

Quick Reference

This appendix contains information on many of the useful objects and methods related to this book. Some are used in the book, others are just extras you may want to explore to add extra capability or features to your projects.

Recall that a **GRect**, **GOval**, **GLabel**, **GLine**, **GImage** or **GCompound** is also a **GObject**. For this discussion assume below that *gObj* is a **GObject**, which means that when you see *gObj*, it can be a **GRect**, **GOval** or other ACM Java graphic object.

A:1 the Graphics Window

A.1.1 Set Size of the Window

public static final int APPLICATION_HEIGHT = <i>xSize</i> ;	set width of window
public static final int APPLICATION_WIDTH = <i>ySize</i> ;	set height of window

A.1.2 Add and Remove Items from the Window

add(GObject gObj) ;	add <i>gObj</i> to window at its location
add(GObject gObj, int xLoc, int yLoc)	add <i>gObj</i> to window at (<i>xLoc</i> , <i>yLoc</i>)
add(GObject gObj, GPoint gp)	add <i>gObj</i> to window at <i>gp</i>
remove(GObject gObj) ;	remove <i>gObj</i> from window
removeAll() ;	remove all graphic objects from the window

A.1.3 Miscellaneous

setBackground(Color cv) ;	set the background color of the window to <i>cv</i>
GObject getElementAt(GPoint gp)	returns the GObject at <i>gp</i>
GObject getElementAt(double xLoc, double yLoc)	returns the GObject at (<i>xLoc</i> , <i>yLoc</i>)

A.1.4 Colors

There are several already named colors which you can use, but you must import the appropriate Java library.

import java.awt.* ;	import the awt library required to use named colors
----------------------------	---

Named Colors	
<u>common colors</u>	<u>other predefined colors</u>
Color.BLACK	Color.PINK
Color.BLUE	Color.CYAN
Color.GREEN	Color.DARK_GRAY
Color.RED	Color.LIGHT_GRAY
Color.ORANGE	Color.MAGENTA
Color.WHITE	
Color.YELLOW	
Color. GRAY	

Color <i>cv</i> = new Color(int <i>r</i>, int <i>g</i>, int <i>b</i>)	create a new color using rgb color codes. Google "w3c rgb" to find the codes
--	--

A. 2 Graphics Objects & Their Methods

A.2.1 Methods Common to All GObjects

size methods

double <i>gObj</i>.getHeight()	return the height of <i>gObj</i>
double <i>gObj</i>.getWidth()	return the width of <i>gObj</i>

color & visibility methods

void <i>gObj</i>.setVisible(boolean <i>bv</i>)	set <i>gObj</i> to be visible (if <i>bv</i> is true) or invisible (<i>bv</i> is false)
boolean <i>gObj</i>.isVisible()	return the visibility of <i>gObj</i>

location methods

void <i>gObj</i>.move(double <i>xMove</i>, double <i>yMove</i>)	move <i>gObj</i> relative to current position
void <i>gObj</i>.movePolar(double <i>radius</i>, double <i>angle</i>)	move <i>gObj</i> <i>radius</i> distance at <i>angle</i> (<i>angle</i> is measured in radians) relative to current position
void <i>gObj</i>.setLocation(double <i>xLoc</i>, double <i>yLoc</i>)	move <i>gObj</i> to (<i>xLoc</i> , <i>yLoc</i>)
double <i>gObj</i>.getX()	return the x location of <i>gObj</i>
double <i>gObj</i>.getY()	return the y coordinate of <i>gObj</i>
GPoint <i>gObj</i>.getLocation()	return the location of <i>gObj</i> as a GPoint

stacking order methods

<code>void gObj.sendBackward()</code>	send <i>gObj</i> backward one level in the stacking order
<code>void gObj.sendForward()</code>	send <i>gObj</i> forward one level in the stacking order
<code>void gObj.sendToBack()</code>	send <i>gObj</i> all the way to the back in the stacking order
<code>void gObj.sendToFront()</code>	send <i>gObj</i> all the way to the front in the stacking order

containment & intersection methods

<code>boolean gObj.contains(double x, double y)</code>	returns true if (x, y) is in <i>gObj</i> , false otherwise
<code>boolean gObj.contains(GPoint pt)</code>	return true if <i>pt</i> is in <i>gObj</i> , false otherwise
<code>GRectangle gObj.getBounds()</code>	return the GRectangle that bounds <i>gObj</i>
<code>boolean gr1.intersects(gr2)</code>	return true if <i>gr1</i> intersects <i>gr2</i> , false otherwise. <i>gr1</i> and <i>gr2</i> are GRectangles .

A.2.2 Additional Methods for GRects

constructors

<code>G Rect rect = new GRect(double xSize, double ySize)</code>	construct a new GRect of size <i>xSize</i> by <i>ySize</i>
<code>GRect rect = new GRect(double xLoc, double yLoc, double xSize, double ySize)</code>	construct a new GRect of size <i>xSize</i> by <i>ySize</i> at $(xLoc, yLoc)$

color & visibility

<code>void rect.setColor(Color cv)</code>	set the color to <i>cv</i>
<code>Color rect.getColor()</code>	return the color
<code>Color rect.getFillColor()</code>	return the fill color
<code>boolean rect.isFilled()</code>	return true if <i>rect</i> is filled, false otherwise
<code>void rect.setFill(Color cv)</code>	set the fill color to <i>cv</i>
<code>void rect.setFill(boolean fill)</code>	set <i>rect</i> to be filled with <i>fill</i> = true and not filled with <i>fill</i> = false

size

<code>void rect.setSize(double xSize, double ySize)</code>	set the size to <i>xSize</i> by <i>ySize</i>
<code>double rect.getHeight()</code>	return the height of <i>rect</i>
<code>double rect.getWidth()</code>	return the width of <i>rect</i>
<code>void rect.scale(double scaleFactor)</code>	scale the object vertically and horizontally by

	<i>scaleFactor</i>
void <i>rect</i> . scale (double <i>scaleFactorX</i> , double <i>scaleFactorY</i>)	scale the object vertically by <i>scaleFactorX</i> and horizontally by <i>scaleFactorY</i>

A.2.3 Additional Methods for GOvals

constructors

GOval <i>oval</i> = new GRect (double <i>xSize</i> , double <i>ySize</i>)	construct a new GOval of size <i>xSize</i> by <i>ySize</i>
GOval <i>oval</i> = new GRect (double <i>xLoc</i> , double <i>yLoc</i> , double <i>xSize</i> , double <i>ySize</i>)	construct a new GOval of size <i>xSize</i> by <i>ySize</i> at (<i>xLoc</i> , <i>yLoc</i>)

color & visibility

void <i>oval</i> . setColor (Color <i>cv</i>)	set the color to <i>cv</i>
Color <i>oval</i> . getColor ()	return the color
Color <i>oval</i> . getFillColor ()	return the fill color
boolean <i>oval</i> . isFilled ()	return true if <i>oval</i> is filled, false otherwise
void <i>oval</i> . setFillColor (Color <i>cv</i>)	set the fill color to <i>cv</i>
void <i>oval</i> . setFilled (boolean <i>fill</i>)	set <i>oval</i> to be filled with <i>fill</i> = true and not filled with <i>fill</i> = false

size

void <i>oval</i> . setSize (double <i>xSize</i> , double <i>ySize</i>)	set the size to <i>xSize</i> by <i>ySize</i>
double <i>oval</i> . getHeight ()	return the height of <i>rect</i>
double <i>oval</i> . getWidth ()	return the width of <i>rect</i>
void <i>oval</i> . scale (double <i>scaleFactor</i>)	scale the object vertically and horizontally by <i>scaleFactor</i>
void <i>oval</i> . scale (double <i>scaleFactorX</i> , double <i>scaleFactorY</i>)	scale the object vertically by <i>scaleFactorX</i> and horizontally by <i>scaleFactorY</i>

A.2.4 Additional Methods for GLabels

constructors

GLabel <i>label</i> = new GLabel (String <i>s</i>)	construct a new GLabel containing the String
GLabel <i>label</i> = new GLabel (String <i>s</i> , double <i>xLoc</i> , double <i>yLoc</i>)	construct a new GLabel containing the String at (<i>xLoc</i> , <i>yLoc</i>)

color

void <i>gObj</i> . setColor (Color <i>cv</i>)	sets the color of the label to <i>cv</i>
Color <i>gObj</i> . getColor ()	returns the color of the label

label and font

<code>void label.setLabel(String s)</code>	set the content of the label to String <i>s</i> , using a default style
<code>void label.setFont(String fontSpecifier)</code>	set the shape, style and size of the text, following the specifications below

font specification

<i>fontSpecifier</i>	a String following the pattern <i>fontFamily-fontStyle-SizeInPoints</i>
<i>fontFamily</i>	the font family may be Serif or SansSerif
<i>fontStyle</i>	the font style may be PLAIN , BOLD , ITALIC or BOLDITALIC
<i>fontSize</i>	measured in points and expressed as an integer
examples	" <i>Serif-PLAIN-14</i> " " <i>SansSerif-BOLDITALIC-20</i> " " <i>*-BOLD-10</i> " ← current font family, bold, size 10 " <i>SansSerif-*-*</i> " ← SansSerif family, current style size

A.2.5 Additional Methods for GLines

constructors

<code>GLine line = new GLine(double startX, double startY, double endX, double endY)</code>	construct a new GLine from (<i>startX</i> , <i>startY</i>) to (<i>endX</i> , <i>endY</i>)
---	--

start point, end point and location

<code>void line.setStartPoint(double xLoc, double yLoc)</code>	set the start point of the line to (<i>xLoc</i> , <i>yLoc</i>); the end point remains the same
<code>void line.setEndPoint(double xLoc, double yLoc)</code>	set the end point of the line to (<i>xLoc</i> , <i>yLoc</i>); the start point remains the same
<code>GPoint line.getStartPoint()</code>	returns the start point of the line as a GPoint
<code>GPoint line.getEndPoint()</code>	returns the end point of the line as a GPoint
<code>void line.setLocation(double xLoc, double yLoc)</code>	set the start point of the line to (<i>xLoc</i> , <i>yLoc</i>). the line maintains its length and orientation (angle in the graphics window)

color

<code>void gObj.setColor(Color cv)</code>	set the color of the line to <i>cv</i>
---	--

Color <i>gObj.getColor()</i>	returns the color of the line
-------------------------------------	-------------------------------

size

void <i>line.scale(double scaleFactor)</i>	
void <i>line.scale(double scaleFactorX, double scaleFactorY)</i>	

A.2.6 Additional Methods for GImages

constructors

GImage <i>image = new GImage(String imageFilename)</i>	construct a new GImage from the <i>imageFilename</i> . The image file may be an existing GIF (file extension is gif) or JPEG (file extension is jpg or jpeg)
GImage <i>image = new GImage(String imageFilename, double xLoc, double yLoc)</i>	creates a new GImage from the <i>imageFilename</i> at (<i>xLoc</i> , <i>yLoc</i>)

A.3 GPoints

constructors

GPoint <i>pt = new GPoint()</i>	construct a new GPoint at (0, 0)
GPoint <i>pt = new GPoint(double xLoc, double yLoc)</i>	construct a new GPoint at (<i>xLoc</i> , <i>yLoc</i>)
GPoint <i>pt = new GPoint(Gpoint oldPt)</i>	construct a new GPoint at <i>oldPt</i>

location

double <i>gpt.getX()</i>	return the <i>x</i> coordinate of <i>gpt</i>
double <i>gpt.getY()</i>	return the <i>y</i> coordinate of <i>gpt</i>
GPoint <i>gpt.getLocation()</i>	return a new GPoint that is equal to <i>gpt</i>
void <i>gpt.setLocation(double xLoc, double yLoc)</i>	set <i>gpt</i> to (<i>xLoc</i> , <i>yLoc</i>)
void <i>gpt.translate(double dX, double dY)</i>	adjust the coordinates of <i>gpt</i> by (<i>dX</i> , <i>dY</i>)

miscellaneous

boolean <i>gpt.equals(GPoint gPt2)</i>	return true if <i>gpt</i> equals <i>gpt2</i> , false otherwise
---	--

A.4 GRectangles

constructors

<code>GRectangle gr = new GRectangle()</code>	construct a new empty GRectangle <i>gr</i>
<code>GRectangle gr = new GRectangle(double width, double height)</code>	construct a new GRectangle <i>gr</i> <i>width</i> by <i>height</i> at (0, 0)
<code>GRectangle gr = new GRectangle(double xLoc, double yLoc, double width, double height)</code>	construct a new GRectangle <i>width</i> by <i>height</i> at (<i>xLoc</i> , <i>yLoc</i>)

location

<code>void gr.setLocation(double xLoc, double yLoc)</code>	move <i>gr</i> to (<i>xLoc</i> , <i>yLoc</i>)
<code>void gr.setLocation(GPoint pt)</code>	move <i>gr</i> to <i>pt</i>

micellaneous

<code>boolean gr.contains(double xLoc, double yLoc)</code>	returns true if GRectangle <i>gr</i> contains the point (<i>xLoc</i> , <i>yLoc</i>), false otherwise
<code>boolean gr.contains(GPoint gp)</code>	returns true if GRectangle <i>gr</i> contains the GPoint <i>gp</i> , false otherwise
<code>boolean gr.intersects(GRectangle gr2)</code>	returns true if GRectangle <i>gr</i> intersects the GRectangle <i>gr2</i>

A.5 MouseEvent

location

<code>e.getX()</code>	return the <i>x</i> coordinate of the MouseEvent <i>e</i>
<code>e.getY()</code>	return the <i>y</i> coordinate of the MouseEvent <i>e</i>

buttons

<code>MouseEvent.BUTTON1</code>	a constant that represents the left mouse button on a two button mouse
<code>MouseEvent.BUTTON3</code>	a constant that represents the right mouse button on a two button mouse
<code>e.getButton()</code>	returns the button which triggered the MouseEvent <i>e</i>

miscellaneous

<code>addMouseListeners()</code>	tells the program to pay attention to mouse clicks
----------------------------------	--

A.6 KeyEvents

keycodes

<code>int e.getKeyCode()</code>	return an int representing the key pressed
<code>KeyEvent.VK_LEFT</code>	a constant that represents the left arrow key
<code>KeyEvent.VK_RIGHT</code>	a constant that represents the right arrow key
<code>KeyEvent.VK_UP</code>	a constant that represents the up arrow key
<code>KeyEvent.VK_DOWN</code>	a constant that represents the down arrow key
<code>KeyEvent.VK_SPACE</code>	a constant that represents the down arrow key
<code>KeyEvent.VK_0</code> → <code>KeyEvent.VK_9</code>	constants that represent the 0 → 9 keys
<code>KeyEvent.VK_A</code> → <code>KeyEvent.VK_Z</code>	constants that represent the A → Z keys

miscellaneous

<code>addKeyListeners()</code>	tells the program to pay attention to key presses
--------------------------------	---

A.7 GCompounds

Recall that a **GCompound** is a **GObject** and so implements all of the methods described above for **GObjects**. Assume that *gc* is a **GCompound**. There are many methods for manipulating the elements that make up the GCompound, their characteristics and their locations.

add and remove

<code>void gc.add(GObject obj)</code>	add <i>obj</i> to <i>gc</i>
<code>void gc.add(GObject obj, double xLoc, double yLoc)</code>	add <i>obj</i> to <i>gc</i> at (<i>xLoc</i> , <i>yLoc</i>). The location is <i>within</i> the coordinate system of <i>gc</i> .
<code>gc.remove(GObject obj)</code>	remove <i>obj</i> from <i>gc</i>

miscellaneous

<code>boolean gc.contains(double x, double y)</code>	return true if <i>gc</i> contains (<i>x</i> , <i>y</i>), false otherwise
<code>GRectangle gc.getBounds()</code>	returns the GRectangle surrounding <i>gc</i>