図形を描画

長方形を描画する

大抵の図形は、これまで説明した線を描画するメソッドで表現できますが
より一般的な長方形や円弧は、一つのメソッドで描画できるべきでしょう
今回は、一般的によく用いられる図形を描画するメソッドを中心に紹介します

長方形を描画するには Graphics.DrawRectangle() を使います
このメソッドでは、長方形の左上部分を基準とした座標と、幅と高さを指定します

```
public void DrawRectangle(Pen pen , Rectangle rect);

public void DrawRectangle(
 Pen pen,
 int x , int y , int width , int height
);
public void DrawRectangle(
 Pen pen ,
 float x , float y , float width , float height
);
```

pen には、図形を描画するためのペンをあらわす Pen オブジェクトを指定します
rect は、描画する長方形の座標と大きさを表す Rectangle 構造体を指定します
x と y には、長方形の左上角となる座標を
width と height には、長方形の座標から相対的な幅と高さを指定します

長方形の幅と高さとは、座標ではないことに注意してください
これらは、長方形の左上角から相対的に数えたピクセル数になります

長方形は System.Drawing.Rectangle 構造体で表すこともできます
この構造体は、単純に Point 構造体と Size 構造体を一つにしたものと考えられます
実際に、このコンストラクタではそれらの構造体を利用することができます

public Rectangle(Point location , Size size);
public Rectangle(int x , int y , int width , int height);

このコンストラクタによって、長方形の座標と幅をあらわす Rectangle のインスタンスを生成し
あとは、この情報を元に長方形を描画するというような処理を行えます
この構造体は、描画以外でも長方形をあらわす必要がある時に用いることがあります

長方形を示す情報にアクセスするには Rectangle.X と Rectangle.Y
Rectangle.Width、Rectangle.Height プロパティを用います

public int X {get; set;}
public int Y {get; set;}
public int Width {get; set;}
public int Height {get; set;}

X は X 座標、Y は Y 座標、Width は幅、Height は高さを表しています
ただし、読み込み専用として Rectangle.Left、Rectangle.Top
Rectangle.Right、Rectangle.Bottom プロパティもあります

public int Left {get;}
public int Top {get;}
public int Right {get;}
public int Bottom {get;}

Left は矩形の左上 X 座標、Top は左上 Y 座標
Right は右下の X 座標、Bottom は右下の Y 座標を表しています
Width や Height プロパティは、左上角からの幅と高さという情報でしたが
Right と Bottom プロパティは座標で表されているということに注意してください

さらに、Rectangle.Location と Rectangle.Size という
Point と Size 構造体として情報にアクセスするためのプロパティも存在します

public Point Location {get; set;}
public Size Size {get; set;}

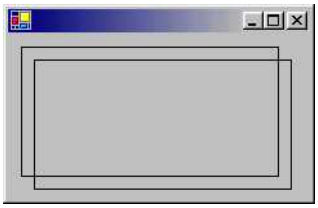
Location プロパティは、矩形の左上角の座標を表す Point 構造体
Size プロパティは、幅と高さを表す Size 構造体を示しています
これらのプロパティを用いて、その場に適した形で矩形の情報にアクセスできます

```
using System.Windows.Forms;
using System.Drawing;

class WinMain : Form {
 public static void Main(string[] args) {
  Application.Run(new WinMain());
 }
 override protected void OnPaint(PaintEventArgs e) {
  Graphics g = e.Graphics;
  Pen myPen = new Pen(Color.FromArgb(0 , 0 , 0) , 1);
  Rectangle rect = new Rectangle(10 , 10 , 200 , 100);

  g.DrawRectangle(myPen , rect);
```

```
  g.DrawRectangle(myPen , 20 , 20 , 200 , 100);
 }
}
```



このプログラムは、Rectangle 構造体を用いた方法と
数値型の情報で矩形の領域を表す2通りの方法で長方形を描画しています

複数の長方形を同時に描画する場合
Rectangle の配列を使う **Graphics.DrawRectangles()** を使えます

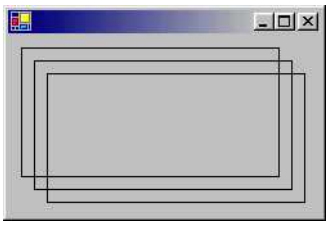
public void DrawRectangles(Pen pen , Rectangle[] rects);
public void DrawRectangles(Pen pen , RectangleF[] rects);

rects には、描画する矩形の情報を格納した構造体の配列を指定します
RectangleF とは **System.Drawing.RectangleF** 構造体のことで
PointF 構造体などの用に、各情報を int 型ではなく float 型を用いて表現しています

```
using System.Windows.Forms;
using System.Drawing;

class WinMain : Form {
 public static void Main(string[] args) {
  Application.Run(new WinMain());
 }
 override protected void OnPaint(PaintEventArgs e) {
  Graphics g = e.Graphics;
  Pen myPen = new Pen(Color.FromArgb(0 , 0 , 0) , 1);
  Rectangle[] rect = {
   new Rectangle(10 , 10 , 200 , 100) ,
   new Rectangle(20 , 20 , 200 , 100) ,
   new Rectangle(30 , 30 , 200 , 100)
  };

  g.DrawRectangles(myPen , rect);
 }
}
```



このプログラムは、3つの Rectangle 構造体の配列から長方形を描画しています
同一のペンで、複数の長方形を描く場合はこのようにしてプログラムします

円を描画する

円弧を描画する場合 **Graphics.DrawEllipse()** メソッドを使います
これは、指定した長方形に外接した円を描画します

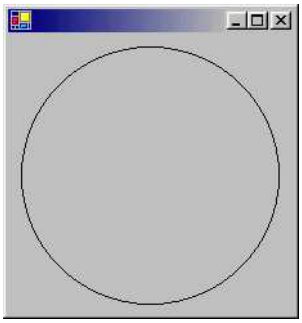
public void DrawEllipse(Pen pen , Rectangle rect);
public void DrawEllipse(Pen pen , RectangleF rect);
public void DrawEllipse(Pen pen , int x , int y , int width , int height);
public void DrawEllipse(Pen pen , float x , float y , float width , float height);

pen には、円弧の輪郭を描画するペンの Pen オブジェクトを指定します
rect には、円の外接する長方形をあらわす Rectangle または RectangleF を
x と y は、長方形の左上角を示す X 座標と Y 座標を
width には長方形の幅、height には高さをそれぞれ指定します

```
using System.Windows.Forms;
using System.Drawing;

class WinMain : Form {
 public static void Main(string[] args) {
  Application.Run(new WinMain());
 }
 override protected void OnPaint(PaintEventArgs e) {
  Graphics g = e.Graphics;
  Pen myPen = new Pen(Color.FromArgb(0 , 0 , 0) , 1);

  g.DrawEllipse(myPen , 10 , 10 , 200 , 200);
 }
}
```



このプログラムは、(10 , 10) から (200 , 200) ピクセルの大きさの長方形に対し
これに外接する円弧を描画しています

円弧は完全に閉じた円でしたが、そうでない円を描画したい場合もあるでしょう
つまり、ある角度から、指定した角度までの円です
これには **Graphics.DrawArc()** メソッドを使います

```
public void DrawArc(
 Pen pen , Rectangle rect ,
 float startAngle , float sweepAngle
);
public void DrawArc(
 Pen pen , RectangleF rect ,
 float startAngle , float sweepAngle
);
public void DrawArc(
 Pen pen ,
 int x , int y , int width , int height ,
 int startAngle , int sweepAngle
);
public void DrawArc(
 Pen pen ,
 float x , float y , float width , float height ,
 float startAngle , float sweepAngle
);
```

pen は円の輪郭を描画するペンの Pen オブジェクトを指定します
rect には、描画する円が外接する長方形をあらわす構造体を
x と y は、長方形の左上角を表す座標、width と height は幅と高さを指定します
ここまでは、DrawEllipse() メソッドと考え方は同じです

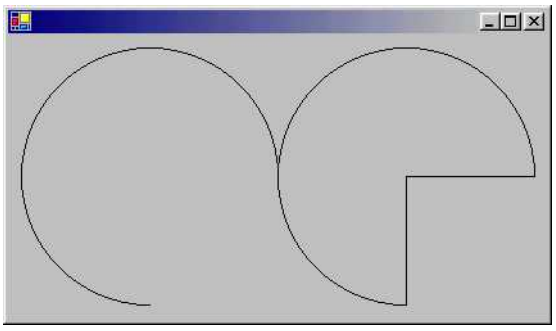
startAngle は円の描画を開始する角度を指定します
sweepAngle は startAngle から時計回りで終端角度を指定します
つまり、startAngle に 0、sweepAngle に 180 と指定したと考えれば
円は 0 度から時計回りに 180 度までが描画されます

DrawArc() メソッドは、円の輪郭を単純に描画するだけですが
円を閉じた図形として描画したい場合は **Graphics.DrawPie()** を使います
パックマンのような形の円を描画したい場合にこれを使うことができます
このメソッドのパラメータは DrawArc() メソッドと同じなので省略します

```
using System.Windows.Forms;
using System.Drawing;

class WinMain : Form {
 public static void Main(string[] args) {
  Application.Run(new WinMain());
 }
 override protected void OnPaint(PaintEventArgs e) {
  Graphics g = e.Graphics;
  Pen myPen = new Pen(Color.FromArgb(0 , 0 , 0) , 1);

  g.DrawArc(myPen , 10 , 10 , 200 , 200 , 0 , -270);
  g.DrawPie(myPen , 210 , 10 , 200 , 200 , 0 , -270);
 }
}
```



このプログラムは、DrawArc() と DrawPie() の2つのメソッドを使って
同じ大きさ、同じ角度のパラメータでその違いを比較するための描画しています
Arc は円を閉じませんが、Pie は円を閉じるというところに違いがあります
