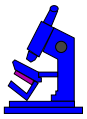
GUI Programming
CS3283

31st March 2003

Chapter 9

Code Samples

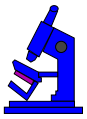


Java Graphics API

```
public void paintComponent(Graphics g) {
     super.paintComponent(g); //paint background
     //Paint a filled rectangle
     if (point != null) {
        g.drawRect(point.x, point.y, rw-1, rh-1);
        g.setColor(Color.yellow);
        g.fillRect(point.x+1,point.y+1,rw-2,rh-2);
     }
}
```


Graphics API

1. Basic/AWT - Abstract Graphics class
2. Java2D



Coordinate system

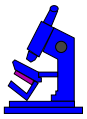
- ✔ Upper left of each component is (0,0)
- ✔ Behind the title bar of a window
- ✔ Container class has **`getInsets`** method
- ✔ Graphics objects contain methods for drawing

Graphics API

- ✔ Swing components have a method **`paintComponent`** which takes a graphics object as an argument

```
public void paintComponent( Graphics g )
```

- ✔ Overide this to draw your objects.
- ✔ Also may call the **`repaint()`** method

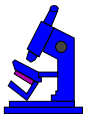


Graphics class methods

```
clearRect(int x, int y, int w, int h);
draw3DRect(int x, int y, int w, int h,          ...
drawImage(Image img, int x, int y, Color bg,    ...
drawLine(int x1, int y1, int x2, int y2);
drawOval(int x, int y,int width, int height);
drawPolygon(int x[], int y[], int               ...
drawRect(int x, int y,int width, int height);
drawString(String str, int x, int y);
```


Graphics class methods

```
fill3DRect(int x, int y, int w, int h,  ...
fillArc(int x, int y, int w, int h,     ...
fillOval(int x, int y, int w, int h);
fillPolygon(int x[], int y[], int       ...
fillRect(int x, int y, int w, int h);
```

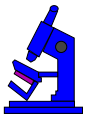


Graphics class methods

```
Color getColor();
Font getFont();
FontMetrics getFontMetrics();
setColor(Color c);
setFont(Font font);
```


Graphics API

- ✔ Use JPanel instead of JComponent
- ✔ UI delegate (for look-and-feel painting) is called in JPanel
- ✔ UI delegate not called in JComponent



Text in Graphics API

- Note - you paint text using **`drawString()`**
- **`getFontMetrics()`** to get a **`FontMetrics`** object

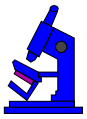
```
getHeight()
getAscent()
getDescent()
charWidth()
```

- and so on...

Images in Graphics API

- Use **`drawImage()`**
- Get an image using **`getImage(URL)`** (for applet)
- Get an image using

```
Toolkit.getDefaultToolkit().getImage()
```

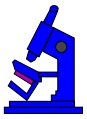


Java2D

- ✔ Graphics2D class - subclass of Graphics
- ✔ Better control over geometry, coordinate transformations, color management, and text layout.
- ✔ This is the fundamental class for rendering 2-dimensional shapes, text and images on the Java(tm) platform.

Java3D

- ✔ Found in javax.media.j3ds
- ✔ A GraphicsContext3D object is used for immediate mode rendering into a 3D canvas.
- ✔ Methods to set 3D graphics state and draw 3D geometric primitives.

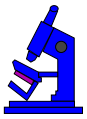


3D Graphics context

- Background object: null
- Fog object: null
- ModelClip object: null
- Appearance object: null
- List of Light objects: empty

Sound context

- AuralAttributes object: null
- List of Sound objects: empty
- stereo mode: STEREO_BOTH



OpenGL

- ✔ API for interactive 3D graphics rendering and 2D imaging.
- ✔ Polygon specification,
- ✔ Lighting control, Transformation specification, and
- ✔ Framebuffer operations like blending, texture mapping and depth-buffering.

OpenGL pipeline

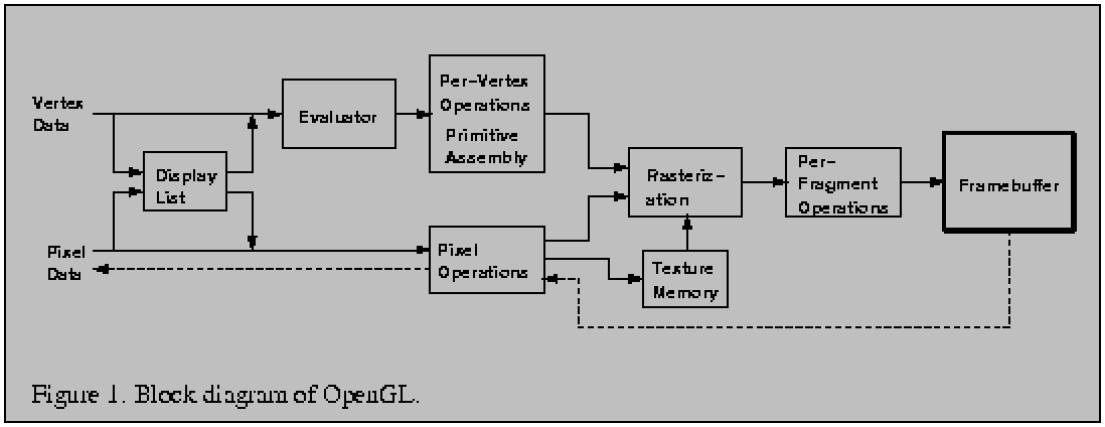
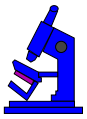


Figure 1. Block diagram of OpenGL.



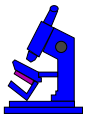
OpenGL in Tk

tkogl: http://hct.ece.ubc.ca/research/tkogl/releaseApr01/

and others...

Display list in Tk

```
proc init_hexahedron {} {
    global poly
    set a [expr 1.0/sqrt(3.0)]
    set poly(hexahedron,vtx) [list \
        [list $a $a $a] [list $a $a -$a] \
        [list -$a $a -$a] [list -$a $a $a] \
        [list -$a -$a $a] [list -$a -$a -$a] \
        [list $a -$a -$a] [list $a -$a $a] ]
    set poly(hexahedron,face) {
       {0 1 2 3} {3 2 5 4} {4 5 6 7}
       {7 6 1 0} {1 6 5 2} {0 3 4 7}
    }
}
```



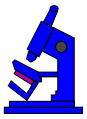
Mainline, 3D widget

```
pack [OGLwin .gl -aspect 1 -stencil 8]
set dlist [.gl newlist]
polybuild .gl hexahedron $dlist
.gl eval \
    -matrixmode projection \
    -perspective 20 1 0.5 20\
    -matrixmode modelview \
    -material front ambient 0.3 0.3 0.3 \
    -material front diffuse 0.5 0.5 0.5 ...
.gl main -call $dlist
```


VRML

- ✔ Virtual Reality Modelling Language
- ✔ 3D views
- ✔ Animated (floops, diag6)

http://www.lighthouse3d.com/vrml/tutorial/



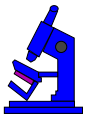

VRML

```
#VRML V2.0 utf8
Background { skyColor .4 .66 1 }
NavigationInfo { type [ "EXAMINE", "ANY" ]
                 speed 400 }
Viewpoint { position 0 400 0
            orientation 0 1 0 4
            description "Camera 1" }
PROTO SHP [] { Transform { translation 0 0 -25
     children [ Shape {
                 appearance Appearance {
                   material Material {
                     diffuseColor 0.2 0.2 1.0 } }
                 geometry Box { size 40 40 10 } }
     ] } }
DEF n7 Transform {scale 1 1 1 children [SHP {}]}
```



Chapter 10

MFC



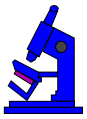
MFC

- ✔ Microsoft Foundation Classes - classes needed to produce GUI Windows programs.
- ✔ Development cycle - RAD, then editing.

MFC menus

A resource file for a simple File/Quit menu:

```
#define MYAPP_EXIT 3210
MyApp MENU
    POPUP "File"
    {
      MENUITEM "Exit",MYAPP_EXIT
    }
}
```



Menus

In the **`Create`** call, you can do something like this:

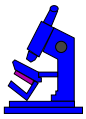
```
Create( NULL, “Example”, ..., CRect(...), NULL, “MyApp” );
```

The **`MYAPP_EXIT`** message may be bound using the **`DECLARE_MESSAGE_MAP()`** macro, and with the following declaration:

```
ON_COMMAND( MYAPP_EXIT,OnExit )
```


Message handler

```
afx_msg void CMenusWin::OnExit()
{
    SendMessage( WM_CLOSE );
}
```



MFC Program

MFC program

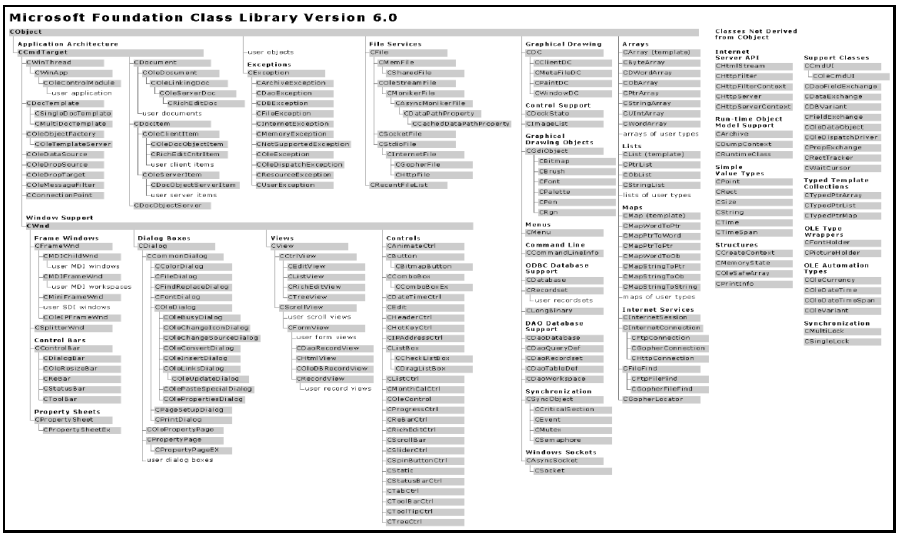
CODE LISTING **FirstApp.cpp**

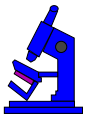
```
#include <afxwin.h>

class CFirstWindow : public CFrameWnd {
public:
    CFirstWindow();
    ~CFirstWindow();
private:
    CStatic *m_pGreeting;
};

CFirstWindow::CFirstWindow()
{
    Create( NULL,
            "First Application",
            WS_OVERLAPPEDWINDOW,
            CRect( 100, 100, 400, 220 ) );
    m_pGreeting = new CStatic;
    m_pGreeting->Create(
       "Hello World!", // text
       WS_CHILD | WS_VISIBLE | WS_BORDER
       | SS_CENTER,
       CRect( 80, 30, 200,50 ),
       this );
}

CFirstWindow::~CFirstWindow()
{
    delete m_pGreeting;
}

class CFirstApp : public CWinApp {
public:
    BOOL InitInstance()
    {
       m_pMainWnd = new CFirstWindow();
       m_pMainWnd->ShowWindow( m_nCmdShow );
       m_pMainWnd->UpdateWindow();
       return TRUE;
    }

} FirstApp;
```

Hungarian notation

Prefix	Meaning
`C`	Class declaration
`m_`	Class member variable
`p`	Pointer
`n` or `i`	Integer
`On`	Event or message handler

MFC class hierarchy

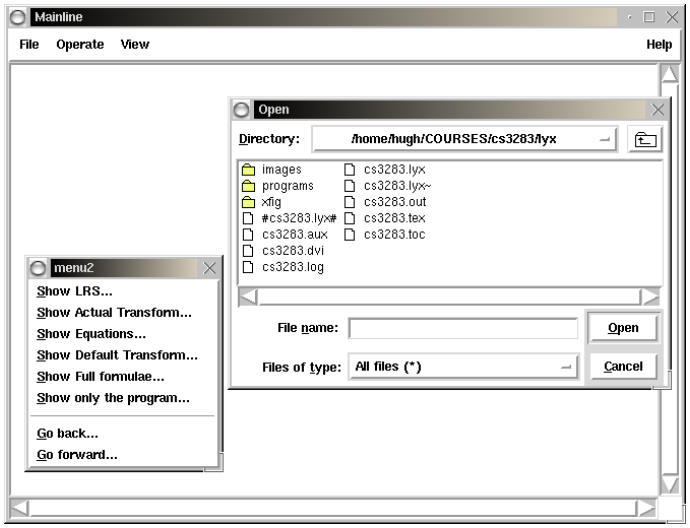


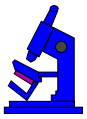



Chapter 11

Case study


GUI





Perl/Tk code

Perl has a Tk module:

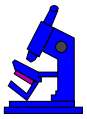
CODE LISTING **mainline.pl**

```
use Tk;
my $currentslice = 0;
my $currentpp    = 0;
my $disptype     = 2;
my $main = new MainWindow;

<<SetupMenu>>
<<SetupFileMenu>>
<<SetupEditMenu>>
<<SetupViewMenu>>

$main->configure(-menu =>$menubar);

<<SetupScrolledMainArea>>

MainLoop;

<<FileOpenDialogBox>>
```


Menu bar

CODE LISTING **SetupMenu.pl**

```
$menubar = $main->Menu;
$filemenu = $menubar->cascade(-label=>"File" );
$editmenu = $menubar->cascade(-label=>"Operate" );
$viewmenu = $menubar->cascade(-label=>"View" );
$helpmenu = $menubar->cascade(-label=>"Help");
$helpmenu->command(-command => \&about_choice,
                   -label => "About TkMenu...",
                   -underline => 0);
```




Menu items

CODE LISTING **SetUpFileMenu.pl**

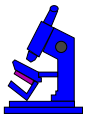
```
$filemenu->command(-command => sub { fileDialog( $main, 'open' );
                                     printf "Opening $thisfile\n";
                                     readfile($thisfile);
                                     writefile($thisfile . ".ppx");},
                   -label => "Open...",
                   -underline => 0);
$filemenu->separator;
$filemenu->command(-label => "Exit",
                   -command => \&exit_choice,
                   -underline => 1);
```



Edit menu

CODE LISTING **SetUpEditMenu.pl**

```
$editmenu->command(-command => sub {Tp($currentslice,1,1);},
                   -label => "Crank with widening...",
                   -underline => 0);
$editmenu->command(-command => sub {Tp($currentslice,1,10);},
                   -label => "Crank with widening (10X)...",
                   -underline => 0);
$editmenu->command(-command => sub {Tp($currentslice,0,1);},
                   -label => "Crank...",
                   -underline => 0);
$editmenu->command(-command => sub {Tp($currentslice,0,10);},
                   -label => "Crank (10X)...",
                   -underline => 0);
$editmenu->command(-command => sub {Cousot($currentslice);},
                   -label => "Cousot...",
                   -underline => 0);
$editmenu->separator;
$editmenu->command(-command => sub {widening($currentslice);},
                   -label => "Widen...",
                   -underline => 0);
```



View menu

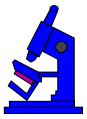
CODE LISTING **SetUpViewMenu.pl**

```
$viewmenu->command(-command => sub {$disptype=0;display($currentslice,0);},
                   -label => "Show LRS...",
                   -underline => 0);
$viewmenu->command(-command => sub {$disptype=1;display($currentslice,1);},
                   -label => "Show Actual Transform...",
                   -underline => 0);
$viewmenu->command(-command => sub {$disptype=2;display($currentslice,2);},
                   -label => "Show Equations...",
                   -underline => 0);
$viewmenu->command(-command => sub {$disptype=3;display($currentslice,3);},
                   -label => "Show Default Transform...",
                   -underline => 0);
$viewmenu->command(-command => sub {$disptype=4;display($currentslice,4);},
                   -label => "Show Full formulae...",
                   -underline => 0);
$viewmenu->command(-command => sub {$disptype=5;display($currentslice,5);},
                   -label => "Show only the program...",
                   -underline => 0);
$viewmenu->separator;
$viewmenu->command(-command => sub { if ($currentslice>0) {
                                        $currentslice=$currentslice-1;
                                        if ($currentpp==0) {
                                           $currentpp=$codesize;
                                        }
                                        $currentpp=$currentpp-1;
                                        display($currentslice,$disptype);
                                     }},
                   -label => "Go back...",
                   -underline => 0);
$viewmenu->command(-command => sub { if ($currentslice+1<$maxslice){
                                        $currentslice=$currentslice+1;
                                        $currentpp=$currentpp+1;
                                        if ($currentpp==$codesize) {
                                           $currentpp=0;
                                        }
                                        display($currentslice,$disptype);
                                     }},
                   -label => "Go forward...",
                   -underline => 0);
```


Dialog box

CODE LISTING **FileOpenDialogBox.pl**

```
sub exit_choice {
   exit;
}

sub fileDialog {
   my $w = shift;
   my $operation = shift;
   my $types; my $file;
   @types = (["Code files", '.pp'],
             ["Work files", '.ppx'],
             ["All files", '*'] );
   $file = $w->getOpenFile(-filetypes => \@types);
   if (defined $file and $file ne '') {
      $thisfile = $file;
   }
}
```



Code

http://www.comp.nus.edu.sg/~cs3283/ftp/original.pl

It may be run by typing “**perl original.pl**”.