

Python is an experiment in how much freedom programmers need. Too much freedom and nobody can read another's code; too little and expressiveness is endangered. -- Guido van Rossum

第 2 章 Python 入门

Python 语言本身也是强大的学习工具，越快上手越好，故本书从第 2 章就开始介绍 Python。

2.1 为何使用 Python

Python 是一款主流的通用编程语言(a general-purpose programming language)。

1989 年 12 月, 荷兰人吉多·范罗苏姆(Guido van Rossum)为打发圣诞假期而开发 Python, 并于 1991 年正式发布。根据他喜欢的英国室内剧 Monty Python 将其命名为 Python。

2000 年, Python 2.0 发布, 并最终定格为 Python 2.7。

2008 年, Python 3.0 发布, 目前最新版为 Python 3.9。

为进一步改进 Python 语言, Python 3 并不向下兼容 Python 2(no backward compatibility), 故有些开发者依然使用 Python 2。

Python 3 无疑是 Python 语言的未来, 而对于 Python 2 的技术支持将于 2020 年停止, 故本书使用 Python 3。

Python 语言的主要优势在于其代码的可读性(readability)与简洁高效, 接近于英语, 被誉为“可执行的伪代码”(executable pseudocode)。

与编程语言 C、C++或 Java 相比, Python 代码的长度通常只是其三分之一至五分之一, 极大提高了开发者的生产率(developer productivity)。

Python 程序便于在各计算机平台之间迁移(program portibility), 易与其他语言对接整合(component integration), 有“胶水语言”(a glue language)的美誉。

由于 Python 用户群体庞大, 故拥有强大的“支持库”(support libraries), 且日益增长。

Python 的缺点：首先，它的运行速度一般不及更为底层的语言，比如 C 或 C++。但随着计算机本身速度不断提升，Python 已足以胜任大多数任务。对于确实需要高速的部分任务，可用 C 语言编写，再连回 Python 程序。

其次，Python 作为通用语言，并非为统计计算或机器学习而设计。在机器学习中应用 Python 时，常需导入一些“标准库”(standard libraries)或“第三方库”(third-party libraries)。

作为机器学习的另一流行语言，R 语言由统计学家为统计计算而发明，故特别便于统计分析与数据处理。

Python 作为计算机科学家打造的通用语言，其用途比 R 更为广泛，且便于部署在企业生产经营的各个环节。

2.2 Python 与 Spyder 的安装

作为开源软件, Python 语言可从官网 <https://www.python.org> 免费下载。

使用 Python 需经常调用一些标准库, 逐一下载这些标准库比较费事。

下载 Python 的最简捷方法, 是从 Anaconda 的平台直接下载。Anaconda 平台已集成了诸多 Python 标准库以及 PyCharm、Jupyter 与 Spyder 等流行的集成开发环境(IDE), 十分方便。

为下载 Anaconda, 可登录其官网:

<https://www.anaconda.com/products/individual>。

在此页面底部选择合适的 Anaconda 安装包(Anaconda Installer), 参见图 2.1。比如, 要在 64 位 Windows 系统下安装 Python 3, 可点击下载左上角的安装包“64-Bit Graphical Installer (466 MB)”。



图 2.1 Anaconda 官网的下载页面

安装 Anaconda 后, 在电脑程序的开始菜单会出现如下选项(参见图 2.2):

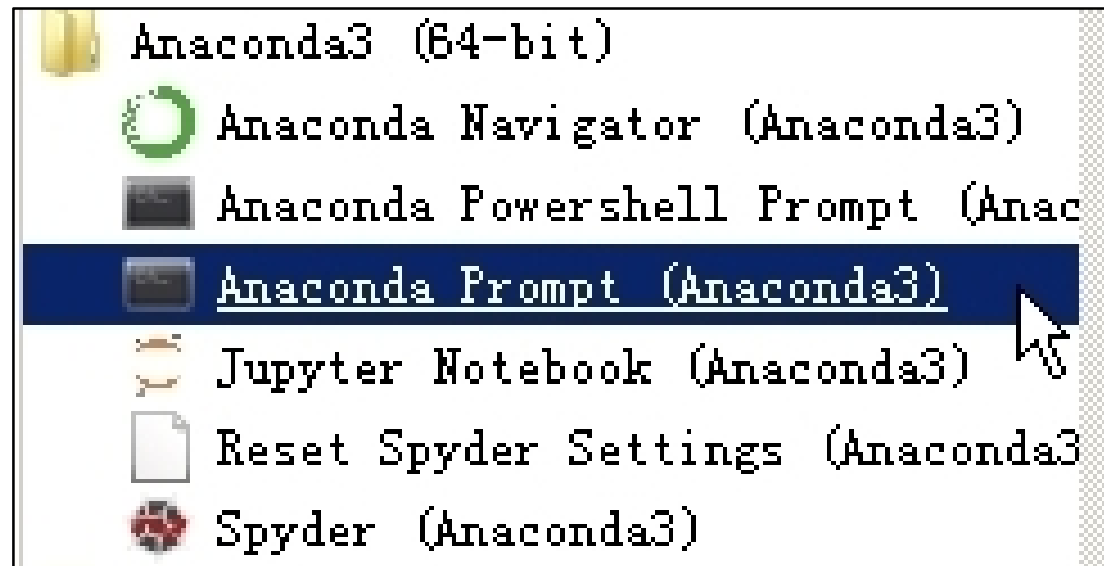
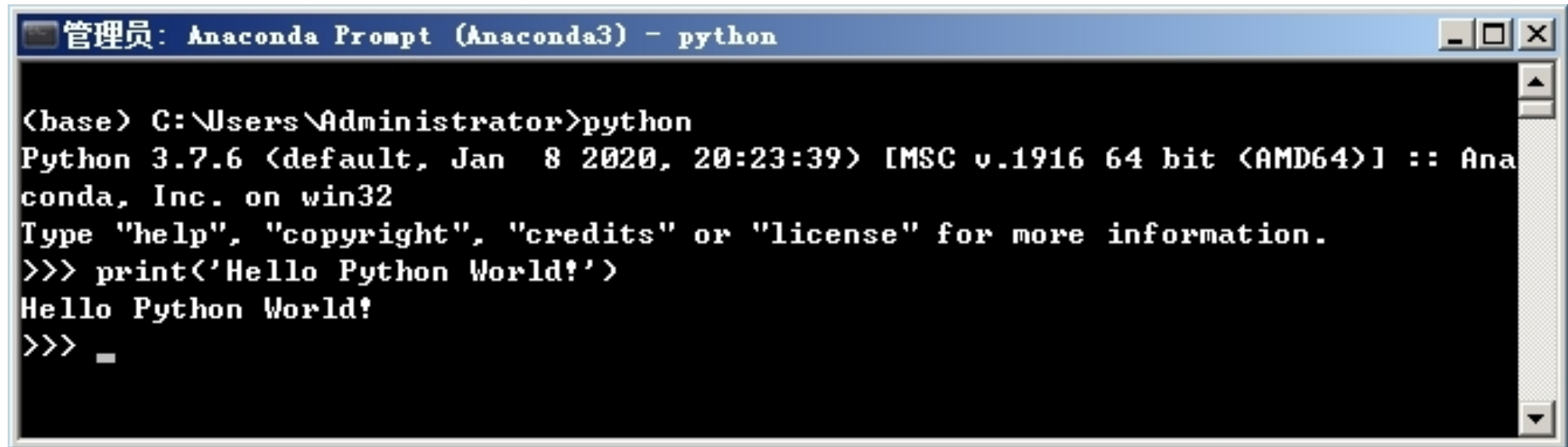


图 2.2 开始菜单中的 Anaconda

点击其中的“Anaconda Prompt”(Anaconda 提示符), 即可打开如下窗口(参见图 2.3):

The image shows a screenshot of a Windows command prompt window titled "管理员: Anaconda Prompt (Anaconda3) - python". The window has a black background with white text. The text inside the window shows a user running the command "python" at the prompt "<base> C:\Users\Administrator>". This opens a Python 3.7.6 shell. The shell displays version information: "Python 3.7.6 (default, Jan 8 2020, 20:23:39) [MSC v.1916 64 bit (AMD64)] :: Anaconda, Inc. on win32". It then prompts the user to type "help", "copyright", "credits", or "license" for more information. The user enters ">>> print('Hello Python World!')", and the shell outputs "Hello Python World!". The prompt ">>> _" is shown at the bottom, indicating the shell is ready for the next command.

```
<base> C:\Users\Administrator>python
Python 3.7.6 (default, Jan 8 2020, 20:23:39) [MSC v.1916 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print('Hello Python World!')
Hello Python World!
>>> _
```

图 2.3 Anaconda Prompt 窗口

Anaconda Prompt 的作用主要在于管理与 Python 相关的库(libraries), 比如安装某个第三方库, 详见后续章节。

在 Anaconda Prompt 输入 “python” 并回车, 即可进行 Python 程序, 其提示符(prompt)为“>>>”。比如, 输入以下命令, 即可打印“Hello Python World!”:

```
>>> print('Hello Python World!')  
Hello Python World!
```

在 Anaconda Prompt 只能交互式地执行 Python 命令, 即每输入一行命令后, 立即回车执行; 然后, 才能再输入下一行命令。

在多数情况下, 我们希望对 Python 命令进行批处理, 即同时输入并执行多行命令。

这就需要使用一个“文本编辑器”(text editor)来写 Python 程序, 以及一个更为友好的“图形用户界面”(Graphical User Interface, 简记 GUI)。

Anaconda 本身已经包含几款常见的“集成开发环境”(Integrated Development Environment, 简记 IDE), 比如 PyCharm、Jupyter Notebook 与 Spyder。

PyCharm 为专业的编程 IDE, 适用于大型项目的开发。

基于网页(web-based)的 Jupyter Notebook 则便于展示运行结果, 常用于数据科学领域。

对于机器学习的初学者, 推荐使用 Spyder, 其全称为 “Scientific Python Development Environment”, 特别便于科学计算。

在图 2.2 中的 Anaconda 菜单, 选择最后一项 “Spyder”, 然后点击右键, 选择 “发送到” → “桌面快捷方式”, 即可在电脑桌面建立一个 Spyder 的快捷方式。

打开 Spyder 后, 会看到如下界面(参见图 2.4):

Spyder 默认的背景颜色为黑色(Spyder Dark), 在此为便于展示, 已将背景设为白色(Spyder), 参见下文。

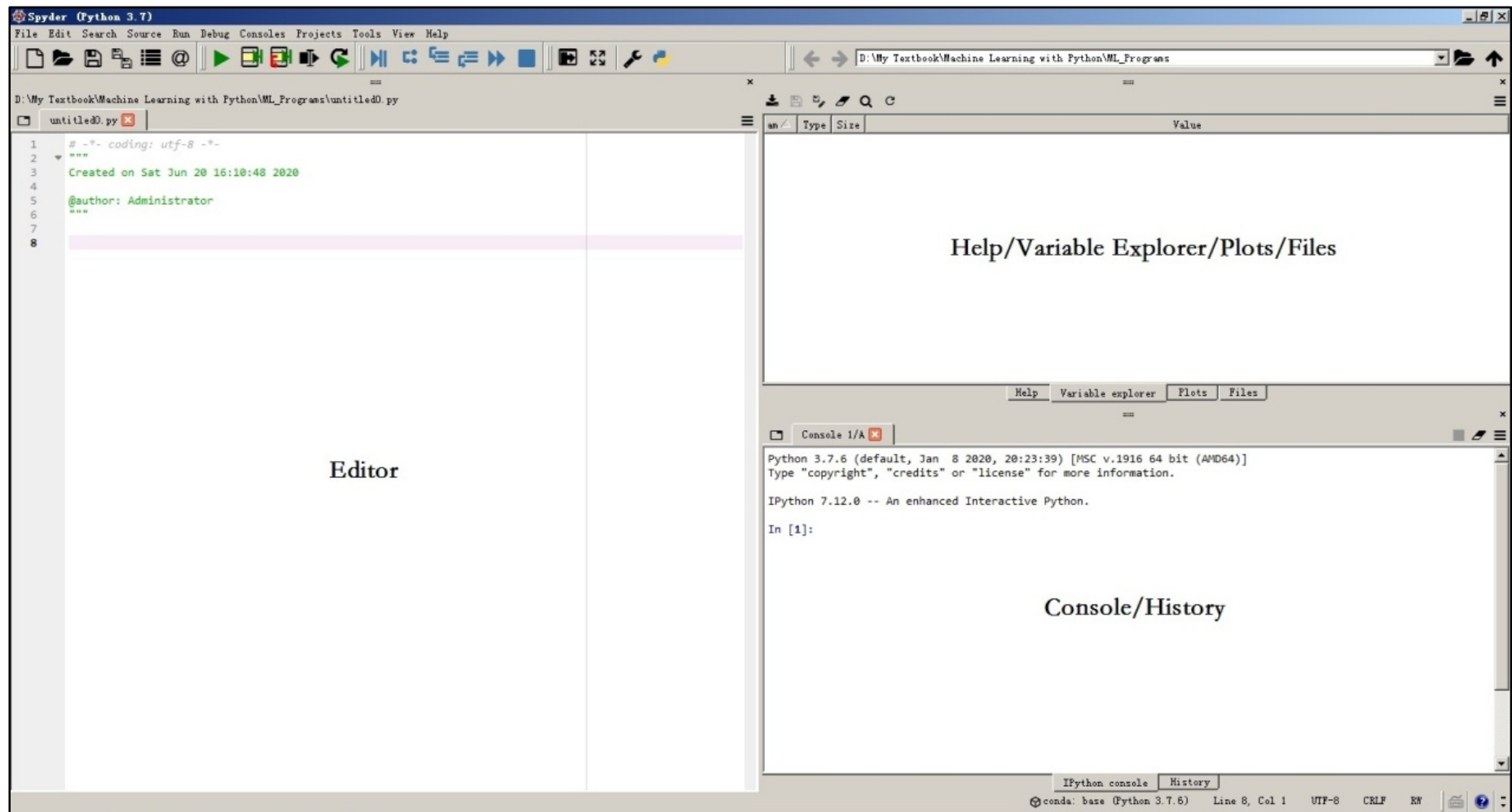


图 2.4 Spyder 的界面

左边为**编辑器(Editor)**窗口, 在此输入 Python 程序(Python Script, 也称为“Python 脚本”), 即可对多个 Python 命令以**批处理方式(batch mode)**运行; 而且很容易随时编辑 Python 程序, 通常比交互式运行命令更为方便。

Spyder 编辑器比一般的文本编辑器功能更强, 比如支持“语法高亮”(syntax highlighting)、使用 Tab 键“自动补齐代码”(auto completion)等。

基于自动补齐功能的重要性, 也可以下载安装 Kite 插件(<https://kite.com/download>)。Kite 使用机器学习的方法预测你的打字结果, 故自动补齐功能比 Spyder 的自带功能更强。

右上角为**帮助/变量探索器/画图/文件**(Help/Variable Explorer/Plots/Files)窗口, 可点击该窗口底部的窗格(tab)进行切换, 分别用于显示帮助文件、内存变量、画图结果与文件目录。

右下角为**控制台(Console)**窗口, 可在此交互式地(interactively)输入单个 Python 命令, 然后回车(按下 Enter 键)即可执行, 并在同一窗口显示输出结果。如想清空控制台的内容, 可同时按下“Ctrl + L”。一般来说, 交互式的命令输入方法仅用于探索性测试。

在右上角的最上方则显示“当前工作路径”(current working directory), 便于修改; 这是 Python 与电脑硬盘交互文件的指定位置(详见第 2.17 节)。

以上窗口布局只是 Spyder 的默认布局(default layout)。可以从菜单中选择“View”→“Window Layouts”，选择你自己喜欢的布局，参见图 2.5。比如，若你习惯于 RStudio 的布局，可选“Rstudio layout”。

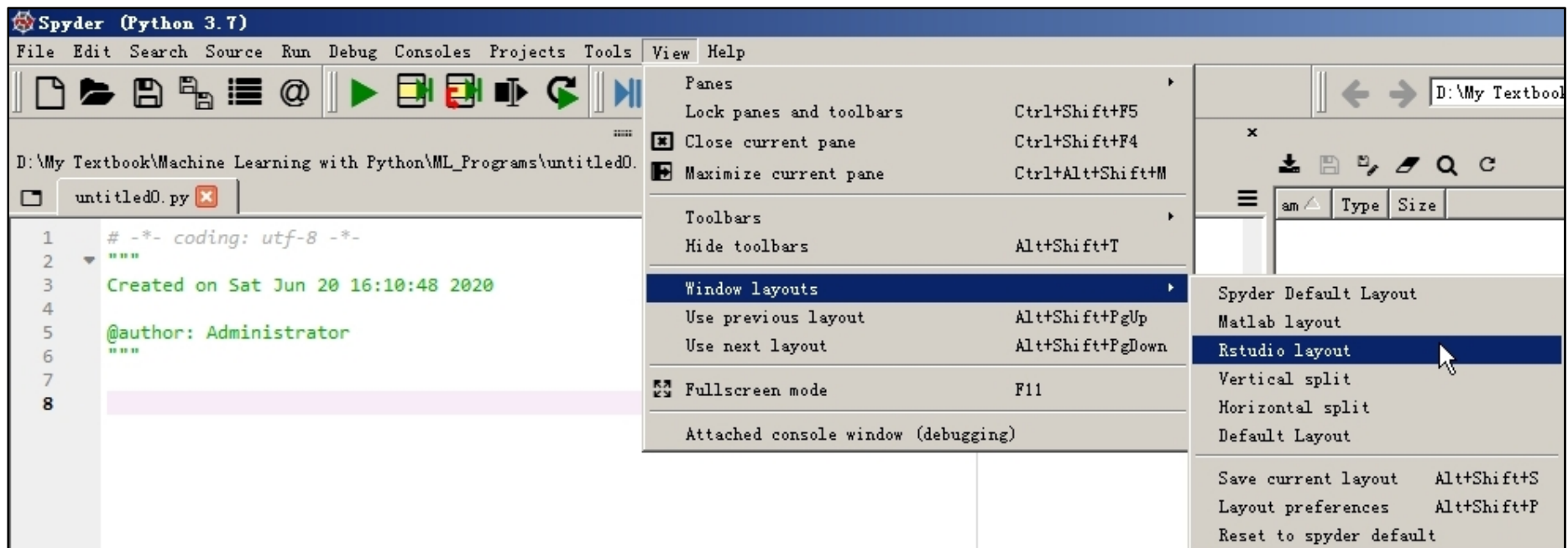


图 2.5 选择 Spyder 的窗口布局

如不喜欢 Spyder 默认的黑色背景, 可更换“主题”(theme)。比如, 从菜单栏中选择“Tools”→“Preferences”→“Appearance”→“Syntax highlighting theme”→“Spyder”, 即可换为白色背景, 参见图 2.6。

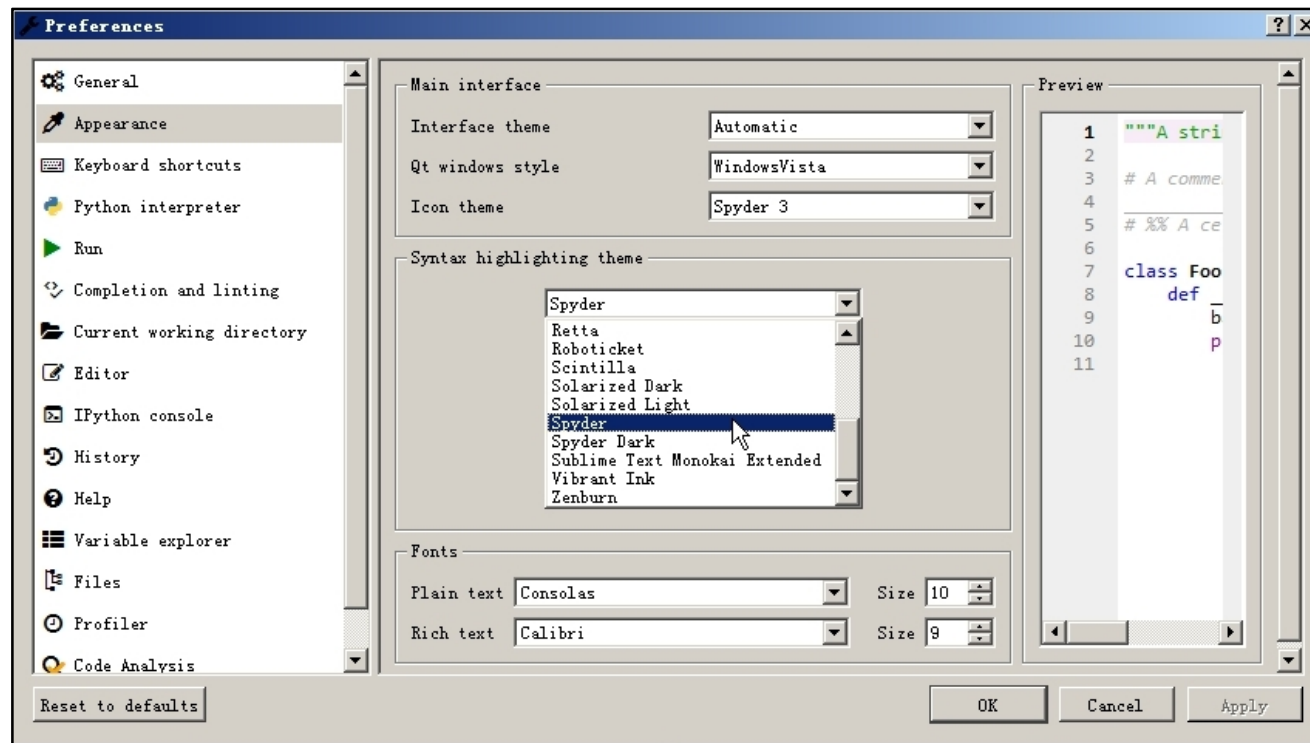


图 2.6 选择 Spyder 的背景主题

如果要运行编辑器的整个文件(run file), 可按快捷键 F5, 或点击菜单栏的绿色三角形图标, 参见图 2.7:



图 2.7 运行整个 Python 程序

如果只想单独运行某行命令, 可将光标放于该行(current line), 然后按快捷键 F9, 或点击菜单栏的如下图标(参见图 2.8 中鼠标所指位置)。

若想同时运行多行命令, 可用鼠标先同时选择这几行命令, 然后按快捷键 F9 或点击相应图标, 即可运行(Run selection)。

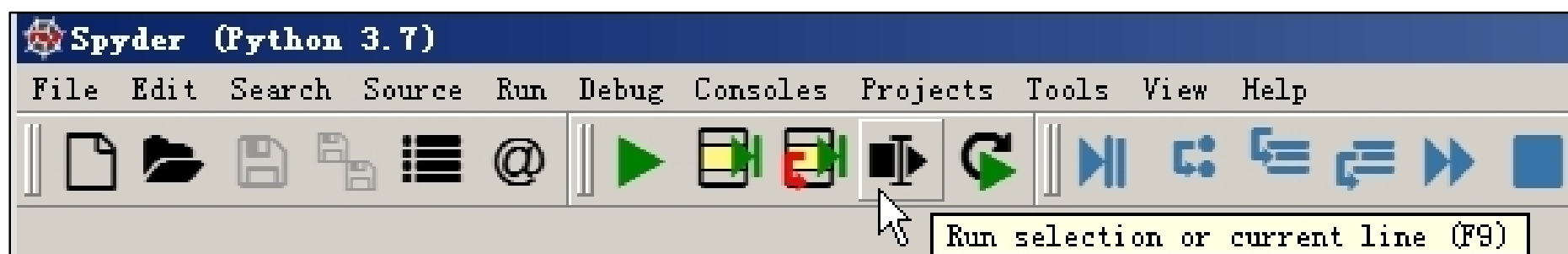


图 2.8 运行某行或选中的部分命令

Spyder 设置了大量的快捷键, 如想查看这些快捷键, 可从菜单栏中选择 “Tools” → “Preferences” → “Appearance” → “Keyboard shortcuts”, 参见图 2.9。

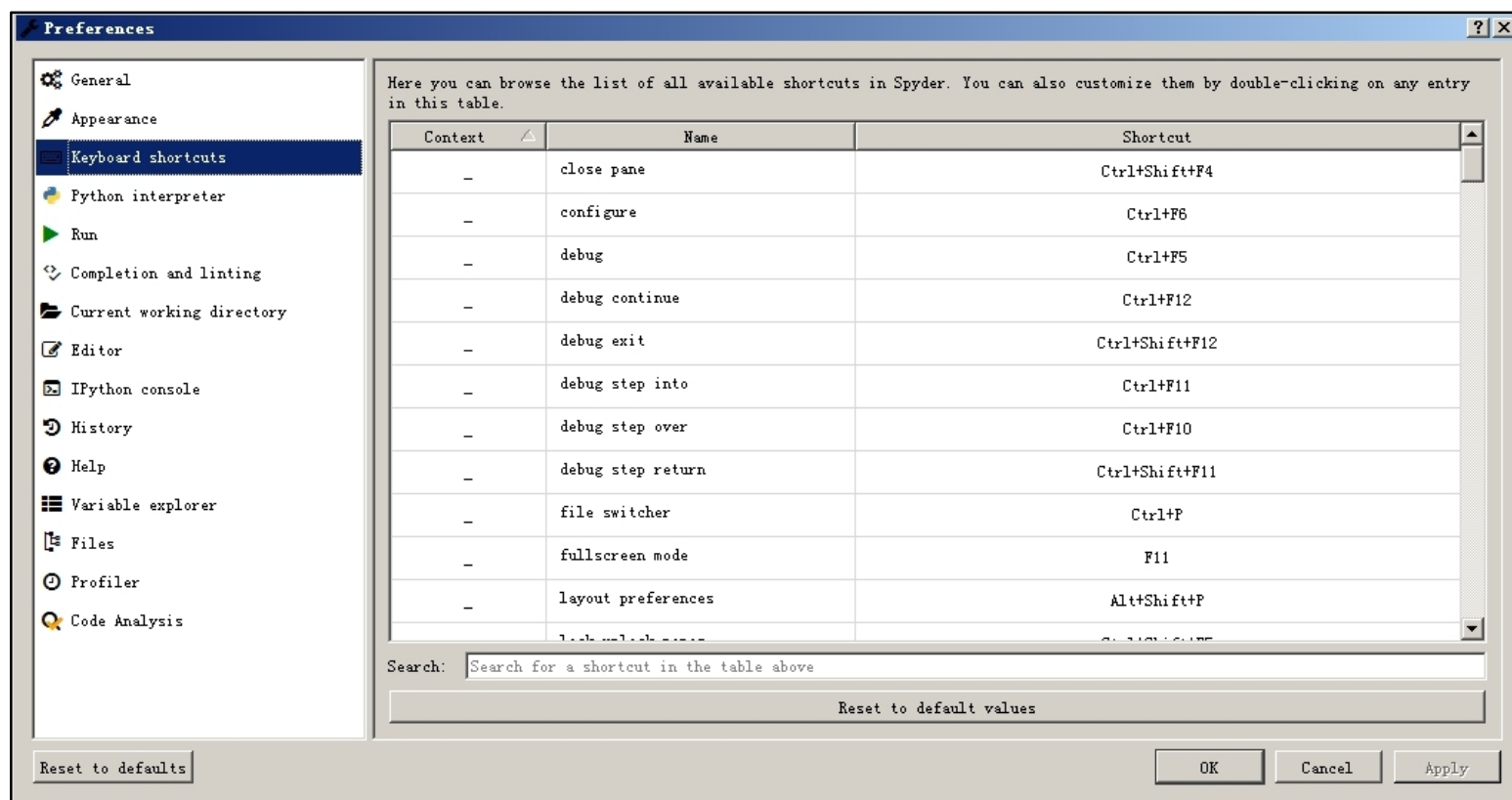


图 2.9 Spyder 的快捷键设置

用鼠标双击其中的某行快捷键设置，还可修改其快捷键。

如要存储 Python 脚本(Python Script), 可点击菜单栏中的“存盘”图标(参见图 2.10), 或在键盘同时按下 “Ctrl + S”, 将其存成扩展名为 “py” 的 Python 脚本文件, 比如 “abc.py”。



图 2.10 存储 Python 脚本文件

在图 2.10 中, 在“存盘”图标的左边, 分别为“建立新文件” (New file) 与“打开文件” (Open file) 的图标。

- * 本章其余内容详见教材, 以及配套 Python 程序 (现场演示)。