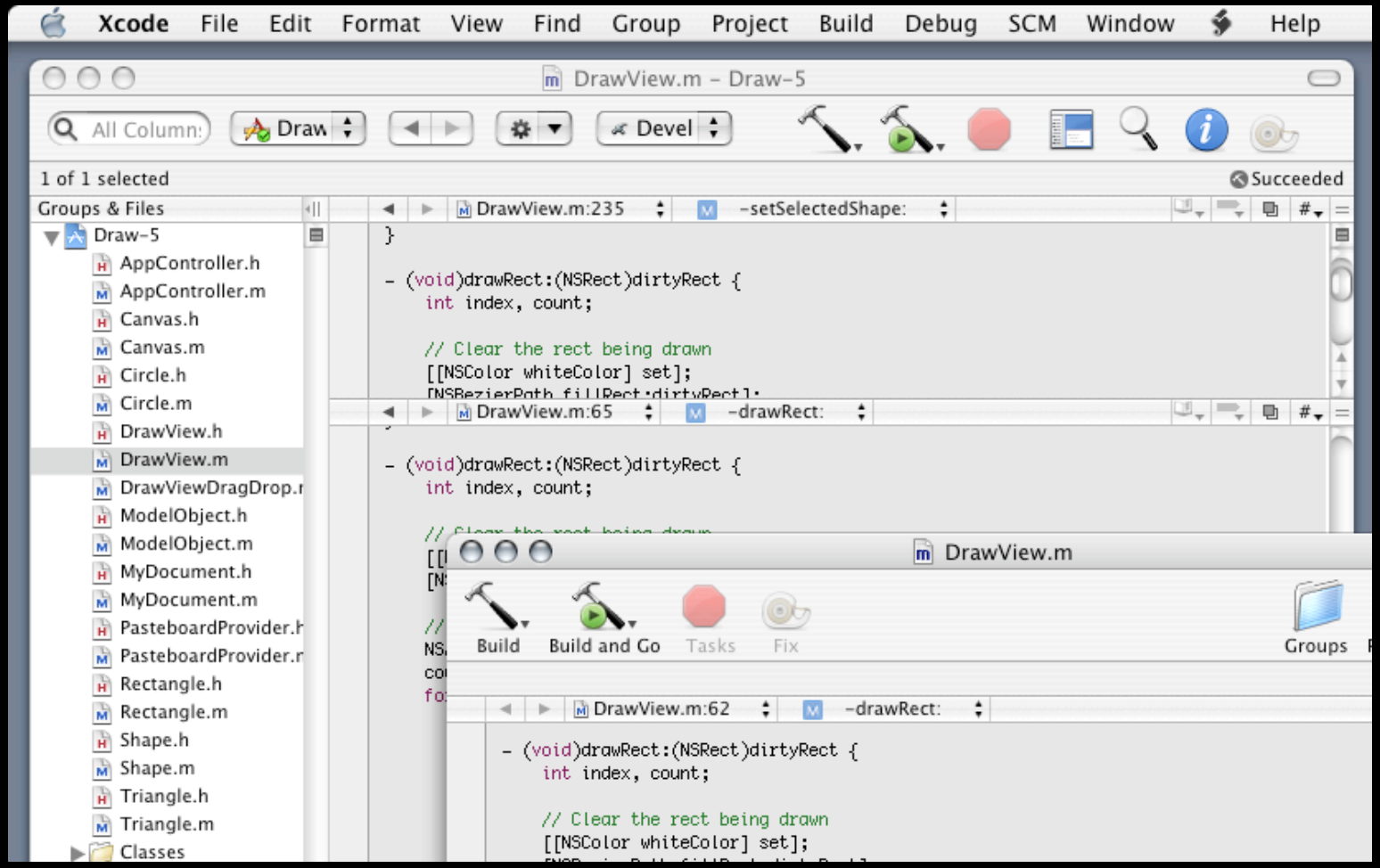
CS193E
Lecture 17

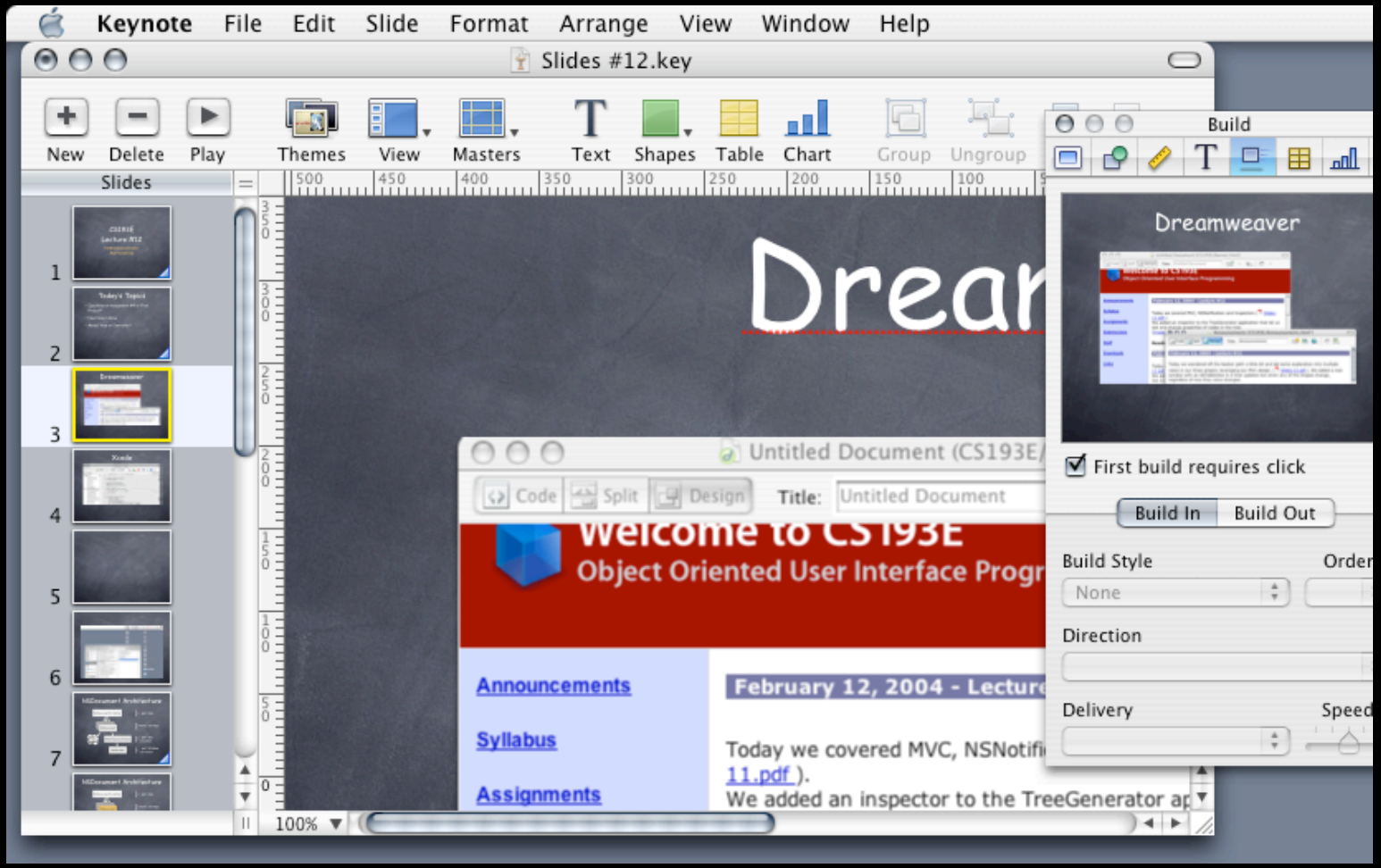
Multiple Document Windows
OpenGL & Cocoa

Multiple Views onto a Model

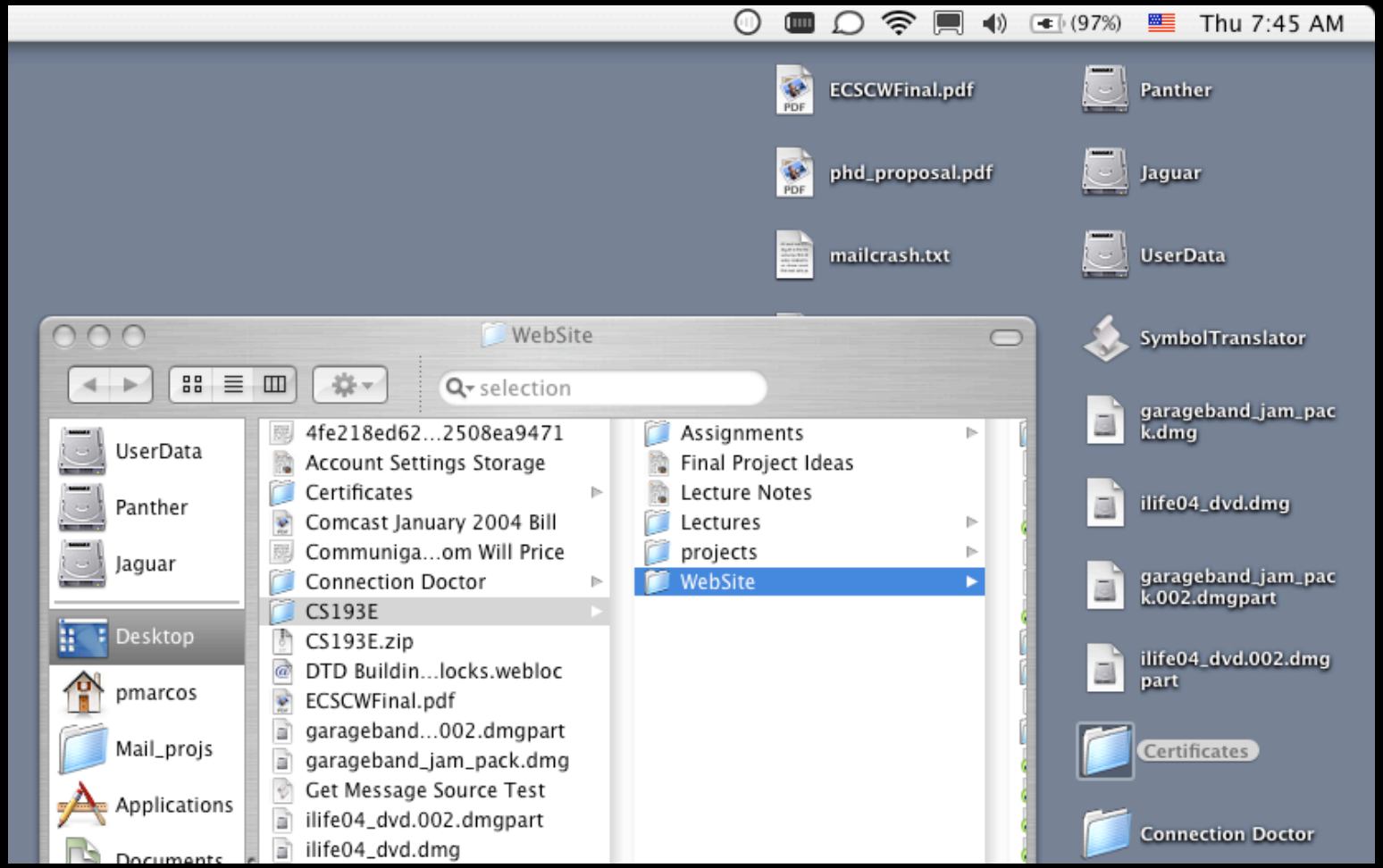
Xcode


Xcode File Edit Format View Find Group Project Build Debug SCM Window Help
DrawView.m – Draw-5
All Column:
Draw
Devel
1 of 1 selected
Succeeded
Groups & Files
Draw-5
AppController.h
AppController.m
Canvas.h
Canvas.m
Circle.h
Circle.m
DrawView.h
DrawView.m
DrawViewDragDrop.r
ModelObject.h
ModelObject.m
MyDocument.h
MyDocument.m
PasteboardProvider.h
PasteboardProvider.r
Rectangle.h
Rectangle.m
Shape.h
Shape.m
Triangle.h
Triangle.m
Classes
DrawView.m:235
-setSelectedShape:
}
- (void)drawRect:(NSRect)dirtyRect {
int index, count;
// Clear the rect being drawn
[[NSColor whiteColor] set];
DrawView.m:65
-drawRect:
- (void)drawRect:(NSRect)dirtyRect {
int index, count;
DrawView.m
Build
Build and Go
Tasks
Fix
Groups
DrawView.m:62
-drawRect:
- (void)drawRect:(NSRect)dirtyRect {
int index, count;
// Clear the rect being drawn
[[NSColor whiteColor] set];

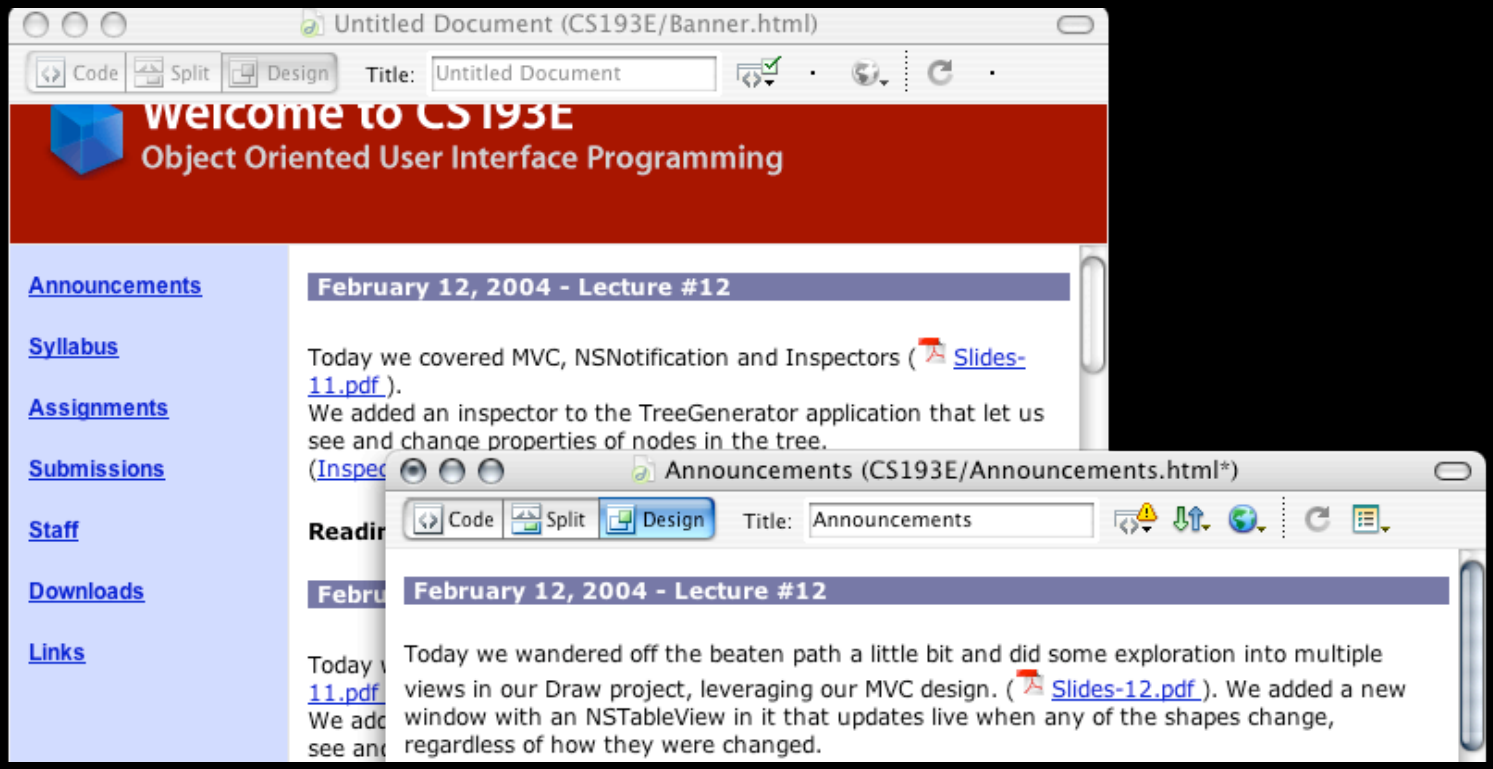
Keynote


Keynote File Edit Slide Format Arrange View Window Help
Slides #12.key
New Delete Play Themes View Masters Text Shapes Table Chart Group Ungroup
Slides
Build
Dreamweaver
First build requires click
Build In Build Out
Build Style
None
Direction
Delivery
Untitled Document (CS193E/
Code Split Design Title: Untitled Document
Welcome to CS193E
Object Oriented User Interface Progr
Announcements
Syllabus
Assignments
February 12, 2004 - Lecture
Today we covered MVC, NSNotifi
11.pdf).
We added an inspector to the TreeGenerator ap
100%

Finder


(97%)
Thu 7:45 AM
ECSCWFinal.pdf
phd_proposal.pdf
mailcrash.txt
Panther
Jaguar
UserData
SymbolTranslator
garageband_jam_pack.dmg
ilife04_dvd.dmg
garageband_jam_pack.002.dmgpart
ilife04_dvd.002.dmgpart
Certificates
Connection Doctor
WebSite
selection
UserData
Panther
Jaguar
Desktop
pmarcos
Mail_projs
Applications
4fe218ed62...2508ea9471
Account Settings Storage
Certificates
Comcast January 2004 Bill
Communiga...om Will Price
Connection Doctor
CS193E
CS193E.zip
DTD Buildin...locks.webloc
ECSCWFinal.pdf
garageband...002.dmgpart
garageband_jam_pack.dmg
Get Message Source Test
ilife04_dvd.002.dmgpart
ilife04_dvd.dmg
Assignments
Final Project Ideas
Lecture Notes
Lectures
projects
WebSite

Dreamweaver



TreeGenerator 3D

NSDocument Architecture

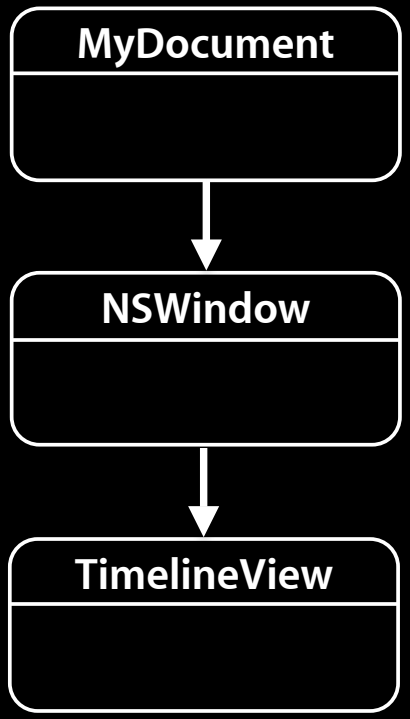
- Supports two approaches:
  - Simple, single window document
  - More complex multi-window document
- We've used the simple model so far
- Today we'll see how to extend it to support multiple windows for a single document

Planning Ahead

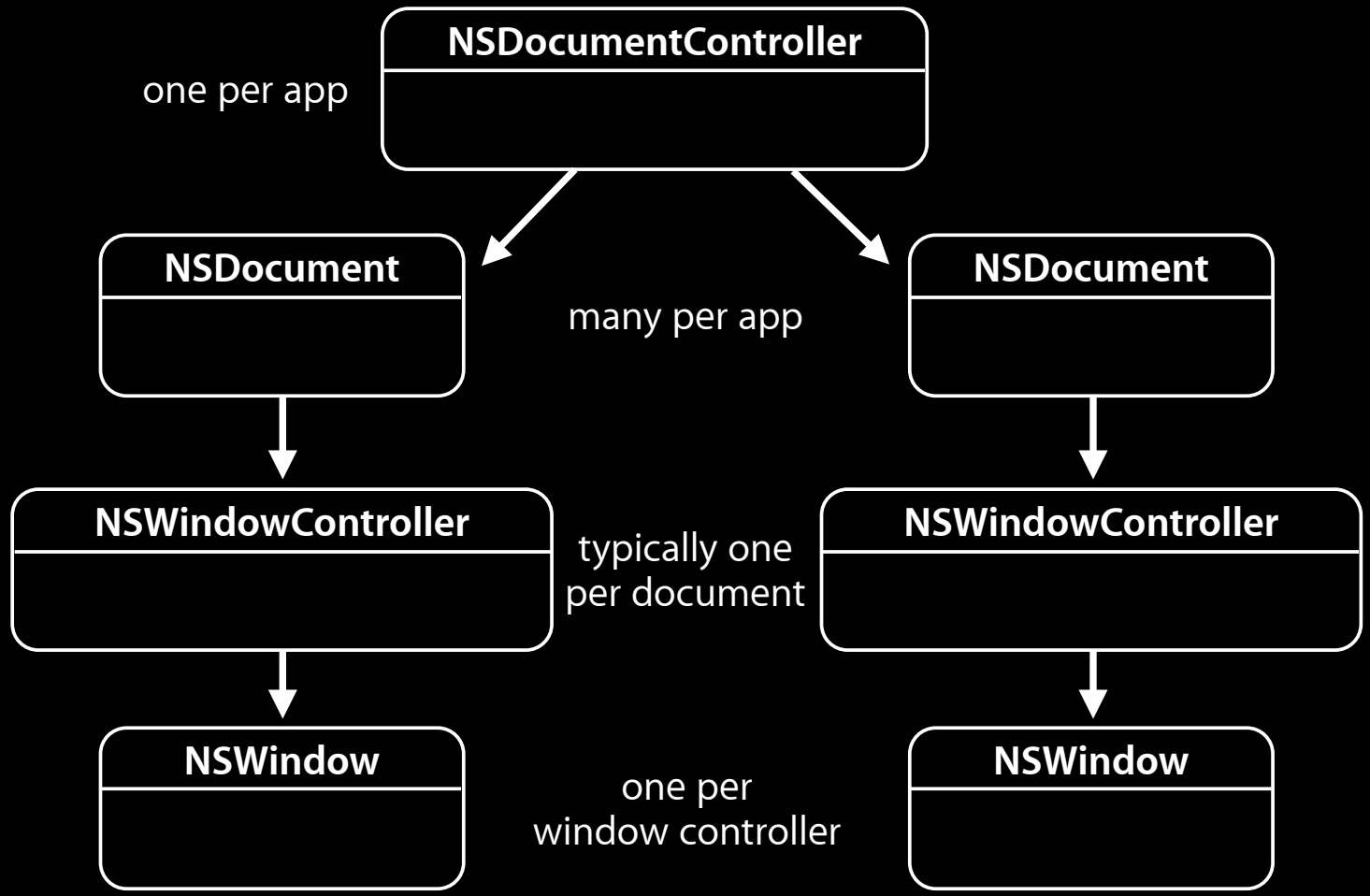
- Make sure your model is isolated
  - Write a command line tool
  - Write a separate test app
  - Can you do it without linking against AppKit framework?
- Make a diagram of your object relationships
- Keep your Model separated!

Timeline Architecture

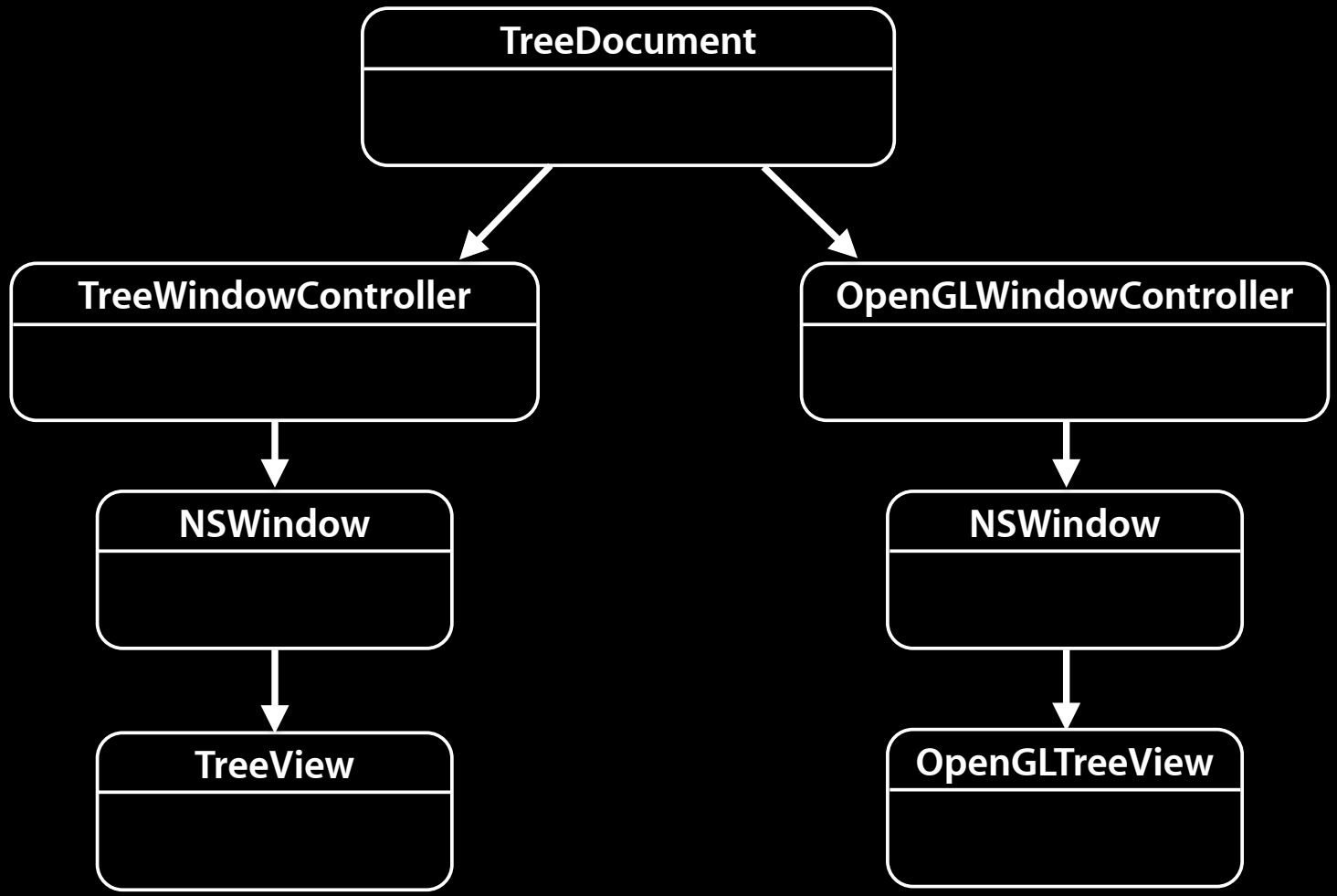
- Simple NSDocument case
- Single window
- NSDocument creates a default window controller and we don't really have to think about it



Document-based Architecture



Multiple Windows Per Document



Making The Switch

- Create NSWindowController subclass for existing window
- Update File's Owner class in nib file to be the window controller
- Relocate any outlets (and associated code) from your document to the window controller
- In the end, the window controller is now the "owner" of the nib

Making The Switch

- Relocate `windowNibName` from document to window controller.
- In your document, implement `makeWindowControllers` to set up default window controller
- Other window controllers can be added later using `addWindowController:`

Adding New Windows

- Create new NSWindowController subclass for new window
- Create nib file for new window, set File's Owner class to new window controller class
- Add code in your document subclass to create this new window controller
- Implement classes/views needed in your new window

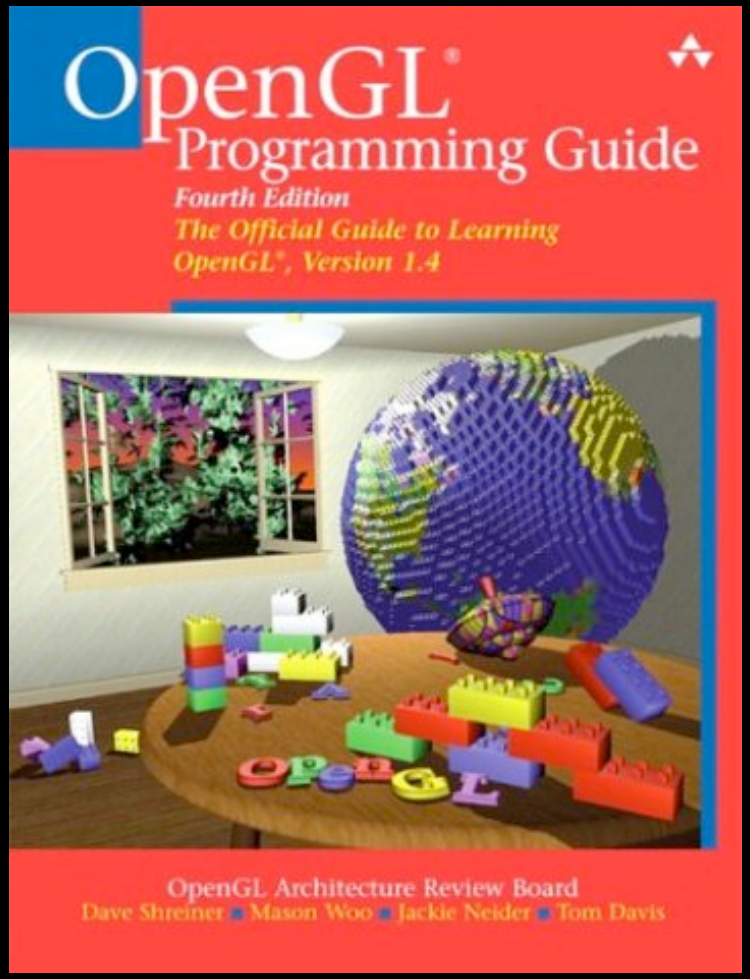
OpenGL

What is OpenGL?

- OpenGL is a software interface to graphics hardware
- Library of about 250 commands, platform and hardware independent
- The base library supports very minimal shape primitives: points, lines and polygons
- Higher level libraries add more rendering abilities and higher level operations (draw spheres, cylinders, teapots, etc)

OpenGL Reference

- "OpenGL Programming Guide" is the book to have
- Plenty of web sites with a ton of examples, easily migrated into Cocoa apps
- Several Cocoa examples on Apple's web site and with developer tools



OpenGL Extensions

- Other libraries add higher level constructs
  - GLU: GL Utility Library
    - Adds support for textures, additional shapes (spheres, cylinders, disks)
  - GLUT: GL Utility Toolkit
    - Adds UI level things like windows, menus, keyboard, mouse events, etc.

Naming Conventions

- Function names indicate what library function belongs to:
  - `glBegin(), glEnd(), glLight(), glPointSize()`
  - `gluBeginCurve(), glCylinder(), glSphere()`
  - `glutCreateWindow(), glutSetCursor()`
- Often indicate parameter types
  - `void glVertex2d(GLdouble x, GLdouble y);`
  - `void glVertex2i(GLint x, GLint y);`
  - `void glVertex4f(GLfloat x, GLfloat y, GLfloat z, GLfloat w);`

Naming Conventions

Parameter types

b	signed char (8-bit)
s	signed short (16-bit)
i	signed int (32-bit)
f	signed float (32-bit)
d	signed double (64-bit)
ub	unsigned char (8-bit)
us	unsigned short (16-bit)
ui	unsigned int (32-bit)

RedBook examples

- More documentation links available from:
  http://en.wikipedia.org/wiki/OpenGL
- Handy collection of examples from
  http://www.sgi.com/products/software/opengl/examples/redbook
- Built copies will be posted on the web site
- They work out of the box (mostly)
  - Had to change #include line and update makefile

Mac OS X & OpenGL

- OpenGL framework included on Mac OS X

```
#import <OpenGL/gl.h>
#import <GLUT/glut.h>
```

  - Don't forget to add frameworks in Xcode!!
- Quartz uses OpenGL for all the whizzy effects like the genie effect when minimizing, Exposé and cube rotations
- man pages for OpenGL functions as well as other OpenGL documentation

OpenGL Examples

Redbook Examples

OpenGL & Cocoa

2D Drawing

- Everything you've been doing up to this point
- NSView implements drawRect: where views do their drawing
- Cocoa handles the setup of the drawing context and flushing bits to the screen
- You draw using NSBezierPath, string drawing, NSImage compositing, etc

3D Drawing

- Conceptually not very different from 2D drawing
- NSOpenGLView does OpenGL context setup similar to how NSView handles setting up the 2D graphics context
- Instead of bezier paths, string drawing, etc. you make OpenGL client function calls to do drawing
- You have to explicitly flush the 3D pipeline to get bits on the screen, though

Cocoa OpenGL Classes

- NSOpenGLView - the Cocoa view that gets inserted into the view hierarchy, manages the OpenGL context
- NSOpenGLContext - ObjC representation of the OpenGL state variables
- NSOpenGLPixelFormat - specifies the buffer and various attributes for use by an NSOpenGLContext

Cocoa & OpenGL

- Cocoa provides an NSOpenGLView that manages the OpenGL context
- 3D equivalent of NSView
- Coordinate system differences:
  - Cocoa origin is in lower left corner of view
  - OpenGL origin is in center of view
  - OpenGL has a “z” coordinate, perpendicular to of the screen

Using NSOpenGLView

- Like using NSView, you create a custom subclass to implement your drawing
  - Either in Xcode or in IB
- In your nib you drag out an NSOpenGLView
- Change the class in the inspector to your custom subclass
- Just like NSViews...

Subclassing NSOpenGLView

- Required:
  - Implement `drawRect:` to do drawing
- Optional:
  - Implement `prepareOpenGL` to do initialization
  - Implement `reshape` to handle resizing

OpenGL & Cocoa Examples

OpenGLPolygon, OpenGLTeapot, & TreeGenerator

TreeGenerator Notes

- Includes a Camera class that encapsulates the view positioning
- OpenGLTreeView doesn't hold onto Tree directly, gets it from the document as needed

Quartz Composer Demo

Example Code

- Many examples ship with Xcode
- /Developer/Examples/OpenGL
  - Cocoa, GLUT, and Shader examples
- /Developer/Examples/QuartzComposer

Questions?