0x00. C - Dynamic libraries

System programming & Algorithm — Linux programming

by Julien Barbier, co-founder at Holberton School

weight: 1

Ongoing project - started 06-05-2017, must end by 06-12-2017 (in 3 days) - you're 100% done.

Manual QA review must be done (request it when you are done on the project)

Readme

Read or watch What is difference between Dynamic and Static library (Static and Dynamic linking) (https://www.youtube.com/watch?v=eW5he5uFBNM), create dymanic libraries on Linux (https://www.google.com/#q=linux+create+dynamic+library).

What you should learn from this project

At the end of this project you are expected to be able to explain to anyone, without the help of Google:

- What is a dymanic library, how does it work, how to create one, and how to use it
- What is the environment variable `$LD_LIBRARY_PATH` and how to use it
- What are the differences between static and shared libraries
- Basic usage `nm`, `ldd`, `ldconfig`

Requirements (C)

- Allowed editors: `vi`, `vim`, `emacs`
- All your files will be compiled on Ubuntu 14.04 LTS
- Your programs and functions will be compiled with `gcc 4.8.4` (`C90`) using the flags `-Wall -Werror -Wextra and -pedantic`
- All your files should end with a new line

- A `README.md` file, at the root of the folder of the project is mandatory
- Your code should use the `Betty` style. It will be checked using betty-style.pl (https://github.com/holbertonschool/Betty/blob/master/betty-style.pl) and betty-doc.pl (https://github.com/holbertonschool/Betty/blob/master/betty-doc.pl)
- You are not allowed to use global variables
- No more than 5 functions per file
- You are not allowed to use the standard library. Any use of functions like `printf` , `puts` , etc... is forbidden
- You are allowed to use _putchar (https://github.com/holbertonschool/_putchar.c/blob/master/_putchar.c)
- You don't have to push `_putchar.c` , we will use our file. If you do it won't be taken into account
- In the following examples, the `main.c` files are showed as examples. You can use them to test your functions, but you don't have to push them to your repo (if you do we won't take them into account). We will use our own `main.c` files at compilation. Our `main.c` files might be different from the one showed in the examples
- The prototypes of all your functions and the prototype of the function `_putchar` should be included in your header file called `holberton.h`
- Don't forget to push your header file

Requirements (Bash)

- Allowed editors: `vi` , `vim` , `emacs`
- All your scripts will be tested on Ubuntu 14.04 LTS
- All your files should end with a new line (why? (http://unix.stackexchange.com/questions/18743/whats-the-point-in-adding-a-new-line-to-the-end-of-a-file/18789))
- The first line of all your files should be exactly `#!/bin/bash`
- A `README.md` file, at the root of the folder of the project, describing what each script is doing
- All your files must be executable

Tasks

0. A library is not a luxury but one of the necessities of life

mandatory

Done!

Help!

Create the dynamic library libholberton.so containing all the functions listed below:

```
int _putchar(char c);
int _islower(int c);
int _isalpha(int c);
int _abs(int n);
int _isupper(int c);
int _isdigit(int c);
int _strlen(char *s);
void _puts(char *s);
char *_strcpy(char *dest, char *src);
int _atoi(char *s);
char *_strcat(char *dest, char *src);
char *_strncat(char *dest, char *src, int n);
char *_strncpy(char *dest, char *src, int n);
int _strcmp(char *s1, char *s2);
char *_memset(char *s, char b, unsigned int n);
char *_memcpy(char *dest, char *src, unsigned int n);
char *_strchr(char *s, char c);
unsigned int _strspn(char *s, char *accept);
char *_strpbrk(char *s, char *accept);
char *_strstr(char *haystack, char *needle);
```

If you haven't coded all of the above functions create empty ones, with the right prototype. Don't forget to push your `holberton.h` file in your repository, containing at least all the prototypes of the above functions.

```
julien@ubuntu:~/0x17. Dynamic libraries$ ls -la lib*
-rwxrwxr-x 1 julien julien 13632 Jan  7 06:25 libholberton.so
julien@ubuntu:~/0x17. Dynamic libraries$ nm -D libholberton.so
0000000000000a90 T _abs
0000000000000aa9 T _atoi
0000000000202048 B __bss_start
                 w __cxa_finalize
0000000000202048 D _edata
0000000000202050 B _end
00000000000011f8 T _fini
                 w __gmon_start__
0000000000000900 T _init
0000000000000bd7 T _isalpha
0000000000000c04 T _isdigit
0000000000000c25 T _islower
0000000000000c46 T _isupper
                 w _ITM_deregisterTMCloneTable
                 w _ITM_registerTMCloneTable
                 w _Jv_RegisterClasses
0000000000000c67 T _memcpy
0000000000000caa T _memset
0000000000000ce9 T _putchar
0000000000000d0e T _puts
0000000000000d4a T _strcat
0000000000000dcf T _strchr
0000000000000e21 T _strcmp
0000000000000e89 T _strcpy
0000000000000eeb T _strlen
0000000000000f15 T _strncat
0000000000001101 T _strncmp
0000000000000fa5 T _strncpy
0000000000001029 T _strpbrk
000000000000109d T _strspn
0000000000001176 T _strstr
                 U write
julien@ubuntu:~/0x17. Dynamic libraries$ cat 0-main.c
#include "holberton.h"
#include <stdio.h>

/**
 * main - check the code for Holberton School students.
 *
 * Return: Always EXIT_SUCCESS.
 */
int main(void)
{
    printf("%d\n", _strlen("Holberton"));
    return (EXIT_SUCCESS);
}
julien@ubuntu:~/0x17. Dynamic libraries$ gcc -Wall -pedantic -Werror -Wextra -L
. 0-main.c -lholberton -o len
julien@ubuntu:~/0x17. Dynamic libraries$ ldd len
    linux-vdso.so.1 =>  (0x00007fff5d1d2000)
    libholberton.so => not found
    libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f74c6bb9000)
    /lib64/ld-linux-x86-64.so.2 (0x0000556be5b82000)
julien@ubuntu:~/0x17. Dynamic libraries$ export LD_LIBRARY_PATH=.:$LD_LIBRARY_P
ATH
julien@ubuntu:~/0x17. Dynamic libraries$ ldd len
    linux-vdso.so.1 =>  (0x00007fff41ae9000)
    libholberton.so => ./libholberton.so (0x00007fd4bf2d9000)
    libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007fd4beef6000)
    /lib64/ld-linux-x86-64.so.2 (0x0000557566402000)
julien@ubuntu:~/0x17. Dynamic libraries$ ./len
9
julien@ubuntu:~/0x17. Dynamic libraries$
```

Repo:

- GitHub repository: `holbertonschool-linux_programming`
- Directory: `0x00-c_dynamic_libraries`
- File: `libholberton.so, holberton.h`

Check your code?

1. Without libraries what have we? We have no past and no future

mandatory

✓ Done!

Help!

Create a script that creates a dynamic library called `liball.so` from all the `.c` files that are in the current directory.

```
julien@ubuntu:~/0x17. Dynamic libraries$ ls *.c
abs.c   isalpha.c  islower.c  memcpy.c  putchar.c  strcat.c  strcmp.c  strlen.c
strncpy.c  strspn.c
atoi.c  isdigit.c  isupper.c  memset.c  puts.c     strchr.c  strcpy.c  strncat.
c  strpbrk.c  strstr.c
julien@ubuntu:~/0x17. Dynamic libraries$ ./1-create_dynamic_lib.sh
julien@ubuntu:~/0x17. Dynamic libraries$ nm -D --defined-only liball.so
0000000000000a90 T _abs
0000000000000aa9 T _atoi
0000000000202048 B __bss_start
0000000000202048 D _edata
0000000000202050 B _end
00000000000011f8 T _fini
0000000000000900 T _init
0000000000000bd7 T _isalpha
0000000000000c04 T _isdigit
0000000000000c25 T _islower
0000000000000c46 T _isupper
0000000000000c67 T _memcpy
0000000000000caa T _memset
0000000000000ce9 T _putchar
0000000000000d0e T _puts
0000000000000d4a T _strcat
0000000000000dcf T _strchr
0000000000000e21 T _strcmp
0000000000000e89 T _strcpy
0000000000000eeb T _strlen
0000000000000f15 T _strncat
0000000000001101 T _strncmp
0000000000000fa5 T _strncpy
0000000000001029 T _strpbrk
000000000000109d T _strspn
0000000000001176 T _strstr
julien@ubuntu:~/0x17. Dynamic libraries$
```

Repo:

- GitHub repository: `holbertonschool-linux_programming`
- Directory: `0x00-c_dynamic_libraries`
- File: `1-create_dynamic_lib.sh`

Check your code?

2. Either write something worth reading or do something worth writing

mandatory

Done!

Help!

Write a blog post describing the differences between static and dynamic libraries. It should cover:

- Why using libraries in general
- How do they work
- How to create them (Linux only)
- How to use them (Linux only)
- What are the differences between static and dynamic libraries
- What are the advantages and drawbacks of each of them

Your posts should have examples and at least one picture, at the top. Publish your blog post on Medium or LinkedIn, and share it at least on Twitter and LinkedIn.

When done, please add all urls below (blog post, tweet, etc.)

Add URLs here:

+

− https://medium.com/@JohnSpence/static-and-dynamic-libraries-8bfc53e1d3af (https://medium.com/@JohnSpence/static-and-dynamic-libraries-8bfc53e1d3af)
− https://www.linkedin.com/feed/update/urn:li:activity:6278748696008105984/ (https://www.linkedin.com/feed/update/urn:li:activity:6278748696008105984/)
− https://twitter.com/johndspence/status/872982573231865856 (https://twitter.com/johndspence/status/872982573231865856)

Repo:

- GitHub repository: `holbertonschool-linux_programming`
- Directory: `0x00-c_dynamic_libraries`

Done with the mandatory tasks? Unlock 2 advanced tasks now! (/projects/317/unlock_optionals)