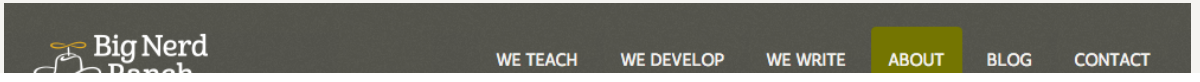
Building Blocks for Isomorphic JavaScript Apps

Chris Aquino, Big Nerd Ranch


Big Nerd
Ranch

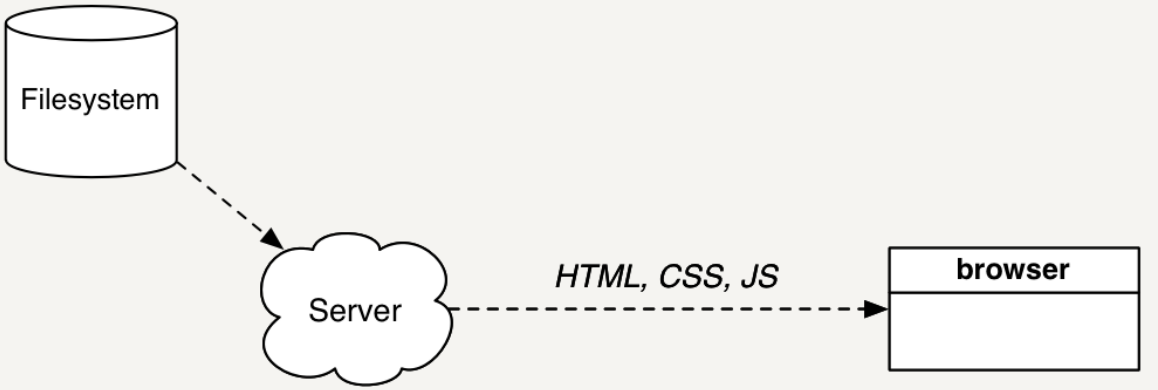
Arbitrarily ridiculous photo of me



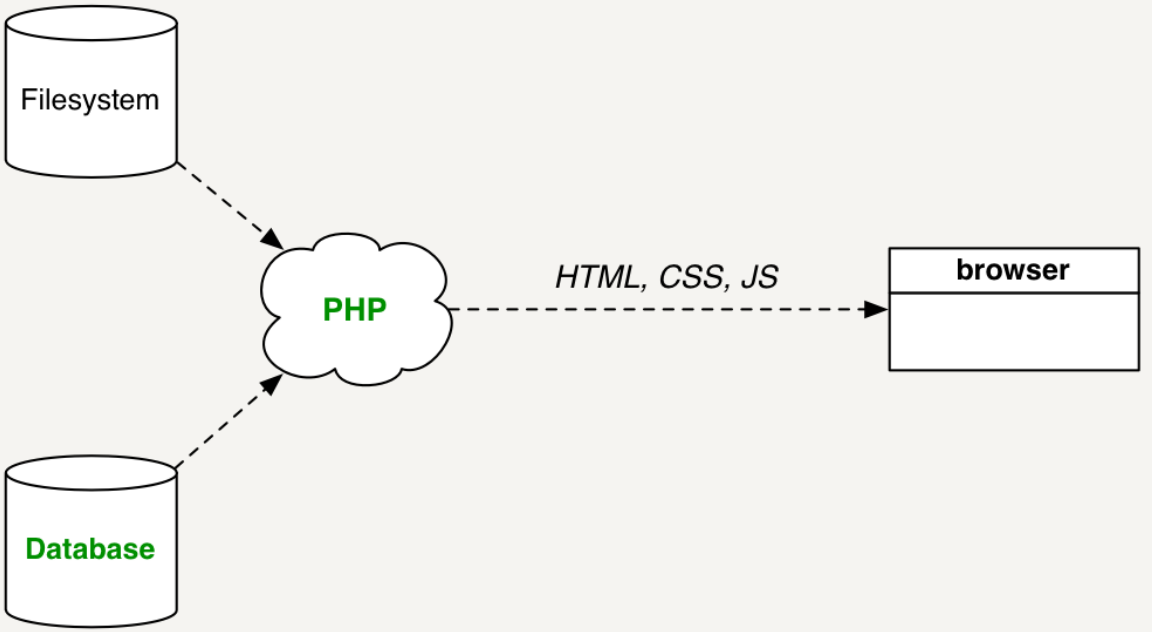
Front end development has gotten complex.

Are we actually doing a better job?

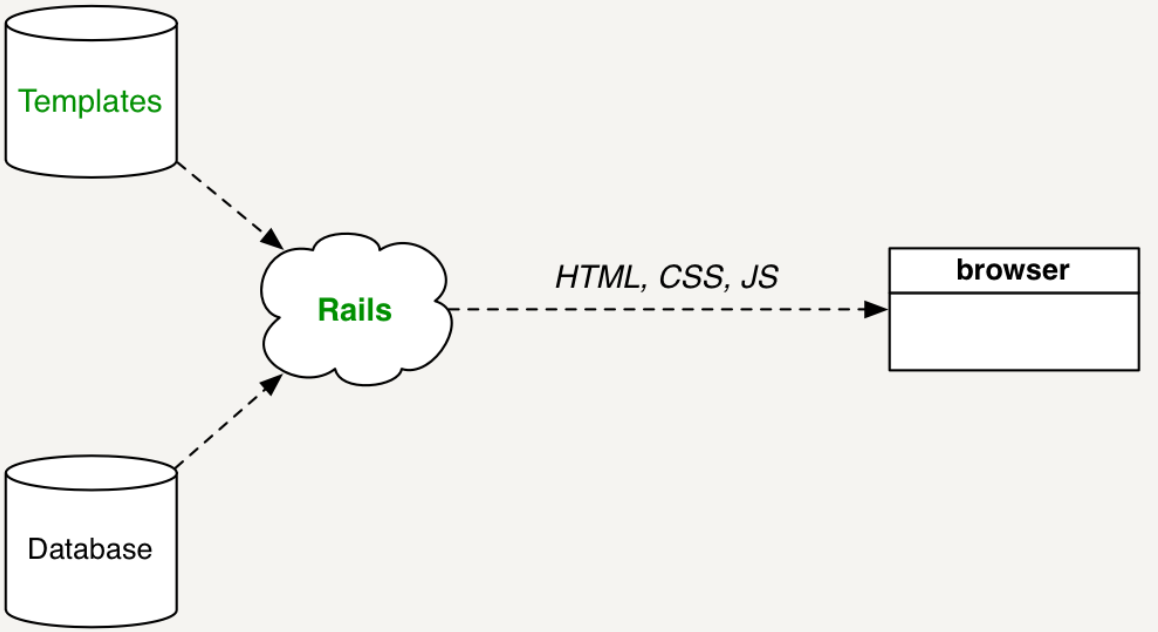
In the beginning...



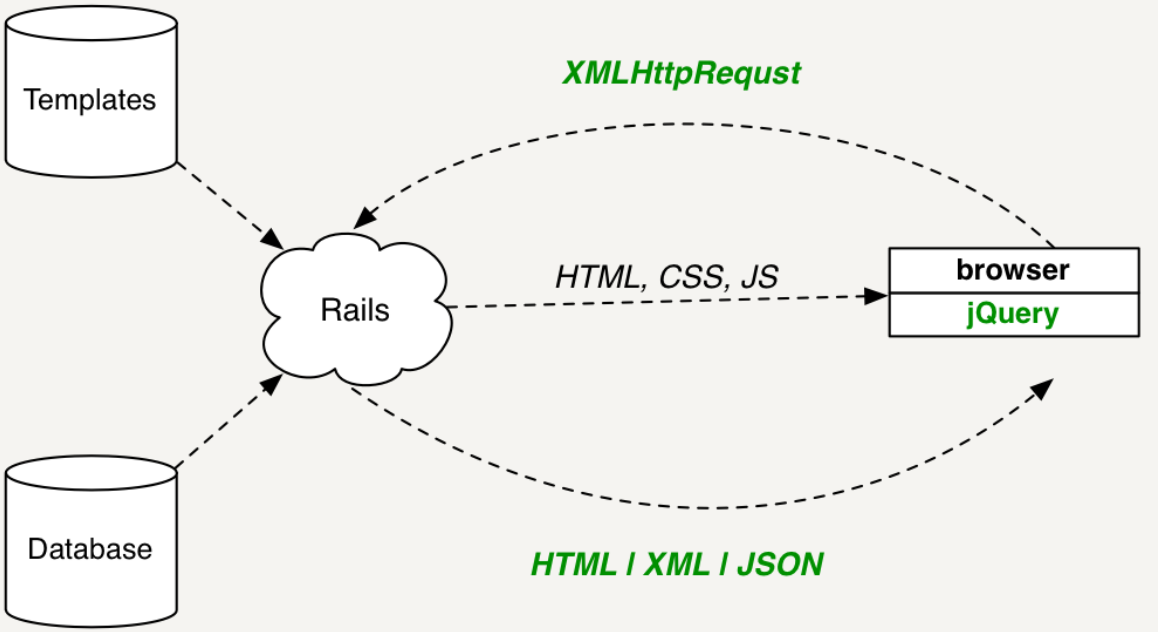
Database-driven



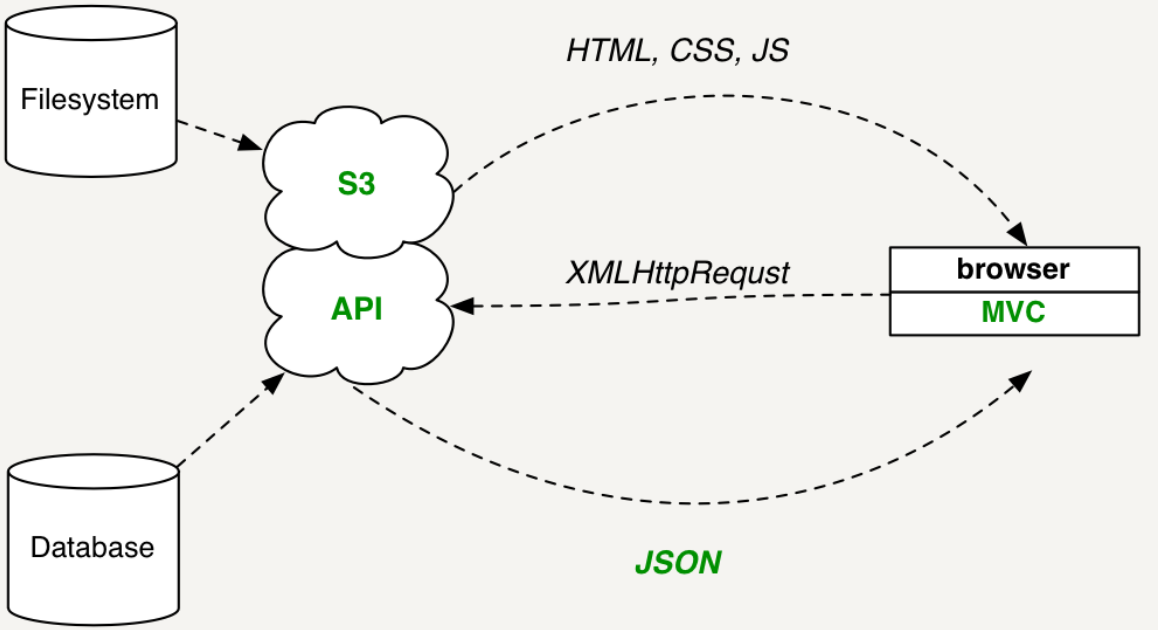
Server-side MVC



Ajax

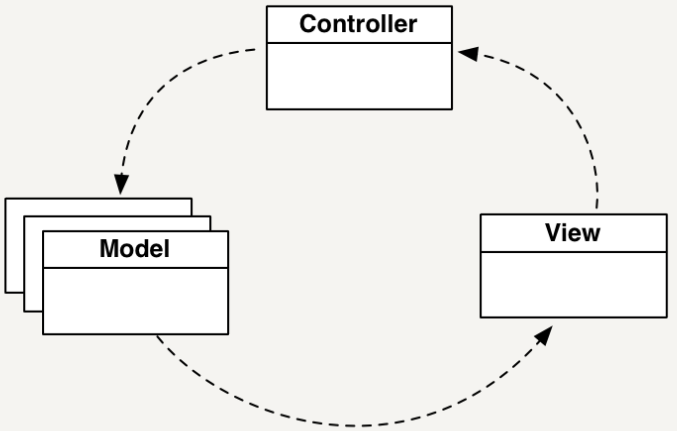

Templates
XMLHttpRequst
Rails
HTML, CSS, JS
browser
jQuery
Database
HTML / XML / JSON

Single Page Applications

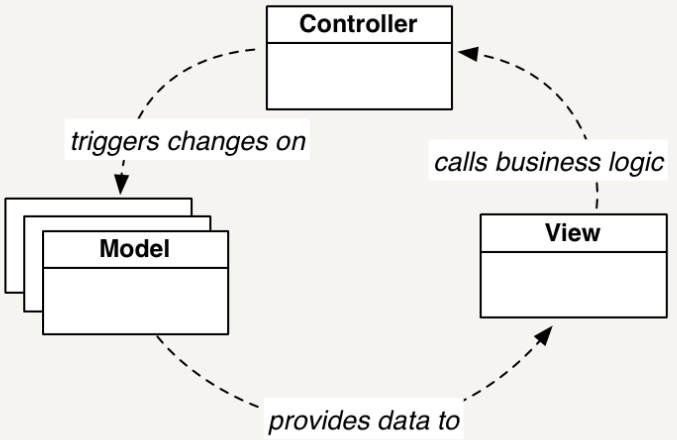


Anatomy of an MVC Application

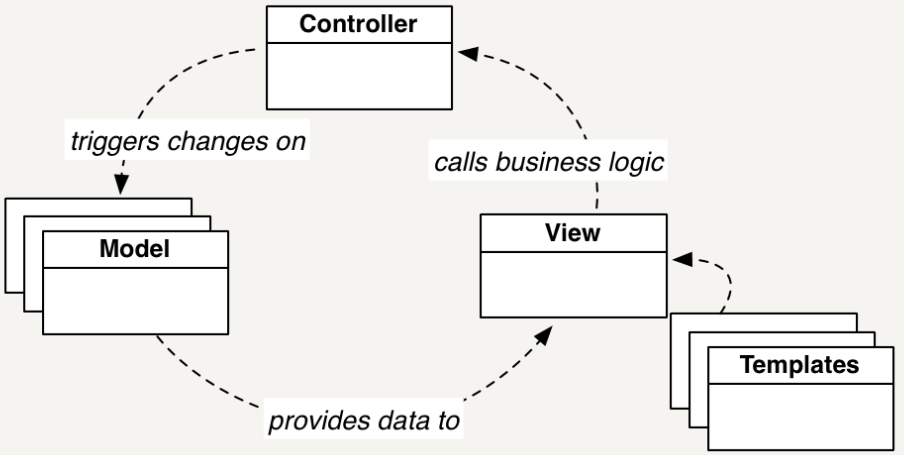
MV* in the Browser



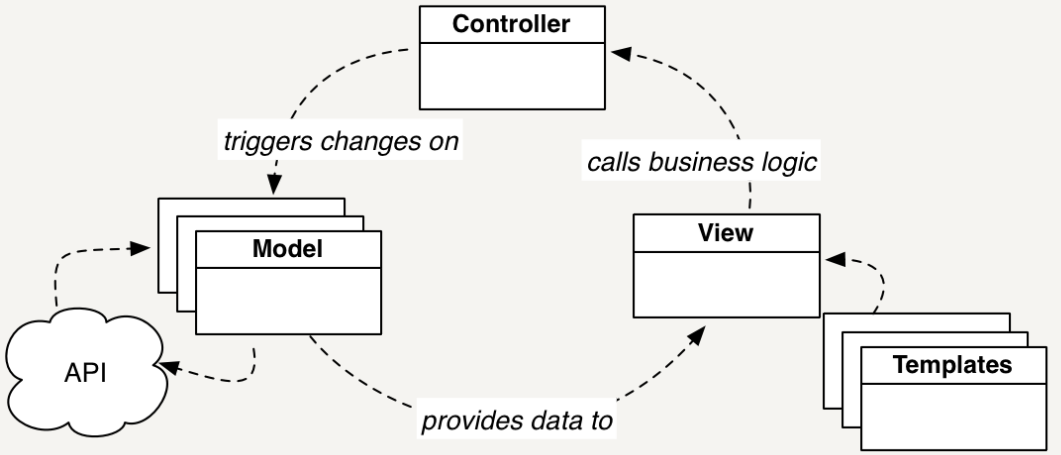
MV* in the Browser



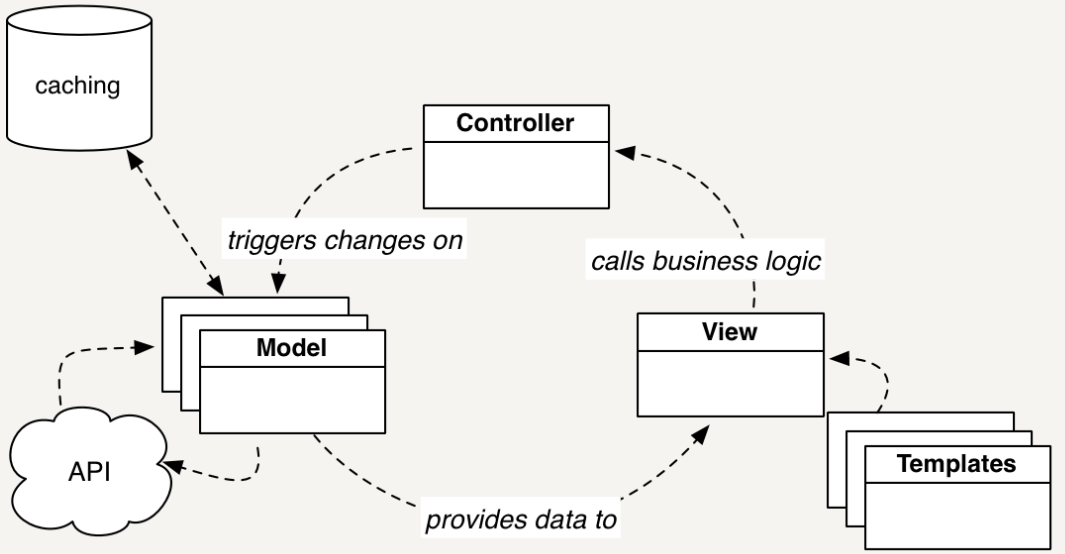
MV* in the Browser



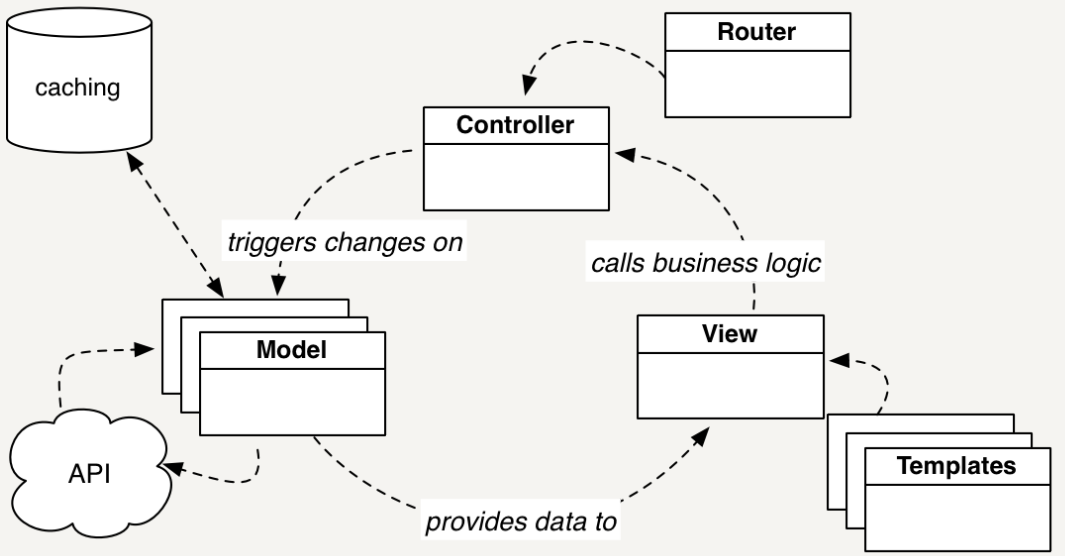
MV* in the Browser



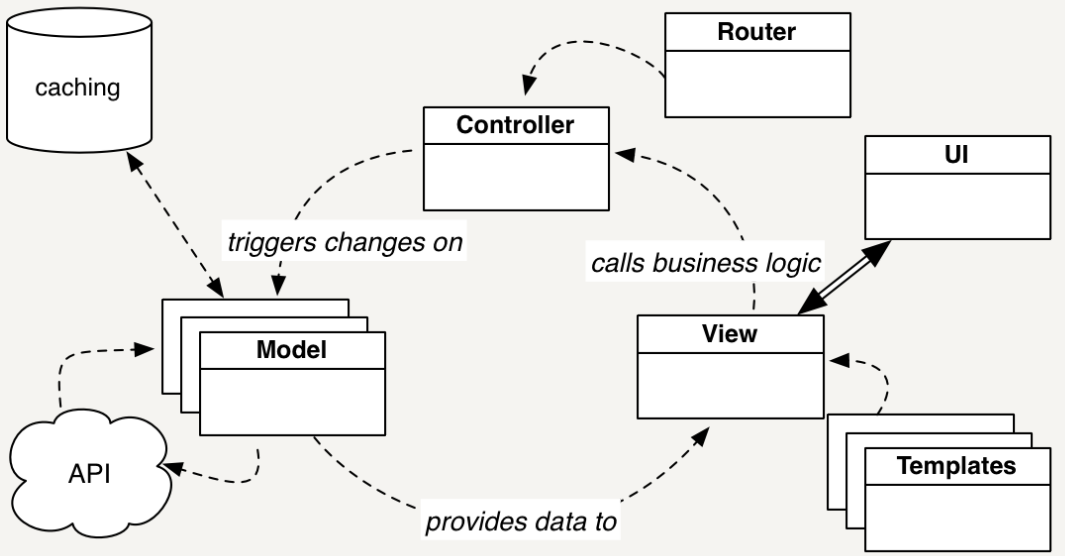
MV* in the Browser



MV* in the Browser

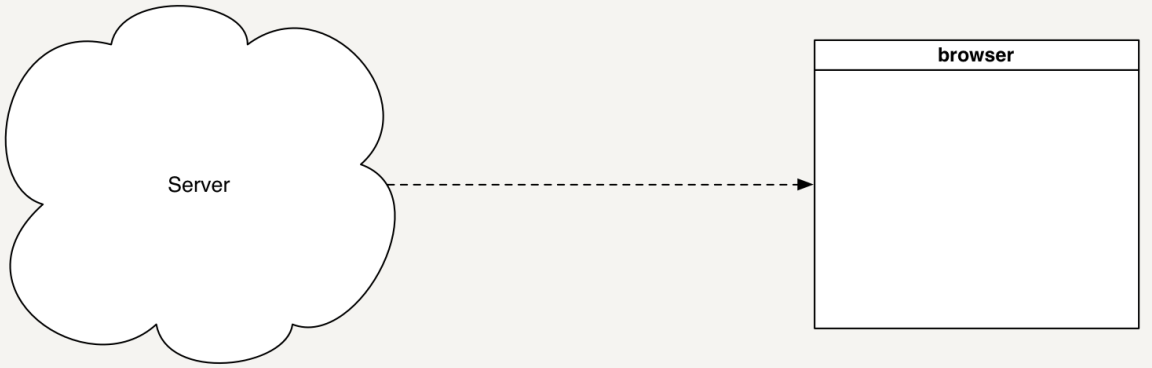


MV* in the Browser

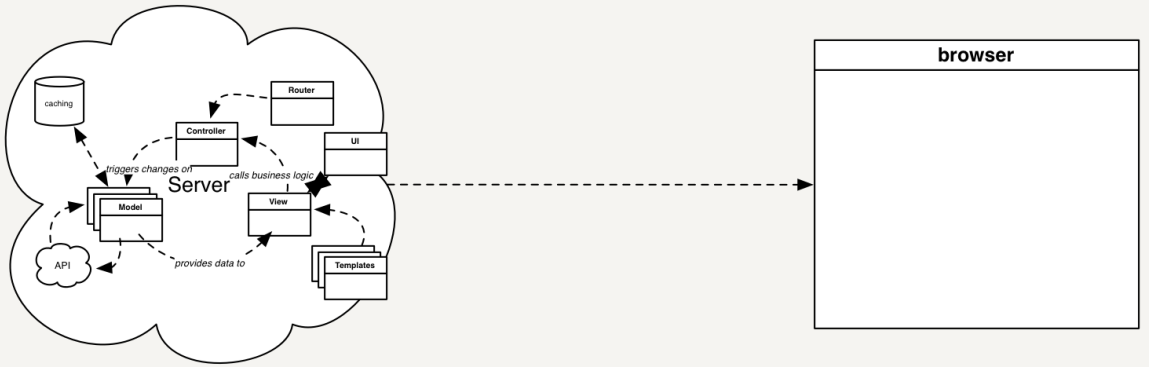


mvcwtf

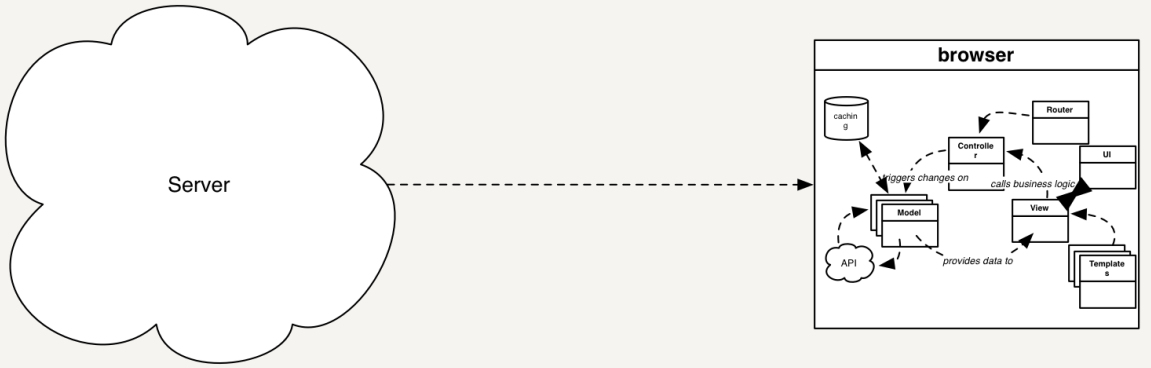
Shifting Responsibility



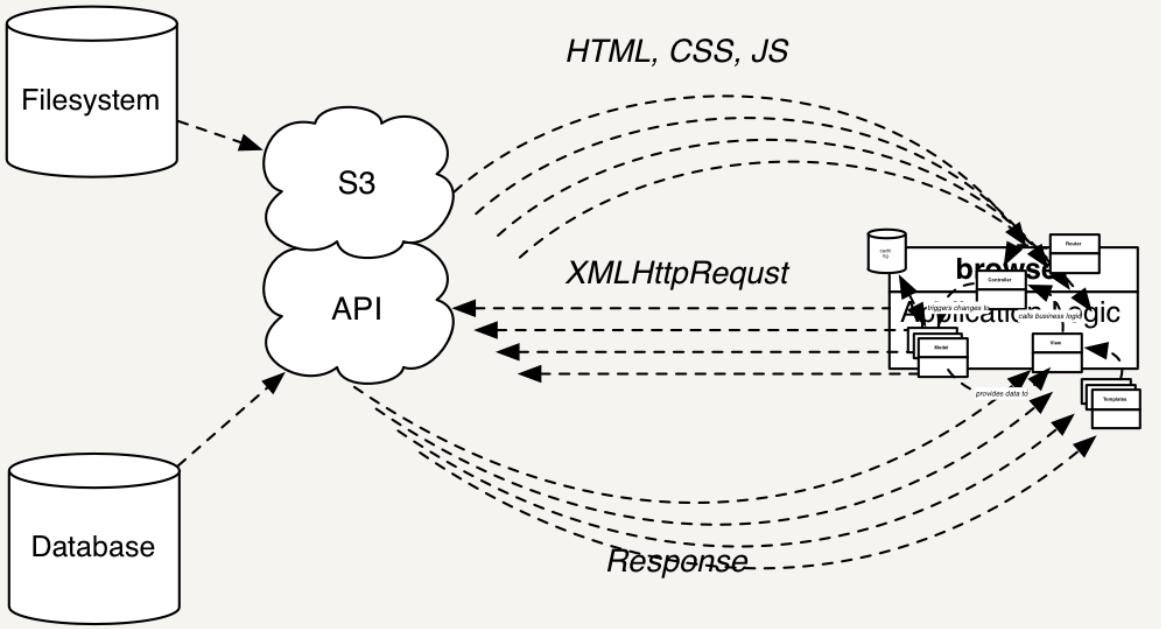
Shifting Responsibility



Shifting Responsibility



Thick clients, slow startup



Back end vs. UI

Many requests

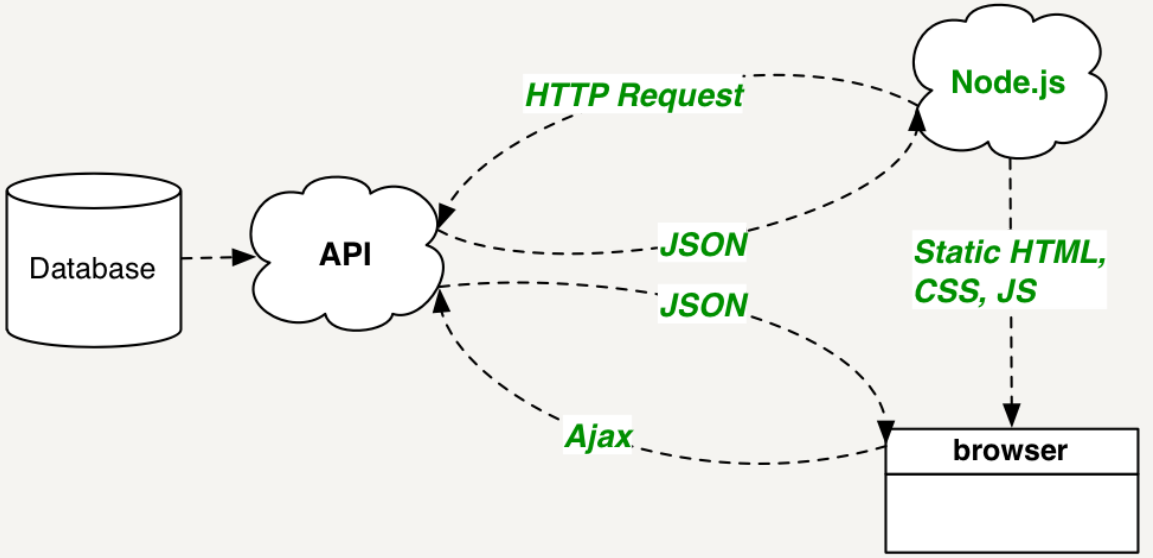
```
$.get('/api/users', function (data) { /* ... */});
$.get('/api/events', function (data) { /* ... */});
$.get('/api/groups', function (data) { /* ... */});
$.get('/api/puppies', function (data) { /* ... */});
```

Or vague endpoints

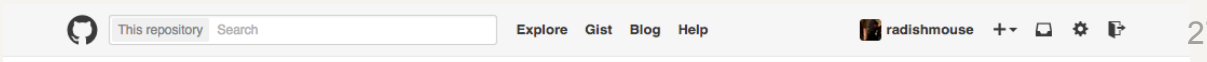
```
$.get('/api/all-the-things-for-dashboard', function (data) { /* ... */});
```

A (Possible) Solution

Isomorphic JavaScript Application Architecture



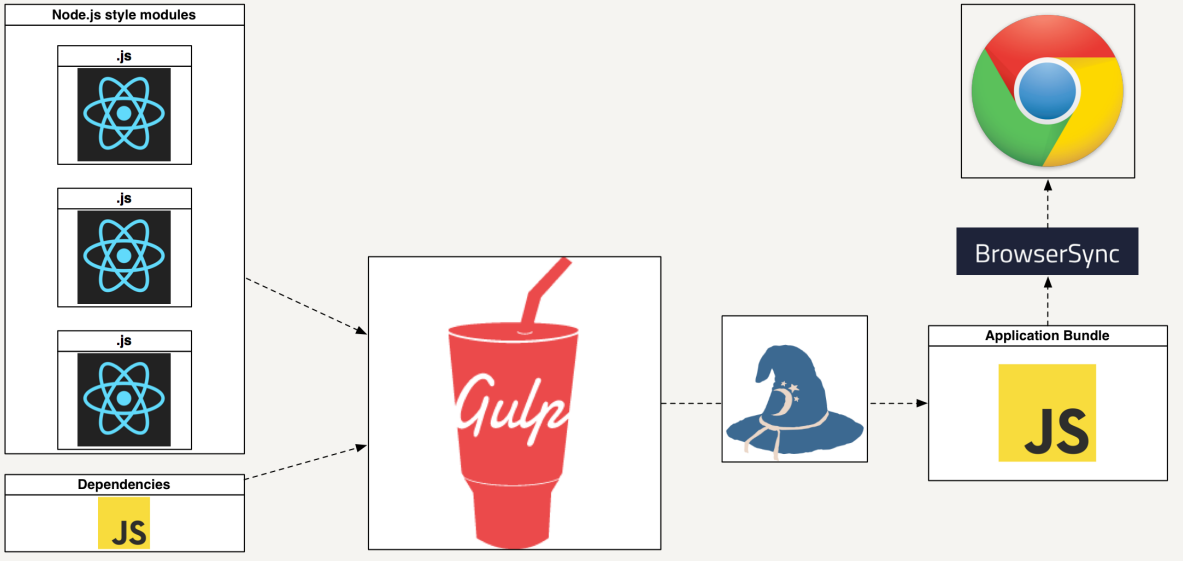
bit.ly/isojs-demo



"Isomorphic" JavaScript

- A single JavaScript application
- Runs in browser or Node.js
- Render views to static HTML from Node.js

Building and development

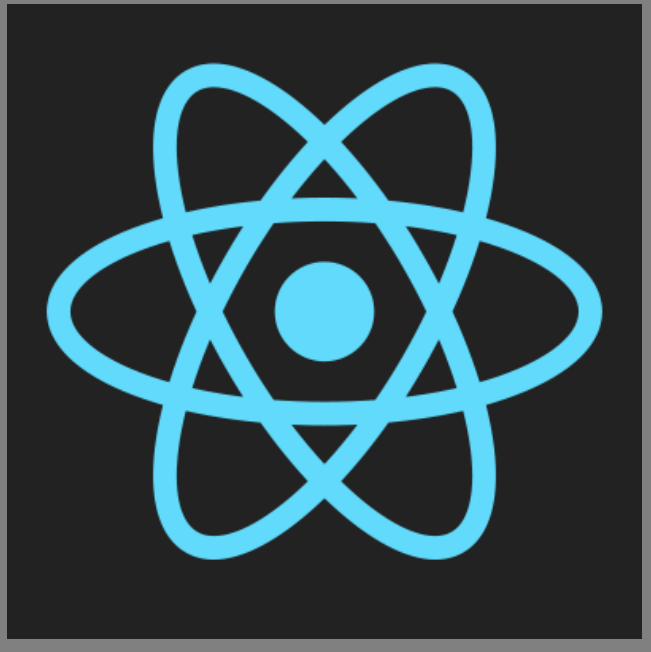


Gulp

```
var gulp = require('gulp');
var config = require('./config');
var paths = config.paths;

// Actual task functions are in `tasks` folder
var bSync = require('./tasks/sync');
var browserify = require('./tasks/browserify')(paths.js.app, paths.bundleNam

gulp.task('build', browserify);
gulp.task('browser-sync', ['build'], bSync.sync(paths.baseDir));

gulp.task('default', ['browser-sync'], function () {
    gulp.watch(paths.js.src, ['build', function () {
        bSync.reload(); // full reload
    }]);
});
```

Browserify

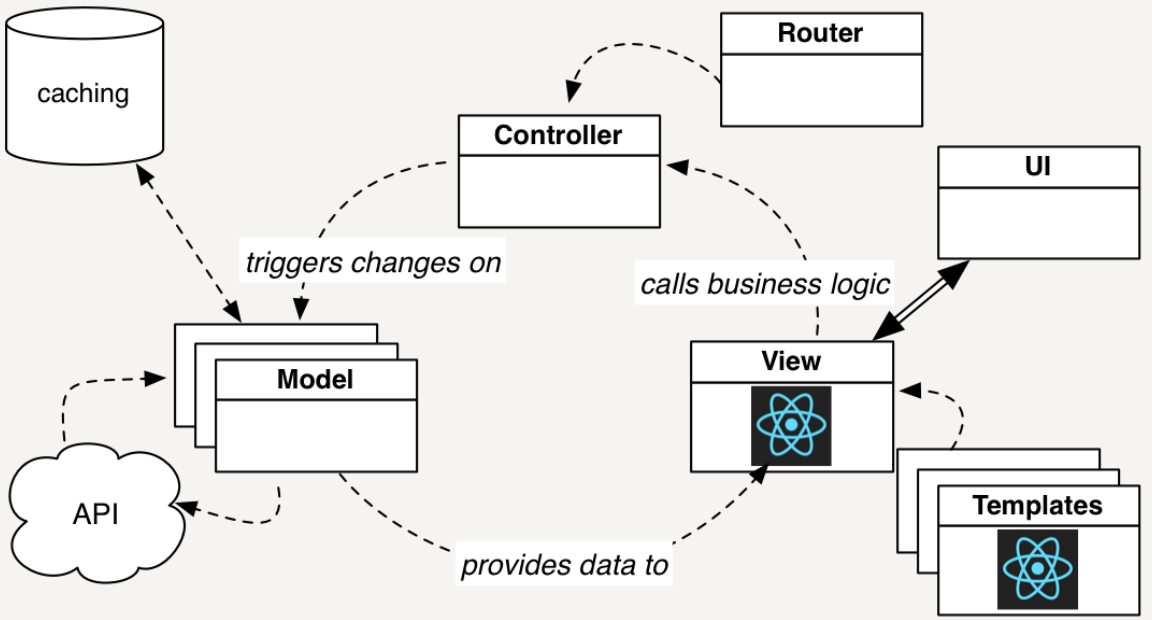
```
var gulp = require('gulp');
var browserify = require('browserify');
var reactify = require('reactify');
var source = require('vinyl-source-stream');

module.exports = function (src, bundleName, dest) {
    return function () {
        var b = browserify({
            detectGlobals : true
        });
        b.transform(reactify);
        b.add([src]);
        return b.bundle()
            .pipe(source(bundleName))
            .pipe(gulp.dest(dest));
    };
};
```



"A Javascript Library For Building User Interfaces"

- V in MVC
- Virtual DOM
- Data Flow
- Views can rendered to HTML strings

MVC with React



"Hello World" Module

```
/** @jsx React.DOM */
var React = require('react');
var HelloWorld = React.createClass({
    render: function () {




    }
});
module.exports = HelloWorld;
```

"Hello World" Module

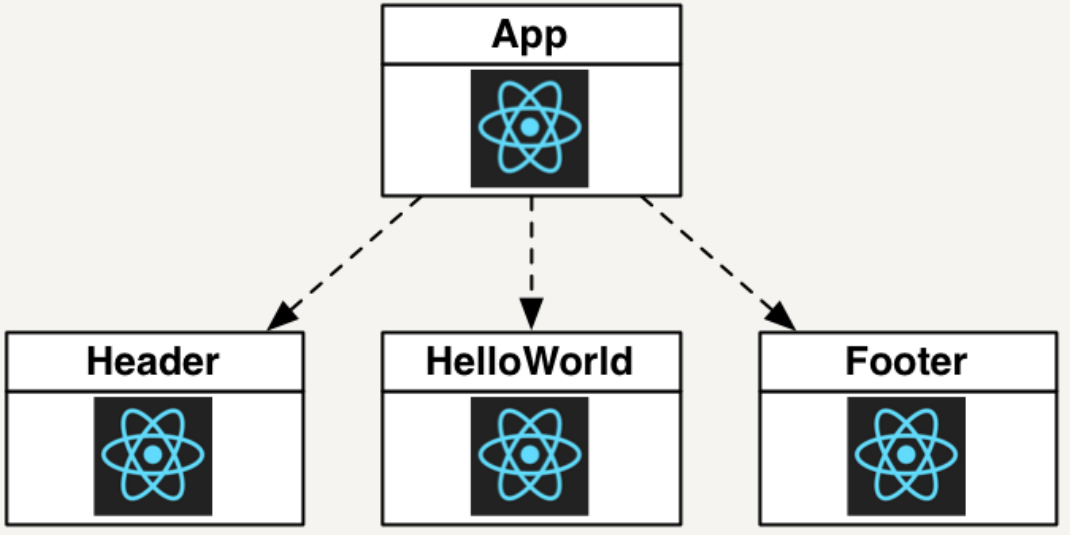
```
/** @jsx React.DOM */
var React = require('react');
var HelloWorld = React.createClass({
    render: function () {
        return (




        );
    }
});
module.exports = HelloWorld;
```

"Hello World" Module

```
/** @jsx React.DOM */
var React = require('react');
var HelloWorld = React.createClass({
    render: function () {
        return (
            <div>
                <h1>Hello, Connect-JS!</h1>
            </div>
        );
    }
});
module.exports = HelloWorld;
```

Views

View Component Structure



View Component: App

```
/** @jsx React.DOM */
var React = require('react');



var App = React.createClass({
    render: function () {
        return (
            <section>




            </section>
        );
    },



});
module.exports = App;
```

View Component: App

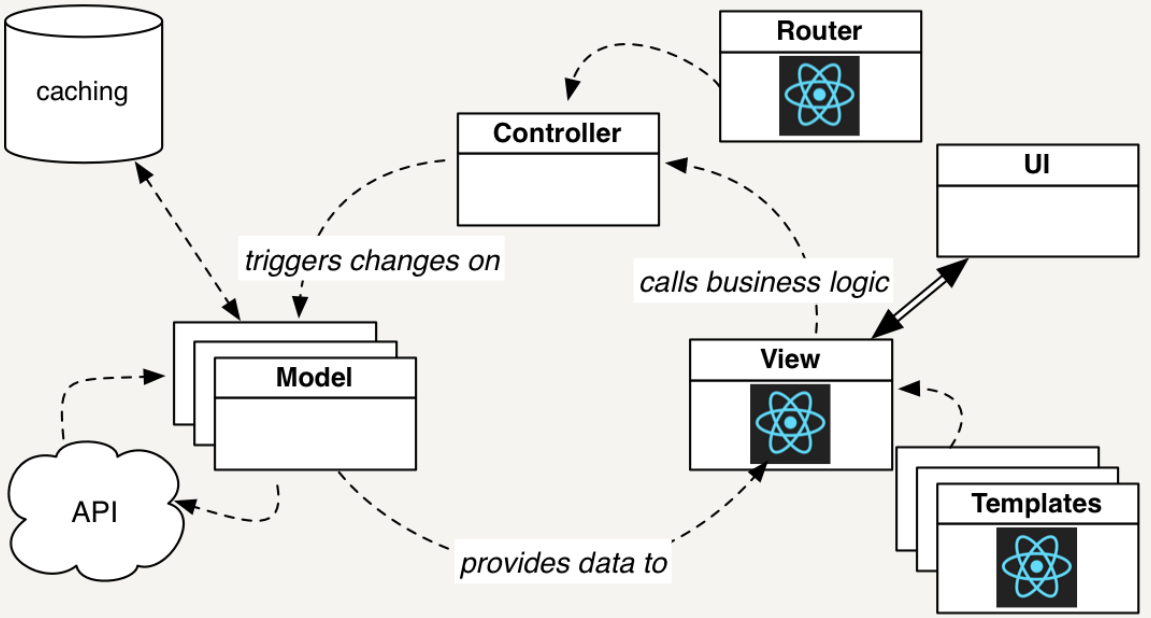
```
/** @jsx React.DOM */
var React = require('react');
var Header = require('./components/header');
var Footer = require('./components/footer');
var HelloWorld = require('./components/hello');
var App = React.createClass({
    render: function () {
        return (
            <section>




            </section>
        );
    },



});
module.exports = App;
```

View Component: App

```
/** @jsx React.DOM */
var React = require('react');
var Header = require('./components/header');
var Footer = require('./components/footer');
var HelloWorld = require('./components/hello');
var App = React.createClass({
    render: function () {
        return (
            <section>
                <Header />
                <HelloWorld />
                <Footer />
            </section>
        );
    },



});
module.exports = App;
```

Routing

Routing with React



Routing with React

Router

Component: Router

```
/** @jsx React.DOM */
var Router = require('react-router');
var Route = Router.Route;
var Routes = Router.Routes;
var DefaultRoute = Router.DefaultRoute;




var routes = (







);

module.exports = routes;
```

Component: Router

```
/** @jsx React.DOM */
var Router = require('react-router');
var Route = Router.Route;
var Routes = Router.Routes;
var DefaultRoute = Router.DefaultRoute;
var App = require('./App');
var HelloWorld = require('./components/hello');
var About = require('./components/about');
var Yep = require('./components/yep');
var routes = (









);

module.exports = routes;
```

Component: Router

```
/** @jsx React.DOM */
var Router = require('react-router');
var Route = Router.Route;
var Routes = Router.Routes;
var DefaultRoute = Router.DefaultRoute;
var App = require('./App');
var HelloWorld = require('./components/hello');
var About = require('./components/about');
var Yep = require('./components/yep');
var routes = (
    <Routes location="history" scrollBehavior="browser">
        <Route name="app" path="/" handler={App}>
            <Route name="yep" handler={Yep}></Route>
            <Route name="about" handler={About}></Route>
            <DefaultRoute handler={HelloWorld}></DefaultRoute>
        </Route>
    </Routes>
);

module.exports = routes;
```

Component: Router

```
/** @jsx React.DOM */
var Router = require('react-router');
var Route = Router.Route;
var Routes = Router.Routes;
var DefaultRoute = Router.DefaultRoute;
 var App = require('./App');
var HelloWorld = require('./components/hello');
var About = require('./components/about');
var Yep = require('./components/yep');
var routes = (
    <Routes location="history" scrollBehavior="browser">
        <Route name="app" path="/" handler={App}>
            <Route name="yep" handler={Yep}></Route>
            <Route name="about" handler={About}></Route>
            <DefaultRoute handler={HelloWorld}></DefaultRoute>
        </Route>
    </Routes>
);

module.exports = routes;
```

Component: Router

```
/** @jsx React.DOM */
var Router = require('react-router');
var Route = Router.Route;
var Routes = Router.Routes;
var DefaultRoute = Router.DefaultRoute;
var App = require('./App');
var HelloWorld = require('./components/hello');
var About = require('./components/about');
var Yep = require('./components/yep');
var routes = (
    <Routes location="history" scrollBehavior="browser">
        <Route name="app" path="/" handler={App}>
            <Route name="yep" handler={Yep}></Route>
            <Route name="about" handler={About}></Route>
            <DefaultRoute handler={HelloWorld}></DefaultRoute>
        </Route>
    </Routes>
);

module.exports = routes;
```

Component: Router

```
/** @jsx React.DOM */
var Router = require('react-router');
var Route = Router.Route;
var Routes = Router.Routes;
var DefaultRoute = Router.DefaultRoute;
var App = require('./App');
var HelloWorld = require('./components/hello');
var About = require('./components/about');
var Yep = require('./components/yep');
var routes = (
    <Routes location="history" scrollBehavior="browser">
        <Route name="app" path="/" handler={App}>
            <Route name="yep" handler={Yep}></Route>
            <Route name="about" handler={About}></Route>
            <DefaultRoute handler={HelloWorld}></DefaultRoute>
        </Route>
    </Routes>
);

module.exports = routes;
```

Server-Side Rendering

Rendering for a route

```
var express = require('express');
var webapp = express();
var path = require('path');
var React = require('react');
require('node-jsx').install(); // Compile JSX on the fly



webapp.get('*', function (req, res) {









});

var PORT = 1337;
webapp.listen(PORT);

console.log('Listening on ' + PORT);
```

Rendering for a route

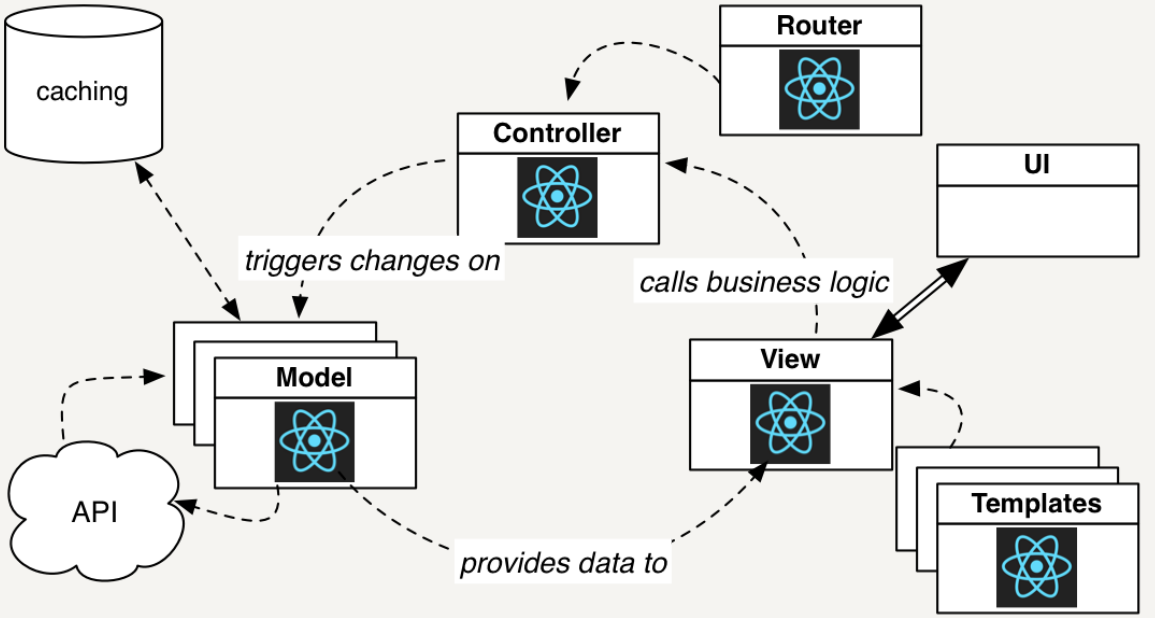
```
var express = require('express');
var webapp = express();
var path = require('path');
var React = require('react');
require('node-jsx').install(); // Compile JSX on the fly
var Router = require('react-router');
var AppRoutes = require('../src/routes');

webapp.get('*', function (req, res) {
    Router.renderRoutesToString(AppRoutes, req.path, function (err, reason,




    });
});

var PORT = 1337;
webapp.listen(PORT);
console.log('Listening on ' + PORT);
```

Rendering for a route

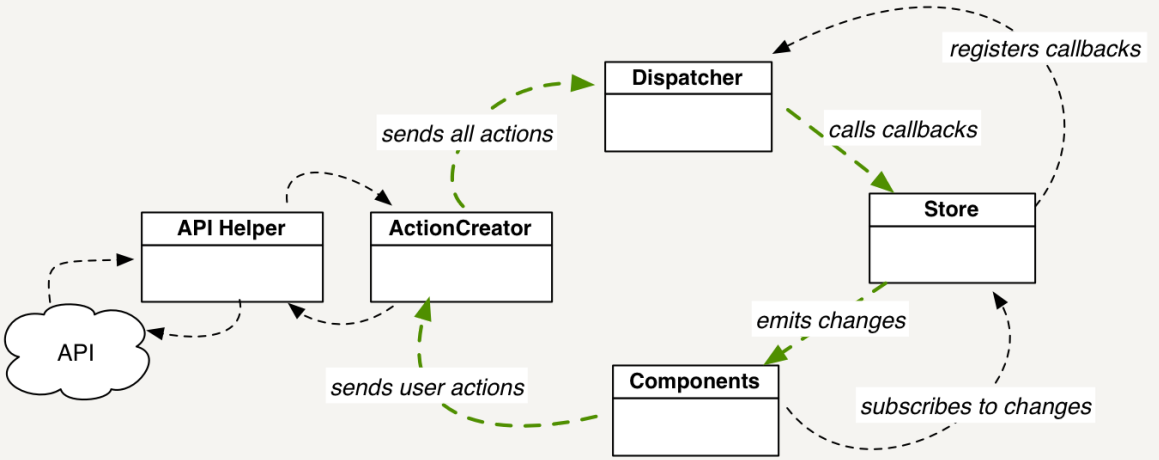
```
var express = require('express');
var webapp = express();
var path = require('path');
var React = require('react');
require('node-jsx').install(); // Compile JSX on the fly
var Router = require('react-router');
var AppRoutes = require('../src/routes');

webapp.get('*', function (req, res) {
    Router.renderRoutesToString(AppRoutes, req.path, function (err, reason,
        var htmlString = '<!doctype html><html><head></head><body>';
        htmlString += string;
        htmlString += '<script src="scripts/bundle.js"></script>';
        htmlString += '</body></html>';
        res.send(htmlString);
    });
});

var PORT = 1337;
webapp.listen(PORT);
console.log('Listening on ' + PORT);
```

Models and Controllers

React all the things



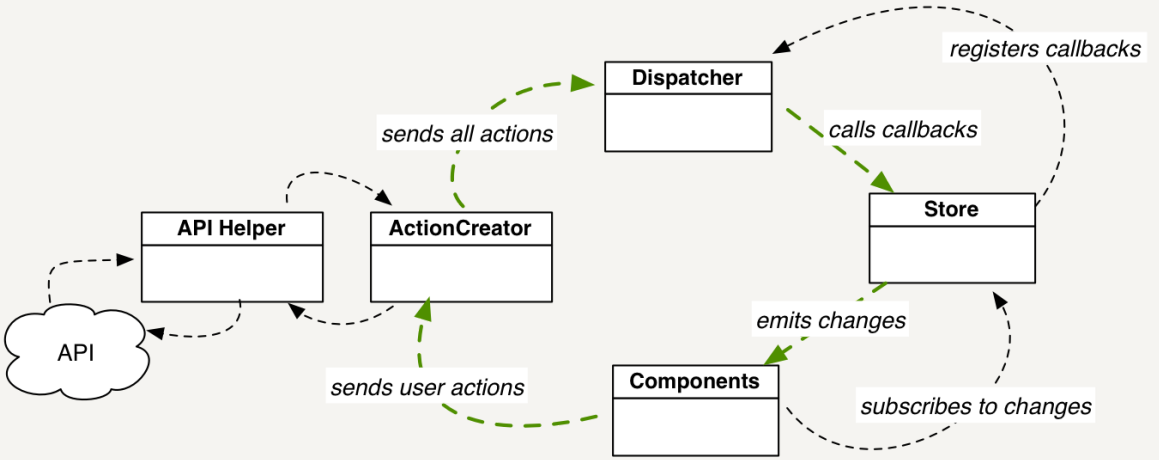
But...


Dispatcher
registers callbacks
sends all actions
calls callbacks
Store
API Helper
ActionCreator
API
emits changes
sends user actions
Components
subscribes to changes

wtflux?

Don't worry, it will make sense

Flux Architecture Overview



Flux Architecture: Dipatcher

```
var Dispatcher = require('flux').Dispatcher;
var copyProperties = require('react/lib/copyProperties');
var AppDispatcher = copyProperties(new Dispatcher(), {

    handleServerAction: function(action) {
        /* ... */
    }
    ...
});

module.exports = AppDispatcher;
```

Flux Architecture: Actions

```
var AppDispatcher = require('../dispatcher/AppDispatcher');

module.exports = {
    receiveAll: function(rawData) {

        AppDispatcher.handleServerAction({
            type: "RECEIVE_DATA",
            rawData: rawData
        });

    }
};
```

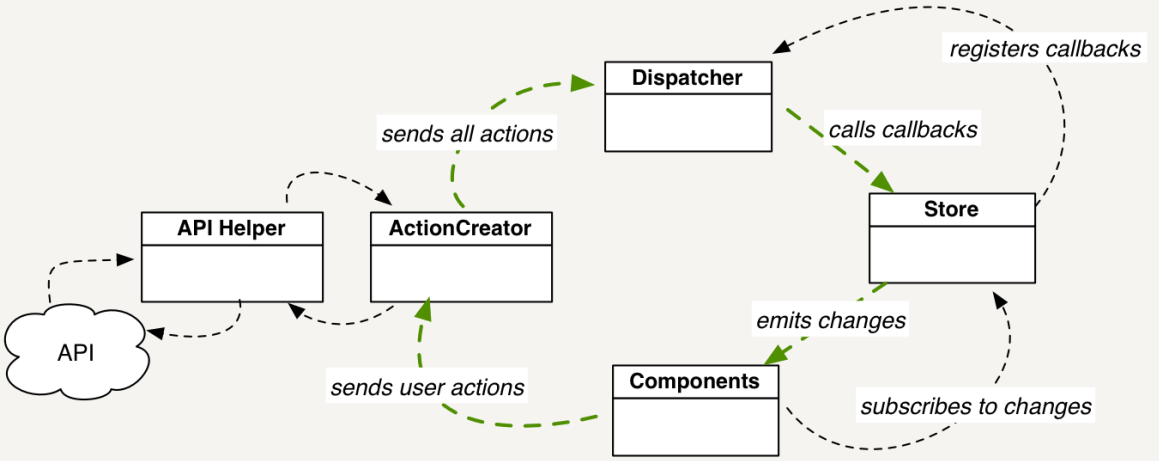
Flux Architecture: Stores

```
var AppDispatcher = require('../dispatcher/AppDispatcher');
...
var Store = merge(EventEmitter.prototype, {
    /* ... */
});
Store.dispatchToken = AppDispatcher.register(function(payload) {
    _data = payload.action.rawData;
    Store.emitChange();
});
module.exports = Store;
```

Flux Architecture: Views Subscribing to Stores

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

    componentDidMount: function() {
      Store.addChangeListener(this._onChange);
    },

    render: function () {...},

    _onChange: function () {
        /* ... */
    }
});

module.exports = HelloWorld;
```

One way data flow



Isomorphic Libraries

SuperAgent

```
var ActionCreator = require('../actions/ActionCreator');
var request = require('superagent');
var URL = 'http://my.remote.server/endpoint';

var _data = [];
module.exports = {

    getAllNodes: function() {



        request.get(URL, function (res){
            _data = res.body;
            ActionCreator.receiveAll(_data);
        });
    }
};
```

Isomorphic Library: SuperAgent

```
var ActionCreator = require('../actions/ActionCreator');
var request = require('superagent');
var URL = 'http://my.remote.server/endpoint';

var _data = [];
module.exports = {

    getAllNodes: function() {


        ActionCreator.receiveAll(_data);

        request.get(URL, function (res){
            _data = res.body;
            ActionCreator.receiveAll(_data);
        });
    }
};
```

So, are we doing better?

(I think yes.)

Focused Development Energy

- Single JavaScript codebase
- API developers no longer concerned with UI

React + Flux

- Simple patterns that scale
- Side-steps issues with data-binding
- Simple server-side rendering of deep links

Affordances for Designers and UI Engineers

- Components allow for atomic design
- JSX is familiar
- Components usable for styleguide-driven development

And, the user?

- Sees stuff sooner!
- HTML pre-rendered with data
- Reduced rendering overhead when data changes

Tradeoffs

Not an established ecosystem

- Some assembly required
- Smaller community
- No training or books

Not an established ecosystem, but...

- Examples exist: bit.ly/isojs-demo
- Setup is `npm install && gulp`
- Community is combination of Node.js and Front End developers
- Big Nerd Ranch offers training for Cross-Platform JavaScript Apps

Want more?

Moar!

- Next: Advanced React.js with Client Ayres
- Soon: Isomorohpic JS Apps on the Big Nerd Ranch Blog
- 2015: Cross-Platform JavaScript Apps Training Course, with the Isomorphic Stack

Thank you for listening!

@radishmouse, @bignerdranch