

15.1.3

Install RStudio

Jeremy's new leadership role is starting to feel a bit more real now that R is actually installed. Jeremy's next step is to install RStudio and some libraries his lead analyst suggested might be helpful.

Once you have completed the installation for R, it's time to install RStudio. Now navigate to the [RStudio Download Page](https://rstudio.com/products/rstudio/download/?utm_source=downloadrstudio&utm_medium=Site&utm_campaign=home-hero-cta#download) (https://rstudio.com/products/rstudio/download/?utm_source=downloadrstudio&utm_medium=Site&utm_campaign=home-hero-cta#download) and select the most appropriate installer link. Refer to the following image:

Installers for Supported Platforms

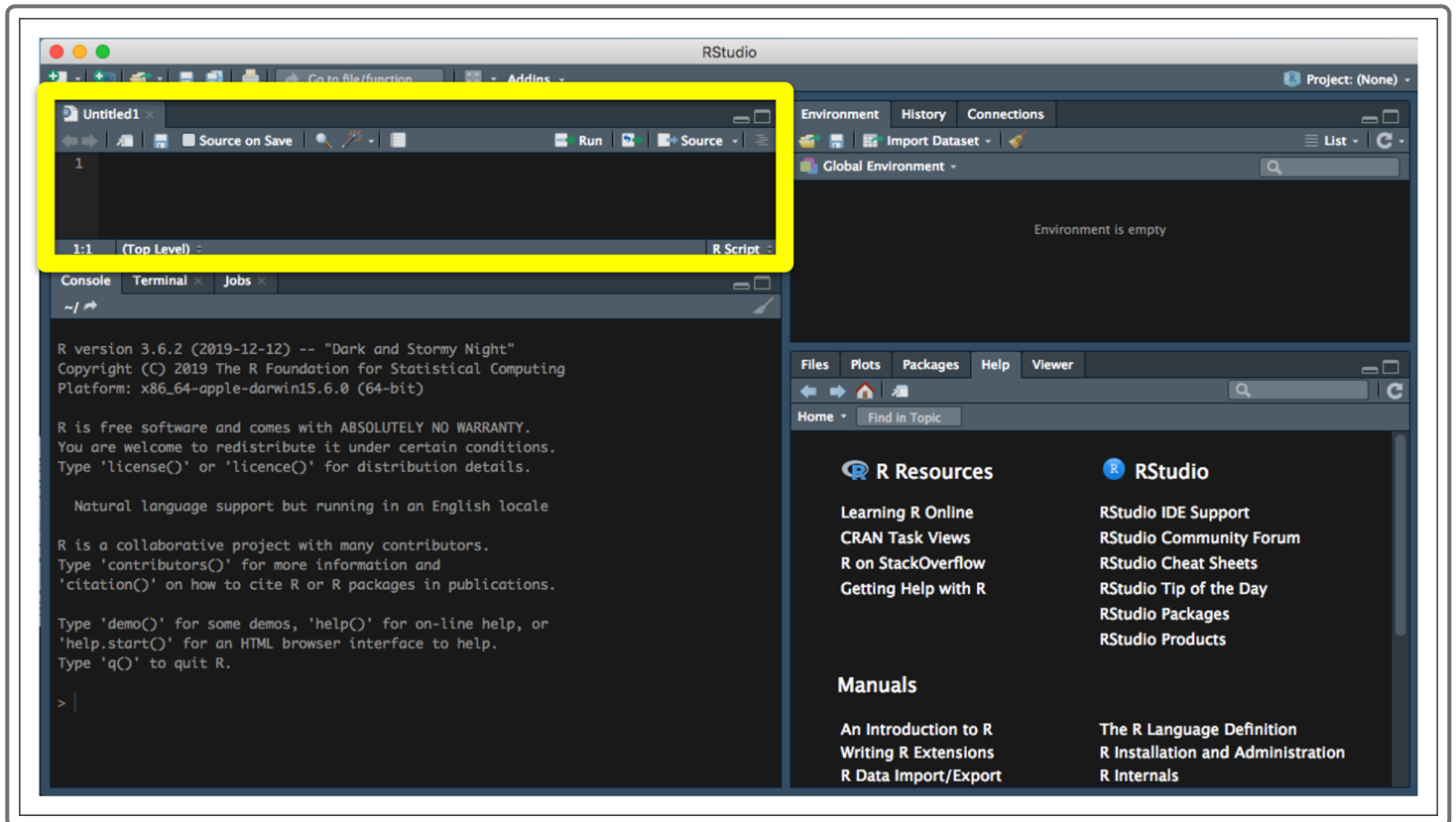
Installers	Size	Date	MD5
RStudio 1.2.5019 - Ubuntu 18/Debian 10 (64-bit)	106.04 MB	2019-11-01	a6c9af3d8b1621eb155d23c879c1a75a
RStudio 1.2.5019 - Debian 9 (64-bit)	106.39 MB	2019-11-01	bc7b0b25b41e39fb6f1aefa74163a133
RStudio 1.2.5019 - Fedora 28/Red Hat 8 (64-bit)	120.89 MB	2019-11-01	2291b1befb02622b3aa02c43638ee5c2
RStudio 1.2.5019 - macOS 10.12+ (64-bit)	126.88 MB	2019-11-01	55738355277e8ec660e628acaf2a401b
RStudio 1.2.5019 - SLES/OpenSUSE 12 (64-bit)	99.04 MB	2019-11-01	3bcbf47f40944cc4a5ef4f6fb42319c1
RStudio 1.2.5019 - OpenSUSE 15 (64-bit)	107.09 MB	2019-11-01	29d07b198b7aac92356f8487911efbfa
RStudio 1.2.5019 - Fedora 19/Red Hat 7 (64-bit)	120.26 MB	2019-11-01	dab1cb5f0ed39f5bcf0c795e2938fa94
RStudio 1.2.5019 - Ubuntu 14/Debian 8 (64-bit)	96.93 MB	2019-11-01	f86811fce50b48850fed259d6ce7ef13
RStudio 1.2.5019 - Windows 10/8/7 (64-bit)	149.82 MB	2019-11-01	4d6521a9b89d70c3bf50414c8b6708f2
RStudio 1.2.5019 - Ubuntu 16 (64-bit)	104.91 MB	2019-11-01	67d5a2c255f2bc1a171c7e417853102c

If you're using macOS, drag the RStudio application into your application folder. If you're a Windows user, run it through the installer as you would with any other Windows program.

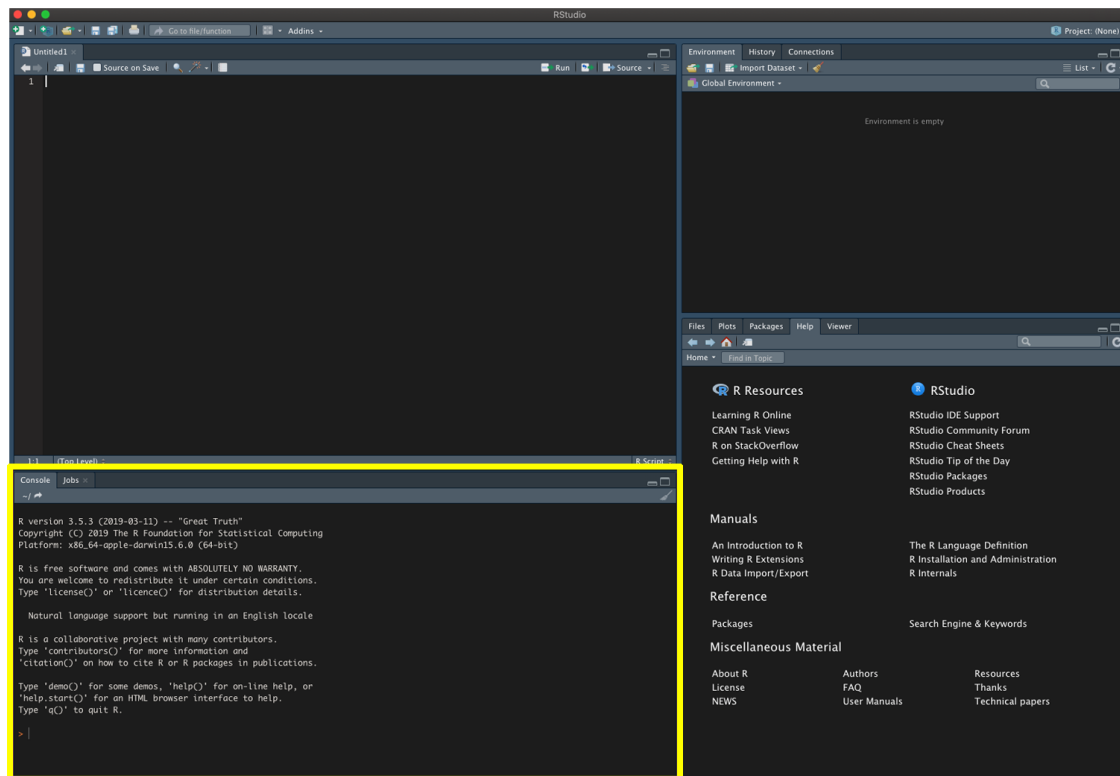
Once you have R and RStudio installed, run RStudio for the first time, get acquainted with the software, and install our required packages.

Navigate and Configure RStudio

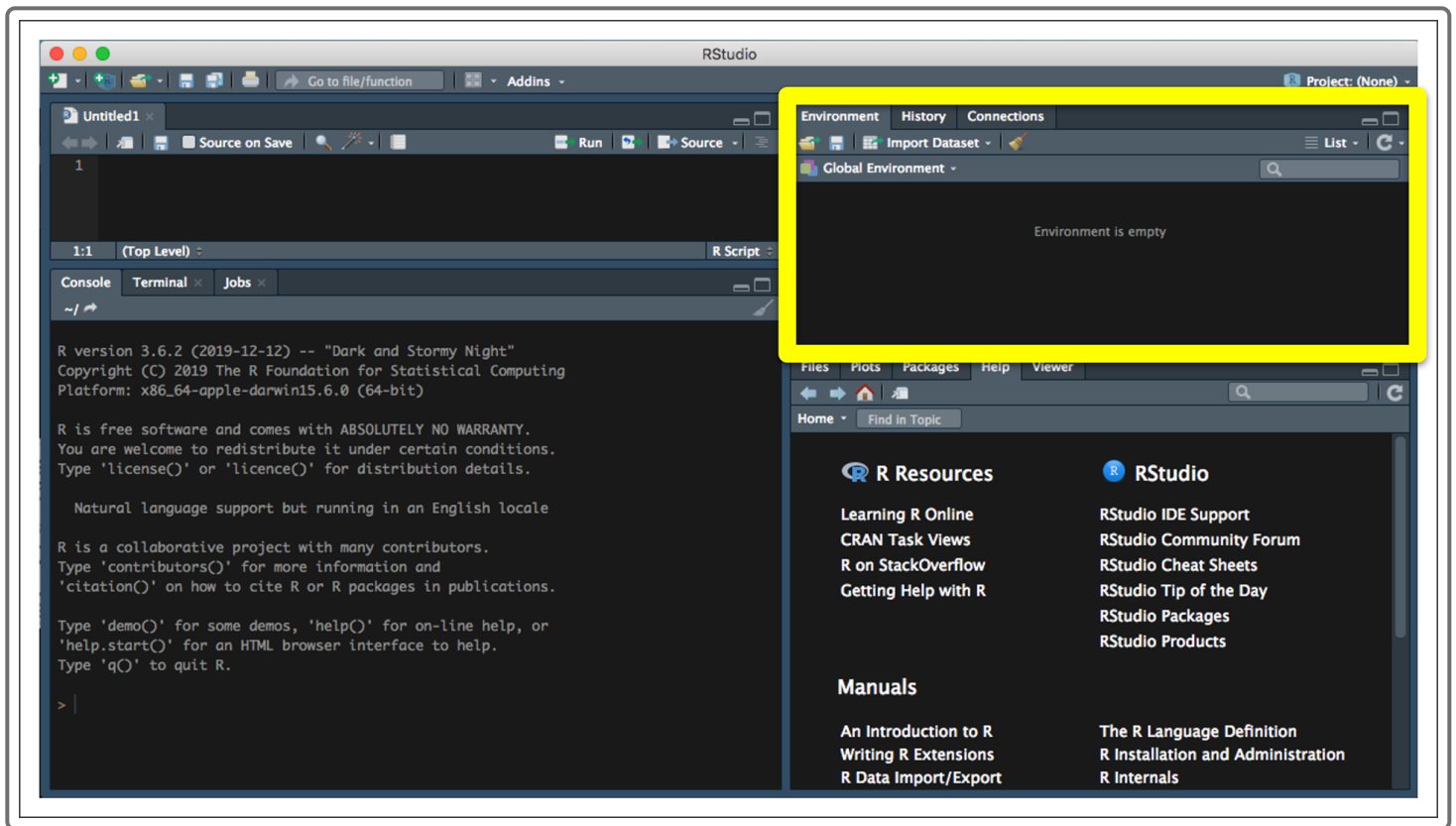
When you first open up RStudio, you'll notice four panes laid out within the application window. The top-left pane contains your source, or (any RScripts, tables, and files you open within RStudio). By default, RStudio will open an untitled RScript file in the pane for you, so you can start programming right away:



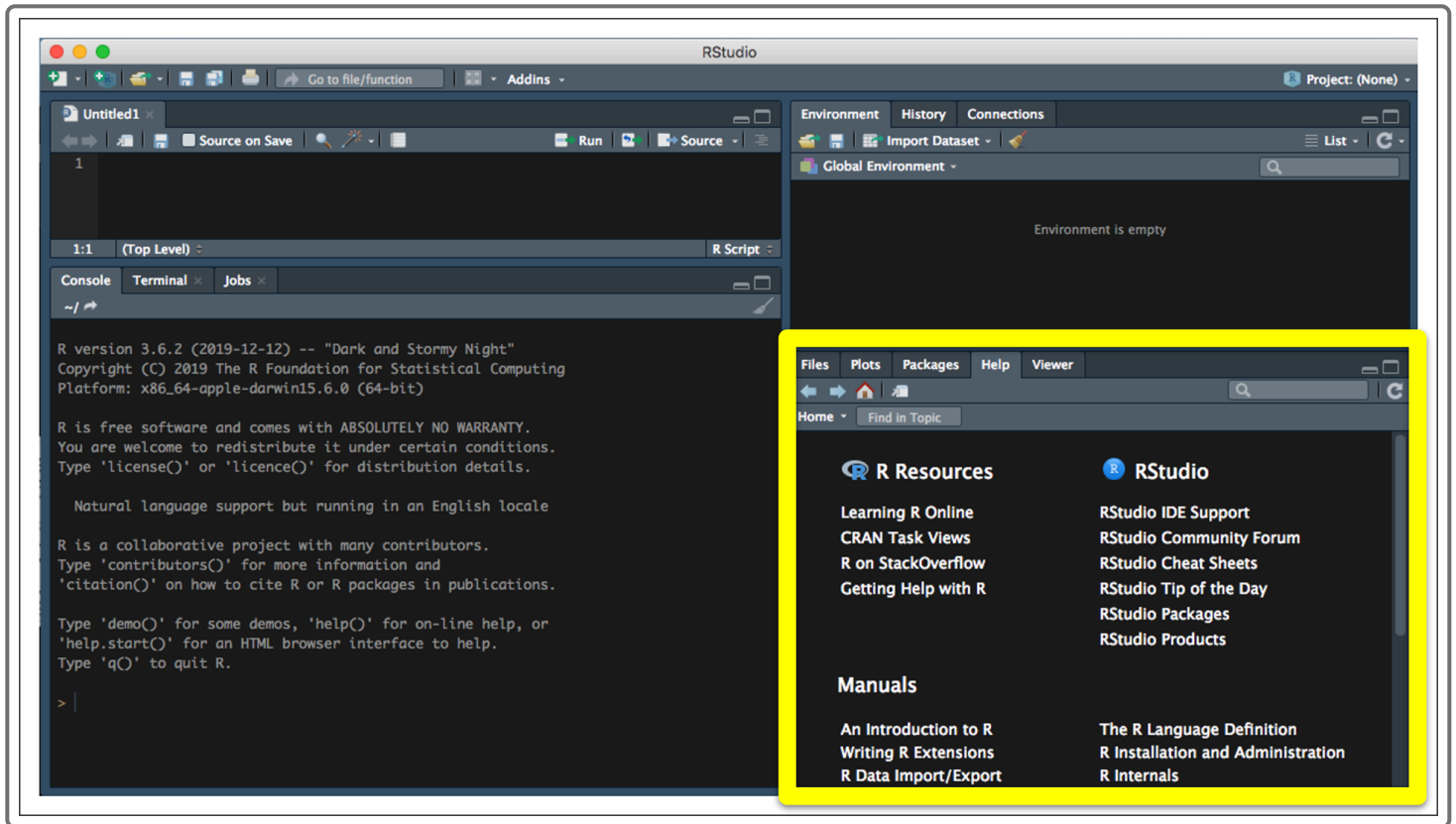
The bottom-left pane contains the R console. Similar to Python, R can either run an RScript as an executable script or R can run interactively. RStudio combines the best of both worlds where the source RScript (in the top-left pane) can be run all at once, or line by line. By including the R console within the application, we can interact with our environment in real time and test parts of our code before we write them in our scripts:



The top-right pane contains our environment objects, such as variables, functions, and data frames. As we execute commands in the R console, either using our source RScript or manually, any objects generated in the R environment will show up in the top-right pane. This environment pane helps us keep track of the shape, data type, and contents of each variable within our environment without having to print out our variables in the console. As we explore R in this module, the environment pane will prove even more useful for tracking what each line of code does to our data:



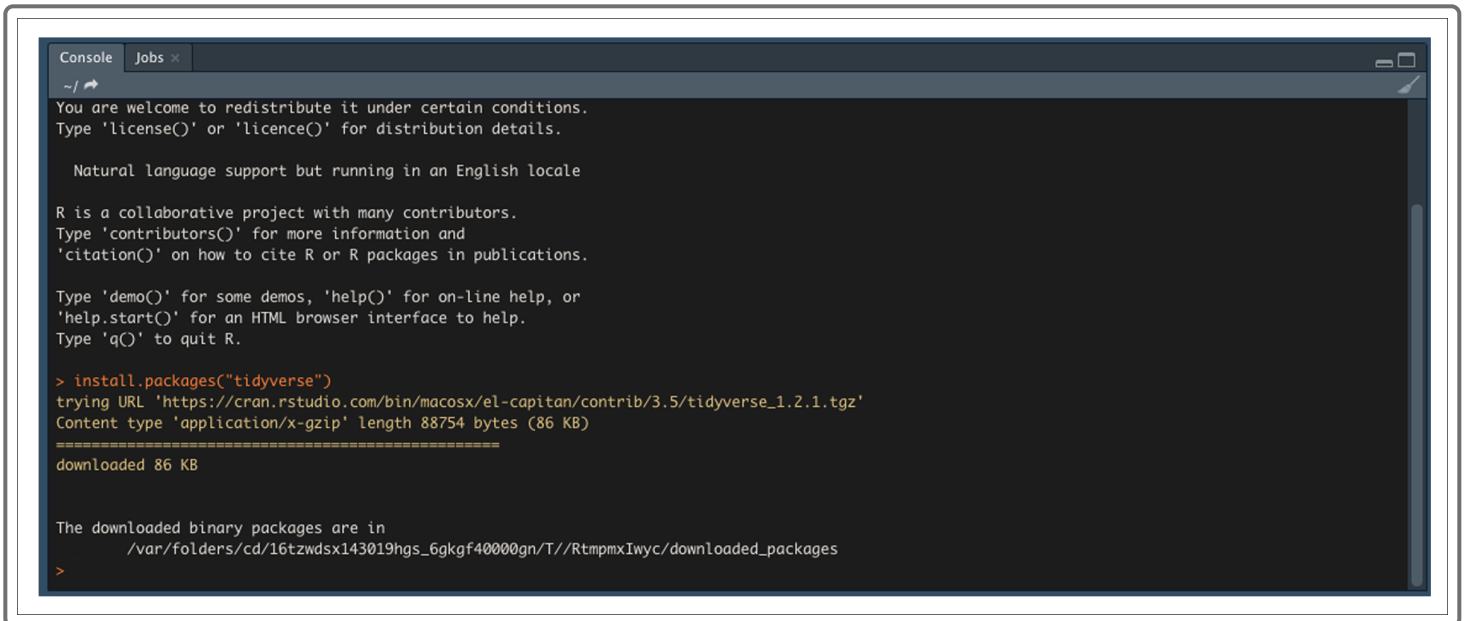
On the bottom right is the multi-tool pane, which contains tabs for a file explorer, R documentation help, installed package list, and a plot viewing tool. Later, we'll refer to the Plots tab for exploring our generated plots. Additionally, you can use the Files tab to open RScripts from your computer or to copy file paths to include within your RScripts. Finally, to learn more about a function or object from a library in R, simply type `?<name of function or object>` in the R console to open the documentation in the Help tab of the multi-tool pane:



[Retake](#)

Now that we understand RStudio's layout, we'll install our required libraries to use them in our RScripts for this module. Thankfully, R developers have built robust library collections, such as the [tidyverse](https://www.tidyverse.org/) (<https://www.tidyverse.org/>), that simplify the installation process for the most common data analysis packages in R. To install packages in our R environment, use the `install.packages()` function. Therefore, to install the tidyverse in our R environment, simply run the following command in the R console:

```
> install.packages("tidyverse")
```



```
Console Jobs x
~/
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> install.packages("tidyverse")
trying URL 'https://cran.rstudio.com/bin/macosx/el-capitan/contrib/3.5/tidyverse_1.2.1.tgz'
Content type 'application/x-gzip' length 88754 bytes (86 KB)
=====
downloaded 86 KB

The downloaded binary packages are in
  /var/folders/cd/16tzwdx143019hgs_6gkgf40000gn/T//RtmpmxIwyc/downloaded_packages
>
```

We'll need the `jsonlite` library for this module. To install our other required libraries and packages, run the following in the R console:

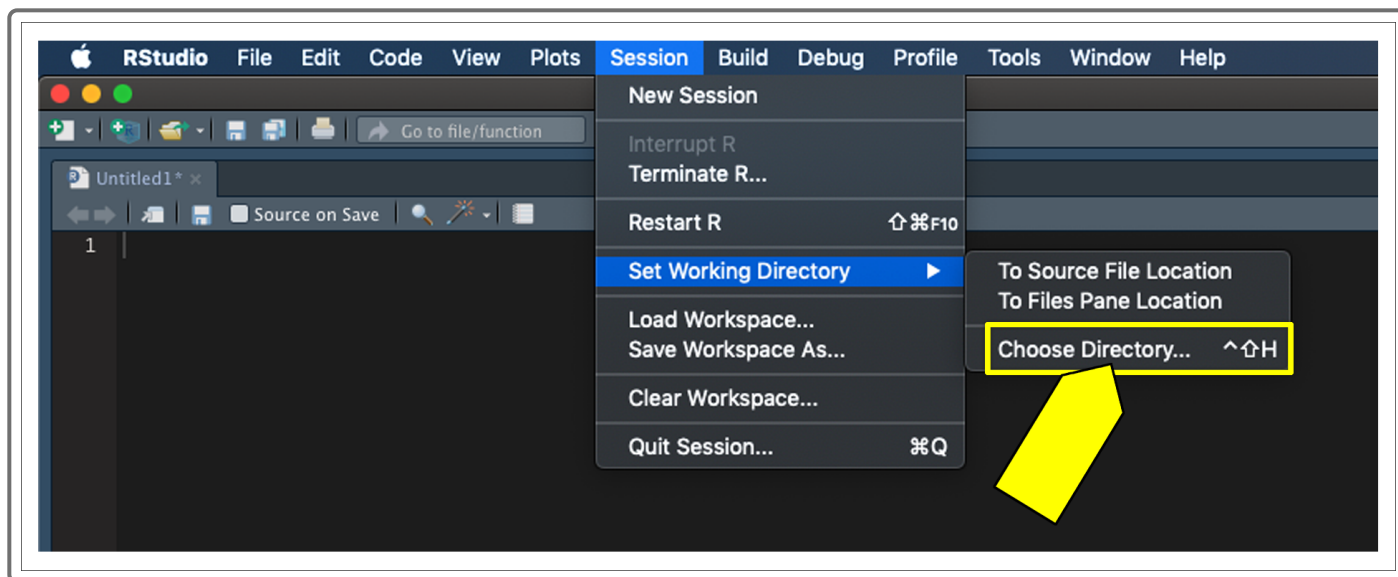
```
> install.packages("jsonlite")
```

CAUTION

When installing packages in R, be sure to wrap the package name in quotation marks, otherwise R will throw an error.

Lastly, before we start learning how to program in R, we need to create a working directory folder structure on our computers. This will allow us to keep all of our RScripts and analysis results in a neat, organized structure and to simplify the process of reading in any external files into our R environment. Follow these steps to create a working directory in R:

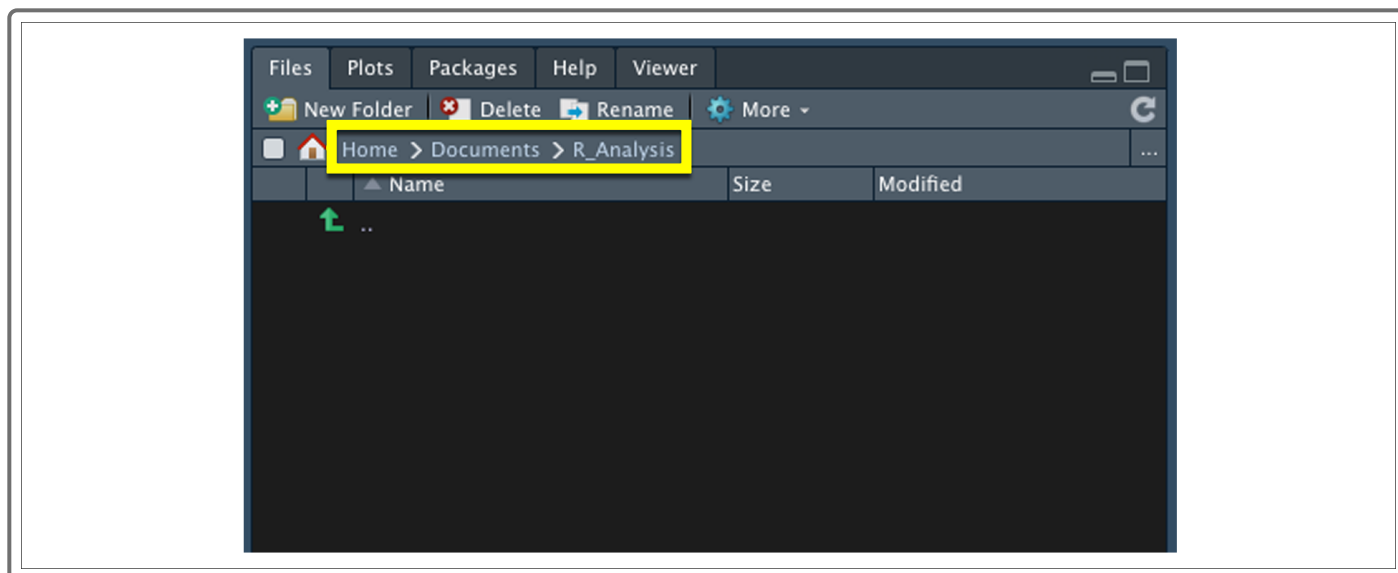
1. Make a folder on your computer called "R_Analysis," or whatever would help identify your R analysis and scripts folder.
2. In the R menu screen, go to Session, click Set Working Directory, then select Choose Directory:



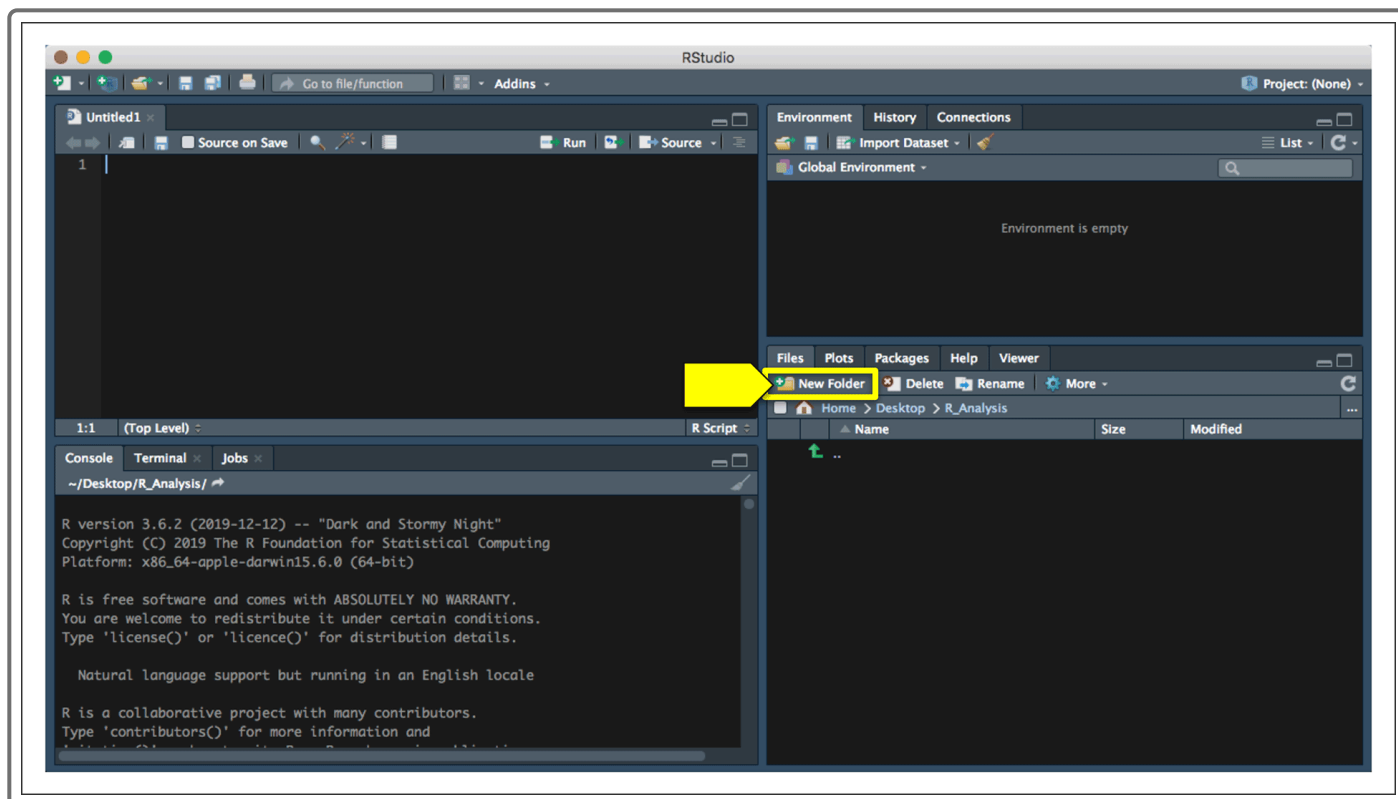
NOTE

Although your menu bar may look slightly different, the menu options are the same for both macOS and Windows.

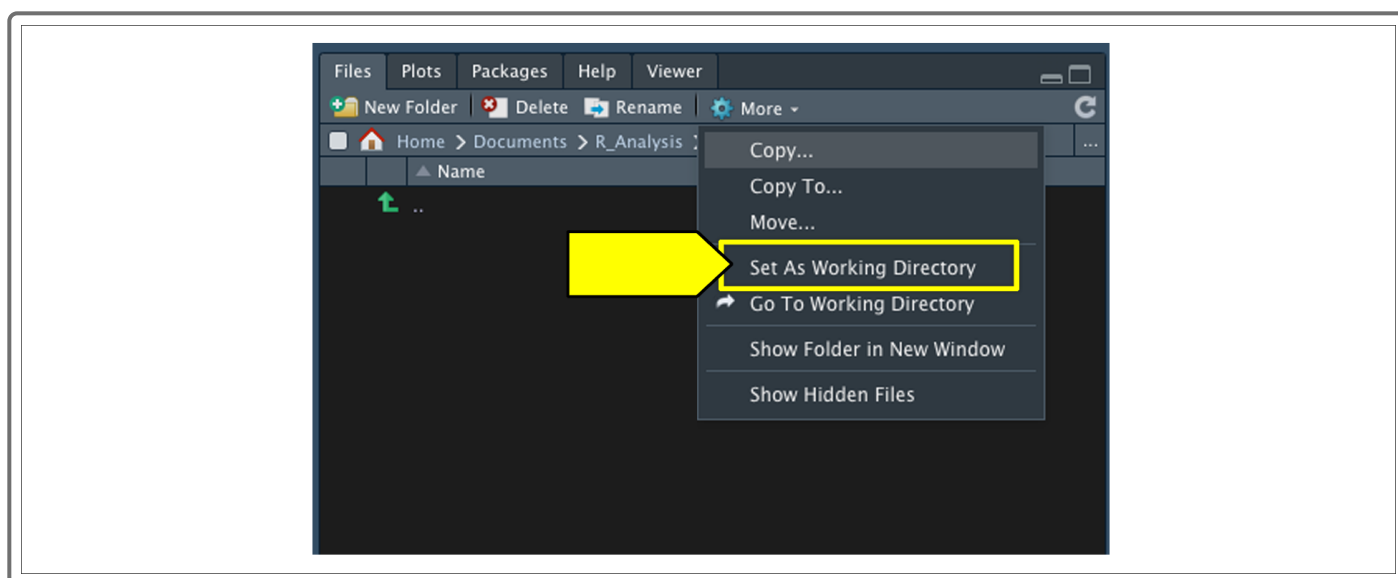
3. Navigate to the folder on your computer and select Open. If you click on the Files tab in your bottom-right multi-tool pane, notice that the folder now represents your active working directory:



4. Now create a new folder in your active directory using the "New Folder" button. For the purposes of this first section of the module, let's name this new folder "01_Demo." Once you have created the folder, press the refresh () button to refresh the directory to see your new folder:



5. Once you have created your new 01_Demo folder, use the file pane and click on the folder to navigate within it. Within the file pane, click on More and select the Set As Working Directory option to make your 01_Demo folder your new working directory:



IMPORTANT

To set a folder as your default working directory, go to Tools, Global Options, General, and specify the location of the folder.

As you progress through the module generating RScripts and analyzing data using R, it's always a good idea to keep a clean working directory. Feel free to make subfolders within your R_Analysis folder, but be sure to always change your working directory so your output tables and figures do not get lost.

© 2020 - 2022 Trilogy Education Services, a 2U, Inc. brand. All Rights Reserved.