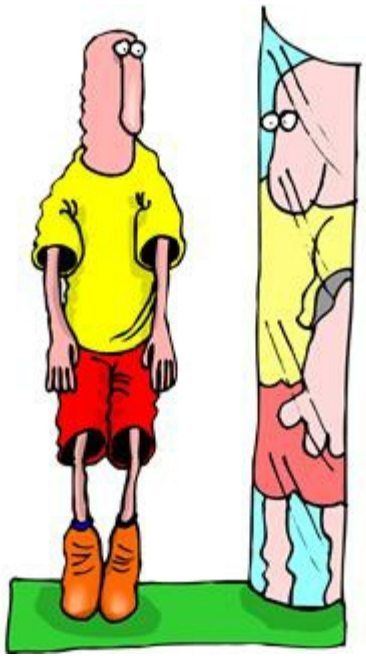
PLTW COMPUTER SCIENCE

Activity 1.4.4

Python Imaging Library API

Introduction

You've learned the basics of writing computer programs. But most programming builds on code that has already been written. Knowing how to find and use code from other people will help make you an efficient and successful software developer.



You could create your own algorithms, for example, to rotate an image or to identify the objects in an image. But others have already solved those problems! There are many advantages to using existing code. You save time, of course. But you also connect to a community of people, making it easier for them to help you, and making it more likely they will be able to use what you create. How will you put other people's code to use?

Materials

- Computer with Enthought Canopy distribution of the *Python*® programming language
- Webcam or other way to capture a digital picture
- Image files and source files in *Python* for Activity 1.4.4

Resources

1.4.4 sourceFiles.zip

Reference Card for Pyplot and PIL

Procedure

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

skills. Set team norms.

2. 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_4.py`.

3. Some of your work in this assignment will involve trial and error using the IPython session. You might want a record of commands you have tried, so begin logging the session.

   ```
   In []: %logstart -ort studentNames_1_4_4.log
   ```

4. In an earlier activity, you learned how to use nested `for` loops to conditionally change color values of individual pixels. You did this using code from the `matplotlib` and `numpy` libraries. You used statements like `img1[r][c] = [255, 0, 255]` to change the pixels in part of an image. To cause an image's sky to turn pink, for example, the statement `img1[r][c] = [255, 0, 255]` was inside an `if` block. The `if` structure was inside a `for` loop, which was inside another `for` loop. Explain the purpose of these three structures (the outer `for`, the inner `for`, and the `if`) in the algorithm used to turn the sky pink.

5. A software developer's job usually includes learning how to use other people's code. It is often much quicker to find and learn how to use code that someone else has created than it is to create the code on your own. We will use code from the `PIL`, the Python Imaging Library, to perform image manipulations.

   Recall that an **API**, or application programming interface, tells you how to use someone else's code. `PIL`, like `matplotlib`, is an **object-oriented** library of code. Object-oriented code creates and manipulates objects, which are data structures defined in the code. PIL's objects include images. The matplotlib library's objects include figures and axes.

   For object-oriented code, the API lists the **attributes** and **methods** for each **class**. Almost always, at least one of the methods instantiates the class. **Instantiation** means creating an object, also known as an instance, of the class. The instantiation methods that do this are sometimes called **constructors**.

   Although you are learning to manipulate images in this activity, a primary objective is to understand how to find, select, and use API documentation. These skills will enable you to learn how to use other people's code.

   We will use nine methods and one attribute from the API for the `PIL.Image` class. Refer to Table 2 in the *Reference Card for Pyplot and PIL*. In the following table, create a short description for the each of the methods that do not have one. The three rows without method names will used in the next step.

Method	Short Description
`open()`	
`new()`	
`crop()`	makes a `PIL.Image` of smaller size
`convert()`	

`resize()`	
`rotate()`	
`paste()`	
`transform()`	makes a `PIL.Image` by distorting an image
`save()`	

Attribute	Short Description
`size`	width, height

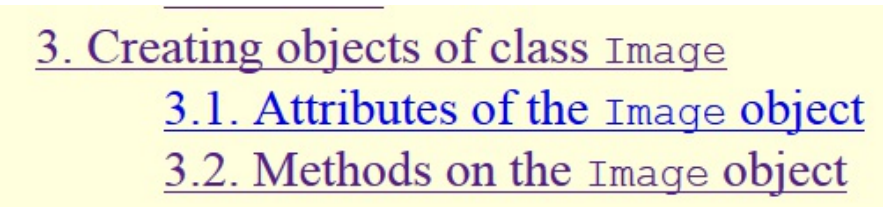
6. There are four kinds of documentation:

    - Official documentation
    - Third-party documentation
    - Tutorials
    - Problem/solution-specific documentation, often in social formats such as Stack Overflow

    The official documentation is usually among the best references for using a library of code. Official documentation is produced by the person or group who created the code. Docstrings fall into this category since they are created by the developers while writing comments in their code.

    Refer to **http://effbot.org/imagingbook**, the official documentation for `PIL`. Identify three methods of objects in the `PIL.Image` class that are not yet listed in the table in Step 5. Add the three methods to the table in Step 5. Record both the method name and a short description of the method from the documentation.

7. Another place to learn about a library is from third-party documentation. New Mexico Tech produced some excellent documentation of `PIL`. Refer to Section 3 of that documentation at **http://infohost.nmt.edu/tcc/help/pubs/pil/**



The attribute `size` is listed in the table above in Step 5. What is another attribute of every `PIL.Image` object?

8. You may be able to find tutorials created specifically to teach, as opposed to reference materials created for later reference. Tutorials might be part of official documentation or developed by third parties. Visit **http://effbot.org/imagingbook/introduction.htm**.

   How is this different than the reference material linked in Step 6?

9. For this activity you have been provided with a *Reference Card for Pyplot and PIL* and three of the best references available online for PIL. Usually when developers find a module of code to use, they still have to identify useful documentation. Find one additional PIL documentation resource with an Internet search. Good search terms are "PIL," "Python," and the name of a method for which you want more information. Another technique is to include an error message in your search terms. Record the URL of a site you select from your search results. Summarize the author's purpose in terms of the categories of documentation listed in Step 6.

   URL:

   Summary of author's intent:

10. As you complete the rest of this activity, continue to use the various sources of documentation linked above and others as appropriate. Which do you think will be the most useful to you and why?

11. Figure 1 in the *Reference Card for Pyplot and PIL* has reference information about three *Python* code libraries. Based on your teacher's description, summarize the purpose of these three libraries:

    - `matplotlib.pyplot` (plt) –
    - `numpy` (np) –
    - `PIL` –

12. Obtain the code and images provided for Activity 1.4.4 in `1.4.4 SourceFiles.zip` as directed by your teacher. Open and execute `earthEyes.py`. This program should display three figures as shown below. The program pastes an image of the Earth over one eye in an image of a student.

    The figures below are each labeled with three variables from `earthEyes.py`. The first variable in each case is a matplotlib `Figure`. The second and third variables are the `AxesSubplot` objects.

To bring the figures to the front of the Windows environment, click on pyplot icons in the Windows taskbar.

Examine all three figures on your screen. The three figures show the identification of the eye's coordinates, the resizing of the Earth's image, and the pasting of the Earth on top of the eye.

13. The code creating the figure above on the left is shown below.

```
# Open and show the student image in a new Figure window
student_img = PIL.Image.open(student_file)
fig, ax = plt.subplots(1, 2)
ax[0].imshow(student_img, interpolation='none')

# Display student in second axes and set window to the right eye
ax[1].imshow(student_img, interpolation='none')
ax[1].set_xticks(range(1050, 1410, 100))
ax[1].set_xlim(1050, 1400) # Measure in plt, experiment in IPython
ax[1].set_ylim(1100, 850)
fig.show()
```

- Line 15 calls the function `open()` from the `PIL.Image` library. The function is being called with one argument: `student_file`. The function returns one object, which is being assigned to `student_img`.

  Practice using this vocabulary by describing line 16: Line 16 calls the function _____ from the ____ library. The function is being called with ___ argument(s): _____. The function returns __ object(s), which is/are being assigned to _____.

- In line 17 the `imshow()` method is called on the object `ax[0]`. Recite to your partner the methods called on each object in lines 20-24:

  Line 17 calls `_imshow()_` on `_ax[0]__`

  Line 20 calls _____ on _____

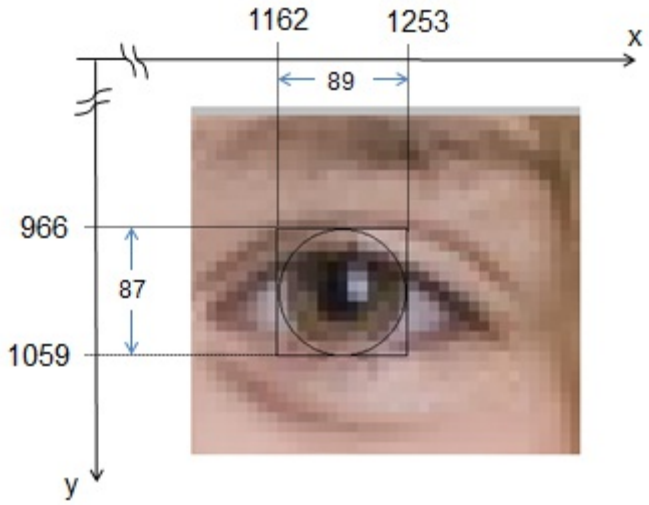
  Line 21 calls _____ on _____

Line 22 calls _____ _____ on _____ _____

Line 23 calls _____ on _____

Line 24 calls _____ on _____

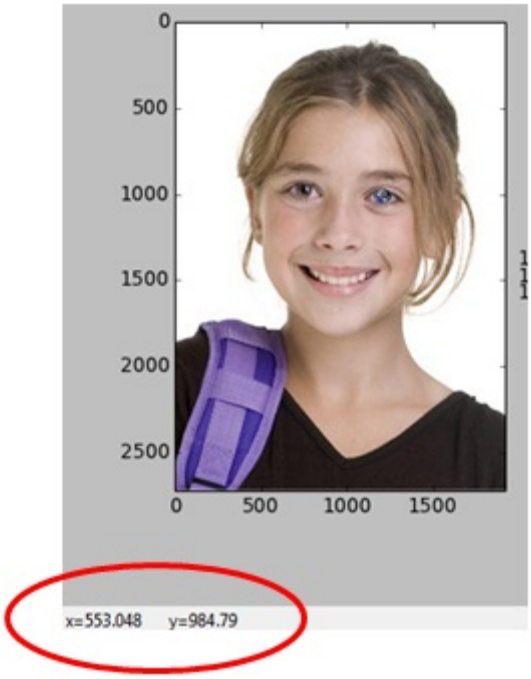
- Lines 21-23 change the axes displaying the single right eye in the figure above. This figure was used to identify the upper left coordinates (1162, 966) of a **bounding box** containing the eye. A bounding box is a rectangle containing a particular part of an image. The bounding box for the right eye's **iris** (the colored part of the eye) is illustrated below.



What are the (x, y) coordinates of the upper left corner of the bounding box?

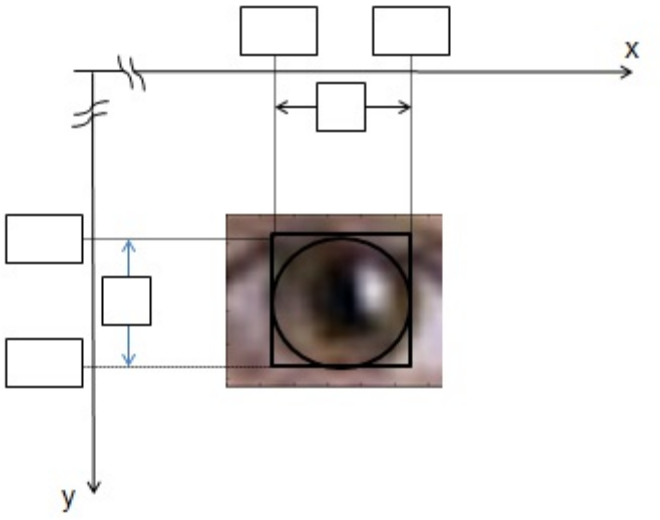
14. In the next several steps, you will use *Python* to paste an image of the Earth over the other eye in the image of the student. We will call this the left eye of the image, though it is the student's right eye. First, identify the coordinates of the iris of the image's left eye.

    - Mouse over the eye and take note of the coordinates.



- Zoom in on the image's left eye by using the figure window's pan/zoom mode.
    - The pan/zoom mode can be toggled on and off using the pan/zoom button.

    - When pan/zoom mode is on, drag with the left mouse button to pan. Drag with the right mouse button to zoom.
    - In the following figure, record the coordinates of the upper left and lower right for the bounding box of the iris for the image's left eye. Also record the width and height of the bounding box.



15. The following part of the source code creates the figure that shows two Earths. Read and analyze lines 26 - 33 of the code.

```
# Open, resize, and display earth
earth_file = os.path.join(directory, 'earth.png')
earth_img = PIL.Image.open(earth_file)
earth_small = earth_img.resize((89, 87)) # w and h measured in plt
fig2, axes2 = plt.subplots(1, 2)
axes2[0].imshow(earth_img)
axes2[1].imshow(earth_small)
fig2.show()
```

- Line 27 uses the `join()` function from the `os.path` module. It is being passed ___ arguments. The value it returns is being assigned to the variable _____.
- In line 28 the `open()` function of the `PIL.Image` module returns a new `PIL.Image` object, which is being assigned to the variable _____.
- In line 29 the `resize()` method takes only one argument: a 2-tuple. Explain why there are two sets of parentheses in this line.
- Refer to the bounding box shown in Step 13c. Explain the purpose of the (89, 87) argument in line 29.
- Practice describing code with your partner by reciting the appropriate one of the following two sentences for each line of code in lines 30-33:

  Use one of the following templates (function or method) for each description:

  - Line _____ calls the **function** _____ from the _____

library with ____ argument(s): _____. The function returns _____ object(s), which is/are being assigned to _____ _____.

- Line ___ calls the **method** _____ on the object _____ with ___ argument(s): _____.

Line 30 calls the (method or function) ...

Line 31 calls the (method or function) ...

Line 32 calls the (method or function) ...

Line 33 calls the (method or function) ...

- Reading documentation can be difficult because the words are unfamiliar, even to the expert. Also, the information you need might be found in more than one place. Be persistent! As an example to show you how to piece together information from different places, refer to both of the following sources of PIL documentation to learn how to reduce the size of an image with the `resize()` method of PIL.Image objects.

  - **http://effbot.org/imagingbook**
    Lundh, Fredrik. (2008). *Python Imaging Library.*
  - **http://infohost.nmt.edu/tcc/help/pubs/pil/**
    Shipman, John. (2013). *Python Imaging Library.*

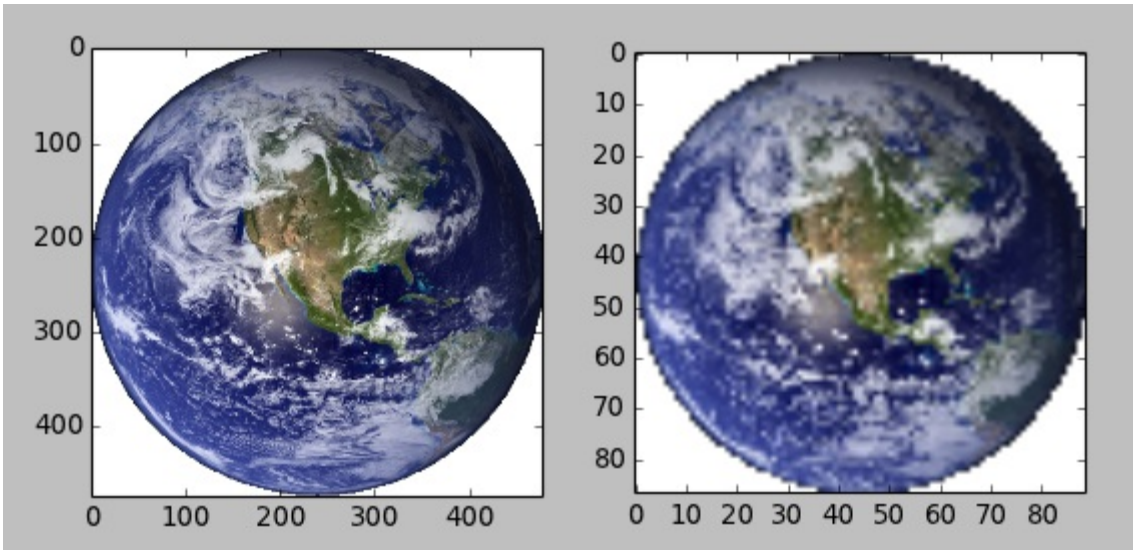
    - What is an additional argument that can be passed to the `resize()` method?
    - What is the default value of that argument?
    - Our code is downsampling the image from NASA. Downsampling means using fewer bytes to represent the data. What value of the optional argument is recommended for downsampling?

- Refer again to the two sources of PIL documentation listed above. What is represented by the size attribute of an Image object?
- Try the following in the IPython shell and explain the output.

```
In []: earth_img.size
```

```
In []: earth_small.size
In []: earth_img.size[1]
```

- Examine the two images in the figure produced by the code. The two images are displayed on the screen using the same number of screen pixels. How can you tell that the two images contain a different number of image pixels?



16. Explain the algorithm you think `resize()` might be using. You can explain it using English sentences, pseudocode, commented code, or any other appropriate method. However, your explanation should address details related to the three or four bytes accounting for each pixel in the Image object returned by the `resize()` method.

17. The following portion of the source code creates the rightmost figure above, `fig3`, showing the girl and a close-up of her pair of eyes. Read and analyze lines 35 - 44 of the code.

```
# Paste earth into right eye and display
# Uses alpha from mask
student_img.paste(earth_small, (1162, 966), mask=earth_small)
# Display
fig3, axes3 = plt.subplots(1, 2)
axes3[0].imshow(student_img, interpolation='none')
axes3[1].imshow(student_img, interpolation='none')
axes3[1].set_xlim(500, 1500)
axes3[1].set_ylim(1130, 850)
fig3.show()
```

- In line 37 the command refers to two `PIL.Image` objects:

  `student_img` is the object on which `paste()` is being called.

  earth_small is passed as the first argument to the `paste()` method.

  These two `PIL.Image` objects have different “mode” attributes.

`student_img` is RGB, while

`earth_small` is RGBA.

As a consequence one image uses three bytes per pixel, while the other image uses four bytes per pixel. Use the IPython session as in Step 15h to determine the size of each image.

Calculate the number of bytes used for the representation of each image, using the following formula. Ignore compression and metadata.

Image bytes = width • height • bytes/pixel

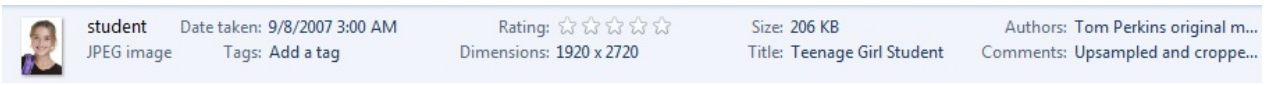
`student_img` bytes =

`earth_small` bytes =

- The RGBA file type includes a fourth byte called the alpha channel. The alpha channel controls how transparent or opaque an image is. Only some file types can store an alpha channel. PNG files can store the alpha channel. JPG files cannot. The `save()` method of `PIL.Image` recognizes filetype extensions in the filename you provide, so transparency is retained if you name the file with PNG. Save `earth_small.png` with the following code in the IPython session.

```
In []: earth_small.save('smallEarth.png')
```

- Examine the file size of these two files in operating system and record the file sizes. Note the other metadata at the bottom of the Windows Explorer window.



`student.jpg` bytes =

`smallEarth.png` bytes =

- Explain the discrepancy between your answers in step a and your answers in step c.

- Refer to the two sources of `PIL` documentation listed previously to learn about the `paste()` method of `PIL.Image` objects. These sources of documentation indicate that the first argument to `paste()` can be either a color or an image. Here we are using an image. According to the documentation, what happens if a color is used for the first argument?

- The documentation also describes what happens if the modes of two images in step 17b are different from each other. According to the documentation, what happens?

- Explain the purpose of each argument being passed to `paste()` in line 32.

18. Add lines of code in the code editor to paste the Earth image onto the girl's other iris. Save your code as directed by your teacher.

19. Save your Python file in the code editor. If you were logging the IPython session, save it with `%logstop`.

    ```
    In []: %logstop
    ```

Conclusion

1. Describe the classes, methods and attributes used in code in this lesson.

2. Consider the statement

   Abstraction is a strategy for handling complexity.

   By giving a name to a complex task and encapsulating the details inside that method. Describe a complex computational task you performed in this activity. Describe the abstraction you used from the PIL library that helped you perform this task.

Lesson 1.4 Reference Card for Pyplot and PIL

Method	Description
`axis('on' | 'off')`	Show/hide axes (and their titles and ticks) *In documentation, the vertical line | usually means "or". Here it shows that the argument can be 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(`*`string`*`)`	Set y-axis title
`set_xticks(`*`list`*`)`	Set which tick marks to label
`set_title(`*`string`*`)`	Set subplot title

Also useful: `fig, ax = plt.subplots(`*`rows, columns`*`) fig.canvas.draw()`

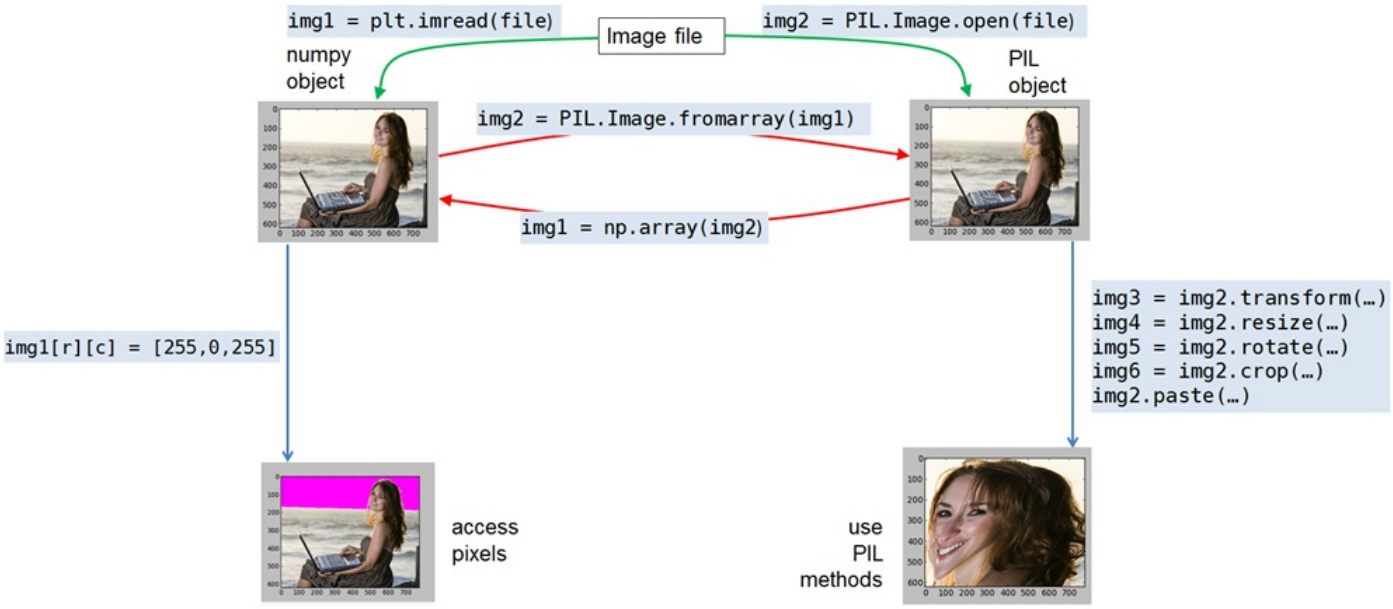


Figure 1. Methods for converting between PIL and numpy objects

Table 2. Methods of the PIL.Image class

Method	Example and Comments
`open()`	`img = PIL.Image.open(`*`filename`*`)` *Creates a new PIL.Image object from a standard image file.*
`new()`	`new_img = PIL.Image.new(`*`mode, size, color`*`)` *Creates a new PIL.Image object of a solid color. For example, to create a 100 x 200 pixel image with all pixels set to RGB=5,5,5 with transparency (alpha=0):* `PIL.Image.new('RGBA', (100, 200), (5, 5, 5, 0) )`
`crop()`	`new_img = img2.crop( (`*`x0, y0, x1, y1`*`) )` *Creates a new image cropped to the coordinates within img2.*
`convert()`	`new_img = img2.convert('RGBA')` *Creates a new image like img2 but with the format specified – a good way to add an alpha channel.*
`resize()`	`new_img = img2.resize( (`*`new_width, new_height`*`) )` *Creates a new image interpolating from img2.*
`rotate()`	`new_img = img2.rotate(angle, expand=False)` *Creates a new image rotated by angle degrees. Use expand=True to avoid cropping.*
	`img2.paste(`*`other_img,(x,y), mask=none`*`)`

`paste()`	*Changes img2 without creating a new image, pasting the top left of the other_img at the given coordinates in img2. Optionally, uses mask's alpha channel.*
`transform()`	`new_img = img2.transform( (new_width, new_height), PIL.Image.QUAD,` $(x_0,y_0,x_1,y_1,x_2,y_2,x_3,y_3,x_4,y_4)$) *Creates a new image by distorting a quadrilateral in img2 into a rectangle.*
`save()`	`img2.save(filename)` *Saves image. Filename should be a string with a standard extension like .jpg or .png .*

Table 3. Methods of the PIL.ImageDraw class

This module must be imported separately from PIL: `import PIL.ImageDraw`

Method	Example and Comments
`Draw()`	`draw = PIL.ImageDraw.Draw(img2)` *makes a drawing object*
`ellipse()`	`draw.ellipse( (`*left, top, bottom, right*`), fill=None, outline=None)`
`line()`	`draw.line(`*[(x1, y1), (x2, y2)]*`, fill=None)`
`polygon()`	`draw.polygon( `*[(x1, y1), (x2, y2), (x3, y3),…]*`, fill=None, outline=None)`
`text()` C	`draw.text(`*(x1, y1)*`, `*message*`, fill=None, font=None)`