Python惯例

“惯例”这个词指的是“习惯的做法，常规的办法，一贯的做法”，与这个词对应的英文单词叫“idiom”。由于Python跟其他很多编程语言在语法和使用上还是有比较显著的差别，因此作为一个Python开发者如果不能掌握这些惯例，就无法写出“Pythonic”的代码。下面我们总结了一些在Python开发中的惯用的代码。

1. 让代码既可以被导入又可以被执行。

```
if __name__ == '__main__':
```

2. 用下面的方式判断逻辑“真”或“假”。

```
if x:
if not x:
```

好的代码：

```
name = 'jackfrued'
fruits = ['apple', 'orange', 'grape']
owners = {'1001': '骆昊', '1002': '王大锤'}
if name and fruits and owners:
    print('I love fruits!')
```

不好的代码：

```
name = 'jackfrued'
fruits = ['apple', 'orange', 'grape']
owners = {'1001': '骆昊', '1002': '王大锤'}
if name != '' and len(fruits) > 0 and owners != {}:
    print('I love fruits!')
```

3. 善于使用in运算符。

```
if x in items: # 包含
for x in items: # 迭代
```

好的代码：

```
name = 'Hao LUO'
if 'L' in name:
    print('The name has an L in it.')
```

不好的代码：

```
name = 'Hao LUO'
if name.find('L') != -1:
    print('This name has an L in it!')
```

4. 不使用临时变量交换两个值。

```
a, b = b, a
```

5. 用序列构建字符串。

好的代码：

```
chars = ['j', 'a', 'c', 'k', 'f', 'r', 'u', 'e', 'd']
name = ''.join(chars)
print(name)  # jackfrued
```

不好的代码：

```
chars = ['j', 'a', 'c', 'k', 'f', 'r', 'u', 'e', 'd']
name = ''
for char in chars:
    name += char
print(name)  # jackfrued
```

6. EAFP优于LBYL。

EAFP - **E**asier to **A**sk **F**orgiveness than **P**ermission.

LBYL - **L**ook **B**efore **Y**ou **L**eap.

好的代码：

```
d = {'x': '5'}
try:
    value = int(d['x'])
    print(value)
except (KeyError, TypeError, ValueError):
    value = None
```

不好的代码：

```
d = {'x': '5'}
if 'x' in d and isinstance(d['x'], str) \
        and d['x'].isdigit():
    value = int(d['x'])
    print(value)
else:
    value = None
```

7. 使用enumerate进行迭代。

好的代码：

```
fruits = ['orange', 'grape', 'pitaya', 'blueberry']
for index, fruit in enumerate(fruits):
    print(index, ':', fruit)
```

不好的代码：

```
fruits = ['orange', 'grape', 'pitaya', 'blueberry']
index = 0
for fruit in fruits:
    print(index, ':', fruit)
    index += 1
```

8. 用生成式生成列表。

好的代码：

```
data = [7, 20, 3, 15, 11]
result = [num * 3 for num in data if num > 10]
print(result)  # [60, 45, 33]
```

不好的代码：

```
data = [7, 20, 3, 15, 11]
result = []
for i in data:
    if i > 10:
        result.append(i * 3)
print(result)  # [60, 45, 33]
```

9. 用zip组合键和值来创建字典。

   好的代码：

   ```
   keys = ['1001', '1002', '1003']
   values = ['骆昊', '王大锤', '白元芳']
   d = dict(zip(keys, values))
   print(d)
   ```

   不好的代码：

   ```
   keys = ['1001', '1002', '1003']
   values = ['骆昊', '王大锤', '白元芳']
   d = {}
   for i, key in enumerate(keys):
       d[key] = values[i]
   print(d)
   ```

> *说明：这篇文章的内容来自于网络，有兴趣的读者可以阅读原文。*