Overview

I wrote this program as learning example on how to convert a straight-line Unix-C program into a finished NextStep/Objective-C Application.   The actual conversion effort turned out to be much easier than I anticipated.

I had already written a command line program that calculated a sphere and output it as PostScript-code to stdout (sphere [parameter list] > output.ps).   I was able to convert this program into a Next App in about 4 hours.

IB & Objective-C

The first step was to create the main window in Interface Builder -- using sliders and buttons in place of the Unix command-line parameters.

Next, I subclassed object to create a SphereDrawerObject which contains all of the drawing & shading routines from my original Unix-C code (some functions still need to be turned into methods, but the code works as is).

I then created a sub-class of view called SphereView and overrode the initFrame: and drawSelf:: methods.   The drawSelf:: method messages the SphereDrawerObject which outputs postscript code into the view.   Because this happens INSIDE drawSelf:: rather than locking-focus & drawing INTO the instance of SphereView, it makes the print-routine trivial to implement (click&drag from the print menu-button to the view and connect the printPSCode: target method).

Last (but far from least) I subclassed object to create my controller (SphereControl).   All of the program variables are stored in this object and methods to read and write each instance variable are implemented.   Also, all of the program controls are routed through this object.

The Sphere Program

The light source is fixed in the Z-direction and the program uses lambert's law of cosines to calculate the shading.

Moving the sliders for theta and phi actually move the eyepoint around the sphere, rather than rotating the sphere (only apparent if shading is turned on).

Be sure to try low settings on the latitude and longitude sliders (interesting non-spherical shapes are generated by the sphere algorithm).

Special thanks to Tyler Gingrich for helping teach me many of the basics about programming on the NeXT.