

CS 115 Lecture 7

More graphics

Neil Moore

Department of Computer Science
University of Kentucky
Lexington, Kentucky 40506
neil@cs.uky.edu

22 September 2015

FAQs: chapters 1–3

A few miscellaneous points we have gotten several questions about:

- IDE versus interpreter.
- Precedence of `*` and `/`.
- “Types” (AKA “data types”).
- Division and floating-point.
- Concatenating empty strings.
- Any other questions?

Drawing text

- You can draw text in the graphics window.
- Need the location (center point) and a string.
`txt = Text(Point(250, 250), "Hello")`
- You can specify the font size before drawing.
`txt.setSize(30) # between 5 and 36`
- Can also make it bold and/or italic:
`txt.setStyle('bold')`
`txt.setStyle('italic')`
`txt.setStyle('bold italic')`
`txt.setStyle('normal') # default`
- Supports a few typefaces:
`txt.setFace('courier')`
`txt.setFace('times roman')`

Changing coordinates

- In the default coordinate system:
 - ▶ (0,0) is in the upper left.
 - ▶ (*width*, *height*) is in the lower right.
 - ▶ Measures coordinates in pixels.
- Why might we want to change that?
 - ▶ To use different window sizes without changing most code.
 - ▶ Or maybe you want to do graphing: (0,0) at the bottom.
 - ▶ Maybe it just makes the math simpler.

`win.setCoords(xll, yll, xur, yur)`

- ▶ Give the *x* and *y* coordinates of:
 - ★ The lower left corner: *xll*, *yll*
 - ★ The upper right corner: *xur*, *yur*
- ▶ All drawing after this will be in this coordinate system.

`win.setCoords(0, 0, 1, 1)`

- ★ Now `Point(0, 1)` is the upper right.
- ★ `Point(0.5, 0.5)` is the center.
- ★ This is why coordinates can be floats, not just ints.

Interacting with the user

- What if a graphical program needs input from the user?
 - ▶ `input` uses standard input (the shell window)
 - ▶ Making the user switch back and forth is annoying.
 - ▶ ...and it doesn't even work right in WingIDE!

- We can make a graphical text-entry box.

```
entry = Entry(center, width)
```

- ▶ `center` is a point.
- ▶ `width` is a number of characters (**not pixels**).
 - ★ Just controls the size.
 - ★ The user can enter more characters (it scrolls).

- Can set the initial text, font size, color...

```
entry.setText("default")  
entry.setSize(24) # 24-point  
entry.setTextColor("green")
```

Getting user input from an Entry

```
entry = Entry(center, width)
```

- To get the text the user entered:

```
in = entry.getText()
```

- Returns a string (like `input`)

- ▶ Type-cast if you need a number:

```
temperature = float(entry.getText())
```

- Have to give the user time to enter their text.

- Wait for a mouse click:

```
entry.draw(win)
```

```
win.getMouse()
```

```
in = entry.getText()
```

- Let's write a program that uses an Entry box: `entry.py`

More about mouse clicks

- We've seen `win.getMouse()` to wait for a click.
- Wouldn't it be nice to know *where* the user clicked?
 - ▶ We don't even need a new method to do that!
 - ▶ `getMouse` actually returns a `Point`.
`clickpos = win.getMouse()`
 - ▶ Now we can get the `x` and `y` coordinates of the point.
`click_x = clickpos.getX()`
`click_y = clickpos.getY()`
 - ▶ Or we can use the `Point` directly.
`line = Line(Point(0, 0), clickpos)`
`line.draw(win)`
- When we called `getMouse` as a statement, we were just throwing this position away.
 - ▶ To wait for a click and get its location: `pt = win.getMouse()`
 - ▶ To just wait for a click: `win.getMouse()`

Aliasing

- You must be careful when using assignment with shapes (`alias.py`).

```
eye = Circle(Point(200, 250), 50)
eye.draw(win)
eye2 = eye
eye2.move(100, 0)
```

- ▶ This moves the first circle! What happened?
- ▶ There is only *one* circle here, with two different names.

- ★ They have the same **identity**:

```
print(id(eye)) →4147736844
print(id(eye2)) →4147736844
```

- ▶ We say that `eye2` is an **alias** for `eye`.
- ▶ You can check for aliasing with the `is` operator:

```
print(eye is eye2) →True
```

- ★ Not the same as asking if they're equal!
- ★ More on that next time.

Preventing aliasing

- Aliasing happens because assignment doesn't create new objects.
- To avoid aliasing, either:
 - ▶ Call the constructor every time you want to make a new object.

```
eye2 = Circle(Point(200, 250), 50)
print(id(eye2)) →4147339756
```
 - ▶ Or **clone** the object (graphics shapes only).

```
eye2 = eye.clone()
print(id(eye2)) →4148132104
```
 - ▶ `alias-fixed.py`
- Aliasing isn't a problem for integers, strings, etc.
 - ▶ These objects are **immutable**.
 - ★ The number 42 never changes.
 - ★ Immutable object can still be aliased, but since they can't be modified, the aliasing doesn't cause problems.
 - ▶ More detail in chapter 8 when we talk about lists.