Libraries, Make and Makefiles

UNX511 Week 2 Class 1

John Sellens

May 13, 2025

Seneca Polytechnic

Outline

Introduction to Libraries

Introduction to Libraries

- Programmers usually use libraries of related code as building blocks
  - A library is a collection of (usually) related functions for general use
- For example, and C stdio (standard I/O) library – `"printf()"` etc.
- The `"libm.a"` mathematics library provides various math functions
- Lots of common libraries, lots of optional libraries
- Programming projects often have their own libraries

Why Libraries?

- A convenient way to share code among projects
  - Or organizations
- And to delegate responsibility
- And to define (and enforce) APIs

Static vs Shared Libraries

- Libraries were originally static – the library code was copied into your compliled binary, and your binary was self-contained
- This meant that there were *many* copies of common code, in many different commands
- And meant that if a new library version with fixed code was released, everything had to be recompiled
- And there were typically many copies of the same code in memory, while different programs were running
- Shared libraries avoid (usually) these problems

How Do Shared Libraries Work?

When you run a command built with shared libraries

- The OS searches for the libraries you will need
- Dynamically links the command code with the library functions
  - Windows shared libraries are DLLs – dynamically linked libraries
- Only one copy of the shared library code needs to be in memory for running commands (typically)
- New versions of libraries are usually backwards compatible, and are immediately available to previously compiled commands

Downsides of Shared Libraries

- Your command isn't self-contained – it may require other things to run
  - I think the go language tends to prefer static linking
- New library versions could potentially break existing code
  - Though this rarely happens these days
- Different programs could require different versions of libraries, and it may not be possible to have more than one at once
  - There was a time when "Windows DLL hell" was a concern

Shared Libraries in Linux

- C (and C++) programs in Linux typically use shared libraries
- Linux allows different versions of shared libraries to co-exist
- Programs can be compiled “statically” if desired/needed

Library Tools in Linux

- `ar` creates and manages static libraries
- `nm` tells you what names are defined in a library (or object file)
- `ldd` tells you what shared libraries a command needs
- `ldconfig` is a system command to manage shared library searching
- The `LD_LIBRARY_PATH` environment variable provides a search path for shared libraries

Make and Makefiles

The Make Command

- `make` is a tool for building programs and projects
- It uses file modification times
- And a set of file relationships and build commands (a `"Makefile"`)
- To build your program/project up to date in the most effcient way
- From the early UNIX days – Stuart Feldman, Bell Labs, 1976
  https://en.wikipedia.org/wiki/Make_(software)
- You can use `make` to run other command sets, if you can define file modification time rules for building
- The GNU make variant adds more features and functionality

Makefile Syntax

- A `Makefile` is a text file, typically in the same directory as your code
  - The file name is (usually) capitalized so it appears near the start of `ls` output
- It defines target / dependency relationships
- And build commands
  - Careful: Build commands must be indented with a tab character, not leading spaces
- And provides and allows for variables

Examples

- Let's look at some simple examples of `make` and `Makefile`s for building C programs and libraries

Alternatives to Make

- These days, it seems like just about every programming language has its own build system
- Many IDEs have (effectively) a `make` command built-in
- GNU automake and autoconf provide tools to build `Makefile`s from templates

Summary

- Libraries are useful, and key building blocks
- `make` (and similar tools) are great productivity and consistency tools