



2. Python 解释器

2.1. 调用解释器

The Python interpreter is usually installed as `/usr/local/bin/python3.11` on those machines where it is available; putting `/usr/local/bin` in your Unix shell's search path makes it possible to start it by typing the command:

```
python3.11
```

这样，就可以在 shell 中运行 Python 了 [1]。因为可以选择安装目录，解释器也有可能安装在别的位置；如果还不明白，就去问问身边的 Python 大神或系统管理员。（例如，常见备选路径还有 `/usr/local/python`。）

On Windows machines where you have installed Python from the [Microsoft Store](#), the `python3.11` command will be available. If you have the [py.exe launcher](#) installed, you can use the `py` command. See [附录：设置环境变量](#) for other ways to launch Python.

在主提示符中，输入文件结束符（Unix 里是 `Control-D`，Windows 里是 `Control-Z`），就会退出解释器，退出状态码为 0。如果不能退出，还可以输入这个命令：`quit()`。

在支持 [GNU Readline](#) 库的系统中，解释器的行编辑功能包括交互式编辑、历史替换、代码补全等。检测是否支持命令行编辑最快速的方式是，在首次出现 Python 提示符时，输入 `Control-P`。听到“哔”提示音，说明支持行编辑；请参阅附录 [交互式编辑和编辑历史](#)，了解功能键。如果没有反应，或回显了 `^P`，则说明不支持行编辑；只能用退格键删除当前行的字符。

解释器的操作方式类似 Unix Shell：用与 `tty` 设备关联的标准输入调用时，可以交互式地读取和执行命令；以文件名参数，或标准输入文件调用时，则读取并执行文件中的 *脚本*。

A second way of starting the interpreter is `python -c command [arg] ...`, which executes the statement(s) in *command*,



3.11.0



转向

the shell, it is usually advised to quote *command* in its entirety.

Python 模块也可以当作脚本使用。输入：`python -m module [arg] ...`，会执行 *module* 的源文件，这跟在命令行把路径写全了一样。

在交互模式下运行脚本文件，只要在脚本名称参数前，加上选项 `-i` 就可以了。

命令行的所有选项详见 [命令行与环境](#)。

2.1.1. 传入参数

解释器读取命令行参数，把脚本名与其他参数转化为字符串列表存到 `sys` 模块的 `argv` 变量里。执行 `import sys`，可以导入这个模块，并访问该列表。该列表最少有一个元素；未给定输入参数时，`sys.argv[0]` 是空字符串。给定脚本名是 `'-'`（标准输入）时，`sys.argv[0]` 是 `'-'`。使用 `-c command` 时，`sys.argv[0]` 是 `'-c'`。如果使用选项 `-m module`，`sys.argv[0]` 就是包含目录的模块全名。解释器不处理 `-c command` 或 `-m module` 之后的选项，而是直接留在 `sys.argv` 中由命令或模块来处理。

2.1.2. 交互模式

在终端 (tty) 输入并执行指令时，解释器在 *交互模式 (interactive mode)* 中运行。在这种模式中，会显示 *主提示符*，提示输入下一条指令，主提示符通常用三个大于号 (`>>>`) 表示；输入连续行时，显示 *次要提示符*，默认是三个点 (`...`)。进入解释器时，首先显示欢迎信息、版本信息、版权声明，然后才是提示符：

```
$ python3.11
Python 3.11 (default, April 4 2021, 09:25:04)
[GCC 10.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```



3.11.0



转向

```
>>> the_world_is_flat = True
>>> if the_world_is_flat:
...     print("Be careful not to fall off!")
...
Be careful not to fall off!
```

>>>

交互模式的内容详见 [交互模式](#)。

2.2. 解释器的运行环境

2.2.1. 源文件的字符编码

默认情况下，Python 源码文件的编码是 UTF-8。这种编码支持世界上大多数语言的字符，可以用于字符串面值、变量、函数名及注释——尽管标准库只用常规的 ASCII 字符作为变量名或函数名，可移植代码都应遵守此约定。要正确显示这些字符，编辑器必须能识别 UTF-8 编码，而且必须使用支持文件中所有字符的字体。

如果不使用默认编码，则要声明文件的编码，文件的 第一行要写成特殊注释。句法如下：

```
# -*- coding: encoding -*-
```

其中，*encoding* 可以是 Python 支持的任意一种 [codecs](#)。

比如，声明使用 Windows-1252 编码，源码文件要写成：

```
# -*- coding: cp1252 -*-
```

第一行的规则也有一种例外情况，源码以 [UNIX "shebang" 行](#) 开头。此时，编码声明要写在文件的第二行。例如：



3.11.0



转向

备注

[1] Unix 系统中，为了不与同时安装的 Python 2.x 冲突，Python 3.x 解释器默认安装的执行文件名不是 `python`。