

(NeXT Tip #28) C, PostScript and C++

Christopher Lane (*lane[at]CAMIS.Stanford.EDU*)
Mon, 12 Apr 1993 14:56:24 -0700 (PDT)

NeXT applications do not have to be coded solely in Objective-C, you can also include files containing subroutines in vanilla C, PostScript and C++.

If you have vanilla C routines (ANSI or otherwise) in a *.c file, you can include them in your project by dropping them into the 'Other Sources' suitcase in ProjectBuilder. You'll need a *.h file that declares the routines to import into your *.m Objective-C files so that you can call the C routines. The ProjectBuilder and Makefile recognize the *.c extension correctly.

If you want to write your graphics routines directly in PostScript, or you have an algorithm that naturally fits into the PostScript RPN programming model (very hard to picture), you can include PostScript routines in your project as 'pswraps'. These are PostScript routines that are 'wrapped' with C calling structures using a special precompiler. Typically, in a *.psw file you define your PostScript routine using 'defineps' and 'endps':

```
defineps PSclearToGray(float gray)
currentgray
gray setgray
clippath
fill
setgray
endps
```

You also include these in the 'Other Sources' suitcase in ProjectBuilder and it will do the right thing based on the *.psw extension -- precompiling the *.psw files first to generate the necessary C *.h files needed to compile other files that import the *.h file and call the routines. (See the online documentation for details.) The routines that you define in a *.psw should only be called under an explicit or implicit 'lockFocus' if they do graphics.

It is also possible to include (Objective-)C and PostScript in the SAME source file via the *.pswm extension/semantics -- since this complicates matters and there are so few examples of this available, I don't recommend it.

Using files containing C++ subroutines is also possible but can be tricky. There currently is no standard C++ library on the NeXTs so you'll only be able to use routines that don't do C++ style I/O. (Either use standard 'C' I/O routines like 'printf' or have your Objective-C code do the I/O through the window system.) The default Makefiles do not have rules for the '*.cc' extension -- the easiest way to include C++ is to use either *.m or *.c extensions for your C++ files and add a Makefile.preamble file with the line:

```
CFLAGS = -ObjC++
```

This lets the compiler know you're mixing C++ and Objective-C.

However, if you want to use BOTH PostScript routines AND C++, it gets really tricky as the 'pswraps' precompiler will do the wrong thing if you set -ObjC++ in CFLAGS. To get around this, the best solution I've been able to work out is using both a Makefile.preamble and a Makefile.postamble with the following:

Makefile.preamble:

```
OBJCFLAGS = -ObjC++
CFILES = $(OTHERLINKED:.cc=.o)
OTHERLINKED =
```

Makefile.postamble:

.SUFFIXES: .cc

.cc.o:

```
$(CC) $(CFLAGS) $(CCFLAGS) -c $*.cc -o $(OFILE_DIR)/$*.o
```

This will allow PostScript wraps with C++ and let you use the *.cc extension for your C++ files -- just drop them into 'Other Sources' in ProjectBuilder. (See the online documentation for more details about C++.)

The -ObjC++ flag is still needed (in OBJCFLAGS) so that your controlling Objective-C routines that do C++ operations will compile correctly. I included but didn't set the 'CCFLAGS' variable so you can pass switches to the compiler for C++ files via setting the CCFLAGS variable in Makefile.preamble.

I've a simple example program in ~lane/Programming/PrimeSpiral++ that uses both PostScript and C++, along with Objective-C, if you're interested in exploring mixing programming languages in a NeXT application.

- Christopher