9. 顶级组件

Python 解释器可以从多种源获得输入：作为标准输入或程序参数传入的脚本，以交互方式键入的语句，导入的模块源文件等等。 这一章将给出在这些情况下所用的语法。

9.1. 完整的 Python 程序

虽然语言规范描述不必规定如何唤起语言解释器，但对完整的 Python 程序加以说明还是很有用的。一个完整的 Python 程序会在最小初始化环境中被执行：所有内置和标准模块均为可用，但均处于未初始化状态，只有 `sys` (各种系统服务), `builtins` (内置函数、异常以及 `None`) 和 `__main__` 除外。最后一个模块用于为完整程序的执行提供局部和全局命名空间。

适用于一个完整 Python 程序的语法即下节所描述的文件输入。

解释器也可以通过交互模式被唤起；在此情况下，它并不读取和执行一个完整程序，而是每次读取和执行一条语句（可能为复合语句）。 此时的初始环境与一个完整程序的相同；每条语句会在 `__main__` 的命名空间中被执行。

一个完整程序可通过三种形式被传递给解释器：使用 `-c` *字符串* 命令行选项，使用一个文件作为第一个命令行参数，或者使用标准输入。 如果文件或标准输入是一个 tty 设置，解释器会进入交互模式；否则的话，它会将文件当作一个完整程序来执行。

9.2. 文件输入

所有从非交互式文件读取的输入都具有相同的形式:

```
file_input: (NEWLINE | statement)* ENDMARKER
```

此语法用于下列几种情况:

- 解析一个完整 Python 程序时（从文件或字符串）；
- 解析一个模块时；
- 解析一个传递给 `exec()` 函数的字符串时；

9.3. 交互式输入

交互模式下的输入使用以下语法进行解析:

```
interactive_input: [stmt_list] NEWLINE | compound_stmt NEWLINE | ENDMARKER
```

请注意在交互模式下一条（最高层级）复合语句必须带有一个空行；这对于帮助解析器确定输入的结束是必须的。

9.4. 表达式输入

eval() 被用于表达式输入。 它会忽略开头的空白。 传递给 eval() 的字符串参数必须具有以下形式:

```
eval_input: expression_list NEWLINE* ENDMARKER
```