Adopting Node.js and CoffeeScript in a Software Design Course

Midwest Instruction and Computing Symposium
April 25, 2014
Maxwell Marti
mart2967@morris.umn.edu
The Software Design and Development Course

- Single project
- Small groups
- Agile methodology
- External customer: LightSide Labs
- Current technology
Node.js

- Framework for network applications
- Uses the V8 JavaScript Engine
- Built in web server
- Asynchronous I/O and JavaScript evaluation
- Single-threaded event loop
- Large number of additional modules
Node Modules

- Express
- Hogan
- Blanket
CoffeeScript

- A language that transpiles to JavaScript
- Clean syntax
- About 1/3 fewer lines of code
- Scope safety and object encapsulation
CoffeeScript to JavaScript Example

```javascript
var myObj, num,
    numSquared, square;

num = 9;
square = function(x) {
    return x * x;
};
numSquared = square(num);
myObj = {
    name: "Mr. Object",
    likes: "squaring numbers",
    mySquare: square
};
```
Adopting Node.js and CoffeeScript in a Software Design Course

MongoDB

- Document oriented database
- Records in JavaScript Object Notation (JSON)
- Queries in JavaScript
- Mongoose module provides extra utility
jQuery

- DOM interaction library
- Ubiquitous
- Powerful selectors and functions
- Integrates with other front-end tools
jQuery Example

// set html for element
$("p:first").html( "This is the <em>first</em> paragraph");

// set value for all inputs of myClass
$("input.myClass").val("default text");

// bind a function to the click event of an element
$("#myButton").on("click", function(){
    alert("Button was clicked");
});
Backbone.js

- MV* Framework for client-side applications
- HTML rendering on the browser
- Models, Views, Routers
- Hierarchical views with event emission
- REST API conformity with Models and Collections
- Minimal opinionation
Bootstrap

- Formerly Twitter Bootstrap
- Responsive grid layouts
- CSS classes and JavaScript utilities for the client
- Easy to customize
- Simplifies web interface design
Development Workflow

- WebStorm IDE (based on IntelliJ IDEA)
- Unit testing with Mocha
- Version control with GitHub
- Continuous integration with Strider
Strider

- Continuous integration and deployment server
- Pulls code from GitHub
- Compiles CoffeeScript and runs tests
- Computes code coverage metrics
- Deploys projects to web server for evaluation
- Supports plugins and custom shell scripts
Unit Testing With Mocha

- Defines structure and syntax of test code
- Choice of assertion libraries
- Different testing styles supported
- Versions for Node.js and client-side code
Conclusion and Observations

- Unified language both helps and hurts students
- Particular difficulty with Backbone.js
- Module driven development
- High velocity and low familiarity affects testing practices
- Customer interaction