Python Programming

Binding

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

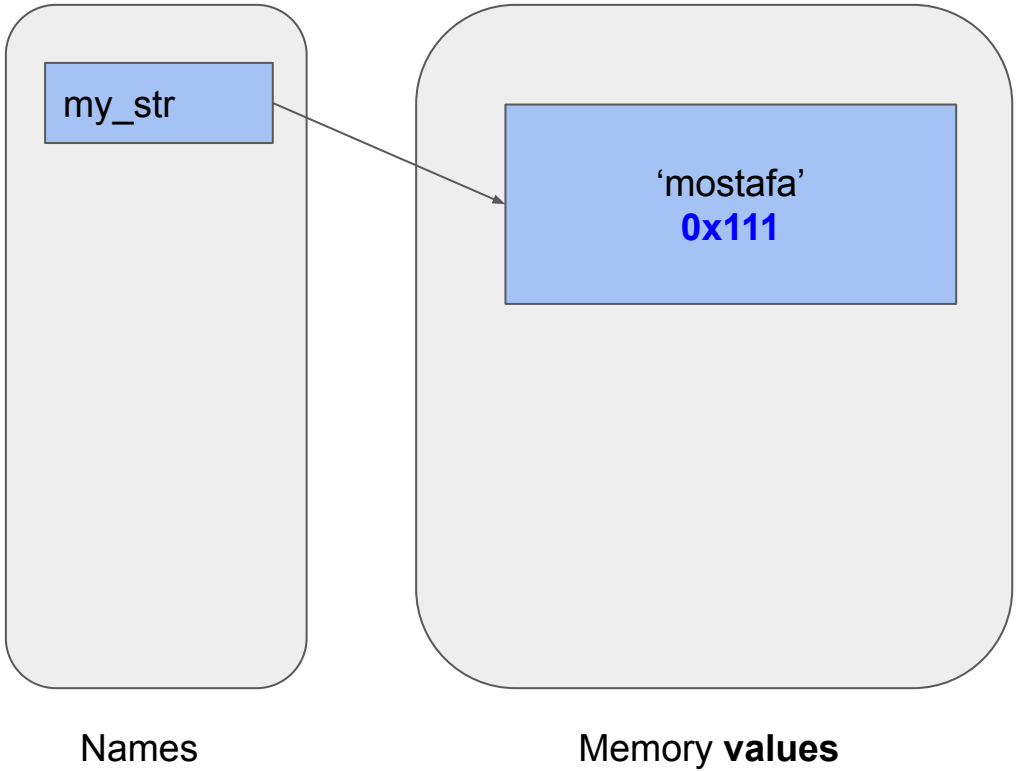
Name binding

- It is the **association** between a **name** and a value (an object)
  - So in Python, we bind (or attach) a name to an object.

```
4   my_str = 'mostafa'
5   # create 'mostafa' in memory. Say memory address 0x111
6   # Attach name my_str to address 0x111
7
8   my_str = 'ziad'
9   # create 'ziad' in memory. Say memory address 0x222
10  # REattach name my_str to address 0x222
11
12  another_str = my_str
13  # Attach  name my_str to address of my_str (0x222)
14
15  third_str = 'mostafa'
16  # attach third_str my_str to address 0x111  (maybe impl depedndent)
```

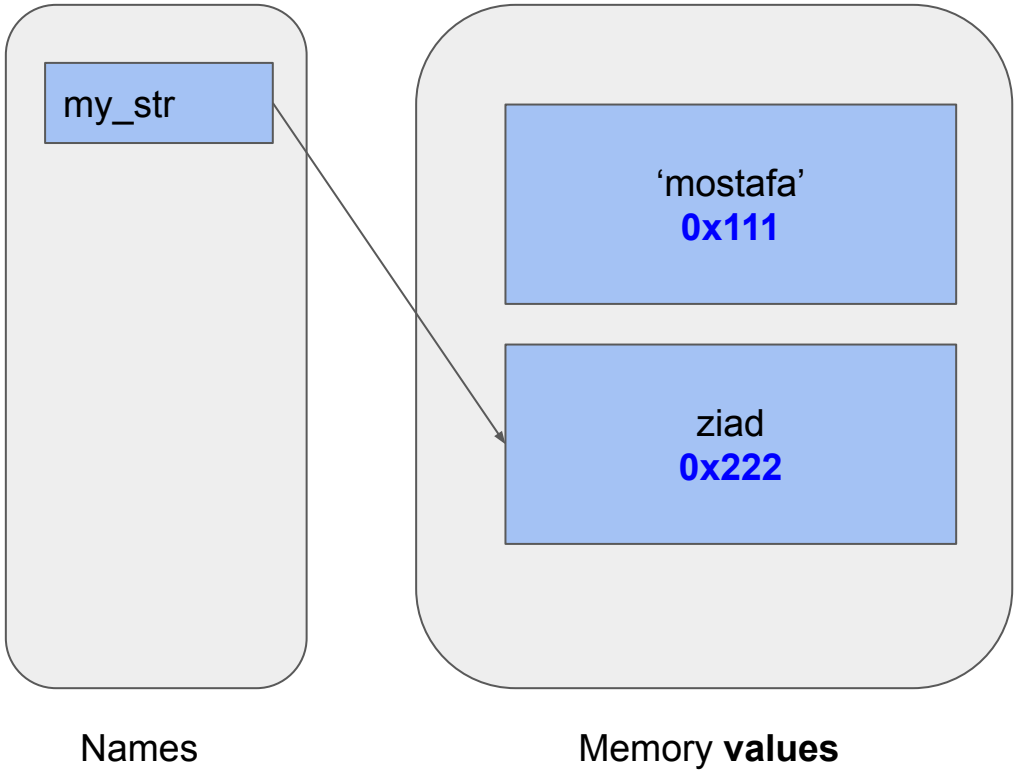
Name binding

```
4    my_str = 'mostafa'
5    # create 'mostafa' in mem
6    # Attach name my_str to
7
```



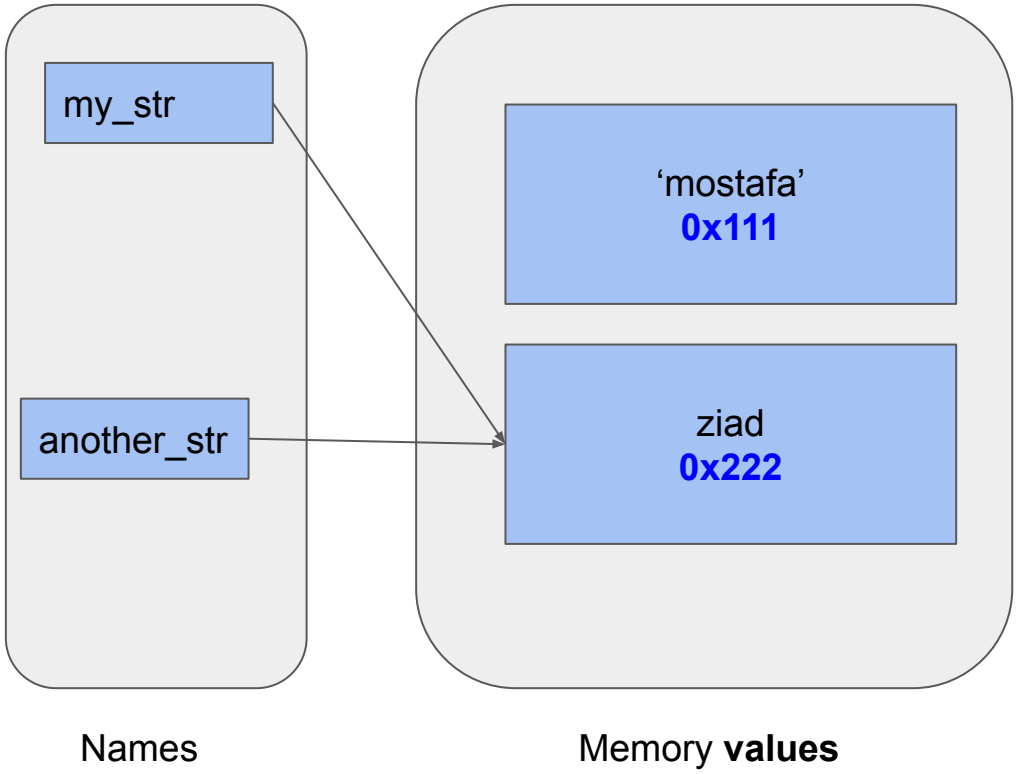
Name binding

```
4    my_str = 'mostafa'
5    # create 'mostafa' in me
6    # Attach name my_str to
7
8    my_str = 'ziad'
9    # create 'ziad' in memor
10   # REattach name my_str t
```



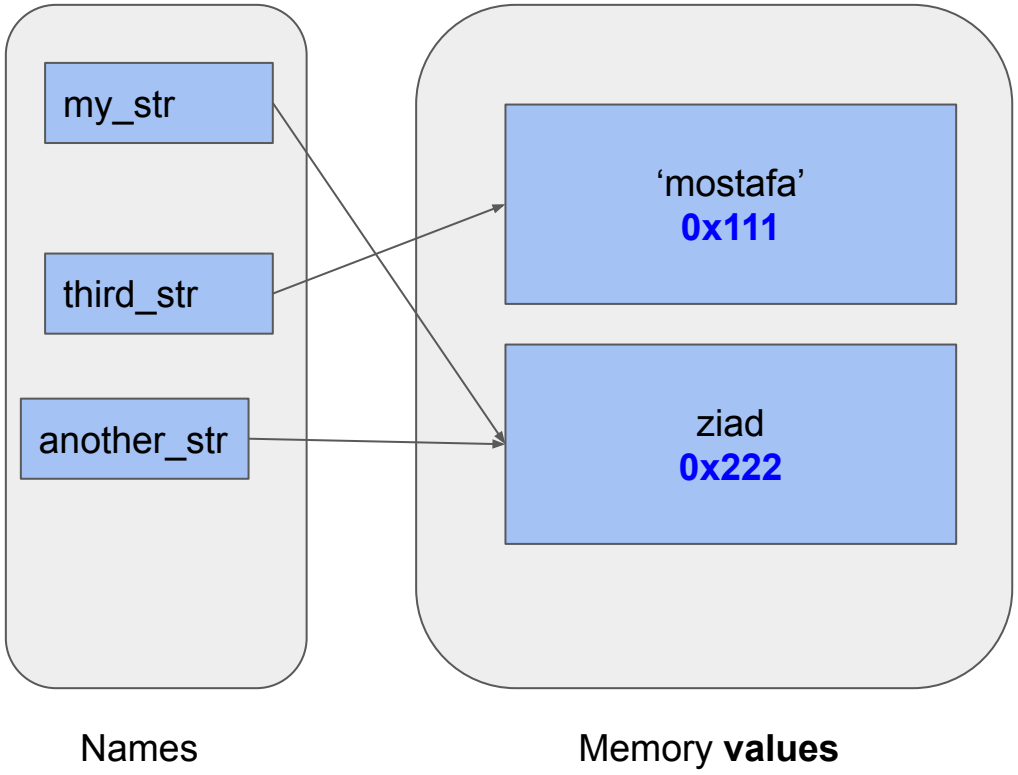
Name binding

```
4   my_str = 'mostafa'
5   # create 'mostafa' in me
6   # Attach name my_str to
7
8   my_str = 'ziad'
9   # create 'ziad' in memor
10  # REattach name my_str t
11
12  another_str = my_str
13  # Attach  name my_str to
```



Name binding

```
4   my_str = 'mostafa'
5   # create 'mostafa' in me
6   # Attach name my_str to
7
8   my_str = 'ziad'
9   # create 'ziad' in memor
10  # REattach name my_str t
11
12  another_str = my_str
13  # Attach  name my_str to
14
15  third_str = 'mostafa'
16  # attach third_str my_st
```



Unbound

Optional Reading

```
x = 10

def f1():
    x = 1       # x is now local var
    print(x)    # 1 = local var

def f2():
    print(x)    # 10 = use global var

def f3():
    print(y)    # NameError: name 'y' is not defined

def f4():
    # UnboundLocalError: local variable 'x' referenced before assignment
    print(y)    # Read next line first. BUT y not yet bounded to value
    y = 2       # this means through f: y is local var

def f5():
    # UnboundLocalError: local variable 'x' referenced before assignment
    print(x)    # same as above. But even global x is cancelled
    x = 2

def f6():
    # UnboundLocalError: local variable 'x' referenced before assignment
    x = x + 1   # similar issue. x = make it local. right side x+1: not bound yet!

def f7():
    # UnboundLocalError: local variable 'x' referenced before assignment
    x += 1  # same as x = x+1 in binding (though a bit more efficient)

def f8():
    global x
    x += 1  # cool
```

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."