A1: DrawSVG

Assignment 1 Instructions

Project repository: **https://github.com/CMU-Graphics/DrawSVG**

Assignment 1 is broken into a Checkpoint and Final.

- The Checkpoint is due Feb 22nd.
- The Final is due Mar 1st.

Late days may be used on both the Checkpoint and Final, but a late day used on the Checkpoint will not extend the Final due date (See Course Info for more details). Tasks graded for the Checkpoint will not be regraded for the Final, but Final Tasks will require a working Checkpoint implementation.

This assignment is to be completed individually. You may discuss the material in general terms with your peers, but the code you submit must be entirely your own.

NOTE: Some of the concepts involved in this assignment will be discussed later in lecture. However, *you should not wait until those lectures to get started on the assignment!* Almost everything you need to know is detailed in the assignment writeup below. Also, simultaneously working on the assignment and going to lecture will enrich your experience on both sides, *i.e.*, lectures will help to clarify things you're implementing, and implementing will help to clarify ideas discussed in lecture.

Remember that making your solution public now or in the future is a violation of the academic integrity policy. We are using Github to distribute code, but **do not work in a public fork!**

Grading

Your code must run on the GHC 5xxxx cluster machines as we will grade on those machines. Do not wait until the submission deadline to test your code on the cluster machines. Keep in mind that there is no perfect way to run on arbitrary platforms. If you experience trouble building on your computer, while the staff may be able to help, but the GHC 5xxx machines will always work and we recommend you work on them if any troubles arise.

Each task will be graded on the basis of correctness. You are not expected to completely reproduce the reference solution pixel-by-pixel as slight differences in implementation strategy or even the order of floating point arithmetic will causes differences, but your solution should not be very far off. (While we provide a diff view in the application to help you debug, we will mostly be using it as a reference when we grade your code.) If you have any questions about whether your implementation is "sufficiently correct", just ask.

The assignment consists of a total of 100 pts. The point breakdown is as follows:

- Task 1 (Hardware Renderer): **5**
- Task 2 (Drawing Lines): **5**
- Task 3 (Drawing Triangles): **15**
- Task 4 (Supersampling): **15**
- Task 5 (Transforms): **10**
- Task 6 (Scaled Images): **15**
- Task 7 (Trilinear Filtering): **15**

- Task 8 (Alpha Compositing): **5**
- Task 9 (Draw Something!): **5**
- Documentation: **5**
- Writeup: **5**

For Checkpoint, please implement Tasks 1-4 along with a Writeup for these tasks. For Final, please implement Tasks 5-9, as well as updating your Writeup for these new tasks and posting Documentation.

The project description includes some additional challenges marked as extra credit. The point value next to these problems indicates a typical number of points that would be earned for successful completion, though we may award more or fewer points for these problems based on your particular level of effort. In general, we encourage you to push the envelope and explore extensions that make your program more powerful or more efficient, and we may even award additional points to you for doing so. If you do anything extra, please explain your work in the writeup! However, no matter how much extra work you do, your overall score on this assignment is capped at 110 points.

Documentation

Clear, well-written documentation is a critical part of software development. Since you (the students) are the ones who will most benefit from good documentation---or will suffer through bad documentation---we are "crowd sourcing" improvements to the course material. **What did you find confusing**---*and then finally figure out?*

As 5% of your assignment grade, you will need to submit suggested edits to the assignment documentation. These could be:

- Suggested changes or additions to the assignment writeup
- Suggested changes or additions to assignment Wiki (for A2, A3, and A4)
- Suggested changes or additions to comments in the skeleton code

Do not wait until the due date to submit these suggested edits. They will be most helpful to your classmates (and to us!) if they are submitted ahead of time, so that we can *immediately* incorporate any good suggestions into the actual course material. To submit your edits, you must therefore:

- Go on Piazza
- Find the thread with the appropriate label: `a1wiki`, `a2wiki`, `a3wiki` or `a4wiki`
- Make an *anonymous* (not private) post with your suggested edit

The TAs will monitor this label, and immediately update the docs with any/all useful suggestions. This Piazza post will also be recorded as part of your assignment grade. Note that we do **not** have to accept your edit in order for you to receive full credit on the assignment. However, we will grade your edits based on whether they appear to be a "good faith effort" to make a useful comment. In other words, did you really think about what is clear/unclear and provide a useful edit? Or did you just write something totally random at the very last minute? :-) Especially useful edits (e.g., those that provide nice insights, or point out serious bugs/errors) may receive extra credit.

Writeup

You will submit a short document explaining how you believe your implementation performs on each of the tasks. The project directory contains a template in `writeup.txt`. For each task, if you believe that you have correctly implemented the requested functionality, simply leave the text "Fully implemented." If you were not able to achieve the requested functionality, write a short explanation of what your submission does and does not do correctly, and (optionally) briefly explain your strategy and how you got stuck. If your explanation indicates significant understanding of the problem, we may award partial credit even though the code does not work.

If there are any parts of the documentation that seems confusing, make sure to also include your thoughts/fixes in the writeup.

Failure to submit this writeup on either Checkpoint or Final will incur a 5 pt penalty on the assignment it is forgotten on.

Code Environment

This codebase should compile on Linux, macOS, and Windows on a typical environment. The build instructions given on the project page will walk you through installing dependencies and building the code. If you have difficulties running the code on your local machine, the GHC 5xxx cluster machines have all the packages required to build the project.

Handin Instructions

As with the previous assignments, **you will be submitting your work to Autolab**. For Checkpoint, you should create a tar archive of your entire `src` subdirectory along with the writeup (`writeup.txt`).

```
$ pwd
> (...)/DrawSVG
$ tar cvzf handin.tgz src/ writeup.txt
> a src
> a src/cmake
> a src/CMakeLists.txt
(...)
> a writeup.txt
```

For Final, you should create a tar archive of your entire `src` subdirectory along with the writeup (`writeup.txt`) and Task 9 submission (`task9.svg`).

```
$ pwd
> (...)/DrawSVG
$ tar cvzf handin.tgz src/ writeup.txt task9.svg
> a src
> a src/cmake
> a src/CMakeLists.txt
(...)
> a writeup.txt
> a task9.svg# Assignment 1 Instructions
```

We have installed an autograder to test if your code complies on the GHC 5xxxx cluster machines. Submissions that build will recieve a score of 0, and submissions that fail will recieve a score of -15462. You can view the autograding logs to see where in your code the build failed. You have unlimited submissions to fix these errors before the deadline.

Happy rasterization!