py4kids (https://github.com/wgong/py4kids)

Python Review

In this lesson, we have a pause and review what we have learned so far:

- Cheatsheets
- Code Review and Analysis

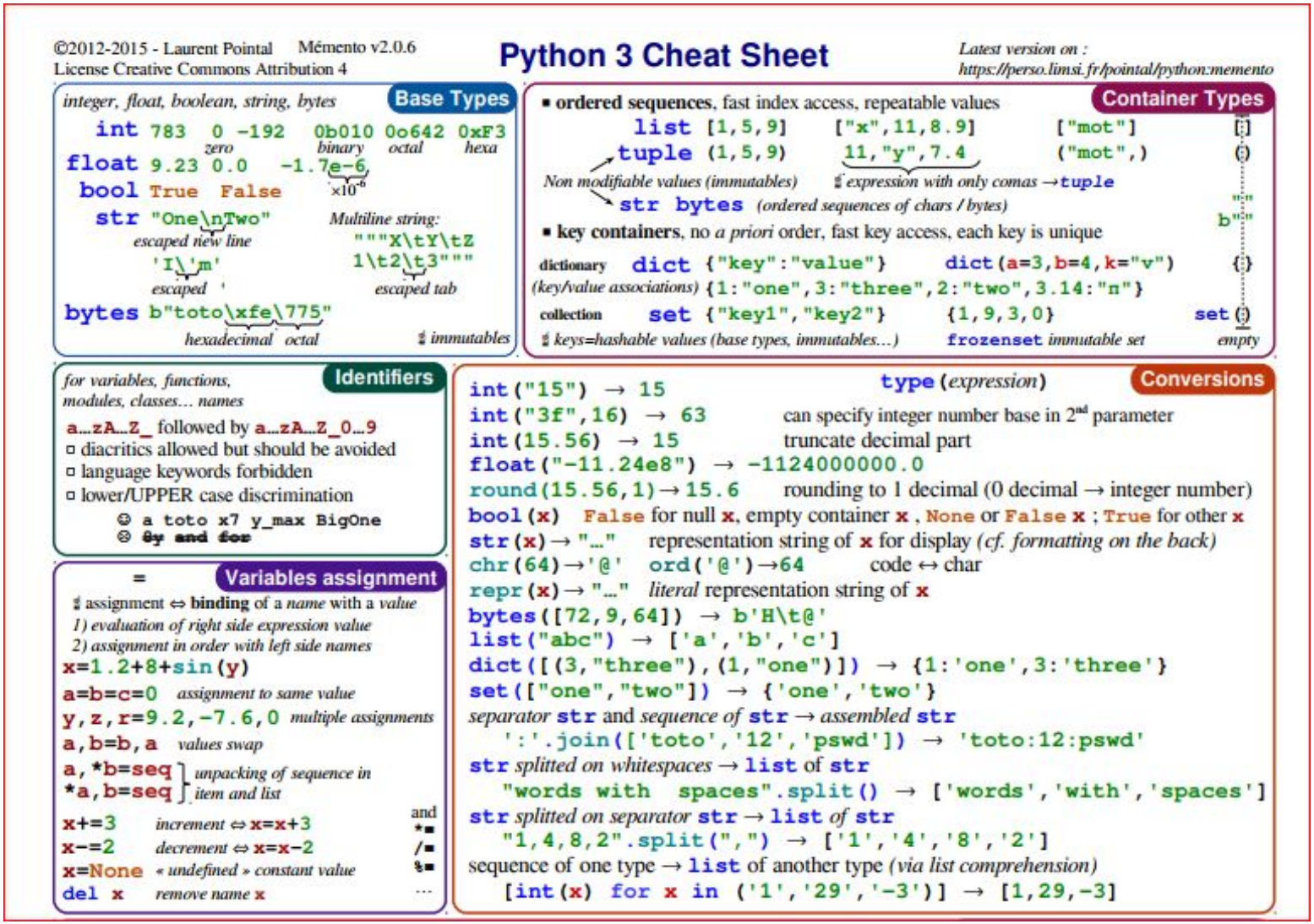
In [1]:
```
from jyquickhelper import add_notebook_menu
add_notebook_menu()
```

Out[1]:

Cheatsheets

Excellent review on various topics:

- Python3 Cheatsheet (https://perso.limsi.fr/pointal/_media/python:cours:mementopython3-english.pdf)



Python 3 Cheat Sheet

Latest version on :
https://perso.limsi.fr/pointal/python:memento

Base Types

integer, float, boolean, string, bytes

```
int 783  0 -192  0b010 0o642 0xF3
          zero   binary octal hexa
float 9.23 0.0  -1.7e-6
                    ×10^-6
bool True  False
str "One\nTwo"        Multiline string:
    escaped new line  """X\tY\tZ
    'I\'m'            1\t2\t3"""
    escaped '         escaped tab
bytes b"toto\xfe\775"
          hexadecimal  octal
```

♯ immutables

Container Types

- **ordered sequences**, fast index access, repeatable values

```
list [1,5,9]   ["x",11,8.9]   ["mot"]   []
tuple (1,5,9)  11,"y",7.4     ("mot",)  ()
```

Non modifiable values (immutables) — *♯ expression with only comas → tuple*

```
str bytes  (ordered sequences of chars / bytes)   ""  b""
```

- **key containers**, no *a priori* order, fast key access, each key is unique

```
dictionary  dict {"key":"value"}   dict(a=3,b=4,k="v")   {}
(key/value associations) {1:"one",3:"three",2:"two",3.14:"π"}
collection  set {"key1","key2"}    {1,9,3,0}            set()
```

♯ keys=hashable values (base types, immutables…) `frozenset` *immutable set* *empty*

Identifiers

for variables, functions, modules, classes… names

`a…zA…Z_` followed by `a…zA…Z_0…9`

- diacritics allowed but should be avoided
- language keywords forbidden
- lower/UPPER case discrimination

```
☺ a toto x7 y_max BigOne
☹ 8y and for
```

Variables assignment

`=`

♯ assignment ⇔ **binding** of a *name* with a *value*

1) evaluation of right side expression value
2) assignment in order with left side names

```
x=1.2+8+sin(y)
a=b=c=0   assignment to same value
y,z,r=9.2,-7.6,0   multiple assignments
a,b=b,a   values swap
a,*b=seq  }
*a,b=seq  }  unpacking of sequence in item and list
```

```
x+=3   increment ⇔ x=x+3      and *= /= %= …
x-=2   decrement ⇔ x=x-2
x=None « undefined » constant value
del x  remove name x
```

Conversions

`type(`*expression*`)`

```
int("15") → 15
int("3f",16) → 63        can specify integer number base in 2nd parameter
int(15.56) → 15          truncate decimal part
float("-11.24e8") → -1124000000.0
round(15.56,1) → 15.6    rounding to 1 decimal (0 decimal → integer number)
bool(x)  False for null x, empty container x , None or False x ; True for other x
str(x) → "…"   representation string of x for display (cf. formatting on the back)
chr(64) → '@'  ord('@') → 64   code ↔ char
repr(x) → "…"  literal representation string of x
bytes([72,9,64]) → b'H\t@'
list("abc") → ['a','b','c']
dict([(3,"three"),(1,"one")]) → {1:'one',3:'three'}
set(["one","two"]) → {'one','two'}
```

separator `str` *and sequence of* `str` → *assembled* `str`

```
':'.join(['toto','12','pswd']) → 'toto:12:pswd'
```

`str` *splitted on whitespaces* → `list` *of* `str`

```
"words with  spaces".split() → ['words','with','spaces']
```

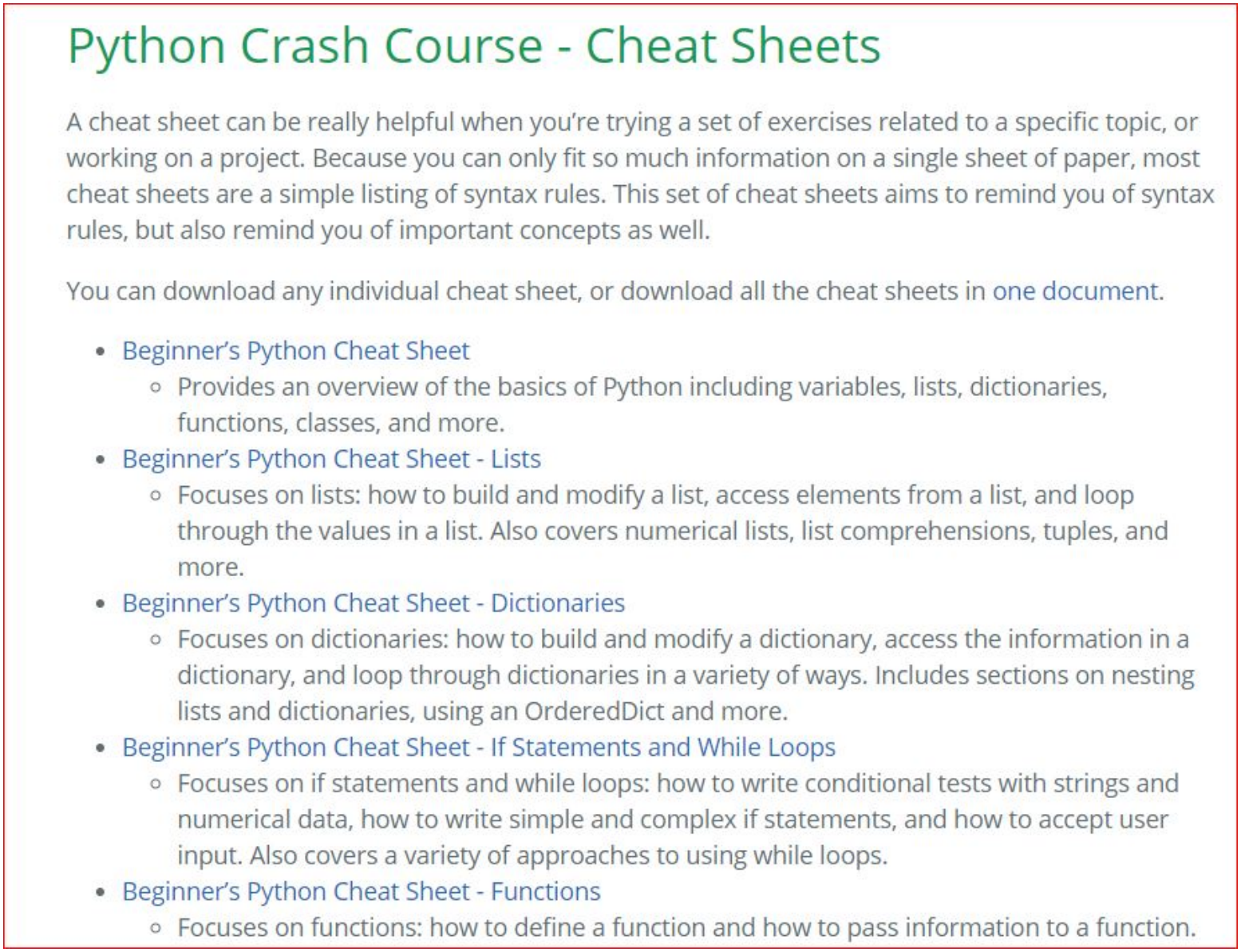
`str` *splitted on separator* `str` → `list` *of* `str`

```
"1,4,8,2".split(",") → ['1','4','8','2']
```

sequence of one type → `list` of another type *(via list comprehension)*

```
[int(x) for x in ('1','29','-3')] → [1,29,-3]
```

- Python Crash Course - Cheat Sheets (http://ehmatthes.github.io/pcc/cheatsheets/README.html)



Python Crash Course - Cheat Sheets

A cheat sheet can be really helpful when you're trying a set of exercises related to a specific topic, or working on a project. Because you can only fit so much information on a single sheet of paper, most cheat sheets are a simple listing of syntax rules. This set of cheat sheets aims to remind you of syntax rules, but also remind you of important concepts as well.

You can download any individual cheat sheet, or download all the cheat sheets in one document.

- Beginner's Python Cheat Sheet
  - Provides an overview of the basics of Python including variables, lists, dictionaries, functions, classes, and more.
- Beginner's Python Cheat Sheet - Lists
  - Focuses on lists: how to build and modify a list, access elements from a list, and loop through the values in a list. Also covers numerical lists, list comprehensions, tuples, and more.
- Beginner's Python Cheat Sheet - Dictionaries
  - Focuses on dictionaries: how to build and modify a dictionary, access the information in a dictionary, and loop through dictionaries in a variety of ways. Includes sections on nesting lists and dictionaries, using an OrderedDict and more.
- Beginner's Python Cheat Sheet - If Statements and While Loops
  - Focuses on if statements and while loops: how to write conditional tests with strings and numerical data, how to write simple and complex if statements, and how to accept user input. Also covers a variety of approaches to using while loops.
- Beginner's Python Cheat Sheet - Functions
  - Focuses on functions: how to define a function and how to pass information to a function.

- Python For Data Science - A Cheat Sheet For Beginners (https://www.datacamp.com/community/tutorials/python-data-science-cheat-sheet-basics)

Code Review and Analysis

Case Studypygame - Chimp

We will walk thru C:\Anaconda3\Lib\site-packages\pygame\examples\chimp.py program line-by-line

In [4]:
```
#!/usr/bin/env python
"""
This simple example is used for the line-by-line tutorial
that comes with pygame. It is based on a 'popular' web banner.
Note there are comments here, but for the full explanation,
follow along in the tutorial.
"""


#Import Modules
import os, pygame
from pygame.locals import *
from pygame.compat import geterror

if not pygame.font: print ('Warning, fonts disabled')
if not pygame.mixer: print ('Warning, sound disabled')

#main_dir = os.path.split(os.path.abspath(__file__))[0]
main_dir = "C:\\Anaconda3\\Lib\\site-packages\\pygame\\examples"
data_dir = os.path.join(main_dir, 'data')

#functions to create our resources
def load_image(name, colorkey=None):
    fullname = os.path.join(data_dir, name)
    try:
        image = pygame.image.load(fullname)
    except pygame.error:
        print ('Cannot load image:', fullname)
        raise SystemExit(str(geterror()))
    image = image.convert()
    if colorkey is not None:
        if colorkey is -1:
            colorkey = image.get_at((0,0))
        image.set_colorkey(colorkey, RLEACCEL)
    return image, image.get_rect()

def load_sound(name):
    class NoneSound:
        def play(self): pass
    if not pygame.mixer or not pygame.mixer.get_init():
        return NoneSound()
    fullname = os.path.join(data_dir, name)
```

```
    try:
        sound = pygame.mixer.Sound(fullname)
    except pygame.error:
        print ('Cannot load sound: %s' % fullname)
        raise SystemExit(str(geterror()))
    return sound


#classes for our game objects
class Fist(pygame.sprite.Sprite):
    """moves a clenched fist on the screen, following the mouse"""
    def __init__(self):
        pygame.sprite.Sprite.__init__(self) #call Sprite initializer
        self.image, self.rect = load_image('fist.bmp', -1)
        self.punching = 0

    def update(self):
        "move the fist based on the mouse position"
        pos = pygame.mouse.get_pos()
        self.rect.midtop = pos
        if self.punching:
            self.rect.move_ip(5, 10)

    def punch(self, target):
        "returns true if the fist collides with the target"
        if not self.punching:
            self.punching = 1
            hitbox = self.rect.inflate(-5, -5)
            return hitbox.colliderect(target.rect)

    def unpunch(self):
        "called to pull the fist back"
        self.punching = 0


class Chimp(pygame.sprite.Sprite):
    """moves a monkey critter across the screen. it can spin the
       monkey when it is punched."""
    def __init__(self):
        pygame.sprite.Sprite.__init__(self) #call Sprite intializer
        self.image, self.rect = load_image('chimp.bmp', -1)
        screen = pygame.display.get_surface()
        self.area = screen.get_rect()
```

```
        self.rect.topleft = 10, 20
        self.move = 9
        self.dizzy = 0

    def update(self):
        "walk or spin, depending on the monkeys state"
        if self.dizzy:
            self._spin()
        else:
            self._walk()

    def _walk(self):
        "move the monkey across the screen, and turn at the ends"
        newpos = self.rect.move((self.move, 0))
        if self.rect.left < self.area.left or \
            self.rect.right > self.area.right:
            self.move = -self.move
            newpos = self.rect.move((self.move, 0))
            self.image = pygame.transform.flip(self.image, 1, 0)
        self.rect = newpos

    def _spin(self):
        "spin the monkey image"
        center = self.rect.center
        self.dizzy = self.dizzy + 12
        if self.dizzy >= 360:
            self.dizzy = 0
            self.image = self.original
        else:
            rotate = pygame.transform.rotate
            self.image = rotate(self.original, self.dizzy)
        self.rect = self.image.get_rect(center=center)

    def punched(self):
        "this will cause the monkey to start spinning"
        if not self.dizzy:
            self.dizzy = 1
            self.original = self.image


def main():
    """this function is called when the program starts.
       it initializes everything it needs, then runs in
```

```
       a loop until the function returns."""
#Initialize Everything
    pygame.init()
    screen = pygame.display.set_mode((468, 100))
    pygame.display.set_caption('Monkey Fever')
    pygame.mouse.set_visible(0)

#Create The Backgound
    background = pygame.Surface(screen.get_size())
    background = background.convert()
    background.fill((250, 250, 250))

#Put Text On The Background, Centered
    if pygame.font:
        font = pygame.font.Font(None, 36)
        text = font.render("Pummel The Chimp, And Win $$$", 1, (210, 10, 10))
        textpos = text.get_rect(centerx=background.get_width()/2)
        background.blit(text, textpos)

#Display The Background
    screen.blit(background, (0, 0))
    pygame.display.flip()

#Prepare Game Objects
    clock = pygame.time.Clock()
    whiff_sound = load_sound('whiff.wav')
    punch_sound = load_sound('punch.wav')
    chimp = Chimp()
    fist = Fist()
    allsprites = pygame.sprite.RenderPlain((fist, chimp))


#Main Loop
    going = True
    while going:
        clock.tick(60)

        #Handle Input Events
        for event in pygame.event.get():
            if event.type == QUIT:
                going = False
            elif event.type == KEYDOWN and event.key == K_ESCAPE:
                going = False
```

```
            elif event.type == MOUSEBUTTONDOWN:
                if fist.punch(chimp):
                    punch_sound.play() #punch
                    chimp.punched()
                else:
                    whiff_sound.play() #miss
            elif event.type == MOUSEBUTTONUP:
                fist.unpunch()

        allsprites.update()

        #Draw Everything
        screen.blit(background, (0, 0))
        allsprites.draw(screen)
        pygame.display.flip()

    pygame.quit()

#Game Over
```

#this calls the 'main' function when this script is executed

```
    if __name__ == '__main__':
        main()
```

In [5]:
```
# run pygame inside jupyter notebook
main()
```

In [ ]:
```

```