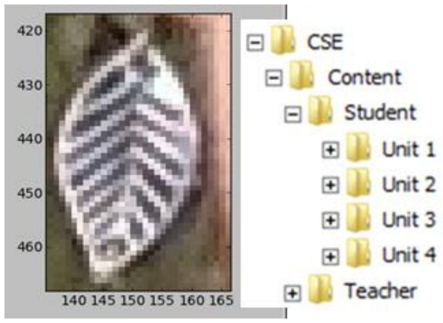
Activity 1.4.2 Objects and Methods

Introduction

In this lesson you will be creative with images. Starting with image files, you will load image objects into memory and then do interesting things with them. What are image files? They're data, really just zeros and ones. Images can be stored and retrieved like any other data files.



Once an image is in memory, we can perform a "simple" action on it like enlarging, brightening, or rotating it. These verbs abstract a complicated operation that can involve millions of calculations. When you move a window on the screen, click on a menu, or even just move the mouse, the pixels on the screen change. These are manipulations of images. In each case the central processing unit and the processors on the graphics card handle millions of ones and zeros to render fresh images on the monitor. How do we use objects and methods to handle these complex operations?

Materials

- Computer with Enthought Canopy distribution of *Python*® programming language
- Webcam or other way to obtain a digital picture
- Practice file `woman.jpg` and `cat1-a.gif`

Procedure

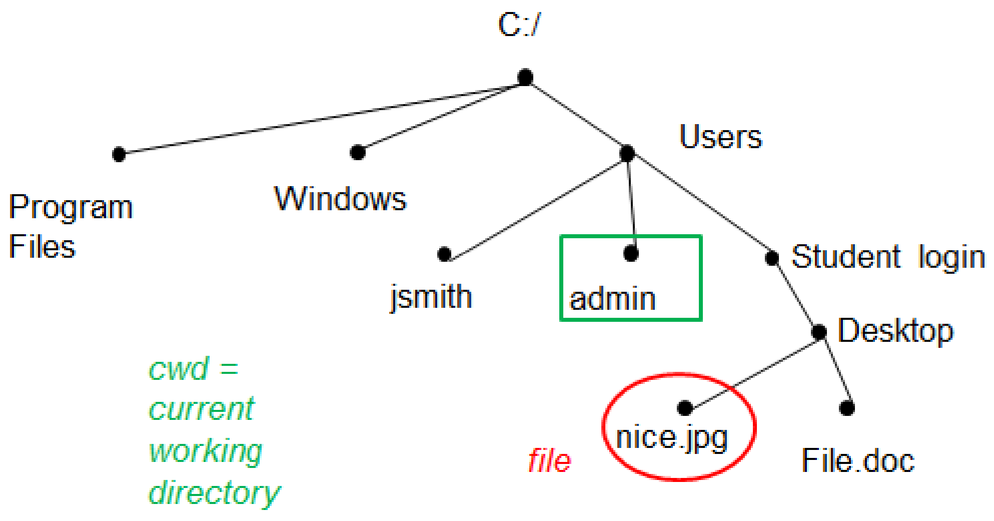
1. Form pairs as directed by your teacher. Meet or greet each other to practice professional skills. Set team norms.

Part I: Working with a Filesystem

2. To open an image in a program, you will need a way to find the file using the programming language. You can use a file's **absolute filename**. Most operating systems and programming languages remember one location in the file system as your "working directory," and a file can be described relative to that location: a **relative filename**. First we deal with absolute filenames.
Most file systems are hierarchical, forming a **tree** that begins with a **root** directory. An absolute filename specifies where the file is stored from the root,

which is typically indicated by `/` in UNIX and Max Operating Systems and by the startup drive letter in Windows, such as `C:\`.

Files and directories are **nodes**, each branching from one **parent** in the tree, with the root considered the “top” of the tree. The absolute filename of `admin` (in the green box below) is `C:/Users/admin`. What is the absolute filename of `nice.jpg` (in the red circle below)?



3. A filename can be specified with a **relative filename**. A relative name means that the file location is described starting from the current working directory. It does not begin with the root `/` or `C:\`. The special symbol of two periods `..` is used by many languages to represent "up one level in the tree."

   If we were currently working in the `admin` directory, what would be the relative filename for `nice.jpg`?

   If we were currently working in the `Windows` directory, what would be the relative filename for `File.doc`?

Part II: Rendering an Image on Screen

4. Launch Canopy. Open an editor window. Set the working directory to your folder. Create a new *Python* file. Save the file as `JDoe_JSmith_1_4_2.py`.

5. Lines 12 and 14 of the code below create an absolute filename for an image by assuming that the image is in the same folder as your *Python* script. Use Windows Explorer to place files `woman.jpg` and `'cat-1a.gif'` in the folder where you are saving *Python* scripts.

Earlier we used the code editor to define functions that we still executed in the IPython session. Coding in the code editor will execute directly with the “play” button” if it is not in a function definition. Execute the following code.

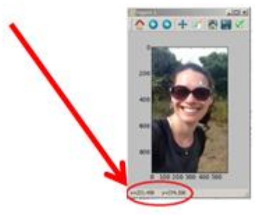
```
'''
JDoe_JSmith_1_4_2: Read and show an image.
'''
import matplotlib.pyplot as plt
import os.path
import numpy as np      # “as” lets us use standard abbreviations

'''Read the image data'''
# Get the directory of this python script
directory = os.path.dirname(os.path.abspath(__file__))
# Build an absolute filename from directory + filename
filename = os.path.join(directory, 'woman.jpg')
# Read the image data into an array
img = plt.imread(filename)

'''Show the image data'''
# Create figure with 1 subplot
fig, ax = plt.subplots(1, 1)
# Show the image data in a subplot
ax.imshow(img, interpolation='none')
# Show the figure on the screen
fig.show()
```

You should see a new window displaying the image of a woman, perhaps hidden behind other windows. You can use **Alt-tab** to cycle through windows.

The figures created by `matplotlib` are interactive **graphical user interfaces (GUIs)**. The GUI shows the coordinates of the mouse pointer, as shown at right. These coordinates are the image coordinates, more or less. (That's not quite true, since the coordinates shown by the GUI can be between integers, unlike the image coordinates. Also, we could have placed the image with its upper left corner somewhere other than (0, 0).)



a. What is the (x, y) coordinate pair of the woman’s nose in the image coordinate system?

b. Change the code so that it shows the cat. What are the image coordinates at the tip of the cat's nose?

Part III: Objects and Methods

6. Consider lines 16-22 of the code from the previous step.

    a. A **class** is a category of **objects** that have **properties** (a set of variables with potentially unique values for each object) and **methods** (a common set of scripts that do things). An object is an **instance** of its class.

    In line 18, `plt.subplots(1, 1)` creates a 1 x 1 grid of subplots in a figure. It returns a 2-tuple. The first element of the tuple is an object in the class Figure. The second element of the tuple is an object in the class AxesSubplot.

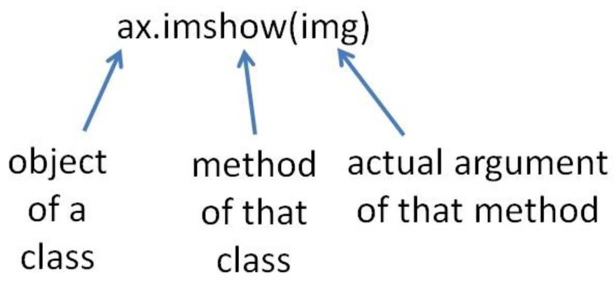
    The particular Figure object is being stored in a new variable, `fig`. The particular AxesSubplot object is being stored in the variable `ax`.

    That was a lot of information! `fig` and `ax` are both objects. What class is each of them in?

    `fig` is an instance of the class ____________________

    `ax` is an instance of the class ____________________

    b. In line 20 a method is being called on the object `ax`.



    The method is `imshow()`. It is being given 1 argument: `img`. The `imshow()` method is being called on the `ax` object. Since `ax` is an instance of the AxesSubplot class, `imshow()` must be a method of the AxesSubplot class.

    c. Comments help us undestand why a method is being called. Which comments explain which lines in the code above?

Part IV: Arrays of Objects

7. Methods often return data. Calling `subplots(1, 1)` returns a tuple of two objects:

`(Figure, AxesSubplot)`

The method subplots can also be used to create a grid of AxesSubplots, as shown below. In this case subplots (1, n) will return

`(Figure, ndarray of AxesSubplots)`

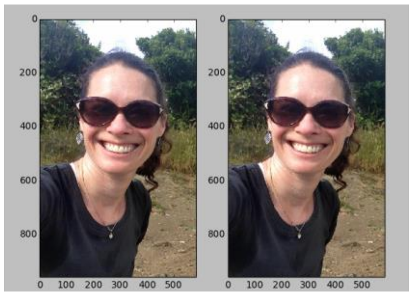
where ndarray is an n-d array, short for an "n-dimensional array." You can access the elements of an ndarray with an index in square brackets:

```
17  # Create a 1x2 grid of subplots
18  # fig is the Figure, and ax is an ndarray of AxesSubplots
19  # ax[0] and ax[1] are the two Axes Subplots
20  fig, ax = plt.subplots(1, 2)
21  # Show the image data in the first subplot
22  ax[0].imshow(img, interpolation='none')
23  # Show the figure on the screen
24  fig.show()
```

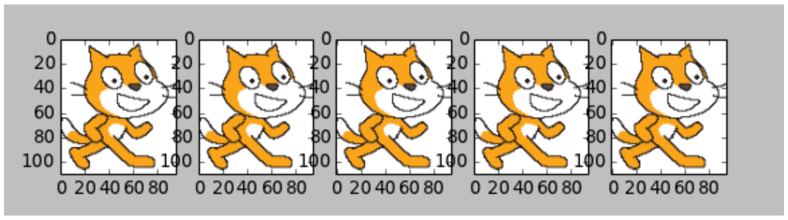
a. Modify lines 18 and 20 of your code to match what is shown above. You will have to re-execute the code to see the effect. Practice using object-oriented syntax by describing line 22: the method _________ is being called on the object __________.

b. Modify the code provided above to create the following figures:

i. An image of the woman in both of the subplots.



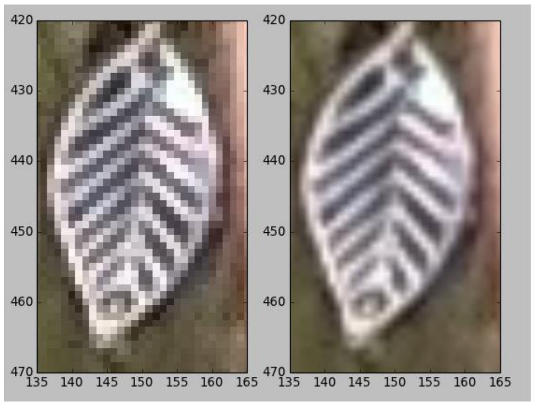
ii. Iterated images of a picture:



Part V: Keyword = Value Pairs

8. Methods and other functions often can be called with additional arguments. If you don't provide an optional argument when you call the function, the default value of that argument is used.

   The `imshow()` method was called on both of the `AxesSuplot`s shown below. For the axes on the left, the method was called with `interpolation='none'`, whereas the axes on the right used the default value of the `interpolation` argument.



```
'''Read the image data'''
# Get the directory of this python script
directory = os.path.dirname(os.path.abspath(__file__))
# Build an absolute filename from directory + filename
filename = os.path.join(directory, 'woman.jpg')
# Read the image data into an array
img = plt.imread(filename)

# Create figure with 2 subplots
fig, ax = plt.subplots(1, 2)
# Show the image data in the first subplot
ax[0].imshow(img)
ax[1].imshow(img, interpolation='none') # Override the default
ax[0].set_xlim(135, 165)
ax[0].set_ylim(470, 420)
ax[1].set_xlim(135, 165)
ax[1].set_ylim(470, 420)
# Show the figure on the screen
fig.show()
```

The matplotlib interface will normally **interpolate** between values of the image pixels, inferring intermediate colors for screen pixels between the centers of image pixels.

The keywords of a function are often important ideas from the library's subject matter. Interpolating is an important idea in math. Describe the connections among interpolation between data points, the `interpolation` argument, the image above, and the code above.

9. An **API (Application Programming Interface)** for a class describes all methods you can call on objects in the class. Here are some of the methods from the API for `AxesSubplot`.

Some methods of `plt.SubplotAxes`	**Description**
`axis('on' | 'off')`	Show/hide axes (and their titles and ticks) *Documentation uses a vertical line | for "or", showing that you can either* `'on'` *or* `'off'`
`set_xlim(xmin,xmax)`	Set lower and upper limits to x-axis
`set_ylim(ymin,ymax)`	Set lower and upper limits to y-axis
`cla()`	Clear axes
`imshow(img)`	Place an image on an axis
`minorticks_on()`	Show minor ticks
`minorticks_off()`	Hide minor ticks
`set_xlabel(string)`	Set x-axis title
`set_ylabel(str)`	Set y-axis title
`set_xticks(list)`	Set major ticks to label
`set_title(string)`	Set subplot title

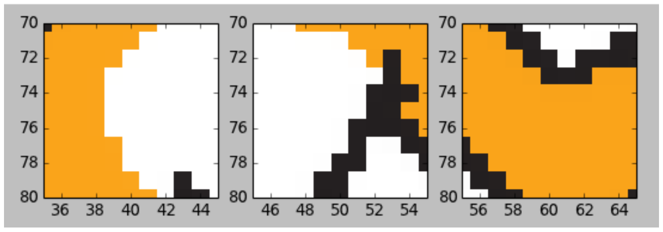
a. Experiment in IPython until you succeed in calling a couple of these methods on an `AxesSubplot`. After an `object.method()` call, you can view the updated figure with its canvas' `draw()` method.

Use `fig.canvas.draw()` after a change in an `AxesSubplot`

```
In []: ax[0].imshow(img)
In []: fig.canvas.draw()
```

Reminder: You can use the up arrow to avoid having to retype the `draw()` command each time.

b. Use the methods above to create three close-ups of an image in a single `Figure`. Each close-up should show a 10 pixel by 10 pixel region. An example using `cat1-a.gif` is shown here.



Share your work for this step as directed by your instructor by showing the teacher your image on screen and by submitting a file of:

- the original image
- a captured image of the Figure as shown above
- your *Python* code

10. The ability to wade through documentation full of unknown terms is an important skill. Go to the documentation at **http://matplotlib.org/api/axes_api.html#matplotlib.axes.Axes.imshow**. This documentation is for the class `Axes`, which includes the `AxesSubplot` subclass. The link provided here points to the documentation for the `imshow()` method. From the documentation, identify one additional method of an `AxesSubplot` . Describe at least one of the optional arguments of that method and state its default value.

11. The class `AxesSubplot` has many methods for displaying data, including the `plot()` method.

    `plot(x, y, 'ro')` places red circles (coded by `'ro'`) at all points ***(x_i, y_i)*** where `x` and `y` are lists of the ***x_i*** and ***y_i*** coordinates, respectively. In an image containing a few faces you find, mark the eyes with red circles using `plot()`.

    Share your work for this step as directed by your instructor by showing the teacher your image on screen and by submitting a file of:

    - the original image
    - a captured image of the Figure
    - your *Python* code