Creating a shared and static library with the gnu compiler [gcc]

Here's a summary on how to create a shared and a static library with gcc. The goal is to show the basic steps. I do not want to go into the hairy details. It should be possible to use this page as a reference.

These examples were tested and run on cygwin/Windows.

The code for the library

This is the code that goes into the library. It exhibits one single function that takes two doubles and calculates their mean value and returns it.

calc_mean.c

```
//#include <stdio.h>

double mean(double a, double b) {
  return (a+b) / 2;
}
```

The header file

Of course, we need a header file.

calc_mean.h

```
double mean(double, double);
```

Creating the static library

A static library is basically a set of **object files** that were copied into a single file. This single file is the static library. The static file is created with the archiver (ar).

First, calc_mean.c is turned into an object file:

```
gcc -c calc_mean.c -o calc_mean.o
```

Then, the archiver (ar) is invoked to produce a static library (named libmean.a) out of the object file calc_mean.o.

```
ar  rcs libmean.a      calc_mean.o
```

Note: the library must start with the three letters *lib* and have the suffix **.a**.

Creating the shared library

As with static libraries, an object file is created. The -fPIC option tells gcc to create position independant code which is necessary for shared libraries.

Note also, that the object file created for the static library will be overwritten. That's not bad, however, because we have a static library that already contains the needed object file.

```
gcc -c -fPIC calc_mean.c -o calc_mean.o
```

For some reason, gcc says:

```
cc1: warning: -fPIC ignored for target (all code is position independent)
```

It looks like -fPIC is not necessary on x86, but all manuals say, it's needed, so I use it too.

Now, the shared library is created

```
gcc -shared -Wl,-soname,libmean.so.1 -o libmean.so.1.0.1  calc_mean.o
```

Note: the library must start with the three letter **lib**.

The programm using the library

This is the program that uses the **calc_mean** library. Once, we will link it against the static library and once against the shared library.

main.c

```
#include <stdio.h>
#include "calc_mean.h"

int main(int argc, char* argv[]) {

  double v1, v2, m;
  v1 = 5.2;
  v2 = 7.9;

  m  = mean(v1, v2);

  printf("The mean of %3.2f and %3.2f is %3.2f\n", v1, v2, m);

  return 0;
}
```

Linking against static library

```
gcc -static main.c -L. -lmean -o statically_linked
```

Note: the first three letters (the **lib**) must not be specified, as well as the suffix (**.a**)

Linking against shared library

```
gcc main.c -o dynamically_linked -L. -lmean
```

Note: the first three letters (the **lib**) must not be specified, as well as the suffix (**.so**)

Executing the dynamically linked programm

```
LD_LIBRARY_PATH=.
./dynamically_linked
```

Thanks

Thanks to **Donn Morrison** who helped me improve this page.
