目 录

1 变量作用域

python 的作用域一共有 4 个：

1. Local，局部作用域。
2. Enclosing，闭包函数外的函数中。
3. Global，全局作用域。
4. Built-in，内建作用域。

1.1 理解闭包函数

如果在一个内部函数里，对它外面的函数中的变量进行引用，那么这个函数就被认为是闭包函数。

举个例子：

```
def addx(x):
    # adder函数就是一个闭包函数
    def adder(y):
        return x+y
    return adder
```

1.2 变量的查找规则

在 Local 域找不到的时候，如果是闭包函数，就会到 Enclosing 域去找。如果不是闭包函数，就会直接去 Global 域找，最后到 Built-in 域去找。

1.3 改变作用域

python 只有 def、class 和 lambda 可以改变变量的作用域。如果在 def/class/lambda 内对变量进行赋值，那么变量在/def/class/lambda 中就是 Local 域，如下所示：

```
g = 1 # Global域
def fun():
    g = 2 # Local域
    return g

print fun() # 结果为2
print g # 结果为1
```

如果想在 def/class/lambda 引用 Global 变量，一旦对变量进行修改，那么它就会在整个 def/class/lambda 中变成 Local 域，那么很可能出现未赋值之前引用的错误：

```
var = 1
def fun1():
    print var # 出现未赋值之前引用的错误
    var = 200

def fun2():
    var = var + 1 # 出现未赋值之前引用的错误
    return var
```

1.3.1 Global 关键字

Global 关键字可以使得变量变为 Global 域，这样 def/class/lambda 中对变量进行修改，全局中的变量也会被修改。

1.4 globals() 和 locals()

1.4.1 globals() 函数

globals() 返回一个字典，这个字典包含了 Global 域中的所有变量，包括函数变量，如下所示：

```
def var():
    print "Hello world"

def fun():
    f = globals()['var']
    f()

fun() # 打印'Hello world'
```

1.4.2 locals() 函数

locals() 返回一个字典，这个字典包含了 Local 域中的所有变量，包括函数变量，如下所示：

```
def view():
    user = 'user'
    article = 'article'
    ip = 'ip'
    s = 'just a string'
    return locals() # 相当于 return {user=user, article=article, ip=ip, s=s}
```