System Prototype and HW/SW Design

教　　師：蘇　　慶　　龍
INSTRUCTOR : CHING-LUNG SU

E-mail: kevinsu@yuntech.edu.tw

cs.yzu.edu.tw

Lab. 2
Makefile 撰寫

簡介

寫程式寫很大時，會分成好幾個模組，就是一個個的C或其他程式的小檔案。Make是編譯大量的source code一定要用到的工具，最常用就是寫一個Makefile，根據裡面的目標(target)所定義的規則(rule)來做編譯的動作，創造可執行的程式來。

oss.ntnu.edu.tw

一個makefile只是一堆規則(rule)所組成。一個規則的形式是這樣的

```
target : prequiste ;
```

command通常是寫成

```
 target  :  prerequiste
                    command
                    command
                    command
```

如果一行不夠寫要分兩行，可以用 \ 來變成兩行底下是一個例子

www.cs.nctu.edu.tw

目標(target)

目標可以是檔名或是一個代表動作的識別符號，如果不是檔名的Target叫phony target，make根據指定的target來做相關動作。若只下make命令而已，第一個rule永遠被執行，這叫default goal。

cs.ysu.edu.tw

一些目標規定

- all　　：內定的編譯動作
- install　　：安裝 binary 檔的動作
- clean　　：清除 obj 動作
- dist　　：產生 configure 的動作
- distclean　　：清除 configure 所產生的檔

sysedu.thu.edu.tw

變數(variables)

make規則裡面的組成可以有很多動態的值，這時就需要用變數來設

- 值取代與轉換以及日後維護。變數也有一些規定：
- 字母大小有差。
- 不要用字母數字底線以外的字元，可以有空格在前面後面。
- 使用變數用$(VAR)或者${VAR}都可以。
- 如果要用$，多加一個$變成$$，在Shell command會用到Shell變數此時就要加$。

Tsi-edu.tw.is

Example:設objects


```
objects = program.o foo.o utils.o
program : $(objects)
        cc -o program $(objects)
```

For Educational Purposes Only

命令(command)

命令是要完成一個目標所要做的動作，有幾個重要的規定：

- Command前面一定要是個 TAB 鍵，不可以是空白鍵。
- 每一行的命令只是喚起一個 sub shell 來執行命令，做完後sub shell 就沒有了。
- 要把錯誤略過不看，在命令前加個 “-“，要不秀出命令在螢幕上加個 “@” 。
- 喚起的 sub shell 要用什麼 shell ，是定義在SHELL這個變數裡。

Speed Up Systems

內隱規則

通常編譯程式時有個共同習慣，就是把foo.c編成foo.o。gnu make有一些內定

規則來編譯，也就是有的target你不寫，make也可以根據內定規則把它編譯出

例如：

```
foo.o:foo.c
        gcc -c -o foo.o foo.c
foo1.o:foo1.c
        gcc -c -o foo1.o foo1.c
....
```

因此，如果不寫foo.o的規則，那麼make當別的規則用到foo.o時，找不到規則來編，會自動找foo.c來編譯。這樣看不到的編譯規則有很多，例如

```
C程式
$(CC) -c $(CPPFLAGS) $(CFLAGS) xxx.c來編譯
或者找不到.c時會去找.cc, .cpp, .C檔

C++程式
$(CXX) -c $(CPPFLAGS) xxx.cpp

TeX
xxx.dvi 由xxx.tex產生

Pascal
$(PC) -c $(PFLAGS) xxx.p
```

自訂規則

- 樣式規則(pattern rule)

可用pattern rule來做一些自訂的內隱規則，像這樣

```
%.o : %.c prog.h
        $(CC) $(CFLAGS) $(DEBUG_FLAG) -c -o $@ $<
```

%表示所有相對於後面先決條件的檔名的意思，它不是*，因為它有一對一相對應關係foo.o就要找foo.c，foo1.o就要找foo1.c。上面的意思是所有碰到.o的target時，去找相對應的.c檔，並根據先決條件prog.h做檢查。

www.edu.tw Linux

Example:簡單Makefile


```
SHELL=/usr/bin/bash

edit:     main.o kbd.o command.o display.o \
          insert.o search.o files.o utils.o
          $(CC) -o edit main.o kbd.o command.o display.o \
          insert.o search.o files.o utils.o

command.o :command.c command.h
          $(CC) -o command.o command.c

clean:
          @rm *.o *~
```

其中定義了SHELL這個變數，有3個目標(target)edit、command.o、clean，雖然沒有定義CC這個變數，用了內定變數$(CC)去編譯程式。如果給

```
$ make clean
```

Make會另外叫起一個shell來執行裡面的字串 rm *.o

```
$ make edit
```

Make檢查需要的先決條件(prerequiste)發現有個檔名target command.o存在，會依序根據規則來編譯。

cs.ntust.edu.tw

参考網址

http://www.study-area.org/tips/opentools/opentools/book1.html

cecs.yzu.edu.tw