**Python之禅**

在Python交互式解释器中输 入import this就会显示Tim Peters的The Zen of Python：

>>> import this

The Zen of Python, by Tim Peters

Beautiful is better than ugly.

Explicit is better than implicit.

Simple is better than complex.

Complex is better than complicated.

Flat is better than nested.

Sparse is better than dense.

Readability counts.

Special cases aren't special enough to break the rules.

Although practicality beats purity.

Errors should never pass silently.

Unless explicitly silenced.

In the face of ambiguity, refuse the temptation to guess.

There should be one-- and preferably only one --obvious way to do it.

Although that way may not be obvious at first unless you're Dutch.

Now is better than never.

Although never is often better than \*right\* now.

If the implementation is hard to explain, it's a bad idea.

If the implementation is easy to explain, it may be a good idea.

Namespaces are one honking great idea -- let's do more of those!

>>>

下面是翻译和解释：

**Python之禅 by Tim Peters**

* 优美胜于丑陋（Python 以编写优美的代码为目标）
* 明了胜于晦涩（优美的代码应当是明了的，命名规范，风格相似）
* 简洁胜于复杂（优美的代码应当是简洁的，不要有复杂的内部实现）
* **复杂胜于凌乱（如果复杂不可避免，那代码间也不能有难懂的关系，要保持接口简洁）**
* **扁平胜于嵌套（优美的代码应当是扁平的，不能有太多的嵌套）**
* 间隔胜于紧凑（优美的代码有适当的间隔，不要奢望一行代码解决问题）
* 可读性很重要（优美的代码是可读的）
* 即便假借特例的实用性之名，也不可违背这些规则（这些规则至高无上）
* 不要包容所有错误，除非你确定需要这样做（精准地捕获异常，不写 except:pass 风格的代码）
* 当存在多种可能，不要尝试去猜测
* 而是尽量找一种，最好是唯一一种明显的解决方案（如果不确定，就用穷举法）
* 虽然这并不容易，因为你不是 Python 之父（这里的 Dutch 是指 Guido ）
* **做也许好过不做，但不假思索就动手还不如不做（动手之前要细思量）**
* **如果你无法向人描述你的方案，那肯定不是一个好方案；反之亦然（方案测评标准）**
* 命名空间是一种绝妙的理念，我们应当多加利用（倡导与号召）