

Introduction à make

2018

Florent Gluck – Florent.Gluck@hesge.ch

Version 0.6

Introduction

Historique

- Créé en 1977 par Stuart Feldman à Bell Labs.
- Devenu très populaire car fourni en standard avec UNIX AT&T.

But de `make` ?

- Permet d'automatiser le processus de conversion de fichiers d'un format à un autre, **en gérant les dépendences**.
- Souvent utilisé lors de la compilation de code source en code objet, puis pour l'édition des liens afin de créer le binaire final.
- Tout projet conséquent utilise un outil similaire à `make`.

Qu'est ce qu'un makefile ?

- Un **makefile** est un fichier contenant une liste de règles et de dépendances utilisées pour construire des cibles.
- En C/C++ make est utilisé pour compiler une liste de fichiers sources
- Pourquoi ? En C/C++, les projets sont composés de fichier headers (.h) et de fichier sources (.c/.cpp).
- Syntaxe d'un règle :

```
cible(s) : dépendance(s)  
          commande(s)  
          ...
```

Exemple

example.c

```
#include <stdio.h>

int main(int argc, char **argv){
    printf("Programme très complexe...");
    return 0;
}
```

makefile

```
example: example.c
    gcc example.c -o example
```

```
> make
gcc example.c -o example
```


Exemple

```
example.c

#include <stdio.h>

char **argv){
    comme très complexe...");
}
```

Cible (*target*)



```
makefile

example: example.c
    gcc example.c -o example
```

```
> make
gcc example.c -o example
```

Exemple

example.c

```
#include <stdio.h>

int main() {
    printf("Exemple...");
    return 0;
}
```

Dépendences



makefile

```
example: example.c
    gcc example.c -o example
```

```
> make
gcc example.c -o example
```

Exemple

example.c

```
#include <stdio.h>

int main(int argc, char *argv[]) {
    printf("Programme\n");
    return 0;
}
```

**Règles de
production (*rules*)**



makefile

```
example: example.c
    gcc example.c -o example
```



```
> make
gcc example.c -o example
```

Principe

- 1) `make` cherche dans le répertoire courant un des fichiers suivants :
`makefile`, `Makefile`, `GNUmakefile`.
- 2) Exécute la **première cible** si appelé sans argument, ou celle spécifiée en ligne de commande.
- 3) Décide si une cible doit être régénérée en comparant les dates de modification des fichiers → **ne recompile que ce qui a été modifié**
- 4) Regarde si les dépendances sont satisfaites :
 - Si elles le sont → exécute les commandes associées à la cible.
 - Sinon → prend la première dépendance pour cible et recommence l'étape 4)
- 5) En terme d'exécution, `make` se comporte donc comme un programme **récuratif**.

Autre exemple

makefile_2

```
# Règle principale
example: example.o
        gcc -o example example.o

# Deuxième règle
example.o: example.c example.h
        gcc -c example.c

# Efface fichiers objets et exécutable
clean:
        rm -f example.o example
```

```
> make -f makefile_2 clean
> make -f makefile_2
```

Exemple

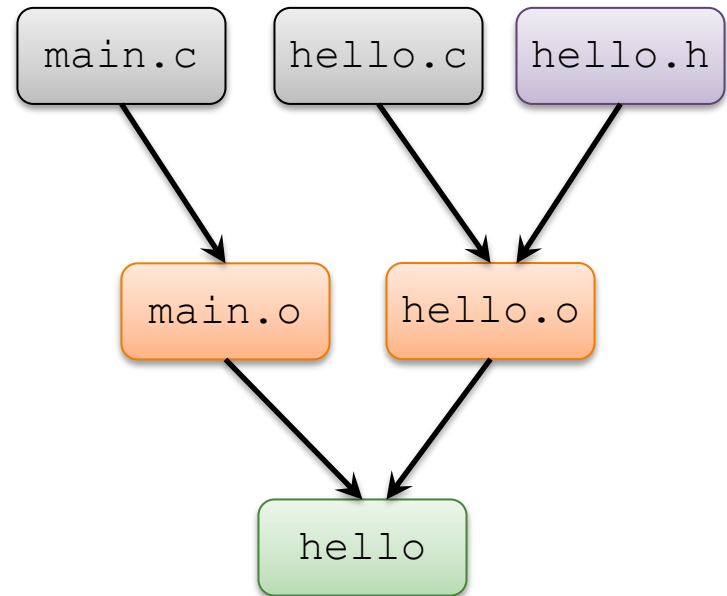
```
hello: hello.o main.o
      gcc hello.o main.o -o hello

hello.o: hello.c hello.h
      gcc -Wall -Wextra -c hello.c

main.o: main.c
      gcc -Wall -Wextra -c main.c

clean:
      rm -f *.o hello

rebuild: clean hello
```



Variables

Variables utilisateur

- Déclaration : `nom=valeur`
`nom=valeur1 valeur2 valeur3`
- Utilisation : `$ (nom)`
- Une variable peut aussi être déclarée en ligne de commande :
`make CFLAGS="-O2 -Wall"`

Variables internes

- `$@` : représente la cible
- `^` : représente la liste des dépendences
- `$<` : représente la première dépendence
- `$*` : représente le nom de la cible sans extension

Exemple de factorisation

```
hello: hello.o main.o
    gcc hello.o main.o -o hello

hello.o: hello.c hello.h
    gcc -Wall -Wextra -c hello.c

main.o: main.c
    gcc -Wall -Wextra -c main.c

clean:
    rm -f *.o hello

rebuild: clean hello
```



```
CC=gcc -Wall -Wextra

hello: hello.o main.o
    $(CC) $^ -o $@

hello.o: hello.c hello.h
    $(CC) -c $<

main.o: main.c
    $(CC) -c $<

clean:
    rm -f *.o hello

rebuild: clean hello
```

Le manuel de GNU make

- <http://www.gnu.org/software/make/manual/make.html>

Quelques tutoriels

- <http://www.opussoftware.com/tutorial/TutMakefile.htm>
- <http://mrbook.org/tutorials/make/>