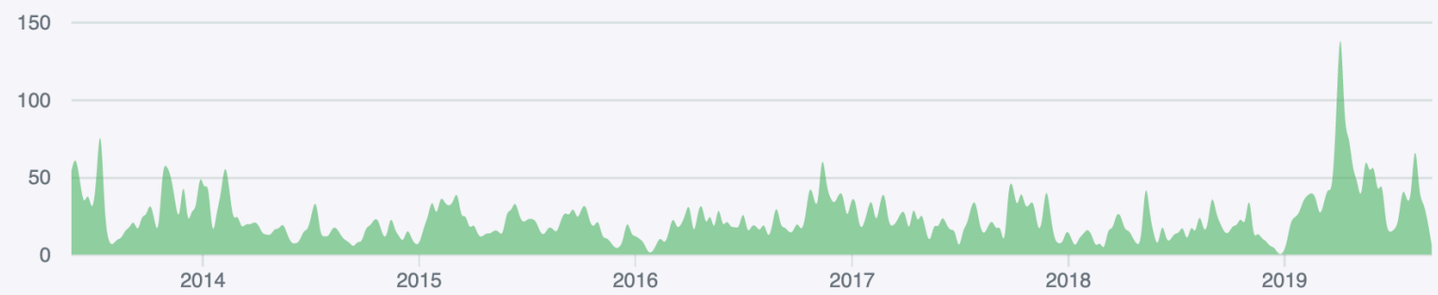
Building Secure React Applications

Disclaimer

May 26, 2013 – Sep 9, 2019

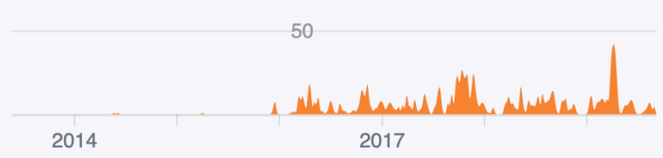
Contributions: **Commits**

Contributions to master, excluding merge commits



gaearon #1

1,264 commits 372,152 ++ 319,592 --



bvaughn #2

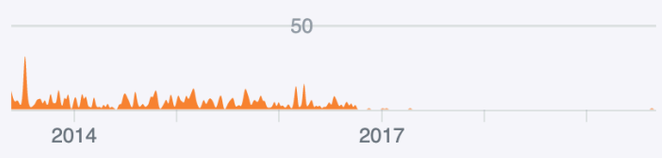
1,072 commits 418,183 ++ 370,319 --

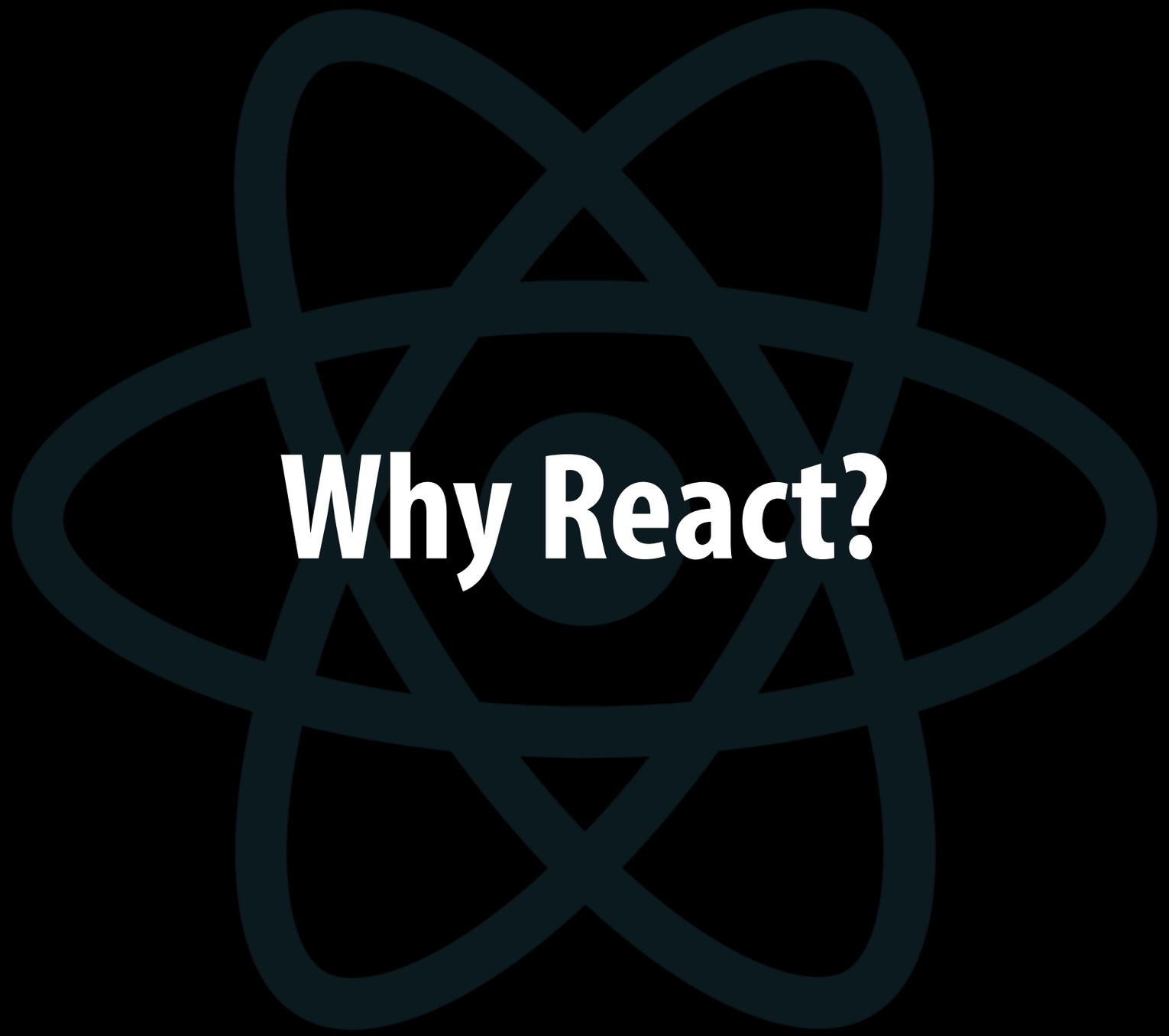
sophiebits #3

869 commits 117,505 ++ 45,665 --

zpao #4

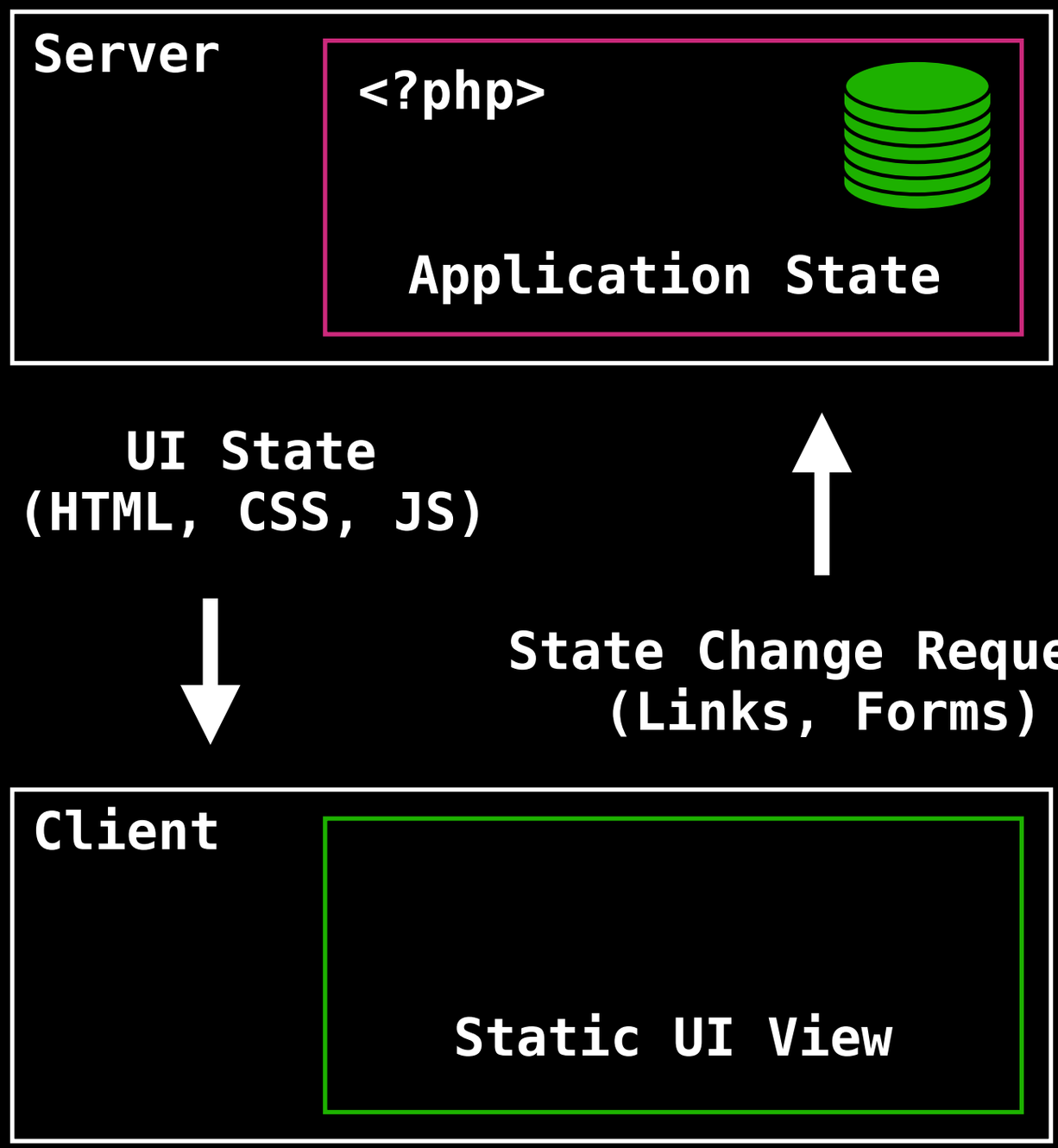
821 commits 87,398 ++ 131,911 --



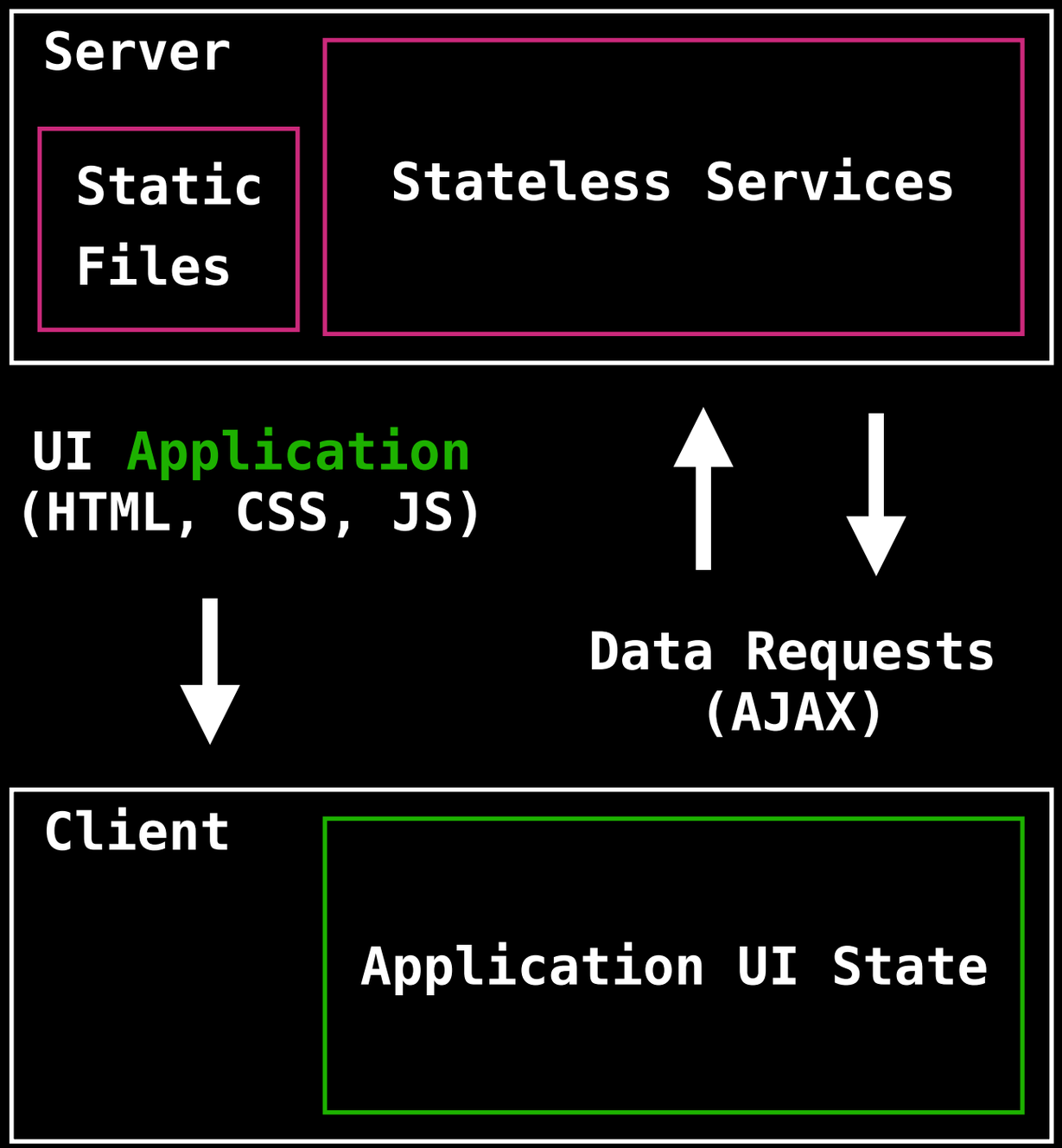


Why React?

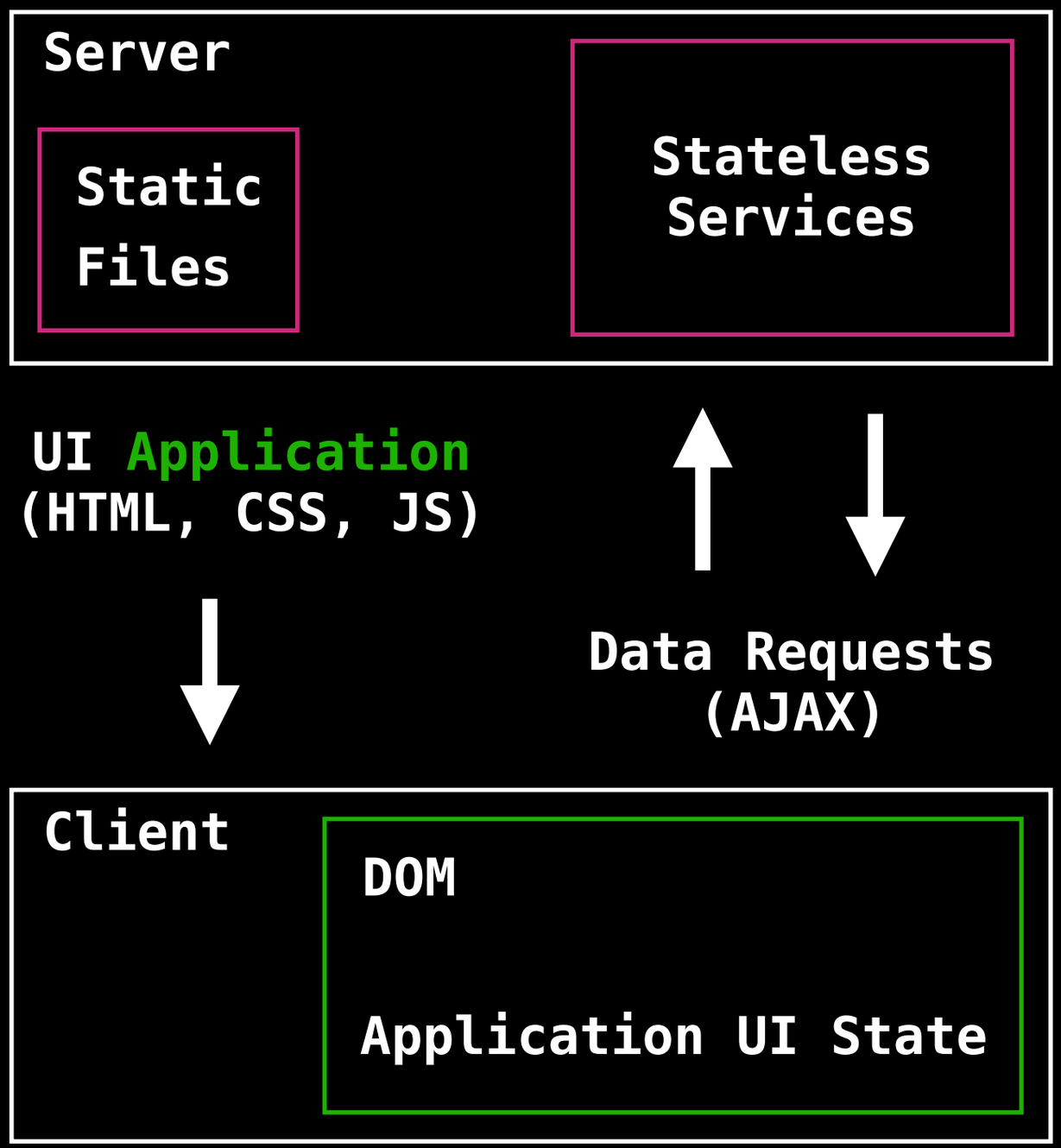
Prior to 2005



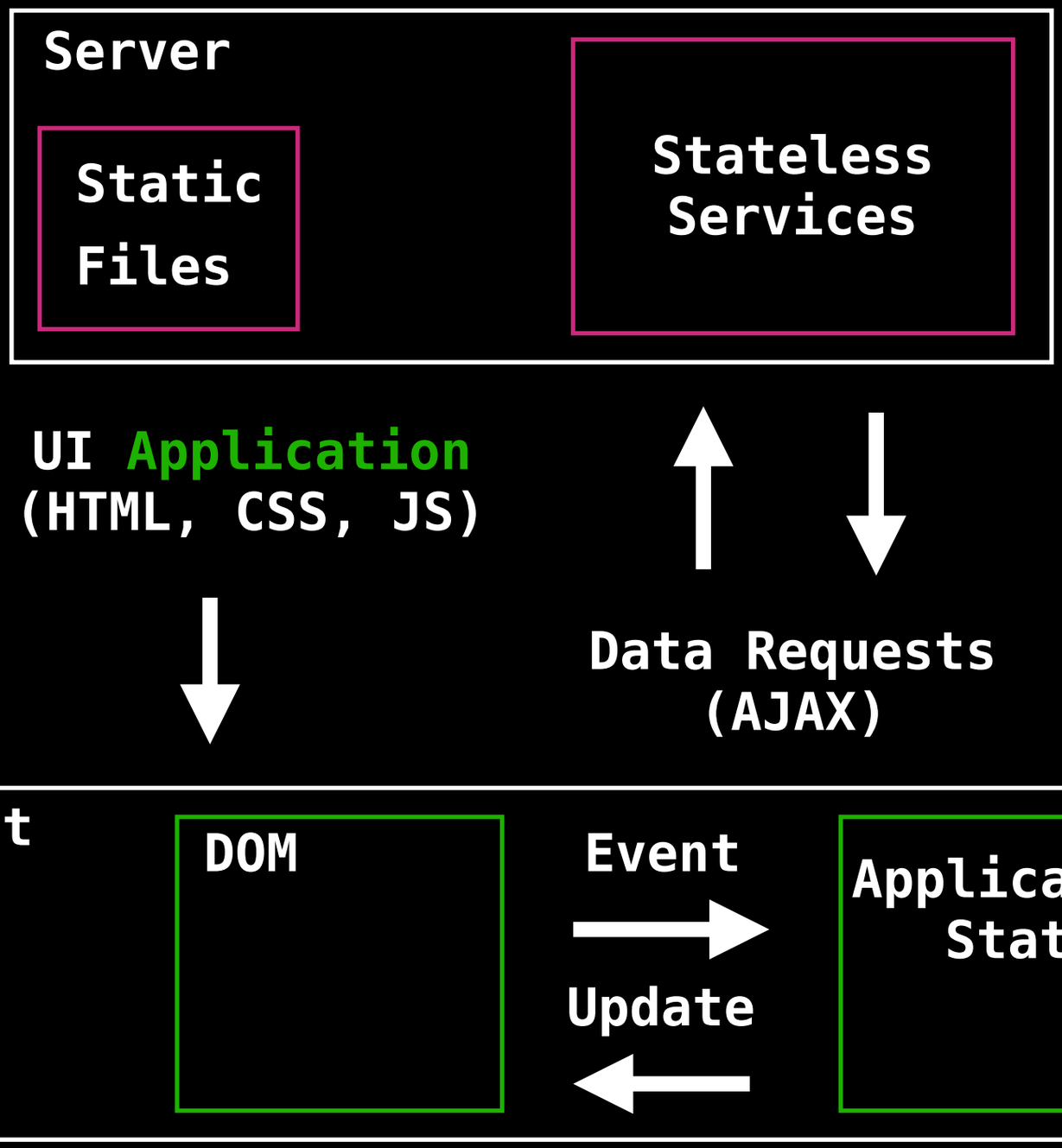
Google Maps 2005



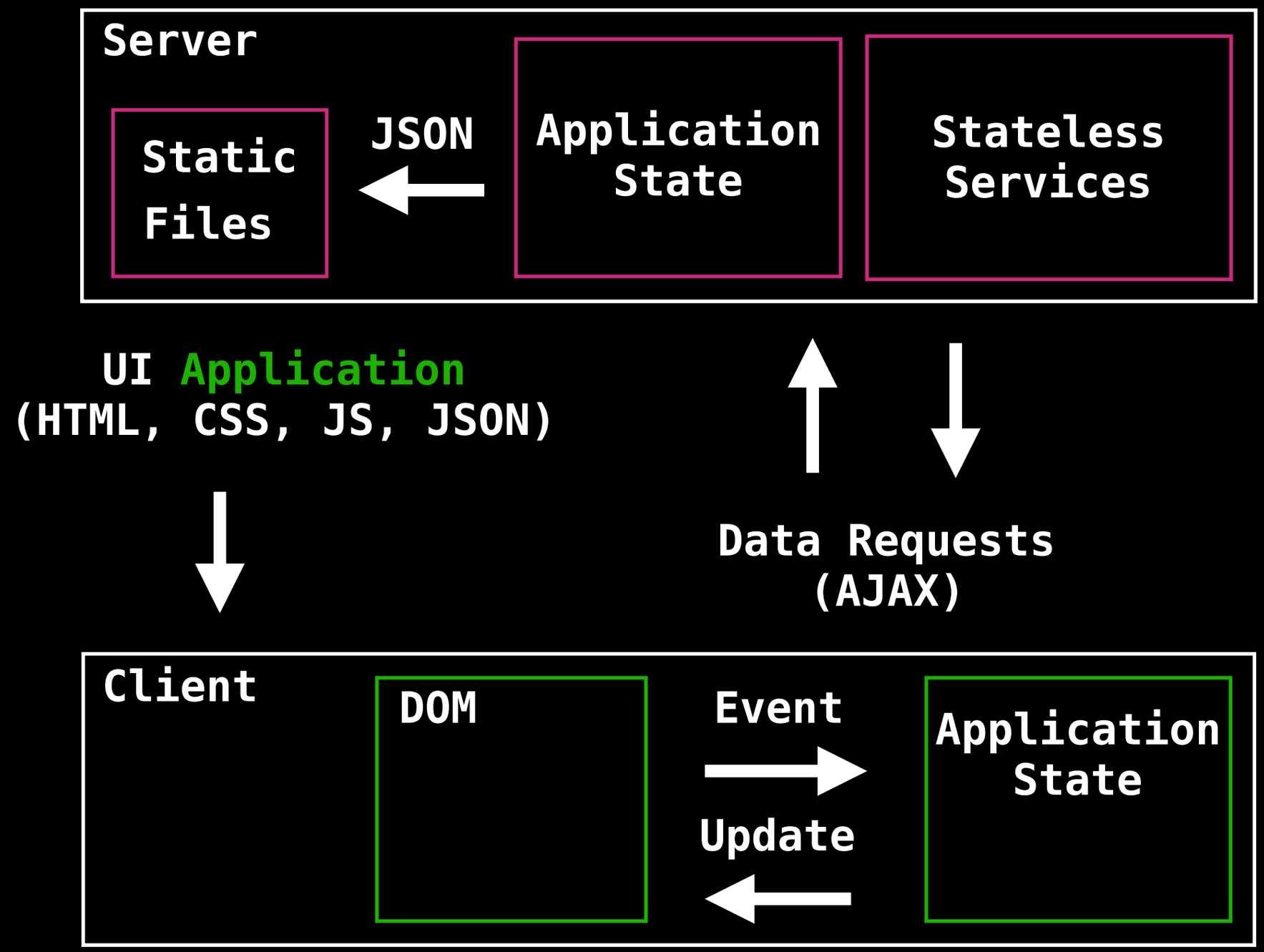
jQuery Era 2005-2013



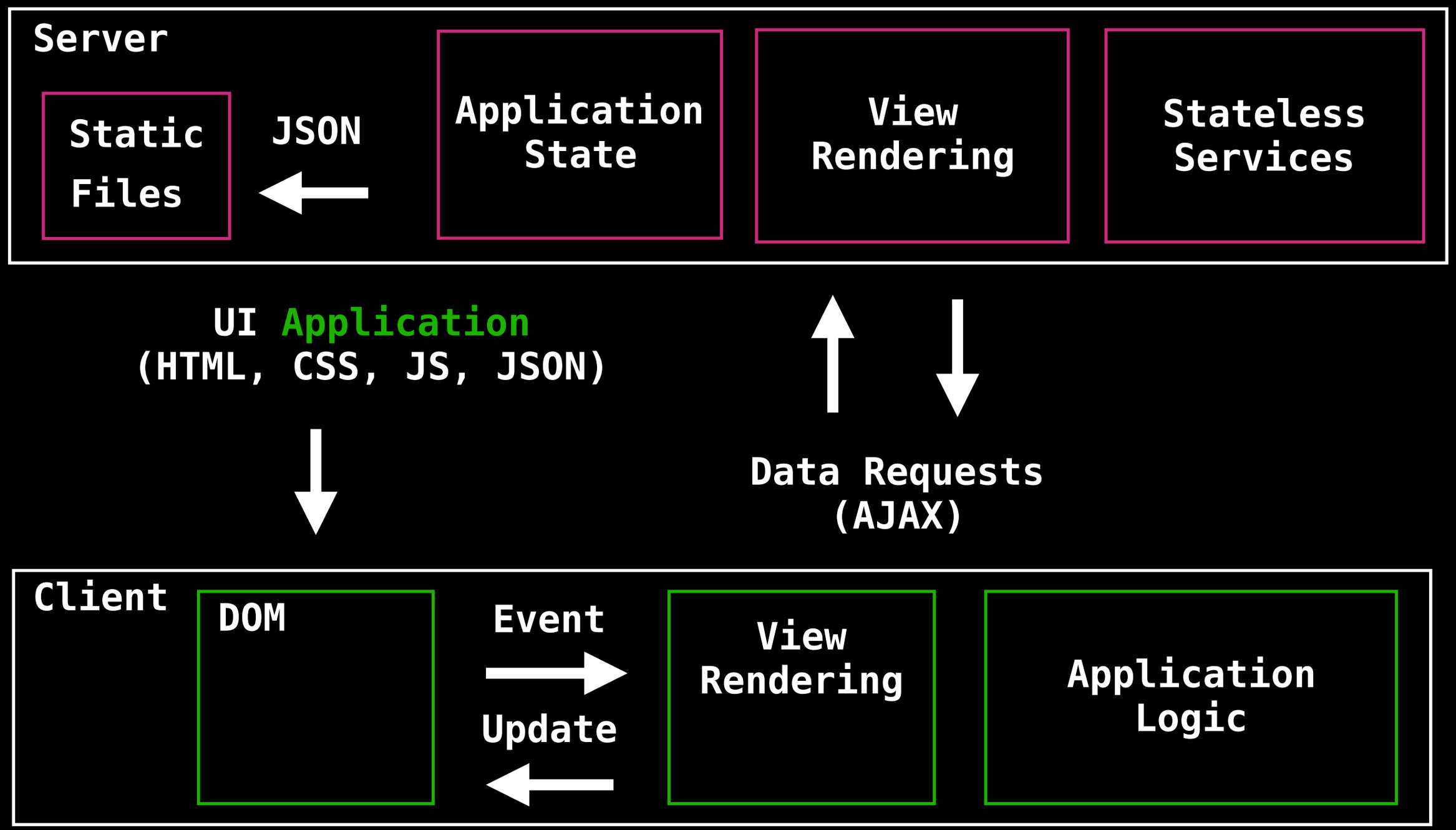
React Circa 2013



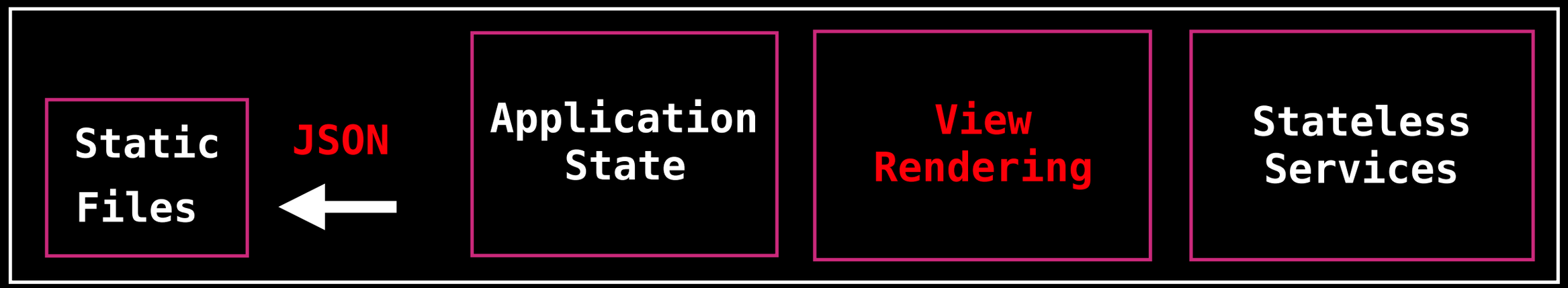
React Server Side 2013-Present



Isomorphic React

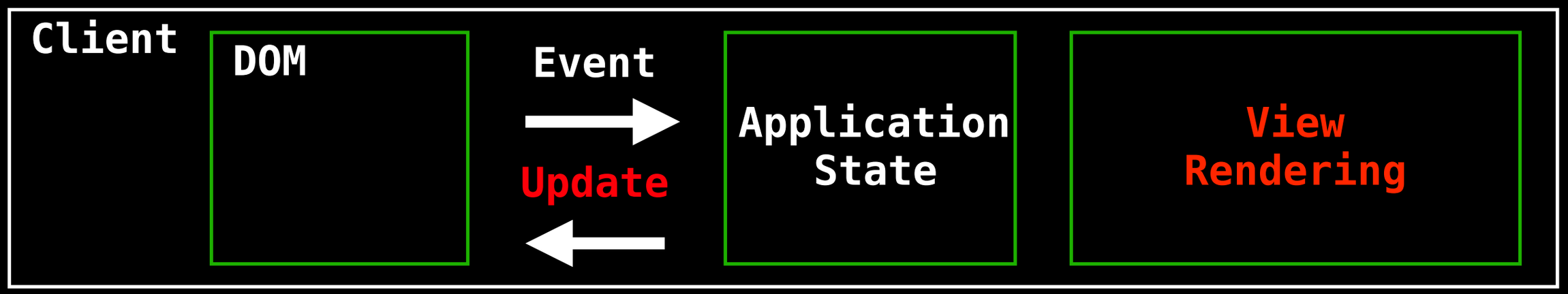


Evolving Threat Model



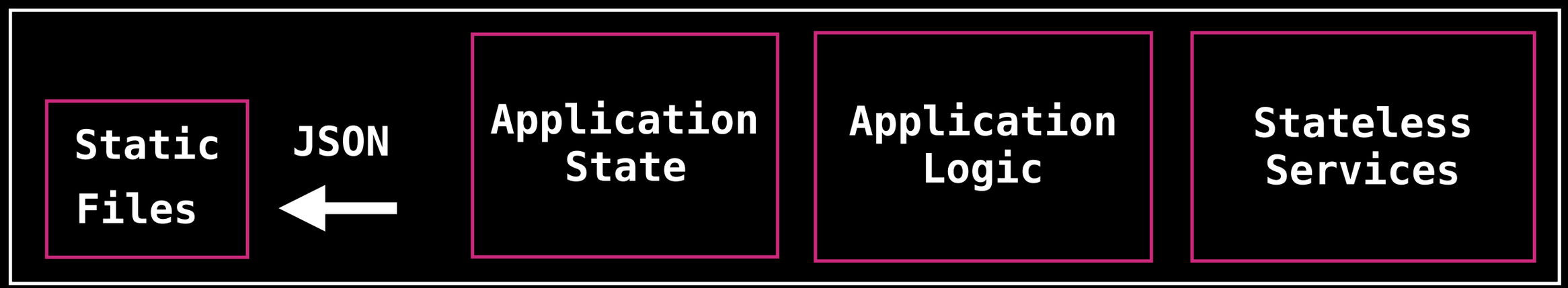
UI Application
(HTML, CSS, JS, JSON)

Data Requests
(AJAX)



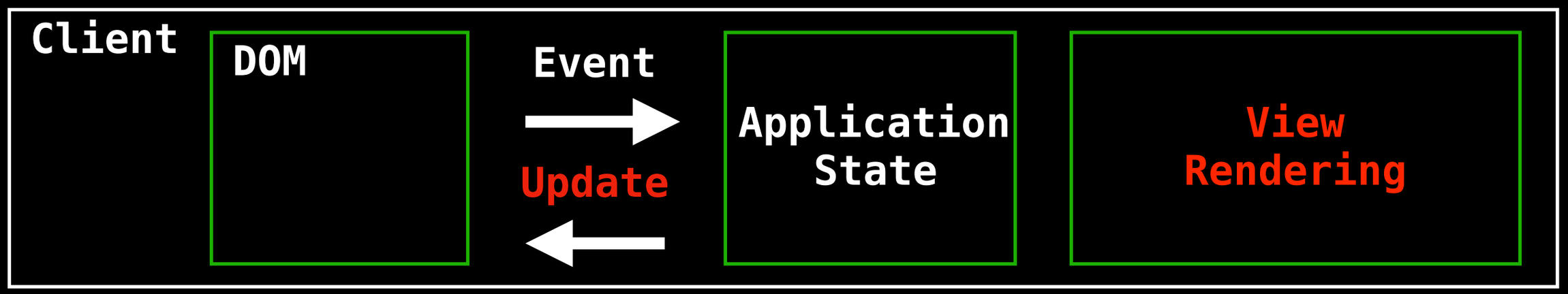
Attack Surface and Vulnerabilities

DOM Based Cross Site Scripting



UI Application
(HTML, CSS, JS, JSON)

Data Requests
(AJAX)



Builtin XSS Defense

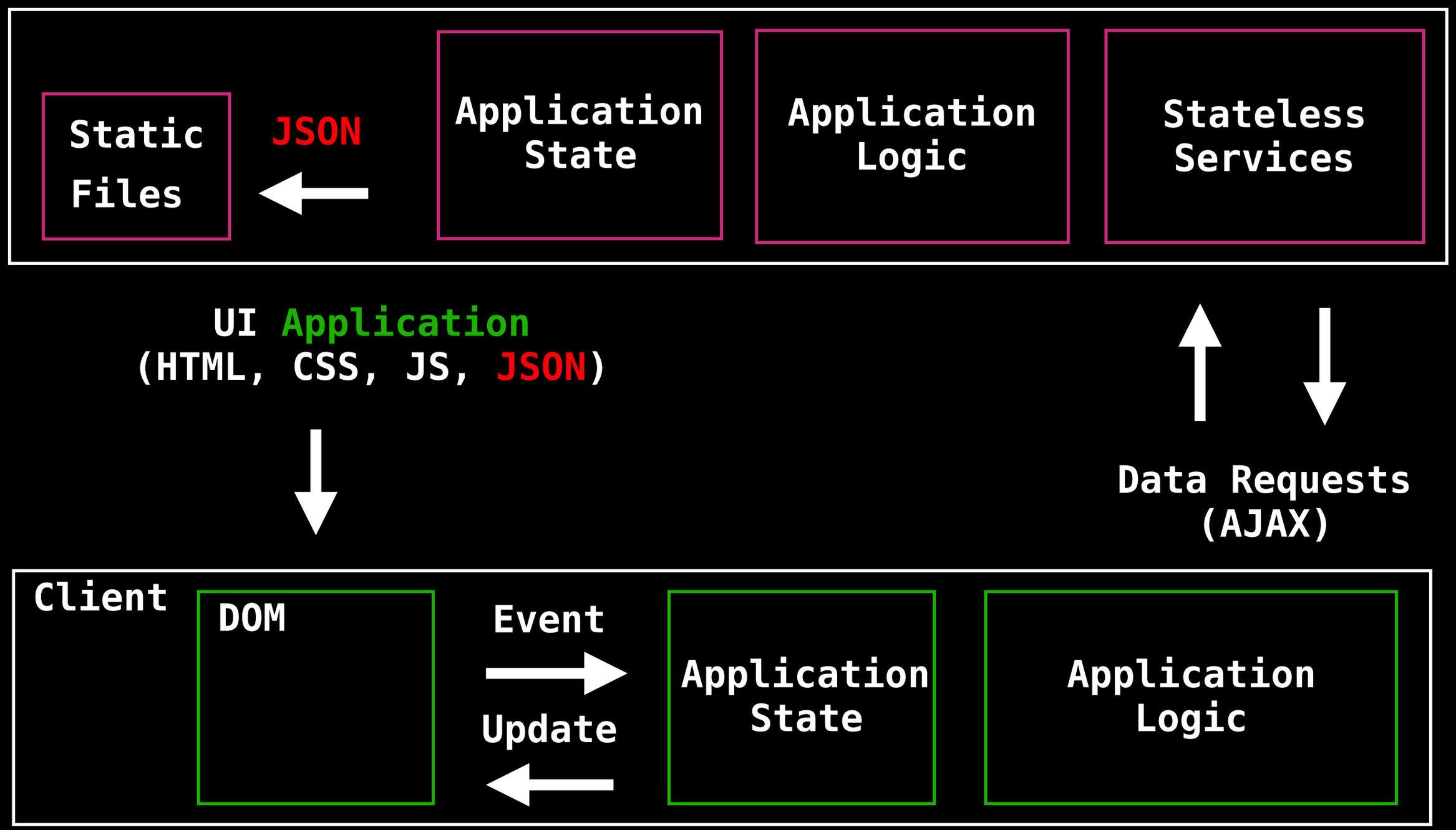
```
1 import React from "react";
2
3 function App() {
4   return <div className=“App">
            {"<img src=x onerror=alert(1) />”}
          </div>;
5 }
6
7 export default App;
```

“"

Bypassing XSS Protections

```
1 import React from "react";
2
3 function App() {
4   return (
5     <div className="App">
6       <a href={"javascript:alert(1)"}>Run a script.</a>
7     </div>
8   );
9 }
10
11 export default App;
```

Script injection via JSON State



Script injection via JSON State

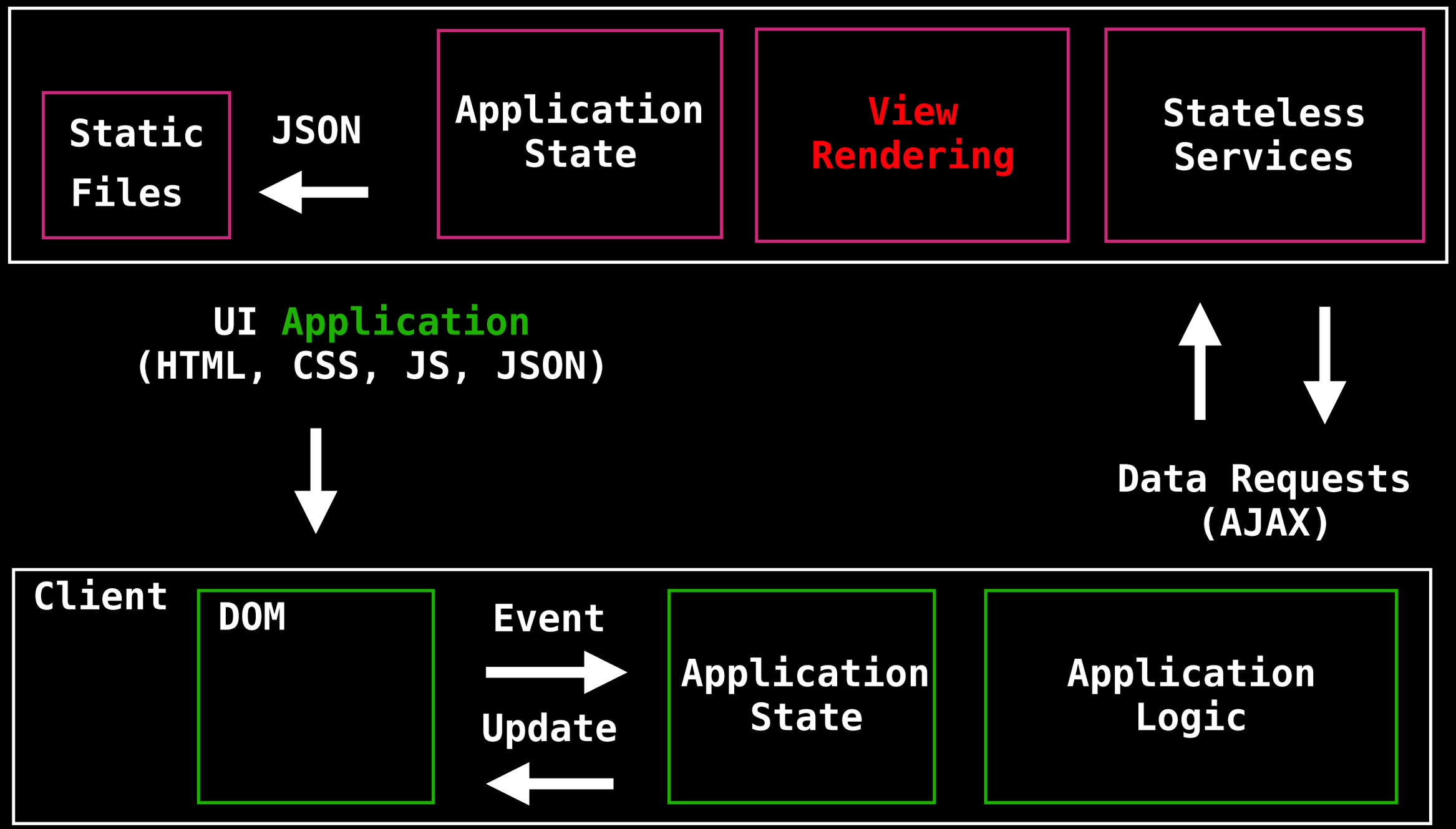
```
1  function renderFullPage(html, preloadedState) {
2    return `
3        <!doctype html>
4        <html>
5          <head>
6            <title>Redux Universal Example</title>
7          </head>
8          <body>
9            <div id="root">${html}</div>
10           <script>
11             window.__PRELOADED__ = JSON.stringify(preloadedState)
12           </script>
13           <script src="/static/bundle.js"></script>
14         </body>
15       </html>
16       `;
17 }
```

Source: Emelia Smith

JSON Injection Mitigation

```
1 const serialize = require("serialize-javascript");
2
…
11            <div id="root">${html}</div>
12            <script>
13              const state = serialize(preloadedState)
14                window.__PRELOADED_STATE__ = state
15              </script>
16              <script src="/static/bundle.js"></script>
17            </body>
18          </html>
19          `;
20 }
```

Server-side Injection Attack



Server Side Vulnerabilities?

`getInitialProps` receives a context object with the following properties:

- `pathname` - path section of URL
- `query` - query string section of URL parsed as an object
- `asPath` - `String` of the actual path (including the query) shows in the browser
- `req` - HTTP request object (server only)
- `res` - HTTP response object (server only)
- `err` - Error object if any error is encountered during the rendering

---

Server-side only method to ensure server-only code is never sent to the browser #5354

Closed **Jauny** opened this issue on Oct 1, 2018 · 16 comments

Searching Github for getInitialProps and SQL returns 5,054 results.

```
class Page extends React.Component {
  static async getInitialProps({ store, isServer, query }) {
    store.dispatch(Sql.favorite(query.type, "id", query.id));
    store.dispatch(Api.details(query.type, query.id));
    return {
      type: query.type,
      id: query.id,
      display: query.display
    };
  }
```

Escape Hatches

dangerouslySetInnerHTML

```
1 function createMarkup() {
2   return {   html: "<h1>Hello World!</h1>" };
3 }
4
5 function MyComponent() {
6   return <div dangerouslySetInnerHTML={createMarkup()} />;
7 }
```

Signal App

```
1 public renderText() {
2     const { i18n, text, attachments } = this.props;
3
4     if (text) {
5       return (
6         <div dangerouslySetInnerHTML={{ __html: text }} />
7       );
8     }
```

Refs Give Direct DOM Access

```
1 class AutoFocusTextInput extends React.Component {
2   constructor(props) {
3     super(props);
4     this.textInput = React.createRef();
5   }
6
7   componentDidMount() {
8     this.textInput.current.focusTextInput();
9   }
10
11   render() {
12     return (
13       <CustomTextInput ref={this.textInput} />
14     );
15   }
16 }
```

Refs Give Direct DOM Access

```
1   injectLinks = () => {
2     this.element.innerHTML = Parse.link(this.props.text)
3   }
4
5   saveRef = (element) => {
6     this.element = element
7   }
8
9   render() {
10
11     const { options, tagName, text, ...rest } = this.props
12
13     return <div ref={this.saveRef}>{this.props.text}</div>
14  }
```

findDOMNode Escape Hatch

```
ReactDOM.findDOMNode(component)
```

If this component has been mounted into the DOM, this returns the corresponding native browser DOM element. This method is useful for reading values out of the DOM, such as form field values and performing DOM measurements. **In most cases, you can attach a ref to the DOM node and avoid using `findDOMNode` at all.**

findDOMNode Escape Hatch

Showing 18,824 available code results ⓘ

Sort: **Best match** ▾

longer150721/react-todo-list – Main.js

JavaScript

Showing the top three matches Last indexed on Jul 11, 2018

```
18        ReactDOM.findDOMNode(this.refs['head']).innerHTML = '<span class="headline-
    main">Lanciy\'s </span>to do list';
19        ReactDOM.findDOMNode(this.refs['part1']).innerHTML = 'This is a stupid app';
20        ReactDOM.findDOMNode(this.refs['part2']).innerHTML = 'You can click the bottom
    left';
```

headzoo/react-moment – index.jsx

JSX

Showing the top four matches Last indexed on Apr 11

```
29          <Moment />
30        );
31        const expected = moment().toString();
32        expect(ReactDOM.findDOMNode(date).innerHTML).toEqual(expected);
...
37        expect(ReactDOM.findDOMNode(date).innerHTML).toEqual(DATE_OUTPUT);
38
39        date = TestUtils.renderIntoDocument(
```

devshawn/react-dayjs – index.test.jsx

JSX

Showing the top four matches Last indexed on Jul 14, 2018

```
12          const date = TestUtils.renderIntoDocument(<DayJS/>)
```

Escape Hatches in the Wild!

12% of the Top 100 React Component Libraries use dangrouslySetInnerHTML

61% of the Top 100 React Component Libraries use Refs

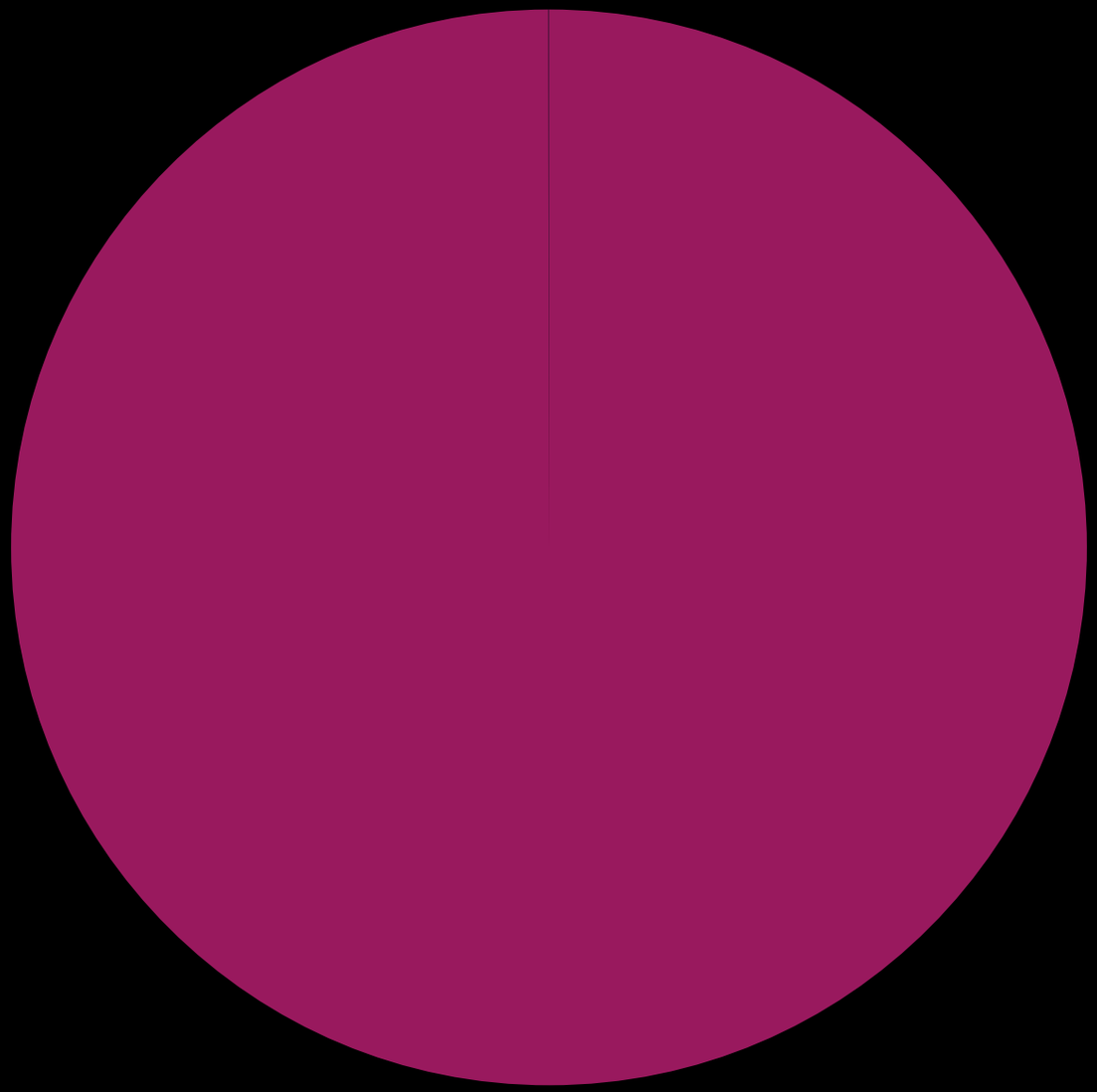
3rd Party React Library Security

I found 66% of the third-party React component library vulnerabilities in 2018.

I found two vulnerabilities.

Packages

Reported Vulnerabilities



react-svg

```
evalScripts -

Optional Run any script blocks found in the SVG.

One of 'always', 'once', or 'never'.

Defaults to ‘never'.
```

```
static defaultProps = {
   afterInjection: () => undefined,
   beforeInjection: () => undefined,
   evalScripts: ‘once',
   fallback: null,
   loading: null,
   renumerateIRIElements: true,
   wrapper: 'div'
 }
```

react-marked-markdown

```
<MarkdownPreview
  value="# Hey !"
  markedOptions={{

    sanitize: true,
    smartLists: true,
     smartypants: false
  }} />
```

```
var renderer = new _marked2.default.Renderer();
      renderer.link = function (href, title, text) {
        return '<a target="_blank" rel="noopener
noreferrer" href="' + href + '" title="' + title + '">' +
text + '</a>';
      };
```

Come help and get a CVE!

https://hackerone.com/nodejs-ecosystem

Shifting Left?

Secure Code Review

Focusing Manual Review

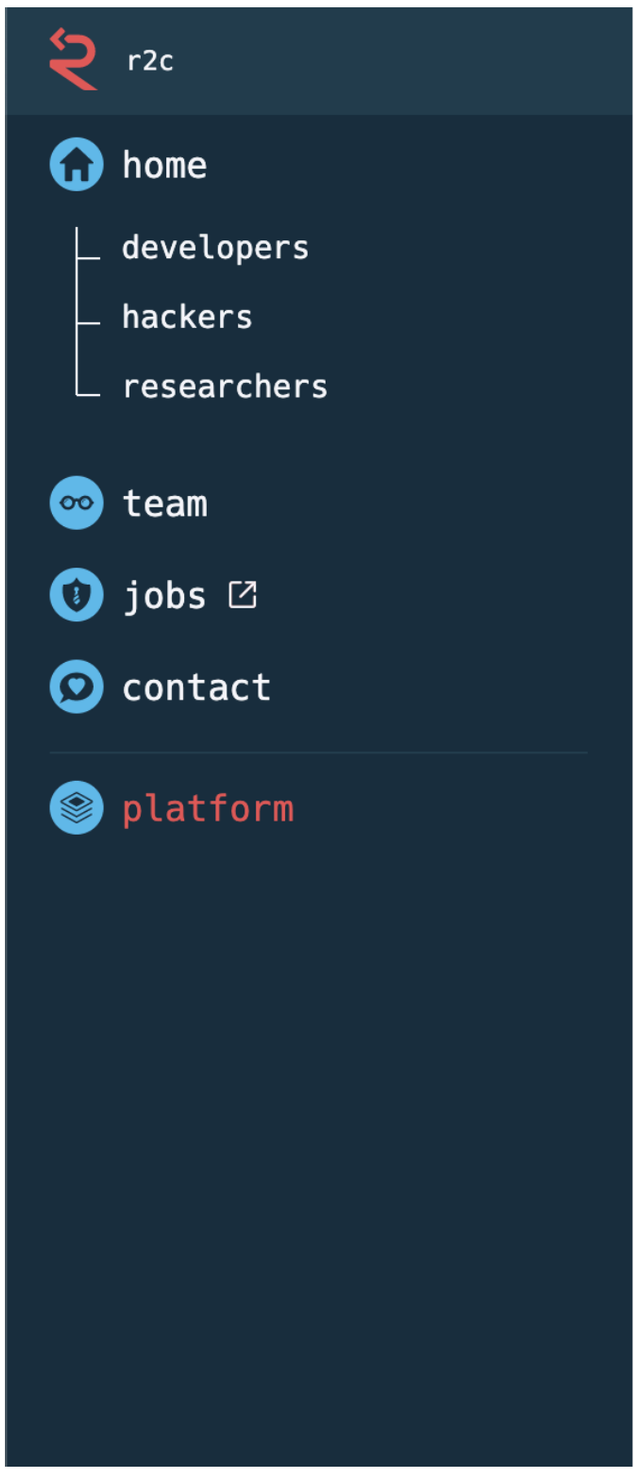
Server Side State Injection
Server Side Rendering
Escape Hatches
Markdown
SVG
Autolinkers

Automating AppSec of React Apps

Automating AppSec of React Apps

Lint for dangerouslySetInnerHTML
Lint for dangerous props
Lint for deserialized state
Lint for ref usage
Lint for findDomNode
Lint for createPortal
Lint for getInitialProps

Crowd Sourced Linter Development



Welcome to the platform!
Distributed program analysis for the open-source community.

r2c is a program analysis platform for security researchers, hackers, and developers. We give analysis a home on the web with tools that offer:

1. **scale**: run your analysis at large-scale across millions of open-source projects and their history

2. **re-use**: create analysis pipelines with reusable analyzers and shared results

3. **determinism**: easily define source code targets from GitHub, npm, PyPI, and others

4. **collaboration & feedback**: collaboratively triage, label, and disclose results

To learn more, join our waitlist or shoot us an email at hello@r2c.dev.

Happy hacking!
The team @ r2c

Register for our beta to start running your own analysis.

your email:

example@example.com

join waitlist

Already a user? Log in here and welcome back!

log in with your github account

CI/CD Tools

OWASP Dependency Check

npm audit

snyk

Tools Have Limitations

```
$ npm install react-dom@16.0.0

npm WARN deprecated react-dom@16.0.0:

This version of react-dom/server contains a minor vulnerability.
Please update react-dom to 16.0.1 or 16.4.2+.

Learn more: https://fb.me/cve-2018-6341

+ react-dom@16.0.0

updated 1 package and audited 654 packages in 3.65s

found 0 vulnerabilities
```

US Government Nation Vulnerability Database
Base Score: 6.1 MEDIUM
https://nvd.nist.gov/vuln/detail/CVE-2018-6341

Security Changes in 17.0.0

```
const isJavaScriptProtocol = /^[\u0000-
\u001F ]*j[\r\n\t]*a[\r\n\t]*v[\r\n\t]*a[
\r\n\t]*s[\r\n\t]*c[\r\n\t]*r[\r\n\t]*i[\
r\n\t]*p[\r\n\t]*t[\r\n\t]*\:/i;
```

Trusted Types

https://github.com/facebook/react/pull/16157

https://github.com/facebook/react/pull/16555

Thank you!