CSCI 200 - Fall 2023
Foundational Programming Concepts & Design

Lab 4C - SFML: Bob Ross

Home | HW Sets | Schedule | Files | Help ▾ | Links ▾

This lab is due by Tuesday, October 31, 2023, 11:59 PM.
As with all labs you may, and are encouraged, to pair program a solution to this lab. If you choose to pair program a solution, be sure that you individually understand how to generate the correct solution.

Jump To: **Rubric Submission**

Concepts

For this assignment, you have the opportunity to play with the power that SFML (a special framework called Simple and Fast Multimedia Library) offers.

SFML

SFML is a multimedia Application Programming Interface (API) written in C++ with bindings for various programming languages, including Java, Python, and Ruby. SFML provides an easy way to write code that utilizes graphics, sound effects, and/or networking. SFML is the chosen platform for many cool games, including the **Atom Zombie Smasher**. You can check out everything SFML has to offer by reading the **SFML 2.6.0 documentation**.

Setup

RMP-H

The first step is to download the source code from SFML from the **SFML Download Page**. Be sure to download the zip for the **Source Code** and not an OS specific package. This is at the bottom of the page. Once downloaded, unzip the package.

Next, download the **cross-platform Makefile package**. Once downloaded, unzip the package and place the contents inside the SFML-2.6.0 folder from the above step alongside the `readme.md` file.

Now in a terminal, navigate into the SFML-2.6.0 folder and type `make` .

- Note 1A: If you are running OS X and on a M1 or M2 machine, you will need to ensure you have installed Rosetta. In a terminal, run the command

  ```
  > softwareupdate --install-rosetta
  ```

- Note 1B: If you are running OS X, you will initially be prompted to enter your user password to allow the OS to unquarantine the SFML files.

- Note 2: If you are building on the lab machines, you will need to modify the `Makefile.win` file. Open this file in a text editor and set the `CXX` variable to be the full absolute path of `C:/mingw-w64/mingw64/bin/g++.exe` . The lab machines have multiple versions of `g++` and we need to be explicit for which one it should use.

This Makefile will build each of the necessary libraries. If you receive errors during this step, please post to the corresponding post on Ed matching your operating system. YMMV in this process but we are here to help!

Once successfully built, you'll now need to copy the library files to your installation. First, you'll need to copy the `include/` headers that declare all the classes within the library. Next, you'll need to copy the precompiled library files. Finally, you'll need to copy the runtime files. Where each of these go will depend on operating system:

- **Windows Personal Machine**:
  1. Find your MinGW installation folder (likely `C:/mingw64` or similar). Copy the entire `SFML` folder from `~/SFML-2.6.0/include` to the `~/mingw64/x86_64-w64-mingw32/include` folder contained within the MinGW folder (if the `x86_64-w64-mingw32` folder does not exist, then place the `SFML` folder inside of the `include` folder in `~/mingw64/include` .
  2. Find your MinGW installation folder (likely `C:/mingw64` or similar). Copy the five `libsfml-*.a` files from `~/SFML-2.6.0/build/libs` to the `~/mingw64/lib` folder contained within the MinGW folder.

3. Find your MinGW installation folder (likely `C:/mingw64` or similar). Copy the five `sfml-*.dll` files from `~/SFML-2.6.0/build/libs` to the `~/mingw64/bin` folder contained within the MinGW folder so they reside alongside the `g++` program.

- **Windows Lab Machine**:
  1. On your Z: drive, create the following folder path: `Z:/CSCI200/include`. Copy the entire `SFML` folder from `~/SFML-2.6.0/include` to this folder.
  2. On your Z: drive, create the following folder path: `Z:/CSCI200/lib`. Copy the five `libsfml-*.a` files from `~/SFML-2.6.0/build/libs` to this folder.
  3. Here is where things are slightly less ideal. Due to the network accounts, there is not a standard folder that persists on your path that you can write files to. Therefore, on your Z: drive, create the following folder path: `Z:/CSCI200/bin`. Copy the five `sfml-*.dll` files from `~/SFML-2.6.0/build/libs` to this folder. Whenever you are working with an SFML project, you will need to copy these five DLL files into the folder where your executable exists so they can be found at runtime.
- **OS X Personal Machine**:
  1. Copy the entire `SFML` folder from `~/SFML-2.6.0/include` to `/usr/local/include`. This can be done through the Finder selecting Go > Go To Folder and entering `/usr/local`. If the `include` folder does not exist, then create it.
  2. Copy the five `*.dylib` files from `~/SFML-2.6.0/build/libs` to `/usr/local/lib`. Again, if this folder does not exist, then create it.
  3. Copy the seven `*.framework` folders from `~/SFML-2.6.0/extlibs/libs-osx/Frameworks` to `/Library/Frameworks`. Again, this can be done through the Finder selecting Go > Go To Folder and entering `/Library/Frameworks`.

Instructions

Download the **SFML Template**. This will create an empty window to start working with. First, take a look at the `main.cpp` file provided. In class, we discussed each of the commands shown (e.g., creation of the window object and the polling for events); ask questions if there is any confusion.

Second, we also saw the development of a smiley face in class today. A few key lines of code covered follow:

```
// Draw a circle object called face and color it yellow
CircleShape face;
face.setPosition( 15, 15 );
face.setRadius( 300 );
face.setFillColor( Color::Yellow );
window.draw( face );
```

```
// Draw a rectangle object called eye and color it blue
RectangleShape eye;
eye.setSize( Vector2f( 45, 150 ) );
eye.setPosition( 200, 150 );
eye.setFillColor( Color(0, 0, 255));
window.draw( eye );

// Draw a text object called label
// place in file loading section
Font myFont;
if( !myFont.loadFromFile( "data/arial.ttf" ) )
    return -1;
// place in drawing section
Text label;
label.setFont( myFont );
label.setString( "Hello World!" );
label.setPosition( 250, 520 );
label.setFillColor( Color::Black );
window.draw( label );
```

Your job is to draw *something* in SFML. What you draw can be anything you want *EXCEPT* a smiley face (e.g., a tree, a bike, a dog, a word using rectangles/circles, etc.). For full credit, you must draw at least five shapes. Be creative and have fun!

Grading Rubric

Your submission will be graded according to the following rubric:

Points	Requirement Description
0.70	Fully meets specifications
0.15	Submitted correctly by Tuesday, October 31, 2023, 11:59 PM
0.15	**Best Practices** and **Style Guide** followed
1.00	**Total Points**

Lab Submission

Always, **always**, **ALWAYS** update the header comments at the top of your `main.cpp` file. And if you ever get stuck, remember that there is LOTS of **help** available.

Zip together your `main.cpp`, `Makefile`, `data/*` files and name the zip file `L4C.zip`. Upload this zip file to Canvas under L4C.

This lab is due by Tuesday, October 31, 2023, 11:59 PM.

As with all labs you may, and are encouraged, to pair program a solution to this lab. If you choose to pair program a solution, be sure that you individually understand how to generate the correct solution.

Last Updated: 10/06/23 09:12

Any questions, comments, corrections, or request for use please contact jpaone {at} mines {dot} edu.

[Jump to Top] [Site Map]