Creating and Using a Custom Static Library in C

A Beginner's Tutorial

Date: December 16, 2025

This guide shows how to create a reusable static library in C with multiple functions, so you can use them in any program without recompiling each `.c` file.

Step 1: Header File (`mylib.h`)

```c
/* mylib.h */
#ifndef MYLIB_H
#define MYLIB_H

void hello(void);
int add(int a, int b);
int multiply(int a, int b);

#endif
```

- Declares all the functions you want to expose.
- Use include guards or `#pragma once` to avoid multiple inclusion.

Step 2: Library Implementation (`mylib.c`)

```c
/* mylib.c */
#include <stdio.h>
#include "mylib.h"

void hello(void) {
    printf("HELLO WORLD! with my library\n");
}

int add(int a, int b) {
    return a + b;
}

int multiply(int a, int b) {
    return a * b;
}
```

- Implement all functions in this single file.

Step 3: Main Program (`main.c`)

```c
#include <stdio.h>
#include "mylib.h"

int main(void) {
    hello();
    printf("3 + 5 = %d\n", add(3, 5));
    printf("4 * 6 = %d\n", multiply(4, 6));
    return 0;
}
```

- Only include `mylib.h` to access all functions.

Step 4: Creating and Using the Static Library

```
gcc -c mylib.c       # Compile library implementation
ar rcs libmylib.a mylib.o  # Create static library
gcc main.c -L. -lmylib -o myprogram  # Link library to main
```

- Now you can add more functions to `mylib.c` and `mylib.h` without changing `main.c`.
- This avoids recompiling every source file individually each time.

Output

```
HELLO WORLD! with my library
3 + 5 = 8
4 * 6 = 24
```

Tips

- Keep all reusable functions in the library.
- Only the library file and header need updates when adding new functions.
- Use a Makefile to automate building and linking multiple functions efficiently.