

# OBJECT METHODS (ATTRIBUTES)

Methods (or attributes) are *function calls* of Python objects. Some are already included with special Python objects, like lists and dictionaries, and some need to be imported.

Examples:

```
my_list = [3,4]      my_dict = {'spam': 2,  
my_list.append(5)   'eggs': 3}  
                    keys = my_dict.keys()
```

```
file_object = open('my_file.txt', 'w')  
file_object.write('Scott is terrific\n')
```

# Python Modules

Python has a whole *library of modules* with useful functions for all sorts of tasks (processing strings, opening and retrieving web pages, calling system commands and so forth). These modules are used in the same way as other objects, like the list and file objects. All the modules have a host of methods (functions) that can be called to do specific tasks.

Example: The *string* module (functions for processing strings)

```
import string    #imports all the string functions (methods)

sequence = 'gattagcgag'

new_seq = string.upper(sequence) #changes string to uppercase
```

Before using an attribute of a module (or any function you use) there are *three key things* you need to know about the attribute.

- (1) How many arguments does it take?
- (2) What types of data do the arguments have to be?
- (3) What types of values does the function return?

### Examples

```
string.lower(s)  
string.split(s [, sep[,maxsplit]])
```

```
sent = 'Walk the dog.'  
new_sent = string.split(sent)  
new_sent -> ['Walk', 'the', 'dog.']  
new_sent = string.split(sent, 'the')
```

# The re module

The *re (regular expression) module* is another extremely useful python module for text and string processing. Using re, you can easily find, search, or replace strings or portions of strings.

```
sub(pattern, repl, str[,count=0])
split(pattern, str[,maxsplit = 0])
match(pattern, str[,flags])
search(patter, str)

>>>sent2 = "My dog is a hog."
>>>new_sent2=re.sub('dog', 'cat',sent2)
>>>new_sent2=re.sub('hog', 'rat',new_sent2)
```

## Return values and if/else statements

```
sentence = 'My cat is a rat.'  
def find_cat(sent):  
    if re.search('cat', sentence):  
        print 'Cat found!'  
    else:  
        print 'Stupid dog...'  
  
find_cat(sentence) #function call
```

# Regular Expression Syntax

'.' (Dot) This matches any type of character.

**Ex:** `re.search('GA.T', sequence)`

'^' (Caret) Matches only at the start of a string.

**Ex:** `re.search('^My', sentence)` will return true because 'My' is at the beginning of sentence

'\$' Matches at the end of a string

'\D' Matches any non-digit character

'\d' Matches any digit character

'\s' Matches any whitespace character (e.g., \t, \n, \f, \v)

'[]' Matches a set of characters included in the brackets.

**Ex:** `re.search('c[agt9]t', sentence)` will return true because it will find 'cat'