sitepoint premium

Handout

Wrapping Your Head Around Python
Lesson 3 - Step 3
Brett Romero

Variable scope

Now we're going to work with **variable scope**. This basically means where can we access a variable under different conditions. So I'm starting off with some data:

```
customer1 = "John Doe"
pickupLocation1 = "350 5th Ave"
package1 = customer1 + pickupLocation1

customer2 = "Jane Doe"
pickupLocation2 = "100 7th Ave"
package2 = customer2 + pickupLocation2

customer3 = "Joe Daniels"
pickupLocation3 = "11 1st Ave"
package4 = customer3 + pickupLocation3

customerList = [customer1, customer2, customer3]
```

Now create a function:

```
def calculateFee(miles):
    result = miles * .5 + 2.50
    return result
```

Inside a `result` variable is used and you may want to reference it outside of the function. However, it will result in an error:

```
#result is out of scope here
#print(result)
```

This is because `result` is out of the scope - it is available only inside the function's body.

Global Space

One the other hand, `customerList` is in the global scope, so it can be referenced freely:

```
#customerList is global
#print(customerList)
```

So What About lambdas?

So what about lambdas?

```
add = lambda n1, n2: n1 + n2
print(add(1,3))
#lambda variables are out of scope here
#print(n1)
```

`n1` is defined in the context of the lambda, so it not available outside as well.

Creating a Lambda inside of a Function

```
def calculateFee3(miles):
    resultTemplate = lambda miles: miles * .5 + 2.5
    return resultTemplate(miles)
```

It works correctly.

Function changeGlobal

```
myint = 10
myint2 = 9
print("myint %d" % (myint))
def changeGlobal():
    #instance variable disappears after function execution
    myint = 20
    #global variable
    global myint2
    myint2 = 15
    print("myint inside func %d" % (myint))
    return

changeGlobal()
print("myint after func call %d" % (myint))
print("myint2 after func call %d" % (myint2))
```

Here we are changing the value of the variable defined in the global scope. Go ahead and run this script to check the resulting values.