Lab 04

First Semester 2014 Sharat

CS 251: Lab 04: [Code Warrior] Makefiles, compilation and linking

- Handed out: 15/08 Due: 18/08 Monday 11pm
- Please write (only if true) the honor code. If you used any source (person or thing) explicitly state it. You can find the honor code on the web page.

Overview

In this lab we will learn about "Makefiles", compiling and linking to libraries in Linux.

A large project has many components, possibly written by different group members, or even by people on the Internet. To produce the final executable, it is unnecessary to recompile code that has already been previously compiled into an intermediate stage. This was valuable when compilers were slow, but even today, it is very relevant (especially when we have optimizing compilers.) A file that tells the compiler what to compile when things change (i.e., when development is ongoing), and what not to, is called a Makefile.

The process of producing an executable involves intermediate files called "object files" and libraries. The final executable involves complicated shared libraries in today's run time environment.

Pre-tasks

1. All exercises in Lab 04 are to be done from the terminal. Before coming to the lab you should learn how to use the terminal and the command line interface (CLI) in Linux. Do not use the GUI for things like renaming, moving, copying, searching files. Read the first few chapters of Learing the Shell

   Now is the time to pick an editor that you are going to be comfortable with. (We heartily recommend the mother, father, and grand parents (so to speak) of all editors, `emacs` but we will be just fine with any other editor that you might want to use.)

2. Please also read the information in Tutorial 4 section of the web page. That's most of all you need to read for the lab, so don't scour the Internet and get confused!

3. Also to do before coming to lab is to place your submission for Lab 03 on the web. Link your Lab 02 HTML page to Lab 03 HTML page (any way you like). What will you put in the Lab 03 HTML page? Whatever you submitted to Moodle but in an unpacked form (i.e., non-zip, non tgz form).

   The purpose of this "linking" process is that when some one comes to the viva in the lab, the teaching team can get a complete summary of all your labs (so that we can ask questions).

4. (Important!) Also to do before coming to lab is a picture of an object that you are going to build in Box2D. This is going to be some sort of mechanical Rube Goldberg like machine simulation has to be on this page. Put a diagram to the actual machine internals that you want to design.

In order to do this, you will need to read about the machine that you want to design. Do not forget to put links to your sources.

Look at this example for inspiration. There are several others on the Internet. Box2D Mountain Bike You can discuss your proposed design with anyone you like, there are no restrictions. However, designs must be sufficiently different and once frozen, no discussions are permitted.

You must make the diagram in Inkscape and save it as a image file. Your design must contain at least 10 different moving elements. The base code has 7-8 elements. Note that your final design for the project can be different in details from this preliminary design. Again, do not forget to add a list of all the sources that you referred to for learning about the machine and Box2D

Create a page called `mydesign.html`. Link this file when you come for Lab 05 (i.e. for the next lab). However, show this file in a home area other than your web area to your TA when you meet him on Monday.

You should get the idea now. When you have come to Lab 04, you have already submitted Lab 03 to Moodle but you are linking up the page NOW. When you come to Lab 05, you would have already submitted to Moodle your Lab 04, but you will link it on that Monday when you have Lab 05.

Why are we doing this one lab apart? One, we want to ask viva questions. Two, we don't want your current Lab to be revealed to other groups. That is, placing on a public page comes only a few days after submission. Viva will cover all contents *except* the current lab (in this example, Lab 04).

In Lab Tasks

1. Study the makefile provided along with `cs251_base_code` and answer the questions given in the quiz. You may want to consider the following points also.
2. What are variables in makefile?
3. Try and understand the rules corresponding to the `$(BINDIR)/(TARGET)`
4. How is gcc invoked in the makefile? What are the flags used while invoking gcc and what do they mean?

Files To Submit

There are no files to submit. You should answer the quiz on Moodle. We repeat the questions here for your convenience and so that you can study this later.

1. Change the appearance of "CS251 Base Code" from line 130 in main.cpp to "CS251 Base Code for Group groupnumber" where groupnumber is your group number. Also include your group member names as approprite. Run make.

   What are the names of files that gets compiled?
2. The provided Makefile uses `$<`. What does this imply in this specific makefile? Give a specific answer. (Do not answer this question in general.)
3. Reflect whether make echoes every recipe that it executes. Has there been any attempt to supress this echo? Give an example, if any, of automatic echo of recipe. Give another example if any of supression. Don't give more than 2 examples.
4. Suppose your team member in your group writes a C++ program that generates a file on the file system called `output`. You are independently writing a Makefile and you want to define a target, and you decide `output` is a good name for the target. Reflect on what problems can arise in this situation.

Write as answer a Makefile fragment that might address the problem. (You should not be writing the problem in English, just write Makefile syntax lines.)

5. Suppose your current directory contains several `.cpp` files along with other files. Write a one line directive to generate `.o` files in a predefined directory called obj, and having the same base name as the `.cpp` files. You should use the `wildcard` keyword.