CMSC 389R: Ruby on Rails

Neil Spring

Winter 2011

Instructor  Neil Spring
E-mail nspring@cs (include “417” in your subject line)
Class  MTuTh 1:00-2:35 CSIC 1122
Office hours  After class in CSIC 1122 or AVW 4133
Web  http://www.cs.umd.edu/class/winter2011/cmsc389r/
Forum  https://forum.cs.umd.edu/ (Don’t use the forum send me private messages unless they have no urgency.)

1 Goals of this course

This course has two goals:

1. To prepare you for professional work creating or modifying Rails-based applications.

2. To provide program design and debugging experience that will prepare you for later CS coursework.

2 Summary

The content of the course will follow the Agile textbook. There are eight class meetings (celebrate Martin Luther King Jr. Day). The schedule is as follows. (Subject to change.)

Meeting 1  Administrative detail (this syllabus), Rails and Ruby philosophy, history, model-view-controller, quick SQL, Ruby review and pitfalls.

Meeting 2  Application construction walkthrough, Views, HTML overview, scaffolds, associations and their implementation, unix shell skills.

Meeting 3  Migrations, plurals, string manipulation and sanitizing, indexes,

Meeting 4  Security, user authentication, filters, validations, observers, flash messages.

Meeting 5  Testing, searching, caching, YAML.

Meeting 6  Debugging, pagination, routing, REST.

Meeting 7  HTTP redirections, project demos.

Meeting 8  Demos and exam.
3 Prerequisites

CMSC 216 and 250. Experience with Ruby in 330 may help you.

No Fear.

4 Laptops

Bring them to class.

5 Lecture Style

I typically present lectures from an emacs buffer, favoring a fluid discussion built interactively instead of a rigid presentation of course material in outline form. The goal of this lecture style is to keep your mind actively engaged with the material and discourage tuning out as you might if you “understand” the whole powerpoint slide. The lecture buffer will be posted to the web sometime after class.

6 Grading

Although I provide the following breakdown and intend to adhere to it closely, it does not have strategic value: failure to complete an assignment will hurt your grade.

6.1 Participation: 10%


You’re going to learn things from web pages. Most people who learn Rails did so on their own using on-line documentation.

Don’t sit in the back row.

6.2 Homeworks: 20%

I will have three homework assignments that will cover the range of topics Rails comprises. You may have to do some online research to answer questions. Cite your sources. (I take citing sources seriously – when your source is wrong, I can try to untangle how you were misled. I once had a student copy code from a forum post that asked “why doesn’t this code work?” but since it was cited, I could chastise him and share this story with you.)

6.3 Final Exam: 30%

My final exams tend to be partly multiple choice. I have not decided on a format for this exam, however. It may need to be open book since the space of topics is so vast, and that may be challenging to provide with on-line copies. My multiple choice questions tend to be difficult to answer based on the structure of the questions.

My exams are also vocabulary-based. I will have a class vocabulary sheet for you to study. For each term, you’re expected to know what it is, why it’s good, and what it stands for (if an acronym).

6.4 Project: 40%

You will construct a project of your choosing. Project ideas will be distributed in a separate sheet for course participants. You may work in pairs; if your project is ambitious, you may add a third member. You may also work alone if you don’t like people... just don’t expect to turn in half the work. Four shalt thou not count.
I will require that you add specific features to your project application as it develops: validations, unit tests, associations, user authentication, and a migration. Every turnin must pass its own unit tests.

Your application need not be pretty; CSS and AJAX are not the subject of the course. If you happen to know how to make web pages look good, I won’t complain, it’s just not required.

7 Lateness

There’s really no time for lateness in this compressed schedule. I need to grade work quickly and return it to you. Expect a 25% per day penalty on any assignment. If you’re going to be late with an assignment for reasons outside your control, please let me know before the deadline.

8 Administrative Cruft

I dislike this section greatly, but codifying each of these policies is important for keeping myself sane and making clear what my expectations are. I’d much prefer a section that said “treat me with respect and I’ll do the same for you;” this section is intended mostly for those who would hope to game the system. Note that I copied verbatim some of these passages; I hope you appreciate irony.

8.1 Excused absences

Students claiming a excused absence must apply in writing and furnish documentary support (such as from a health care professional who treated the student) for any assertion that the absence qualifies as an excused absence. The support should explicitly indicate the dates or times the student was incapacitated due to illness. Self-documentation of illness is not itself sufficient support to excuse the absence. An instructor is not under obligation to offer a substitute assignment or to give a student a make-up assessment unless the failure to perform was due to an excused absence. An excused absence for an individual typically does not translate into an extension for team deliverables on a project.

8.2 Religious observances

Please inform me in the first or week of class of religious observances that will interfere with your ability to complete assignments on time.

8.3 Disability Support

(Although the people who need this already know it…)

Any student eligible for and requesting reasonable academic accommodations due to a disability is requested to provide, to the instructor in office hours, a letter of accommodation from the Office of Disability Support Services (DSS) within the first two weeks of the semester.

8.4 Honor code

The University of Maryland, College Park has a nationally recognized Code of Academic Integrity, administered by the Student Honor Council. This Code sets standards for academic integrity at Maryland for all undergraduate and graduate students. As a student you are responsible for upholding these standards for this course. It is very important for you to be aware of the consequences of cheating, fabrication, facilitation, and plagiarism. For more information on the Code of Academic Integrity or the Student Honor Council, please visit [http://www.studenthonorcouncil.umd.edu/whatis.html](http://www.studenthonorcouncil.umd.edu/whatis.html).
8.5 What constitutes cheating?

(And now back to stuff I wrote.)

Copying other assignments, looking over someone’s shoulder in the lab, emailing function code, using google to find a code fragment without understanding, looking for code in other people’s directories, pulling code printouts off printers, not working with your partner, alternating assignments with your partner, and in any other way attempting to gain a grade without learning.

Note: cheating goes both ways; leaving someone your code because you want to help is just as bad as borrowing someone else’s code. We can tell when code looks and acts too similar to be independent work; we can’t (easily) tell which of two implementations was the original.

8.6 What constitutes legal collaboration?

Interaction via mailing list or discussion of problem and code solutions governed by the Gilligan’s Island rule[1]Collaborative interoperability testing, experimenting with whether two independent implementations can talk to each other, is also permitted, so long as code is not shared.

Using google is OK. Using wikipedia is encouraged. If you find a particularly good solution on either, please cite it; there is no penalty for citing sources and I’m more likely to consider answers that disagree with textbook or lecture legitimate. If you find a question far too easy because an answer is present on-line, please let me know.

---

[1]You understand the concept only if you can watch one half-hour complete episode of Gilligan’s Island and still retain the concept. You may then begin coding with your newfound knowledge safe that it is your own work. Without the thirty minute pause, it is not your work.