CMSI 370-01

Interaction Design

http://myweb.lmu.edu/dondi/fall2014/cmsi370

Fall 2014—Doolan 222, TR 3–4:15pm, 3 semester hours
Office Hours: TR 2–3pm, T 9–10:30am, W 3–5pm,
R 4:30–6pm, or by appointment

John David N. Dionisio, PhD
email: dondi@lmu.edu
Doolan 106; (310) 338-5782

Objectives and Outcomes

This course explores the computer science subfield of *interaction design* (IxD), a.k.a. *computer-human* (or *human-computer*) *interaction* (CHI/HCI). IxD seeks to understand human behavior when interacting with computing systems and studies metrics, techniques, and theories for achieving effective interaction. Long after you finish this course, my hope is that you will be able to:

1. **Appreciate and express the art and science of interaction design, including its theories, principles, methodologies, and role in software design and development.**
2. **Understand and report on how humans behave and interact with the user interfaces of real-world systems and software.**
3. **Demonstrate the fundamentals behind designing and implementing user interfaces.**

In addition to the course-specific content, you are also expected to:

4. **Follow disciplinary best practices throughout the course.**

Prerequisites/Prior Background

Intermediate to advanced proficiency in any programming language is *very* helpful. Concurrent or prior taking of CMSI 386 Programming Languages provides exposure to common language concepts with varying syntax. Some material in this course carries directly into CMSI 371 Computer Graphics.

Materials and Texts

- Ben Shneiderman and Catherine Plaisant. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, 5th Edition, Addison Wesley/Pearson, 2009.
- Jakob Nielsen. *Usability Engineering*, Morgan Kaufmann, 1994.
- Donald A. Norman. *The Design of Everyday Things*, Basic Books, 2002.
- Assorted handouts, articles, and sample code to be distributed throughout the semester.

The following text, especially Chapters 6–8, can serve as a programming tutorial and reference:

- John David N. Dionisio and Ray Toal. *Programming with JavaScript: Algorithms and Applications for Desktop and Mobile Browsers*, Jones & Bartlett Learning, 2011.

In addition, do not hesitate to look for further information regarding the concepts, techniques, tools, and paradigms that we will discuss.

Course Work and Grading

This course uses standards-based grading: your proficiency in each course objective is directly evaluated according to the outcomes shown on page 4 of this syllabus. Proficiency is measured according to the following key:

+	Advanced proficiency
|	Appropriate proficiency
/	Approaching appropriate proficiency
–	Needs practice and support
O	Not yet evaluated

Your submitted work is used to evaluate these outcomes. Letter grades are then assigned as follows:

	+	|	/	–
A	**many**		*none*	*none*
B		**many**	**few**	*none*
C			**some**	**few**
D				**some**
F				**many**

A–, B+, B–, C+, and C– grades are assigned when there are "close calls" between the above thresholds. Qualitative considerations (e.g., degree of difficulty, effort, class participation, time constraints, overall attitude) may improve proficiency measures. You will receive feedback and proficiency updates after every assignment.

Term Portfolio

Your accumulated writings and software for the semester comprise the *term portfolio*—the final, definitive artifact that demonstrates the proficiencies you have reached for each course outcome. It is how you show that you have, indeed, accomplished the objectives of this course.

The final proficiency for any given outcome is the statistical mode over the set of assigned proficiencies. Ties are broken through qualitative assessment by the instructor. Incomplete portfolios are evaluated on a case-to-case basis.

Formal Written Work

Your portfolio will include at least three (3) pieces of formal written work of varying length:

- A report on a hands-on usability study
- A research paper on an interaction design topic chosen from an instructor-provided list
- A user interface design "vision document"

Your work here determines your proficiencies for outcomes *1a*, *1b*, *2a*, *2b*, and *4d–4f*.

Programming Work

The second major type of work in your portfolio is software. Programming work includes:

- User interface construction and implementation
- Front-end development "on top of" a functional application back end
- Implementation of reusable user interface components and behaviors

Your work here determines your proficiencies for outcomes *3a*, *3b*, and *4a–4f*.

Resubmitting Work for Re-evaluation: Once Within Two Weeks of Feedback

Standards-based grading focuses on achieving proficiency, not accumulating scores. Thus, portfolio work may be resubmitted for re-evaluation *once within two weeks of receiving feedback* on it. I have no way of automatically knowing when you're finished with something, so *please notify me by email* when a resubmission is ready.

You must still submit all portfolio work by their respective deadlines—late work detracts from outcome *4f*. An assignment's number is its due date in *mmdd* format. Resubmissions for work whose feedback arrives less than two weeks before the end of the semester must be received by December 12.

Version Control

Version control is an indispensable part of today's computer science landscape in industry, the academe, and the open source community. We use version control heavily in this course: make sure that you get the hang of it.

Workload Expectations

In line with LMU's *Credit Hour Policy*, the workload expectation for this course is that for every one (1) hour of classroom instruction (50 scheduled minutes), you will complete at least two (2) hours of out-of-class work each week. This is a 3-unit course with 3 hours of instruction per week, so you are expected to complete $3 \times 2 = 6$ hours of weekly work outside of class.

Attendance

Attendance at all sessions is expected, but not absolutely required. If you must miss class, it is your responsibility to keep up with the course. The last day to add or drop a class without a grade of W is August 29. The withdrawal or credit/no-credit deadline is October 31.

Academic Honesty

Academic dishonesty will be treated as an extremely serious matter, with serious consequences that can range from receiving no credit to expulsion. It is never permissible to turn in work that has been copied from another student or copied from a source (including the Internet) without properly acknowledging the source. It is your responsibility to make sure that your work meets the standard of academic honesty set forth in the *LMU Honor Code and Process*.

Special Accommodations

Students with special needs who require reasonable modifications or special assistance in this course should promptly direct their request to the Disability Support Services (DSS) Office. Any student who currently has a documented disability (ADHD, autism spectrum, learning, physical, or psychiatric) needing academic accommodations should contact DSS (Daum 224, x84216) as early in the semester as possible. All requests and discussions will remain confidential. Please visit *http://www.lmu.edu/dss* for additional information.

Topics and Important Dates

Correlated outcomes are shown for each topic. Specifics may change as the course progresses. University dates (italicized) are less likely to change.

August	Background and history of interaction design *(1a, 2a)*; version control setup *(4e)*
August 29	*Last day to add or drop a class without a grade of W*
September	Usability metrics *(1b, 2a, 2b)*; guidelines, principles, and theories *(1b, 2a, 2b)*; introduction to modern web apps *(3a, 4a–4e)*
October	Overview of interaction styles *(1b, 2b)*; menus, forms, and dialogs *(1b, 2b)*; implementation in HTML/CSS/JavaScript *(3a, 3b, 4a–4e)*
October 31	*Withdraw/credit/no-credit deadline*
November	Direct manipulation *(1b, 2b)*; affordances *(1a, 1b, 2b)*; implementation in HTML/CSS/JavaScript *(3a, 3b, 4a–4e)*
November 26–28	*Thanksgiving; no class*
December	Portfolio improvement workshops *(1a–4e)*; miscellaneous topics *(varies; time permitting)*
December 12	*Final term portfolios due*

You can view my class calendar and office hour schedule in any iCalendar-savvy client. Its subscription link can be found on the course web site (it's too long to provide in writing).

Tentative Nature of the Syllabus

If necessary, this syllabus and its contents are subject to revision; students are responsible for any changes or modifications distributed in class or posted to the course web site.

Course Outcomes

1 Appreciate and express the art and science of interaction design, including its theories, principles, methodologies, and role in software design and development.

1a	*Understand and express how interaction design relates to mental models.*	This is derived mainly from Don Norman's big picture view of interaction design, as explained in *The Design of Everyday Things*.
1b	*Understand and describe core interaction design concepts: usability metrics; interaction design guidelines, principles, & theories; interaction styles; and affordances & natural mappings.*	For these outcomes, "understand and describe" includes being able to list, define, explain, and give examples of relevant concepts, always with clarity, coherence, intellectual force, and stylistic control.

2 Understand and report on how humans behave and interact with the user interfaces of real-world systems and software.

2a	*Conduct and document a real-world study of how a cohort of users responds to a particular user interface, including but not limited to capturing and prioritizing usability metrics and correlating results to mental models and interaction design theories.*	One such study will be "hands-on" and experimental—you'll do it yourself, then report on the results. Another assignment will be more research-oriented, where you will be given a selection of interaction design topics to investigate in the literature (and optionally take on directly).
2b	*Effectively use: usability metrics; interaction design guidelines, principles, & theories; interaction styles; and affordances & natural mappings to make appropriate, well-founded interaction design decisions.*	Such decisions include user interface analysis, diagnosis of interaction design problems, evaluation or comparison of user interfaces, choosing interaction styles, and envisioning new user interface designs. Such choices or decisions must also be clearly explained or justified.

3 Demonstrate the fundamentals behind designing and implementing user interfaces.

3a	*Know and understand how user interfaces are constructed, especially the model-view-controller (MVC) paradigm.*	These outcomes are all demonstrated by writing programs that involve one or more of these areas. Thus, some specific set of technologies, languages, and libraries must be learned and used. However, it must also be understood that these concepts are general and technology-independent: when called for, one should be able to transfer this knowledge to other platforms.
3b	*Know and understand event-driven programming.*	

4 Follow disciplinary best practices throughout the course.

4a	*Write syntactically correct, functional code.*	Code has to compile. Code has to work. No errors, no bugs. Use unit tests as much as possible.
4b	*Demonstrate proper separation of concerns, especially MVC.*	This is the basis of good software design. It makes software easier to maintain, improve, and extend. Proper separation of concerns includes but is not limited to correct scoping of variables & functions and zero duplication of code.
4c	*Write code that is easily understood by programmers other than yourself.*	This outcome involves all aspects of code readability and clarity for human beings, including but not limited to documentation & comments, spacing & indentation, proper naming, and adherence to conventions or standards.
4d	*Use available resources and documentation to find required information.*	The need to look things up never goes away. Remember also that the course instructor counts as an "available resource," so this outcome includes asking questions and using office hours.
4e	*Use version control effectively.*	In addition to simply using version control correctly, effective use also involves appropriate commit frequency and descriptive commit messages.
4f	*Meet all designated deadlines.*	

Sample Standards Development Report

Based on these final proficiencies, the student will get a B–.

1	**Appreciate and express the art and science of interaction design, including its theories, principles, methodologies, and role in software design and development.**	
1a	*Understand and express how interaction design relates to mental models.*	|
1b	*Understand and describe core interaction design concepts: usability metrics; interaction design guidelines, principles, & theories; interaction styles; and affordances & natural mappings.*	+
2	**Understand and report on how humans behave and interact with the user interfaces of real-world systems and software.**	
2a	*Conduct and document a real-world study of how a cohort of users responds to a particular user interface, including but not limited to capturing and prioritizing usability metrics and correlating results to mental models and interaction design theories.*	|
2b	*Effectively use: usability metrics; interaction design guidelines, principles, & theories; interaction styles; and affordances & natural mappings to make appropriate, well-founded interaction design decisions.*	|
3	**Demonstrate the fundamentals behind designing and implementing user interfaces.**	
3a	*Know and understand how user interfaces are constructed, especially the model-view-controller (MVC) paradigm.*	+
3b	*Know and understand event-driven programming.*	/
4	**Follow disciplinary best practices throughout the course.**	
4a	*Write syntactically correct, functional code.*	|
4b	*Demonstrate proper separation of concerns, especially MVC.*	+
4c	*Write code that is easily understood by programmers other than yourself.*	|
4d	*Use available resources and documentation to find required information.*	|
4e	*Use version control effectively.*	|
4f	*Meet all designated deadlines.*	/

Totals

+	3
|	7
/	2
–	0
O	0

This student reached appropriate proficiency in 7 out of the 12 outcomes. Advanced proficiency was reached in 3 out of 12, but proficiency was not reached in 2 out of 12.

The student is a "close call" between a B and B–, but never managed to submit anything on time. The habitual lateness becomes a determining factor for going with a B– rather than a B.

The student would need more **+**'s for a B or B+. More **/**'s would have taken the student to a C-level grade. A combination of many more **+**'s and zero **/**'s is required to qualify for an A-level grade.