Intro to React

getting reactive….

The DOM sucks

- inconsistent
- its slow and hard to test
- its expensive

Thus….React

Javascript is fast

- assigning a value is fast
- assigning that value to a DOM element is slow

Thus virtual DOM

- javascript object representation of the DOM
- you makes Changes to the virtual DOM
- react compares the virtual DOM with the real
- does a diff
- only updates those parts

Components

- Reusable chunks of HTML
- one at a time, or 20 at a time
- break your UI down into pieces

```
var ParentComponent = React.createClass({
  render: function() {
    return(
      <h1>I'm a parent</h1>
    );
  }
});

  React.render(
    ParentComponent,
    domnode);
```

```
var ParentComponent = React.createClass({
  render: function() {
    return(<ChildComponent name=”rob” />);
  }
});
var ChildComponent = react.createClass({
  render: function() {
  fullName = this.props.name + ‘wilkinson’;
    return(<div>
                <GrandChildComponent name={fullName} />
                </div>);
  }
});
GrandChildComponent = react.createClass({
  render: function() {
    return(<div>{this.props.name}</div>);
  }
});
```

```
var ParentComponent = React.createClass({
  render: function() {
    return(<ChildComponent name=”rob” />);
  }
});
var ChildComponent = react.createClass({
  render: function() {
  fullName = this.props.name + ‘wilkinson’;
    return(<div>{fullName}</div>);
  }
});
```

Props are Immutable though…

State

- things that change
- its an object
- minimize state, its another thing that has to be watched
- change state by calling this.setState

```
var ChildComponent = React.createClass({
  wakeUp: function() {
    this.setState({ awake: true});
  },
  render: function() {
    return(<input type="checkbox" checked={this.state.awake} />);
  }
```

but wait…..doesn’t data flow up?!

handlers

we pass in handlers as properties that change the state of the parent!

Back to the DOM

- inconsistent
- its slow and hard to test
- its expensive

The Virtual DOM

- loaded in-memory!
- render() is called when something changes
- React does a diff against the real DOM and applys changes

Lifecycle methods

- componentDidMount
- componentWillMount

Why this is awesome

- this one way data flow keeps complexity under control.
- web components are the future and this is right there
- easy to debug self contained components
- the library is small, it doesn't force you to do anything crazy
- the future is super bright!