
VIJAY PANCHAL
@vijaykpanchal

Libraries Linking: Static and Dynamic

Platform: Linux

Includes:

- **Intro**
- **Explanation**
- **Comparison**

VIJAY PANCHAL
@vijaykpanchal

Intro

- In Linux, C & C++ application link with “**libc.so”** at runtime.
- libraries need to be linked while compiling the application.
- Two ways to link:
  - Dynamic (shared): The “**lib.so**” is loaded at runtime (Before Running the App).
  - Static The ”**lib.a**” is loaded at build/compile time.
- Dynamic -> Less size & more time
- Static -> More size & less time

Explanation

VIJAY PANCHAL
@vijaykpanchal

- Sample app is using -> “find_square()”.
- Function “find_square()” is defined in “lib-src.c”.
- Build -> $ source build-all.sh
- Execution Binary App:
  - Static -> $ app/sample-app-static
  - Dynamic -> $ app/sample-app-dynamic

```c
/*sample-app.c*/
int main (int argc, char *argv[])
{
    int N = 10;
    int result = find_square(N);
    printf("Square of %d is %d !\n", N, result);
    return 0;
}


/*lib-src.c*/
int find_square (int n){
    return n*n;
}
```

- Performance (sec):
  - Static -> .013 average in 10 execution
  - Dynamic -> .018 average 10 execution **(More time as load libtest.so at runtime)**
- Memory:
  - App Object file size is the same (1096 Bytes)
  - App Binary Size:
    - Static 8480 Bytes as **it includes a libtest.a library at compile time**
    - Dynamic 7640 Bytes

- GitHub Repo: https://github.com/vijaypanchal/lets-learn

Comparison

Static Library

- Linked at compile time
- Less time to execute
- Rebuild need if a change in the library
- Used in platform with no file system support
- Used for Bare-metal & RTOS based app

Dynamic (Shared) Library

- Linked at runtime
- More time to execute
- No build (Replace *.so)
- File system support needed to store *.so
- Used for app developed on OS


VIJAY PANCHAL
@vijaykpanchal

Thank you for Motivation

Interested? -> Follow to view your feed!

Query? -> Please feel free to ask in the comments or DM!