Q: How does one mix C and Objective-C code? How can one call Objective-C from C?

A: Just do it! C functions can freely send Objective-C messages, and Objective-C methods can freely call C functions. For an example, look at **main()** of one of the applications in /NextDeveloper/Examples.

If you place Objective-C code in a ".c" file, you need to compile the file with the Objective -C option. The simplest way to do this is to change the ".c" suffix to ".m", and then add the file name under ".m (other)" in InterfaceBuilder's Project Inspector. If it's not possible to change the suffix, compile the file using the **-ObjC** compiler flag.

You can place any C code in a ".m" file without any special handling. The C code needs to extern id's as per regular C scoping rules.

Only within Objective-C methods can you directly access instance variables and the hidden variable "self." You must pass an object to a C function that needs access to it. For example, this C function returns the width of a View:

```
NXCoord widthOfView(View *self)
{
   NXRect b;
   [self getBounds:&b];
  return b.size.width;
}
```

You may access **public** instance variables like this:

```
void setNumber(MyObject *self,int newValue)
{
    self->number = newValue;
}
```

Remember that all instance variables are considered private unless explicitly declared public. However, within the confines of a class implementation, all instance variables of that class are considered public. So in the implementation for MyControl (a subclass of Control), you

could write a C function to set the tag of a MyControl object like this:

```
void setTag(MyControl *self,int newTag)
{
    self->tag = newTag;
}
```

QA27

Valid for 1.0, 2.0, 3.0