


Testing React

with Karma and Jasmine

Karma and Jasmine



Karma and Jasmine

A photograph of two fluffy, light-colored kittens with dark markings around their eyes, sitting in a light-colored wicker basket. The kitten on the right is looking directly at the camera, while the one on the left is looking slightly to the side. The background is a soft-focus outdoor scene with green foliage and small purple flowers.

Nikolaj Lundsgaard
Freelance software developer

Nikolaj@konfus.dk

doing React for almost 1 month

Actually, 2 presentations

Setting up a React dev environment

Testing components.

```
npm install --alot
```

npm install --alot

```
$ npm install --save-dev react
```

npm install --alot

```
$ npm install --save-dev react
```

```
$ npm install --save-dev browserify
```

```
$ npm install --save-dev reactify
```

```
$ npm install --save-dev watchify
```

npm install --alot

```
$ npm install --save-dev react
```

```
$ npm install --save-dev browserify
```

```
$ npm install --save-dev reactify
```

```
$ npm install --save-dev watchify
```

```
$ npm install --save-dev gulp
```

```
$ npm install --save-dev gulp-install
```

```
$ npm install --save-dev gulp-load-plugins
```

```
$ npm install --save-dev gulp-util
```

```
$ npm install --save-dev vinyl-buffer
```

```
$ npm install --save-dev vinyl-source-stream
```


gulpfile.js

```
var browserify = require('browserify');
var source = require('vinyl-source-stream');

var watchify = require('watchify');

gulp.task('browserify', function () {
  var bundler = watchify(browserify({
    entries: [config.input],
    cache: {},
    debug: true,
    packageCache: {},
    fullPaths: true
  }));

  function rebundle() {
    return bundler.bundle()
      .on('error', $.util.log.bind($.util, 'Browserify Error'))
      .pipe(source(config.output.file))

      .pipe(gulp.dest(config.output.folder + '/scripts'));
  }

  bundler.on('update', rebundle);

  return rebundle();
});
```

gulpfile.js

```
var browserify = require('browserify');
var source = require('vinyl-source-stream');
var buffer = require('vinyl-buffer');
var watchify = require('watchify');

gulp.task('browserify', function () {
  var bundler = watchify(browserify({
    entries: [config.input],
    cache: {},
    debug: false,
    packageCache: {},
    fullPaths: true
  }));

  function rebundle() {
    return bundler.bundle()
      .on('error', $.util.log.bind($.util, 'Browserify Error'))
      .pipe(source(config.output.file))
      .pipe(buffer())
      .pipe($.uglify())
      .pipe(gulp.dest(config.output.folder + '/scripts'));
  }

  bundler.on('update', rebundle);

  return rebundle();
});
```

package.json

```
{  
  "name": "ReactExample",  
  "version": "0.0.1",  
  "description": "React example",  
  
  "browserify": {  
    "transform": [  
      "reactify"  
    ]  
  },  
  ...  
}
```

gulpfile.js

```
var browserSync = require('browser-sync');
var reload = browserSync.reload;

gulp.task('serve', ['browserify'], function () {
  browserSync({
    notify: false,
    port: 9000,
    server: {
      baseDir: ['.dev']
    }
  });

  // watch for changes
  gulp.watch([
    '.dev/**/*.html',
    '.dev/styles/**/*.css',
    '.dev/scripts/**/*.js',
    '.dev/images/**/*.*'
  ]).on('change', reload);
});
```

Let's build some React

Let's build some React

☐ I have read the terms and conditions.

Accept

Let's build some React



I have read the terms and conditions.

Accept

React: Example1.jsx

```
var ExamplePage = React.createClass({
```

```
  render: function() {  
    return (  
      <div>
```

```
      </div>  
    );  
  }  
});
```


React: Example1.jsx

```
var ExamplePage = React.createClass({

  render: function() {
    return (
      <div>
        <div>
          <label>
            I have read the terms and conditions.
          </label>
        </div>

      </div>
    );
  }
});
```

React: Example1.jsx

```
var ExamplePage = React.createClass({

  render: function() {
    return (
      <div>
        <div>
          <label>
            <input type="checkbox" /> I have read the terms and conditions.
          </label>
        </div>

      </div>
    );
  }
});
```

React: Example1.jsx

```
var ExamplePage = React.createClass({

  handleChange: function(e) {

  },

  render: function() {
    return (
      <div>
        <div>
          <label>
            <input type="checkbox" onChange={this.handleChange} /> I have read the terms and conditions.
          </label>
        </div>

      </div>
    );
  }
});
```

React: Example1.jsx

```
var ExamplePage = React.createClass({

  handleChange: function(e) {
    this.setState({disabled: !e.target.checked});
  },

  render: function() {
    return (
      <div>
        <div>
          <label>
            <input type="checkbox" onChange={this.handleChange} /> I have read the terms and conditions.
          </label>
        </div>

      </div>
    );
  }
});
```

React: Example1.jsx

```
var ExamplePage = React.createClass({

  getInitialState: function() {
    return { disabled: true };
  },

  handleChange: function(e) {
    this.setState({disabled: !e.target.checked});
  },

  render: function() {
    return (
      <div>
        <div>
          <label>
            <input type="checkbox" onChange={this.handleChange} /> I have read the terms and conditions.
          </label>
        </div>

      </div>
    );
  }
});
```

React: Example1.jsx

```
var ExamplePage = React.createClass({

  getInitialState: function() {
    return { disabled: true };
  },

  handleChange: function(e) {
    this.setState({disabled: !e.target.checked});
  },

  render: function() {
    return (
      <div>
        <div>
          <label>
            <input type="checkbox" onChange={this.handleChange} /> I have read the terms and conditions.
          </label>
        </div>
        <div>
          <button> Accept </button>
        </div>
      </div>
    );
  }
});
```

React: Example1.jsx

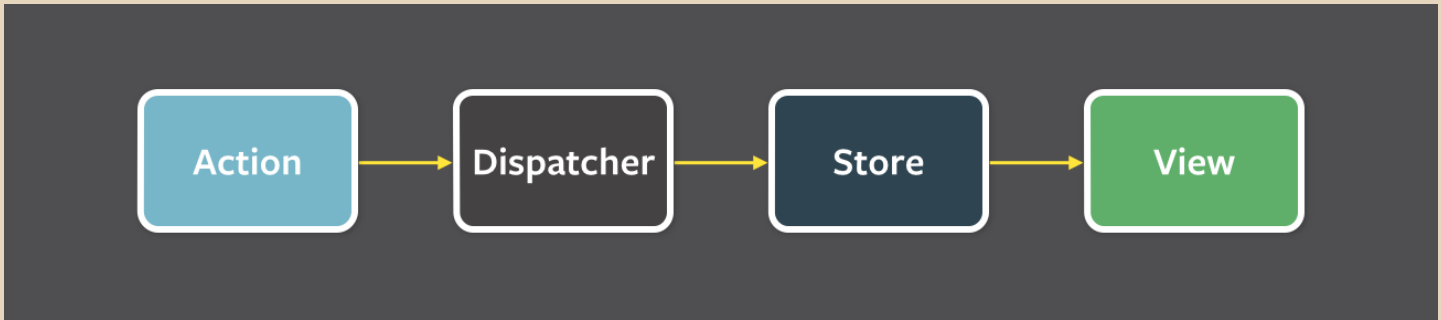
```
var ExamplePage = React.createClass({

  getInitialState: function() {
    return { disabled: true };
  },

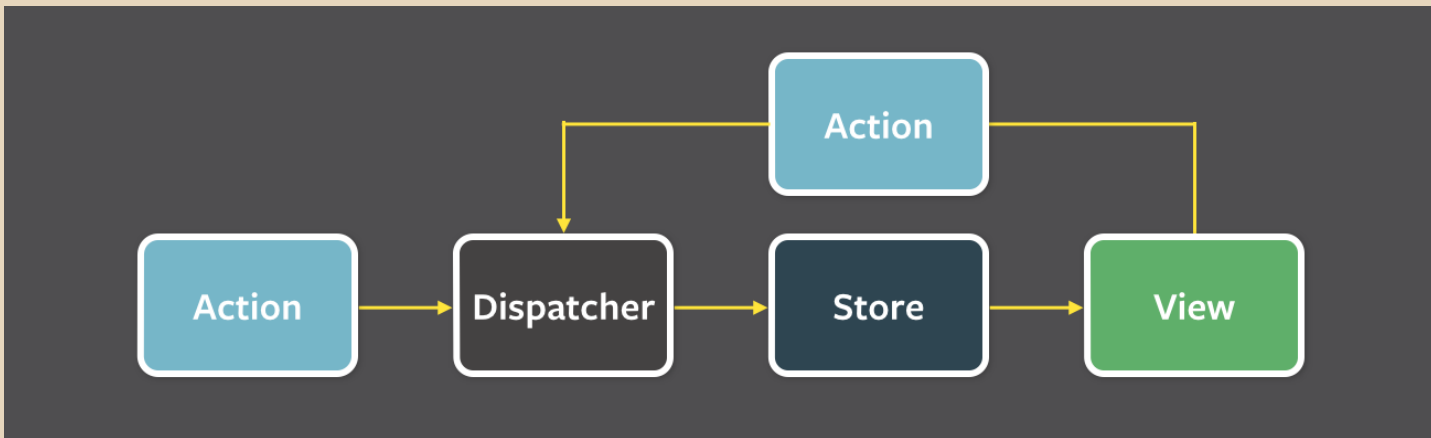
  handleChange: function(e) {
    this.setState({disabled: !e.target.checked});
  },

  render: function() {
    return (
      <div>
        <div>
          <label>
            <input type="checkbox" onChange={this.handleChange} /> I have read the terms and conditions.
          </label>
        </div>
        <div>
          <button className={this.state.disabled ? 'btn disabled' : 'btn'}> Accept </button>
        </div>
      </div>
    );
  }
});
```

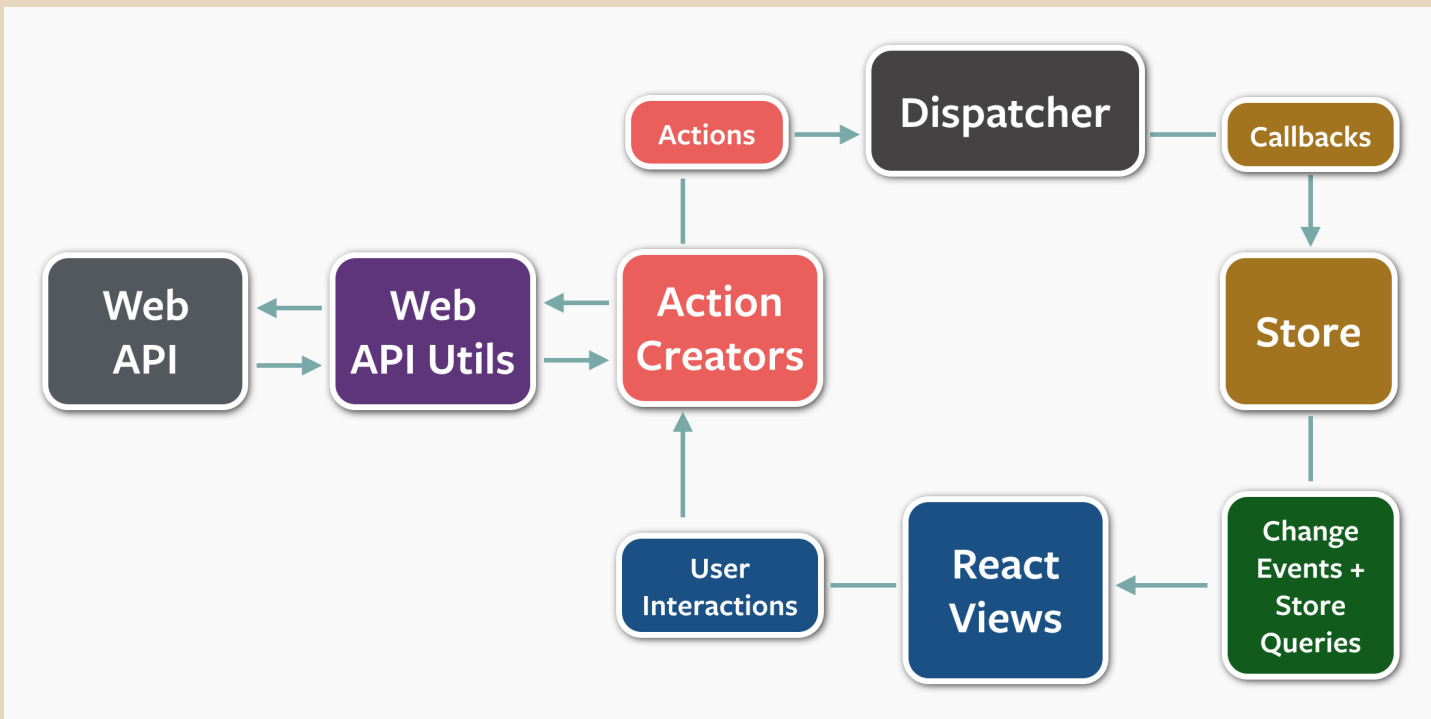
Flux



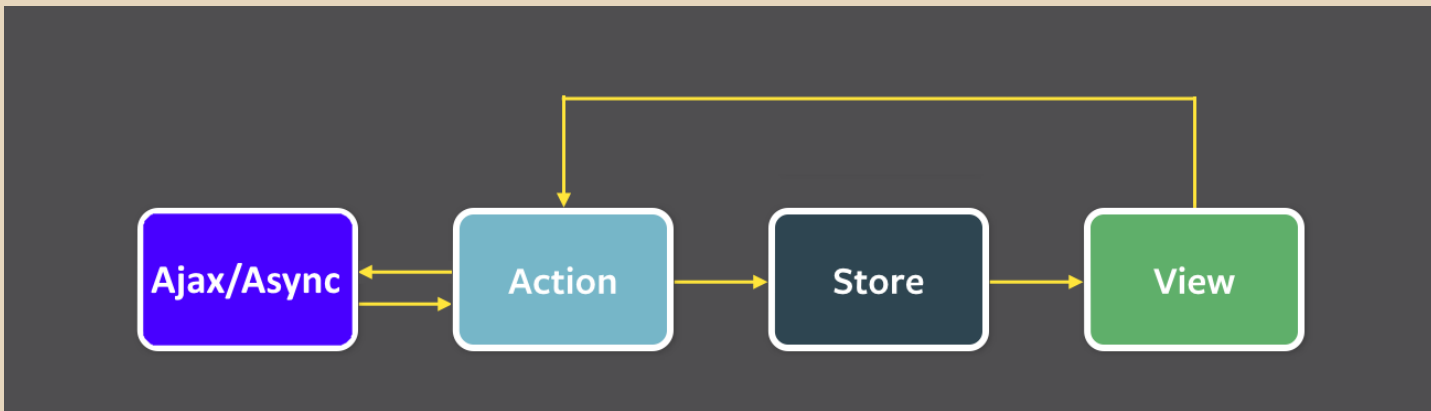
Flux



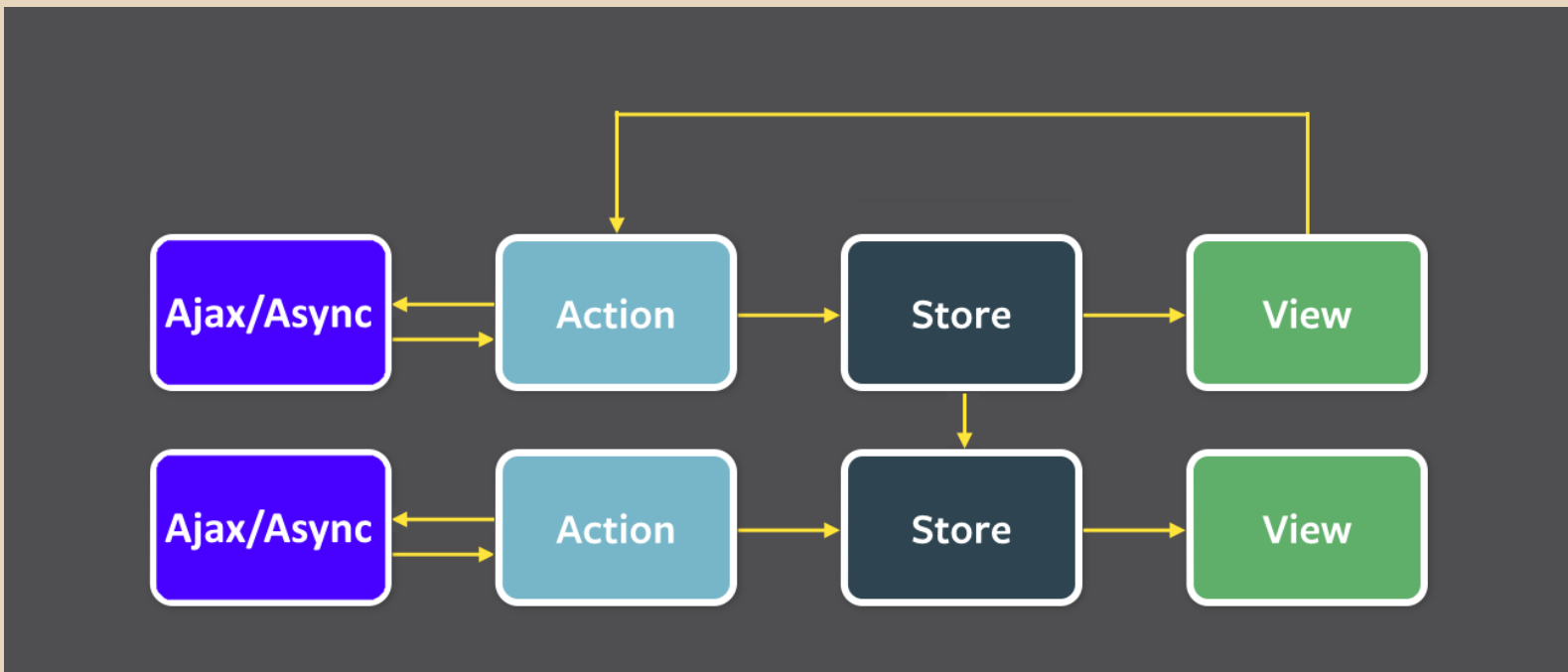
Flux



Reflux



Reflux



React: Example2.jsx

```
var ExamplePage = React.createClass({  
  render: function() {  
    return (  
      <div>  
  
      </div>  
    );  
  }  
});
```

React: Example2.jsx

```
var ExamplePage = React.createClass({  
  render: function() {  
    return (  
      <div>  
        <TermsAndConditionsElement />  
        <ButtonElement />  
      </div>  
    );  
  }  
});
```

React: Example2.jsx

```
var TermsAndConditionsElement = require('../components/TermsAndConditionsElement.jsx');
var ButtonElement = require('../components/ButtonElement.jsx');

var ExamplePage = React.createClass({

  render: function() {
    return (
      <div>
        <TermsAndConditionsElement />
        <ButtonElement />
      </div>
    );
  }
});
```

TermsAndConditionsElement.jsx

```
var TermsAndConditionsElement = React.createClass({  
  render: function() {  
    return (  
      <div>  
  
      </div>  
    );  
  },  
  
});
```


TermsAndConditionsElement.jsx

```
var TermsAndConditionsElement = React.createClass({  
  render: function() {  
    return (  
      <div>  
        <label>  
          <input type="checkbox" /> I have read the terms and conditions.  
        </label>  
      </div>  
    );  
  },  
});
```

TermsAndConditionsElement.jsx

```
var TermsAndConditionsElement = React.createClass({  
  render: function() {  
    return (  
      <div>  
        <label>  
          <input type="checkbox" onChange={this.handleAgreed} /> I have read the terms and conditions.  
        </label>  
      </div>  
    );  
  },  
  handleAgreed: function(e) {  
  }  
});
```

TermsAndConditionsElement.jsx

```
var actions = require('../flux/actions/TermsAndConditionsActions');

var TermsAndConditionsElement = React.createClass({

  render: function() {
    return (
      <div>
        <label>
          <input type="checkbox" onChange={this.handleAgreed} /> I have read the terms and conditions.
        </label>
      </div>
    );
  },

  handleAgreed: function(e) {
    actions.agreed(e.target.checked);
  }
});
```

TermsAndConditionsActions.js

```
var reflux = require('reflux');  
  
var TermsAndConditionsActions = reflux.createActions([  
  'agreed'  
]);
```

TermsAndConditionsStore.js

```
var reflux = require('reflux');

var TermsAndConditionsStore = reflux.createStore({

});
```

TermsAndConditionsStore.js

```
var reflux = require('reflux');

var TermsAndConditionsActions = require('../actions/TermsAndConditionsActions');

var TermsAndConditionsStore = reflux.createStore({

});
```

TermsAndConditionsStore.js

```
var reflux = require('reflux');

var TermsAndConditionsActions = require('../actions/TermsAndConditionsActions');

var TermsAndConditionsStore = reflux.createStore({
  init: function() {
    this.listenTo(TermsAndConditionsActions.agreed, ...);
  },

});
```

TermsAndConditionsStore.js

```
var reflux = require('reflux');

var TermsAndConditionsActions = require('../actions/TermsAndConditionsActions');

var TermsAndConditionsStore = reflux.createStore({
  init: function() {
    this.listenTo(TermsAndConditionsActions.agreed, 'onAgreedChecked');
  },

  onAgreedChecked: function(checked) {

  }
});
```


TermsAndConditionsStore.js

```
var reflux = require('reflux');

var TermsAndConditionsActions = require('../actions/TermsAndConditionsActions');

var TermsAndConditionsStore = reflux.createStore({
  init: function() {
    this.listenTo(TermsAndConditionsActions.agreed, 'onAgreedChecked');
  },

  onAgreedChecked: function(checked) {
    this.trigger({agreed: checked});
  }
});
```

ButtonElement.jsx

```
var ButtonElement = React.createClass({

  render: function() {
    return (
      <div>
        <button> Accept </button>
      </div>
    );
  }
});
```

ButtonElement.jsx

```
var ButtonElement = React.createClass({
  getInitialState: function() {
    return {disabled: true};
  },

  render: function() {
    return (
      <div>
        <button className={this.state.disabled ? 'btn disabled' : 'btn'}> Accept </button>
      </div>
    );
  }
});
```

ButtonElement.jsx

```
var TermsAndConditionsStore = require('../flux/stores/TermsAndConditionsStore');

var ButtonElement = React.createClass({
  getInitialState: function() {
    return {disabled: true};
  },

  render: function() {
    return (
      <div>
        <button className={this.state.disabled ? 'btn disabled' : 'btn'}> Accept </button>
      </div>
    );
  }
});
```

ButtonElement.jsx

```
var TermsAndConditionsStore = require('../flux/stores/TermsAndConditionsStore');

var ButtonElement = React.createClass({
  getInitialState: function() {
    return {disabled: true};
  },

  componentWillMount: function() {

  },

  render: function() {
    return (
      <div>
        <button className={this.state.disabled ? 'btn disabled' : 'btn'}> Accept </button>
      </div>
    );
  }
});
```

ButtonElement.jsx

```
var TermsAndConditionsStore = require('../flux/stores/TermsAndConditionsStore');

var ButtonElement = React.createClass({
  getInitialState: function() {
    return {disabled: true};
  },

  componentWillMount: function() {
    TermsAndConditionsStore.listen(...);
  },

  render: function() {
    return (
      <div>
        <button className={this.state.disabled ? 'btn disabled' : 'btn'}> Accept </button>
      </div>
    );
  }
});
```

ButtonElement.jsx

```
var TermsAndConditionsStore = require('../flux/stores/TermsAndConditionsStore');

var ButtonElement = React.createClass({
  getInitialState: function() {
    return {disabled: true};
  },

  componentWillMount: function() {
    TermsAndConditionsStore.listen(this.onTermsAndConditionsChecked);
  },

  onTermsAndConditionsChecked: function(data) {

  },

  render: function() {
    return (
      <div>
        <button className={this.state.disabled ? 'btn disabled' : 'btn'}> Accept </button>
      </div>
    );
  }
});
```

ButtonElement.jsx

```
var TermsAndConditionsStore = require('../flux/stores/TermsAndConditionsStore');

var ButtonElement = React.createClass({
  getInitialState: function() {
    return {disabled: true};
  },

  componentWillMount: function() {
    TermsAndConditionsStore.listen(this.onTermsAndConditionsChecked);
  },

  onTermsAndConditionsChecked: function(data) {
    this.setState({disabled: !data.agreed});
  },

  render: function() {
    return (
      <div>
        <button className={this.state.disabled ? 'btn disabled' : 'btn'}> Accept </button>
      </div>
    );
  }
});
```


So, what about testing?



So, what about testing?

```
$ npm install --save-dev karma  
$ npm install --save-dev karma-browserify  
$ npm install --save-dev karma-chrome-launcher  
$ npm install --save-dev karma-coverage  
$ npm install --save-dev karma-jasmine
```

So, what about testing?

```
$ npm install --save-dev karma
$ npm install --save-dev karma-browserify
$ npm install --save-dev karma-chrome-launcher
$ npm install --save-dev karma-coverage
$ npm install --save-dev karma-jasmine

$ npm install --save-dev proxyquireify
```

So, what about testing?

```
$ npm install --save-dev karma
$ npm install --save-dev karma-browserify
$ npm install --save-dev karma-chrome-launcher
$ npm install --save-dev karma-coverage
$ npm install --save-dev karma-jasmine

$ npm install --save-dev proxyquireify

$ npm install -g karma-cli
```

So, what about testing?

```
$ npm install --save-dev karma
$ npm install --save-dev karma-browserify
$ npm install --save-dev karma-chrome-launcher
$ npm install --save-dev karma-coverage
$ npm install --save-dev karma-jasmine

$ npm install --save-dev proxyquireify

$ npm install -g karma-cli

$ karma init
```

So, what about testing?

```
$ npm install --save-dev karma
$ npm install --save-dev karma-browserify
$ npm install --save-dev karma-chrome-launcher
$ npm install --save-dev karma-coverage
$ npm install --save-dev karma-jasmine

$ npm install --save-dev proxyquireify

$ npm install -g karma-cli

$ karma init

$ karma start
```

karma.conf.js

```
module.exports = function(config) {
  config.set({
    // frameworks to use
    frameworks: ['browserify', 'jasmine'],

    // list of files / patterns to load in the browser
    files: [
      'src/**/*.spec.js'
    ],

    // preprocess matching files before serving them to the browser
    preprocessors: {
      'src/**/*.spec.js': [ 'browserify' ]
    },

    browserify: {
      transform: [ 'reactify' ],
      plugin: [ 'proxyquireify/plugin' ]
    },

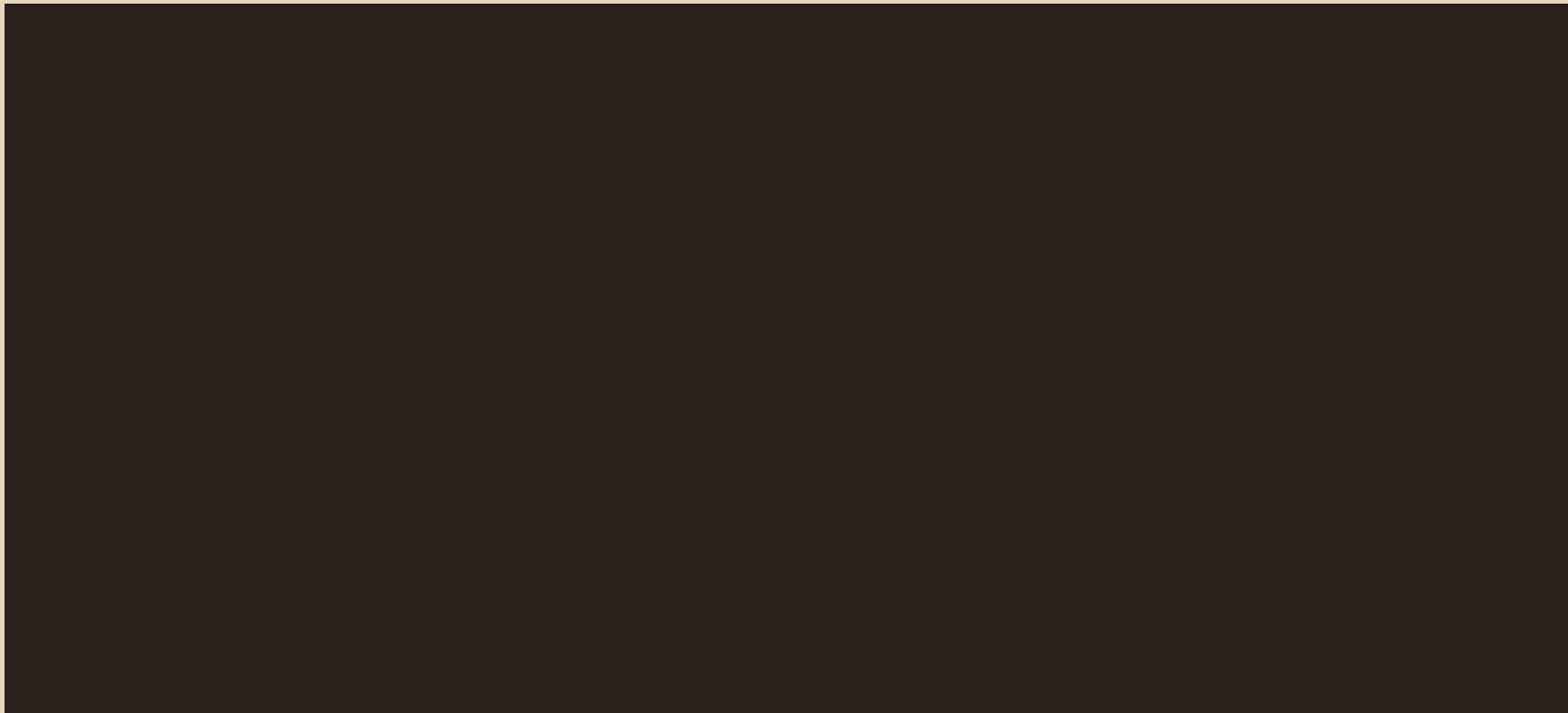
    // test results reporter to use
    reporters: ['progress'],

    // start these browsers
    browsers: ['Chrome']
  });
};
```

Folder structure

```
$ tree src
|__ components
| |__ ButtonElement.jsx
| |__ ButtonElement.spec.js
| |__ TermsAndConditionsElement.jsx
| |__ TermsAndConditionsElement.spec.js
|__ flux
| |__ actions
| | |__ TermsAndConditionsActions.js
| |__ stores
| | |__ TermsAndConditionsStore.js
| | |__ TermsAndConditionsStore.spec.js
|__ package.json
|__ pages
| |__ ExamplePage1.jsx
| |__ ExamplePage1.spec.js
| |__ ExamplePage2.jsx
| |__ ExamplePage2.spec.js
|__ shared
| |__ react
| | |__ TestUtils.js
```


ExamplePage1.spec.js



ExamplePage1.jsx

```
var ExamplePage = React.createClass({

  getInitialState: function() {
    return { disabled: true };
  },

  handleChange: function(e) {
    this.setState({disabled: !e.target.checked});
  },

  render: function() {
    return (
      <div>
        <div>
          <label>
            <input type="checkbox" onChange={this.handleChange} /> I have read the terms and conditions.
          </label>
        </div>
        <div>
          <button className={this.state.disabled ? 'btn disabled' : 'btn'}> Accept </button>
        </div>
      </div>
    );
  }
});
```

ExamplePage1.spec.js

```
var ExamplePage1 = require('./ExamplePage1.jsx');  
var TestUtils = require('../shared/react/TestUtils');  
  
describe('ExamplePage1', function() {
```

```
});
```

ExamplePage1.spec.js

```
var ExamplePage1 = require('./ExamplePage1.jsx');
var TestUtils = require('../shared/react/TestUtils');

describe('ExamplePage1', function() {
  var x;
  beforeEach(function() {
    x = TestUtils.testComponent(ExamplePage1);
  });

  afterEach(function() {
    x.remove();
  });

});
```

ExamplePage1.spec.js

```
var ExamplePage1 = require('./ExamplePage1.jsx');
var TestUtils = require('../shared/react/TestUtils');

describe('ExamplePage1', function() {
  it('should have a checkbox', function() {
    expect(x.find('input[type="checkbox"]').length).toBe(1);
  });

});
```

ExamplePage1.spec.js

```
var ExamplePage1 = require('./ExamplePage1.jsx');
var TestUtils = require('../shared/react/TestUtils');

describe('ExamplePage1', function() {
  it('should have a checkbox', function() {
    expect(x.find('input[type="checkbox"]').length).toBe(1);
  });

  it('should have a button', function() {
    expect(x.find('button').length).toBe(1);
  });
});
```

ExamplePage1.spec.js

```
var ExamplePage1 = require('./ExamplePage1.jsx');
var TestUtils = require('../shared/react/TestUtils');

describe('ExamplePage1', function() {
  it('should have a checkbox', function() {
    expect(x.find('input[type="checkbox"]').length).toBe(1);
  });

  it('should have a button', function() {
    expect(x.find('button').length).toBe(1);
  });

  it('should have a disabled button, initially', function() {
    expect(x.find('button').is(':disabled')).toBe(true);
  });

});
```

ExamplePage1.spec.js

```
var ExamplePage1 = require('./ExamplePage1.jsx');
var TestUtils = require('../shared/react/TestUtils');

describe('ExamplePage1', function() {
  it('should have a checkbox', function() {
    expect(x.find('input[type="checkbox"]').length).toBe(1);
  });

  it('should have a button', function() {
    expect(x.find('button').length).toBe(1);
  });

  it('should have a disabled button, initially', function() {
    expect(x.find('button').is(':disabled')).toBe(true);
  });

  describe('when the checkbox is checked', function() {
    it('should enable the button', function() {
      expect(x.find('button:disabled').length).toBe(0);
    });
  });
});
```


ExamplePage2.jsx

```
var TermsAndConditionsElement = require('../components/TermsAndConditionsElement.jsx');
var ButtonElement = require('../components/ButtonElement.jsx');

var ExamplePage = React.createClass({

  render: function() {
    return (
      <div>
        <TermsAndConditionsElement />
        <ButtonElement />
      </div>
    );
  }
});
```

ExamplePage2.spec.js

```
var TestUtils = require('../shared/react/TestUtils');

describe('ExamplePage2', function() {
  var x;
  beforeEach(function() {

    x = TestUtils.testComponent(ExamplePage2);
  });

  afterEach(function() {
    x.remove();
  });

});
```

ExamplePage2.spec.js

```
var TestUtils = require('../shared/react/TestUtils');
var proxyquire = require('proxyquireify')(require);

describe('ExamplePage2', function() {
  var x;
  beforeEach(function() {
    var stubs = {
      '../components/TermsAndConditionsElement.jsx': TestUtils.stubComponent('div', {'data-test-id': 'TermsAndAgreementEle..'}),
      '../components/ButtonElement.jsx': TestUtils.stubComponent('div', {'data-test-id': 'ButtonElement'}),
    };
    var ExamplePage2 = proxyquire('./ExamplePage2.jsx', stubs);
    x = TestUtils.testComponent(ExamplePage2);
  });

  afterEach(function() {
    x.remove();
  });

});
```

ExamplePage2.spec.js

```
var TestUtils = require('../shared/react/TestUtils');
var proxyquire = require('proxyquireify')(require);

describe('ExamplePage2', function() {
  var x;
  beforeEach(function() {
    var stubs = {
      '../components/TermsAndConditionsElement.jsx': TestUtils.stubComponent('div', {'data-test-id': 'TermsAndAgreementEle..'}),
      '../components/ButtonElement.jsx': TestUtils.stubComponent('div', {'data-test-id': 'ButtonElement'}),
    };
    var ExamplePage2 = proxyquire('../ExamplePage2.jsx', stubs);
    x = TestUtils.testComponent(ExamplePage2);
  });

  afterEach(function() {
    x.remove();
  });

  it('should render TermsAndAgreementElement', function() {
    expect(x.find('[data-test-id="TermsAndAgreementElement"]').length).toBe(1);
  });
});
```

ExamplePage2.spec.js

```
var TestUtils = require('../shared/react/TestUtils');
var proxyquire = require('proxyquireify')(require);

describe('ExamplePage2', function() {
  var x;
  beforeEach(function() {
    var stubs = {
      '../components/TermsAndConditionsElement.jsx': TestUtils.stubComponent('div', {'data-test-id': 'TermsAndAgreementEle..'}),
      '../components/ButtonElement.jsx': TestUtils.stubComponent('div', {'data-test-id': 'ButtonElement'}),
    };
    var ExamplePage2 = proxyquire('./ExamplePage2.jsx', stubs);
    x = TestUtils.testComponent(ExamplePage2);
  });

  afterEach(function() {
    x.remove();
  });

  it('should render TermsAndAgreementElement', function() {
    expect(x.find('[data-test-id="TermsAndAgreementElement"]').length).toBe(1);
  });

  it('should render ButtonElement', function() {
    expect(x.find('[data-test-id="ButtonElement"]').length).toBe(1);
  });
});
```

ButtonElement.spec.js

```
var TestUtils = require('../shared/react/TestUtils');
var ButtonElement = require('./ButtonElement.jsx');
var TermsAndConditionsStore = require('../flux/stores/TermsAndConditionsStore');

describe('ButtonElement', function() {
  it('should have a button', function() {
    expect(x.find('button').length).toBe(1);
  });

  it('should have a disabled button, initially', function() {
    expect(x.find('button:disabled').length).toBe(1);
  });

  describe('when the store signals true', function() {
    it('should enable the button if store signals agreed', function() {
      expect(x.find('button:enabled').length).toBe(1);
    });
  });
});
```

ButtonElement.spec.js

```
var TestUtils = require('../shared/react/TestUtils');
var ButtonElement = require('./ButtonElement.jsx');
var TermsAndConditionsStore = require('../flux/stores/TermsAndConditionsStore');

describe('ButtonElement', function() {
  it('should have a button', function() {
    expect(x.find('button').length).toBe(1);
  });

  it('should have a disabled button, initially', function() {
    expect(x.find('button:disabled').length).toBe(1);
  });

  describe('when the store signals true', function() {
    it('should enable the button if store signals agreed', function() {
      expect(x.find('button:enabled').length).toBe(1);
    });
  });

  describe('when the store signals false', function() {
    beforeEach(function() {
      TermsAndConditionsStore.trigger({agreed: false});
    });

    it('should disable the button', function() {
      expect(x.find('button:disabled').length).toBe(1);
    });
  });
});
```

ButtonElement.spec.js

```
var TestUtils = require('../shared/react/TestUtils');
var ButtonElement = require('./ButtonElement.jsx');
var TermsAndConditionsStore = require('../flux/stores/TermsAndConditionsStore');

describe('ButtonElement', function() {

  describe('when the store signals false', function() {
    beforeEach(function() {
      TermsAndConditionsStore.trigger({agreed: false});
    });

    it('should disable the button', function() {
      expect(x.find('button:disabled').length).toBe(1);
    });
  });
});
```


TermsAndConditionsElement.spec.js

```
var TestUtils = require('../shared/react/TestUtils');
var TermsAndConditionsElement = require('./TermsAndConditionsElement.jsx');
var TermsAndConditionsActions = require('../flux/actions/TermsAndConditionsActions');

describe('TermsAndConditionsElement', function() {

  it('should have a checkbox', function() {
    expect(x.find('input[type="checkbox"]').length).toBe(1);
  });

  describe('when the checkbox is checked', function() {
    it('should signal agreed', function() {
      spyOn(TermsAndConditionsActions, 'agreed');
      x.find('input[type="checkbox"]').trigger('click');

      var checked = x.find('input[type="checkbox"]').is(':checked');
      expect(TermsAndConditionsActions.agreed).toHaveBeenCalledWith(checked);
    });
  });
});
```

TermsAndConditionsStore.spec.js

```
var TermsAndConditionsActions = require('../actions/TermsAndConditionsActions');
var x = require('../TermsAndConditionsStore');

describe('TermsAndConditionsStore', function() {

  describe('when TermsAndConditionsActions.agreed is invoked', function() {

    it('should notify listeners {agreed: true}', function(done) {

      TermsAndConditionsActions.agreed(true);
    });

  });

});
```

TermsAndConditionsStore.spec.js

```
var TermsAndConditionsActions = require('../actions/TermsAndConditionsActions');
var x = require('../TermsAndConditionsStore');

describe('TermsAndConditionsStore', function() {

  describe('when TermsAndConditionsActions.agreed is invoked', function() {

    it('should notify listeners {agreed: true}', function(done) {
      var removeListener = x.listen(function(d) {
        expect(d).toEqual({agreed: true});
        removeListener();
        done();
      });

      TermsAndConditionsActions.agreed(true);
    });

  });

});
```

shared/react/TestUtils.js

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

var TestUtils = {
  testComponent: function(component, props) {
    var $dom = $('

</div>');
    React.render(React.createElement(component, props), $dom[0]);
    $dom.appendTo('body');
    return $dom.children();
  },

  stubComponent: function(tag, props) {
    tag = tag || 'div';
    var simpleRender = function() {
      return React.createElement(tag, props);
    };

    return React.createClass({render: simpleRender});
  }
};

module.exports = TestUtils;


```

Example1.spec.js

```
$ karma start
INFO [framework.browserify]: registering rebuild (autoWatch=true)
INFO [karma]: Karma v0.12.31 server started at http://localhost:9876/
INFO [launcher]: Starting browser Chrome
INFO [Chrome 41.0.2272 (Windows 8.1)]: Connected on socket BhFSAPF2ZNpyPc4FtSWJ with id 44938074
INFO [framework.browserify]: 2466742 bytes written (7.80 seconds)
INFO [framework.browserify]: bundle built
Chrome 41.0.2272 (Windows 8.1): Executed 36 of 36 SUCCESS (0.218 secs / 0.204 secs)
```

So, what now?

So, what now?

It takes a lot to get started

Many dependencies

Error: Cannot read property 'firstChild' of undefined

So, what now?

It takes a lot to get started

Many dependencies

Error: Cannot read property 'firstChild' of undefined

It's very brittle

proxyquire plugin

defining transforms in browserify API

So, what now?

It takes a lot to get started

Many dependencies

Error: Cannot read property 'firstChild' of undefined

It's very brittle

proxyquire plugin

defining transforms in browserify API

Yeoman generator