Les makefiles

Définition :

Les Makefiles sont des fichiers, généralement nommés makefile ou Makefile, utilisés par le programme make pour exécuter un ensemble d'actions telles que la compilation d'un projet, l'archivage des documents, etc.

Un Makefile est constitué de plusieurs règles de la forme :

CIBLE: DEPENDANCE

<Tabulation>COMMANDE

L'évaluation du fichier makefile se fait selon les règles suivantes :

- Les dépendances de chaque règle sont analysées. Si une dépendance est la cible d'une autre règle du Makefile, cette règle est à son tour évaluée.
- Si la cible ne correspond pas à un fichier existant ou si la dépendance est plus récente que la cible, les différentes commandes sont exécutées.

On lance un fichier makefile via la commande make.

Exemple :

Principal.c

```
#include<stdio.h>
#include "Prog1.h"
int main() {
coucou();
return 0;
}
```

Prog1.c

```
void coucou() {
printf("coucou \n");
}
```

Prog1.h

```
void coucou();
```

Makefile:

```
Principal: Prog1.o principal.c
        Gcc –o principal principal.c Prog1.o
Prog1.o: Prog1.c Prog1.h
        Gcc –c Prog1.c
Clean:
        Rm –rf *.o
```

On peut ajouter une cible pour supprimer tous les fichiers .o par exemple

Les variables :

Une variable dans un makefile se déclare sous la forme : NOM_VARIABLE=VALEUR

L'accès à la valeur de la variable se fait comme suit : $(Nom de la variable)

Les variables internes :

Les makefile possèdent des variables internes qui pourront faciliter l'accès aux différents champs d'une règle.

$@	Le nom de la cible
$<	Le nom de la première dépendance
$^	La liste des dépendances
$?	La liste des dépendances les plus récentes que la cible
$*	Le nom du fichier sans suffixe

Les cibles .PHONY :

Dans l'exemple présent, clean est la cible d'une règle ne présentant aucune dépendance. Supposons que clean soit également le nom d'un fichier présent dans le répertoire courant, il serait alors forcément plus récent que ses dépendances et la règle ne serait alors jamais exécutée.
Pour pallier ce problème, il existe une propriété particulière nommée .PHONY dont on affecte à un ensemble de cibles. Chaque cible possédant la propriété .PHONY sera systématiquement reconstruite dans le cas où on l'appelle dans le makefile.
Les cibles .PHONY sont déclarées comme suit :

```
.PHONY : clean …
```

Génération des fichiers sources et objets :

Plutôt que d'énumérer la liste des fichiers sources et objets dans les dépendances de la règle de construction de notre exécutable, il est possible de la générer automatiquement à partir de la liste des fichiers sources. Pour cela nous rajoutons deux variables au Makefile:

- SRC qui contient la liste des fichiers sources du projet.
- OBJ pour la liste des fichiers objets.

Les variables SRC et OBJ sont remplies de la manière suivante :

```
SRC=$(wildcard *.c) ou $(shell find . –name '*.c')

OBJ= $(SRC:.c=.o)
```

Au lieu d'écrire une règle pour chaque fichier objet que vous voulez générer, vous pouvez utiliser la façon suivante :

```
%.o :%.c
        commande
```

Les sous makefiles:

Il est aussi possible de lancer des makefiles à partir d'un autre makefile en utilisant la variable $(MAKE) comme suit :

```
Nom de la cible :
        $(MAKE) –C chemin du sous repertoire

Ou bien
Nom de la cible :
        Cd chemin du sous repertoire && $(MAKE)
```