# C Development Environment

# Life Cycle of a C program Development

| | |
|---|---|
| Designing | [Your Brain™] |
| Editing | [xemacs , vi ?!] |
| Compiling | [gcc] |
| Code Checking | [lint, smash] |
| Testing / Debugging | [gdb , gof printf] |
| Purifying | [e-fence, valgrind , purify] |
| Profiling & Optimizing | [gcov , gprof] |
| Packaging | [deb , rpm] |

# Compiling with GCC

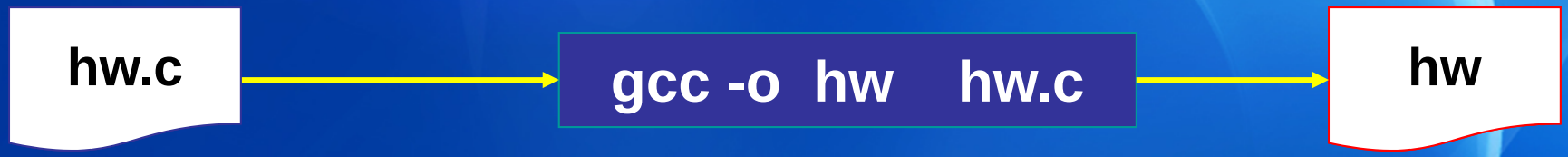
```
#include <stdio.h>

int main ( )
{
   printf ("hi\n");
}
```

*Source code File  hw.c*


*Executable File*

**Let's get it done the simplest way first**

# Compiling with GCC in one step



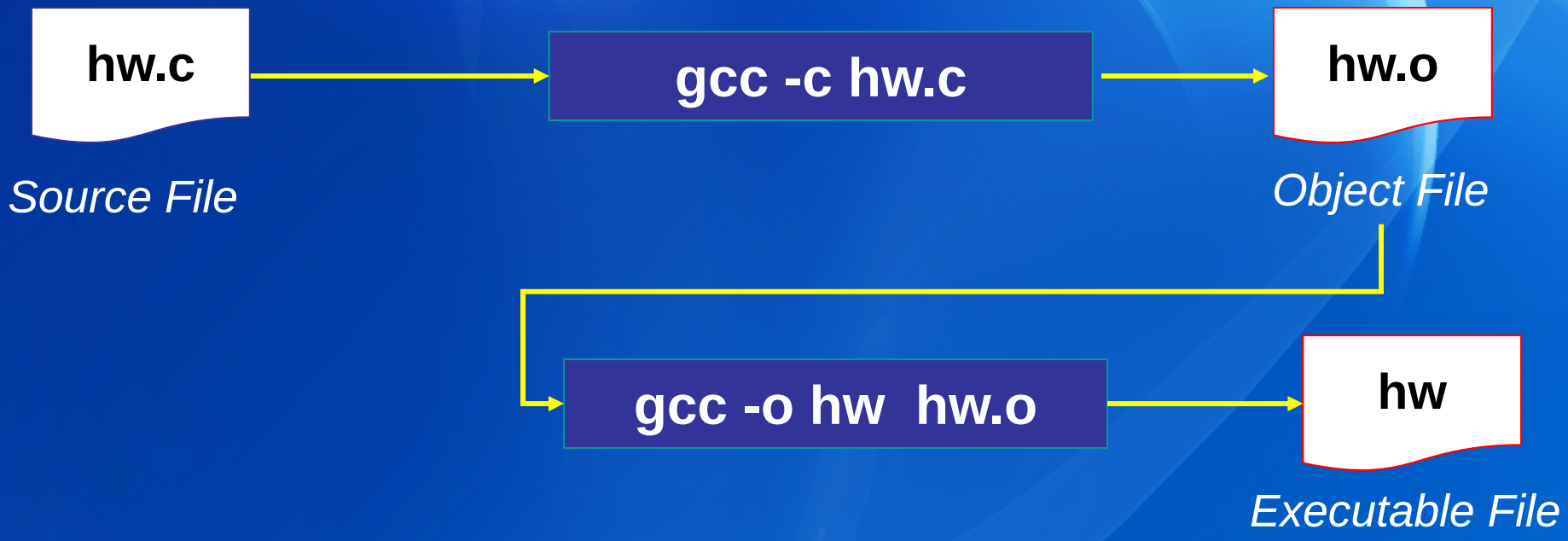

*Source File*

*Executable File*

```
[tux:~/hw]:  GCC  -o  hw hw.c

[tux:~/hw]:  ls  -l  hw
-rwxr-xr-x   1 gaspara  voicedb    13877 Aug  8 15:05 hw

[tux:~/hw]:  ./hw
hi
```

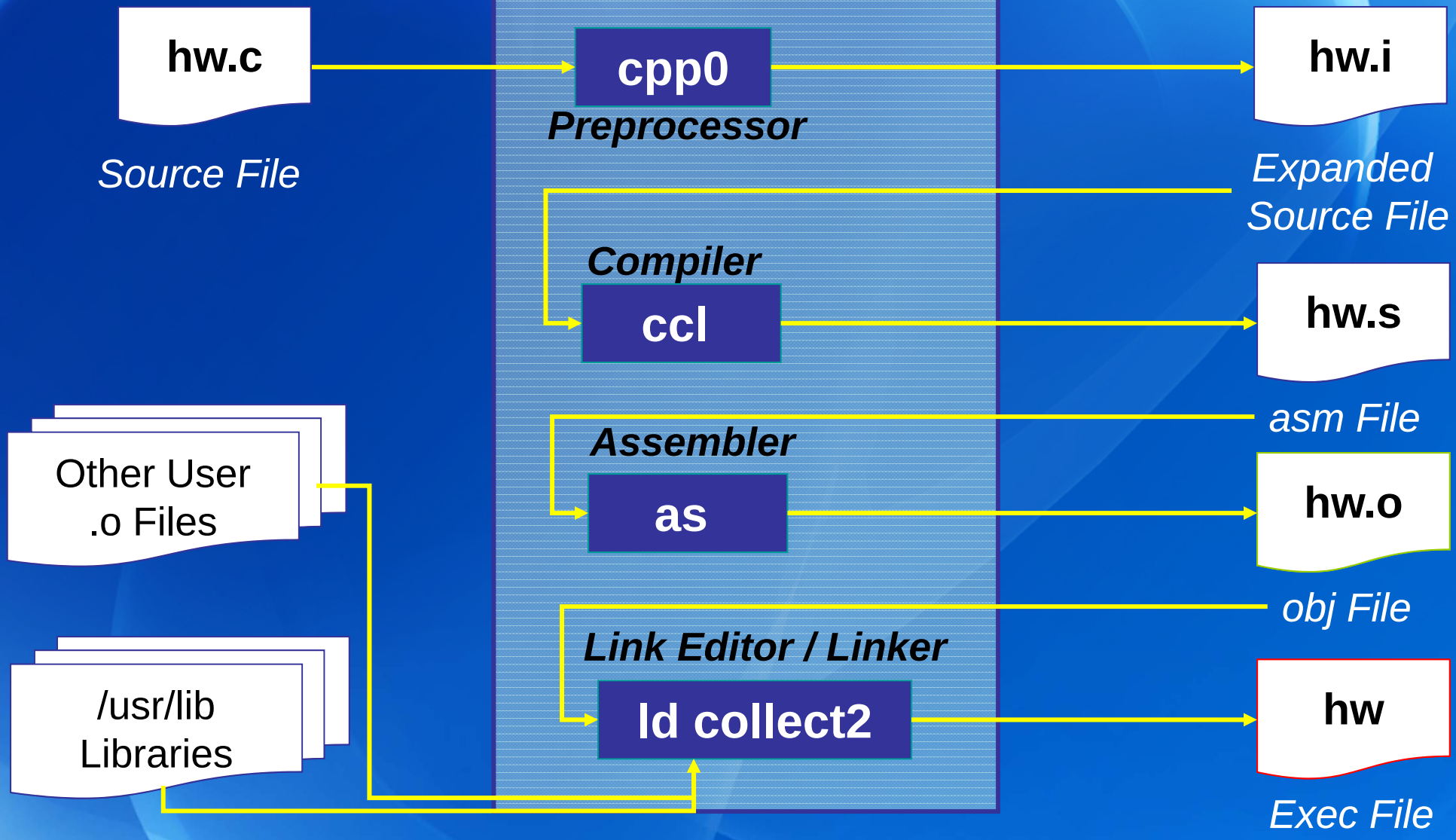
```
[tux:~/hw]:  file  hw
hw: ELF 32-bit LSB executable, Intel 80386, version 1, dynamically linked (uses shared
libs), not stripped
[tux:~/hw]:  more /usr/share/magic
```

# Compiling with GCC in two steps




```
[tux:~/hw]:  GCC  -c  hw.c
[tux:~/hw]:  ls  -l  hw.o
[tux:~/hw]:  file  hw.o
[tux:~/hw]:  GCC  -o hw2  hw.o
[tux:~/hw]:  ls -l  hw  hw2
[tux:~/hw]:  file hw2
```

# What's behind GCC ?



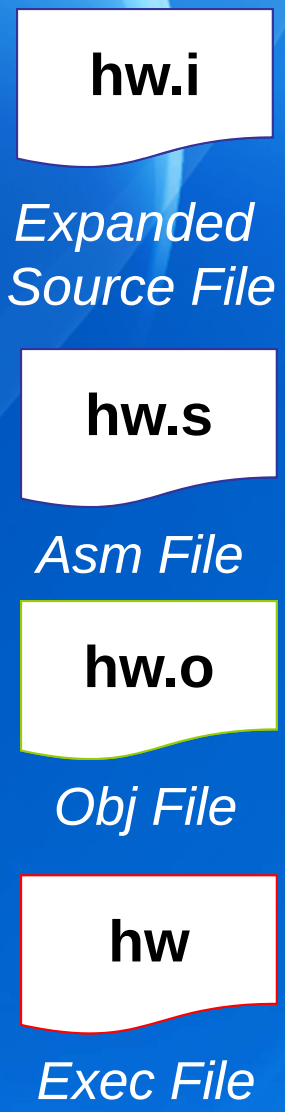
# More Informations about GCC ?

Different Types of Files' Extensions are detailed in

**man gcc**

Similarly, options to stop GCC at any step of the source code processing chain are available

| | |
|---|---|
| -c | Stops after hw.o |
| -S | Stops after hw.s |
| -E | Stops after hw.i |



# Precompiling

```
#include <stdio.h>

int main ( )
{
   printf ("hi\n");
}
```

*Source code File hw.c*


GCC -E hw.c

**Contents of**
/usr/include/stdio.h

```
int main ( )
{
   printf ("hi\n");
}
```

*Expanded Source File hw.i*

**hw.i**

*Expanded Source File*

**hw.s**

*Asm File*

**hw.o**

*Obj File*

**hw**

*Exec File*

# Compiling

**Contents of**
/usr/include/stdio.h

```
int main ( )
{
   printf ("hi\n");
}
```

*Expanded Source File hw.i*

```
        (…)
main:
        pushl   %ebp
        movl    %esp, %ebp
        subl    $8, %esp
        subl    $12, %esp
        pushl   $.LC0
        call    printf
        addl    $16, %esp
        (…)
```

*Asm Source File hw.s*

**ccl**

GCC -S hw.c

**hw.i**

*Expanded Source File*

**hw.s**

*Asm File*

**hw.o**

*Obj File*

**hw**

*Exec File*

# Assembling

```
        (…)
main:
        pushl   %ebp
        movl    %esp, %ebp
        subl    $8, %esp
        subl    $12, %esp
        pushl   $.LC0
        call    printf
        addl    $16, %esp
        (…)
```

*Asm Source File hw.s*

GCC -c hw.c

**as**

**hw.o**

*Obj File*

**hw.i**

*Expanded Source File*

**hw.s**

*Asm File*

**hw.o**

*Obj File*

**hw**

*Exec File*

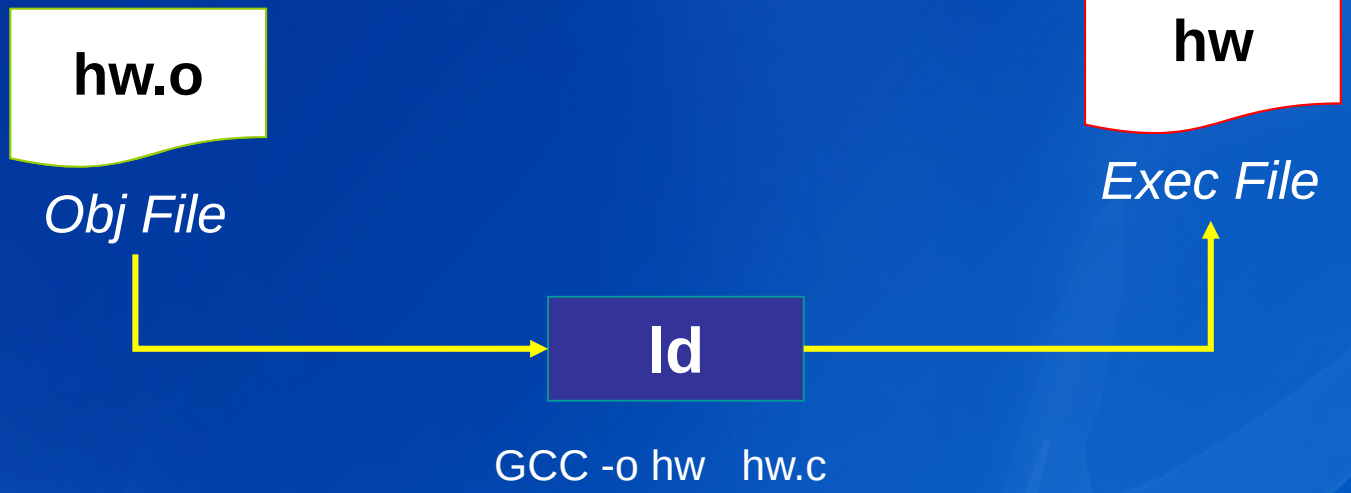
Relocatable Object File

= assembled asm code (architecture dependent)

= ELF32_i386 format (linux)

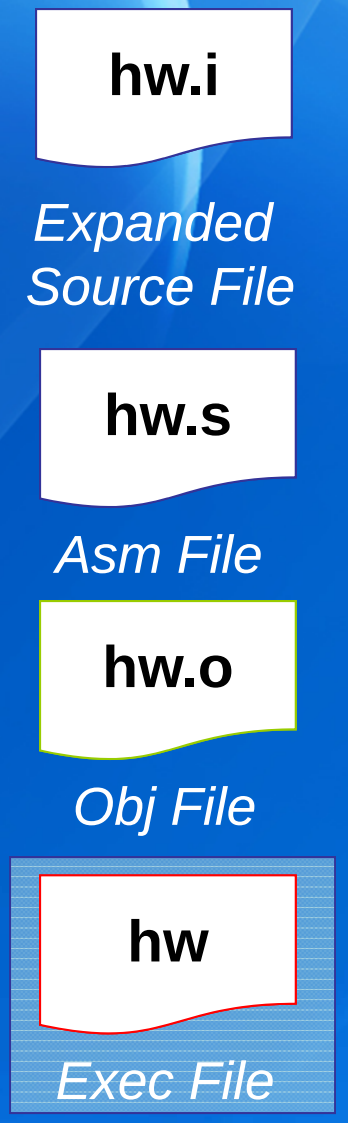
Ready for link w/ other obj to form an executable file

# Linking

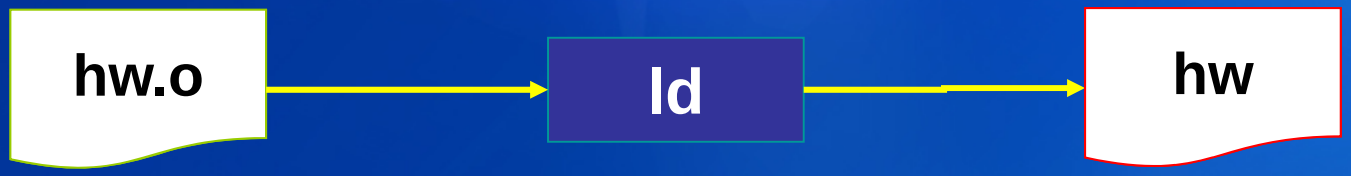



Gathering altogether

Your Object Files
Object Files from libraries
The C runtime code



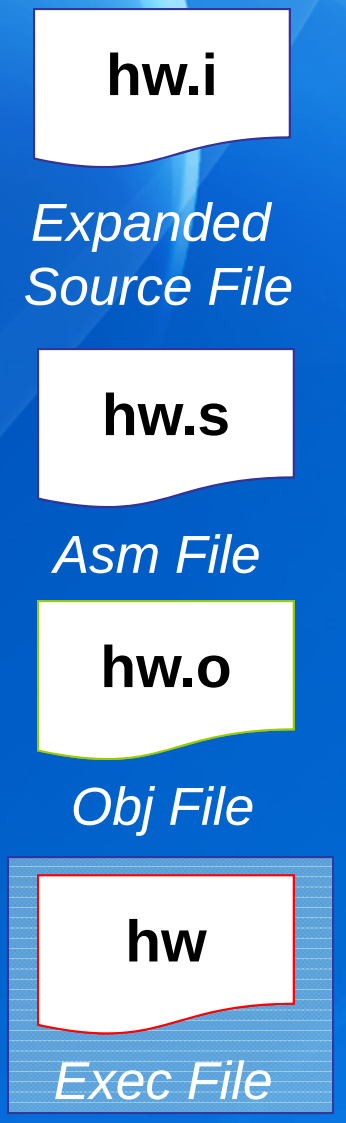
# Linking




*Obj File* *Exec File*

**A taste of what linking is really about… (edited)**

**ld -m** elf_i386
**-dynamic-linker** /lib/**ld-linux.so.2**
**-o hw**
/usr/lib/crt1.o /usr/lib/**crti.o**
/usr/lib/gcc-lib/i386-redhat-linux/2.96/**crtbegin.o**
**-L**/usr/lib/gcc-lib/i386-redhat-linux/2.96
**-L**/usr/lib/
**hw.o**
**-lgcc -lc -lgcc**
/usr/lib/gcc-lib/i386-redhat-linux/2.96/**crtend.o**
/usr/lib/gcc-lib/i386-redhat-linux/2.96/../../../**crtn.o**



# Blank Slide…