Q: When I try to compile my C++ program which contains this header file:

```
 typedef enum {
        aaa,
        bbb,
        ccc
} Type;

class myClass {
private:
        Type myType;
public:
        void setType(Type newType);
};
```

I get the link error:

/bin/ld:Undefined symbols:_setType__7myClass3$_1

and the compilation stops. The same program would compile fine on a non-NEXTSTEP c++ compiler. What's the problem?

A: There is indeed a bug in the current C++ compiler, but the workaround is easy.

The problem is with the declaration:

```
typedef enum {
        aaa,
        bbb,
        ccc
} Type;
```

This causes problems with C++'s name mangling, since this enum has no tag. The current compiler does not know how to encode this type, so it simply uses an integer

for the number of the unencodable type. There is no reason to think that this matches the encoding used in some other file which may have had a different number of unencodable types or the same types but in a different order. This is indeed what was happening in your case, and explains why the link failed.

You should just use this declaration instead:

```
enum Type {
        aaa,
        bbb,
        ccc
};
```

In C++, a typedef is automatically published for each struct, union, or enum with the same name as the tag, so you can continue to use "Type" rather than "enum Type".

This bug will be fixed in a future release of the C++ compiler.

QA780

Valid for 2.0, 3.0, 3.1