## Specific Issues moving from GCC 2.95.3 to GCC 4.x #

These are the most common errors that emerge with gcc 4.

### error: lvalue required as left operand of assignment#

Older gcc versions contained an extension called cast-as-lvalue which would allow assignment of a variable as a different type as in:

```
int foo;
float bar;
(float)foo = bar;
```

This extension is deprecated because it's dangerous. The correct way get behaviour like above is to use a union. If it's pointers being assigned, just cast the right side to the type of the left, instead of the other way around.

Note that accessing a member of a cast struct is still valid. For example:

```
((some_struct *)(myStruct))->member = foo;      // this is valid
(int)((some_struct *)(myStruct))->member = foo; // this will produce an error
```

### error: label at end of compound statement#

This one is very simple, and usually occurs with a switch statement like:

```
switch (foo) {
case 1:
// do stuff
      break;
default:
}
```

Simply remove "default:" or add "break;" after it, and gcc will be happy.

### file.c:456: error: conflicting types for 'some_function'#

### file.c:123: error: previous implicit declaration of 'some_function' was here#

or

### file.c:49: error: expected '=', ',', ';', 'asm' or 'attribute' before 'some_function' #

Solution: Didn't your programming teacher tell you to always make function prototypes? He/she had good reason. It's also best to have them at the top, global space of the file, not in weird places. Yes, I have seen a function declared within another function, just before it gets called. Try to avoid that.