Objectives and Outcomes
This course explores the computer science subfield of computer graphics—the study and development of algorithms for synthesizing, manipulating, and displaying visual information. Long after the course concludes, my hope is that you will be able to:

1. Represent, model, and create visual information digitally.
2. Manipulate and display visual information in 2D and 3D.
3. Use and develop computer graphics APIs in both 2D and 3D.

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

4. Follow academic and technical best practices throughout the course.

Prerequisites/Prior Background
Mastery of a programming language such as JavaScript, Java, or C; expert knowledge of data structure and algorithm design; some familiarity with object-oriented programming, computer hardware, and operating systems.

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

The following text is of general use, with Chapter 9 pertaining specifically to graphics on the web:

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 |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

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.
Resubmitting Work for Re-evaluation
Standards-based grading focuses on achieving proficiency, not accumulating scores. Thus, within reason, work may be resubmitted for re-evaluation throughout the semester. You must still submit all assignments by their respective deadlines—late work detracts from outcome 4f. An assignment’s number is its due date in mmdd format.

Term Portfolio
Your accumulated assignments for the semester comprise 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 polish the work done throughout the semester; it is how you show that you learned from your mistakes or improved on already established knowledge.

The final version of the term portfolio is due on May 9. Late portfolios detract from outcome 4f. Incomplete portfolios are evaluated on a case-to-case basis.

2D Graphics
Your portfolio will include the following 2D graphics programs:
- Exercises using canvas
- A tweened, 2D animated scene
- Simple image processing filters
- Exercises in implementing graphics primitives

Your work here affects your proficiencies for outcomes 1a, 2a, 2c, 3a–3c, and 4a–4f.

3D Graphics
The second major type of work in your portfolio involves 3D graphics. This work includes:
- A 3D object library with polygon meshes, object composition, and instance transformations
- Vector and matrix libraries with commonly-used operations and transformations
- An interactive 3D scene based on these libraries

Your work here affects your proficiencies for outcomes 1b, 1c, 2a, 2b, 2d, 3a, 3d, 3e, and 4a–4f.

Details on both types of assignments will be provided later. They will all have an initial due date, after which subsequent versions may be submitted. The term portfolio will hold the final versions of these assignments.

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.

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 January 17. The withdrawal or credit/no-credit deadline is March 21.

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.

| January    | 2D graphics with JavaScript and the canvas element (1a, 2a); introduction to animation (1c, 3a) |
| January 17 | Last day to add or drop a class without a grade of W |
| February   | Graphics and memory (1a); 2D graphics primitives (3b, 3c); 2D and 3D graphics with JavaScript and WebGL (1b, 1c, 2a, 2c); introduction to programmable shaders (3e) |
| March      | Object modeling (1b, 1c); transforms (2a, 3b, 3d); viewing and projection (2b, 3d); clipping and hidden surface removal (2d) |
| March 3–7  | Spring break; no class |
| March 21   | Withdraw/credit/no-credit deadline |
| April      | Lighting and shading (2c, 3e); portfolio workshops (1a–3e) |
| April 16–18| Easter break; no class |
| May 9      | 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).

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

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. **Represent, model, and create visual information digitally.**
   - 1a. ...in terms of pixels and geometric primitives.
   - 1b. ...in terms of polygon meshes: vertices, edges, and faces.
   - 1c. ...as a composition of multiple discrete objects (scenes).

   With a few exceptions, these outcomes will be demonstrated within a single, cumulative “scene” program throughout the semester. More than in prior courses, assignments will incrementally build on previous ones. It will thus be more important than usual that you keep up and keep current.

2. **Manipulate and display visual information in 2D and 3D.**
   - 2a. Apply transforms to 2D and 3D objects.
   - 2b. Project 3D objects onto a 2D viewport.
   - 2c. Perform color and light computations.
   - 2d. Perform clipping and hidden surface removal (HSR).

   In the same way that the study of general-purpose data structures starts with the structures themselves, then goes into algorithms and operations on those structures, so goes the study of computer graphics. Learning objective 1 looks at structure; learning objective 2 looks at algorithms and computations.

3. **Use and develop computer graphics APIs in both 2D and 3D.**
   - 3a. Animate scenes in 2D and 3D.
   - 3b. Implement 2D graphics primitives such as line segments, circles, and polygon fills.
   - 3c. Perform bit-level color manipulation.
   - 3d. Develop a library of geometric primitives, operations, and matrix transformations.
   - 3e. Render a 3D scene using programmable shaders.

   There is some overlap between these outcomes and the ones for learning objective 2. This is by design—the outcomes in objective 2 focus on understanding these computations conceptually, including doing them manually; the outcomes in objective 3 look at your ability to implement them in code.

4. **Follow academic and technical best practices throughout the course.**
   - 4a. Write syntactically correct, functional code.
   - 4b. Demonstrate proper separation of concerns.
   - 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.

   Code has to compile. Code has to work. No errors, no bugs. Use unit tests as much as possible.

   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.

   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.

   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.

   In addition to simply using version control correctly, effective use also involves appropriate commit frequency and descriptive commit messages.
### Sample Standards Achievement Report

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

1. **Represent, model, and create visual information digitally.**
   - 1a. …in terms of pixels and geometric primitives.
   - 1b. …in terms of polygon meshes: vertices, edges, and faces.
   - 1c. …as a composition of multiple discrete objects (scenes).

2. **Manipulate and display visual information in 2D and 3D.**
   - 2a. Apply transforms to 2D and 3D objects.
   - 2b. Project 3D objects onto a 2D viewport.
   - 2c. Perform color and light computations.
   - 2d. Perform clipping and hidden surface removal (HSR).

3. **Use and develop computer graphics APIs in both 2D and 3D.**
   - 3a. Animate scenes in 2D and 3D.
   - 3b. Implement 2D graphics primitives such as line segments, circles, and polygon fills.
   - 3c. Perform bit-level color manipulation.
   - 3d. Develop a library of geometric primitives, operations, and matrix transformations.
   - 3e. Render a 3D scene using programmable shaders.

4. **Follow academic and technical best practices throughout the course.**
   - 4a. Write syntactically correct, functional code.
   - 4b. Demonstrate proper separation of concerns.
   - 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.