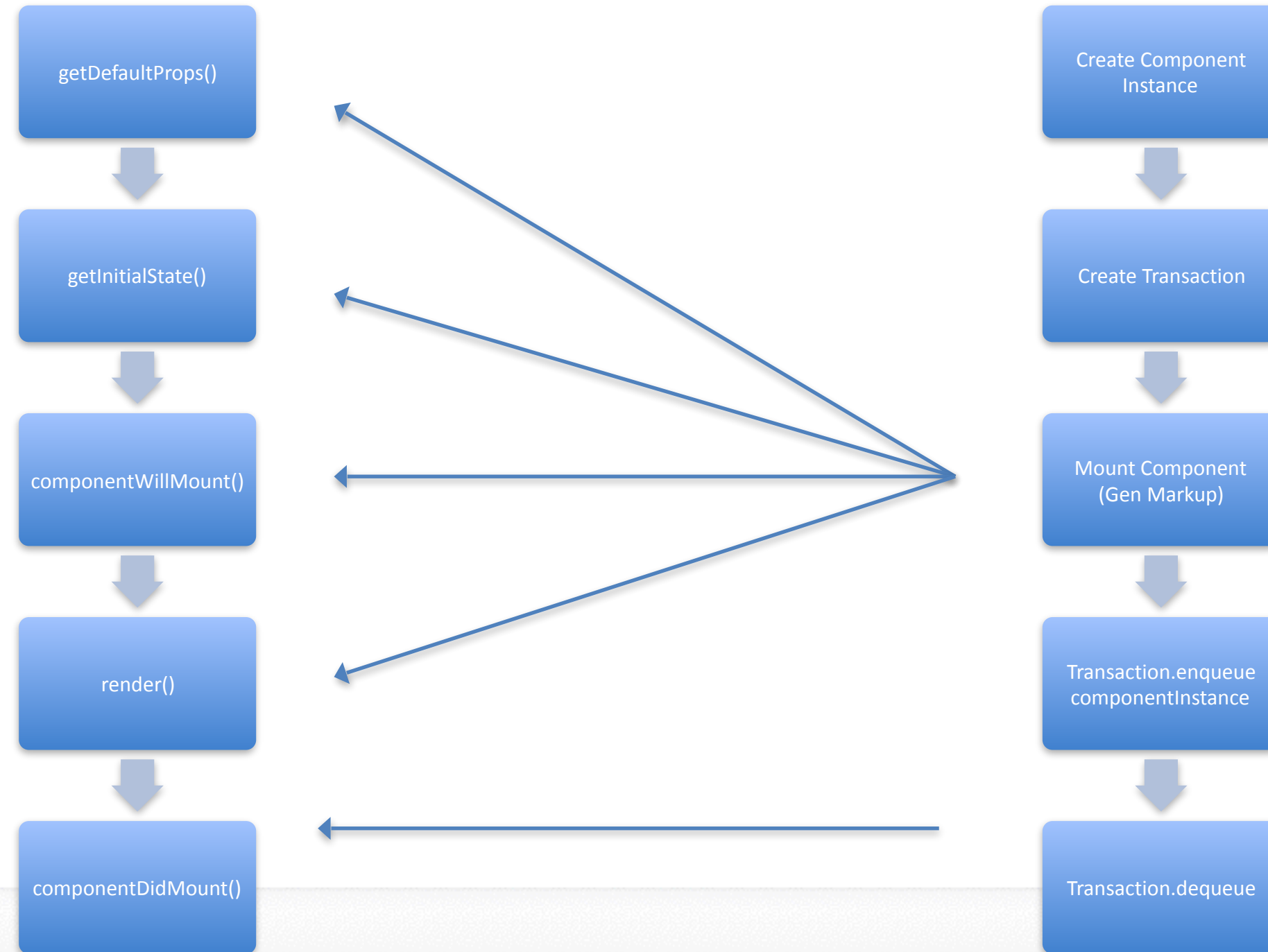


ReactJS SF meetup in January 2016

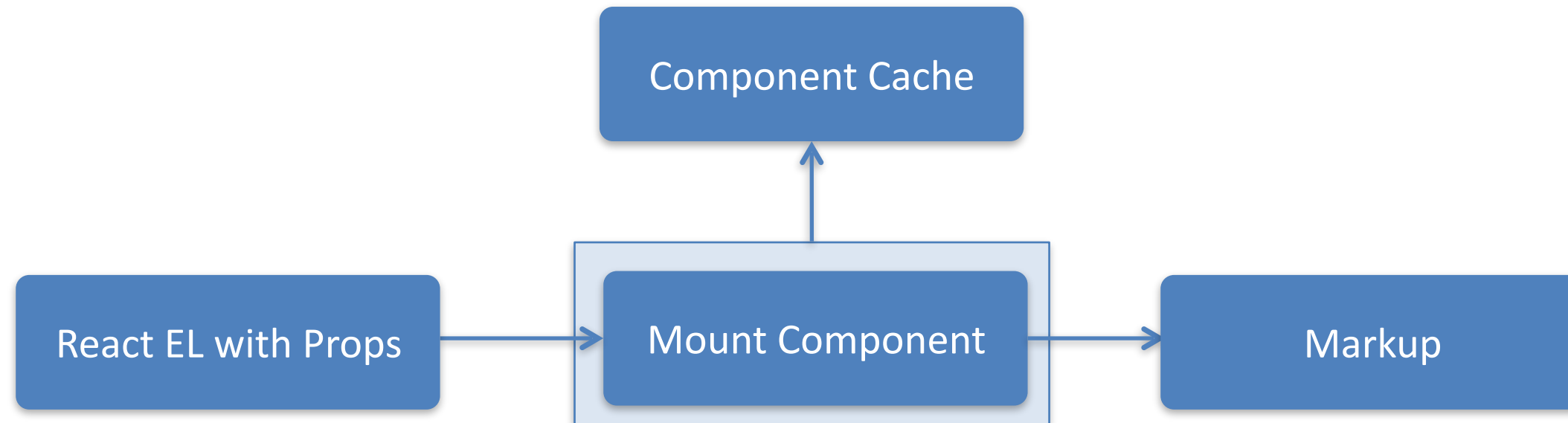
<https://github.com/aickin/react-dom-stream>

- Looked at *react-dom-stream* and *redfin/react-server* to improve TTFB and ATF improvement
- Challenges
 - Forks react and introduces new interfaces, i.e. not “React at heart”
 - Doesn't solve CPU total time

React Rendering Lifecycle



Double Clicking into Mount Component



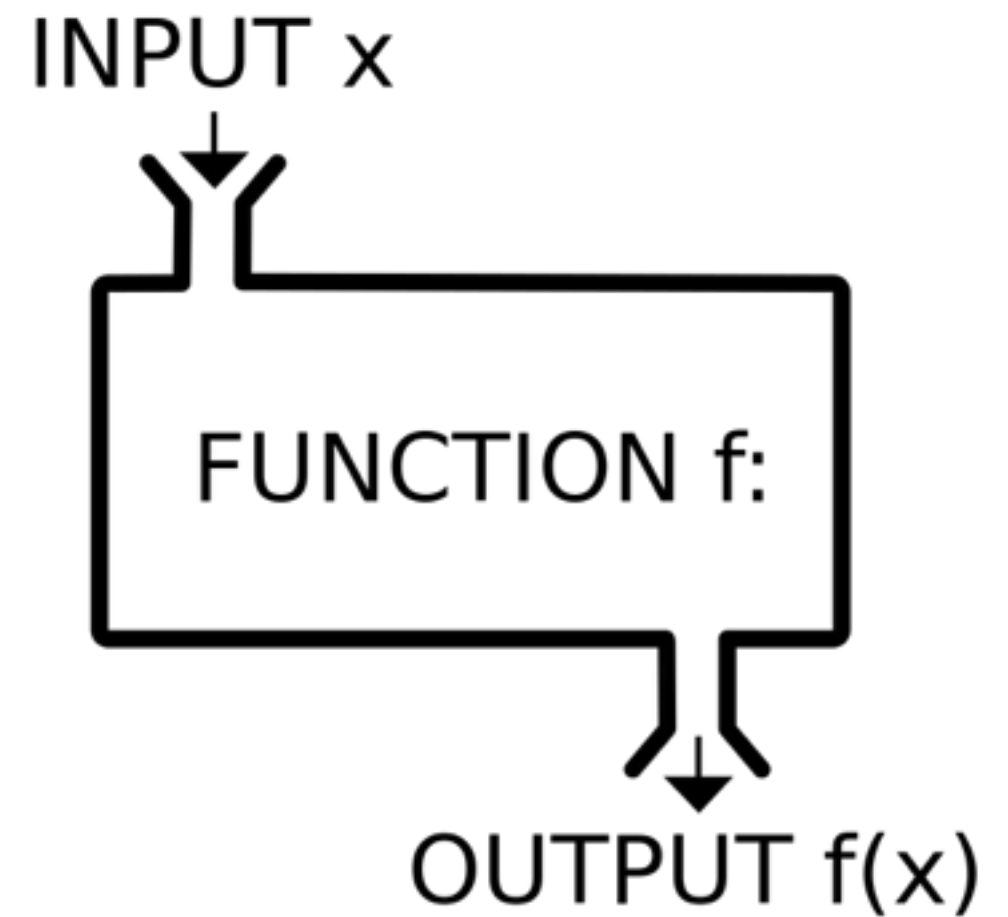
Given a set of properties, the markup generated is always the same

Memoizing Components

The same “text” prop will always return the same helloworld html string

```
class HelloWorldComponent extends React.Component {  
  render() {  
    return <div>Hello {this.props.text}!</div>;  
  }  
}
```

When text is “World” the output is:
<div>Hello World!</div>



Memoizing Components

The same “text” prop will always return the same helloworld html string

```
class HelloWorldComponent extends React.Component {  
  render() {  
    return <div>Hello {this.props.text}!</div>;  
  }  
}
```

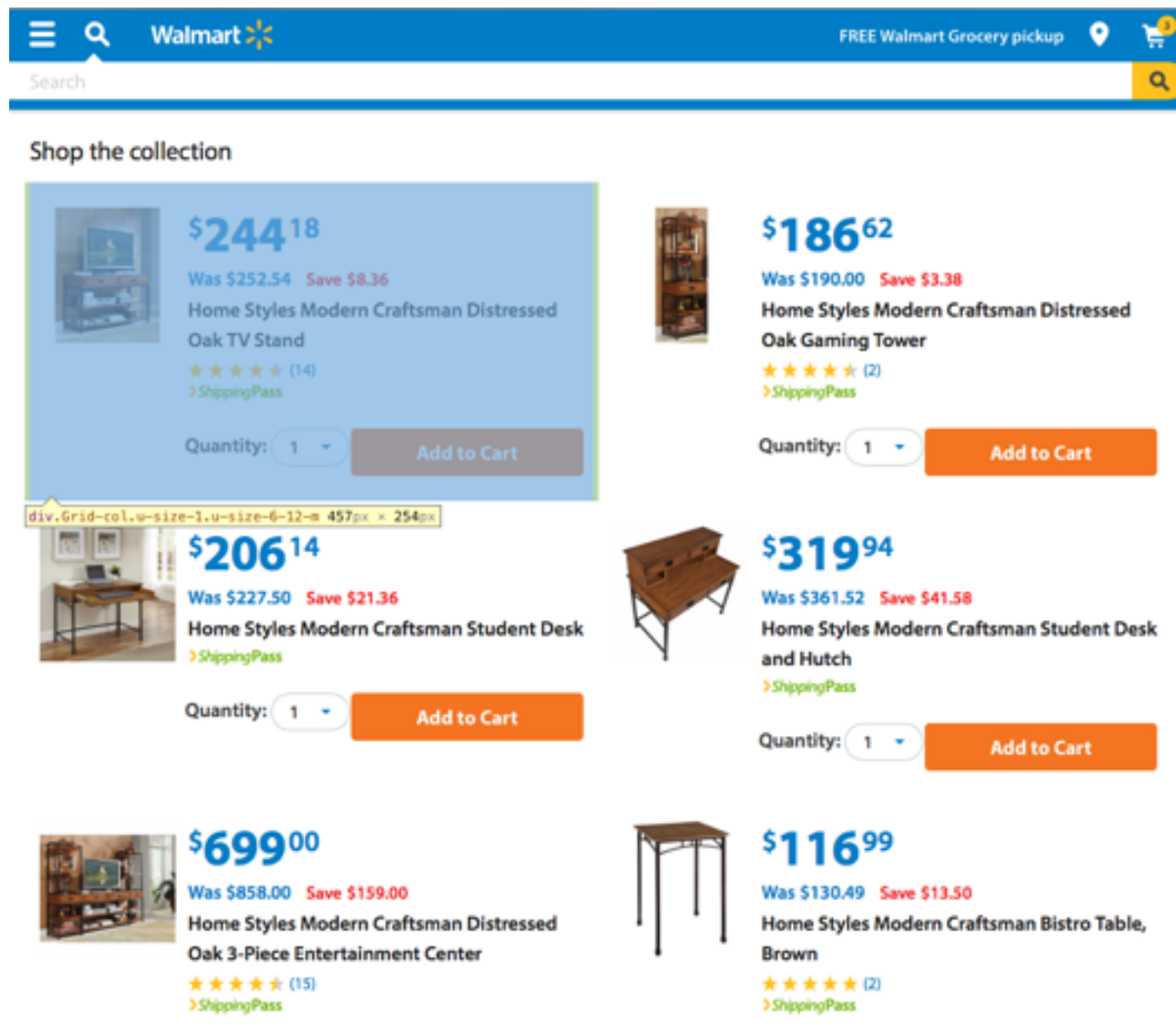
When text is “World” the output is:
<div>Hello World!</div>

```
var componentOptimization =  
  require("electrode-react-ssr-  
  optimization");  
  
var componentOptimizationRef =  
  componentOptimization({  
    components: {  
      'HelloWorldComponent': {  
        keyAttrs: ["text"]  
      }  
    }  
  });
```

OUTPUT f(x)

Memoizing **Demo**

Templatizing Components



```
var React = require('react');

var ProductView = React.createClass({
  render: function() {
    return (
      <div className="product">
        <img src={this.props.product.image}/>
        <div className="product-detail">
          <p className="name">{this.props.product.name}</p>
          <p className="description">{this.props.product.description}</p>
          <p className="price">Price: ${this.props.product.price}</p>
          <button type="button" onClick={this.addToCart}>
            Add To Cart
          </button>
        </div>
      </div>
    );
  }
});

module.exports = ProductView;
```


Templatizing Components

- Dynamic props that would be different for each product.
- Switch the corresponding props with template delimiters (i.e. `${ prop_name }`) during react component rendering cycle.
- The template is then compiled, cached, executed and the markup is handed back to React.

```
var React = require('react');

var ProductView = React.createClass({
  render: function() {
    return (
      <div className="product">
        <img src={this.props.product.image}/>
        <div className="product-detail">
          <p className="name">{this.props.product.name}</p>
          <p className="description">{this.props.product.description}</p>
          <p className="price">Price: ${this.props.product.price}</p>
          <button type="button" onClick={this.addToCart}>
            Add To Cart
          </button>
        </div>
      </div>
    );
  }
});

module.exports = ProductView;
```

Templatizing Components

- Dynamic props that would be different for each product.
- Switch the corresponding props with template delimiters (i.e. `${ prop_name }`) during react component rendering cycle.
- The template is then compiled, cached, executed and the markup is handed back to React.

```
<div className="product">  
  <img src=${product_image}/>  
  <div className="product-detail">  
    <p className="name">${product_name}</p>  
    <p className="description">${product_description}</p>  
    <p className="price">Price: ${selected_price}</p>  
    <button type="button" onClick={this.addToCart}>  
      Add To Cart  
    </button>  
  </div>  
</div>
```

Templatizing Components

```
var componentOptimization =
require("electro-react-ssr-
optimization");

var componentOptimizationRef =
componentOptimization({
  components: {
    "ProductView": {
      templateAttrs: ["product.image",
        "product.name",
        "product.description",
        "product.price"]
    },
  },
});
```

```
var React = require('react');
```

```
var ProductView = React.createClass({
  render: function() {
    return (
      <div className="product">
        <img src={this.props.product.image}/>
        <div className="product-detail">
          <p className="name">{this.props.product.name}</p>
          <p className="description">{this.props.product.description}</p>
          <p className="price">Price: ${this.props.product.price}</p>
          <button type="button" onClick={this.addToCart}>
            Add To Cart
          </button>
        </div>
      </div>
    );
  }
});
```

```
module.exports = ProductView;
```



\$699⁰⁰

Was \$858.00 Save \$159.00

Home Styles Modern Craftsman Distressed
Oak 3-Piece Entertainment Center

★★★★★ (15)
ShippingPass



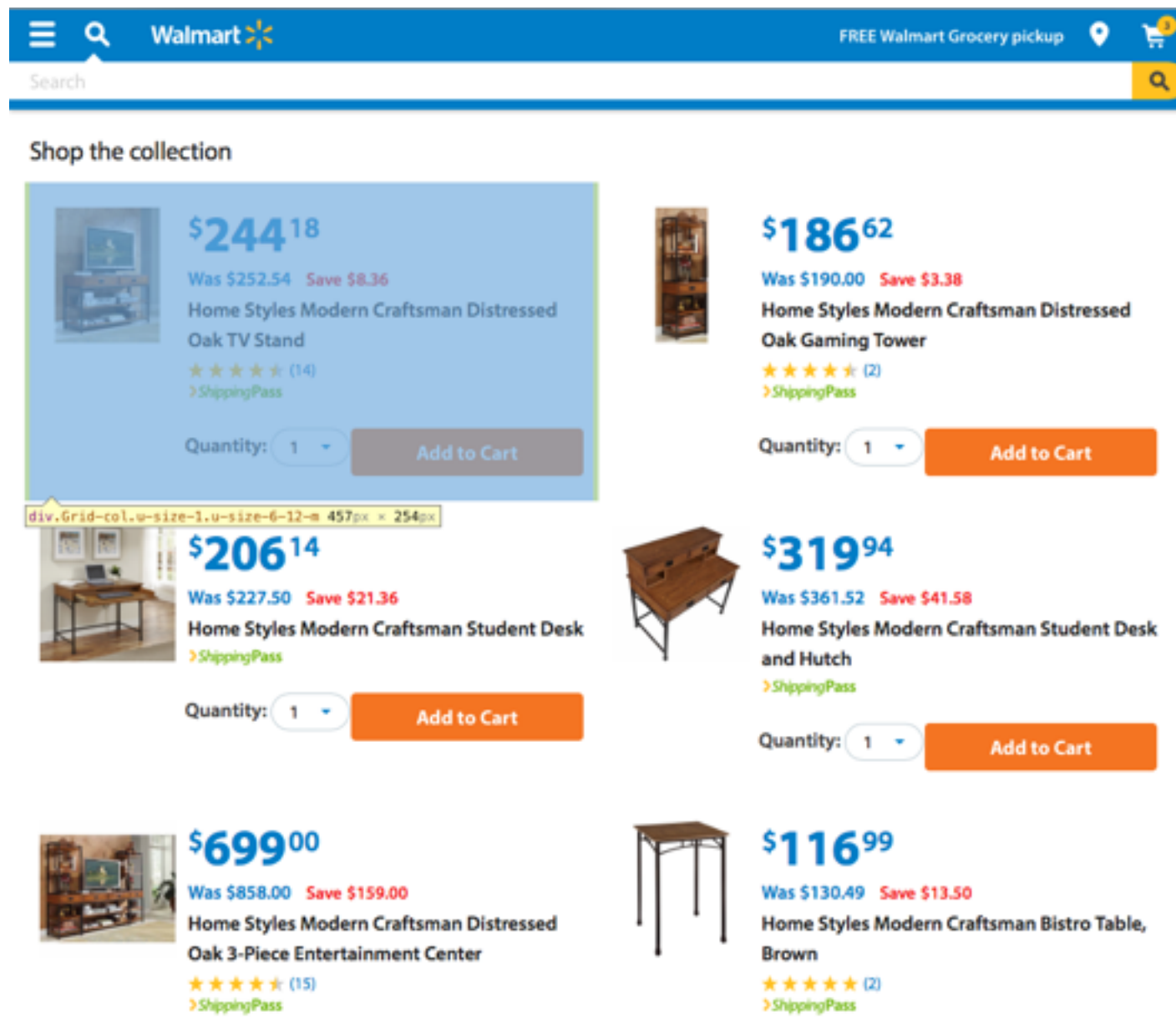
\$116⁹⁹

Was \$130.49 Save \$13.50

Home Styles Modern Craftsman Bistro Table,
Brown

★★★★★ (2)
ShippingPass

Templatizing Components



```
var React = require('react');

var ProductView = React.createClass({
  render: function() {
    return (
      <div className="product">
        <img src={this.props.product.image}/>
        <div className="product-detail">
          <p className="name">{this.props.product.name}</p>
          <p className="description">{this.props.product.description}</p>
          <p className="price">Price: ${this.props.product.price}</p>
          <button type="button" onClick={this.addToCart}
            {this.props.inventory ? 'Add To Cart' : 'Sold Out'}
          </button>
        </div>
      </div>
    );
  }
});

module.exports = ProductView;
```

Templatizing Components

```
var componentOptimization =
require("electro-react-ssr-
optimization");

var componentOptimizationRef =
componentOptimization({
  components: {
    "ProductView": {
      templateAttrs: ["product.image",
        "product.name",
        "product.description",
        "product.price"],
      keyAttrs: ["product.inventory"]
    },
  },
});
```

```
var React = require('react');

var ProductView = React.createClass({
  render: function() {
    return (
      <div className="product">
        <img src={this.props.product.image}/>
        <div className="product-detail">
          <p className="name">{this.props.product.name}</p>
          <p className="description">{this.props.product.description}</p>
          <p className="price">Price: ${this.props.product.price}</p>
          <button type="button" onClick={this.addToCart}
            {this.props.inventory ? 'Add To Cart' : 'Sold Out'}
          </button>
        </div>
      </div>
    );
  },
});

module.exports = ProductView;
```



Templatizing Components

Cached Template 1 - Sold Out

```
<div className="product">  
  <img src=${product_image}/>  
  <div className="product-detail">  
    <p className="name">${product_name}</p>  
    <p className="description">${product_description}</p>  
    <p className="price">Price: ${selected_price}</p>  
    <button type="button" onClick={this.addToCart}>  
      Sold Out  
    </button>  
  </div>  
</div>
```

Cached Template 2 - Add To Cart

```
<div className="product">  
  <img src=${product_image}/>  
  <div className="product-detail">  
    <p className="name">${product_name}</p>  
    <p className="description">${product_description}</p>  
    <p className="price">Price: ${selected_price}</p>  
    <button type="button" onClick={this.addToCart}>  
      Add To Cart  
    </button>  
  </div>  
</div>
```

Templatizing **Demo**

Walmart Gift Cards Registry Links Weekly Ads Store Finder Track Order Credit Card Help

All Departments Savings Showcase My Local Store Pick it up TODAY! Tips & Ideas FREE Walmart Grocery pickup


Home > Furniture > Bedroom Furniture > Bedroom Sets

Home Styles American Craftsman Furniture Collection


Home Styles American Craftsman Collection

[Read more...](#)


Shop the collection




\$251.28
Was \$252.50 Save \$1.22
Home Styles Modern Craftsman Distressed Oak TV Stand
★★★★★ (14)
Quantity: 1




\$186.62
Was \$190.00 Save \$3.38
Home Styles Modern Craftsman Distressed Oak Gaming Tower
★★★★★ (2)
Quantity: 1




\$214.12
Was \$227.50 Save \$13.38
Home Styles Modern Craftsman Student Desk
Quantity: 1



\$337.65
Was \$361.52 Save \$23.87
Home Styles Modern Craftsman Student Desk and Hutch
Quantity: 1



\$699.00
Was \$858.00 Save \$159.00
Home Styles Modern Craftsman Distressed Oak 3-Piece Entertainment Center
★★★★★ (15)
Quantity: 1



\$122.99
Was \$130.49 Save \$7.50
Home Styles Modern Craftsman Bistro Table, Brown
★★★★★ (2)
Quantity: 1

Be the first to save! [Privacy policy](#)

Stay connected [Facebook](#) [Pinterest](#) [Twitter](#) [YouTube](#) [Walmart.com](#)

Financial Services	Get to Know Us	Walmart.com	Customer Service	In the Spotlight
Walmart MoneyCenter	About Walmart.com	Terms of Use	Returns Policy	[+] Baby
Walmart Credit Card	Corporate	About Our Ads	Product Recalls	[+] Electronics
Apply Now	Suppliers	Affiliate Program	Int'l Customers	[+] Fitness
Manage Card Account	Marketplace	CA Privacy Rights	Tax Exempt Program	[+] Spring Cleaning
Product Care Plans	Careers	Privacy & Security	Contact Us	[+] International Sites
Walmart Pay	@Walmartlabs		Feedback	

© Walmart Stores, Inc. To ensure we're able to help you as best we can, please include your reference number: Z3FOCTB90E

green blocks indicating markup that was cached on the server

Server-Side React

with rendering optimization



CPU profile for one request

Thank You



<https://github.com/walmartlabs/electro-react-ssr-optimization>