Objectives and Outcomes

This course explores the computer science subfield known as interaction design (IxD), also called computer-human (or human-computer) interaction (CHI/HCI). IxD is concerned with how human beings interact with computing systems, provides measures for its effectiveness, and explores techniques and theories for achieving effective interaction. Long after the course concludes, my hope is that you will:

1. Know and understand the art and science of interaction design, particularly its first principles and key metrics.
2. Apply this knowledge by studying, comparing, and evaluating the user interfaces of actual systems.
3. Know the fundamentals behind programming and implementing user interfaces, with working knowledge of technologies such as HTML/CSS/JavaScript, Cocoa/ Objective-C, and OpenGL/GLUT.

While there are no absolute prerequisites to this course, intermediate to advanced programming proficiency in any language will be helpful. Students concurrently taking CMSI 386 Programming Languages will benefit from exposure to common language concepts with varying syntax. Some of the material in this course carries directly into CMSI 371 Computer Graphics.

Materials and Texts

- Assorted handouts, articles, and sample code to be distributed throughout the semester.

The following text is recommended and not required, but it will fill in some details:


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 |
|-------------------------|---------------------------|----------------------------------------|---------------------------------|-------------------------|
| A                       | many                      | none                                   | none                            | none                    |
| B                       | many                      | none                                   | none                            | none                    |
| C                       | some                      | none                                   |                                | some                    |
| D                       |                           | some                                   |                                | many                    |

A–, B+, B–, C+, and C– grades may be assigned based on “close calls” along the proficiency measure thresholds and qualitative considerations such as degree of difficulty, effort, class participation, time constraints, and overall attitude throughout the course. You may inquire at any time about the proficiency measures that I currently have on record for you.
Homework
Homework consists of questions, exercises, and programming assignments to be given throughout the semester. Homework is where you can learn from your mistakes without penalty. It is meant to develop proficiency, not demonstrate it. Thus, homework is optional.

Of course, with great flexibility comes great accountability. By doing homework (and paying attention to the feedback I provide), you will know precisely how you are doing in class. This is because feedback on submitted homework is standards-based: it is evaluated according to the same proficiency scale as your final grade. Students who submit homework receive a clear, up-to-date picture of their current strengths and weaknesses. By submitting homework, you will know what you need to work on, and what you have already learned sufficiently.

To receive this feedback in a timely fashion, submit your homework by the designated deadline. This deadline is always the beginning of class on the designated due date; the due date is encoded in the homework number. Quid pro quo: submissions after the deadline may eventually get feedback, but its timing is not guaranteed.

Term Portfolio
Your accumulated homework for the semester comprises the term portfolio — the final, definitive artifact that demonstrates the proficiencies you have reached for each course outcome. The term portfolio provides you with an opportunity to improve upon the homework submitted throughout the semester; it is how you show that you learned from your mistakes or improved upon already established knowledge.

After receiving feedback on your homework, you are encouraged to improve your work based on that feedback, and show it to me for re-evaluation. Improvements in proficiency are recorded and give you a good idea of how your term portfolio will fare long before its final version is submitted.

The final version of your term portfolio is due on December 15. Late portfolios will not be accepted.

Extra Credit
In terms of standards-based grading, “extra credit” takes on a different meaning: it indicates work that, if successfully performed, would indicate advanced proficiency (+). Extra credit tasks may be assigned for either homework or the term portfolio. Accomplish them successfully to rack up those +’s. You do not need to perform extra credit work to show advanced proficiency; it merely demonstrates such proficiency more readily.

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

Attendance
Attendance at all sessions is expected, but not absolutely required. If you must miss one or more class sessions, 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 September 2. The withdrawal or credit/no-credit deadline is November 4.

University Policy on Academic Honesty
Loyola Marymount University expects high standards of honesty and integrity from all members of its community. All students are expected to follow the LMU Honor Code and Process, as stated in the LMU Undergraduate Bulletin.

Americans with Disabilities Act
Students with special needs as addressed by the Americans with Disabilities Act who need reasonable modifications, special assistance, or accommodations in this course should promptly direct their request to the Disability Support Services (DSS) Office. Any student who currently has a documented disability (physical, learning, or psychological) needing academic accommodations should contact DSS (Daum Hall, Room 224, x84535) as early in the semester as possible. All 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.

<table>
<thead>
<tr>
<th>Month</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>September</td>
<td>Background and history of interaction design (<strong>1a–1b, 1f, 2a</strong>); usability metrics (<strong>1c, 2b, 2b</strong>); guidelines, principles, and theories (<strong>1d–1e, 2e–2e, 2b</strong>); introduction to HTML, CSS, and JavaScript (<strong>3a–3e, 3i–3l</strong>)</td>
</tr>
<tr>
<td>September 2</td>
<td>Last day to add or drop a class without a grade of <strong>W</strong></td>
</tr>
<tr>
<td>October</td>
<td>Overview of interaction styles (<strong>1g, 2f, 2b</strong>); menus, forms, and dialogs (<strong>1g–1b, 2f–2b</strong>); implementation in HTML/CSS/JavaScript (<strong>3a–3e, 3i–3l</strong>); and in Cocoa/Objective-C (<strong>3a–3d, 3f, 3i–3l</strong>)</td>
</tr>
<tr>
<td>November</td>
<td>Direct manipulation (<strong>1g–1b, 2f–2h</strong>); affordances (<strong>1i, 2b</strong>); cognitive psychology talk (<strong>1a</strong>); implementation in HTML/CSS/JavaScript (<strong>3a–3e, 3b–3l</strong>), Cocoa/Objective-C (<strong>3a–3d, 3f, 3b–3l</strong>), and OpenGL/GLUT (<strong>3a–3d, 3g–3l</strong>)</td>
</tr>
<tr>
<td>November 4</td>
<td>Withdraw/credit/no-credit deadline</td>
</tr>
<tr>
<td>November 23–25</td>
<td>Thanksgiving; no class</td>
</tr>
<tr>
<td>December</td>
<td>Portfolio improvement workshops (<strong>1a–3l</strong>); miscellaneous topics (varies)</td>
</tr>
<tr>
<td>December 15</td>
<td>Term portfolios due</td>
</tr>
</tbody>
</table>

You can view my class calendar in any iCalendar-savvy client such as Google Calendar or Apple iCal by subscribing to:

```
webcal://www.me.com/ca/sharesubscribe/1.9392690/M2CD-5-1-5B14D70A-E341-4026-A665-D391D97E01B8.ics
```

(I know, it’s ugly; the link is also available on my web site for convenience.)

If necessary, this syllabus and its contents are subject to revision. Students are responsible for any changes or modifications announced in class.
Course Outcomes

1 Know and understand the art and science of interaction design, particularly its first principles and key metrics.

1a Know the overall history of the field, its relationship to other fields both within computer science (e.g., computer graphics) and beyond it (e.g., cognitive psychology), and its major practitioners and contributors.

1b Know Norman's big picture view of interaction design.

1c Know the definitions of the five key usability metrics.

1d Define and differentiate interaction guidelines, principles, and theories.

1e List examples of interaction design guidelines, principles, and theories.

1f Know the difference between usability and utility.

1g Know what interaction styles are, how they developed historically, their areas of applicability, their strengths, their weaknesses, and how they generally fare in the five key usability metrics.

1h Know the characteristics or features that exemplify effective implementations of each interaction style.

1i Know what affordances are, and what they contribute to interaction design decisions.

2 Apply this knowledge by studying, comparing, and evaluating the user interfaces of actual systems.

2a Map real-world cases and/or situations to Norman's big picture view of interaction design.

2b Prioritize the five usability metrics for certain categories of systems.

2c Use appropriate metrics, guidelines, principles, and theories to make an interaction design decision.

2d Analyze, evaluate, or compare systems or designs based on interaction design guidelines, principles, and theories.

2e Analyze, diagnose, or compare usability problems using interaction design guidelines, principles, and theories.

2f Choose and justify the most appropriate interaction style for a given application.

2g Assess or compare the quality of user interfaces based on the guidelines and principles for a given interaction style.

2h Envision new or innovative user interface designs based on usability metrics, interaction design guidelines, principles, theories, and affordances.

3 Know the fundamentals behind programming and implementing user interfaces, with working knowledge of technologies such as HTML/CSS/JavaScript, Cocoa/Objective-C, and OpenGL/GLUT.

3a Know what event-driven programming is at a general, non-platform-specific level.

3b Know how user interfaces are constructed at a general, non-platform-specific level.

3c Know what the model-view-controller (MVC) paradigm is, in general, non-platform-specific terms.

3d Know the major types of events, the information they provide, and how they are typically handled.

3e Write syntactically correct, functional web browser implementations of the major interaction styles.

3f Write syntactically correct, functional Cocoa/Objective-C implementations of the major interaction styles.

3g Write syntactically correct, functional C programs using OpenGL and GLUT.

3h Break down a high-level user action into a sequence of lower-level user or system events.

3i Demonstrate proper separation of concerns, especially MVC.

3j Provide clear, appropriate inline documentation (i.e., comments).

3k Write code that is properly indented and spaced for human readability.

3l Use available resources and documentation to find any required technical or developer information.
Sample Standards-Based Evaluation

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

1. **Know and understand the art and science of interaction design, particularly its first principles and key metrics.**
   
   1a. Know the overall history of the field, its relationship to other fields both within computer science (e.g., computer graphics) and beyond it (e.g., cognitive psychology), and its major practitioners and contributors.
   
   1b. Know Norman's big picture view of interaction design.
   
   1c. Know the definitions of the five key usability metrics.
   
   1d. Define and differentiate interaction guidelines, principles, and theories.
   
   1e. List examples of interaction design guidelines, principles, and theories.
   
   1f. Know the difference between usability and utility.
   
   1g. Know what interaction styles are, how they developed historically, their areas of applicability, their strengths, their weaknesses, and how they generally fare in the five key usability metrics.
   
   1h. Know the characteristics or features that exemplify effective implementations of each interaction style.
   
   1i. Know what affordances are, and what they contribute to interaction design decisions.

2. **Apply this knowledge by studying, comparing, and evaluating the user interfaces of actual systems.**
   
   2a. Map real-world cases and/or situations to Norman's big picture view of interaction design.
   
   2b. Prioritize the five usability metrics for certain categories of systems.
   
   2c. Use appropriate metrics, guidelines, principles, and theories to make an interaction design decision.
   
   2d. Analyze, evaluate, or compare systems or designs based on interaction design guidelines, principles, and theories.
   
   2e. Analyze, diagnose, or compare usability problems using interaction design guidelines, principles, and theories.
   
   2f. Choose and justify the most appropriate interaction style for a given application.
   
   2g. Assess or compare the quality of user interfaces based on the guidelines and principles for a given interaction style.
   
   2h. Envision new or innovative user interface designs based on usability metrics, interaction design guidelines, principles, theories, and affordances.

3. **Know the fundamentals behind programming and implementing user interfaces, with working knowledge of technologies such as HTML/CSS/JavaScript, Cocoa/Objective-C, and OpenGL/GLUT.**
   
   3a. Know what event-driven programming is at a general, non-platform-specific level.
   
   3b. Know how user interfaces are constructed at a general, non-platform-specific level.
   
   3c. Know what the model-view-controller (MVC) paradigm is, in general, non-platform-specific terms.
   
   3d. Know the major types of events, the information they provide, and how they are typically handled.
   
   3e. Write syntactically correct, functional web browser implementations of the major interaction styles.
   
   3f. Write syntactically correct, functional Cocoa/Objective-C implementations of the major interaction styles.
   
   3g. Write syntactically correct, functional C programs using OpenGL and GLUT.
   
   3h. Break down a high-level user action into a sequence of lower-level user or system events.
   
   3i. Demonstrate proper separation of concerns, especially MVC.
   
   3j. Provide clear, appropriate inline documentation (i.e., comments).
   
   3k. Write code that is properly indented and spaced for human readability.
   
   3l. Use available resources and documentation to find any required technical or developer information.