**这个星期开始学习Python了，因为看的书都是基于Python2.x，而且我安装的是Python3.1，所以书上写的地方好多都不适用于Python3.1，特意在Google上search了一下3.x和2.x的区别。特此在自己的空间中记录一下，以备以后查找方便，也可以分享给想学习Python的friends.**

**1.性能**   
Py3.0运行 pystone benchmark的速度比Py2.5慢30%。Guido认为Py3.0有极大的优化空间，在字符串和整形操作上可   
以取得很好的优化结果。   
Py3.1性能比Py2.5慢15%，还有很大的提升空间。   
**2.编码**Py3.X源码文件默认使用utf-8编码，这就使得以下代码是合法的：   
    >>> 中国 = 'china'   
    >>>print(中国)   
    china   
**3. 语法**   
1）去除了<>，全部改用!=   
2）去除``，全部改用repr()   
3）关键词加入as 和with，还有True,False,None   
4）整型除法返回浮点数，要得到整型结果，请使用//   
5）加入nonlocal语句。使用noclocal x可以直接指派外围（非全局）变量   
6）去除print语句，加入print()函数实现相同的功能。同样的还有 exec语句，已经改为exec()函数   
   例如：   
     2.X: print "The answer is", 2\*2   
     3.X: print("The answer is", 2\*2)   
     2.X: print x,                              # 使用逗号结尾禁止换行   
     3.X: print(x, end=" ")                     # 使用空格代替换行   
     2.X: print                                 # 输出新行   
     3.X: print()                               # 输出新行   
     2.X: print >>sys.stderr, "fatal error"   
     3.X: print("fatal error", file=sys.stderr)   
     2.X: print (x, y)                          # 输出repr((x, y))   
     3.X: print((x, y))                         # 不同于print(x, y)!   
7）改变了顺序操作符的行为，例如x<y，当x和y类型不匹配时抛出TypeError而不是返回随即的 bool值    
8）输入函数改变了，删除了raw\_input，用input代替：   
   2.X:guess = int(raw\_input('Enter an integer : ')) # 读取键盘输入的方法   
   3.X:guess = int(input('Enter an integer : '))

9）去除元组参数解包。不能def(a, (b, c)):pass这样定义函数了   
10）新式的8进制字变量，相应地修改了oct()函数。   
   2.X的方式如下：   
     >>> 0666   
     438   
     >>> oct(438)   
     '0666'   
   3.X这样：   
     >>> 0666   
     SyntaxError: invalid token (<pyshell#63>, line 1)   
     >>> 0o666   
     438   
     >>> oct(438)   
     '0o666'   
11）增加了 2进制字面量和bin()函数   
    >>> bin(438)   
    '0b110110110'   
    >>> \_438 = '0b110110110'   
    >>> \_438   
    '0b110110110'   
12）扩展的可迭代解包。在Py3.X 里，a, b, \*rest = seq和 \*rest, a = seq都是合法的，只要求两点：rest是list   
对象和seq是可迭代的。   
13）新的super()，可以不再给super()传参数，   
    >>> class C(object):   
          def \_\_init\_\_(self, a):   
             print('C', a)   
    >>> class D(C):   
          def \_\_init(self, a):   
             super().\_\_init\_\_(a) # 无参数调用super()   
    >>> D(8)   
    C 8   
    <\_\_main\_\_.D object at 0x00D7ED90>   
14）新的metaclass语法：   
    class Foo(\*bases, \*\*kwds):   
      pass   
15）支持class decorator。用法与函数decorator一样：   
    >>> def foo(cls\_a):   
          def print\_func(self):   
             print('Hello, world!')   
          cls\_a.print = print\_func   
          return cls\_a   
    >>> @foo   
    class C(object):   
      pass   
    >>> C().print()   
    Hello, world!   
class decorator可以用来玩玩狸猫换太子的大把戏。更多请参阅PEP 3129   
**4. 字符串和字节串**   
1）现在字符串只有str一种类型，但它跟2.x版本的unicode几乎一样。

2）关于字节串，请参阅“数据类型”的第2条目   
**5.数据类型**   
1）Py3.X去除了long类型，现在只有一种整型——int，但它的行为就像2.X版本的long   
2）新增了bytes类型，对应于2.X版本的八位串，定义一个bytes字面量的方法如下：   
    >>> b = b'china'   
    >>> type(b)   
    <type 'bytes'>   
str对象和bytes对象可以使用.encode() (str -> bytes) or .decode() (bytes -> str)方法相互转化。   
    >>> s = b.decode()   
    >>> s   
    'china'   
    >>> b1 = s.encode()   
    >>> b1   
    b'china'   
3）dict的.keys()、.items 和.values()方法返回迭代器，而之前的iterkeys()等函数都被废弃。同时去掉的还有   
dict.has\_key()，用 in替代它吧   
**6.面向对象**1）引入抽象基类（Abstraact Base Classes，ABCs）。   
2）容器类和迭代器类被ABCs化，所以cellections模块里的类型比Py2.5多了很多。   
    >>> import collections   
    >>> print('\n'.join(dir(collections)))   
    Callable   
    Container   
    Hashable   
    ItemsView   
    Iterable   
    Iterator   
    KeysView   
    Mapping   
    MappingView   
    MutableMapping   
    MutableSequence   
    MutableSet   
    NamedTuple   
    Sequence   
    Set   
    Sized   
    ValuesView   
    \_\_all\_\_   
    \_\_builtins\_\_   
    \_\_doc\_\_   
    \_\_file\_\_   
    \_\_name\_\_   
    \_abcoll   
    \_itemgetter   
    \_sys   
    defaultdict   
    deque   
另外，数值类型也被ABCs化。关于这两点，请参阅 PEP 3119和PEP 3141。   
3）迭代器的next()方法改名为\_\_next\_\_()，并增加内置函数next()，用以调用迭代器的\_\_next\_\_()方法   
4）增加了@abstractmethod和 @abstractproperty两个 decorator，编写抽象方法（属性）更加方便。   
**7.异常**   
1）所以异常都从 BaseException继承，并删除了StardardError   
2）去除了异常类的序列行为和.message属性   
3）用 raise Exception(args)代替 raise Exception, args语法   
4）捕获异常的语法改变，引入了as关键字来标识异常实例，在Py2.5中：   
    >>> try:   
    ...    raise NotImplementedError('Error')   
    ... except NotImplementedError, error:

    ...    print error.message   
    ...   
    Error   
在Py3.0中：   
    >>> try:   
          raise NotImplementedError('Error')   
        except NotImplementedError as error: #注意这个 as   
          print(str(error))   
    Error   
5）异常链，因为\_\_context\_\_在3.0a1版本中没有实现   
**8.模块变动**   
1）移除了cPickle模块，可以使用pickle模块代替。最终我们将会有一个透明高效的模块。   
2）移除了imageop模块   
3）移除了 audiodev, Bastion, bsddb185, exceptions, linuxaudiodev, md5, MimeWriter, mimify, popen2,    
rexec, sets, sha, stringold, strop, sunaudiodev, timing和xmllib模块   
4）移除了bsddb模块(单独发布，可以从<http://www.jcea.es/programacion/pybsddb.htm>获取)   
5）移除了new模块   
6）os.tmpnam()和os.tmpfile()函数被移动到tmpfile模块下   
7）tokenize模块现在使用bytes工作。主要的入口点不再是generate\_tokens，而是 tokenize.tokenize()   
**9.其它**1）xrange() 改名为range()，要想使用range()获得一个list，必须显式调用：   
    >>> list(range(10))   
    [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]   
2）bytes对象不能hash，也不支持 b.lower()、b.strip()和b.split()方法，但对于后两者可以使用 b.strip(b’    
\n\t\r \f’)和b.split(b’ ‘)来达到相同目的   
3）zip()、map()和filter()都返回迭代器。而apply()、 callable()、coerce()、 execfile()、reduce()和reload   
()函数都被去除了

现在可以使用hasattr()来替换 callable(). hasattr()的语法如：hasattr(string, '\_\_name\_\_')

4）string.letters和相关的.lowercase和.uppercase被去除，请改用string.ascii\_letters 等   
5）如果x < y的不能比较，抛出TypeError异常。2.x版本是返回伪随机布尔值的   
6）\_\_getslice\_\_系列成员被废弃。a[i:j]根据上下文转换为a.\_\_getitem\_\_(slice(I, j))或 \_\_setitem\_\_和   
\_\_delitem\_\_调用   
7）file类被废弃，在Py2.5中：   
    >>> file   
    <type 'file'>   
在Py3.X中：   
    >>> file   
    Traceback (most recent call last):   
    File "<pyshell#120>", line 1, in <module>   
       file   
    NameError: name 'file' is not defined