

Graphics

Constructors

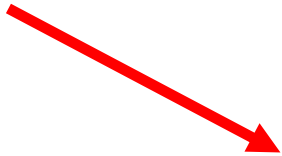
Constructors always have the same name as the class.

GraphOne test = new GraphOne();

Monster rob = new Monster();

Constructors

reference variable



```
Scanner keyboard =  
    new Scanner(System.in);
```



object instantiation / constructor call

Constructors

```
public class GraphicsRunner extends JFrame  
{
```

```
    private static final int WIDTH = 640;  
    private static final int HEIGHT = 480;
```

```
    public GraphicsRunner()  
    {
```

the constructor



```
        setSize(WIDTH,HEIGHT);  
        getContentPane().add( new Circles() );  
        setVisible(true);  
    }
```

```
    public static void main( String args[] )  
    {
```

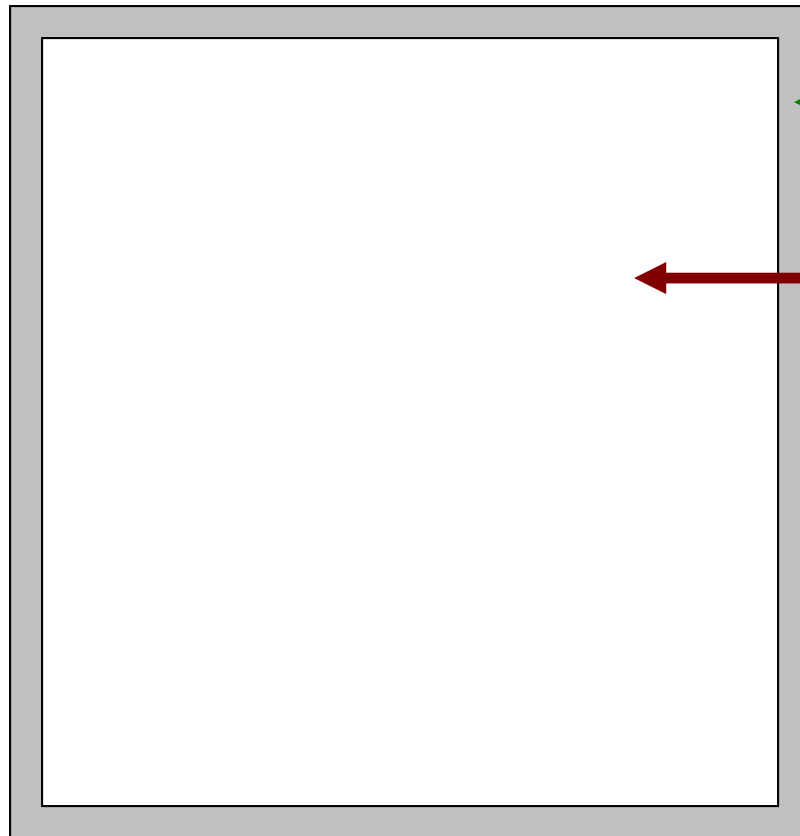
constructor call



```
        GraphicsRunner run = new GraphicsRunner();  
    }
```

```
}
```

Frame



Frame / JFrame

Canvas / JPanel

The Frame is used to hold up / display a Canvas or Panel.

paintO

```
public class Circles extends Canvas  
{
```

```
//constructors
```

```
public void paint( Graphics window )  
{  
    window.setColor(Color.BLACK);  
    window.drawString("Circles", 50, 50);  
  
    window.setColor(Color.BLUE);  
    window.drawOval(500,300,40,40);  
}
```

```
//other methods
```

```
}
```

paint



paint() is called automatically when you instantiate the class containing the paint method.

When an event is triggered that requires a redraw, paint is called again.

To call paint() without a Graphics parameter, you can use the repaint() method.

Open

graphicsrunner.java

circles.java

Parameters and Graphics methods

Graphics

frequently used methods

Name	Use
setColor(x)	sets the current drawing color to x
drawString(s,x,y)	draws String s at spot x,y
drawOval(x,y,w,h)	draws an unfilled oval at spot x,y that is w wide and h tall
fillOval(x,y,w,h)	draws a filled oval at spot x,y that is w wide and h tall

```
import java.awt.Graphics;  
import java.awt.Color;  
import javax.swing.JFrame;
```

passing parameters

A parameter/argument is a channel used to pass information to a method. `setColor()` is a method of the `Graphics` class that receives a `Color`.

void setColor(Color theColor)



The diagram illustrates the process of passing parameters in a method call. It features a large blue-bordered box containing the code `window.setColor(Color.RED);`. Two red arrows originate from this box: one points from the `setColor` method name to the `void setColor` part of the signature above, and the other points from the `Color.RED` argument to the `Color theColor` parameter part of the signature above. Below the code, a smaller blue-bordered box contains the text `method call with parameter`.

```
window.setColor( Color.RED );
```

method call with parameter

passing parameters

void fillRect (int x, int y, int width, int height)



The diagram consists of four red arrows pointing upwards from the parameters of a method call to the parameters of a method signature. The first arrow points from the value '10' to the parameter 'int x'. The second arrow points from the value '50' to the parameter 'int y'. The third arrow points from the value '30' to the parameter 'int width'. The fourth arrow points from the value '70' to the parameter 'int height'. The method call is enclosed in a blue rectangular box, and the text 'method call with parameters' is written in blue below it.

window.fillRect(10, 50, 30, 70);

method call with parameters

passing parameters

void fillRect(int x, int y, int width, int height)

window.fillRect(10, 50, 30, 70);

Four red arrows originate from the arguments in the function call 'window.fillRect(10, 50, 30, 70);' and point to the corresponding parameters in the function signature 'void fillRect(int x, int y, int width, int height)'. The arrows connect '10' to 'x', '50' to 'y', '30' to 'width', and '70' to 'height'.

The call to fillRect would draw a rectangle at position 10,50 with a width of 30 and a height of 70.

Graphics

frequently used methods

Name	Use
drawLine(a,b,c,d)	draws a line starting at point a,b and going to point c,d
drawRect(x,y,w,h)	draws an unfilled rectangle at spot x,y that is w wide and h tall
fillRect(x,y,w,h)	draws a filled rectangle at spot x,y that is w wide and h tall

```
import java.awt.Graphics;  
import java.awt.Color;  
import javax.swing.JFrame;
```

The Graphics Screen



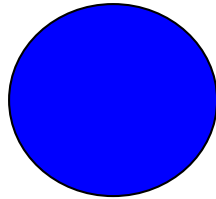
The Graphics Screen

0,0

X goes across →

**Y
goes
down**

X=100 y=100



width=50 height=50

```
window.fillOval( 100, 100, 50, 50 );
```

Rectangles

```
public void paint( Graphics window )  
{  
    window.setColor(Color.BLUE);  
    window.fillRect(150, 300, 100, 20 );  
    window.setColor(Color.GRAY);  
    window.drawRect(200,80,50,50);  
}
```


Open

rectangles.java

**Open
lines.java**

Graphics

frequently used methods

Name	Use
<code>drawArc(x,y,w,h,startAngle,arcAngle)</code>	draws an arc at spot x,y that is w wide and h tall
<code>fillArc(x,y,w,h,startAngle,arcAngle)</code>	draws a filled arc at spot x,y that is w wide and h tall
<code>startAngle</code> specifies the start of the arc <code>arcAngle</code> specifies the length of the arc	

```
import java.awt.Graphics;  
import java.awt.Color;  
import javax.swing.JFrame;
```

Open
ares.java

**Open
fonts.java**

Open
colors.java