In this course, you've learned the basics of R programming. Here at Dataquest, we advocate a "learn by doing" approach. We teach you a specific concept and then you immediately apply that concept. Missions are a highly structured way of "learning by doing."

Guided projects, on the other hand, are less structured and focus more on exploration. Guided projects help you synthesize concepts learned during the Dataquest missions and start building a portfolio.

In this guided project, we'll learn how to install & work in the RStudio. RStudio has a ton of added features compared to using a text editor or console. RStudio consolidates all these functionalities into one environment. In addition, R is a powerful visualization language. RStudio will execute your visualization and display the visualization right next to your code:

You'll see, that multiple sections make up the interface. We'll explain each section of this interface later in this mission.

RStudio is known as an Integrated Development Environment. An [Integrated Development Environment](https://en.wikipedia.org/wiki/Integrated_development_environment) is a software suite that consolidates the basic tools developers need to write and test software. RStudio is one of the most commonly used IDE's in the R community. However, there are many other IDE's as well such as:

* [Microsoft Visual Studio for R](https://www.visualstudio.com/vs/rtvs/)
* [Eclipse](https://www.eclipse.org/)

The main benefits of using RStudio is that the text editor has features such as autocompletion, detect functions while also weaving notebooks, console etc into one interface.

For this mission, there is no answer checking or coding. You'll progress by going through each screen. We'd recommend toggling back and forth between this Guided Project and using RStudio on your machine. While going through the setup process, if you run into issues, here are a few things you can try:

* Search Google and StackOverflow for the error message you received. Try debugging the issue yourself.
* Post to Dataquest's Slack community. Use the **#setup** channel for all installation-related questions.

The first step is to install R on your machine. To install R on your machine, navigate to https://cran.r-project.org/.

You should see the following section when arriving on the home page:

**Mac**

1. Select the **Download R for (Mac) OSX** option.
2. Select **R-3.4.3** as it's the most up to date version of R.
3. Follow the standard instructions for installing applications on Mac.
4. Drag & drop the R application into your **Applications folder**.
5. Open **R** from your applications folder. The R console should appear. Try writing a few expressions to get a feel for the console.

**Windows**

1. Select the **Download R for Windows** option.
2. Select **Base R**. Base R is the package built into the language. A package is a collection of functions & code in a well-defined format. All functions we've used in the previous four missions come from the Base R package. Later on in this track, we'll introduce additional packages.
3. Follow the standard instructions for installing programs onto Windows. When going through the installation, it might ask you:
   * **32 bit** or **64 bit**: The installation will automatically install both. If you're curious to know what your machine is, read [this](http://support.wdc.com/KnowledgeBase/answer.aspx?ID=9405).
   * **Customize Startup** or **Accept Default Startup Options**: This is just customizing the windows that pop up when you open the program. This'll depend on your personal preference, but we'll go with default.
4. After installation is complete, open up **R** from programs.
5. Open the R console and play around. Try writing a few expressions to get a feel for the console.

**Linux/Ubuntu**

1. Select the **Download R for Linux** option.
2. Select the **ubuntu** option.
3. After installation is complete, open up **R**.
4. Open the R console and play around. Try writing a few expressions to get a feel for the console.

Now that we've installed R, let's install RStudio. To install RStudio, head over to the RStudio downloads page [here](https://www.rstudio.com/products/rstudio/download/).

The screenshots for this mission will be on a Mac. The general steps should be the same. We'll let you know if there are any specific differences. If you have specific Windows or Linux setup installation questions, feel free to reach out on the #setup slack channel.

When you reach the RStudio downloads page, click download on the **RStudio Desktop Open Source License FREE** option. Clicking this should take you to the following page:

If you're using a Mac or Windows, you'll select the corresponding option. For ubuntu, you'll notice multiple 32 bit or 64 bit options. 32 or 64 bit refer to a computer's [instruction set](https://en.wikipedia.org/wiki/Instruction_set_architecture).

To discover whether you use a 32 or 64 bit on Ubuntu, you can check [here](https://askubuntu.com/questions/41332/how-do-i-check-if-i-have-a-32-bit-or-a-64-bit-os).

To install RStudio, follow these steps:

**Mac**:

1. Download RStudio from http://rstudio.org/download/desktop. Choose **RStudio Desktop (FREE)**.
2. Select the Mac OSX version.
3. Install RStudio by dragging into Applications folder.

**Windows**:

1. Download RStudio from http://rstudio.org/download/desktop. Choose **RStudio Desktop (FREE)**.
2. Select the Windows version.
3. Download and open RStudio from programs.

**Ubuntu**:

1. Download RStudio from http://rstudio.org/download/desktop. Choose **RStudio Desktop (FREE)**.
2. Select the Ubuntu 32 or 64 bit version. Click [here](https://en.wikipedia.org/wiki/Instruction_set_architecture) to find out which instruction set corresponds with your machine.
3. Download and open RStudio.

Now that you've installed RStudio, for the next few screens, we'd recommend you keep RStudio open and toggle back and forth between the mission screens and the RStudio interface, playing around with the different functionalities.

After loading R Studio, you should see an interface with multiple sections:

Notice the files tab on the bottom:

The files tab will tell you your location. Whenever you write & save your code, your code will be saved to your current working directory. Always be weary of which working directory you're in. If you're looking to load a csv file but you're in the wrong working directory, the interpreter will return an error.

You can change your working directory by clicking through the different locations on your computer:

For a more in-depth explanation on navigating working directories, feel free to check out the Command Line: Beginner course.

Once you make sure you're working in the correct working directory, let's introduce you to the functionalities in RStudio. The beauty of using RStudio, is that you can use the R console, R notebook & text editor all within the IDE. We'll introduce an R notebook in a later screen.

Let's first show you how where you can find the R Console. The R Console in RStudio performs the same function as the console opened directly from R. Here's where the console is located:

Click the box shape on the right hand corner of the **Console** window to open. After opening the console, you'll see that you can write code there:

Here, we created a vector containing the values 1,2,3,4,5. Printing vector will display the results right underneath your code. The console, is a great "sandbox" where you can test out concepts or code immediately. Feel free to play around with the console.

5. Using the R Console

In the previous screen, you'll see that once you create an object, the object is displayed in the Global Environment section:

Whenever you create a new object, the object will be stored in the *environment*. When you're writing hundreds of lines of code, this'll be useful in helping you keep track of what objects you created.

For those of you familiar with python, the R markdown ( .Rmd ) file is R's version of a jupyter notebook. For those unfamiliar, the R markdown file combines a text file & script file into one document. Using a markdown file is great when you want to produce reports containing code, visualizations & explanations, all in one location. It allows you to write & code all in one location.

Taken from the [RStudio website](http://rmarkdown.rstudio.com/lesson-1.html,): *An R notebook is an R markdown file with chunks that can be executed independently and interactively, with output visible immdiately beneath the input.*

In other words, R notebooks are one type of markdown file. There are other types of markdown document types you could use as well.

To create an R notebook, you can either use **File** -> **New File** -> **R Notebook** or click here:

After creating the markdown, RStudio will display the default text + code showing you how to run your code.

The markdown will provide you with a number of keyboard shortcuts you can use:

To run a code chunk:

**Mac**: *Cmd + Shift + Enter* **Windows**: *Ctrl + Shift + Enter*

To create a code chunk:

**Mac**: *Cmd + Option + I* **Windows**: *Ctrl + Alt + I*

Here's an example of running the code block:

We've gone over two ways to write code in RStudio. Another method is to use an R Script. An R script in RStudio is equivalent to a text file containing R code, executed from the command line. Compared to markdowns, writing scripts are less useful for writing reports. They are more useful when we want to write production-level code or automate a task.

To open an R script, you'll go to the same location where you opened your R notebook. This time, you'll select **R Script**:

The keyboard shortcuts for running code on an R script is the same as the R notebook:

To run a code chunk:

**Mac**: *Cmd + Shift + Enter* **Windows**: *Ctrl + Shift + Enter*

Since we can only write code on an R script, not text, the shortcut for inserting a code chunk won't work. The results of our code will be shown in the console window after running: