LABORATÓRIO TPSE I

Prática Makefile

Prof. Francisco Helder

21 de março de 2022

Introdução

Neste tutorial irá ser visto como usar o makefile. O makefile nada mais é do que um arquivo que contém diretrizes usadaspara uma compilação altomática. Usando um código C como exemplo, toda vez que um arquivo de origem é modificado ele tem que ser recompilado e caso um cabeçalho seja alterado, todos os arquivos que dependem dele tem de ser recompilados. Isso pode ser facilmente resolvido com o auxílio de um makefile.

Em projetos grandes com muito código, como o kernel do linux o makefile diminui substancialmente o tempo de compilação. Pois caso apenas alguns arquivos que sejam modificados, todos os arquivos que não tem nenhuma dependência do arquivo modificado não precisa ser recompilado, portanto se tiver usando 100 arquivos e modificar apenas 1, o make vai comparar datas e modificações e só compilara o que for preciso.

Programando um makefile

Como dito, um makefile é um arquivo que contem diretrizes de compilação, esse arquivo é interpretado pelo programa make que é nativo do linux, o programa make interpreta o arquivo e executa as regras definidas no arquivo. Para o tutorial será usado códigos C.

Como um primeiro exemplo, crie uma pasta com três arquivos: main.c, header.h e makefile, onde o arquivo main.c é o seguinte:

```
#include "header.h"

int main(){
        printf("Hello World");
        return 0;
}
```

O arquivo "header.h"simplesmente será usado para incluir bibliotecas em geral, o arquivo ficou assim:

```
#ifndef HEADER_H
#define HEADER_H
/* Para evitar a declaracao de uma biblioteca mais de uma vez */
#include <stdio.h>
#include <stdlib.h>

#endif
```

Para a criação do makefile existe algumas regras. O código a seguir mostra basicamente a estrutura de um makefile.

```
# Comentario
all: label1     #Onde label e a label escolhida

label1: dependencia1, dependencia2, ...
<tab>   comando

label2: dependencia1, dependencia2, ...
<tab>   comando

# O uso de <tab> e necessario para a interpretacao do codigo
```

Onde as dependencias podem ser labels ou arquivos. Portanto para o primeiro exemplo o código do makefile ficará desta maneira:

```
all: main

main: main.o
        gcc -o app main.o

main.o: main.c
        gcc main.c -I. -c -o main.o

clean:
        rm -rf *.o
```

Veja que na parte do código é o mesmo que é usado no terminal, como por exemplo o comando para remover arquivos. No código bem simples foi criado três labels, onde cada uma representa alguma parte da compilação.

- all: regras a serem executadas
- main: label com dependencia de main.o
- main.o: label com dependencia do arquivo main.c
- clean: label sem dependencia para excluir arquivos intermediarios

Variaveis

No makefile, é possivel tambem declaração de variaveis que auxiliam na criação das regras de compilação. Para a declarar uma variavel, basicamente usa-se NOME=CONTEUDO e para a utilização no código $(NOME). O makefile abaixo mostra o exemplo 1 modificado com variaveis.

```
all: main

SRC=main

$(SRC): $(SRC).o
        gcc -o app $(SRC).o

$(SRC).o: $(SRC).c
        gcc $(SRC).c -I. -c -o $(SRC).o

clean:
        rm -rf *.o
```