

Drawing w/ Pen

Pens are used for lines (not filled drawing)

Pen property float Width get/set

```
Pen (Color c); // default width of 1
```

```
Pen (Color c, float w);
```

DrawLines

```
DrawLine (Pen p, int x1, int y1, int x2, int y2);
```

```
DrawLine (Pen p, Point p1, Point p2);
```

also float x, y and PointF's values

includes the second point positions

Assume Graphics g

```
g.DrawLine (pen, 3, 3, 3, 3); // draws nothing
```

```
DrawLines (Pen, Point[] pt);
```

draws a poly line, connect the points

also array of PointF

Math class

constants: Math.PI and Math.E

transcendental functions (sin, cos, ...)

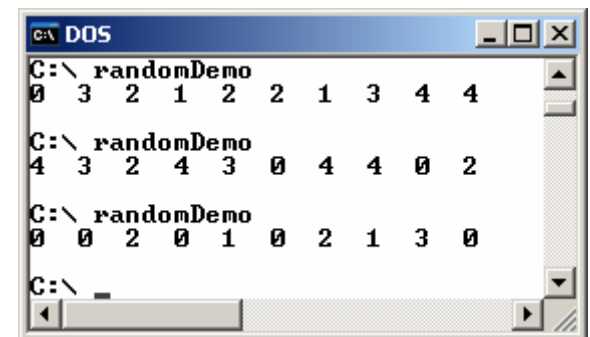
double Math.Sin(double radians);

methods: Abs, Floor, Round, Ceiling, Exp, Pow ...

Random class

```
using System;
class RandomDemo {
    [STAThread] // single threaded application
    static void Main(string[] args) {
        Random r = new Random (); // use time dependent seed
        for(int i = 0; i < 10; i++)
            Console.Write ("{0} ", r.Next (0, 5));
        Console.WriteLine ();
    }
}
```

generates 10 integers values {0, 1, 2, 3, 4}



```
C:\ DOS
C:\ randomDemo
0 3 2 1 2 2 1 3 4 4
C:\ randomDemo
4 3 2 4 3 0 4 4 0 2
C:\ randomDemo
0 0 2 0 1 0 2 1 3 0
C:\ _
```

Draw rectangles

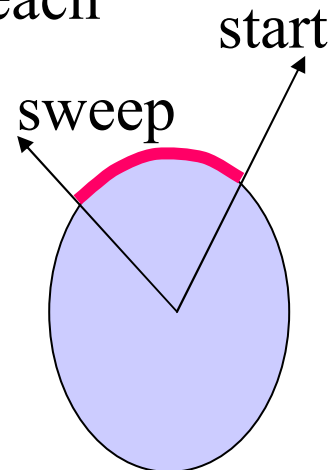
```
DrawRectangle (Pen p, Rectangle r);  
DrawRectangle (Pen p, int x, int y, int w, int h);  
    // also w/ floats
```

Draw Polygons

```
DrawPolygon (Pen p, Point[] pt);    // also w/ array of PointF  
like DrawLines with array except a closing line is draw between the  
    first and last points in the array
```

Other draw commands -- there are overloaded methods for each

```
DrawEllipse (Pen p, Rectangle r);  
DrawArc (Pen p, Rectangle r,  
    float start, float sweep);  
DrawPie (Pen p, Rectangle r,  
    float start, float sweep);
```



Arc is part of an ellipse. Pie is an Arc with edges to center.

Drawing with Brush

There are several overloaded methods for each filled shape

FillRectangle (Brush b, Rectangle r);

FillRectangles (Brush b, Rectangle[] r);

FillEllipse(...);

FillPie (...)

FillPolygon (Brush b, Point[] pt, FillMode fm);

FillMode has effect if lines defining polygon body overlap!

FillMode enumerations:

Alternate fill, unfilled, fill, unfilled alternate (default)

Winding most interiors are filled (usually fills all)

Graphics class

The Graphics class provides several transformations and there is a matrix class! Useful for coordinate transformations.

World \longrightarrow Page \longrightarrow Device

```
TranslateTransform (float dx, dy);
```

```
ScaleTransform (float sx, sy);
```

```
RotateTransform (float radian);
```

all transforms can specify a MatrixOrder (for matrix composition)

```
RotateTransform (float radian, MatrixOrder mo);
```

MatrixOrder enumeration Prepend, Append

Page coordinates are transformed to Device coordinates with

PageTransformations

using Graphics properties PageScale and PageUnit.

Graphics properties, GraphicState class, can be saved and restored

```
GraphicsState gs = g.Save(); Or g.Restore = gs;
```

Matrix class namespace System.Drawing.Drawing2D

Matrix is a 3 by 3 matrix useful for 2D transformations.

```
Matrix ();           an identity matrix
Matrix ( float sx, float ry,
         float rx, float sy,
         float dx, float dy);
```

$$\begin{vmatrix} s_x & r_y & 0 \\ r_x & s_y & 0 \\ d_x & d_y & 1 \end{vmatrix}$$

Graphics class has a Transform property that is a Matrix.

Matrices can be composed

To multiply the existing (transform) matrix with Matrix mx

```
MultiplyTransform(Matrix mx); // default is prepend MatrixOrder
```

```
MultiplyTranform(Matrix mx, MatrixOrder mo);
```

Java has matrix objects in the javax.vecmath package
part of the Java 3D API download

Mouse events

mouse	input device
mouse cursor	on screen pointer
hotspot	actual active (selection) location of mouse pointer

SystemInformation properties

int	DoubleClickTime	get	msec
Size	DoubleClickSize	get	area in pixel
bool	MousePresent	get	
int	MouseButtons	get	1 to 5
bool	MouseButtonsSwapped	get	
bool	MouseWheelPresent	get	
int	MouseWheelScrollLines	get	lines / scroll turn

Mouse events (Control events)

Event	Method	Delegate	Argument
MouseDown	OnMouseDown	MouseEventHandler	MouseEventArgs
MouseUp	OnMouseUp	MouseEventHandler	MouseEventArgs
MouseMove	OnMouseMove	MouseEventHandler	MouseEventArgs
MouseWheel	OnMouseWheel	MouseEventHandler	MouseEventArgs

MouseEventArgs properties

int	X	get	location in client area
int	Y	get	
MouseButton	Button	get	mouse buttons used
int	Clicks	get	2 for double clicks
int	Delta	get	mouse wheel movement 1 click is $\approx \pm 120$

MouseButton enumeration (bit flags, can be combined "and")

None, Left, Right, Middle, XButton1, XButton2
(IntelliMouse Explorer has 5 buttons)

Event	Method	Delegate	Argument
Click	OnClick	EventHandler	EventArgs
DoubleClick	OnDoubleClick	EventHandler	EventArgs
MouseEnter	OnMouseEnter	EventHandler	EventArgs
MouseLeave	OnMouseLeave	EventHandler	EventArgs
MouseHover	OnMouseHover	EventHandler	EventArgs

EventArgs does not have any mouse specific information.

Control class does have properties

Point	MousePosition	get	location in screen coordinates
MouseButtons	MouseButtons	get	buttons currently pressed
Keys	ModifierKeys	get	status of Shift, Cntrl, Alt keys

```
if (MouseButtons == MouseButtons.Left &&
aControl.ModifierKeys == Keys.Shift)
```

Use PointToClient to convert screen coordinates to client area coordinates.

```
aControl.PointToClient(aControl.MousePosition);
```

Three events: `MouseDown`, `MouseMove`, and `MouseUp` are used to track mouse movement events.

`MouseDown`, `OnMouseDown`

set up the state (or store) to record mouse move events

`MouseMove`, `OnMouseMove`

store information on movement (usually X, Y or Points)

if drawing is done need to

```
Graphics g = CreateGraphics ();  
// do drawing  
g.Dispose ();
```

`MouseUp`, `OnMouseUp`

stop or unset the state for recording mouse move events

use `Invalidate ()` to have `OnPaint (...)` called.

Cursor and Cursors class

Static Cursors properties can be used to set the control's cursor shape

Cursor properties

Cursor	Current	get/set	set the cursor
Point	Position	get/set	position the cursor !advised
Rectangle	Clip	get/set	

Cursor methods `void Show ();` `void Hide ();`

To set the cursor its better to use the Control's Cursor property

```
aControl.Cursor = Cursors.Hand;
```

ContextMenu Show (menu property set with Designer)

(pop ups) are initiated with mouse events and displayed with
`Show(Control c, Point p);` Control is "associated" with menu

```
private void contextMenu_Popup(object s, EventArgs e) {  

    if (MouseButton == MouseButton.Right)  

        contextMenu.Show(this, MousePosition);    }
```

Simple font usage

Control Properties related to fonts

Font	Font	get / set	font
Font	DefaultFont	get	font
int	FontHeight	get / set	height

FontFamily class is indentified by string ie. "Times New Roman"

Font is a combination of a font family (object or string identifier), attributes (bold, italic) and a point size

```
Font (Font font, FontStyle fs);  
Font (string family, float points);  
Font (string family, float points, FontStyle fs);
```

FontStyle enumeration {Regular, Bold, Italic, Underline, Strinkout }

```
Font aFont = new Font ("Times New Roman", 18);  
Font bFont = new Font (aFont,  
    FontStyle.Bold | FontStyle.Italic);
```

Timer class

Send an event after counting down for an interval

Time methods

```
Timer timer = new Timer ();  
void Start ();  
void Stop ();
```

Timer Properties

int	Interval	get / set	tick time in msec default 100
bool	Enabled	get / set	true when timer is running

Time event

Tick	OnTick	EventHandler	EventArgs
------	--------	--------------	-----------

hint: check out RandomRectangle.cs wrt project 2

component

CommonDialog (abstract)

colorDialog

FileDialog (abstract)

OpenFileDialog

SaveFileDialog

FontDialog

PageSetupDialog

PrintDialog

Common Dialog Boxes

Add to controls using Designer

Use properties to get values

Open dialogs in event handler method

```
DialogResult dr =  
    aDialog.ShowDialog();  
if (dr == DialogResult.OK)  
    // get property of  
    // aDialog
```

FontDialog Properties

Font	Font	get / set		
Color	Color	get / set		
bool	ShowEffects	get / set	true	underline, strikethrough ?
bool	ShowColor	get / set	false	combo box for color
bool	ShowApply	get / set	false	apply button ?
bool	ShowHelp	get / set	false	help button ?
bool	FixedPitchOnly	get / set	false	

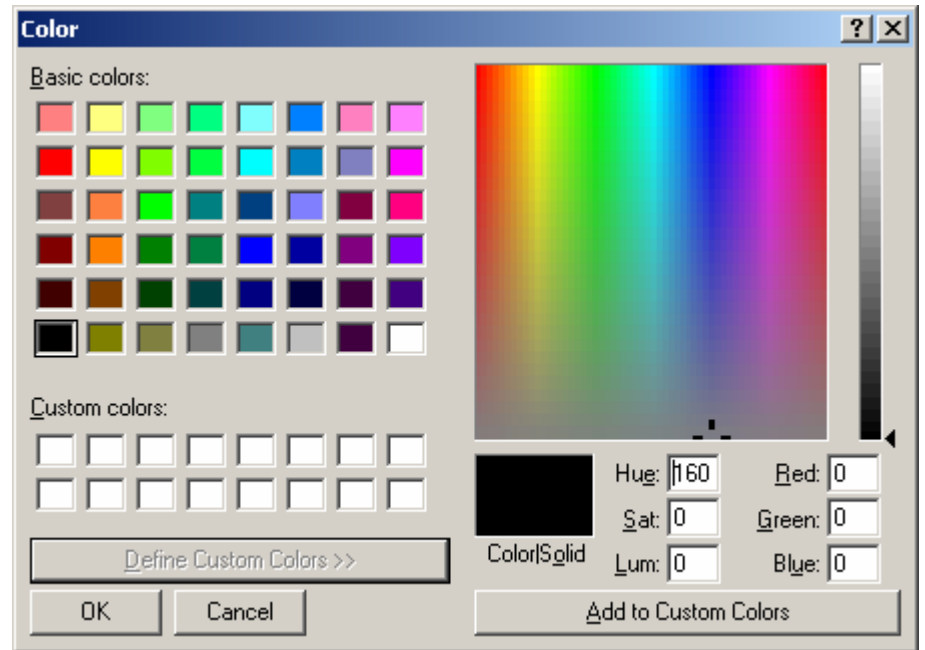
FontDialog events

Apply	OnApply	EventHandler	EventArgs
HelpRequest	OnHelpRequest	EventHandler	EventArgs

Apply button semantics forces application to "apply" effects of dialog before the ShowDialog method returns.

ColorDialog properties

Color	Color	get / set		
bool	FullOpen	get / set	false	open wide custom color
bool	AllowFullOpen	get / set	true	allow custom color open
bool	SolidColorOnly	get / set	false	
bool	AnyColor	get / set	false	
bool	ShowHelp	get / set	false	
int[]	CustomColors	get / set		



FileDialog is an abstract class with properties shared by Open and Save

FileDialog properties

string	FileName	get / set	""	filename set by dialog
string[]	FileNames	get / set	""	array of filenames (save ?)
string	Filter	get / set		
bool	AddExtension	get / set	true	add extension from Filter ?
string	InitialDirectory	get / set	""	
bool	RestoreDirectory	get / set		restore current directory ?
bool	CheckPathExists	get / set	true	check before closing ?
bool	CheckFileExists	get / set	true	check before closing ?

```
dialogFilter = "Source files | *.cpp;*.cc; *.c|" + "Header files | *.h ";
```

filter displays string up to "|" in filter combo box

string to the right of "|" is the actual extensions separated with ";"

terminated with "|"

Designer don't put filter string in "quotes", designer will add "quotes"

OpenFileDialog properties

bool	Multiselect	get / set
bool	ShowReadOnly	get / set
bool	ReadOnlyChecked	get / set

for user to create new file with OpenFileDialog set CheckFileExists to false.

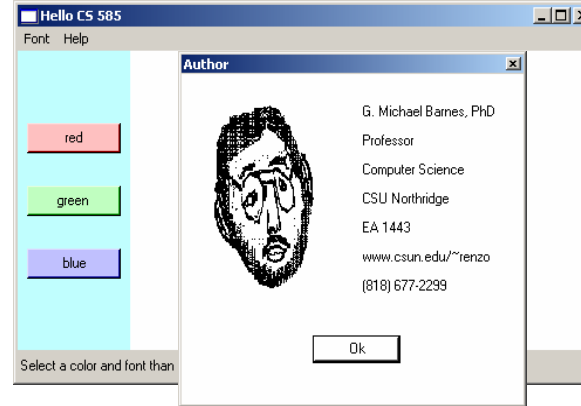
SaveFileDialog properties

bool	CreatePrompt	get / set	false	set true to prompt
bool	OverwritePrompt	get / set	true	ask for confirmation

SaveFileDialog will append the first extension listed in the Filter property filter line that user selects in "Save As Type" combo box.

Custom Dialogs

<< see Hello585.cs >>
 customDialog.cs >>



Visual Studio .NET

Project | add Windows Form eg: Author.cs

Use Toolbox to add controls to new form // PictureBox, Button, Labels

In owner (application) form class source file (Hello585.cs)

declare object of new form's Class source file

```
private Author author;
```

in "initator's" delegate // usually a menu item's delegate

```
private void authorMenuItem_Click(object sender,  
EventArgs e) {  
    author.Show(); } // for modal use ShowDialog()
```

Based on DialogResult in application's source get dialog's properties if desired or write accessor methods in dialog class to return properties of desired controls.

```
namespace Hello585 {           // project's namespace

    public class Author : Form {
        // declare controls for dialog
        private Button okButton;

    public Author() {
        // create controls for dialog
        okButton = new Button();
        // set properties of dialog's controls
        okButton.Text = "Ok";
        okButton.Location = new Point(xPos, yPos);
        okButton.Click += new
            EventHandler(this.okButton_Click);
        //set clientSize, location, properties of Author
    }

    private void okButton_Click(object sObj, EventArgs e) {
        Hide(); }
    ... } }
```