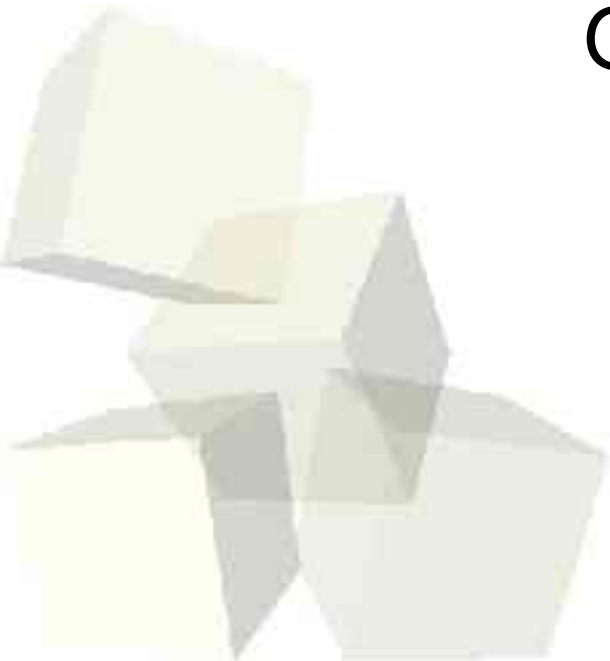




GNU Compiler Collection





Why gcc?

- Powerful
- Advanced
- Many languages
- Works on various platforms
- Large number of architectures
- bit programs (bit are outdated)
- Easy to use in a network environment





Compiling C programs

■ hello_world.c

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

```
int main ()
```

```
{
```

```
    printf ("Hello World!\n");
```

```
    return ;
```

```
}
```

■ Compiling

```
$ gcc hello_world.c
```





Compiling C++ programs

■ hello_world.cpp

```
#include <iostream>
using namespace std;

int main ()
{
    cout << "Hello World!" << endl;
}
```

■ Compiling

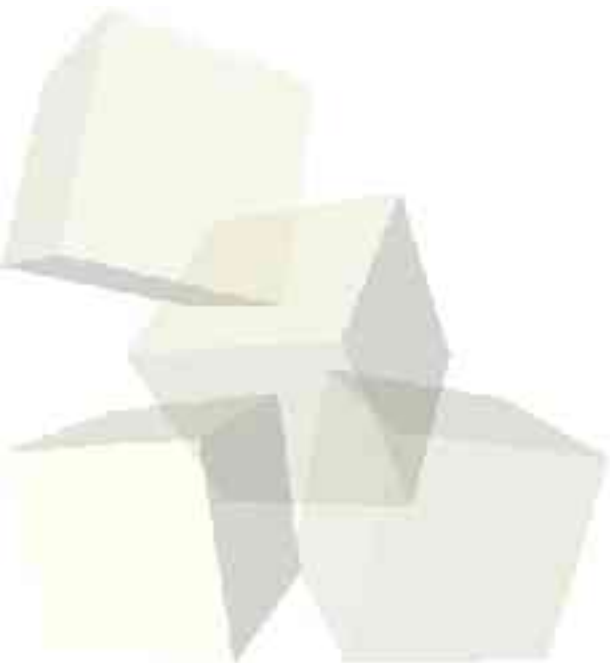
```
$ g++ hello_world.cpp
```





Installing gcc and g++

- Open package manager
- Install packages "gcc" and "g++"
- Thats it! start using





Debugging with gdb

■ Advantages

- ◆ Multi-platform and Mutli-architecture
- ◆ advanced feature like calling functions at will
- ◆ Remote debugging

■ Programs should be compiled with -g option to gcc

■ Demo using gdb with hello_world.c

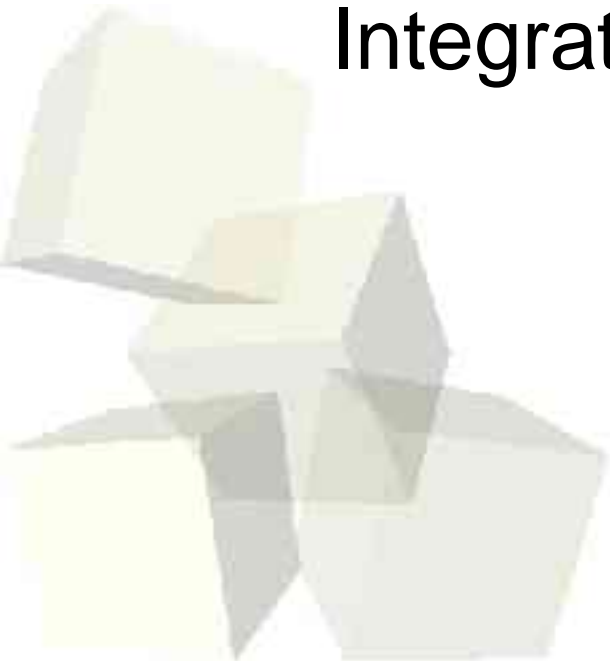
■ Graphical front-ends like xgdb and ddd

■ Installing gdb – "gdb" in package manager





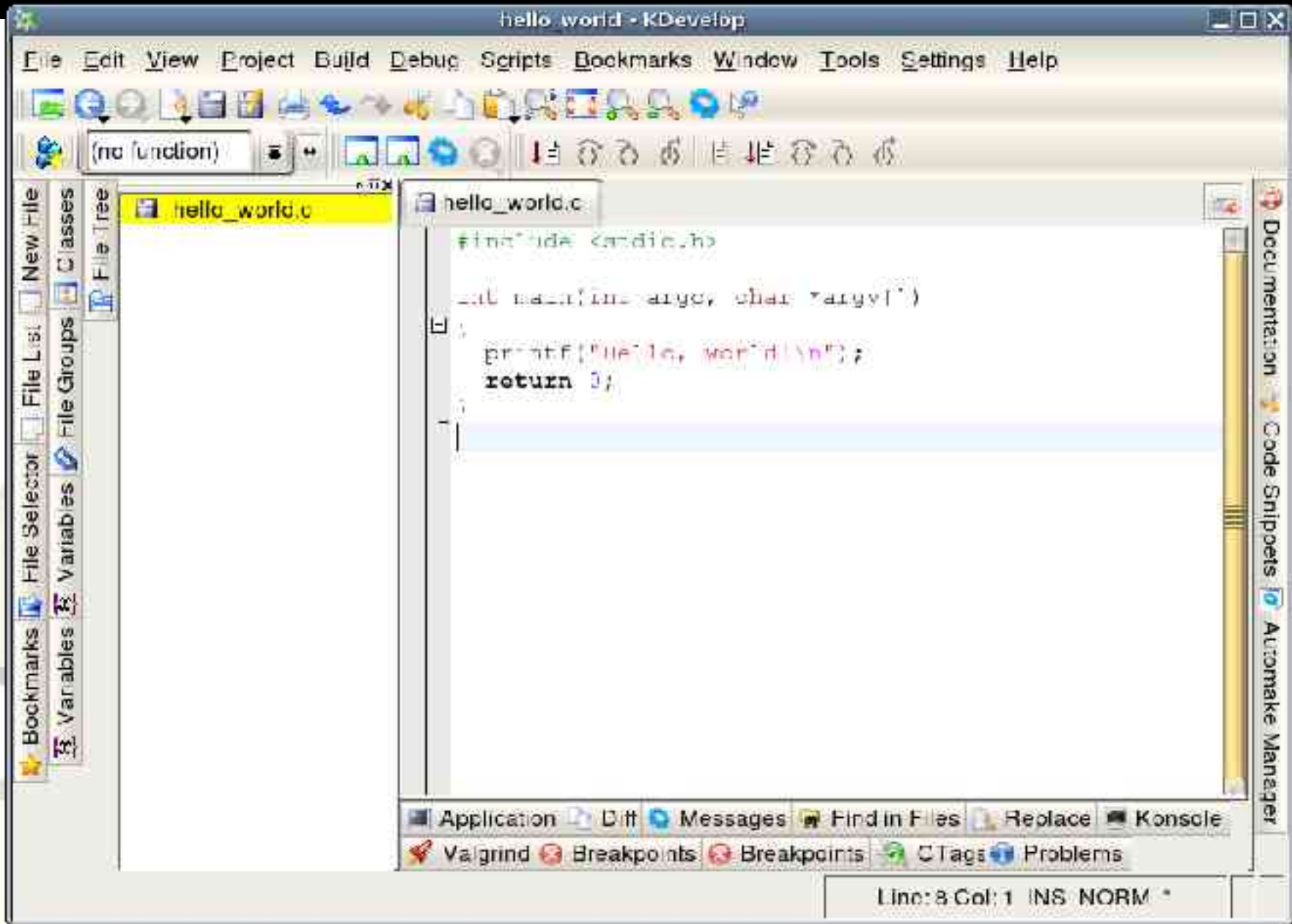
Integrated Development Environments



Project Creation with Kdevelop

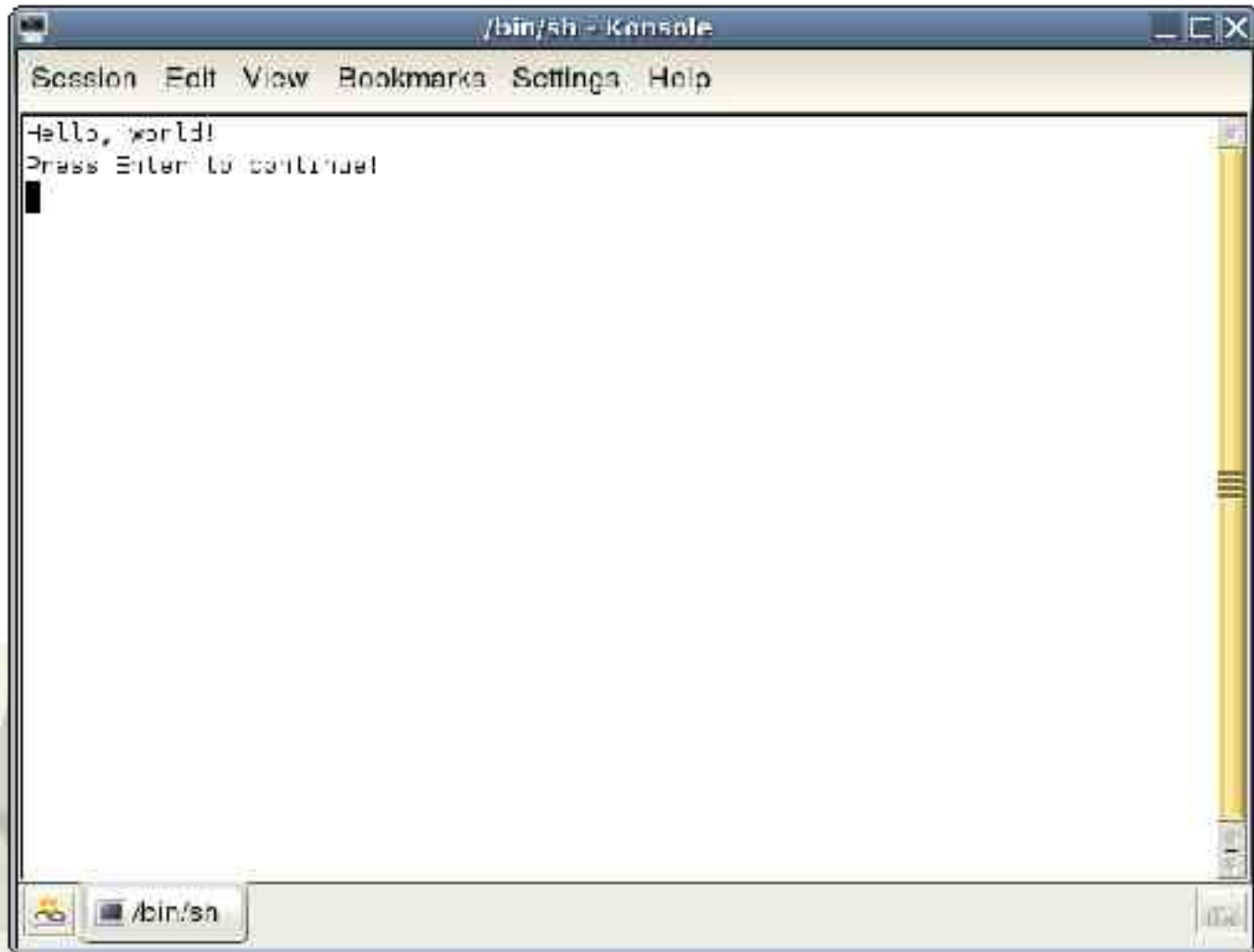


Hello World in KDevelop

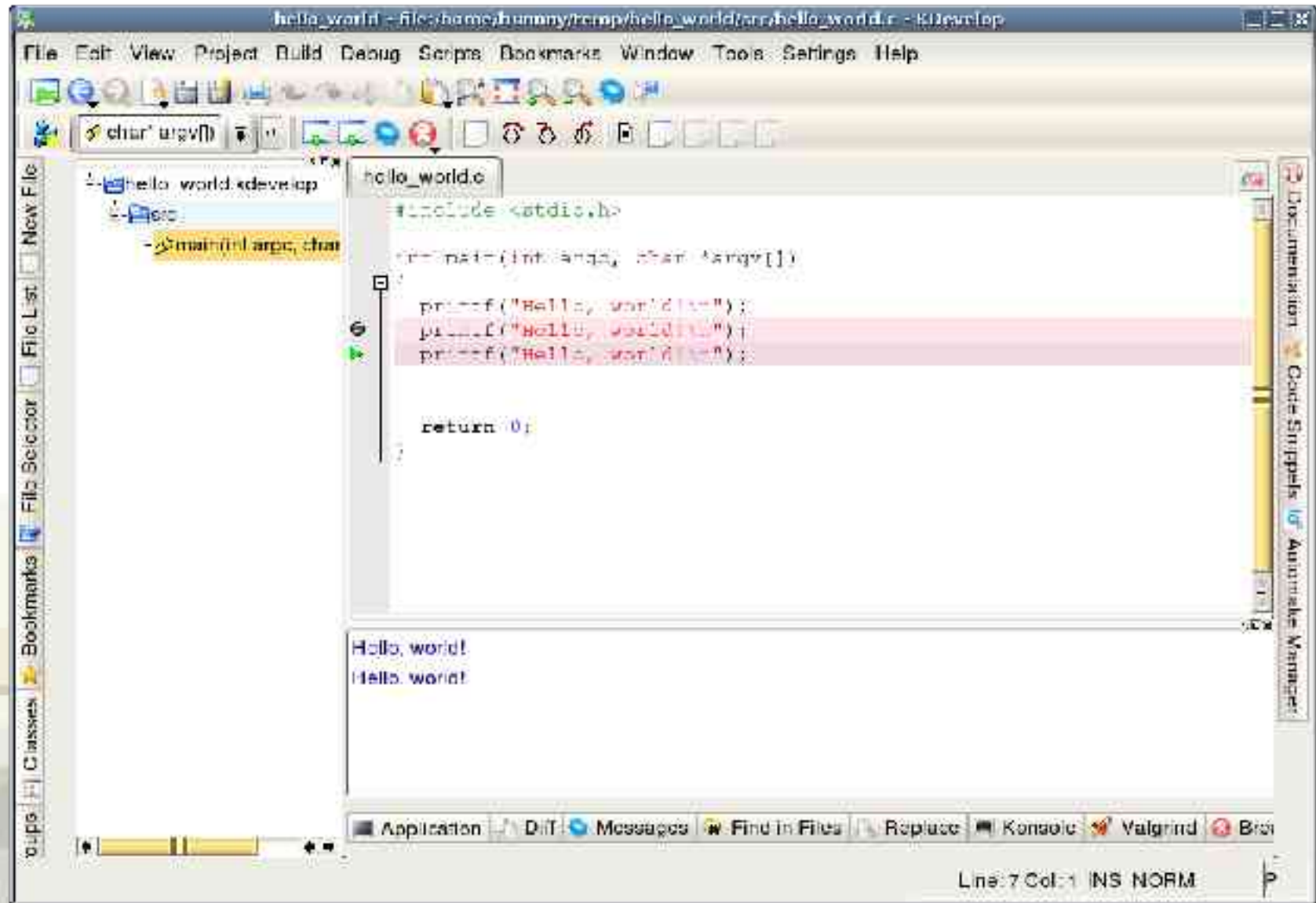




Execution

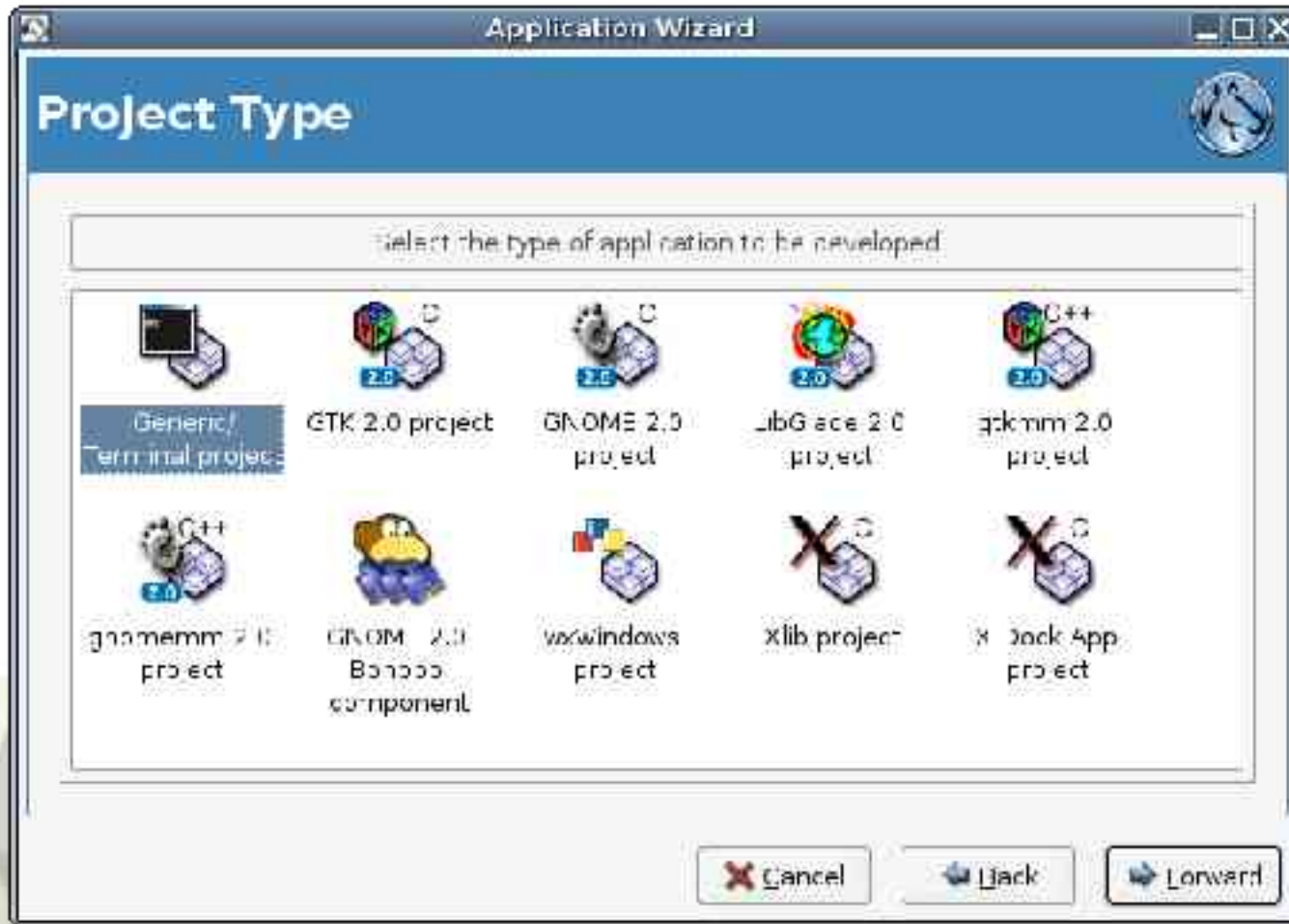


Debugging



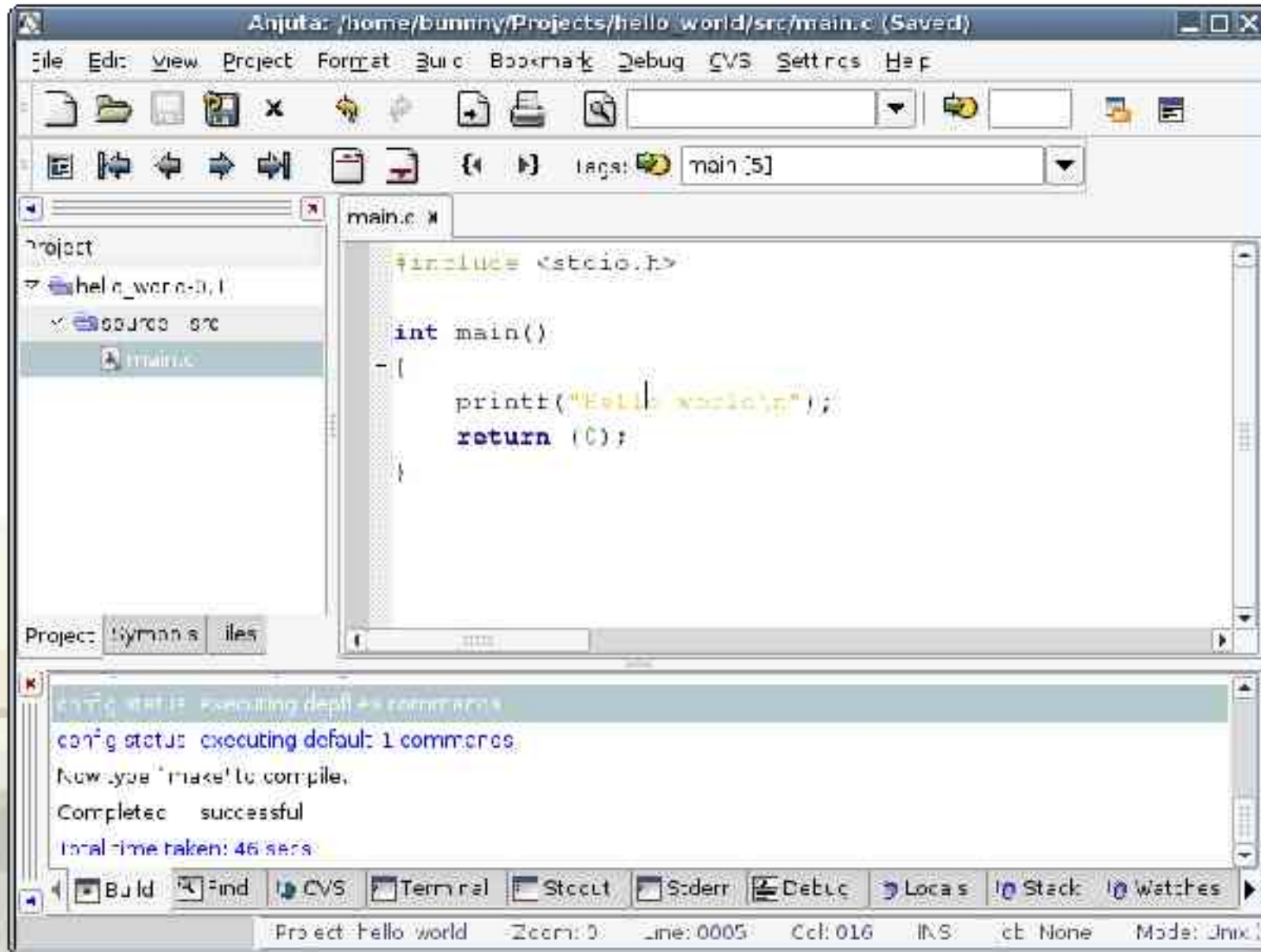


Project Creation with Anjuta

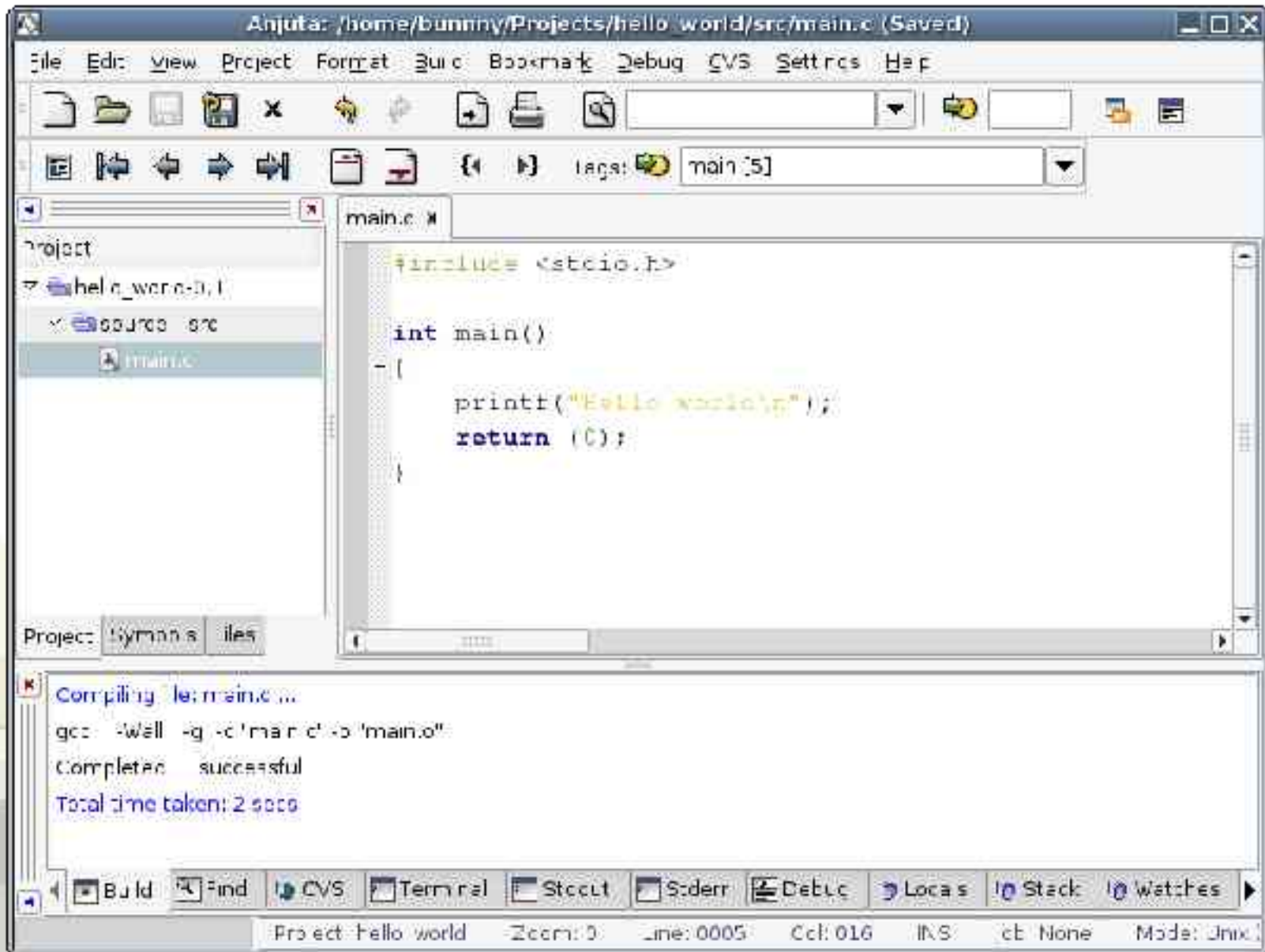




Hello World



Compiling



Anjuta: /home/bunny/Projects/hello_world/src/main.c (Saved)

File Edit View Project Format Build Bookmark Debug CVS Settings Help

Project: /home/bunny/Projects/hello_world
src
main.c

```
#include <stdio.h>

int main()
- {
    printf("Hello world!\n");
    return (0);
}
```

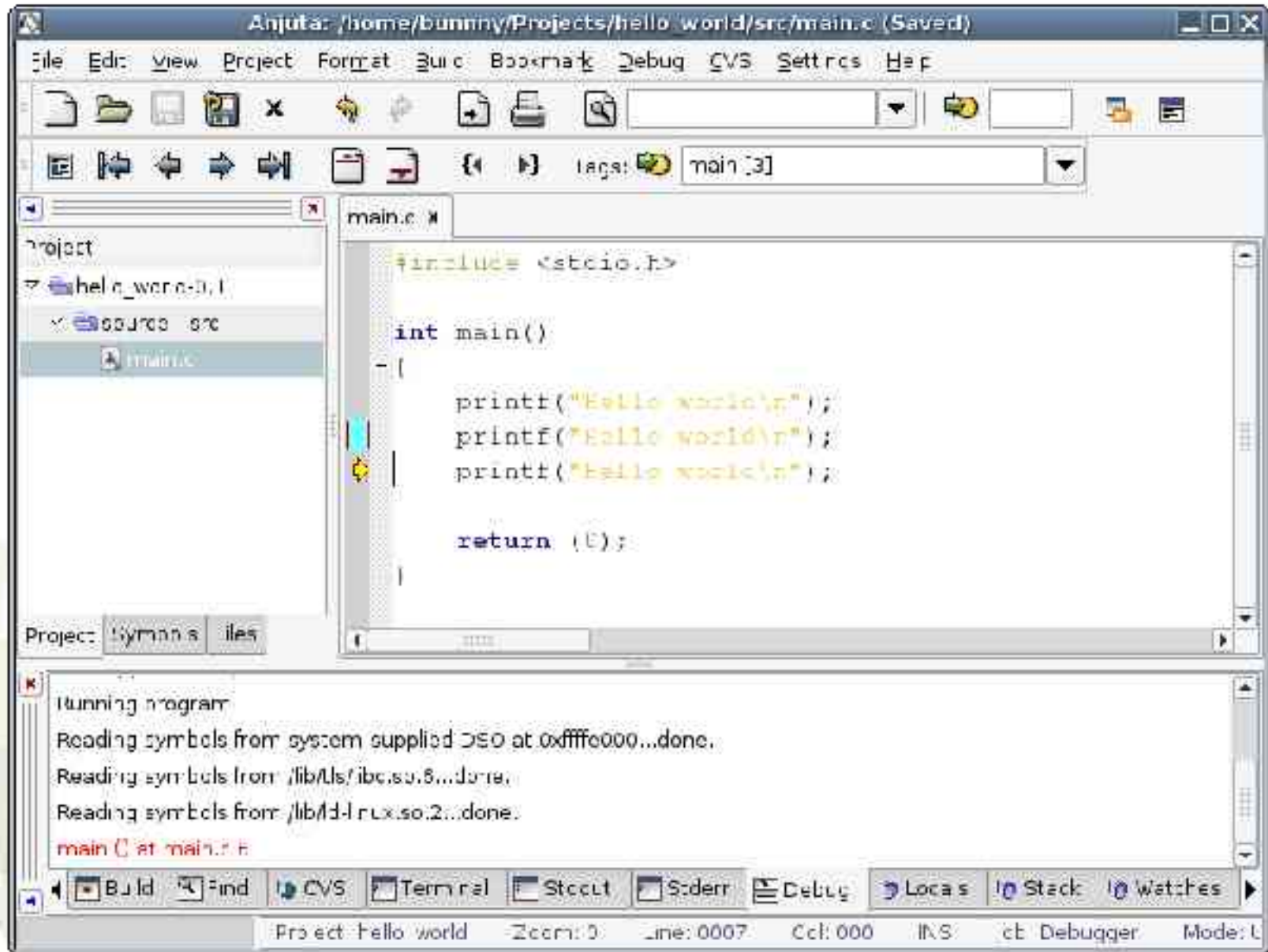
Project: /home/bunny/Projects/hello_world

Compiling /home/bunny/Projects/hello_world/src/main.c ...
gcc -Wall -g -c 'main.c' -o 'main.o'
Completed successful
Total time taken: 2 secs

Build Find CVS Terminal Stdout Stderr Debug Locals Stack Watches

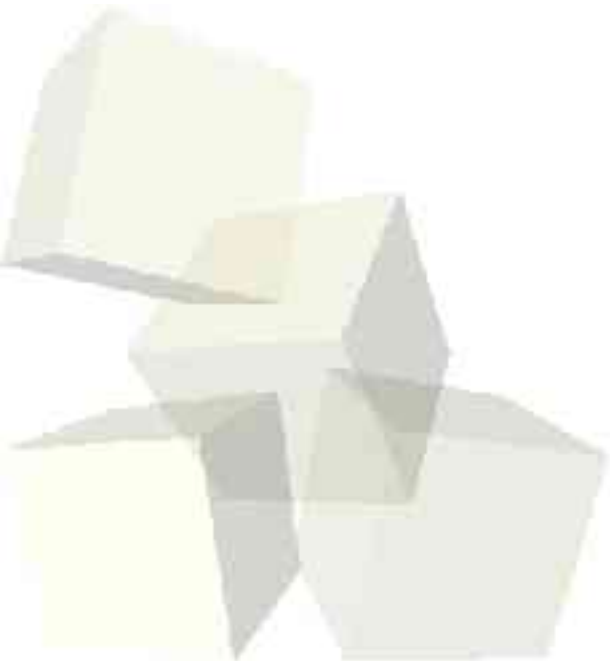
Project: hello_world | Zoom: 0 | Line: 0005 | Col: 016 | INS | cb: None | Mode: Unix

Debugging

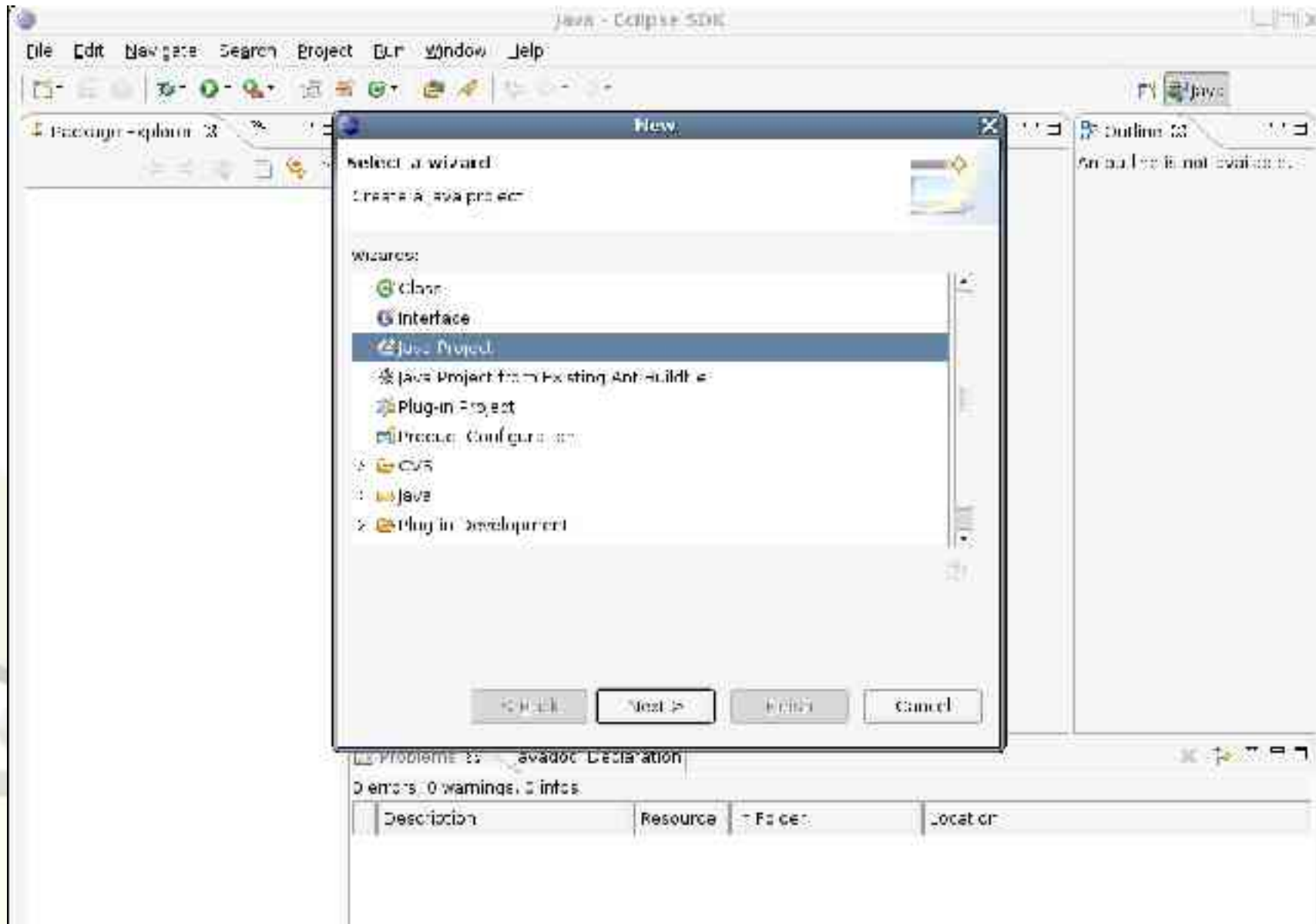




Eclipse



Project Creation with Eclipse



Hello World

The screenshot shows the Eclipse IDE interface. The main editor displays the following Java code:

```
public class Hello_world {  
    public static void main(String[] args) {  
        System.out.println("Hello World!");  
    }  
}
```

A context menu is open over the `System.out.println("Hello World!");` line, with the option "Change to System (java.lang)" selected. The menu items include:

- Create local variable 'system'
- Create field 'system'
- Create parameter 'system'
- Create constant 'system'
- Create class 'system'
- Create interface 'system'
- Rename in file (Ctrl+Z R direct access)
- Change to System (java.lang)
- Change to SystemClass (java.lang)
- Change to SystemSigner (sun.security.pro...

The Console window at the bottom shows the following output:

```
terminated> hello_world [Java Application] /usr/java/jdk-1.6.0_03/bin/java: 18 Sep 2010 10:38:45 AM  
Exception in thread "main" java.lang.Error: Unresolved compilation problem:  
    system cannot be resolved  
  
    at Hello_world.main(Hello_world.java:6)
```

Debugging

