[笔记][LIKE-Python-1]

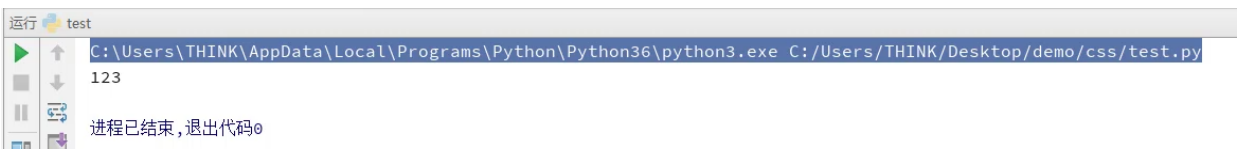
Python

014. PyCharm软件的使用

注意还可以从版本仓库检出

菜单栏，工具条，导航栏，代码编辑区域

重命名 `Shift + F6`



代码执行后的窗口的第一行文字，实际上可以复制了以后在 `cmd` 中运行。

`Ctrl + /` 注释

`Ctrl - Alt - L` 自动调整为 `PEP 8` 规范

展开、折叠快捷键

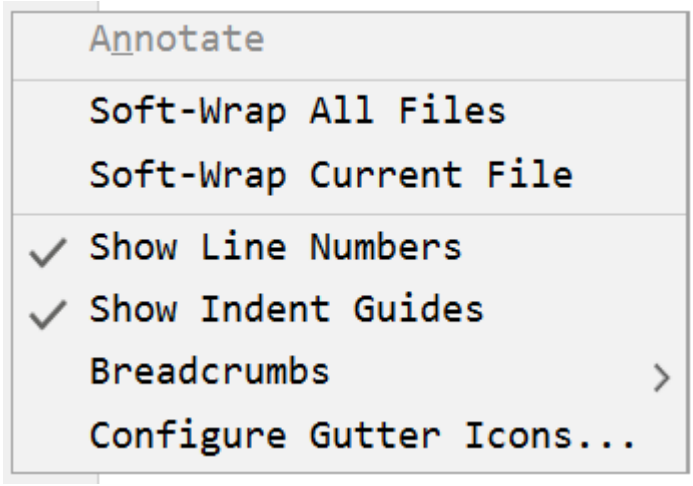
展开点加号，还可以双击后面的三个点 `...`

`Ctrl - 点击` 代码跳转，或者 `Ctlr - b`

`PyCharm` 自带的终端跟 `cmd` 还是有一点区别的，比如在自带终端输入 `python` 就无法进入 `Python` 交互模式。

> 【!】**注意**：`PyCharm 2017` 不可以，`PyCharm 2018` 可以。

右击可以显示行号、隐藏行号。

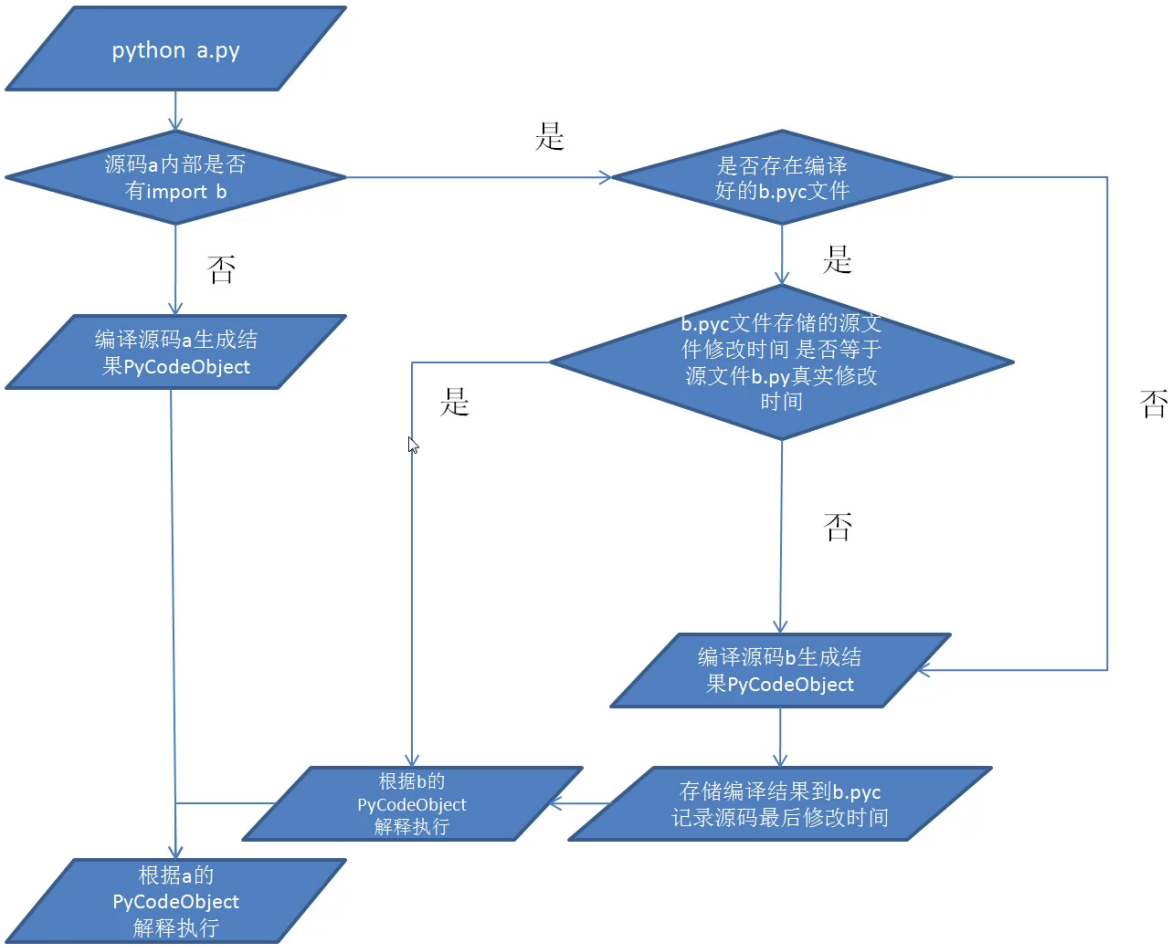


软换行

File - Settings - Editor - General

勾选 `Use soft wraps in editor`

015. Python程序执行机制



先编译成中间的字节码，然后由解释器根据中间的字节码，再去解释执行。

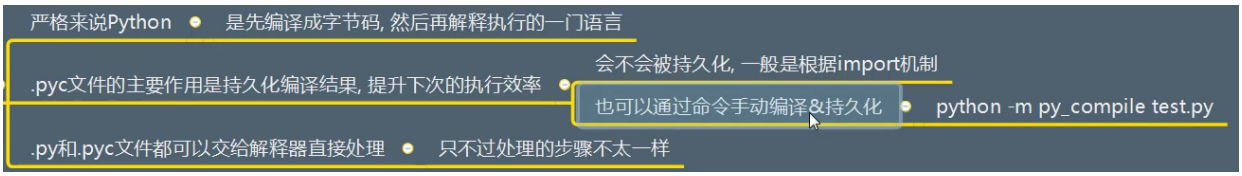
> 【!】只有导入的模块才会有存储在硬盘上的 `*.pyc` 文件
> 换个说法：只有导入的模块才会把字节码文件以 `pyc` 后缀名持久化。

`pyc` 的类型描述是：Compiled Python File，编译过的 Python 文件

注意：假如只有 `pyc` 而没有 `py` 文件，`Python` 解释器会直接执行 `pyc` 文件。
也就是不进行 `pyc` 文件和 `py` 文件的时间戳比较了。

可以直接这样执行 `pyc` 文件：`python test.pyc`

注意



为什么会把 `import` 导入的模块持久化呢？
在这里其实做了一个假设，因为 `import` 进来的东西一般是作为工具使用，是已经实现好的代码，不会经常改动，所以编译成字节码并持久化可以提升执行效率。

`.pyc` 文件的作用：

- 提升程序执行效率
- 一定程度上保护源码

注意：`python a.py` 是不会执行 `a.pyc` 的，而且也不会修改 `a.pyc`。

完成于 `201810161444`