Headers & Library Files

GCC

Joses Paul | Zilogic Systems

Header Files

What Does the Header Files contain ?

- Function declarations
- Macro definitions

```
>bash

$locate stdio.h
$cat  /usr/include/stdio.h
```

Why Header files ?

- Convinience.
- Splitting one big program file into multiple files.
- So that basic functions can be accesed by all source files.

Header Files

What happens when we add a header file is included ?

- Including a header file produces the same results as copying the header file into each source file that needs it

-**E** (Pr**e**processor stage)

-**S** (As**s**embeler code file ".s")

-c (**C**ompile stage ".o")

-o filename (**O**utput in file)

-v **v**erbose

```
>bash

$gcc -E cprogram.c
```

Include syntax

#include <*filename*>

#include "*filename*" ***(or)*** **#include "*location/filename*"**

<filename>

- /usr/local/include
- libdir/gcc/target/version/include
- /usr/target/include
- /usr/include

"filename"

source file directory

<> locations

```
$gcc -I/location program.c
```

```
>bash

$gcc -v program.c
```

Header Files

- **How to add a default header path ?**

To set :
$ export

To remove the path
$ unset

```
>bash


$C_INCLUDE_PATH=/home/user/Desktop
$export  C_INCLUDE_PATH
$
$printenv | grep “C_INCLUDE_PATH”
$gcc -v program.c
```

Once only Header

Wrapper #ifndef Construct

/* File stdio.h */

Controling macros or gaurd macros

```
#ifndef _STDIO_H
#define _STDIO_H
```

the entire file content

```
#endif
```

Note:
--Controling/gaurd macros--
must starts with _ for system header files
must not start with _ for user defined .h files

```
>bash

$cat /usr/include/stdio.h |
> nl | grep “STDIO”
```

Library Files

A library file contains the function definitions.
During the linking porcess, linker automatically links the these library files to the main source file.

- ***Standard library***
- libc.a
  .a → archive file
- Contains object files “ .o “ of function definitions.

```
$ar  t  /usr/lib/x86_64-linux-gnu/libc.a | grep -w "printf"
```

Library Files

- **AR - Archive modifier tool**

  -cr → create (new)

  -q → quick append (add)

  -t → table of content (view)

  -x → extract (unzip)

```
$ar  t  /usr/lib/x86_64-linux-gnu/libc.a | grep -w "printf"
```

Lets make Our Library

- **We have to c programs add & mult**

  Step 2: Compile and get object code

```
>Bash

  $gcc -c add.c
  $gcc -c mul.c
```

  Step 3: make a library libarith.a

```
>Bash

  $ar  cr  libarth.a  add.o  mul.o
```

Lets make Our Library

- **Step 4: create a header file header.h**

  #include <stdio.h>

  int add(int a , int b);

  int mul(int a ,int b);

  Step 5: Make source.c program with the functions:

  step 6: Compile

```
>bash

 $gcc  -Wall  source.c  libarith.a   -o  a.out
```

Lets make Our Library

- **Append the object files to the libc.a**
- **Set Environment Variable: LD_LIBRARY_PATH**
- **gcc source.c -L/home/user/Desktop -l arith -o a.out**

Library name should have a prefix “lib”. Eg:libc , libarith

Final Verdict

Header Files : Included at the top of any program. If a function is used inside a program, then the header file containing declaration of that function has to be included. Example: printf() → #incldue<stdio.h>

Library Files: These are the files which the compiler uses in order to define the functions which have been used in the program and had been declared inside the header file. Example: printf() has its complete definition inside the library file libc.a

Difference:

- Header files are TEXT files
- Library files are BINARY.
- Header file has to be included by the programmer
- Compiler automatically relates the library file(s) with the program!

Headers & Library Files

Reference: GCC

https://gcc.gnu.org/onlinedocs/cpp/Header-Files.html
https://gcc.gnu.org/onlinedocs/cpp/Include-Syntax.html#Include-Syntax
https://gcc.gnu.org/onlinedocs/cpp/Include-Operation.html#Include-Operation
https://gcc.gnu.org/onlinedocs/cpp/Once-Only-Headers.html#Once-Only-Headers

The C Programming Language (Edition 2) - Dennis Ritchie, Brian Kernighan