Programming

Software Engineering

Collaboration

Multimedia

Networking

Supervised Learning
Topics

- Multimedia Capture & Render w/ collaboratory
- Distributed groupware with GroupLab. Network...
- Put the two together
Demo 1:
Video Mirror

Shows:
- Getting Collabrary
- Working with C#
- Simple video capture
- Simple video rendering
Simple Video Mirror

Camera → Computer → Application Window
**Step 1:**
Download / install

[http://grouplab.cpsc.ucalgary.ca/collabrary](http://grouplab.cpsc.ucalgary.ca/collabrary)
Simple Audio Mirror

Microphone

Push to talk

Application

Speakers
Demo 3

Shared Inkpad

- Demonstrates
- Getting Started w/ GroupLab
- Networking
- Shared Dictionary Concepts
Shared Dictionary

"Dictionary" == "hashtable"

"map"

"associative array"

String keys → values of any type

→ look like paths in a filesystem

e.g. 

\[ d["/users/boylem/name"] = "Mike"; \]
\[ d["/users/boylem/mass"] = 87.3; \]
\[ d["/users/boylem/picture"] = Camera.Frame; \]
Publish/Subscribe
Producer/Consumer
Model/View
Abstraction/Presentation

**Bottom Line**

Put data in ➔ get notified of new data ➔ update GUI
Demo Architecture

Mouse Down
Mouse Move
Mouse Up

\[ \Rightarrow \text{Store in SD} \]

Subscription Notified \Rightarrow Paint
Data Model

/strokes : Vector of Guid
  stroke ids
/strokes/id/pts : vector of Point
  Could extend w/ colour, thickness, etc.
Get GroupLab Networking

http://grouplab.cpsc.ucalgary.ca/software/networking
Demo 4: Media Space
Media Space Architecture

Client

Shared Dictionary

Server

Client
Dictionary Structure

/id \rightarrow transient = id

/info = Map
- "id" = id
- "name" = "Michael Boyle"

/video = Buffer
MPEG-4 320x240 @ 1fps
Digression
Collabrary
Photo
Object
Important Methods

Resize

p.Resize(int width,
int height, bool smooth)

pretty straightforward

Copy: returns a new

photo that is a copy
of a portion of the
original

Object p = Copy(int x, int y,
int width, int height,
PhotoCopyStyle style)

Photo p2 = p(copy(0,0,0,
PhotoCopyStyle.Pixels) as
Photo;

(Zeroes for width, height mean
everything; in COM there
is no overloading, but there
are default param values)

Cast the result as Photo
because actual return type
is just object
C# as operator

- Casting operator
- Only works with reference types (classes) not value-types (structs)

```csharp
String s = "hi";
Point p = new Point(2.4);
object o = s;
string s2 = o as string;
string s3 = (string) o;

o = p;
Point p0 = o as Point;
Point p3 = (Point) o;
```

What's the difference?
```csharp
string s2 = o as string;
- FAST
- returns null if o not string
String s3 = (string)s;
- SLOW
- throws exception if o is not string
```
PhotoPaste(
  Photo p,
  int x, int y,
  float alpha);

pl: p1: (200,100)   p2:

pl.Paste(p2, 200, 100, 0.5f);

pl: p1: (200,100)
```
PhotoPasteEx(
  Photo p, int x, int y,
  Photo mask);

pl:

p2:

mask:

pl.PasteEx(p2, 0, 0, mask):

pl:

Note alpha blending

Note only blue channel pasted
```
photoDistort(
float clarity,
PhotoDistortStyle style)

clarity: 0.0 → 1.0

extreme distortion

no distortion
Photo. Load (string filename)
Photo. Store (string filename)

fairly straightforward

supported file types:

bmp
jpeg
png

24-bit colour

no per-pixel alpha
photo .Grab( object src): 

SRC:
- System . Reflection . Missing . Value
  - Grab whole screen (primary display only)
  - Automatically resizes to fit
- Control or Control . IWin32 . Window . Handle
  - Could also be Form
  - Screen grab given control, form, or window

string url
- Launches IE in the background to capture bitmap of webpage
  - If size initially 0x0, automatically resizes to about 800 px wide, then however tall needed to fit whole document
  - If size not 0x0 pix initially, layout doc in window of same size and grab just that part
- ActiveX not allowed
  - No SWF, PDF, x UMV, MOV, etc.
- Can take a while to load IE, browse/download page — use same photo obj
- DHTML sometimes inconsistent
Photo.LockMemory(
    out IntPtr p,
    out int len);

Photo.UnlockMemory(...);

Format: Standard Windows 24bpp DIB

3 bytes per pixel B,G,R
left-to-right, bottom-to-top
60 first byte is B of bottom-left pixel (0, H-1)
4th byte is B of (1, H-1)
last byte is R of (w-1, 0)

Rows aligned on 4 byte boundaries. 0, 1, 2 or 3 pad bytes may be inserted between rows e.g. 21 pixel wide image
row = 21x3 = 66 bytes data
each row has 2 additional pad bytes for alignment

rowH-1:
| byte 0: | B, G, R, ..., B, G, R, 0, 0. |
| col 0 | col 20 |

rowH-2:
| byte 68: | B, G, R, ..., B, G, R, 00, |
| col 0 |

Title: Sep 18 - 1:50 PM (26 of 26)