CS-701 Lecture 16

March 27, 2001

Administrivia

- Use binary mode when transferring tar files using *ftp.*
- Bulletin Board message on printing web pages.
- Exam next Tuesday
  - C coding: arrays, pointers, memory regions, dispatch tables, header files, function prototypes, multiple source modules …
  - Chapters 1, 2, 5, 8 of Stevens (see Blackboard)
  - Development Tools
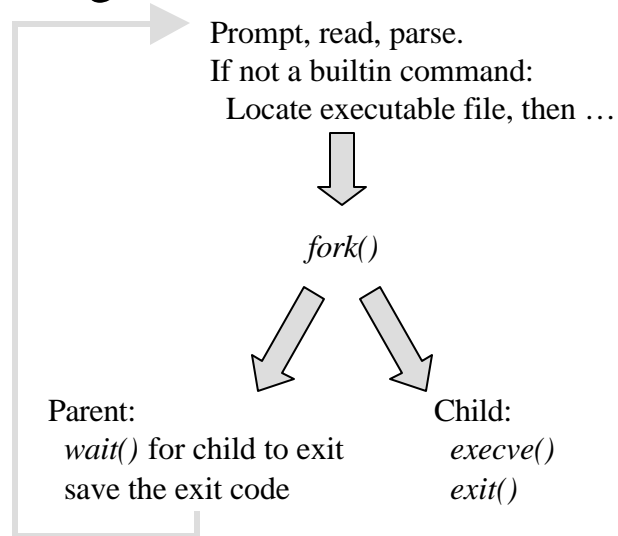   - Compiler driver
    - RCS
    - *make*

A Makefile

```
LDFLAGS     = -L/usr/local/pgsql/lib -lpq
CXXFLAGS  = -g -Wall -I/usr/local/pgsql/include
EXECS       = mkstudent mkffenum
SRCS         = mkstudent.cc mkffenum.cc
OBJS         = $(SRCS:.cc=.o)
HEADERS    = flat_fields.h flat_fields_enum.h
all : $(EXECS)
clean :
     rm -f *.bak *.o core $(EXECS)
depend : $(SRCS) $(HEADERS)
     makedepend -Y -- $(CXXFLAGS) -- $(SRCS) >/dev/null 2>&1
flat_fields_enum.h : mkffenum
     mkffenum > $@
# DO NOT DELETE
mkstudent.o: /usr/local/pgsql/include/libpq-fe.h
mkstudent.o: /usr/local/pgsql/include/postgres _ext.h flat_fields.h
mkstudent.o: flat_fields_enum.h
mkffenum.o: flat_fields.h
```

Unix File Names

- There are some conventions about file name
  extensions, but there are no formal associations
  between file name extensions and applications, the
  way there are in DOS/Windows.
    - *make* bases its builtin rules on filename extensions.
- README *vs.* readme.txt
    - Uppercase filenames appear first in directory listing,
      and thus are easier to find.
    - Extensions like *.txt* and *.exe* are meaningless and just
      cause more typing for the user.

Shell Algorithm for External Commands

Prompt, read, parse.
If not a builtin command:
  Locate executable file, then …

fork()

Parent:
  wait() for child to exit
  save the exit code

Child:
  execve()
  exit()

Checking a File's Status

char *pathname = "/usr/bin/ls;
struct stat statBuf;
  stat(char* pathname, & statBuf)

– *Returns zero if file exists. statBuf will contain lots of information about the file, such as size, time/date of last modification, etc.*

  • *But for this assignment, all we need to know is that it exists; execve() will check that it is executable, etc.*

  • *It would be better to check the permissions and skip files the user cannot execute. (Optional part of the assignment.)*