**S1T2 一些学习笔记**

1. [阅读python-binance “client.py”source code时] “class [ClassName]” vs. “class [ClassName(**object**)]”:

**In Python 2.x**, when you **inherit from "object"** you class is a **"new style" class** - that was implemented back in Python 2.2 (around 2001) - **The non inheriting from "object" case creates an "old style" class**, that was actually maintained only for backwards compatibility.

The great benefit of "new style" classes is the unification of types across Python - prior to that, one could not subclass built-in types such as int, list, dict, properly. There was also specified a "descriptor protocol" which describes a protocol for retrieving and setting attributes in an object, giving the language a lot of flexibility. (It is more visible when one does use a Python "property" in a class).

It is strongly recommended that in **Python 2.x**, all your classes are new style - using old style classes may work for some single straightforward cases, but they can generate a lot of subtle, difficult to find, errors, when you try to use properties, pickle, descriptors, and other advanced features. Above all, when you try to check the "type" of an object, it will be the same (type "instance") for all objects from old style classes, even if they are from different user defined classes.

**In python 3, new style classes are used by default and you will no longer need to subclass “object”.**

**[source:** [**https://stackoverflow.com/questions/10043963/class-classname-versus-class-classnameobject**](https://stackoverflow.com/questions/10043963/class-classname-versus-class-classnameobject)**]**