JSX

Liam McLennan

pluralsight
hardcore developer training

What is JSX?

- Supports xml-like syntax inline in JavaScript.
- Each element is transformed into a Javascript function call.
- Use http://facebook.github.io/react/jsx-compiler.html to experiment

Live editor

```
/** @jsx React.DOM */
var reactJsx = <Component>content</Component>;
```

```
/** @jsx React.DOM */
var reactJsx = Component(null, "content");
```

Live editor

```
/** @jsx React.DOM */
var reactJsx = <Component att1="a" att2="b">content</Component>;
```

```
/** @jsx React.DOM */
var reactJsx = Component( {att1:"a", att2:"b"}, "content");
```

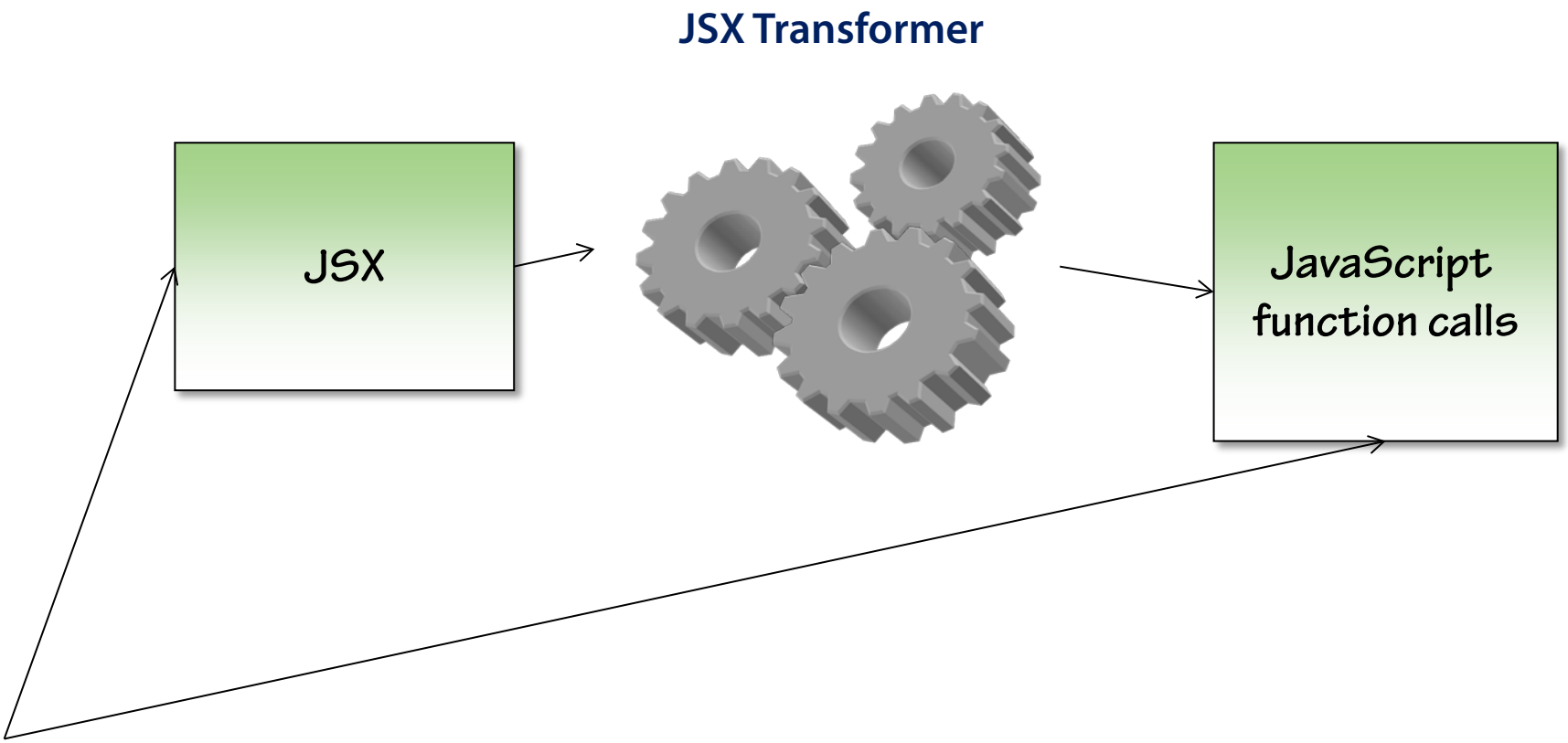
Live editor

```
/** @jsx React.DOM */
var reactJsx =  <Component att1="a" att2="b">
                    <Inner/>
                </Component>;
```

```
/** @jsx React.DOM */
var reactJsx =  Component( {att1:"a", att2:"b"},
                           Inner(null)
                           );
```

Not using JSX

- Skip directly to the output of the JSX transformer



JSX For Custom React Components

```
var Hello = React.createClass({…});
```

JSX

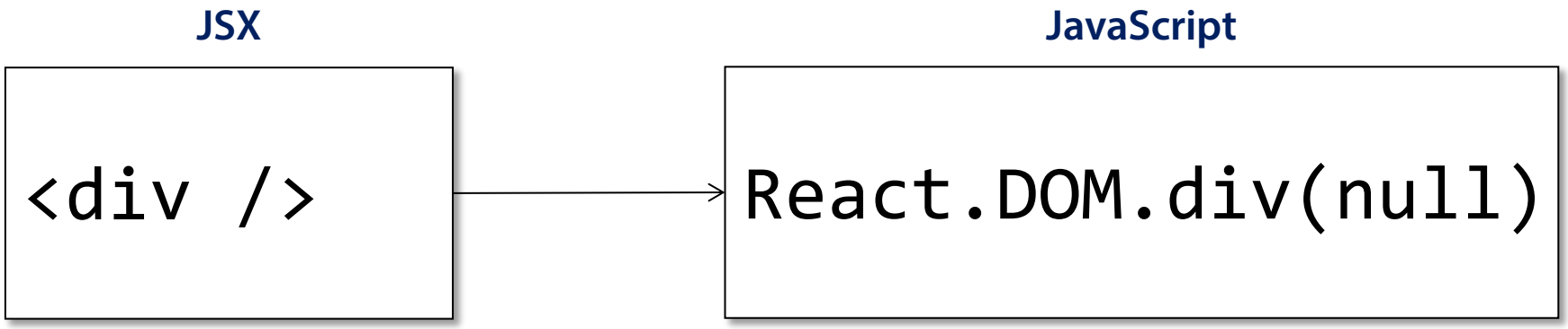
```
<Hello />
```


JavaScript

```
Hello(null)
```

JSX For HTML Components

- HTML elements are special-case react components.



Just-in-time JSX Transformer

- **Transform JSX to pure JavaScript immediately before execution**
- **Convenient for the developer**
- **Significant performance penalty**
- **Other disadvantages related to not being valid JavaScript**
  - Can't be minified
  - Can't be linted
  - Can't be debugged in the browser
  - Can't be formatted with a JavaScript syntax highlighter

```
<script src="JSXTransformer.js"></script>
```

```
<script type="text/jsx">…</script>
```

Pre-process Transformer

- Transform JSX to pure JavaScript as a build step.
- Much faster than the just-in-time transformer.
- Works with JavaScript tools
- Node module react-tools

```
> npm install -g react-tools
```

```
> jsx jsxdirectory/ compiledjs/
```

```
> jsx --watch jsxdirectory/ compiledjs/
```

JSX Attribute Expressions

- JSX attributes -> react component props

```
<Hello now={new Date().toString()} />
```

```
<Hello now="2013-12-07" />
```

Child Expressions and Elements

- Any valid JSX expression
- A JSX component may have more than one child expression or element

```
<Hello>
  <First />
  <Second />
</Hello>
```

- A component accesses its child elements via

```
this.props.children
```

Whitespace

- **Whitespace between {} expressions is not preserved**

```
{"Bob"} {"Alice“}        /* BobAlice */
```

```
{"Bob"} {" " + "Alice"}        /* Bob Alice */
```

```
{"Bob" + " " + "Alice"}        /* Bob Alice */
```

```
{"Bob"}{" "}{"Alice"}        /* Bob Alice */
```

HTML Attributes

- Attributes set on JSX DOM elements are only passed through to the rendered html if they are valid according to the HTML specification.
- Custom attributes are supported via data-

```
<div myCustomAttribute="foo" /> <!-- won‘t work -->
```

```
<div data-myAttribute="foo" /> <!-- will work -->
```

- Attributes cannot use JavaScript reserved words

```
<label htmlFor="name" className="big" />
```

HTML Attributes - style

- **Style is a special react attribute that expects a JavaScript object with camelCase properties.**

```
var Appender = React.createClass({
  render: function () {
    return <div style={{
      border:'1px solid #999',
      padding: '10px'}}>
      {this.props.children}</div>;
  }
});
```

HTML Attributes - style

- **Style is a special react attribute that expects a JavaScript object with camelCase properties.**

```
var Appender = React.createClass({
  render: function () {
    var s = {
      border:'1px solid #999',
      padding: '10px'
    };
    return <div style={s}>{this.props.children}</div>;
  }
});
```

Unescaping Content

- React escapes everything by default.
- Bypass using dangerouslySetInnerHTML.
- Expects an object with an __html property.

```
<div dangerouslySetInnerHTML={{__html="<p>foo</p>"}} />
```

The Author Quiz

Author Quiz

Select the book written by the author shown

- The Adventures of Huckleberry Finn
- Hamlet
- The Shining
- Romeo and Juliet

Summary

- **JSX is the default syntax and pre-processor for React.**
- **JSX is optional.**
- **Elements are translated to function calls.**
- **Attributes**
  - Cannot be JavaScript reserved words
  - The values can be strings or JavaScript expressions
- **Transformation can be**
  - Just-in-time
  - Precompiled by react-tools
- **Child expressions allow dynamic component nesting.**
- **Use dangerouslySetInnerHTML to unescape content .**