



1


Introduction to Computer Graphics Project Notes



This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License



Oregon State University
Mike Bailey
mjb@cs.oregonstate.edu



Project Notes 402.500.000
mjb - September 9, 2024

1

2

Why Are These Notes Here?

These notes are here to:

1. Help you setup and run your projects
2. Help you get everything in the right format for submission
3. Help you get a **better grade** by doing all of this correctly!
better grade!
better grade!
better grade!
better grade!
better grade!

mjb - September 9, 2024

2

3

Office Hours

- The TAs and I will host Office Hours on Zoom. We will set those hours as soon as we see (from the web form on the Class Resources Page) what hours people are most likely to attend.
- There are no Office Hours exclusively assigned to particular sections. *Anybody* can go to any Office Hours.
- Try different people's Office Hours. The TAs and I all have different OHs days, times, and styles. See whose OHs best fit your schedule and needs.
- If Office Hours are not working for you, let me know right away. We will find a way to make them work for you.
- We enthusiastically encourage Office Hour "**Lurkers**". That is, it is OK to come to OHs without any specific questions and just listen to the other questions and answers. Because we will be on Zoom, you can get other things done while you listen in. I have noticed that lurking on OHs is an extremely effective way for you to catch hints and advice on doing the projects. Take advantage of this! Others will. Don't be left out.

mjb - September 9, 2024

3

4

Running Your Projects

- As this is a computer graphics course, you need to find *somewhere* to run your programs that have graphics display hardware on them. **flip is not one of these places.**
- If you don't have access to your own graphics system, then you can use what we have at OSU. **On-campus users** will have access to Windows and Linux systems on-campus.
- **Off-campus users** can access our Citrix system. This is a good solution, but not a great solution. Your life will be smoother if you can find a local graphics system. See the next slide.

mjb - September 9, 2024

4

5

Compiling and Running Your Projects via Citrix

Citrix allows you to get remote desktop access to other systems. To put Citrix on your own machine, go to <https://citrix.com/downloads>, select your operating system, and click on **Download**

Click **Add Account** and enter your ONID email (e.g., jgraphics@oregonstate.edu)

Click on **Continue** to configure your account

Enter your ONID email and password in the dialog box, and click **Logon**

When you run Citrix, click on the **Desktops** icon at the top


Go to: <https://it.engineering.oregonstate.edu/citrix/> for more information

mjb - September 9, 2024

5

6

Windows Visual Studio Compilation Notes



If you are on your own **Windows system**, and don't have Visual Studio 2022 already, you can get it going to:

<https://azureforeducation.microsoft.com/devtools>

Click the blue **Sign In** button on the right.

Login using your onid@oregonstate.edu username and password.

I recommend you get **Visual Studio 2022 Enterprise**. Don't get Express.

Note that vscode is not a compiler. It is a way to interface to your file system.

Once you have Visual Studio, download the class file **SampleWindows.zip**, unzip it on your system, and then double-click on the **.sln** file

mjb - September 9, 2024

6

Linux Compilation Notes

If you are on your own **Linux system**, compile using g++:


The typical g++ compile sequence is:

```
g++ -o sample sample.cpp -lGL -lGLU -lglut -lm
```

Note that the second character in the sequences "-lGL", "-lGLU", and "-lglut" is the letter ell, i.e., a lower-case L. This is how you link in the **OpenGL** libraries.

Note that the second character in the 3-character sequence "-lm" is the letter ell, i.e., a lower-case L. This is how you link in the **Math** library.

Download the file **SampleLinux.tar**, un-tar it on your system (tar -xvf SampleLinux.tar), and then type **make**

 cs331 - September 9, 2014

7

Mac Compilation Notes

If you are on your own **Apple Mac system**, compile using g++:

The typical g++ compile sequence is:


```
g++ -framework OpenGL -framework GLUT sample.cpp -o sample -Wno-deprecated
```

Download the file **SampleMac.tar**, un-tar it on your system (tar -xvf SampleMac.tar), and then type **make**

The standard place to put OpenGL include files looks like this:
`#include <GL/gl.h>`

Apple changed it to this to:
`#include <OpenGL/gl.h>`

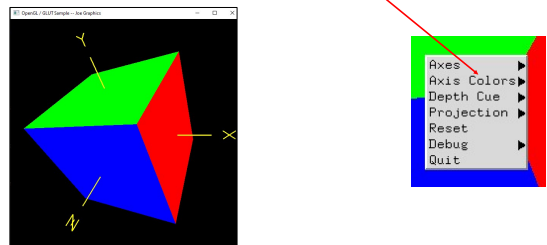
This change has been made in your sample code.

 cs331 - September 9, 2014

8

Here's What the sample.cpp Program Should Look Like

- Holding down the left mouse button and moving the mouse causes a 3D rotation
- Holding down the middle mouse button and moving the mouse causes a scaling
- Holding down the right mouse button brings up this menu:



cs331 - September 9, 2014

9

Project Turn-in Procedures

Your project turnins will all be electronic on **Canvas**. We will not be using Teach.

Your project turnins will be done via Canvas and will consist of:

1. Source files of everything (.cpp, .vert, .frag)
2. A report in PDF format.
3. No other files!

Submit these files separately. Don't zip or tar (etc...) anything!

Electronic submissions are due at 23:59:59 Pacific Time on the listed due date.

Your PDF report needs to include:

1. A title area on the first page: your name, email, project number, and project name
2. A description of what you did to get the display that you got
3. A couple of screen captures to show your program in action
4. An accessible web link to a video showing your program in action

Your project will be graded and the score posted to Canvas.

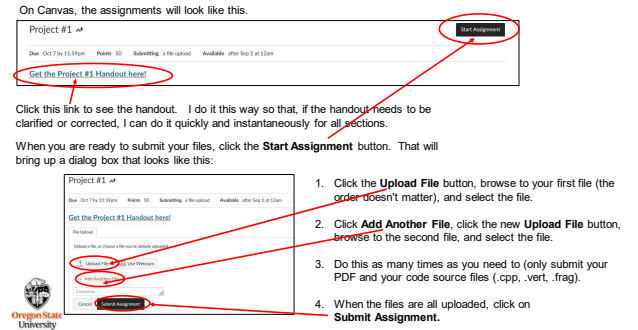
If you did not get full credit, your grade will have an attached Canvas note telling you why.

cs331 - September 9, 2014

10

How to Turn In a Project on Canvas


On Canvas, the assignments will look like this.



Click this link to see the handout. I do it this way so that, if the handout needs to be clarified or corrected, I can do it quickly and instantaneously for all sections.

When you are ready to submit your files, click the **Start Assignment** button. That will bring up a dialog box that looks like this:

1. Click the **Upload File** button, browse to your first file (the order doesn't matter), and select the file.
2. Click **Add Another File**, click the new **Upload File** button, browse to the second file, and select the file.
3. Do this as many times as you need to (only submit your PDF and your code source files (.cpp, .vert, .frag)).
4. When the files are all uploaded, click on **Submit Assignment**.


 cs331 - September 9, 2014

11

Project Videos

In order to get your project graded, you need to make a video of your program in action:

- You can use any video-capture tool you want. If you have never done this before, I recommend **Kaltura** or **Zoom**.
- You can get Kaltura instructions here: <http://cs.oregonstate.edu/~mjb/cs557/Handouts/kaltura.1pp.pdf>
- If you use Kaltura, be sure your video's permissions are set to **Unlisted**. If the permission isn't set to **Unlisted**, then we won't be able to see it and we can't grade your project. *You need to proactively do this through <http://media.oregonstate.edu> -- this is not what the default setting is.* A good way to see if this is set properly is to give the link to a friend and see if they can open the video.
- Although not required, we love it when you narrate your video so you can tell us what you did. We have found that our grading of your project is far more fair and accurate when there is narration..
- Don't make your video overly long! Show what we need to see to grade it. Do not walk us through your code!! If we want to see your code, we will look at your .cpp file.**
- Be sure that you include the accessible web-link to your video in your PDF report!

 cs331 - September 9, 2014


12


Silly Ways to Lose Points on Your Project 13

- You didn't put your name and email on the title page of the PDF report (-5)
- You submitted some other file type for your report other than a PDF (-5)
- You zipped or tarred (etc.) some of your submission files (-5)
- You didn't put a link to your video in your PDF report (-5)
- You didn't change your video permission to *Unlisted* (-5)

Basically *anything* (that isn't my fault) that makes it take longer for me to score your project is a loss of 5 points.

Aaargh! Beware:
don't do any of these!!



 400 - September 9, 2014


13

Bonus Days 14

Projects are due at 23:59:59 Pacific Time on the listed due date, with the following exception:

Each of you has been granted **5** Bonus Days, which are no-questions-asked one-day project extensions which may be applied to most projects, subject to the following rules:

1. You cannot use any Bonus Days on the Final Project Proposal, the Final Project, and the 550-only Paper Project.
2. Of the other projects, no more than **2** Bonus Days may be applied to any one of them.
3. Weekends and holidays count as "days late".
4. Really what I do is look at your turnin *date*. Your turnin date minus the due date is how many "days late" your project is.
5. The project grade is zero if you turn it in later than you have Bonus Days left to cover.
6. The project grade is zero if you turn it in more than 2 days late.


 400 - September 9, 2014

14

Bonus Days 15

To use Bonus Days on a given project:

- You don't need to let me know ahead of time.
- Turn-in promptness is measured **by date, not by time**. After 23:59:59 Pacific Time on the posted due date, **it's late!**
- *Canvas* will allow your turn-in, no matter when you do it. But we will only grade it if it is turned in ≤ 2 days after the due date.
- I run a script to identify the projects that need to have Bonus Days deducted.
- I keep my own spreadsheet of who has used how many Bonus Days. If you are not sure how many you have left, send me an email and ask.

 400 - September 9, 2014


15

Keys to Succeeding on Your Programming Projects, I
(based on me having been through this a lot with your predecessors) 16

Start early! Stop rolling your eyes at me. You don't have to *finish* early but make a small start right away. Get the project folder setup. Bring in any other files you will need. Make sure it still compiles. Once you are over the speed bump of *starting*, you will be surprised by how easy it is to do little bits of the project during those nooks and crannies of time we all have. Also, you will be surprised by how much more meaningful all those hints are that you are going to be hearing in class, Live Lecture, and Office Hours. Starting early really does smooth the path to finishing.

Lurk in Office Hours! Because they are on Zoom, you can still be doing other things, but just lurk and listen for hints or for questions you realize you need to know too.

Work through the projects a small chunk at a time! Don't try to do the whole project at once. It won't work, and you will be stuck with a big, fat, confusing debugging mess. Do something really small and get that to work. Draw an object all by itself at the origin. Use simple, trusted, objects at first (e.g., *Osusphere*). Then add another small piece. Then another. Then another. If a step doesn't work, you will know *exactly* what additional code was at fault.

 400 - September 9, 2014

16

Keys to Succeeding on Your Programming Projects, II
(based on me having been through this a lot with your predecessors) 17


Track what your program is doing! It sounds old-fashioned, but I have noticed that many of you are best at visualizing what your program is doing when you read print statements like you are reading a story. Use strategic print statements! Guessing is futile and a waste of your time. And don't use `printf` – instead, `fprintf` to `stderr`, like this:

```
fprintf(stderr, "I am now drawing to x=%f, y=%f, z=%f\n", x, y, z);
```

The text still shows up in the same place as a `printf` does, but if the program crashes, you get to see *all* the printout.

If I give you code, don't delete any of it! Amazingly, I have seen people in this class delete lines of code that I gave them because they didn't understand what they did. Don't do that. If I gave you code, I did it for a reason!

Backup your code! Something with version control, like `git` is best, but simply zip'ing or tar'ing the entire project folder, giving it a filename of the current date (e.g., Oct06b.zip), and storing it in your Google Drive or Dropbox (or something else in the cloud, not on your local machine) works wonders for when you accidentally delete multiple lines of code that used to work.

 400 - September 9, 2014

17