Quellcodehilfen – Grafik mit GDI+ und mehr

Georg-Simon-Ohm Berufskolleg

Punkt-Objekte
Konstruktor z.B.: Point(int x, int y)
Eigenschaften: X, Y (jeweils int)

Size-Objekte
Konstruktor z.B.: Size(int width, int height)
Eigenschaften: Width, Height (jeweils int)

Rechteck-Objekte
Konstruktor z.B.: Rectangle(int x, int y, int width, int height)
Eigenschaften: X, Y, Width, Height (jeweils int)
bool Contains(Point pt); // Treffer prüfen in Rechteck
bool Contains(Rectangle rect);
bool Contains(int x, int y);
Klasse: RectangleF wie Rectangle nur mit *float* Größen
Rectangle rect = Rectangle.Round(rectF); // RectangleF à Rectangle umwandeln

Vektorarithmetik
Point p1 = p2 + (Size)p3; // Vektoraddition von Point-Werten
Point p1 = p2 - (Size)p3; // Vektorsubtraktion von Point-Werten

Fenstermaße - Form, UserControl, Control usw.
this.ClientSize.Width, this.ClientSize.Height (jeweils int)

Farben
[System.Drawing.]Color à Color.White, Colol.Red ...
[System.Drawing.]SystemColors à SystemColors.WindowText ...
Color.FromArgb(…), Color.KnownColor(…), Color.FromName(…)
ColorTranslator.FromHtml(...) mit string ”Red” oder “#FF00C7”

Zeichenmittel
```
Pen penThin = new Pen(Color.DarkRed);
Pen penThick = new Pen(Color.DarkRed, 2);
Brush brush = new SolidBrush(Color.Orange);
```

Zeichenbefehle der Klasse Graphics
```
void DrawLine(Pen pen, int x1, int y1, int x2, int y2);
void DrawLine(Pen pen, Point pt1, Point pt2);
void DrawRectangle(Pen pen, Rectangle rect);
void DrawRectangle(Pen pen, int x, int y, int width, int height);
void FillRectangle(Brush brush, Rectangle rect);
void FillRectangle(Brush brush, int x, int y, int width, int height);
```

Tortenstück zeichnen
startWinkel: von 3 Uhr (x-Achsenrichtung) aus, oeffnungsWinkel: Größe des Tortenstücks
Rechteck/Quadrat gibt Ellipse/Kreis an in der das/die Tortenstück/e liegen sollen
```
g.FillPie(brush, new Rectangle(10,10, 100, 100), startWinkel, oeffnungsWinkel);
g.DrawPie(pen, new Rectangle(10,10, 100, 100), 15, 90);
```

Textmessung und Textausgabe
```
Brush textBrush = new SolidBrush(Color.Black);
g.DrawString("Text rechts unter Position", this.Font, textBrush, 5, 5);
string text = "Ein anderer Text";
SizeF textSize = g.MeasureString(text, this.Font);
float x = this.ClientSize.Width - textSize.Width - 5;
float y = 5;
g.DrawString(text, this.Font, textBrush, x, y);
```

List<Item>
```
Clear()             // Alle Elemente der Liste löschen
Add(Item item)      // Ein Element zufügen
Remove(Item item)   // Ein Element löschen
RemoveAt(int index) // Ein Element per Listenindex löschen
Item[] copy = myList.ToArray() // Array als Kopie erstellen
```

Dateien
```
StreamReader sr = new StreamReader(fileName, Encoding.Default);
```
à sr.EndOfStream, sr.ReadLine(), sr.Close()
OpenFileDialog à ofd.ShowDialog(), ofd.FileName, ofd.Filter = “Alle Dateien (*.*)|*.*“

Methoden von Math
Math.Min(a,b), Math.Max(a,b)
Wurzelfunktion: Math.Sqrt(x), Potenzfunktion x hoch y: Math.Pow(x, y)

Zufallszahlen
```
Random random = new Random();  // Zufallsgenerator erzeugen (nur ein mal)
int zufall = random.Next(a,b); // Zufallszahl z.B. zwischen a...(b-1) erzeugen
```