Week 7
CSC111 — Fall 2015
Dynamic Web Page Example

IF Statements & Boolean Expression
An Application: Generating Dynamic Web Pages
Introduction to HTML

(Hyper Text Markup Language)
HTML uses TAGS

<title>CSC111 Web Page Example</title>
HTML uses TAGS

```
<title>CSC111 Web Page Example</title>
```
HTML uses TAGS

<title>CSC111 Web Page Example</title>
HTML uses TAGS

And if we want words to be in bold or italics, there are special tags for that too!
HTML uses TAGS

```html
<title>CSC111 Web Page Example</title>

And if we want words to be in <strong>bold</strong> or <em>italics</em>, there are special tags for that too!
Go to http://www.onlinehtmleditor.net/
<html>
  <head>
    <title>CSC111 Web Page Example</title>
  </head>
  <body>
    <h1>News of the Day</h1>
    Fall break was this past weekend, and was too short.
  </body>
</html>
<html>
  <head>
    <title>CSC111 Web Page Example</title>
  </head>
  <body>
    <h1>News of the Day</h1>
    Fall break was this past weekend, and was too short.
  </body>
</html>
An Application

Democratic Polls

<h1>Results</h1>
<data />

dempolls.raw
An Application

<html>
<head>
<title>
Democratic Polls
</title>
</head>
<body>
<h1>Results</h1>
<data />
</body>
</html>

dempolls.raw

dempolls.data

Biden 19.1
Chafee 0.6
Clinton 44.4
Lessig 0.0
O'Malley 1.0
Sanders 25.1
Webb 1.2
An Application

D. Thiebaut, Computer Science, Smith College

Democratic Polls

Results

Biden 19.1
Chafee 0.6
Clinton 44.4
Lessig 0.0
O'Malley 1.0
Sanders 25.1
Webb 1.2

dempolls.data

dempolls.raw
How to Write Programs in Top-Down Fashion

1. Start with the problem

2. **Break** problem down into a **few** simple parts

3. Call a **function** for each **part**

4. Take each part as a separate, **new, smaller problem**

5. **Go back** to Step 2 until the smaller problem can be solved by **just a few** Python **statements**.
Chapter 7 in Zelle
Bits, binary switch

Relational Operators

Boolean Operators

Teller Machine Revisited

Exercises
Bits & Boolean Values

• Bits: 0 and 1

• Boolean values: True and False

• Boolean switches: Imagine a world where every decision has a binary outcome:
  • Do you want to go out or do you want to stay in?
  • If you go out, do you walk or do you take the car?
if answer to question is True:
    True
else:
    False
Boolean Expression

if question:

else:

True

else:

False
Boolean Expression

if question:
  True
else:
  False

George Boole
1815-1864
(wikipedia.org)
Life Examples

if today is school day:
    go to class
else:
    go away for weekend

if food at Duckett > food at your house:
    go eat at Duckett
else:
    go eat at your house

if fire alarm rings:
    evacuate building
Life Examples

if driving parents' car:
    if low on fuel:
        if close to home:
            drive home and let parents will fill up car
        else:
            stop next gas station and buy minimum gas to get home
Python Example

Amount to withdraw? 71
3 $20-bill(s)
1 $10-bill(s)
0 $5-bill(s)
1 $1-bill(s)

**Exercise:** print ’s’ whenever necessary
Example

```python
amount = int(input("Amount? "))
no20s = amount // 20
amount = amount % 20

no10s = amount // 10
amount = amount % 10

no5s = amount // 5
no1s = amount % 5

print(no20s, "$20-bill(s)"
...```
Example

```python
amount = int( input( "Amount? " ) )

no20s = amount // 20
amount = amount % 20

no10s = amount // 10
amount = amount % 10

no5s = amount // 5
no1s = amount % 5

if no20s == 1:
    print( no20s, "$20-bill" )
else:
    print( no20s, "$20-bills" )

...```
Assume no20s contains 1...

```
if no20s == 1:
    print( no20s, "$20-bill" )
else:
    print( no20s, "$20-bills" )
```
Assume no20s contains 3…

```python
if no20s == 1:
    print( no20s, "$20-bill" )
else:
    print( no20s, "$20-bills" )
```
Bits, binary switch

Relational Operators

Logical Operators

Teller Machine Revisited

Exercises
# Relational Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>==</td>
<td>equal to</td>
</tr>
<tr>
<td>!=</td>
<td>not equal to</td>
</tr>
<tr>
<td>&lt;</td>
<td>less than</td>
</tr>
<tr>
<td>&lt;=</td>
<td>less than or equal to</td>
</tr>
<tr>
<td>&gt;</td>
<td>greater than</td>
</tr>
<tr>
<td>&gt;=</td>
<td>greater than or equal to</td>
</tr>
</tbody>
</table>
if no20s == 1:
    print( no20s, "$20-bill" )
else:
    print( no20s, "$20-bills" )

amount = input( "Amount to withdraw? " )
if amount > 400:
    print( "You are limited to $400 a week." )
    amount = 400

no20s, no10s, no5s, nols = breakDown( amount )
Coding Exercise

Recode the Teller-Machine program, so that

- the output correctly displays "bill" or "bills"
- a number of bills of 0 is not displayed
- only an amount less than $400 is allowed
- only amounts multiples of $5 are allowed.
def f0():
    a = 3
    b = 5
    c = 10

    if a < b:
        print( "statement 1" )
    else:
        print( "statement 2" )

    print( "f0 done!" )

f0()
def f0():
    a = 30  # <= changed!
    b = 5
    c = 10

    if a < b:
        print("statement 1")
    else:
        print("statement 2")

    print("f0 done!")

f0()
def f0():
a = 30  # <= changed!
b = 5
c = 10
if a < b:
    # Code block
else:
    # Code block
print( "f0 done!" )
def f0():
a = 3  # <= changed again!
b = 5
c = 10
if a < b:
    if c == 10:
        print( "statement 1" )
    else:
        print( "statement 2" )
else:
    print( "statement 3" )
print( "f0 done!" )
# Exercises (Group 1)

http://cs.smith.edu/dftwiki/index.php/CSC111_Exercises_with_If_Statements_(Python_3)

```python
def f0():
a = 3
b = 1  # <= changed!
c = 101  # <= changed!
if a < b:
    if c == 10:
        print( "statement 1" )
    else:
        print( "statement 2" )
else:
    print( "statement 3" )

print( "f0 done!" )
```

D. Thiebaut, Computer Science, Smith College
def f0():
a = 3
b = 1 # <= changed!
c = 101 # <= changed!
if a < b:
    if c == 10:
        print( "statement 1" )
    else:
        print( "statement 2" )
else:
    print( "statement 3" )
print( "f0 done!" )
We stopped here last time...
Bits, binary switch

Relational Operators

**Boolean Operators**

Teller Machine Revisited

Exercises
Boolean Operators
And, Or, Not
Boolean Operators
And, Or, Not

True and False are Python values!
if expression1 and expression2:
    statement
    statement
    statement
else:
    statement
    statement
    statement
if expression1 and expression2:
    statement
    statement
    statement
else:
    statement
    statement
    statement
if expression1 and expression2:
    statement
    statement
    statement
else:
    statement
    statement
    statement
if expression1 and expression2:
    statement
    statement
    statement
else:
    statement
    statement
    statement
    statement
if `expression1` and `expression2`:
    statement
    statement
    statement
else:
    statement
    statement
    statement
    statement
if expression1 and expression2:
  statement
  statement
  statement
else:
  statement
  statement
  statement
  statement
if expression1 and expression2:
  statement
  statement
  statement
else:
  statement
  statement
  statement
  statement
if expression1 and expression2:
    statement
    statement
    statement
else:
    statement
    statement
    statement
if expression1 and expression2:
    statement
    statement
    statement
else:
    statement
    statement
    statement

True  True
True  False
False  True
False  False
if expression1 or expression2:
    statement
statement
statement
else:
    statement
statement
statement
if not expression:
    statement
    statement
    statement
else:
    statement
    statement
    statement
Bits, binary switch
Relational Operators
Boolean Operators

Teller Machine Revisited
Exercises
Coding Exercise

Recode the Teller-Machine program, with And, Or, and Not

• the output correctly displays "bill" or "bills"
• a number of bills of 0 is not displayed
• only an amount less than $400 is allowed
• only amounts multiples of $5 are allowed.
else is not always used...

```python
if no20s == 1:
    print( no20s, "$20-bill" )
else:
    print( no20s, "$20-bills" )
```
else is not always used...

caption = "$20-bill"
if no20s != 1:
    caption = caption + "s"

print( no20s, caption )
Computer Graphics is Covered in Chapter 4
Graphics Library

- Can be found here: http://cs.smith.edu/dftwiki/index.php/Zelle%27s_Graphics.py_for_Python_3

- And in the Links and Resources section of the class Web page
# graphics.py

"""Simple object oriented graphics library

The library is designed to make it very easy for novice programmers to
experiment with computer graphics in an object oriented fashion. It is
written by John Zelle for use with the book "Python Programming: An
Introduction to Computer Science" (Franklin, Beedle & Associates).

LICENSE: This is open-source software released under the terms of the
GPL (http://www.gnu.org/licenses/gpl.html).

PLATFORMS: The package is a wrapper around Tkinter and should run on
any platform where Tkinter is available.

INSTALLATION: Put this file somewhere where Python can see it.

OVERVIEW: There are two kinds of objects in the library. The GraphWin
class implements a window where drawing can be done and various
GraphicsObjects are provided that can be drawn into a GraphWin. As a
simple example, here is a complete program to draw a circle of radius
10 centered in a 100x100 window:

```python
from graphics import *

def main():
    win = GraphWin("My Circle", 100, 100)
    c = Circle(Point(50,50), 10)
    c.draw(win)
    win.getMouse() # Pause to view result
    win.close()    # Close window when done

main()
```

GraphWin objects support coordinate transformation through
setCoords method and pointer-based input through getMouse method.

The library provides the following graphical objects:

- Point
- Line
Doing Graphics:

1. **Open** a graphic window

2. **Draw** on it

3. **Close** it

4. Terminate the program
from graphics import *

def main():
    win = GraphWin("CSC111", 600, 400)
    c = Circle(Point(50,50), 10)
    c.draw(win)

    win.getMouse()  # Pause to view result
    win.close()     # Close window when done

main()
from graphics import *

def main():
    win = GraphWin("CSC111", 600, 400)
    c = Circle(Point(50,50), 10)
    c.draw(win)

    win.getMouse()  # Pause to view result
    win.close()  # Close window when done

main()
Objects to Play With:

- Points
- Circles
- Rectangles
- Labels (text)
Points

• Used to anchor other objects (circles or rectangles)

• Defined by \(x\) and \(y\) coordinates

```python
# create a point at location (50, 50)
p1 = Point(50,50)

# create a point at location (50, 50)
p2 = Point(150,50)
```
Circles

• Defined by a **center** and a **radius**

• The center is a **Point**

```python
# create a circle centered at (50, 50)
# with radius 70
cl = Circle( Point(50,50), 70 )
cl.draw( win )
```
Rectangles

- Defined by a **top-left**, and a **bottom-right point**

```python
# create a rectangle with top-left corner
# at (5,5) and bottom-right at (50,50)

r3 = Rectangle( Point(5,5), Point( 50, 50) )
r3.draw( win )
```
Labels

• Defined by an anchor **Point**, and

• A **string** that is displayed, centered on the anchor point.

```python
# Create a text label centered at (100,100) # and containing "Smith College"
label3 = Text( Point(100,100), "Smith College" )
label3.draw( win )
```
# create a rectangle with top-left corner
# at (5,5) and bottom-right at (50,50)

r3 = Rectangle( Point(5,5), Point( 50, 50) )
r3.setFill( "red" )
r3.draw( win )
Filling an Object with Color

```python
# create a rectangle with top-left corner at (5,5) and bottom-right at (50,50)
r3 = Rectangle( Point(5,5), Point( 50, 50) )
color = color_rgb( 200, 100, 150 )
r3.setFill( color )
r3.draw( win )
```
Lot's of Colors to Choose from

http://cs.smith.edu/dftwiki/index.php/Tk_Color_Names
Demo Time
from graphics import *

def main():
    win = GraphWin( "Lab 7 Moving ball", 600, 400 )

    # create and draw a red circle
    center = Point( 100, 100 )
    circ = Circle( center, 30 )
    circ.setFill( 'red' )
    circ.draw( win )

    # set initial direction of ball
    dx = 1.5
    dy = 0.25

    # move ball on screen
    while win.checkMouse() == None:
        circ.move( dx, dy )
        x = circ.getCenter().getX()
        y = circ.getCenter().getY()

    win.close()  # Close window when done

main()
Understanding Motion

Center at (3,2)
Understanding Motion

Center at (3,2)

deltaX = 2.5

deltaY = 1
Understanding Motion

Center at (3,2)

circ

deltaX = 2.5

deltaY = 1

circ.move( deltaX, deltaY )
Understanding Motion

deltaX = 2.5

deltaY = 1

Center at (5.5, 3)

circ

circ.move( deltaX, deltaY )
Understanding Motion

circ.move( deltaX, deltaY )
circ.move( deltaX, deltaY )

Delta X = 2.5
Delta Y = 1

Center at (5.5, 3)
Understanding Motion

deltaX = 2.5
deltaY = 1

circ.move( deltaX, deltaY )
circ.move( deltaX, deltaY )

Center at (8, 4)
Concepts of Graphics & Animation
Every element is an OBJECT

Examples

Organization of a graphic program

Something completely different...
Every Graphic Element is an Object
p = Point(50, 150)

Point

getX()
p = Point(50, 150)

p.getX()
Circle

Circle Object

center
radius
Circle

Circle Object

getCenter()
Circle

Circle Object

getCenter()

center
radius

Point Object

getX()  x, y
Circle

Circle Object

getCenter()

center
radius

Point Object

getX()

50
number

x, y
Circle

Circle Object

getCenter()

center
radius

Point Object

getX()

x, y

50 number
Rectangle

Rectangle Object

getP1()

getP2()

Point Object

x, y
Rectangle

Rectangle Object

getP1()

getP2()

Point Object

getX()

x, y
Rectangle

Rectangle Object

getP1()
ggetP2()

P1

P2

getX()

x, y

Point Object

50
number
Text (label)

Text Object

getText()

getAnchor()
Text (label)

Text Object

- getAnchor()
- getText()
Text (label)

Text Object

anchor

text

getAnchor()

getText()

Point Object

getX()

x, y
Text (label)

Text Object

anchor

text

getAnchor()

getText()

Point Object

getX()

x, y
Text (label)

Text Object
- getAnchor()
- getText()

Point Object
- getX()
- y

245 number
Where is all this Defined?

Zelle's Graphics.py for Python 3

--D. Thiebaut (talk) 11:12, 8 March 2015 (EDT)

The file below, copyrighted by John Zelle, was downloaded from http://mcsp.wartburg.edu/zelle/python/graphics.py on 3/8/15, and mirrored here for convenience.

```
# graphics.py

"""Simple object oriented graphics library

The library is designed to make it very easy for novice programmers to experiment with computer graphics in an object oriented fashion. It is written by John Zelle for use with the book "Python Programming: An Introduction to Computer Science" (Franklin, Beedle & Associates).

LICENSE: This is open-source software released under the terms of the GPL (http://www.gnu.org/licenses/gpl.html).

PLATFORMS: The package is a wrapper around Tkinter and should run on any platform where Tkinter is available.

INSTALLATION: Put this file somewhere where Python can see it.

OVERVIEW: There are two kinds of objects in the library. The GraphWin class implements a window where drawing can be done and various GraphicsObjects are provided that can be drawn into a GraphWin. As a simple example, here is a complete program to draw a circle of radius 10 centered in a 100x100 window:
```

http://cs.smith.edu/dftwiki/index.php/Zelle%27s_Graphics.py_for_Python_3
Every element is an OBJECT

Examples

Organization of a graphic program

Something completely different...
Examples
Point

\[ p = \text{Point}(50, 150) \]

\[ x = \text{point1}.getX() \]
\[ y = \text{point1}.getY() \]

if \( x \leq 0 \) or \( y \leq 0 \):
    # the point is outside the window
    ...

\[ \text{point1} = \text{Point}(50, 150) \]
```python
point1 = Point( 50, 150 )
circ1 = Circle( point1, 30 )

center1 = circ1.getCenter()
x = center1.getX()
y = center1.getY()
if x <= 0 or y <= 0:
    # the center is outside the window
    ...
```
point1 = Point( 50, 150 )
circ1 = Circle( point1, 30 )

center1 = circ1.getCenter()
x = center1.getX()
y = center1.getY()
if x <= 0 or y <= 0:
    # the center is outside the window
    ...

x = circ1.getCenter().getX()
y = circ1.getCenter().getY()
if x <= 0 or y <= 0:
    #
D. Thiebaut, Computer Science, Smith College

Rectangle

Rectangle Object

getP1()

g getP2()

getP2()

Rectangle Object

Point Object

x, y

50 number

r = Rectangle( Point( 50, 150 ),
               Point( 150, 150 ) )

r.move( dx, dy )

x1 = r.getP1().getX()
y1 = r.getP1().getY()
x2 = r.getP2().getX()
y2 = r.getP2().getY()

mouseP = win.checkMouse()
if mouseP != None:
    x = mouseP.getX()
y = mouseP.getY()
    if x1 <= x <= x2 and ...
Every element is an OBJECT

Examples

Organization of a graphic program

Something completely different...
Organization of a Graphic Program
# open the graphic window

# define and initialize the graphic objects

# start animation loop. Stop on specific user interaction

    # move/update each object according to its speed
    # and direction

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects

# start animation loop. Stop on specific user interaction

    # move/update each object according to its speed
    # and direction

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction

# move/update each object according to its speed
# and direction

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:
    # move/update each object according to its speed
    # and direction

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop.  Stop on specific user interaction
while win.checkMouse() == None:

    # move/update each object according to its speed
    # and direction
    circ.move( dx, dy )

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:

    # move/update each object according to its speed
    # and direction
    circ.move( dx, dy )

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:
    # move/update each object according to its speed
    # and direction
    circ.move( dx, dy )

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:
    # move/update each object according to its speed
    # and direction
    circ.move( dx, dy )

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:
    # move/update each object according to its speed
    # and direction
    circ.move( dx, dy )

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:

    # move/update each object according to its speed
    # and direction
    circ.move( dx, dy )

# Loop is over.
# close the graphic window
win.close()
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:
    # move/update each object according to its speed
    # and direction
    circ.move( dx, dy )

# Loop is over.
# close the graphic window
win.close()
Measuring Distances
\[d^2 = a^2 + b^2\]
\[ d = \sqrt{ (y_2-y_1)^2 + (x_2-x_1)^2} \]
from math import *

def distance(x1, y1, x2, y2):
    return sqrt((x1-x2)*(x1-x2) + (y1-y2)*(y1-y2))
# distanceDemo.py
# D. Thiebaut

from math import *

def distance(x1, y1, x2, y2):
    return sqrt((x1-x2)*(x1-x2) + (y1-y2)*(y1-y2))

def main():
    x1 = 10
    y1 = 10
    x2 = 50
    y2 = 40
    d = distance(x1, y1, x2, y2)
    print(d)

main()
Every element is an OBJECT

Examples

Organization of a graphic program

Something fun...

http://cs.smith.edu/dftwiki/index.php/Turing_Test_and_Eliza