CS 201 – Web Programming
Course Syllabus and Calendar – Winter 2005

Professor Don Colton

Brigham Young University Hawaii

1 Course Overview

By the end of this semester, you will be writing useful web programs using databases. A web program is one that displays information on your browser and lets you fill in the blanks or press buttons to make requests. A database lets you store and retrieve a large amount of information.

For example, you could write a web program to key in your CD collection and later key in a song title and have the computer tell you which CD it is on. But you probably have them downloaded to your MP3 player already.

Or you could make a database for a used textbook business, where you key in title, price, owner, and phone number. Maybe you could collect a commission from every sale.

In this class you will learn valuable technical skills to help you become the master of the computers in your life. You will learn to write useful (not just toy) programs. You will find out what it takes. Maybe you will love it and go on to be a world-class programmer. Maybe you will write an occasional program to solve a need. Maybe you will just ask someone else, but you will know what to ask and expect.

This class teaches you to write programs in Perl for the Web using SQL databases at an intermediate level of programming proficiency.

Perl is a popular scripting language. It is very powerful, has been around for a long time, and broadly available. It is even free. Amazon.com, one of the most famous web sites in history, was originally written in perl. (For speed, many parts of amazon.com have been rewritten in faster languages.) Perl makes things easy for the programmer, at the expense of being slower than C and C++.

The **Web** means your favorite browser, such as Microsoft Internet Explorer, Mozilla Firefox, Opera, or even Lynx. You will create web pages dynamically, using forms, checkboxes, radio buttons, fill in the blanks, and press "submit." You will learn to create those web pages, and respond to the data entered into them. We will use CGI, the common gateway interface standard, as our processing method.

SQL, structured query language, is the international standard for database access. You will be introduced to the basic commands for storing data, retrieving data, and updating data. We will use MySQL, a powerful and inexpensive implementation of SQL, as our database tool.

It's all free. We will not tie you down to any proprietary and expensive languages and interfaces. Instead we will learn things that you can take with you right now and use on your first job without needing the chief financial officer to write a big check to get you started. You can run these tools at home or generally any place in the world.

But what if your next job requires you to use Microsoft SQL? Or Microsoft ASP? No problem. The tools and design methods you learn will prepare you to understand and work with specialized tools that may be in use elsewhere. Those of you that drive "stick" will realize that it is easy to drive a car with an automatic transmission. Those of you that only drive an "automatic" may find it is difficult to drive "stick." That is why we are teaching you to drive "stick" in this class.

Programming proficiency means that you and get something done without a lot of help. If you always need help for everything, you are not proficient, and you are a drag on your organization. You will need to be a contributor. In this class you need to be able to write decent code on the midterms and final exam.

By the time you finish this course, you will be programming web pages. Not just HTML, but Forms and server-side CGI, relational data base entry, maybe even games. You will have some marketable skills. If you are a Math major or an Information Systems major, completion of this course will

constitute "programming proficiency" for your major. IS and Math are not all about programming, but those without programming skills are at a substantial disadvantage in the job market. For Math Ed, a little programming skill can let you develop custom math-based learning games on the computer.

For **Computer Science** majors, or other students who want more than an intermedate level of programming proficiency, there is yet more to learn. This course is just a stepping stone.

Prerequisites: The common prerequisite is CS 101 (Beginning Computer Programming). In this case, I assume that you have written some programs. You probably know how to do formatted printing. You can use **if**, **else**, **while**, **do while**, **for**, subroutines, and arrays. You are no longer clueless, but you are not yet confident.

Alternately, you can take the class if you have already completed some Calculus. Your level of mathematical maturity should allow you to learn the prerequisite material during our review (the first three weeks of class).

If you do not have prior Calculus or programming experience, you should confer with the instructor to make sure this is the right class for you.

1.1 The Course

- **Course Number:** CS 201
- **Title:** Web Programming
- **Course Description:** Review of CS 101. Problem solving, stacks, queues, hash tables, mathematical analysis of algorithms, regular expressions. Web programming including CGI and database. (Prerequisites: CS 101 and Math 106 or higher.)
- **Textbook (highly recommended):** *Perl by Example*, by: Ellie Quigley. ISBN 0-13-028251. SRP new $44.99. Used about $35.
- **Section 1:** Class Time: MWF 9:00–9:50 AM Final Exam: Mon 18 Apr, 7:00–10:00 AM Classroom: GCB 140
- **Section 2:** Class Time: MWF 3:00–3:50 PM Final Exam: Mon 18 Apr, 3:00–6:00 PM Classroom: GCB 140

1.2 The Instructor

- **Instructor (me):** Don Colton
- **My email:** don@colton.byuh.edu
- **My Office:** GCB 130 B
- **Office Hours:** MWF 10 AM to 11 AM

1.3 The Tutors

- **T.A. Hours:** Mon–Thu, 3 PM to 11:30 PM **
- ** May stay open til 1 AM based on demand.
- **T.A. Hours:** Fri, 3 PM to 8 PM
- **T.A. Hours:** Sat, 3 PM to 6 PM
- **T.A. Location:** GCB 101 (CS Lab)

1.4 Open-Door Policy

My office hours are shown above. You can contact me by email to make an appointment at another time. I also have an open-door policy: **If my door is open (even just a bit) feel free to knock and come in.**

1.5 Grading Overview

Your final grade will be computed using several methods, and the highest grade will be yours. Your total-points grade is based on 1000 points of assigned work. Some extra credit points may be available. The total-points grading will be as follows:

930+	A	900–929	A–	870–899	B+
830–869	B	800–829	B–	770–799	C+
730–769	C	700–729	C–	670–699	D+
630–669	D	600–629	D–	0–599	F

Method 1: Normal This method is based to a larger extent on your activity in the class. Major points are earned by "safety net" activities such as in-class practices. There is a penalty for late work.

pgm	programming labs	400 (430) pts
prac	in-class practices	150 (165) pts
qic	in-class quizes	150 pts
qtc	testing center quizes	150 (180) pts
final	final exam (in class)	150 pts
tot	total points assigned	1000 (1075) pts

The notation "400 (430) pts" means that up to 400 points go toward your final grade, but 430 points are actually available to be earned. The extra 30 points a life line. You can lose up to 30 points in that category and still get full credit. If you have more than 400 points, then you have a perfect score in that area, but the extra points do not carry over into other areas.

Method 2: Genius This method is suited to those geniuses that do really well on the labs and the final, and find class boring. The in-class practices are eliminated and the points redistributed.

pgm	programming labs	400 (430) pts
qic	in-class quizes	150 pts
qtc	testing center quizes	150 (180) pts
final	final exam (in class)	300 pts
tot	total points assigned	1000 (1060) pts

The Final Exam: This is the part that gets people into trouble. I have been compelled to add this rule because of occasional students who do well on points but cannot perform at the end of the semester. I believe that a grade of B or A means you can perform well on request.

Therefore, you must ALSO achieve a sufficient score on the final exam, as shown in this table. Your final grade is be **limited** by both your total-points grade (above) and the grade in this table based on your final-exam percentage. Notice that these percentages are substantially more generous than the point scales shown above.

Note that an 83 is not an A on the final. It is a B, but it still allows you to get an A for the course. Similarly a 55 is not a C–. It is an F. But with it you can still earn a C– in the course.

83+	A	80–82	A–	77–79	B+
73–76	B	70–72	B–	65–69	C+
60–64	C	40–59	C–	20–39	D+
0–19	D				

By way of explanation, I have had some students who did very well on the labs, sometimes incredibly well, but did very poorly on the exams, sometimes incredibly poorly. I discovered that one such student was just very good at getting other people to help him, but never did develop the skill to do the job himself. The labs are meant to be serious learning experiences. The exams are meant to verify that the learning took place.

Grading is discussed further below.

1.6 Subject to Change

It is possible that I will revise some aspects of the course as we go along. Any changes I make are likely to be to your advantage. If any of my changes seems unfair to you, let me know. I will try to correct it.

2 Course Calendar

Lectures and in-class practices will match the assignments that are due. In-class tests will take one hour each (except the final). I have allowed an extra day (Tue, Thu, Sat instead of Mon, Wed, Fri) for many assignments, but you are encouraged to complete them earlier.

First block: During the first (approximately) twelve hours of class, we will review everything we assume you learned in your previous programming classes. We will show you how to do these same things using Perl. There is a small program due almost every day. We end with the first midterm exam.

```
: hello     10  hello world
: celsius   20  convert Fahrenheit
: tc1       10  precedence simple
: tc2       15  precedence mixed
: phonecard 25  phone card comparison
: starbox   20  draw a box of stars
: lessthan  25  how many were less
: argv      15  command line access
: ducks     25  carnival duck game
: mid1      50  midterm 1 in class
```

Second block: During the second (approximately) twelve hours of class, we focus on CGI and the web. There is a program due about every two days, but the programs are a bit harder. We also learn some regular expression processing. We end with the second midterm exam.

```
: tc3    30  regular expressions
: pick   25  pick a number
: calc   25  tutor: cgi calculator
: nuts   35  tutor: nuts, bolts, pins
: cgi1a  25  cgi input parsing (simp)
: cgi1b  25  cgi input parsing (comp)
: mid2   50  midterm 2 in class
```

Third block: During the third (approximately) twelve hours of class, we focus on DBI, the database interface, using SQL. We store, retrieve, and update information in databases. We create new tables. We build upon our CGI skills. We end with the third midterm exam.

```
: buyone    25  tutor: buyone
: manager   25  tutor: manager
: shopcart  30  tutor: shopcart
: checkout  50  tutor: checkout
: mid3      50  midterm 3 in class
```

Fourth block: During the last few hours of class, we deal with programming complexity and simple software engineering. We study "big oh" analysis. This also gives us some time when labs are not due, so those that have gotten behind have a chance to submit late work before the end of the semester. We end with the final exam.

```
: tc4     15  big oh analysis (simp)
: tc5     20  big oh analysis (comp)
: tc6     90  comprehensive exam
: final  150  final in class
```

3 Knowledge Units

The CS 201 course covers the following CC2001 Knowledge Units. These are defined in Computing Curricula 2001, a joint project of IEEE-CS and ACM. The IEEE Computer Society and the Association for Computing Machinery are the two major professional societies in computer science.

Much of this material is covered in more than one course.

PF1. Fundamental programming constructs

We review material that was introduced in CS 101.
- Basic syntax, semantics of a higher-level language
- Variables, types, expressions, and assignment
- Simple I/O
- Conditional and iterative control structures
- Functions and parameter passing
- Structured decomposition

PF2. Algorithms and problem-solving

- Problem-solving strategies
- The role of algorithms in problem-solving process
- Implementation strategies for algorithms
- Debugging strategies
- The concept and properties of algorithms

AL1. Basic algorithmic analysis

We introduce material covered more fully in CS 301.
- Asymptotic analysis of upper complexity bounds
- Big O and Θ (theta) notation
- Standard complexity classes (linear, log, exponential)

NC4. The web as an example of client-server computing

- Web technologies
  - Server-side programs
  - Common gateway interface (CGI) programs
  - Client-side scripts
- Characteristics of web servers
  - Handling permissions
  - File management
  - Capabilities of common server architectures
- Role of client computers
- Nature of the client-server relationship
- Web protocols
- Tools for web site creation and management
- Developing Internet information servers
- Publishing information and applications

IM5. Database query languages

We introduct material covered more fully in IS 350.
- Overview of database languages
- SQL (query formulation, update sublanguage)
- Embedding queries in a procedural language

4 Grading

Your grade is earned by getting points for things like completing labs and tests. Progress reports are available to you by computer at any time.

Attendance: I take roll in this class to help me learn your names. I also do some activity for points each class. Usually it is some sort of in-class practice where you demonstrate the ability to do something we have been talking about. These generally occur during the last half of the class hour. In the past I have given points for attendance, but I no longer do that because I am giving points for the in-class practices. On some days I may not have time for a practice. In that case I give full practice points to everyone that is in attendance. Missing and unnoticed persons get zeros, so be sure to alert me after class if you came in late.

Due to INS (immigration) and VA (veterans) requirements the Vice President for Student Life is supposed to be notified whenever a student misses four consecutive class days. I try to do this.

Reading: I recommend *PERL by Example* by Quigley. Student response to this book has been quite good, and it is in its third edition, suggesting

that it has sold fairly well. I continue to be interested in your feedback as students. I want to know whether you like this book.

O'Reilly's *Programming Perl* by Wall, Christiansen, and Schwartz is a much more detailed treatment of the Perl Language. Larry Wall is the primary creator of the language, so this book is very authoritative and much more complete. It is also quite well written. If your programming skills are more advanced, you may want to look at this book. I believe it is available in the BYUH bookstore in the IS 431 section.

Labs: Most of your time will be spent writing programs. I am not sure how much time it would take a good student programmer to complete all of these assignments. A professional could probably do all of them in a week. Maybe less. But you are not a professional. The work is difficult mostly because it is unfamiliar. Our task is to make it familiar, and therefore easier. You will find that assignments you did in three or four hours early in the semester can be done much more quickly later in the semester. You should feel a great sense of achievement.

Much like CS 101, you will write programs that are graded by my robotic grader, GradeBot. Class time will be devoted to understanding basic concepts of computer programming as applied to the World Wide Web.

Some of the programs will be graded by demonstration in class or for the tutor. In such cases you will have more freedom in the user interface. Programs graded by GradeBot allow freedom in the methods used to construct the answer, but they do not allow freedom in how the answer is presented: the user interface is specified for you.

GradeBot: GradeBot is my robotic program grader. It is generally available 24 hours a day, seven days a week, to grade and return your lab assignments. This is currently done via web, "turnin," or email.

For grading, GradeBot is correct and authoritative. It is your boss. It is your client. It is your Drill Sergeant. There is always a particular correct behavior that it demands. You must make your program behave in exactly the way that GradeBot is requiring (including spelling errors, if any). Be sure to look at a sample "conversation" with GradeBot before you start writing your program.

If you discover a case where you believe that GradeBot is wrong, tell me about it. If you found an error in GradeBot, I generally reward you with some extra credit. Otherwise, you must assume GradeBot is right.

Tests: There are several tests given in the testing center using bubble sheets, including one that is comprehensive and is due the last day of class. You can complete the tests as soon as you want. I allow unlimited time and scratch paper, but no books, no notes, and no calculators. For each test, a sample test is available through GradeBot for you to use as a study guide. You only get one chance to take each test. (If you feel there is some special reason you should get another chance, such as sudden illness, discuss it with me.)

There are three tests given in class, plus the final. These are programming tests where you will be asked to write some particular program without the aid of notes of a computer: written by hand, graded by hand. Time is limited. Scratch paper is provided. No books, no notes, and no calculators.

Deadlines: Each assignment has a deadline. You can see these deadlines by asking GradeBot. Most lab deadlines are "soft." Before the deadline an item is worth a certain number of points (100%). After the deadline, it is worth somewhat less (usually one point) each day until it reaches maybe 60% of its original value. It then remains near the 60% level until the last day of class. All work must be completed by the end of the last day of class. The final exam has a separate deadline.

Incomplete and UW: If you quit working in the class before achieving a passing grade, I will probably give you a "UW" grade instead of an "F." A UW is worse than an F because it does not count as credits attempted.

I do not give "I" grades (incompletes) except in unusual circumstances. In my experience only a small fraction of incompletes are ever completed. I will consider giving you an incomplete if you request it, seem to have a good reason, have a pretty solid time line for completion, and you get the necessary paperwork filled out.

5 Lab Submission Rules

Cheating is sometimes a problem in this class. I have some rules. I am unhappy when I see cheating in

any class. There have been cases where one student gives a copy of their completed program to another student, and the second student keys it in, possibly with minor changes, such as changing the names of variables. In worse cases, the second student uses cut-and-paste to copy the program, or sections of it. In almost every case, the second student *does not understand how the program works,* or why the program says what it says. I consider any such behavior to be plagiarism and an honor code violation. Your goal is to learn, not merely do things that might let you complete the assignments (without learning).

Open-Neighbor versus Copying: On all labs, I allow you to *confer* with other people, including other students. You may read their code (if they let you). You may show your code to them. You may talk about your code, your approach, your difficulties, and your ideas. You may draw pictures and make analogies and ask questions. You may use their ideas. However, *you must not make a copy of their code or submit their code to GradeBot, even if you first modify it.*

PLEASE: Never let another student take, borrow, or keep a copy of any program you wrote for this class. You can look at it *together*. If it is printed, please look at it away from any computers. If it is online, look at it on the author's own screen. Never bring up a window on the second student's screen so they can look at the first student's program. Never give a disk or email a copy. You can talk about what the program does, and why it is that way. Do NOT leave them with a copy of your program.

If you receive a copy of a program from someone, and use it as the basis for the program you are submitting, you are **cheating**.

6 Special Needs

Brigham Young University Hawaii is committed to providing a working and learning atmosphere, which reasonably accommodates qualified persons with disabilities. If you have any disability that may impair your ability to complete this course successfully, please contact the students with Special Need Coordinator, Leilani A'una at 293-3518. Reasonable academic accommodations are reviewed for all students who have qualified documented disabilities. If you need assistance or if you feel you have been unlawfully discriminated against on the basis of disability, you may seek resolution through established grievance policy and procedures. You should contact the Human Resource Services at 780-8875.

7 Preventing Sexual Harassment

Title IX of the education amendments of 1972 prohibits sex discrimination against any participant in an educational program or activity that receives federal funds, including Federal loans and grants. Title IX also covers student-to-student sexual harassment. If you encounter unlawful sexual harassment or gender-based discrimination, please contact the Human Resource Services at 780-8875 (24 hours).