Class java.awt.Graphics

```
java.lang.Object
   |
   +----java.awt.Graphics
```

---

public class **Graphics**
extends Object

Graphics is the abstract base class for all graphic contexts for various devices.

Version:
1.23, 09/18/95
Author:
Sami Shaio, Arthur van Hoff

Constructor Index

o **Graphics**()
Constructs a new Graphics Object.

Method Index

o **clearRect**(int, int, int, int)
Clears the specified rectangle by filling it with the current background color of the current drawing surface.
o **clipRect**(int, int, int, int)
Clips to a rectangle.
o **copyArea**(int, int, int, int, int, int)
Copies an area of the screen.
o **create**()
Creates a new Graphics Object that is a copy of the original Graphics Object.
o **create**(int, int, int, int)
Creates a new Graphics Object with the specified parameters, based on the original Graphics Object.
o **dispose**()
Disposes of this graphics context.

- **draw3DRect**(int, int, int, int, boolean)
  Draws a highlighted 3−D rectangle.
- **drawArc**(int, int, int, int, int, int)
  Draws an arc bounded by the specified rectangle from startAngle to endAngle.
- **drawBytes**(byte[], int, int, int, int)
  Draws the specified bytes using the current font and color.
- **drawChars**(char[], int, int, int, int)
  Draws the specified characters using the current font and color.
- **drawImage**(Image, int, int, ImageObserver)
  Draws the specified image at the specified coordinate (x, y).
- **drawImage**(Image, int, int, int, int, ImageObserver)
  Draws the specified image inside the specified rectangle.
- **drawLine**(int, int, int, int)
  Draws a line between the coordinates (x1,y1) and (x2,y2).
- **drawOval**(int, int, int, int)
  Draws an oval inside the specified rectangle using the current color.
- **drawPolygon**(int[], int[], int)
  Draws a polygon defined by an array of x points and y points.
- **drawPolygon**(Polygon)
  Draws a polygon defined by the specified point.
- **drawRect**(int, int, int, int)
  Draws the outline of the specified rectangle using the current color.
- **drawRoundRect**(int, int, int, int, int, int)
  Draws an outlined rounded corner rectangle using the current color.
- **drawString**(String, int, int)
  Draws the specified String using the current font and color.
- **fill3DRect**(int, int, int, int, boolean)
  Paints a highlighted 3−D rectangle using the current color.
- **fillArc**(int, int, int, int, int, int)
  Fills an arc using the current color.
- **fillOval**(int, int, int, int)
  Fills an oval inside the specified rectangle using the current color.
- **fillPolygon**(int[], int[], int)
  Fills a polygon with the current color.
- **fillPolygon**(Polygon)
  Fills the specified polygon with the current color.
- **fillRect**(int, int, int, int)
  Fills the specified rectangle with the current color.
- **fillRoundRect**(int, int, int, int, int, int)
  Draws a rounded rectangle filled in with the current color.
- **finalize**()
  Disposes of this graphics context once it is no longer referenced.
- **getClipRect**()
  Returns the bounding rectangle of the current clipping area.
- **getColor**()
  Gets the current color.
- **getFont**()
  Gets the current font.
- **getFontMetrics**()

Gets the current font metrics.

- **getFontMetrics**(Font)
  Gets the current font metrics for the specified font.
- **scale**(float, float)
  Scales the graphics context.
- **setColor**(Color)
  Sets the current color to the specified color.
- **setFont**(Font)
  Sets the font for all subsequent text−drawing operations.
- **setPaintMode**()
  Sets the default paint mode to overwrite the destination with the current color.
- **setXORMode**(Color)
  Sets the paint mode to alternate between the current color and the new specified color.
- **toString**()
  Returns a String object representing this Graphic's value.
- **translate**(int, int)
  Translates the specified parameters into the origin of the graphics context.

Constructors

- **Graphics**

```
protected Graphics()
```

Constructs a new Graphics Object. Graphic contexts cannot be created directly. They must be obtained from another graphics context or either a component can create them.

See Also:

getGraphics, create

Methods

- **create**

```
public abstract Graphics create()
```

Creates a new Graphics Object that is a copy of the original Graphics Object.

- **create**

```
public Graphics create(int x,
                       int y,
                       int width,
                       int height)
```

Creates a new Graphics Object with the specified parameters, based on the original Graphics Object. This method translates the specified parameters, x and y, to the proper origin coordinates and then clips the Graphics Object to the area.

Parameters:

x – the x coordinate
y – the y coordinate
width – the width of the area
height – the height of the area

See Also:

translate

o **translate**

```
public abstract void translate(int x,
                               int y)
```

Translates the specified parameters into the origin of the graphics context. All subsequent operations on this graphics context will be relative to this origin.

Parameters:

x – the x coordinate
y – the y coordinate

See Also:

scale

o **scale**

```
public abstract void scale(float sx,
                           float sy)
```

Scales the graphics context. All subsequent operations on this graphics context will be affected.

Parameters:

sx – the scaled x coordinate
sy – the scaled y coordinate

See Also:

translate

o **getColor**

```
public abstract Color getColor()
```

Gets the current color.

See Also:

setColor

o **setColor**

```
public abstract void setColor(Color c)
```

Sets the current color to the specified color. All subsequent graphics operations will use this specified color.

Parameters:

c – the color to be set

See Also:

Color, getColor

o **setPaintMode**

```
public abstract void setPaintMode()
```

Sets the default paint mode to overwrite the destination with the current color.

o **setXORMode**

```
public abstract void setXORMode(Color c1)
```

Sets the paint mode to alternate between the current color and the new specified color.
Parameters:
c1 – the second color

o **getFont**

```
public abstract Font getFont()
```

Gets the current font.
See Also:
setFont

o **setFont**

```
public abstract void setFont(Font font)
```

Sets the font for all subsequent text–drawing operations.
Parameters:
font – the specified font
See Also:
Font, getFont, drawString, drawBytes, drawChars

o **getFontMetrics**

```
public FontMetrics getFontMetrics()
```

Gets the current font metrics.
See Also:
getFont

o **getFontMetrics**

```
public abstract FontMetrics getFontMetrics(Font f)
```

Gets the current font metrics for the specified font.
Parameters:
f – the specified font

See Also:

getFont, getFontMetrics

o **getClipRect**

```
public abstract Rectangle getClipRect()
```

Returns the bounding rectangle of the current clipping area.

See Also:

clipRect

o **clipRect**

```
public abstract void clipRect(int x,
                              int y,
                              int width,
                              int height)
```

Clips to a rectangle. The resulting clipping area is the intersection of the current clipping area and the specified rectangle. Graphic operations have no effect outside of the clipping area.

Parameters:

x – the x coordinate
y – the y coordinate
width – the width of the rectangle
height – the height of the rectangle

See Also:

getClipRect

o **copyArea**

```
public abstract void copyArea(int x,
                              int y,
                              int width,
                              int height,
                              int dx,
                              int dy)
```

Copies an area of the screen.

Parameters:

x – the x–coordinate of the source
y – the y–coordinate of the source
width – the width
height – the height
dx – the horizontal distance
dy – the vertical distance

o **drawLine**

```
public abstract void drawLine(int x1,
                              int y1,
                              int x2,
```

```
                                int y2)
```

Draws a line between the coordinates (x1,y1) and (x2,y2). The line is drawn below and to the left of the logical coordinates.

Parameters:

x1 – the first point's x coordinate
y1 – the first point's y coordinate
x2 – the second point's x coordinate
y2 – the second point's y coordinate

o **fillRect**

```
public abstract void fillRect(int x,
                              int y,
                              int width,
                              int height)
```

Fills the specified rectangle with the current color.

Parameters:

x – the x coordinate
y – the y coordinate
width – the width of the rectangle
height – the height of the rectangle

See Also:

drawRect, clearRect

o **drawRect**

```
public void drawRect(int x,
                     int y,
                     int width,
                     int height)
```

Draws the outline of the specified rectangle using the current color. Use drawRect(x, y, width−1, height−1) to draw the outline inside the specified rectangle.

Parameters:

x – the x coordinate
y – the y coordinate
width – the width of the rectangle
height – the height of the rectangle

See Also:

fillRect, clearRect

o **clearRect**

```
public abstract void clearRect(int x,
                               int y,
                               int width,
                               int height)
```

Clears the specified rectangle by filling it with the current background color of the current drawing surface. Which drawing surface it selects depends on how the graphics context was created.

Parameters:

x – the x coordinate
y – the y coordinate
width – the width of the rectangle
height – the height of the rectangle

See Also:

fillRect, drawRect

o **drawRoundRect**

```
public abstract void drawRoundRect(int x,
                                   int y,
                                   int width,
                                   int height,
                                   int arcWidth,
                                   int arcHeight)
```

Draws an outlined rounded corner rectangle using the current color.

Parameters:

x – the x coordinate
y – the y coordinate
width – the width of the rectangle
height – the height of the rectangle
arcWidth – the diameter of the arc
arcHeight – the radius of the arc

See Also:

fillRoundRect

o **fillRoundRect**

```
public abstract void fillRoundRect(int x,
                                   int y,
                                   int width,
                                   int height,
                                   int arcWidth,
                                   int arcHeight)
```

Draws a rounded rectangle filled in with the current color.

Parameters:

x – the x coordinate
y – the y coordinate
width – the width of the rectangle
height – the height of the rectangle
arcWidth – the diameter of the arc
arcHeight – the radius of the arc

See Also:

drawRoundRect

o **draw3DRect**

```
public void draw3DRect(int x,
                       int y,
                       int width,
                       int height,
                       boolean raised)
```

Draws a highlighted 3–D rectangle.
Parameters:
x – the x coordinate
y – the y coordinate
width – the width of the rectangle
height – the height of the rectangle
raised – a boolean that states whether the rectangle is raised or not

o **fill3DRect**

```
public void fill3DRect(int x,
                       int y,
                       int width,
                       int height,
                       boolean raised)
```

Paints a highlighted 3–D rectangle using the current color.
Parameters:
x – the x coordinate
y – the y coordinate
width – the width of the rectangle
height – the height of the rectangle
raised – a boolean that states whether the rectangle is raised or not

o **drawOval**

```
public abstract void drawOval(int x,
                              int y,
                              int width,
                              int height)
```

Draws an oval inside the specified rectangle using the current color.
Parameters:
x – the x coordinate
y – the y coordinate
width – the width of the rectangle
height – the height of the rectangle
See Also:
fillOval

o **fillOval**

```
public abstract void fillOval(int x,
```

```
                                int y,
                                int width,
                                int height)
```

Fills an oval inside the specified rectangle using the current color.

Parameters:

x – the x coordinate
y – the y coordinate
width – the width of the rectangle
height – the height of the rectangle

See Also:

drawOval

o **drawArc**

```
public abstract void drawArc(int x,
                             int y,
                             int width,
                             int height,
                             int startAngle,
                             int arcAngle)
```

Draws an arc bounded by the specified rectangle from startAngle to endAngle. 0 degrees is at the 3–o'clock position.Positive arc angles indicate counter–clockwise rotations, negative arc angles are drawn clockwise.

Parameters:

x – the x coordinate
y – the y coordinate
width – the width of the rectangle
height – the height of the rectangle
startAngle – the beginning angle
arcAngle – the angle of the arc (relative to startAngle).

See Also:

fillArc

o **fillArc**

```
public abstract void fillArc(int x,
                             int y,
                             int width,
                             int height,
                             int startAngle,
                             int arcAngle)
```

Fills an arc using the current color. This generates a pie shape.

Parameters:

x – the x coordinate
y – the y coordinate
width – the width of the arc
height – the height of the arc
startAngle – the beginning angle

arcAngle – the angle of the arc (relative to startAngle).

See Also:

drawArc

o **drawPolygon**

```
public abstract void drawPolygon(int xPoints[],
                                 int yPoints[],
                                 int nPoints)
```

Draws a polygon defined by an array of x points and y points.

Parameters:

xPoints – an array of x points
yPoints – an array of y points
nPoints – the total number of points

See Also:

fillPolygon

o **drawPolygon**

```
public void drawPolygon(Polygon p)
```

Draws a polygon defined by the specified point.

Parameters:

p – the specified polygon

See Also:

fillPolygon

o **fillPolygon**

```
public abstract void fillPolygon(int xPoints[],
                                 int yPoints[],
                                 int nPoints)
```

Fills a polygon with the current color.

Parameters:

xPoints – an array of x points
yPoints – an array of y points
nPoints – the total number of points

See Also:

drawPolygon

o **fillPolygon**

```
public void fillPolygon(Polygon p)
```

Fills the specified polygon with the current color.

Parameters:

p – the polygon

See Also:

drawPolygon

o **drawString**

```
public abstract void drawString(String str,
                                int x,
                                int y)
```

Draws the specified String using the current font and color. The x,y position is the starting point of the baseline of the String.

Parameters:

str – the String to be drawn
x – the x coordinate
y – the y coordinate

See Also:

drawChars, drawBytes

o **drawChars**

```
public void drawChars(char data[],
                      int offset,
                      int length,
                      int x,
                      int y)
```

Draws the specified characters using the current font and color.

Parameters:

data – the array of characters to be drawn
offset – the start offset in the data
length – the number of characters to be drawn
x – the x coordinate
y – the y coordinate

See Also:

drawString, drawBytes

o **drawBytes**

```
public void drawBytes(byte data[],
                      int offset,
                      int length,
                      int x,
                      int y)
```

Draws the specified bytes using the current font and color.

Parameters:

data – the data to be drawn
offset – the start offset in the data
length – the number of bytes that are drawn
x – the x coordinate
y – the y coordinate

See Also:

drawString, drawChars

o **drawImage**

```
public abstract boolean drawImage(Image img,
                                  int x,
                                  int y,
                                  ImageObserver observer)
```

Draws the specified image at the specified coordinate (x, y). If the image is incomplete the image observer will be notified later.

Parameters:

img – the specified image to be drawn
x – the x coordinate
y – the y coordinate
observer – notifies if the image is complete or not

See Also:

Image, ImageObserver

o **drawImage**

```
public abstract boolean drawImage(Image img,
                                  int x,
                                  int y,
                                  int width,
                                  int height,
                                  ImageObserver observer)
```

Draws the specified image inside the specified rectangle. The image is scaled if necessary. If the image is incomplete the image observer will be notified later.

Parameters:

img – the specified image to be drawn
x – the x coordinate
y – the y coordinate
width – the width of the rectangle
height – the height of the rectangle
observer – notifies if the image is complete or not

See Also:

Image, ImageObserver

o **dispose**

```
public abstract void dispose()
```

Disposes of this graphics context. The Graphics context cannot be used after being disposed of.

See Also:

finalize

o **finalize**

```
public void finalize()
```

Disposes of this graphics context once it is no longer referenced.

See Also:

dispose

o **toString**

```
public String toString()
```

Returns a String object representing this Graphic's value.

Overrides:

toString in class Object