

Working in the Console 控制台中的工作

Overview 概述

The RStudio console includes a variety of features intended to make working with R more productive and straightforward. This article reviews these features. Learning to use these features along with the related features available in the [Source](#) and [History](#) panes can have a substantial payoff in your overall productivity with R.

RStudio 控制台包含许多使 R 工作更为有效和直观的特征。本文综述这些特征。学习使用这些特征以及 [Source](#) 和 [History](#) 窗口中可用的相关特征可以对你使用 R 的效率有实际性的帮助。

Code Completion 代码补全

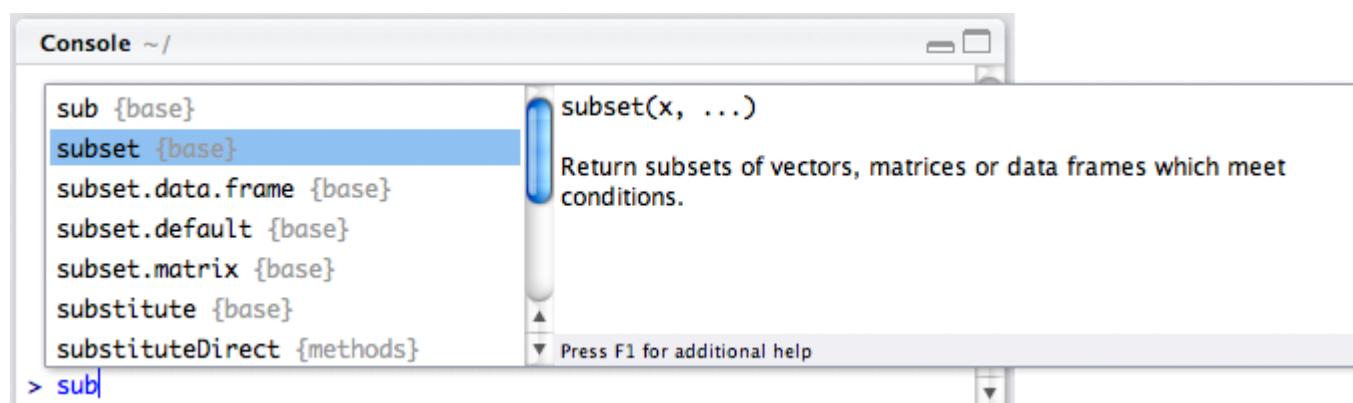
RStudio supports the automatic completion of code using the **Tab** key. For example, if you have an object named `pollResults` in your workspace you can type `poll` and then **Tab** and RStudio will automatically complete the full name of the object.

TAB补全代码

RStudio 支持使用 **Tab** 键来自动补全代码。例如，如果你在工作空间中有名为 `pollResults` 的对象，你可输入 `poll` 然后按 **Tab** 键，RStudio 将自动完成该对象的全名。

The code completion feature also provides inline help for functions whenever possible. For example, if you typed `sub` then pressed **Tab** you would see:

代码补全特征同样提供了尽可能的嵌入式帮助。例如，如果你可输入 `sub` 然后按 **Tab** 键，你将看到：



Code completion also works for function arguments, so if you typed `subset(` and then pressed **Tab** you'd see the following:

代码补全还可以对参数功能进行工作，如果你可输入 `subset`(然后按 **Tab** 键，你将看到：



Retrieving Previous Commands 检索以前命令

As you work with R you'll often want to re-execute a command which you previously entered. As with the standard R console, the RStudio console supports the ability to recall previous commands using the arrow keys:

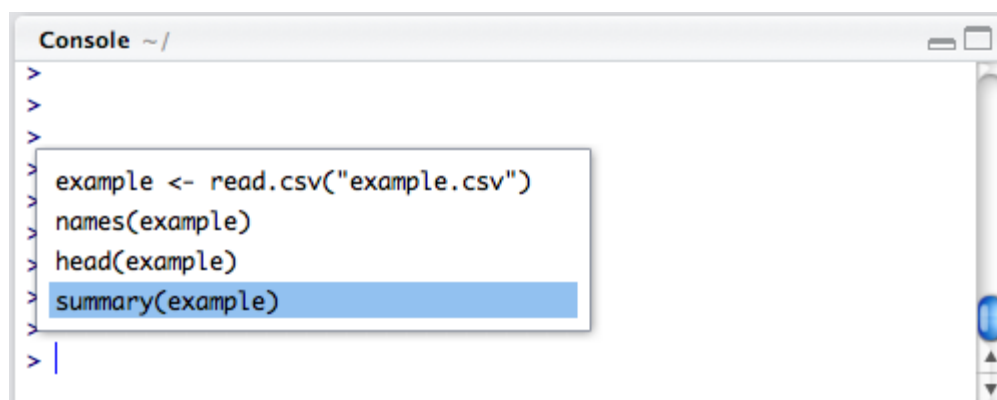
当你使用 **R** 进行工作，你将经常可输入重新执行你之前输入的命令。与标准的 **R** 控制台一样，**RStudio** 控制台支持使用以下方向键回忆之前命令的功能：

- **Up** — Recall previous command(s)
- **Down** — Reverse of Up
- **Up** —回忆前一条命令；
- **Down** —回 **Up** 相反；

If you wish to review a list of your recent commands and then select a command from this list you can use **Ctrl+Up** to review the list (note that on the Mac you can also use **Command+Up**):

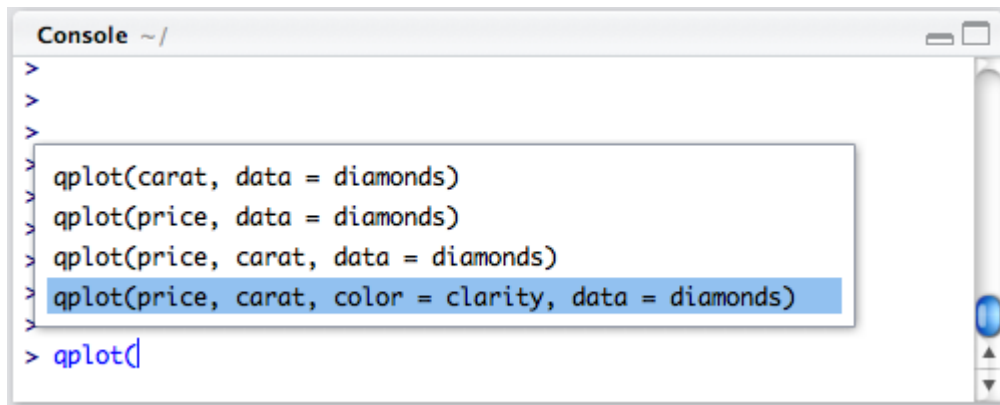
[CTRL up回顾](#)

如果你希望回顾你当前的命令列表并从中选择一条命令，则可使用 **Ctrl+Up** 来回顾命令列表（注意：在 **Mac** 你还可使用 **Command+Up**）



You can also use this same **keyboard shortcut** to quickly search for commands that match a given prefix. For example, to search for previous instances of the **plot** function simply type **plot** and then **Ctrl+Up**:

你还可以使用同样的快捷键来快速查找匹配给定前缀的命令。例如，查找前面出现过的 **plot** 函数，先输入 **plot** 然后 **Ctrl+Up**

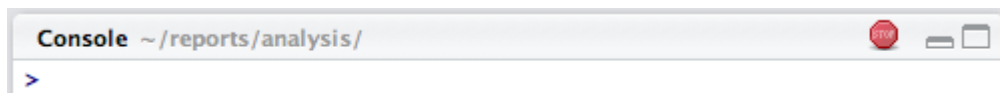


Console Title Bar 控制台标题栏

This screenshot illustrates a few additional capabilities provided by the Console title bar:

这个截屏阐明了控制台标题栏的一些额外功能。

- Display of the current working directory.
- The ability to interrupt R during a long computation.
- Minimizing and maximizing the Console in relation to the Source pane (using the buttons at the top-right or by double-clicking the title bar).
- 显示当前工作目录；
- 能够在长期的计算里中断 **R**；
- 最小化和最大化 Console 和 Source 窗口（使用右上按钮或双击标题栏）



Keyboard Shortcuts 快捷键

Beyond the history and code-completion oriented keyboard shortcuts described above, there are a wide variety of other shortcuts available. Some of the more useful shortcuts include:

除了上述的历史和代码补全快捷键外，还有很多其他快捷键可用，其中最为有用的快捷键包括

- **Ctrl+1** — Move focus to the Source Editor
- **Ctrl+2** — Move focus to the Console
- **Ctrl+L** — Clear the Console
- **Esc** — Interrupt R
- **Ctrl+1** —移动焦点到 Source 编辑器。
- **Ctrl+2** —移动焦点到 Console。
- **Ctrl+L** —清理控制台；
- **Esc** —中断 R

You can find a list of all shortcuts in the [Keyboard Shortcuts](#) article.

你可从 [Keyboard Shortcuts](#) 文中找到所有快捷键列表。

Related Topics 相关主题

- [Editing and Executing Code](#) 编辑和执行代码
- [Using Command History](#) 使用命令历史

Editing and Executing Code 编辑执行代码

Overview 概述

RStudio's source editor includes a variety of productivity enhancing features including syntax highlighting, code completion, multiple-file editing, and find/replace.

RStudio 的 **source** 编辑器包含各种提高效率的特征，包括[语法高亮显示](#)，[代码自动补全](#)，[多文件编辑](#)以及查找和替换。

RStudio also enables you to flexibly execute R code directly from the source editor. For many R developers this represents their preferred way of working with R. By executing commands from within the source editor rather than the console it is much easier to reproduce sequences of commands as well as package them for re-use as a function. These features are described in the [Executing Code](#) section below.

RStudio 还可以使你直接通过 **source** 编辑器灵活地执行 R 代码。对于许多 R 开发者来说，这是他们使用 R 的首选方式。通过 **source** 编辑器执行命令相对于控制台来说更便于复制命令序列和将其做为再次使用的函数进行打包；这些特征将在后面的执行代码部分讲解。

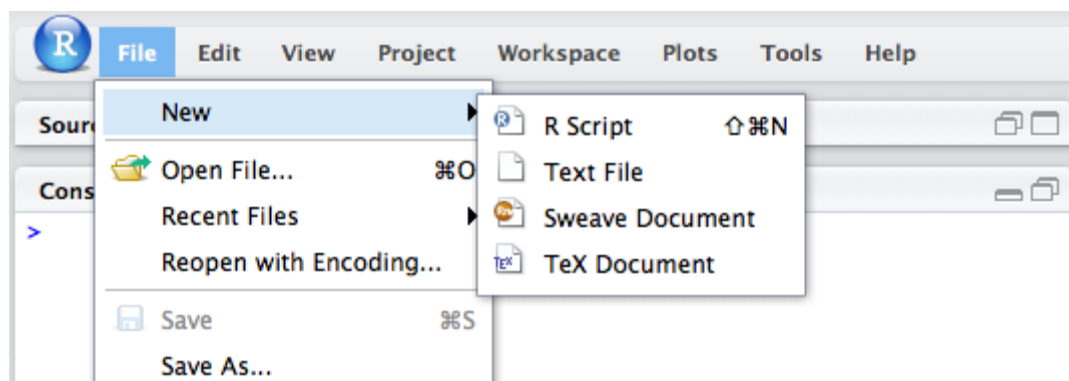
Managing Files 文件管理

RStudio supports syntax highlighting and other specialized code-editing features for the following types of files:

Rstudio 支持语法高亮显示和其他专业化的代码编辑功能，针对以下类型文件。

- R scripts **R** 脚本文件
- Sweave documents **Sweave** 文件
- TeX documents **Tex** 文件

To create a new file you use the **File -> New** menu: 你可通过 **File -> New** 菜单创建新文件。（你也可以使用 **Ctrl+Shift+N** 快捷键）。

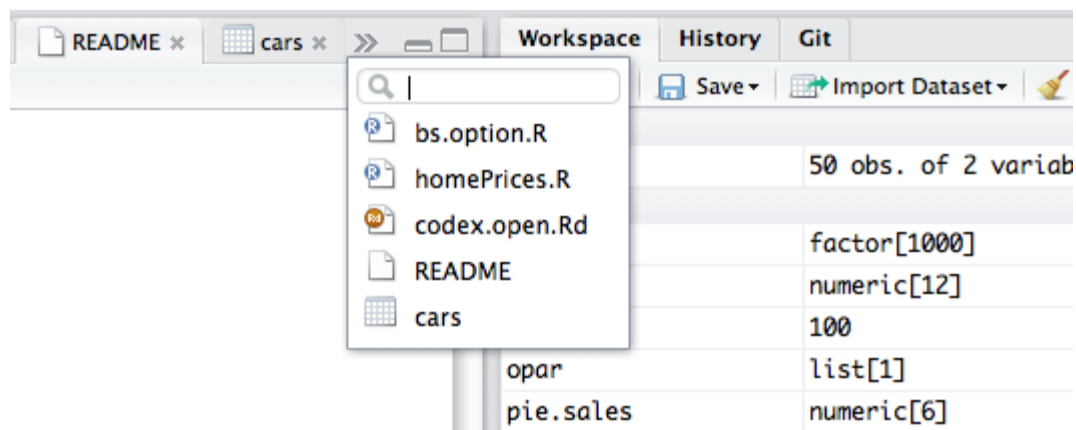


To open an existing file you use either the **File -> Open** menu or the **Open Recent** menu to select from recently opened files.

你可通过 **File -> Open** 菜单或者 **Open Recent** 菜单选择来打开已有文件。(你也可使用 **Ctrl+O** 快捷键)。

If you open several files within RStudio they are all available as tabs to facilitate quick switching between open documents. If you have a large number of open documents you can also navigate between them using the **>>** icon on the tab bar or the **View -> Switch to Tab** menu item:

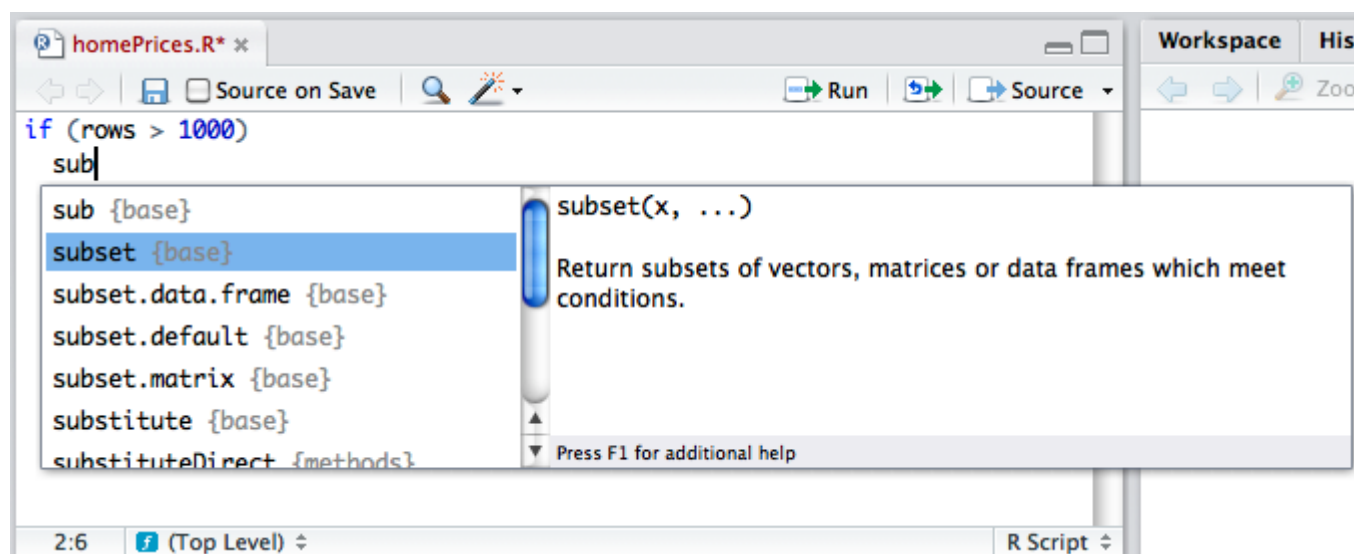
如果你通过 **RStudio** 打开许多文件，那么他们都可以通过标签进行快速切换。如果你有大量的打开文件，你也可在它们间通过标签栏中的>>图标来进行导航，或者 **View -> Switch to Tab** 菜单项 (你也可使用 **Ctrl+O** 快捷键)。



Code Completion 代码补全

RStudio supports the automatic completion of code using the **Tab** key. For example, if you have an object named **pollResults** in your workspace you can type **poll** and then **Tab** and RStudio will automatically complete the full name of the object.

RStudio 可以使用 **Tab** 键来支持代码自动补全，例如，如果你在工作空间中有名为 **pollResults** 的对象，你可输入 **poll** 然后按 **Tab** 键，RStudio 将自动完成该对象的全名。

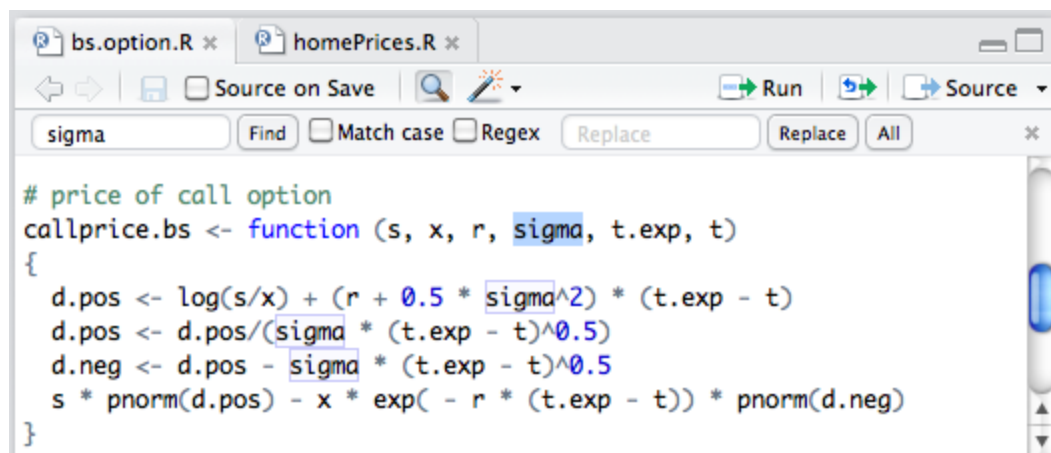


Code completion also works in the console, and more details on using it can be found the console [Code Completion](#) documentation.

代码补全同样在控制台中工作，具体用法可参阅控制台代码自动完成 [Code Completion](#) 文档。

Find and Replace 查找和替换

RStudio supports finding and replacing text within source documents: **Rstudio** 支持在 source 文件中查找和替换。



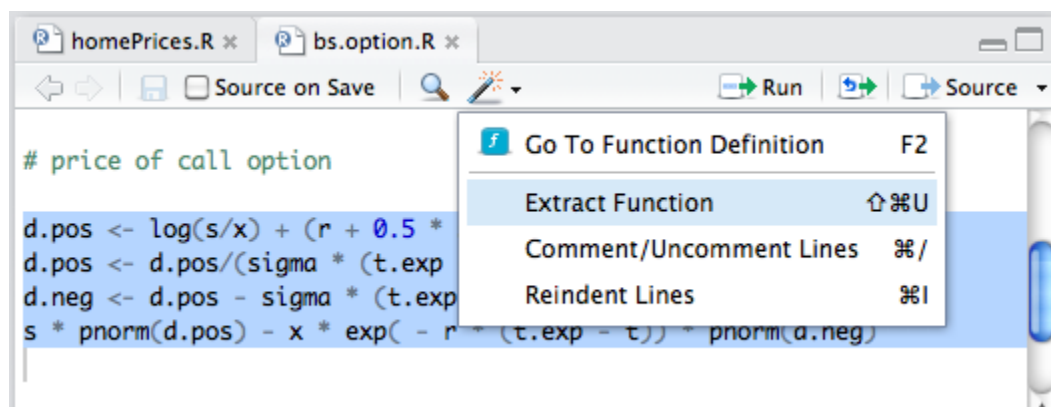
Find and replace can be opened using the **Ctrl+F** shortcut key, or from the **Edit -> Find and Replace** menu item.

可使用 **Ctrl+F** 快捷键来打开查找和替换栏，或者使用 **Edit -> Find and Replace** 菜单项。

Extract Function 提取函数

RStudio can analyze a selection of code from within the source editor and automatically convert it into a re-usable function. Any "free" variables within the selection (objects that are referenced but not created within the selection) are converted into function arguments:

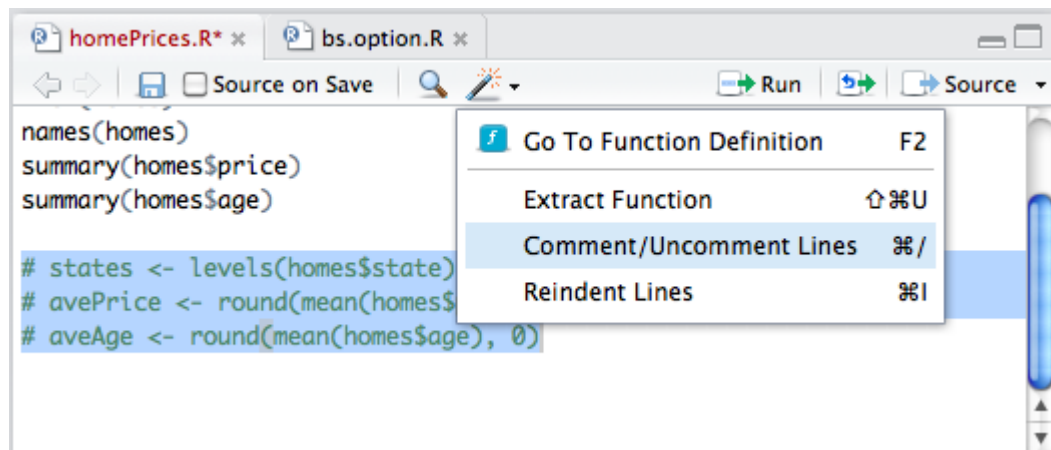
RStudio 可以在 **source** 编辑器中分析一组选择的代码，并自动将其转化成再次使用的函数。任何选择中的"free"变量（选择引用对象但不创建）将转化为函数参数。（你也可使用 **Ctrl+Shift+U** 快捷键）。



Comment/Uncomment 注释/取消注释

You can comment and uncomment entire selections of code using the **Edit -> Comment/Uncomment Lines** menu item (you can also do this using the **Ctrl+/** keyboard shortcut):

你可使用使用 **Edit -> Comment/Uncomment Lines** 菜单项来对所选的整体代码进行注释或取消注释（你也可使用 **Ctrl+Shift+C** 快捷键）。



Indentation 首行缩进

As you write R code in RStudio it is automatically indented according to the current indentation options (see [Customizing RStudio](#)). R code is also re-indented:

如果你在 **RStudio** 中写 **R** 代码，他将自动根据当前的缩进选项（见 [Customizing RStudio](#)）进行缩进。**R** 代码也可再缩进。

1. Whenever new code is pasted into a source document.
2. When the Reindent Lines command (pictured above) is invoked.

当新代码黏贴到一个 **source** 文件中；

当缩进行命令（上图）被调用

Note that RStudio automatic indentation is R syntax-aware and is therefore only used on source files containing R code.

注意：**RStudio** 自动缩进针对的是 **R** 语法意识，因此，只有在包含 **R** 代码的 **source** 文件中使用

Executing Code 执行代码

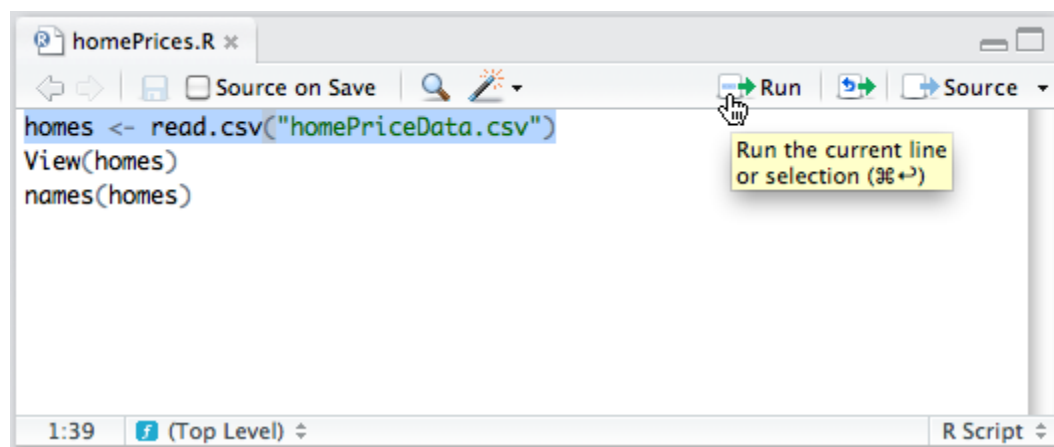
RStudio supports the direct execution of code from within the source editor (the executed commands are inserted into the console where their output also appears).

RStudio 支持从 **source** 编辑器中直接执行代码（执行代码将插入控制台，并在此输出结果），

Executing a Single Line 执行一行代码

To execute the line of source code where the cursor currently resides you press the **Ctrl+Enter** key (or use the **Run** toolbar button):

你可通过 **Ctrl+Enter** 键来执行当前光标所在行的 **source** 代码(或者使用 **Run** 工具条按钮)



After executing the line of code, RStudio automatically advances the cursor to the next line. This enables you to single-step through a sequence of lines.

执行完一行代码，**RStudio** 自动将光标跳到下一行，这将使你可以在一系列行中单步执行代码。

Executing Multiple Lines 执行多行代码

There are three ways to execute multiple lines from within the editor:

有三种方法从编辑器中执行多行代码。

- Select the lines and press the **Ctrl+Enter** key (or use the **Run** toolbar button); or
- After executing a selection of code, use the **Re-Run Previous Region** command (or its associated toolbar button) to run the same selection again. Note that changes to the selection including additional, removal, and modification of lines will be reflected in this subsequent run of the selection.
- To run the entire document press the **Ctrl+Shift+Enter** key (or use the **Source** toolbar button).
- 选择这些行，按 **Ctrl+Enter** 键（或者使用 **Run** 工具条按钮）

- 执行选择的代码后，使用 **Re-Run Previous Region** 命令（或相应的工具栏按钮）再次运行相同的选择。注意：现在发生变化，包括增增加、删除和修改行将反映到后续运行中。（或者使用 **Ctrl+Shift+P** 快捷键）
- 运行整个文件按 **Ctrl+Shift+Enter** 键（或使用 **source** 工具条按钮）??

The difference between running lines from a selection and invoking **Source** is that when running a selection all lines are inserted directly into the console whereas for **Source** the file is saved to a temporary location and then sourced into the console from there (thereby creating less clutter in the console).

从选择的行进行运行和调用 **Source** 的区别在于，当运行所选的所有行将直接插入到控制台中，而对于 **Source** 文件则保存到临时位置然后从那里反映到控制台（因此在控制台中产生较少的杂乱）。

Source on Save 资源保存

When editing re-usable functions (as opposed to freestanding lines of R) you may wish to set the **Source on Save** option for the document (available on the toolbar next to the Save icon). Enabling this option will cause the file to automatically be sourced into the global environment every time it is saved.

当编辑一个再次调用函数（相对于独立的 **R** 行而言），你可能希望对文件设置 **Source on Save** 选项（通过工具条中 Save 图标的一项实现）。启用该选项导致文件每次保存都将被全球自动采用。

Setting **Source on Save** ensures that the copy of a function within the global environment is always in sync with its source, and also provides a good way to arrange for frequent syntax checking as you develop a function.

设置 **Source on Save** 选项确保该函数的版本在全球环境总是能够资源同步，并且也能在你开发一个函数是提供一种好的方式安排频繁的语法检查。

Keyboard Shortcuts 快捷键

Beyond the keyboard shortcuts described above, there are a wide variety of other shortcuts available. Some of the more useful ones include:

除了上述的快捷键外，还有很多其他快捷键可用，其中最为有用的快捷键包括。

- **Ctrl+Shift+N** — New document 新文档
- **Ctrl+O** — Open document 打开文档
- **Ctrl+S** — Save active document 保存活动文档
- **Ctrl+1** — Move focus to the Source Editor 移动焦点到 Source 编辑器
- **Ctrl+2** — Move focus to the Console 移动焦点到控制台

You can find a list of all shortcuts in the [Keyboard Shortcuts](#) article.

Related Topics 相关主题

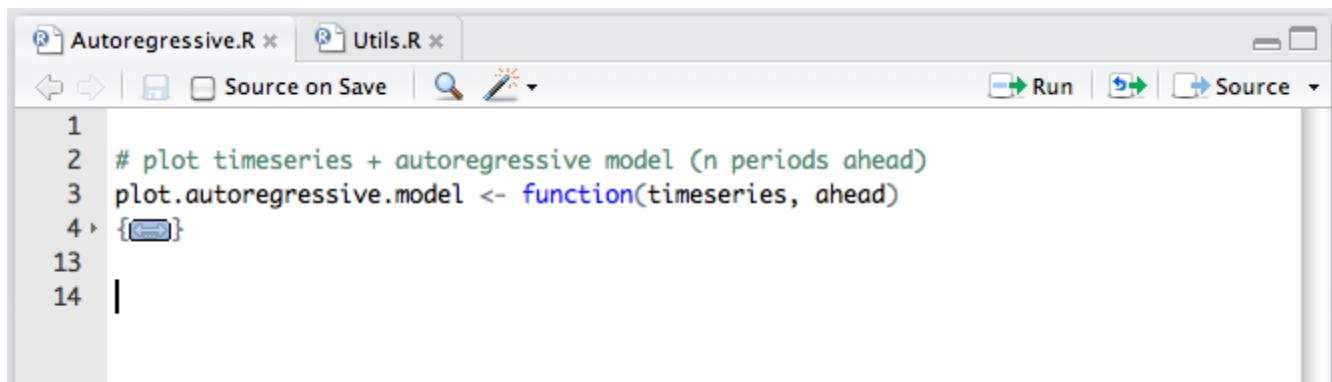
- [Working in the Console](#) 在控制台中工作
- [Using Command History](#) 使用历史命令
- [Navigating Code](#) 代码导航
- [Code Folding and Sections](#) 代码折叠和截取

Code Folding and Sections 代码折叠和截取

Code Folding 代码折叠

RStudio supports both automatic and user-defined folding for regions of code. Code folding allows you to easily show and hide blocks of code to make it easier to navigate your source file and focus on the coding task at hand. For example, in the following source file the body of the `plot.autoregressive.model` has been folded:

RStudio 支持自动和使用者定义的地区代码折叠。代码折叠允许你很方便地显示和隐藏代码块，这将使你的 `source` 文件导航更简易，并可以将中重点放在手头的代码任务上。例如，在下面的 `source` 文件中 `plot.autoregressive.model` 部分已经被折叠。



You can expand the folded region by either clicking on the arrow in the gutter or on the icon that overlays the folded code.

你可展开折叠部分，通过点击箭头槽或者覆盖折叠代码的图标。

Foldable Regions 折叠区域

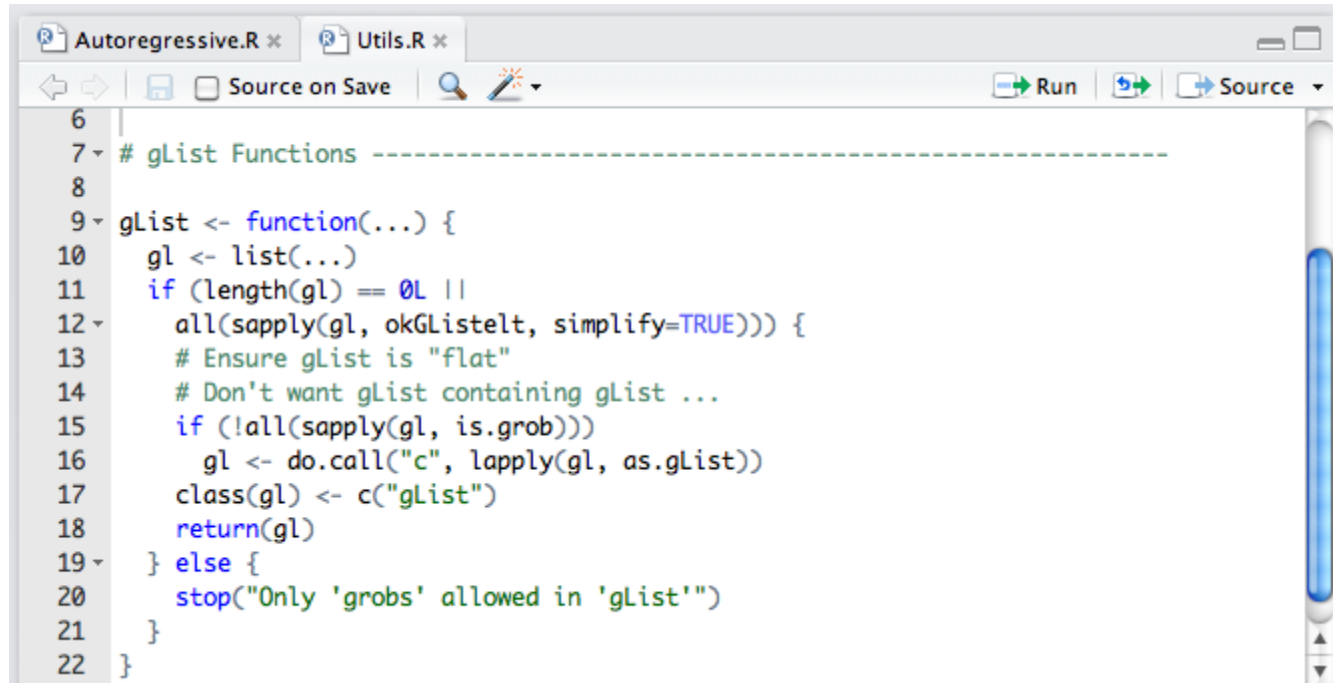
The following types of code regions are automatically foldable within RStudio:

RStudio 中以下类型的代码区域将自动折叠：

- Braced regions (function definitions, conditional blocks, etc.)
- Code chunks within R Sweave or R Markdown documents
- Code sections (see below for details)
- 支撑区域（函数定义、条件块等）
- R Sweave 的代码块或者是 R 的 Markdown 文件。
- 代码段（具体见后文）

In the following example you can see that the top-level code section, function body, and conditional blocks are all foldable:

在以下案例中，即将看到顶级代码部分、函数体和条件块都被折叠。



You can also fold an arbitrary selection of code by using **Edit -> Folding -> Collapse (Alt-L)**.

你也可折叠任意选定的代码，通过 **Edit -> Folding -> Collapse**，或者使用 **Alt-L** 快捷键。

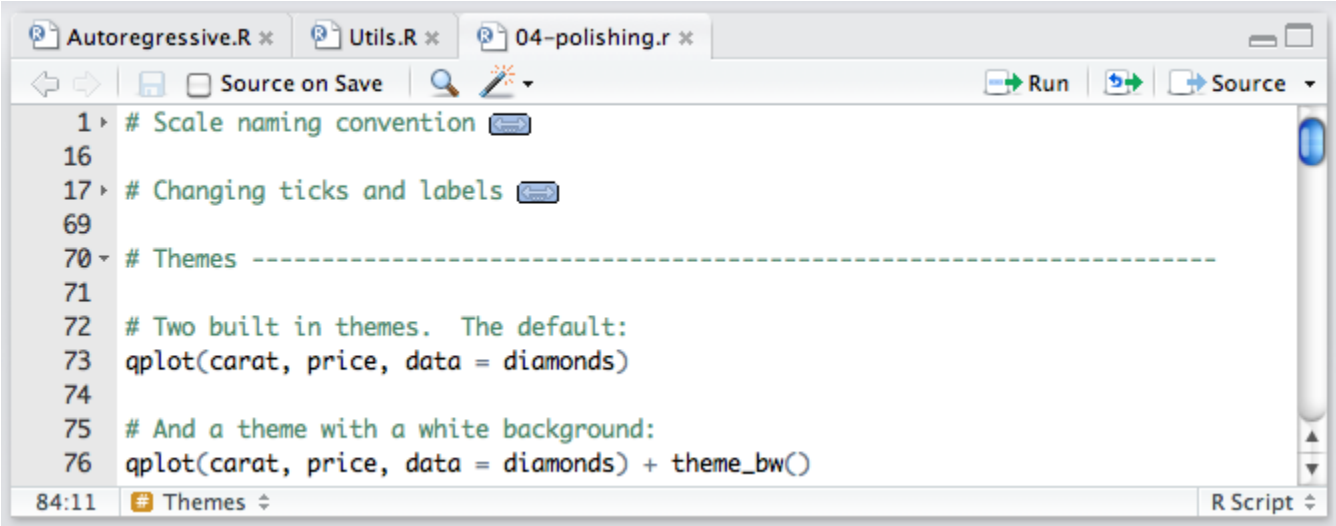
Folded regions are preserved while editing a document however when a file is closed and re-opened all foldable regions are by default shown expanded.

折叠部分在编辑文件时将被保证，但是当文件关闭或者再次打开时，所有折叠部分将默认展开显示。

Code Sections 代码段

Code sections allow you to break a larger source file into a set of discrete regions for easy navigation between them. Code sections are automatically foldable—for example, the following source file has three sections (one expanded and the other two folded):

代码段允许你将一个大型的 **source** 文件分解成一组独立的区域以方便再他们之间漫游。代码段将自动折叠，例如，以下源文件有三个部分（一个展开而另两个折叠）。



```
1 ▸ # Scale naming convention
16
17 ▸ # Changing ticks and labels
69
70 ▸ # Themes -----
71
72 # Two built in themes. The default:
73 qplot(carat, price, data = diamonds)
74
75 # And a theme with a white background:
76 qplot(carat, price, data = diamonds) + theme_bw()
```

84:11 # Themes R Script

To insert a new code section you can use the **Code -> Insert Section** command. Alternatively, any comment line which includes at least four trailing dashes (-), equal signs (=), or pound signs (#) automatically creates a code section. For example, all of the following lines create code sections:

插入一个新的代码段你使用 **Code -> Insert Section** 命令。或者，任意注释行中包含至少四个破折号（-），等号（=）或井号（#）将自动创建代码段。

```
# Section One -----

# Section Two =====

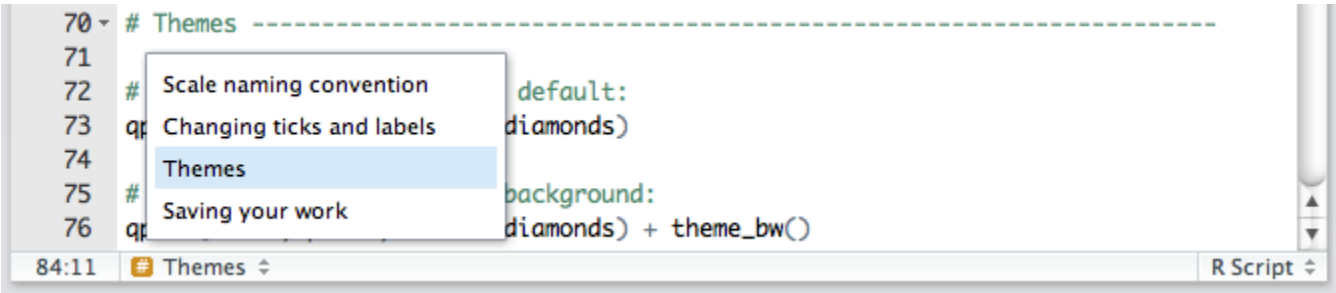
### Section Three #####
```

Note that as illustrated above the line can start any number of pound signs (#) so long as it ends with four or more -, =, or # characters.

注意：如前所述，可输入超过 4 个或更多的数量的#号，-，=等。

To navigate between code sections you can use the **Jump To** menu available at the bottom of the editor:

你可使用编辑器低端的 **Jump To** 菜单在代码段间漫游（或者使用 **Alt+Shift+J** 快捷键）。



```
70 ▸ # Themes -----
71
72 # Scale naming convention default:
73 q Changing ticks and labels diamonds)
74
75 # Themes
76 q Saving your work background:
  diamonds) + theme_bw()
```

84:11 # Themes R Script

Menu Commands and Shortcuts 菜单命令和快捷键

The following menu commands and shortcuts are available for working with folded regions and code sections:

以下菜单命令和快捷键可以对折叠区域和代码段进行操作：

- **Edit -> Folding:**
 - **Collapse** — Alt+L
 - **Expand** — Shift+Alt+L
 - **Collapse All** — Alt+A
 - **Expand All** — Shift+Alt+A
- **Code:**
 - **Insert Section** — Ctrl+Shift+R (Cmd+Shift+R on the Mac)
 - **Jump To** — Shift+Alt+j

Note that the Collapse All command collapses all of the outermost foldable regions (rather than all of the nested regions within the source file).

注意：Collapse All 命令折叠所有外层可折叠区域（而不是源文件中的所有嵌套区域）

Related Topics

- [Navigating Code](#) 导航代码
- [Editing and Executing Code](#) 编辑和执行代码

Navigating Code 导航代码

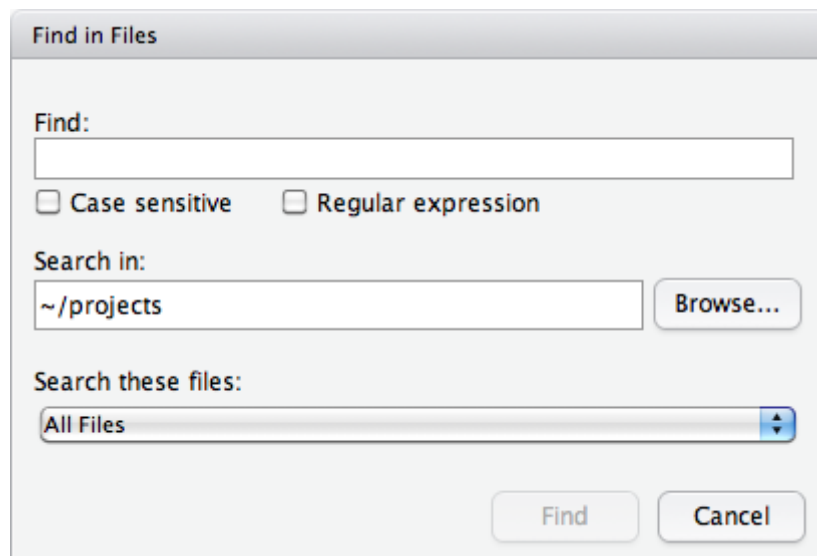
RStudio includes a number of features to enable rapid navigation through R source code. Learning these features can be a major productivity enhancement and can also assist in gaining a better understanding of source code written by others on your team or within external packages.

RStudio 包括一些在 R 源代码中加速导航的功能。学习这些功能可以提高效率并帮助更好地理解你的团队中其他人所写的源代码，或者是外部程序包中的源代码。

Find in Files 查找文件

Given a specific directory, Find in Files allows you to **recursively** search every file for each occurrence of a given string. To display the Find in Files dialog box, go to the edit menu and select **Find in Files**:

给出一个特定目录，查找文件（**Edit->Find in Files**）允许你对每个出现的字符串递归查找每一个文件。显示查找文件对话框，编辑菜单并选择查找文件：



You can further customize your search with regular expressions and filters for specific file types. Your search results will display in the pane adjacent to the console. For each matching string, the following will be displayed:

你可以进一步通过正式表达式和特定文件类型来自定义你的搜索。你的搜索结果将显示在控制台的相邻窗口中。每一个匹配的字符串都将显示以下信息：

- The file path 文件路径
- The line number of each occurrence 每个出现的行号
- The matching string (highlighted), in the context of the current line

- 当前行内容中匹配的字符串（高亮）

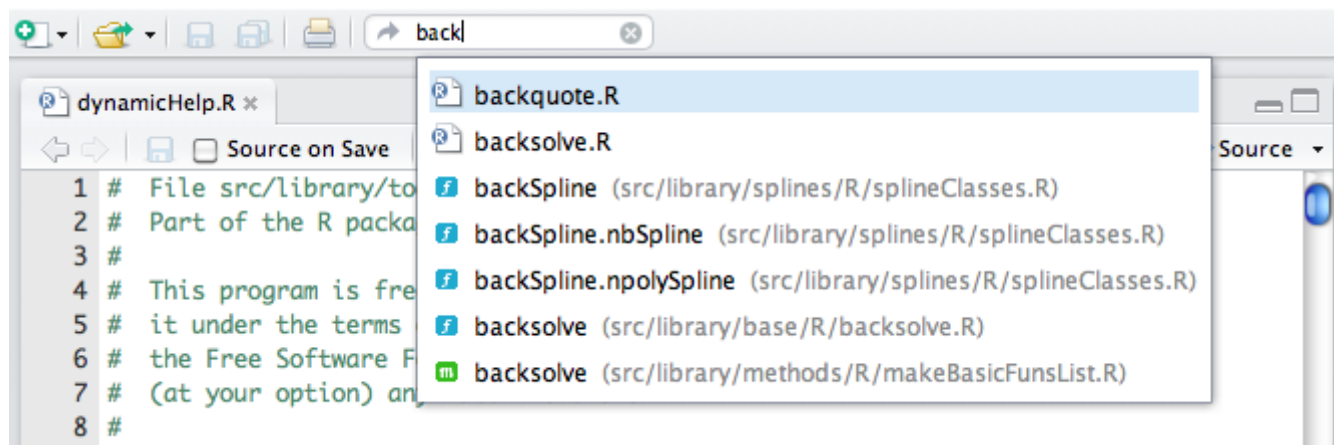
Double clicking the line, will open the file in the RStudio source editor. The keyboard shortcut for Find in Files is **Ctrl+Shift+F**

双击这一行，将在 **RStudio** 的 **source** 编辑器中打开该文件。查找文件的快捷键是 **Ctrl+Shift+F**。

Go to File/Function 转到文件/函数

If you know the name of the source file or function that you want to edit next you can quickly navigate to it using the Go to File/Function search box on the main RStudio toolbar:

如果你知道源文件或函数的名字并想对其进行编辑，你可以使用 **RStudio** 工具条中的 Go to File/Function 搜索框迅速导航。



The Go to File/Function feature works off a constantly updated index of your source code. The specific source files to index are determined as follows:

Go to File/Function 功能就不断更新索引你的源代码。专业源文件的索引将决定于：

1. If an **RStudio Project** is active then all R source files within the project directory are indexed.
2. If a Project is not active then all currently open R source files are indexed.

1. **RStudio Project** 是活动的，然后所有的 **R** 源文件在程序包列表中将被索引；

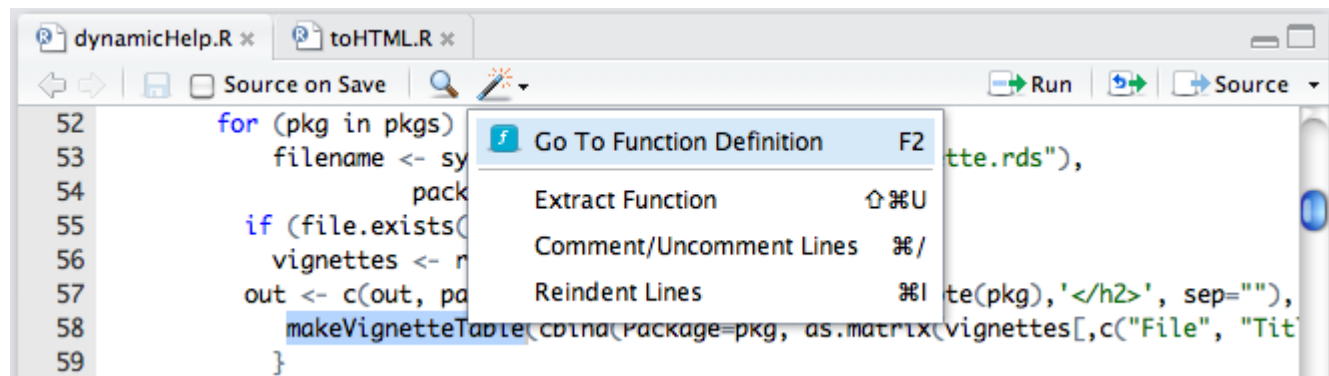
2. 如果一个 Project 不活动，那么所有当前打开的 **R** 源程序将被索引；

The keyboard shortcut for Go to File/Function is **Ctrl+.**. Go to File/Function 的快捷键是 **Ctrl+.**

Go to Function Definition 转到函数定义

Since an index of your R source code (as described above) is maintained, RStudio can also help you quickly navigate to definition of any R function. To navigate a function definition you place your cursor on the function name (it doesn't have to fully selected) and then choose the **Go to Function Definition** command:

当一个你的 R 源代码指数（如上所述）被维护，RStudio 也可帮助你迅速对任何 R 函数进行导航。你可将光标放在函数名称上（不需要完全选定）来导航函数定义，然后选择 **Go to Function Definition** 命令：



You can also access go to function definition:

你也可这样去访问函数定义：

- Using the F2 keyboard shortcut 使用 F2 快捷键
- Using **Ctrl+Click** with the mouse 使用 **Ctrl+Click** 鼠标点击
- From either the source editor or the console 从 **source** 编辑器或控制台

It is possible to navigate to both your own functions (defined in R source files) as well as any other function defined within an R package. For functions defined within packages the code is displayed in a special **Source Viewer** pane which is read-only.

它同样能够导航你自己的函数（在 R 源文件中定义）和其他所有在 R 程序包中定义的函数。程序包中定义的函数，其代码将显示在特定的 **Source Viewer** 窗口中，它仅能只读不能编辑，但可以复制粘贴出来使用。

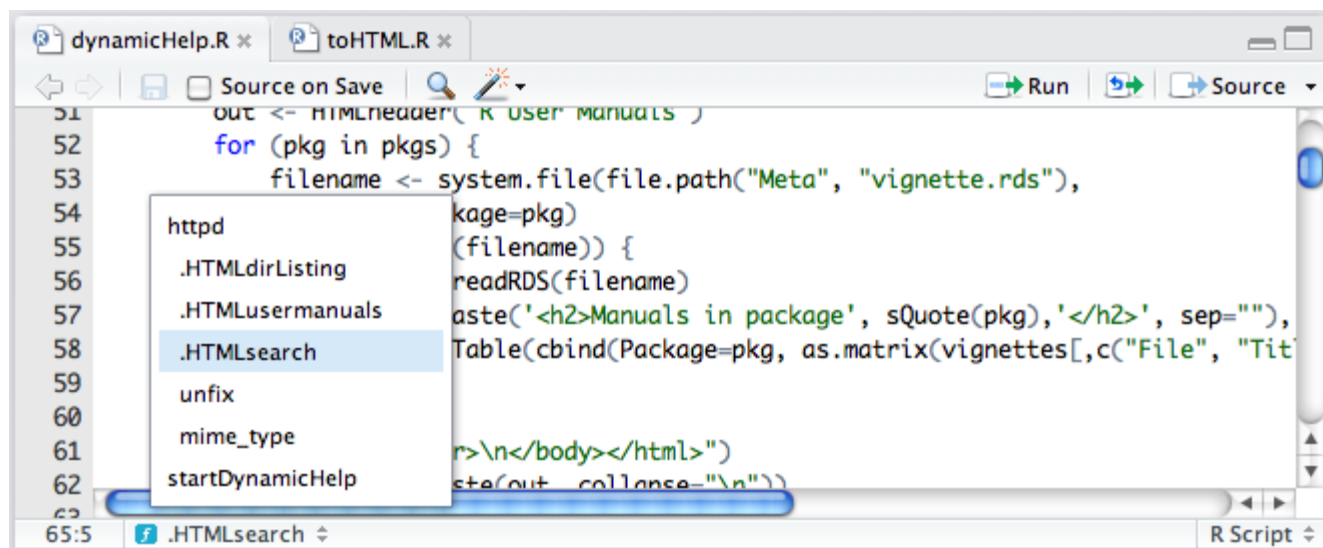
Navigating within a Source File 源文件导航

There are several ways to navigate through the code within a source file:

有多种方式可以对源文件中的代码进行导航：

- The go to function definition feature described above also works for functions within the current source file (giving **precedence** to functions defined within the calling scope).

- The **Jump to Line** command (shortcut: **Command+G**) can be used to go to any line in the current file.
- You can also use the function menu (shown below) to quickly jump to functions by name.
- 上述的 **go to function definition** 功能同样可以对当前源文件的函数进行工作（在调用范围中给出函数定义优先级。
- **Jump to Line** 命令（快捷键 **Command+G**）可用于转到当前文件的任意一行；
- 你也可使用函数命令（如下显示），通过名称迅速跳到函数。



Note that the function menu currently supports standard R functions however does not yet support S4 methods.

注意：当前函数菜单支持标准 R 函数，但是不支持 S4 方法。

Going Back and Forward 向前与向后

When navigating through code (especially when navigating through a sequence of function calls) you often want to quickly return to the previous editing location. RStudio maintains a list of active navigations and allows you to traverse them using the **Back** and **Forward** commands (available on the Edit menu and on the far left of the source editor toolbar).

当代码导航（特别是在一系列函数中导航），你经常要快速返回之前编辑的位置。RStudio 维护有一个活动的导航列表，允许你使用 **Back** 和 **Forward** 命令（**Edit** 菜单中或在 **source** 编辑条的最左边）进行来回遍历。

Back and **Forward** apply to the following navigation **gestures**:

Back 和 **Forward** 适用于以下导航方式：

- Opening a document (or switching tabs) 打开一个文件（或切换标签）
- Going to a function definition 转到函数定义
- Jumping to a line 跳到一行
- Jumping to a function using the function menu 使用函数菜单跳到一个函数

You can also invoke **Back** and **Forward** using the Ctrl+F9/Ctrl+F10 (Cmd+F9/Cmd+F10 on the Mac) keyboard shortcuts.

你同样可以使用快捷键 **Ctrl+F9/Ctrl+F10** (Cmd+F9/Cmd+F10 on the Mac)来调用 **Back** 和 **Forward**:

Using Projects 使用项目

RStudio projects make it straightforward to divide your work into multiple contexts, each with their own working directory, workspace, history, and source documents.

RStudio 项目直接让你的工作多语境，每一个自带工作目录、工作空间、历史和源文件。

Creating Projects 创建项目

RStudio projects are associated with R working directories. You can create an RStudio project:

RStudio 项目与 R 工作目录相连，你可通过以下方式创建 RStudio 项目：

- In a brand new directory
- In an existing directory where you already have R code and data
- By cloning a version control (Git or Subversion) repository
- 在全新的目录下；
- 在现有的你已存放有 R 代码和数据的目录下；
- 通过克隆一个版本控制（Git 或 Subversion）元数据库；

To create a new project use the **New Project** command (available on the Projects menu **and o**

可使用创建 **New Project** 命令（在 Projects 菜单中）一个新项目。

Using Command History 使用命令历史

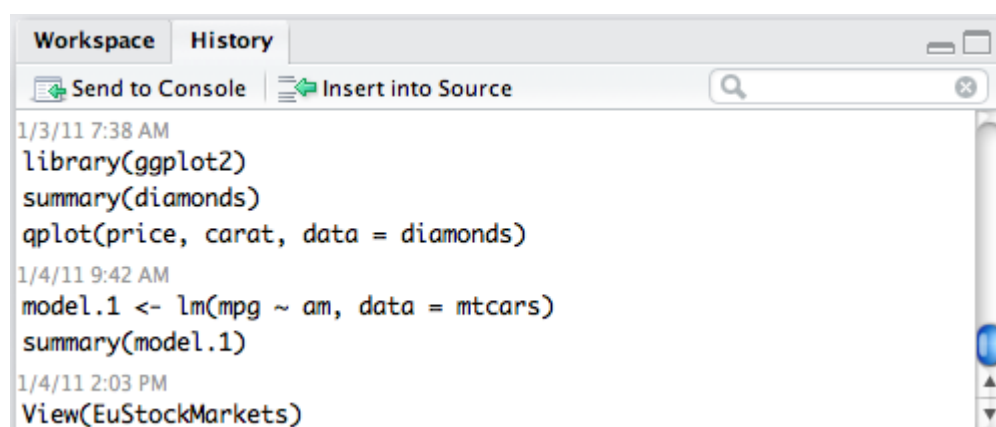
RStudio maintains a database of all commands which you have ever entered into the Console. You can browse and search this database using the History pane.

RStudio 维护有所有你曾输入控制台的命令的数据库，你可使用历史窗口来浏览和搜索这个数据库。

Browsing History 浏览历史

Commands you have previously entered in the RStudio console can be browsed from the History tab. The commands are displayed in order (most recent at the bottom) and grouped by block of time:

你之前输入 RStudio 控制台的命令可以通过历史标签进行浏览，这些命令将按顺序显示出来（最近的命令在底部）并按时间分组。

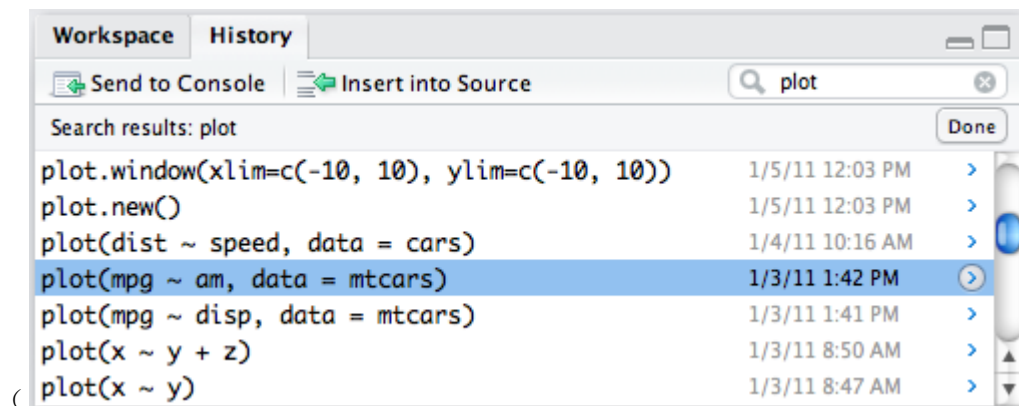


Searching History 搜索历史

Executing a Search 执行一个搜索

You can use the search box at the top right of the history tab to search for all instances of a previous command (e.g. `plot`). The search can be further refined by adding additional words separated by spaces (e.g. the name of particular dataset):

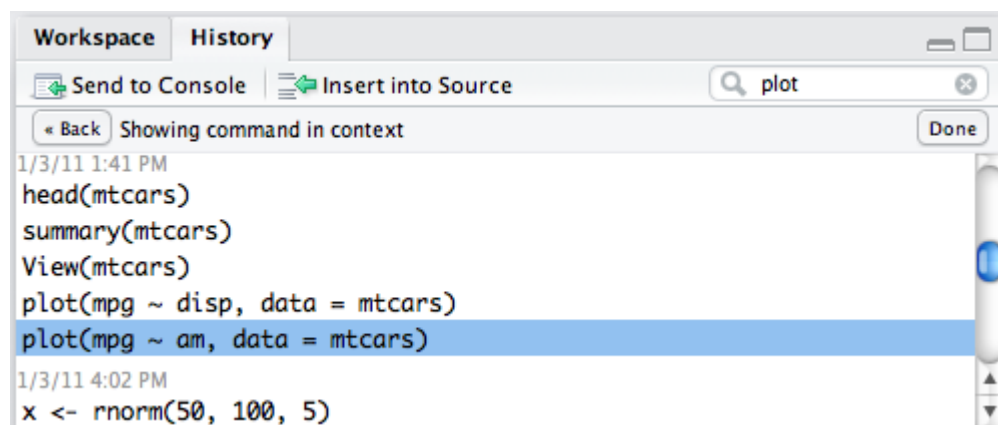
你可使用历史窗口右上角的搜索栏搜索之前命令的所有实例（例如 **plot**）。搜索可进一步根据添加的用空格隔开的单词来进行再次查找（例如一个特定的数据集）。



Showing Command Context 显示命令上下文

After searching for a command within your history you may wish to view the other commands that were executed in proximity to it. By clicking the arrow in the right margin of the search results you can view the command within its context:

在你的历史中搜索完一个命令后，你可能希望看到与之接近的其他命令。点击搜索结果的右侧边缘的箭头，你将看到命令的上下文。



Using Commands 使用命令

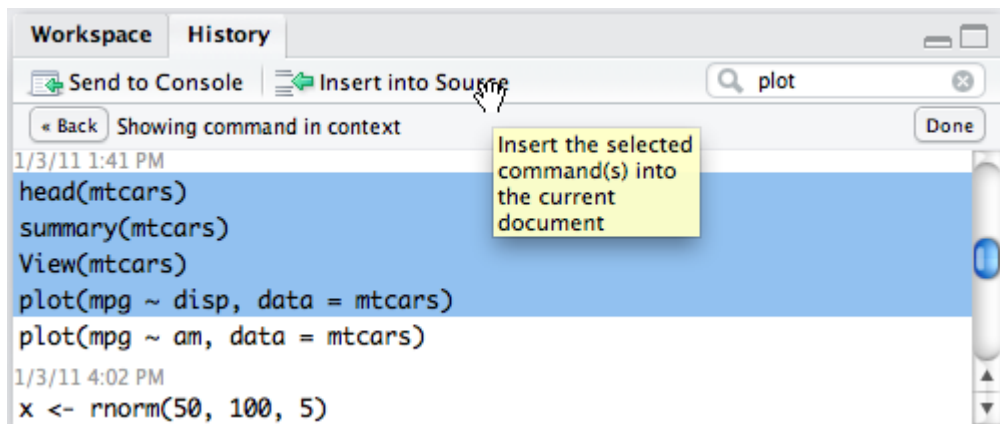
Commands selected within the History pane can be used in two fashions (corresponding to the two buttons on the left side of the History toolbar):

在历史窗口中所选的命令可以通过两种方式进行使用（对应历史工具栏左侧的两个按钮）：

- **Send to Console**— Sends the selected command(s) to the Console. Note that the commands are inserted into the Console however they are not executed until you press **Enter**.
- **To Console**—将所选命令发送到控制台。注意命令插入控制台后直到你按 **Enter** 键才会执行。
- **Insert into Source**— Inserts the selected command(s) into the currently active Source document. If there isn't currently a Source document available then a new untitled one will be created.
- **To Source**—将所选命令发送到当前活动的源文件。如果当前没有源文件则将创建一个新的未命名的源文件。

Within the history list you can select a single command or multiple commands:

历史列表中，你可选择单个命令或者多个命令（按住 **Ctrl** 键不连续多选或按住 **Shift** 键选择连续的命令段）：



Related Topics 相关主题

- [Working in the Console](#) 在控制台中工作
- [Editing and Executing Code](#) 编辑和执行代码

Working Directories and Workspaces

工作目录和工作空间

The default behavior of R for the handling of `.RData` files and workspaces encourages and facilitates a model of breaking work contexts into distinct working directories. This article describes the various features of RStudio which support this workflow.

R 对所操作的 `.RData` 文件和工作空间的默认行为支持和推动将工作环境划分为不同目录的模型。本文阐述 RStudio 支持这种工作流程的多种功能，

IMPORTANT NOTE: In version v0.95 of RStudio a new [Projects](#) feature was introduced to make managing multiple working directories more straightforward. The features described below still work however Projects are now the recommended mechanism for dealing with multiple work contexts.

重要提示：在 RStudio v0.95 中，一个新的项目功能被引入进来以使得多个工作目录的管理更直接。以下阐述的功能仍然工作，尽管项目作为推荐机制来处理多个工作环境。

Default Working Directory 默认工作目录

As with the standard R GUI, RStudio employs the notion of a global default working directory. Normally this is the user home directory (typically referenced using `~` in R). When RStudio starts up it does the following:

与标准的 R GUI 一样，RStudio 采用全局默认工作目录的概念。通常这是使用者的主目录（通常参考 R 中使用`~`）。当 RStudio 开始他将如下操作：

- Executes the `.Rprofile` (if any) from the default working directory.
- Loads the `.RData` file (if any) from the default working directory into the workspace.
- Performs the other actions described in [R Startup](#).
- 运行默认工作目录下的 `.Rprofile`；
- 装载默认工作目录下的 `.RData` 文件到工作空间中；
- 执行其他活动描述在 [R Startup](#) 中

When RStudio exits and there are changes to the workspace, a prompt asks whether these changes should be saved to the `.RData` file in the current working directory.

当 RStudio 退出和工作空间发生变化，一个提示将问及是否保存这些改变到当前工作目录的 `.RData` 文件中。

This default behavior can be customized in the following ways using the [RStudio Options](#) dialog:

可使用 [RStudio Options](#) 对话框通过以下方式来自定义默认行为:

- Change the default working directory
- Enable/disable the loading of .RData from the default working directory at startup
- Specify whether .RData is always saved, never saved, or prompted for save at exit.
- 改变默认工作目录;
- 启用/禁用开始时从默认工作目录中装载.RData;
- 指定.RData 退出时是否总是保存、不保持或者提示保存。

Changing the Working Directory 修改工作目录

The current working directory is displayed by RStudio within the title region of the Console. There are a number of ways to change the current working directory:

当前的工作目录显示在 RStudio 控制台的标题区域。有多种方法可以改变当前工作目录:

- Use the `setwd` R function
- Use the **Tools | Change Working Dir...** menu. This will also change directory location of the Files pane.
- From within the Files pane, use the **More | Set As Working Directory** menu. (Navigation within the Files pane alone will not change the working directory.)
- 使用 R 函数 `setwd`;
- 使用 **Tools | Change Working Dir...** 菜单, 他也将改变文件窗口的目录位置;
- 从文件窗口中使用 **More | Set As Working Directory** 菜单。(在文件窗口中导航将不改变工作目录)

Be careful to consider the **side effects** of changing your working directory:

考虑改变你的工作目录请小心, 它有以下副作用:

- Relative file references in your code (for datasets, source files, etc) will become invalid when you change working directories.
- The location where .RData is saved at exit will be changed to the new directory.
- 应用你代码中的相关文件 (如数据集、源文件等) 将无效, 当你改变工作文件;
- .RData 退出时所保持的位置将改变到新目录中;

Because these side effects can cause confusion and errors, it's usually best to start within the working directory associated with your project

and remain there for the duration of your session. The section below describes how to set RStudio's initial working directory.

因为这些副作用可能导致迷失和错误，最好是通常从与你项目相连的工作目录开始，并保持在那里进行你的操作。后文将阐述怎样设置 RStudio 的初始工作目录。

Starting in Other Working Directories 开始其他工作目录

If all of the files related to a project are contained within a single directory then you'll likely want to start RStudio within that directory. There are a number of ways (which vary by platform) to do this.

如果所有的文件与一个包含在单个目录下的项目相关，那么你最好从那个目录开始 RStudio。有多种方式（平台不同将有所变化）来实现这个。

File Associations 文件关联

On all platforms RStudio registers itself as a handler for .RData, .R, and other R related file types. This means that the system file browser's context-menu will show RStudio as an **Open With** choice for these files.

RStudio 的所有平台自己注册成一个处理程序如 .RData, .R 和其他 R 相关文件类型。这意味着系统文件浏览器的上下文菜单将对在 RStudio 对这些文件进行开放选择。

You can also optionally create a default association between RStudio and the .RData and/or .R file types.

你也可以有选择地创建 RStudio 和 .RData 或 .R 文件的默认关联。

When launched through a file association, RStudio automatically sets the working directory to the directory of the opened file. Note that RStudio can also open files via associations when it is already running—in this case RStudio simply opens the file and does not change the working directory.

当开始通过了一个文件关联，RStudio 将自动设置工作目录到打开文件的目录。注意 RStudio 当它已经运行也可通过连接打开文件——在这种情况下，RStudio 仅打开文件，但不改变工作目录。

Shortcuts (Windows) 快捷键

On Windows, you can create a shortcut to RStudio and customize the "Start in" field. When launched through this shortcut RStudio will startup within the specified working directory.

在 Windows 中，你可创建一个 RStudio 快捷键来自定义 "Start in" 部分。当启动这个 RStudio 快捷键，将通过特定工作目录来开始。

Drag and Drop (Mac)

On Mac, dragging and dropping a folder from the Finder on the RStudio Dock icon will cause RStudio to startup with the dropped folder as the current working directory.

Run from Terminal (Mac and Linux)

On Mac and Linux systems you can run RStudio from a terminal and specify which working directory to startup within. Additionally, on Linux systems if you run RStudio from a terminal and specify no command line argument then RStudio will startup using the current working directory of the terminal.

For example, on the Mac you could use the following commands to open RStudio (respectively) in the '~/projects/foo' directory or the current working directory:

```
$ open -a RStudio ~/projects/foo
$ open -a RStudio .
```

On Linux you would use the following commands (note that no '.' is necessary in the second invocation):

```
$ rstudio ~/projects/foo
$ rstudio
```

Handling of .Rprofile

When starting RStudio in an alternate working directory the **.Rprofile** file located within that directory is sourced. If (and only if) there is not an **.Rprofile** file in the alternate directory then the global default profile (e.g. ~/.Rprofile) is sourced instead.

Loading and Saving Workspaces 装载和保存工作空间

If you want to save or load a workspace during an RStudio session you can use the following commands to save to or load from the `.RData` file in the current working directory:

如果你想在 RStudio 会话过程中保存或装载工作空间, 你可使用一下命令从当前工作目录下的 `.RData` 文件来进行保存和装载。

```
> save.image()  
> load(".RData")
```

Note that the **load** function appends (and overwrites) objects within the current workspace rather than replacing it entirely. Prior to loading you may therefore wish to clear all objects currently within the workspace. You can do so using the following command:

注意: **load** 函数

```
> rm(list=ls())
```

Note that since loading is handled at startup and saving is handled at exit, in many cases you won't require these commands. If however you change working directories within a session you may need them in order to sync your workspace with the directory you have changed to.

The RStudio **Workspace** menu also includes items that execute the above described commands, as well as enables you to load or save specific `.RData` files.

Handling of .Rhistory 处理.Rhistory

The `.Rhistory` file determines which commands are available by pressing the up arrow key within the console. By default, RStudio handles the `.Rhistory` file differently than the standard R console or GUI, however can be configured to work the same was as those environments if you wish.

The conventional handling of `.Rhistory` files is as follows:

- Load and save `.Rhistory` within the current working directory
- Only save the `.Rhistory` file when the user chooses to save the `.RData` file

Whereas the default RStudio handling of .Rhistory files is:

- Load and save a single global .Rhistory file (located in the default working directory)
- Always save the .Rhistory file (even if the .RData file is not saved)

The RStudio defaults are intended to make sure that all commands entered in previous sessions are available when you start a new RStudio session. If you prefer the conventional R treatment of .Rhistory files you can customize this behavior using the General panel of the Options dialog.

Customizing RStudio 自定义 RStudio

Overview 概述

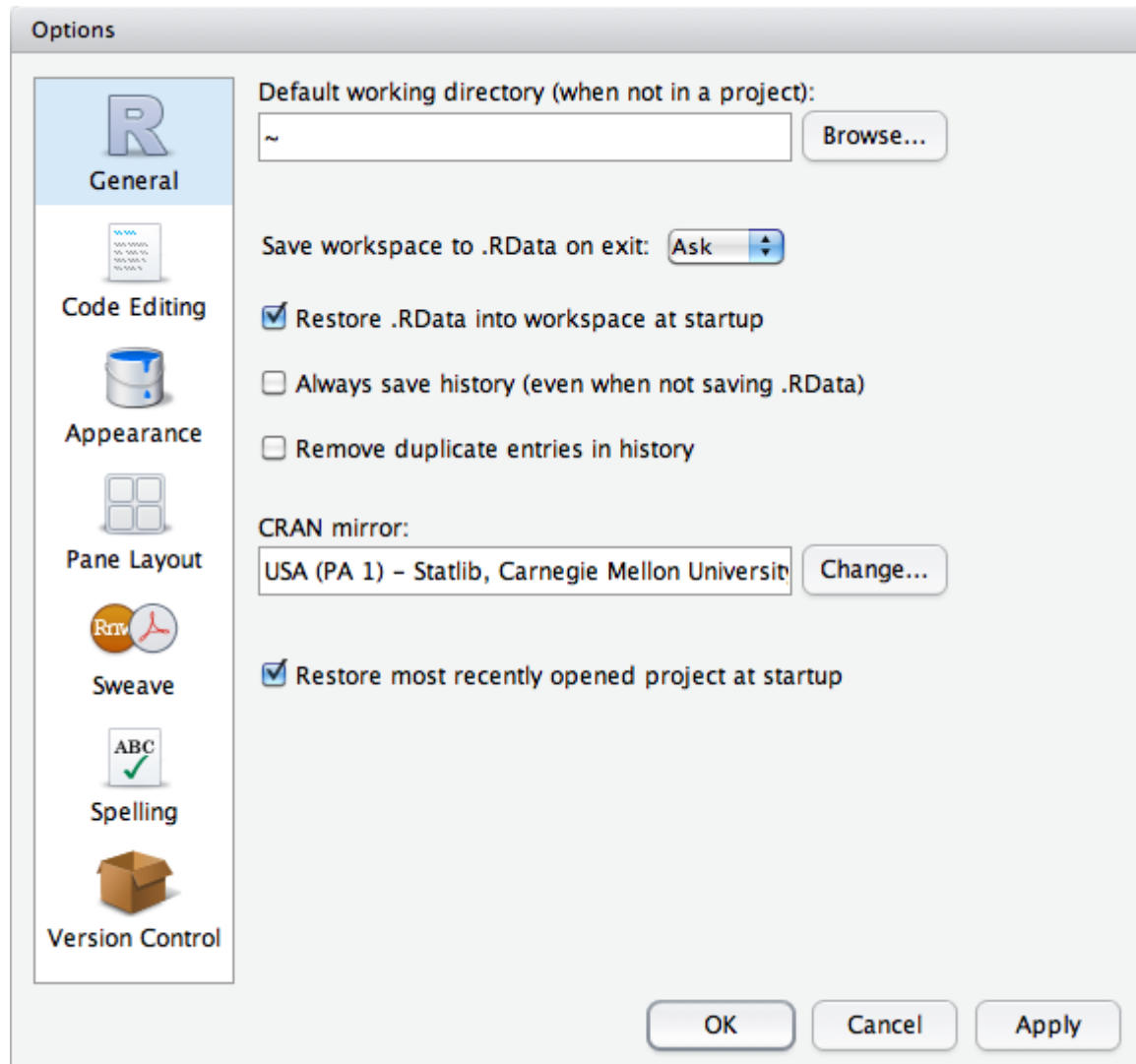
RStudio options are accessible from the Options dialog **Tools : Options** menu and include the following categories:

RStudio 可通过 **Tools : Options** 菜单的选项对话框来进行，包括以下类：

- [General R Options](#) — Default CRAN mirror, initial working directory, workspace and history behavior.
- [Source Code Editing](#) — Enable/disable line numbers, selected word and line highlighting, soft-wrapping for R files, paren matching, right margin display, and console syntax highlighting; configure tab spacing; set default text encoding.
- [Appearance and Themes](#) — Specify font size and visual theme for the console and source editor.
- [Pane Layout](#) — Locations of console, source editor, and tab panes; set which tabs are included in each pane.
- [Sweave](#) — Configure Sweave compiling options and PDF previewing.
- [Spelling](#) — Choose main dictionary language and specify spell checking options.
- [Version Control](#) — Configure locations of Git and Svn binaries and create and/or view SSH RSA keys.

Details on the various settings are provided in the sections below.

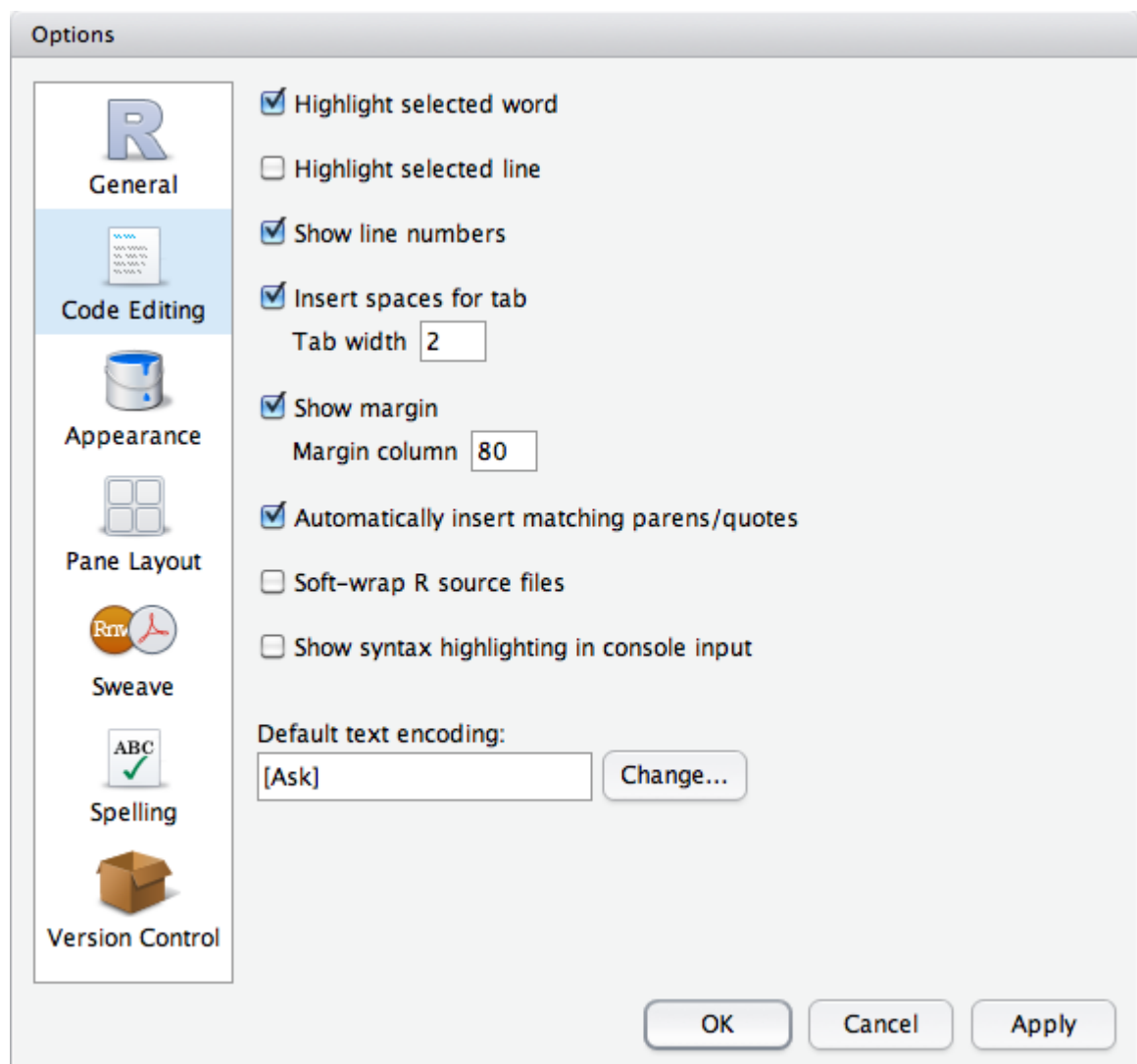
General R Options 一般的 R 选项



- **Default working directory** — Startup directory for RStudio (when not in a project). The initial .RData and .Rprofile files (if any) will be read from this directory. The current working directory and Files pane will also be set to this directory. Note that this setting can be overridden when launching RStudio using a file association or a terminal with a command line parameter indicating the initial working directory.
- **Save workspace to .RData on exit** — Ask whether to save .RData on exit, always save it, or never save it. Note that if the workspace is not dirty (no changes made) at the end of a session then no prompt to save occurs even if Ask is specified.
- **Restore .RData into workspace at startup** — Load the .RData file (if any) found in the initial working directory into the R workspace (global environment) at startup. If you have a very large .RData file then unchecking this option will improve startup time considerably.
- **Always save history (even when not saving .RData)** — Make sure that the .Rhistory file is always saved with the commands from your session even if you choose not to save the .RData file when exiting.

- **Remove duplicate entries in history** — Prevent addition of commands to history if they are the exact same as the most recently added command.
- **CRAN mirror** — Set the CRAN mirror used for installing packages (can be overridden using the `repos` argument to `install.packages`).
- **Restore most recently opened project at startup** — When opening RStudio automatically re-open the most recently used project.

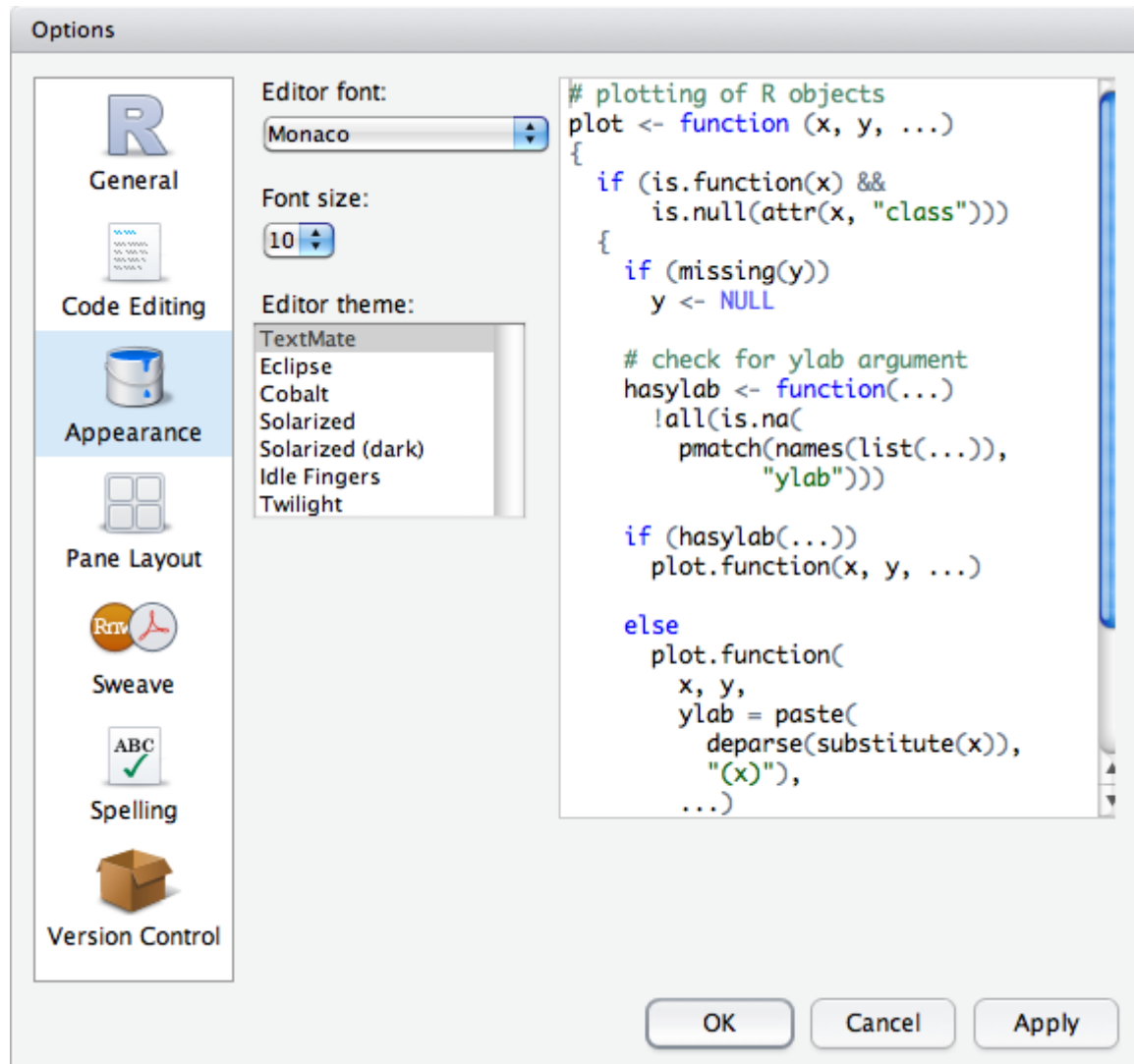
Source Code Editing 源代码编辑



- **Highlight selected word** — Add a background highlight effect to all instances of the currently selected word within the document.
- **Highlight selected line** — Add a background highlight effect to the currently selected line of code.
- **Show line numbers** — Show or hide line numbers within the left margin.

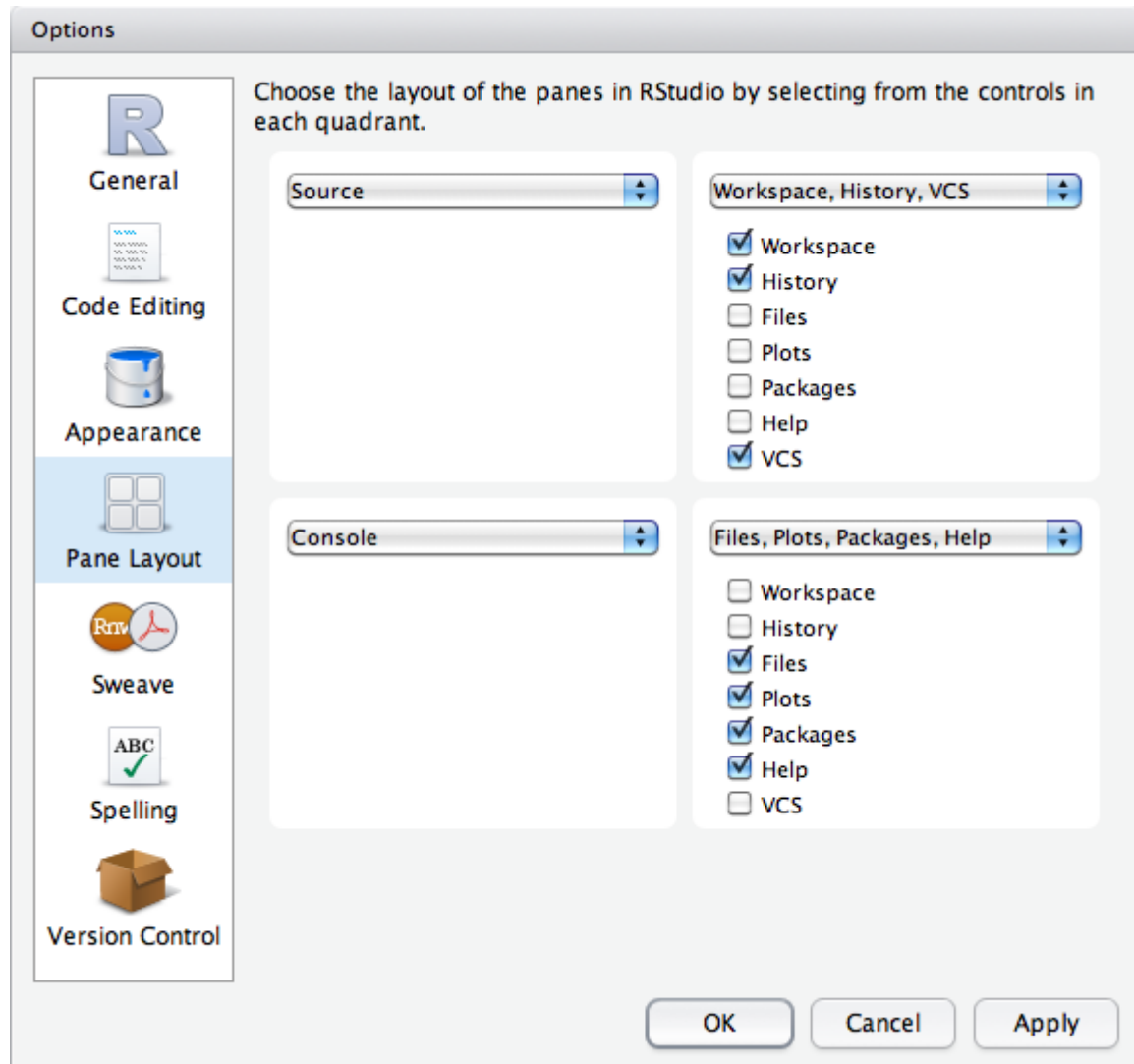
- **Insert spaces for tab** — Determine whether the tab key inserts multiple spaces rather than a tab character (soft tabs). Configure the number of spaces per soft-tab.
- **Show margin** — Display a margin guide on the right-hand side of the source editor at the specified column.
- **Automatically insert matching parens/quotes** — When typing a paren, quote, or brace automatically insert a matching one and position the cursor between them.
- **Soft-wrap R source files** — Wrap lines of R source code which exceed the width of the editor onto the next line. Note that this does not insert a line-break at the point of wrapping, it simply displays the code on multiple lines in the editor.
- **Show syntax highlighting in console input** — Apply R syntax highlighting to the console input line.
- **Default text encoding** — Specify the default text encoding for source files. Note that source files which don't match the default encoding can still be opened correctly using the **File : Reopen with Encoding** menu command.

Appearance and Themes 外观和主题



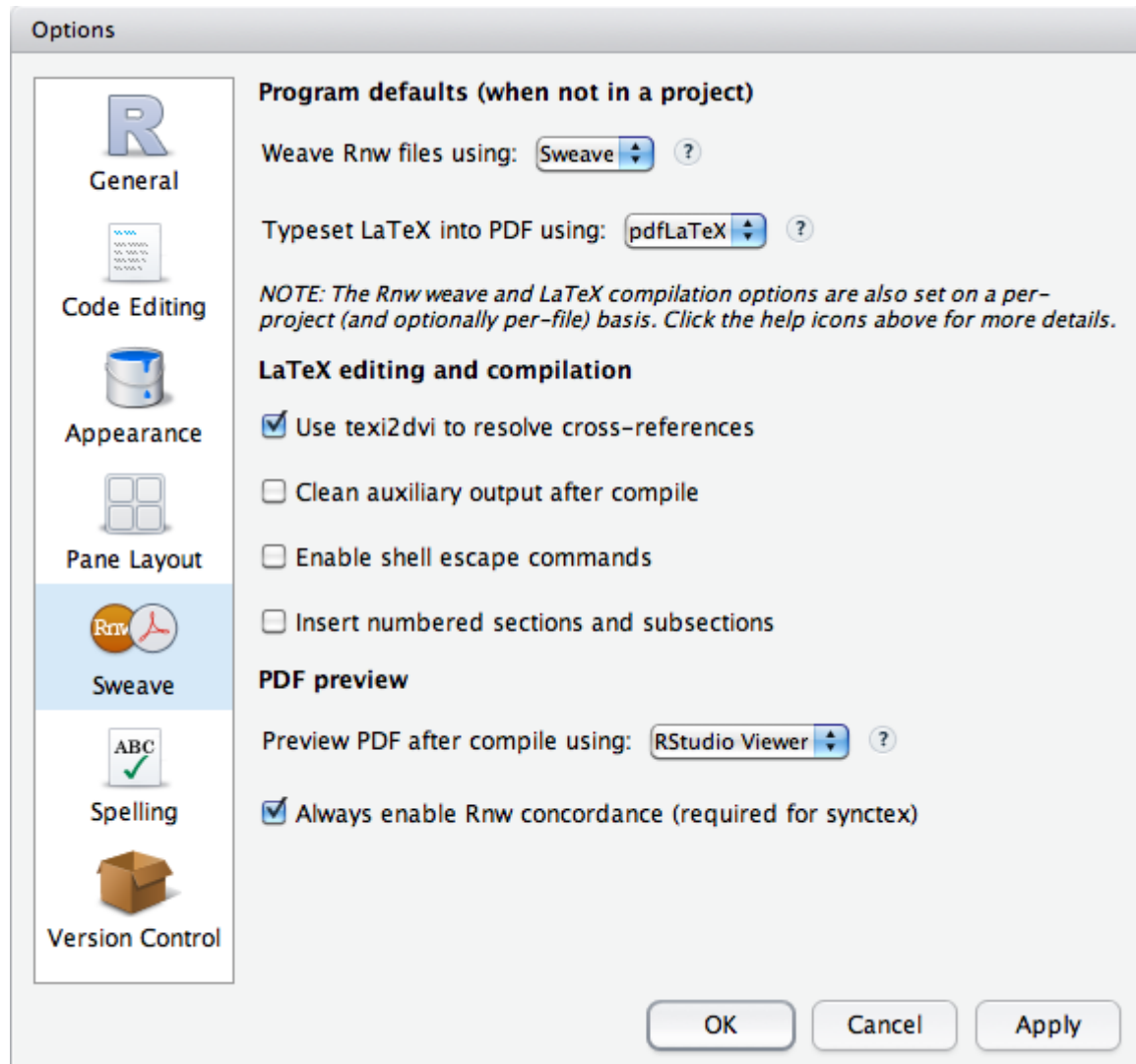
- **Editor font** — Customize the font for panes which display code (Console, Source, History, and Workspace).
- **Font size** — Set the font size (in points) for panes which display code (Console, Source, History, and Workspace).
- **Editor theme** — Specify the visual theme for the Console and Source panes. You can preview the theme using the inline preview or by pressing the Apply button.

Pane Layout 窗体布局



- Specify the location and tab sets of panes within RStudio.
- Each of the 4 panes is always displayed (it isn't currently possible to hide a pane).

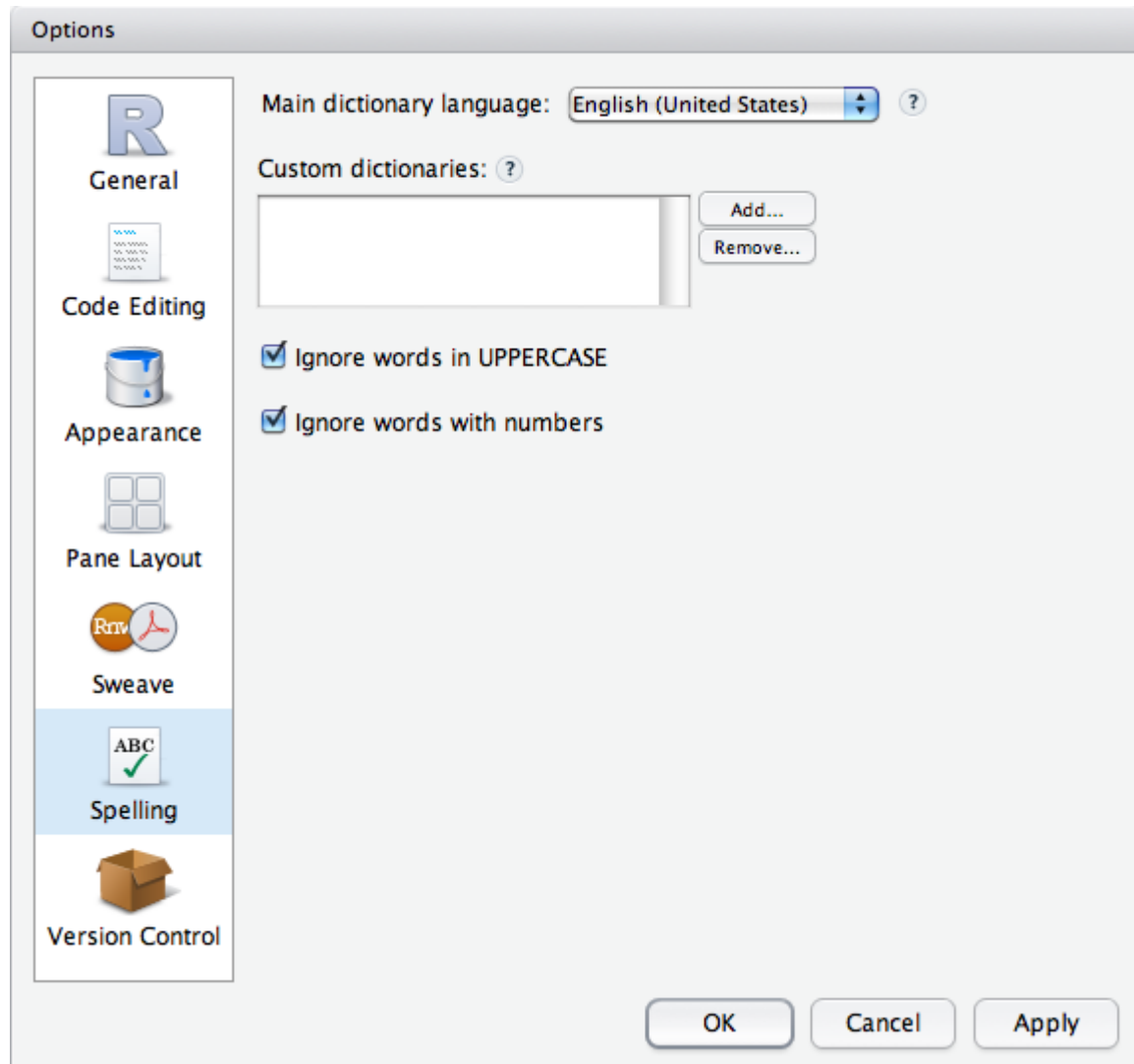
Sweave



- **Weave Rnw files using** — Choose the default global option for weaving files with the **Compile PDF** button. (See [Weaving Rnw Files](#) for additional documentation).
- **Typeset LaTeX into PDF using** — Choose from a list of supported typesetting engines or add a customized one (See additional help for [Customizing LaTeX Options](#)).
- **Use texi2dvi to resolve cross-references** — texi2dvi is utility program included in TeX distributions which automatically runs (and re-runs) the LaTeX, bibtex, and makeindex programs as necessary to correctly resolve all cross-references. You should only disable this option if you have a specific reason not to use texi2dvi (such as a known bug you need to work around). When this option is disabled RStudio will attempt to emulate the behavior of texi2dvi by directly calling the LaTeX program, bibtex, and makeindex as necessary to fully compile the PDF.
- **Clean auxiliary output after compile** — Running the LaTeX program produces various intermediate files (e.g. 'target.out', 'target.aux'). Enabling this option causes these files to be automatically removed after the compile. Note that log files ('target.blg' and 'target.log') are also cleaned up, however they are preserved if errors occur during compilation.

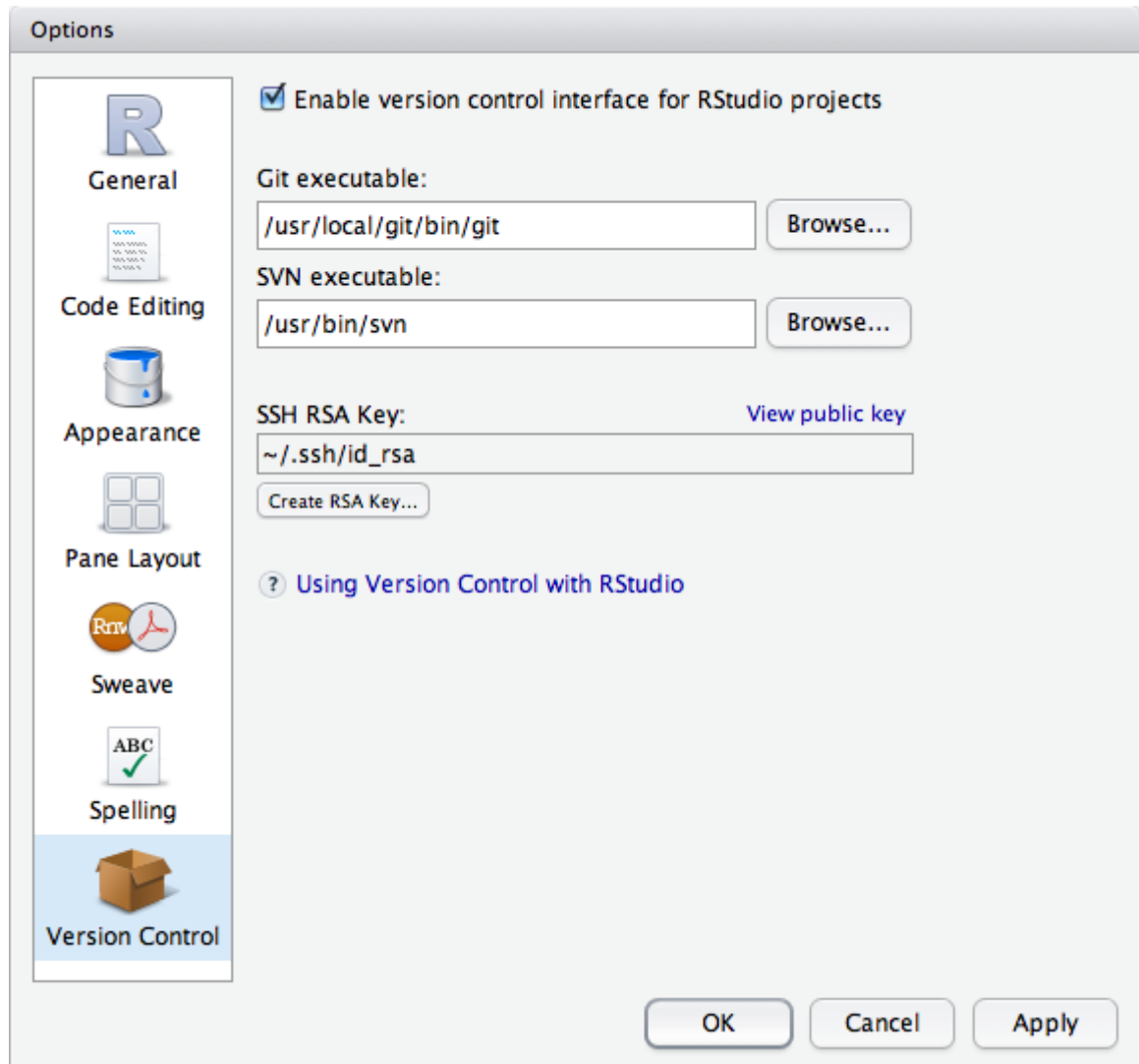
- **Enable shell escape commands** — Enable execution of shell commands using `\write18` by passing the `-shell-escape` option to the LaTeX program. This option is disabled by default because allowing documents to execute embedded shell commands may pose a security risk.
- **Insert numbered sections and subsections** — Specifies that sections and sub-sections inserted using the Format menu should include numbering (if this option is not enabled then sections will be inserted as `\section*`).
- **Preview PDF after compile using** — Choose which application to view your PDF file with after the compile is complete. (See additional help for [PDF Preview](#)).
- **Always enable Rnw concordance** — Automatically set the options required to emit concordance information during Rnw weaving. Concordance provides a mapping between line numbers in the Rnw file and the generated TeX file. This enables both direct navigation to compilation errors within Rnw files and compatibility with Synctex for PDF previewing.

Spelling 拼写



- **Main dictionary language** — RStudio's default installation includes English dictionaries for the US, UK, Canada, and Australia. In addition, dictionaries for 28 other languages can be installed. For additional information on dictionaries, see the [Spelling Dictionaries](#) documentation.
- **Custom dictionaries** — You can also configure RStudio to use Hunspell custom dictionaries that include additional terms commonly used within your field (for example, the [OpenMedSpel](#) dictionary of medical terms). For additional information on customizing dictionaries, see the [Spelling Dictionaries](#) documentation.
- **Ignore words in UPPERCASE** — Ignore words in all UPPERCASE letters while spell checking
- **Ignore words with numbers** — Ignore words with numbers while spell checking

Version Control 版本控制



- **Git executable** — Location of Git executable. By default RStudio locates the Git executable in the system path. If you want to use a version of Git not on the path you can specify it here.
- **SVN executable** — Location of SVN executable. By default RStudio locates the SVN executable in the system path. If you want to use a version of Git not on the path you can specify it here.
- **SSH RSA Key** — When using SSH with RSA public/private key authentication against a remote Git or Subversion repository you need to generate an RSA key and then provide the public key to the remote server. You can both create a new RSA key and view/copy its public key within the Version Control options panel.

Keyboard Shortcuts 快捷键

Console 控制板

Description	Windows & Linux	Mac
Move cursor to Console	Ctrl+2	Ctrl+2
Clear console	Ctrl+L	Command+L
Move cursor to beginning of line	Home	Command+Left
Move cursor to end of line	End	Command+Right
Navigate command history	Up/Down	Up/Down
Popup command history	Ctrl+Up	Command+Up
Interrupt currently executing command	Esc	Esc
Change working directory	Ctrl+Shift+K	Ctrl+Shift+K

Source 源文件

Description	Windows & Linux	Mac
Go to File/Function	Ctrl+.	Ctrl+.
Move cursor to Source Editor	Ctrl+1	Ctrl+1
New document (except on Chrome/Windows)	Ctrl+Shift+N	Command+Shift+N
Open document	Ctrl+O	Command+O
Save active document	Ctrl+S	Command+S
Close active document (except on Chrome)	Ctrl+W	Command+W
Close active document (Chrome only)	Ctrl+Shift+L	Command+Shift+L
Close all open documents	Ctrl+Shift+W	Command+Shift+W

Preview HTML	Ctrl+Shift+Y	Command+Shift+Y
Knit to HTML	Ctrl+Shift+H	Command+Shift+H
Compile PDF (TeX and Sweave)	Ctrl+Shift+I	Command+Shift+I
Insert chunk	Ctrl+Shift+<	Command+Shift+<
Insert code section	Ctrl+Shift+R	Command+Shift+R
Run current line/selection	Ctrl+Enter	Command+Enter
Re-run previous region	Ctrl+Shift+P	Command+Shift+P
Run current document	Ctrl+Alt+R	Command+Option+R
Run from document beginning to current line	Ctrl+Alt+B	Command+Option+B
Run from current line to document end	Ctrl+Alt+E	Command+Option+E
Run the current function definition	Ctrl+Alt+F	Command+Option+F
Run the current chunk	Ctrl+Alt+C	Command+Option+C
Run the next chunk	Ctrl+Alt+N	Command+Option+N
Source a file	Ctrl+Shift+O	Command+Shift+O
Source the current document	Ctrl+Shift+S	Command+Shift+S
Source the current document (with echo)	Ctrl+Shift+Enter	Command+Shift+Enter
Fold selected	Alt+L	Option+L
Unfold selected	Shift+Alt+L	Shift+Option+L
Fold all	Alt+A	Option+A
Unfold all	Shift+Alt+A	Shift+Option+A
Go to line	Ctrl+G	Command+G
Jump to	Shift+Alt+J	Shift+Option+J
Switch to tab	Ctrl+Alt+Down	Ctrl+Option+Down
Previous tab	Win: Ctrl+Alt+Left, Ctrl+PageUp	Linux: Ctrl+Option+Left
Next tab	Win: Ctrl+Alt+Right, Ctrl+PageDown	Linux: Ctrl+Option+Right

First tab	Ctrl+Shift+Alt+Left	Ctrl+Shift+Option+Left
Last tab	Ctrl+Shift+Alt+Right	Ctrl+Shift+Option+Right
Navigate back	Ctrl+F9	Command+F9
Navigate forward	Ctrl+F9	Command+F9
Reindent lines	Ctrl+I	Command+I
Extract function from selection	Ctrl+Shift+U	Command+Shift+U
Comment/uncomment current line/selection	Ctrl+Shift+C	Command+Shift+C
Reflow comment	Ctrl+Shift+/	Command+Shift+/
Transpose Letters		Ctrl+T
Move Lines Up/Down	Alt+Up/Down	Option+Up/Down
Copy Lines Up/Down	Ctrl+Alt+Up/Down	Command+Option+Up/Down
Find and Replace	Ctrl+F	Command+F
Find in Files	Ctrl+Shift+F	Command+Shift+F
Check spelling	F7	F7

Editing (Console and Source) 编辑

Description	Windows & Linux	Mac
Undo	Ctrl+Z	Command+Z
Redo	Ctrl+Shift+Z	Command+Shift+Z
Cut	Ctrl+X	Command+X
Copy	Ctrl+C	Command+C
Paste	Ctrl+V	Command+V
Select All	Ctrl+A	Command+A
Jump to Word	Ctrl+Left/Right	Option+Left/Right
Jump to Start/End	Ctrl+Home/End Ctrl+Up/Down	or Command+Home/End Command+Up/Down
Delete Line	Ctrl+D	Command+D

Select	Shift+[Arrow]	Shift+[Arrow]
Select Word	Ctrl+Shift+Left/Right	Option+Shift+Left/Right
Select to Line Start	Shift+Home	Command+Shift+Left or Shift+Home
Select to Line End	Shift+End	Command+Shift+Right or Shift+End
Select Page Up/Down	Shift+PageUp/PageDown	Shift+PageUp/Down
Select to Start/End	Ctrl+Shift+Home/End Shift+Alt+Up/Down	or Command+Shift+Up/Down
Delete Word Left	Ctrl+Backspace	Option+Backspace or Ctrl+Option+Backspace
Delete Word Right		Option+Delete
Delete to Line End		Ctrl+K
Delete to Line Start		Option+Backspace
Indent	Tab (at beginning of line)	Tab (at beginning of line)
Outdent	Shift+Tab	Shift+Tab
Yank line up to cursor	Ctrl+U	Ctrl+U
Yank line after cursor	Ctrl+K	Ctrl+K
Insert currently yanked text	Ctrl+Y	Ctrl+Y
Insert assignment operator	Alt+-	Option+-
Show help for function at cursor	F1	F1
Show source code for function at cursor	F2	F2

Completions (Console and Source) 完成

Description	Windows & Linux	Mac
Attempt completion	Tab or Ctrl+Space	Tab or Command+Space
Navigate candidates	Up/Down	Up/Down
Accept selected candidate	Enter, Tab, or Right	Enter, Tab, or Right
Dismiss completion popup	Esc	Esc

Views 浏览

Description	Windows & Linux	Mac
Goto File/Function	Ctrl+.	Ctrl+.
Move cursor to Source Editor	Ctrl+1	Ctrl+1
Move cursor to Console	Ctrl+2	Ctrl+2
Show Workspace	Ctrl+3	Ctrl+3
Show History	Ctrl+4	Ctrl+4
Show Files	Ctrl+5	Ctrl+5
Show Plots	Ctrl+6	Ctrl+6
Show Packages	Ctrl+7	Ctrl+7
Show Help	Ctrl+8	Ctrl+8
Show Git/SVN	Ctrl+9	Ctrl+9
Sync Editor and PDF Preview	Ctrl+F8	Command+F8

Plots 画图

Description	Windows & Linux	Mac
Previous plot	Ctrl+Shift+PageUp	Command+Shift+PageUp
Next plot	Ctrl+Shift+PageDown	Command+Shift+PageDown
Show manipulator	Ctrl+Shift+M	Command+Shift+M

Git/SVN

Description	Windows & Linux	Mac
Diff active source document	Ctrl+Shift+D	Command+Shift+D
Commit changes	Ctrl+Shift+T	Command+Shift+T
Scroll diff view	Ctrl+Up/Down	Ctrl+Up/Down

Stage/Unstage (Git)

Spacebar

Spacebar

Stage/Unstage and move next
(Git)

Enter

Enter