Artemis Graphics

A simple graphics class which inherits from zellegraphics

Date: July 4, 2015

Updates: Aug 1, 2015

Aug 5, 2015

Author: Joanna Simwinga

Example:

CLASSNAME(PARENT_CLASS):

Important information about the class

__init__(self, things we need to make the class work):

Available functions(variables for things we need to make these methods work)→ What type of thing this method will return, if it returns nothing it will say None

An explanation of the variables

Note:

A class can do anything that a parent class can do. Because Rectangle is the child of zg.Rectangle, or it **inherits** from zg.Rectangle, you can call the zg.Rectangle's methods on the Artemis Rectangle.

These adaptable classes have been made to make the zellegraphics classes easier to use.
The Button class is adapted from a class found in a zellegraphics teaching book.

RGB Color:

To use rgb color write "import zellegraphics as zg" at the top of your file and use zg.rgb_color(rValue,gValue,bValue) in place of a color string.

Available Classes:

- **Window**
- **Point**
- **Text**
- **Image**
- **Rectangle**
- **Circle**
- **Triangle**
- **EquilateralTri**
- **LabeledObject**
- **LabeledCircle**
- **LabeledRect**
- **LabeledImage**
- **Button**
- **CircularButton**
- **ImageButton**
- **ButtonList**
- **Timer**
- **Entry**

Window(GraphWin):

__init__("title", width, heght):

Title: string that appears at top of GUI

width/height: integers

setBackground(color): → None

Changes the color of the window

Color: a color string (ex. 'black', 'white', 'blue') or as color_rgb(red, green, blue)

getMouse(): → Point where the mouse was clicked

Point(zg.Point):

__init__(x,y):

x,y: x and y values on coordinate system

getX(): → X value

getY(): → Y value

Text(zg.Text):

__init__(centerPoint, text):

centerPoint: a Point which is written as Point(x,y)

setSize(size): → None

size: an integer between 5 and 36

getSize(): → integer representing the size of the text

setFace(face): →None

face: string representing font family.

Possible strings: ‘helvetica’, ‘courier’, ‘times roman’, ‘arial’

getFace() → font family

setStyle(style)→ None

style: string representing style of font

Possible strings: ‘normal’, ‘bold’, ‘italic’, ‘bold italic’

getStyle() →style of font

setFill(color): → None

Changes the color of the rectangle

Color: a color string (ex. ‘black’, ‘white’, ‘blue’) or as color_rgb(red, green, blue)

getFill(): → Color

setOutline(color): → None

getOutline(): → Color

Image(zg.Image):

__init__(centerPoint, “filename”):

centerPoint: Point

filename: a string, the name of the image you want to use. Note: must be a .gif or .ppm

Rectangle(zg.Rectangle):

__init__(centerPoint, width, height):

centerPoint: a Point which is written as Point(x,y)

width/height: integers

setFill(color): → None

Changes the color of the rectangle

Color: a color string (ex. ‘black’, ‘white’, ‘blue’) or as color_rgb(red, green, blue)

getFill(): → Color

setOutline(color): → None

getOutline(): → Color

Circle(zg.Circle):

__init__(centerPoint, radius)

centerPoint: a Point which is written as Point(x,y)

width/height: integers

setFill(color): → None

Changes the color of the rectangle

Color: a color string (ex. ‘black’, ‘white’, ‘blue’) or as color_rgb(red, green, blue)

getFill(): → Color

setOutline(color): → None

getOutline(): → Color

Triangle(zg.Polygon)

__init__(p1,p2,p3)

p1,p2,p3: the three points of the triangle written as Point(x,y)

setFill(color): → None

Changes the color of the rectangle

Color: a color string (ex. ‘black’, ‘white’, ‘blue’) or as color_rgb(red, green, blue)

getFill(): → Color

setOutline(color): → None

getOutline(): → Color

EquilateralTri(Triangle):

__init__(p1, width)

p1: the top point of the triangle written as Point(x,y)

width: integer

setFill(color): → None

Changes the color of the rectangle

Color: a color string (ex. 'black', 'white', 'blue') or as color_rgb(red, green, blue)

getFill(): → Color

setOutline(color): → None

getOutline(): → Color

LabeledObject ():

__init__(centerPoint, width, height, text)

centerPoint: a Point which is written as Point(x,y)

width/height: integers

text: your label as a string

setWidth(width): → None

width: integer

getWidth() → width of object

setHeight(height)→ None

height: integer

getHeight() → height of object

setTextSize(size) → None

size of text: integer between 5 and 36

setTextPlacement(placement): → None

placement: a string representing where the text is placed in the object

Possible strings: ‘center’, ‘bottom’, ‘top’

setFill(color): → None

Changes the color of the rectangle

Color: a color string (ex. ‘black’, ‘white’, ‘blue’) or as color_rgb(red, green, blue)

getFill(): → Color

setOutline(color): → None

getOutline(): → Color

draw(graphwin):→ None

graphwin: window

LabledRect(LabledObject):

__init__(centerPoint, width, height, text)

centerPoint: a Point which is written as Point(x,y)

width/height: integers

text: a string representing your label

setFill(color): → None

Changes the color of the rectangle

Color: a color string (ex. ‘black’, ‘white’, ‘blue’) or as color_rgb(red, green, blue)

setTextColor(color): → None

LabledImage():

__init__(centerPoint, img, text)

centerPoint: a Point(x,y)

img: the filename written as a string ex. ‘picture.gif’ (.gif or .ppm files only)

setTextSize(size): →None

size: integer between 6-36

setTextPlacement(placement): → None

placement: a string representing where the text is placed in the object

Possible strings: ‘center’, ‘bottom’, ‘top’

setTextColor(color) → None

draw(graphwin)→ None

Button:

__init__(center, width, height, label):

Win: window

Center: center Point

Width/height: integers

Label: string

Clicked(p): → Boolean

P: point

getLabel()→ string (the button’s label)

activate() → None

deactivate() → None

setLabel(label):

label: string

setFill(color): → None

Changes the color of the rectangle

Color: a color string (ex. ‘black’, ‘white’, ‘blue’) or as color_rgb(red, green, blue)

getFill() → Color

setTextColor(color): → None

getTextColor()→ color