

Computational Social Science 605
Fall 2011 Semester
SYLLABUS

George Mason University
Innovation Hall, Room 320
Thursdays, 4:30-7:10pm

Object-Oriented Modeling in Social Science

Maksim Tsvetovat, Ph.D.
Asst. Professor of Computational Social
Science
mtsvetov@gmu.edu
Tel. (703) 993-1405

Welcome to CSS 605! This is a course about discovery and invention in the social sciences, so be prepared to learn in a way that may be unlike any previous social science course you may have taken. We hope you will learn from this course as much as we have learned in envisioning, designing, developing, and implementing it for you.

This syllabus covers the main features of CSS 605—or “attributes” and “methods”, as you shall soon learn to view and understand these characteristics in the “object paradigm”—while additional information is contained in separate handouts that will be distributed as the course develops. This Syllabus is Handout #1 and covers course description, assumptions, learning objectives, grading guidelines, material to be covered, and some initial references. Welcome aboard!

Description: This graduate course presents and applies concepts and principles from the object-based modeling (OOM) paradigm, specifically applied to social science domains. We will start with a crash-course on programming Java -- and continue with fun problems that will “bake your noodle” (to quote The Matrix) -- but teach you to churn out beautiful code.

This course moves at a very fast pace. I expect you to study on your own, and to help each other. This especially applies to novice programmers. Work in pairs and you’ll be less likely to suffer.

Prerequisites or co-requisites: CSS 600 Introduction to Computational Social Science, or permission of instructor. This is also a core requirement for candidates in the

computational social science Ph.D. program, and an elective course for the CSS Certificate.

This course assumes basic foundations in computing and some programming language (e.g., BASIC, Fortran, or SAS code). *Neither* knowledge of object-oriented programming (OOP) or advanced mathematics are required. A sample of specific desirable programming skills is explained in Barker (2003: 4). Basic computing skills include understanding simple data types (integer, floating point, string, etc.); variables and their scope (including the notion of global data); control flow (IF-THEN-ELSE statements, FOR/DO/WHILE loops, etc.); what arrays are and how to use them; the notion of a function/subroutine/subprogram: how to pass data in and get results back out (Barker 2003: 4). These programming assumptions will be discussed the first week.

On the first day of class, you will be given a pre-test that will place your knowledge of major programming concepts. The goal is to both understand your abilities and to tailor the course to be neither “over-the-head” nor too easy for all.

Speaking of “too easy”... Don’t count on it :-)

Learning Objectives and Grading

I’m an easy grader. Way too easy. I don’t care what you do on a test or a homework -- I care that you learn. Learning your way with computers is like learning to play guitar -- it requires practice. My guitar teacher can show me where the notes are on the fret-board, but if I want to sound like B.B. King I can only sit there for hours and play the blues.

This course will make you practice, a lot. Some of you will find it too easy -- in which case I’ll either find new ways to challenge you, or ask you to teach what you do to others.

Homeworks

There will be something due every week. Mostly incremental steps building from previous week.

If your code works and is not too messy, you’ll get an **A** and some comments on how to make it better. If it doesn’t, I’ll point out ways to fix it. When you do figure it out, I’ll give you an **A** anyway (because you learned the material -- isn’t that the point?)

Warning 1: you can’t get away with procrastination. You’ll be swamped with work and then you’ll be *eaten by the grue*¹. Also, you cannot get away with not finishing an assignment -- the next assignment will build upon it and you’ll suffer twice as much.

¹ gratuitous references to nerd folklore and ancient computer games are par for the course ;-)
If you don’t know what a grue is, google it.

Warning 2: Bad grades are reserved for people that do not work diligently week to week and do not show improvement.

Final Project

This will be your crowning achievement and your main grade contribution, and the source of your headaches for the next 15 weeks.

We, as a collective, will undertake building an ambitious model of the world political system, one country at a time. Each of you will build a model of a country -- complete with a simple economy, peasants, natural resources, and an army -- and compete with your classmates for WORLD DOMINATION.

As the end of semester gets closer, we will provide a simulation framework that will integrate all of our simulated states into a single world-system. At the final presentations for the course, we will put all of our simulated states together, and watch in amazement as they battle to the death - or build a peaceful and harmonious world society.

What happens at the final battle is up to you.

You may work as individuals or in small groups (3 people max.). For groups, you will both receive the same grade. Also, feel free to share information about your strategies to the other groups (strategic lying is par for the course, naturally).

The grades will not depend on whether your agent takes over the world, but rather on the quality of implementation and your understanding of the modeling paradigms, as well as on the realism of your strategies.

Facebook

<http://www.facebook.com/pages/CSS-605-Fall-2011/196566293744691?sk=wall>

Please “like” this page -- we’ll be using this as an informal bulletin board

Source Control

Source control is a religion. I will teach you to use it daily, and require it of everyone. Hear the gospel, it will save your a** later.

The repository for the class is here:

<https://github.com/maksim2042/CSS605-Fall2011>

LEARN IT, USE IT!

Schedule of Study: Calendar, Topics, Assignments

Schedule for this course will change a lot, depending on (a) who is here, (b) how well people are doing. I will adjust topics and assignments to make sure that everyone is challenged but nobody is dying of exhaustion.

Books

Head First Java --> excellent intro book from O'Reilly. If you've never programmed, get it, and do the exercises.

Head First Object Oriented Modeling --> same thing for OOM

How to think like a computer scientist --> excellent NEW textbook. FREE e-book.

Tools:

NetBeans: <http://netbeans.org/downloads/index.html> (download the Java SE version)

Some graphics program (OmniGraffle, Visio) (NOT UML software -- it's too smart for its own good and it won't teach us what we need to learn).

Paper and pencil (seriously. Get yourself a Moleskine and a good drafting pencil)

Plagiarism and Group Work

Our software is open-source. Everything done in the final project for the will be under provisions of the BSD license, meaning -- you can use other people's work if (a) you do not pass it off as your own, and (b) give credit and attribution to the author.

In homeworks, the rule of thumb is different -- I want you to do them on your own or (better) in pairs, and we will bake them FROM SCRATCH. We will use NO code from anyone else, and will even avoid the standard libraries. Why? I want you to learn how it's done -- not how to cut-n-paste some examples from a book and get it to work by some voodoo.