Week 3: Linux Program Installation

CLI,compilation,installation,linux,make

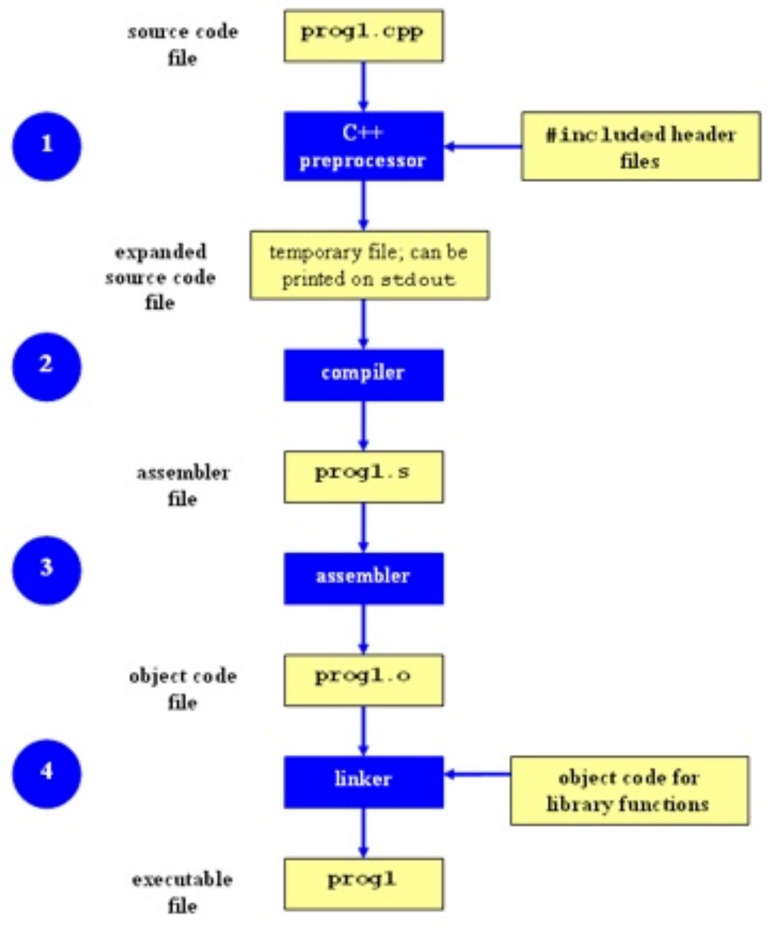
Installing Software

- rpm (Redhat Package Management)
    - RedHat Linux (.rpm)
- apt-get (Advanced Package Tool)
    - Debian Linux, Ubuntu Linux, .deb
- **good old build process**
    - configure, make, make install

Decompressing Files

- you generally receive Linux software in the **tarball** format (.tgz) or (.gz)
- decompress file in current directory
    - `$ tar -xzvf filename.tar.gz`
        - option -x: —extract
        - option -z: —gzip
        - option -v: —verbose
        - option -f: —file
    - `$ tar -cz`: create
    - `$ tar -xz`: extract

Compilation Process



Command-Line Compilation

Given source files:

- shop.cpp
  - #includes shoppingList.h and item.h
- shoppingList.cpp
  - #includes shoppingList.h
- item.cpp
  - #includes item.h

Compile Command

```
g++ -Wall shoppingList.cpp item.cpp shop.cpp -o shop
```

- option -Wall: show all errors and process information

Changes to Header/Source Files

Change one header or source file

- rerun command to generate new executable

Small change (to item.cpp)

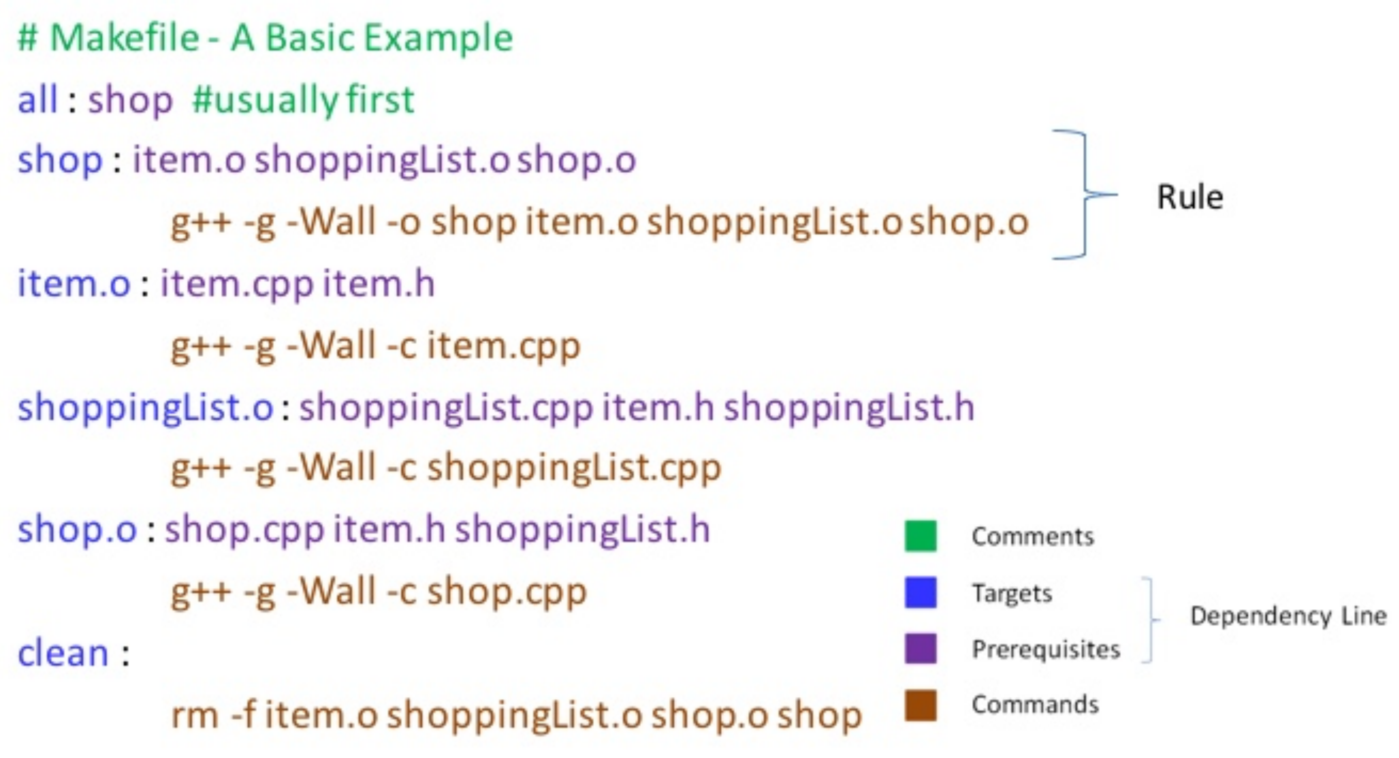
- not effecient to recompile shoppingList.cpp and shop.cpp
- **Solution:** avoid waste by producing a seaprate object code file for each source file
    - `g++ -Wall item.cpp -c item.o` (create object executable for each source file)
    - `g++ item.o shoppingList.o shop.o -o shop` (link)
    - less work for compiler, saves time but more commands

Change item.h

- need to recompile every source file that includes it & every source file that includes a header tha includes it
    - here, that includes item.cpp and shop.cpp
- difficult to keep track of files when project is large
- **USE MAKE**

Make

- utility for managing large software projects
- compiles files and keeps them up-to-date
- efficient compilation (only files that need to be recompiled)



```
# Makefile - A Basic Example
all : shop  #usually first
shop : item.o shoppingList.o shop.o
        g++ -g -Wall -o shop item.o shoppingList.o shop.o
item.o : item.cpp item.h
        g++ -g -Wall -c item.cpp
shoppingList.o : shoppingList.cpp item.h shoppingList.h
        g++ -g -Wall -c shoppingList.cpp
shop.o : shop.cpp item.h shoppingList.h
        g++ -g -Wall -c shop.cpp
clean :
        rm -f item.o shoppingList.o shop.o shop
```

Build Process

configure

- script that checks details about the machine before installation
    - dependency between packages
- creates ‘Makefile’

make

- requires ‘Makefile’ to run
- compiles all the program code and creates executables in current temporary directory

make install

- make utility searches for a label named install within the Makefile, and executes only that section of it
- executables are copied into the final directories (system directories)

Lab 3

- coreutils 7.6 has a problem
    - different users see different date formats due to different locales
    - `$ -l /bin/bash`
        - `-rwxr-xr-x 1 root root 729040 2009-03-02 06:22 /bin/bash`
        - `-rwxr-xr-x 1 root root 729040 Mar 2 2009 /bin/bash`
- want traditional Unix format for all users
- fix the ls program