Download and install R and Rstudio

The language of data analysis is the R programming language. It powers the work of people like data scientists, statisticians, social and natural scientists all over the world by allowing them to perform and communicate their data analyses in a principled way.

Throughout this course you'll learn how to program in R and use the R programming language to make graphics, produce reports and even create interactive web applications. Before you do, you'll need to get yourself ready by installing:

- the R programming language on your computer
- an integrated development environment (IDE) called RStudio that allows you to write R programs and interact with R code on your computer.

What's the difference?

You might be asking: so what's the difference between R and RStudio? Julie Lowndes (http://jules32.github.io/resources/RStudio_intro/) gives a great answer:

> *If R were an airplane, RStudio would be the airport, providing many, many supporting services that make it easier for you, the pilot, to take off and go to awesome places. Sure, you can fly an airplane without an airport, but having those runways and supporting infrastructure is a game-changer.*

The following provides an overview of how to download, install and get up and running with R and RStudio.

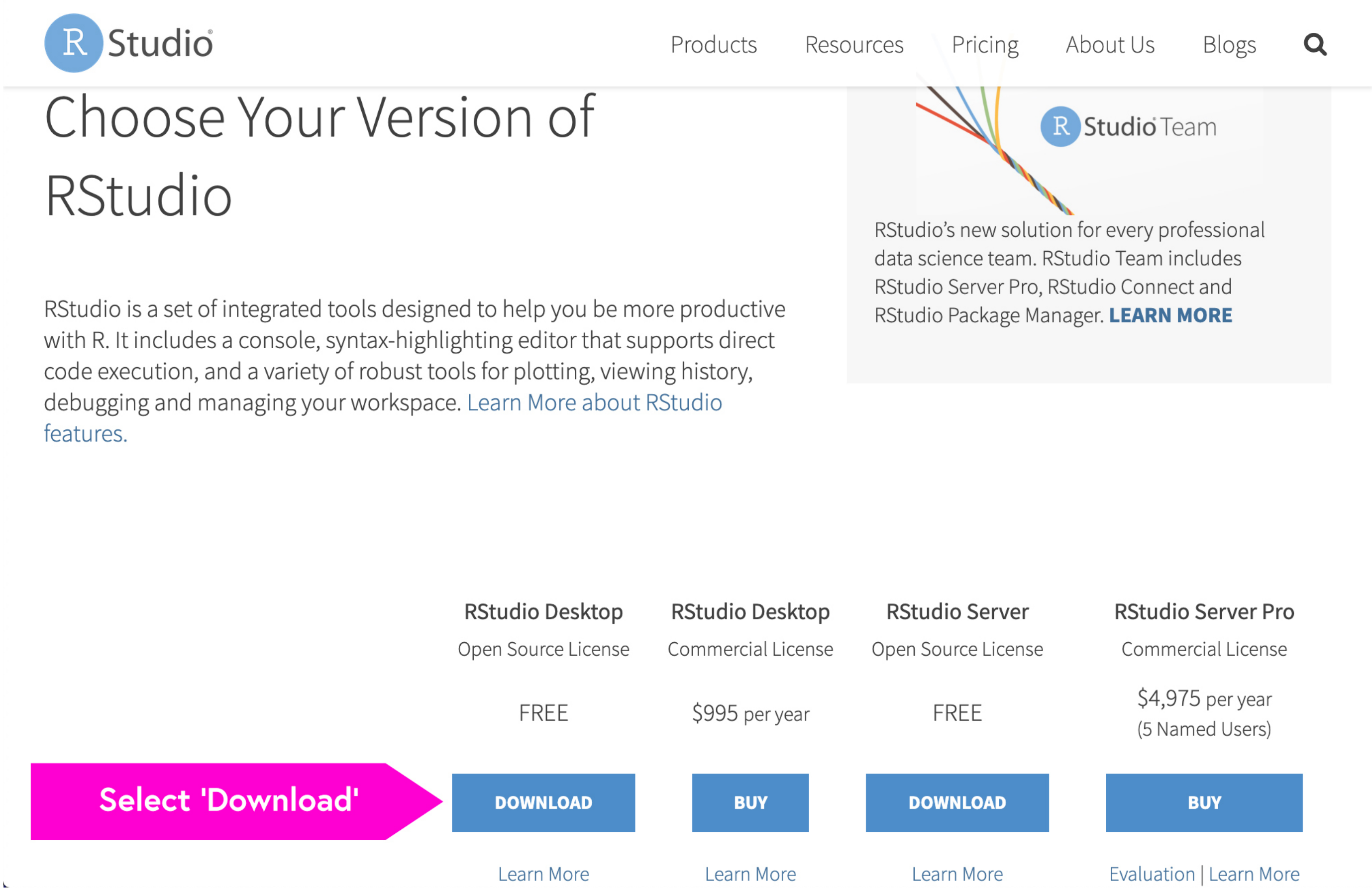
Download and install R

You can download the R language from the Comprehensive R Archive Network (https://cran.r-project.org/) or CRAN for short.

On the CRAN website, there are a list of links to download R for different operating systems, for example, Linux, (Mac) OS X and Windows. To download R to your computer, select the link that corresponds to your operating system. You can then install R.

Download and install RStudio

Once you have installed R, you are ready to install RStudio. For this course, the free version of RStudio is suitable. Get RStudio from the RStudio downloads page (https://www.rstudio.com/products/rstudio/download/). Select the **'Download'** link and then select the installer that corresponds to your operating system.



Choose Your Version of RStudio

RStudio is a set of integrated tools designed to help you be more productive with R. It includes a console, syntax-highlighting editor that supports direct code execution, and a variety of robust tools for plotting, viewing history, debugging and managing your workspace. Learn More about RStudio features.

RStudio's new solution for every professional data science team. RStudio Team includes RStudio Server Pro, RStudio Connect and RStudio Package Manager. **LEARN MORE**

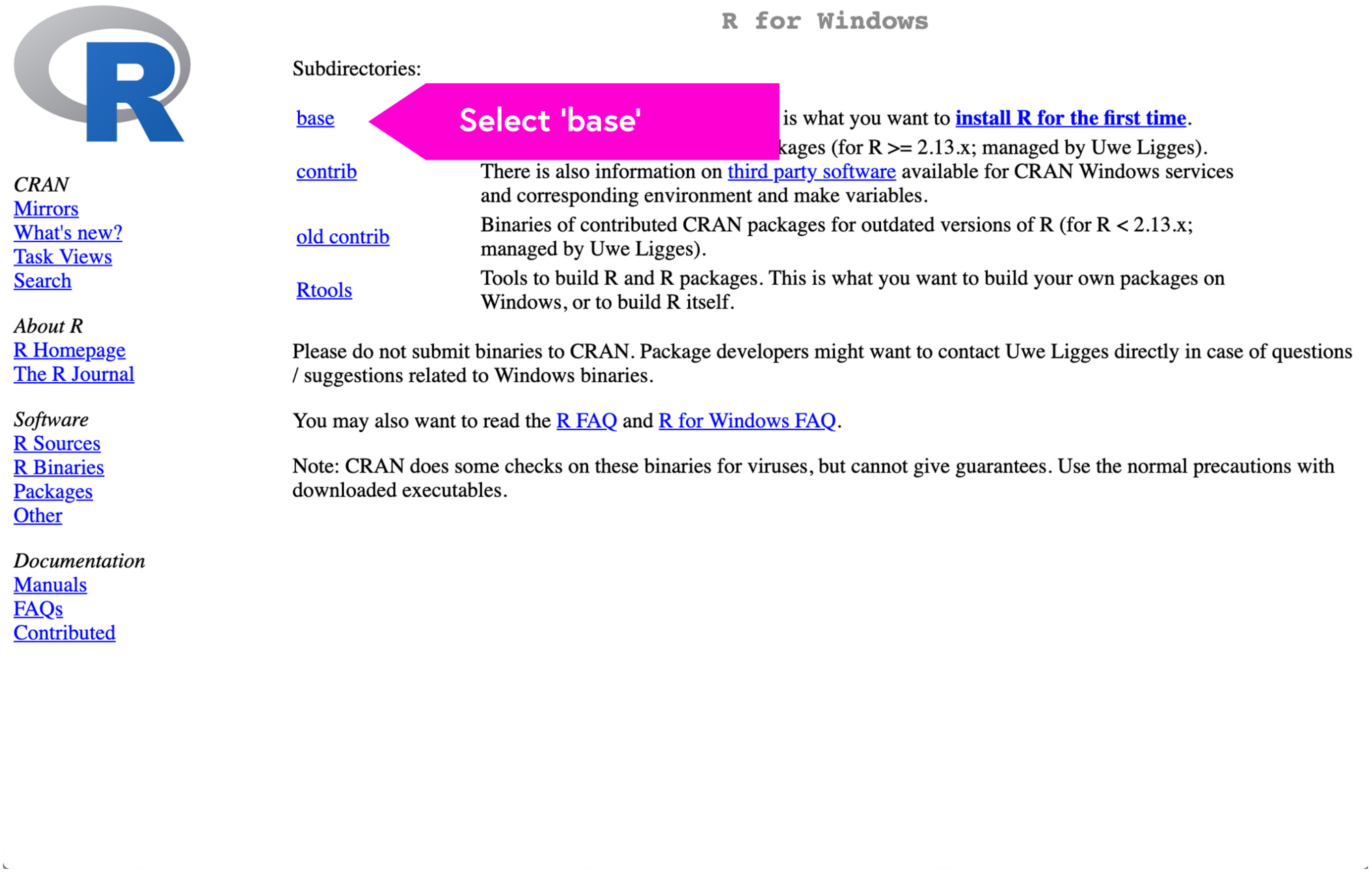
	RStudio Desktop	RStudio Desktop	RStudio Server	RStudio Server Pro
	Open Source License	Commercial License	Open Source License	Commercial License
	FREE	$995 per year	FREE	$4,975 per year (5 Named Users)
Select 'Download'	DOWNLOAD	BUY	DOWNLOAD	BUY
	Learn More	Learn More	Learn More	Evaluation | Learn More

View a larger version of the image of the RStudio website. (https://ugc.futurelearn.com/uploads/assets/fd/ef/fdef5070-1aa9-4f57-9e2f-3f036de1b624.jpg)

Although we recommend following these instructions, there are alternative ways of installing R and RStudio. If that's of interest to you, consider exploring How to install R on Windows, Mac OS X and Ubuntu (https://www.datacamp.com/community/tutorials/installing-R-windows-mac-ubuntu) to find out more.

R and RStudio on Windows

If you are a Windows user, the CRAN website will redirect you to a page with a list of subdirectories. Select the link called `base` .



R for Windows

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

Subdirectories:

base	is what you want to **install R for the first time**.
contrib	kages (for R >= 2.13.x; managed by Uwe Ligges). There is also information on third party software available for CRAN Windows services and corresponding environment and make variables.
old contrib	Binaries of contributed CRAN packages for outdated versions of R (for R < 2.13.x; managed by Uwe Ligges).
Rtools	Tools to build R and R packages. This is what you want to build your own packages on Windows, or to build R itself.

Please do not submit binaries to CRAN. Package developers might want to contact Uwe Ligges directly in case of questions / suggestions related to Windows binaries.

You may also want to read the R FAQ and R for Windows FAQ.

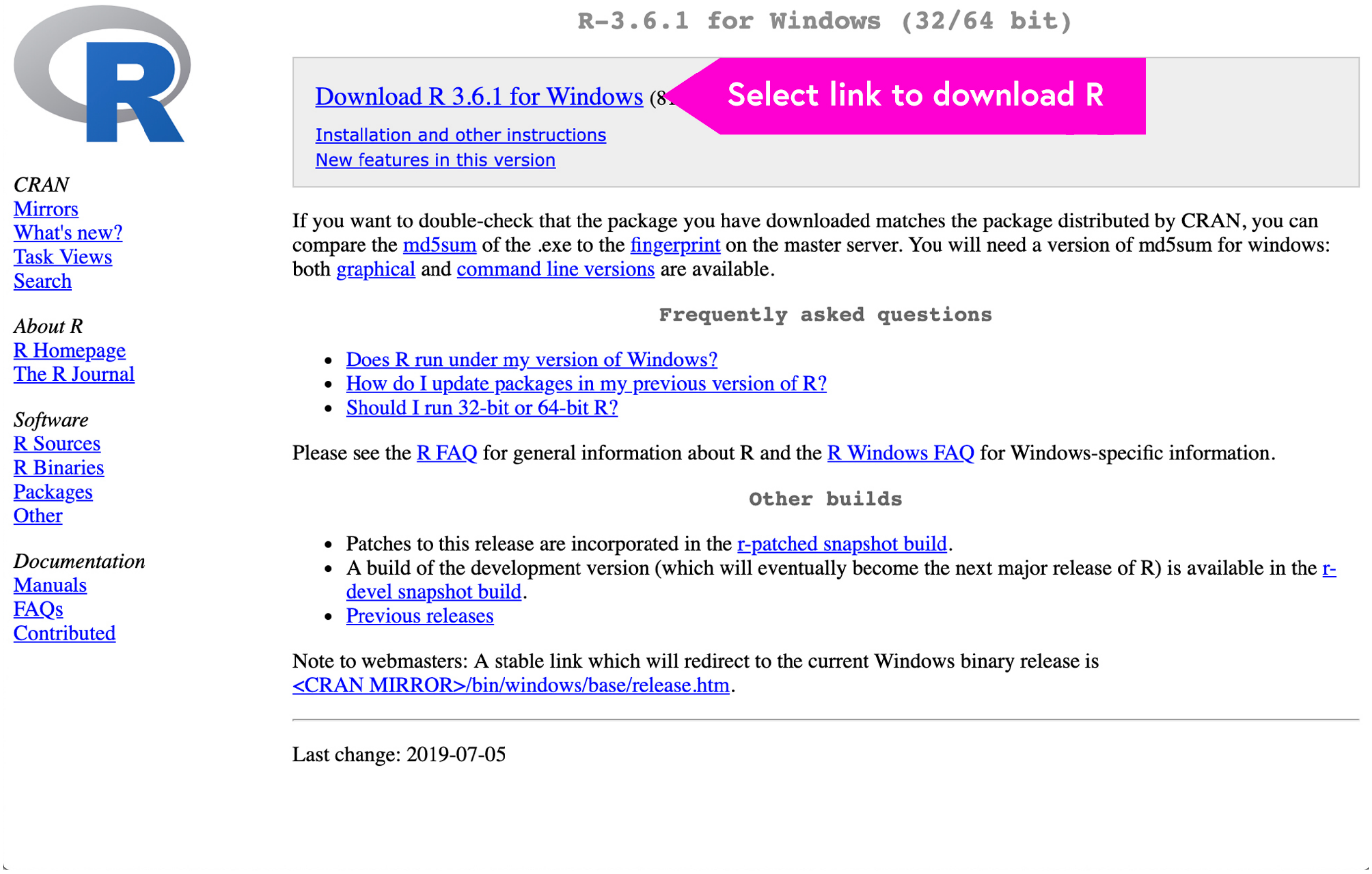
Note: CRAN does some checks on these binaries for viruses, but cannot give guarantees. Use the normal precautions with downloaded executables.

View a larger version of the image of the CRAN website for a base Windows installation of R.

(https://ugc.futurelearn.com/uploads/assets/60/bd/60bdf454-697f-4087-89a9-c16329d979f1.jpg)

You will be directed to a new page with a link to download R, for example, `Download R 3.6.x for Windows` . Select the link to download R for Windows.

Please note, the following image is an example only; the version listed on the website may differ.



R-3.6.1 for Windows (32/64 bit)

Download R 3.6.1 for Windows (8

Installation and other instructions

New features in this version

If you want to double-check that the package you have downloaded matches the package distributed by CRAN, you can compare the md5sum of the .exe to the fingerprint on the master server. You will need a version of md5sum for windows: both graphical and command line versions are available.

Frequently asked questions

- Does R run under my version of Windows?
- How do I update packages in my previous version of R?
- Should I run 32-bit or 64-bit R?

Please see the R FAQ for general information about R and the R Windows FAQ for Windows-specific information.

Other builds

- Patches to this release are incorporated in the r-patched snapshot build.
- A build of the development version (which will eventually become the next major release of R) is available in the r-devel snapshot build.
- Previous releases

Note to webmasters: A stable link which will redirect to the current Windows binary release is <CRAN MIRROR>/bin/windows/base/release.htm.

Last change: 2019-07-05

View a larger version of the image of the CRAN website for the latest version of the Windows installation of R.

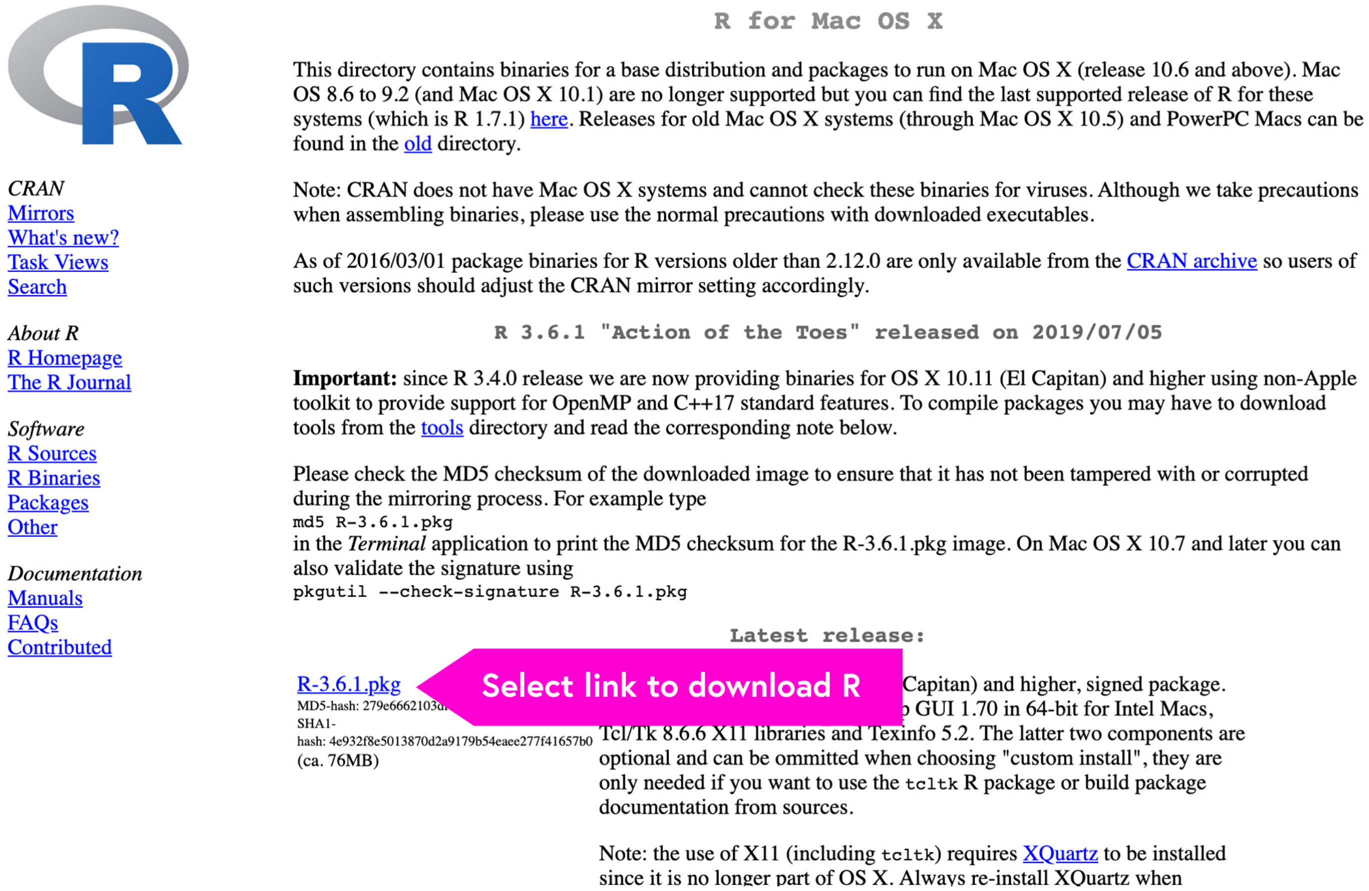
(https://ugc.futurelearn.com/uploads/assets/ca/96/ca96efe3-d671-4481-b59c-0dfd4fdc34be.jpg)

Once R has finished downloading, open the **.exe** file and work through the setup prompts. Once the setup is complete, download the latest version of the RStudio (https://www.rstudio.com/products/rstudio/download//#download) executable. Then, open the **'.exe'** file and work through the setup prompts.

R and RStudio on Mac OS X

If you are a Mac OS X user, the CRAN website will redirect you to a page with a list of releases. Select the link, for example, `R-3.6.1.pkg` to download the latest release.

Please note, the following image is an example only; the version listed on the website may differ.



R for Mac OS X

This directory contains binaries for a base distribution and packages to run on Mac OS X (release 10.6 and above). Mac OS 8.6 to 9.2 (and Mac OS X 10.1) are no longer supported but you can find the last supported release of R for these systems (which is R 1.7.1) here. Releases for old Mac OS X systems (through Mac OS X 10.5) and PowerPC Macs can be found in the old directory.

Note: CRAN does not have Mac OS X systems and cannot check these binaries for viruses. Although we take precautions when assembling binaries, please use the normal precautions with downloaded executables.

As of 2016/03/01 package binaries for R versions older than 2.12.0 are only available from the CRAN archive so users of such versions should adjust the CRAN mirror setting accordingly.

R 3.6.1 "Action of the Toes" released on 2019/07/05

Important: since R 3.4.0 release we are now providing binaries for OS X 10.11 (El Capitan) and higher using non-Apple toolkit to provide support for OpenMP and C++17 standard features. To compile packages you may have to download tools from the tools directory and read the corresponding note below.

Please check the MD5 checksum of the downloaded image to ensure that it has not been tampered with or corrupted during the mirroring process. For example type

```
md5 R-3.6.1.pkg
```

in the *Terminal* application to print the MD5 checksum for the R-3.6.1.pkg image. On Mac OS X 10.7 and later you can also validate the signature using

```
pkgutil --check-signature R-3.6.1.pkg
```

Latest release:

R-3.6.1.pkg
MD5-hash: 279e6662103d
SHA1-hash: 4e932f8e5013870d2a9179b54eaee277f41657b0
(ca. 76MB)

Select link to download R

Capitan) and higher, signed package.
GUI 1.70 in 64-bit for Intel Macs, Tcl/Tk 8.6.6 X11 libraries and Texinfo 5.2. The latter two components are optional and can be ommitted when choosing "custom install", they are only needed if you want to use the tcltk R package or build package documentation from sources.

Note: the use of X11 (including tcltk) requires XQuartz to be installed since it is no longer part of OS X. Always re-install XQuartz when

View a larger version of the image of the CRAN website for the latest version of the Mac OS X installation of R.

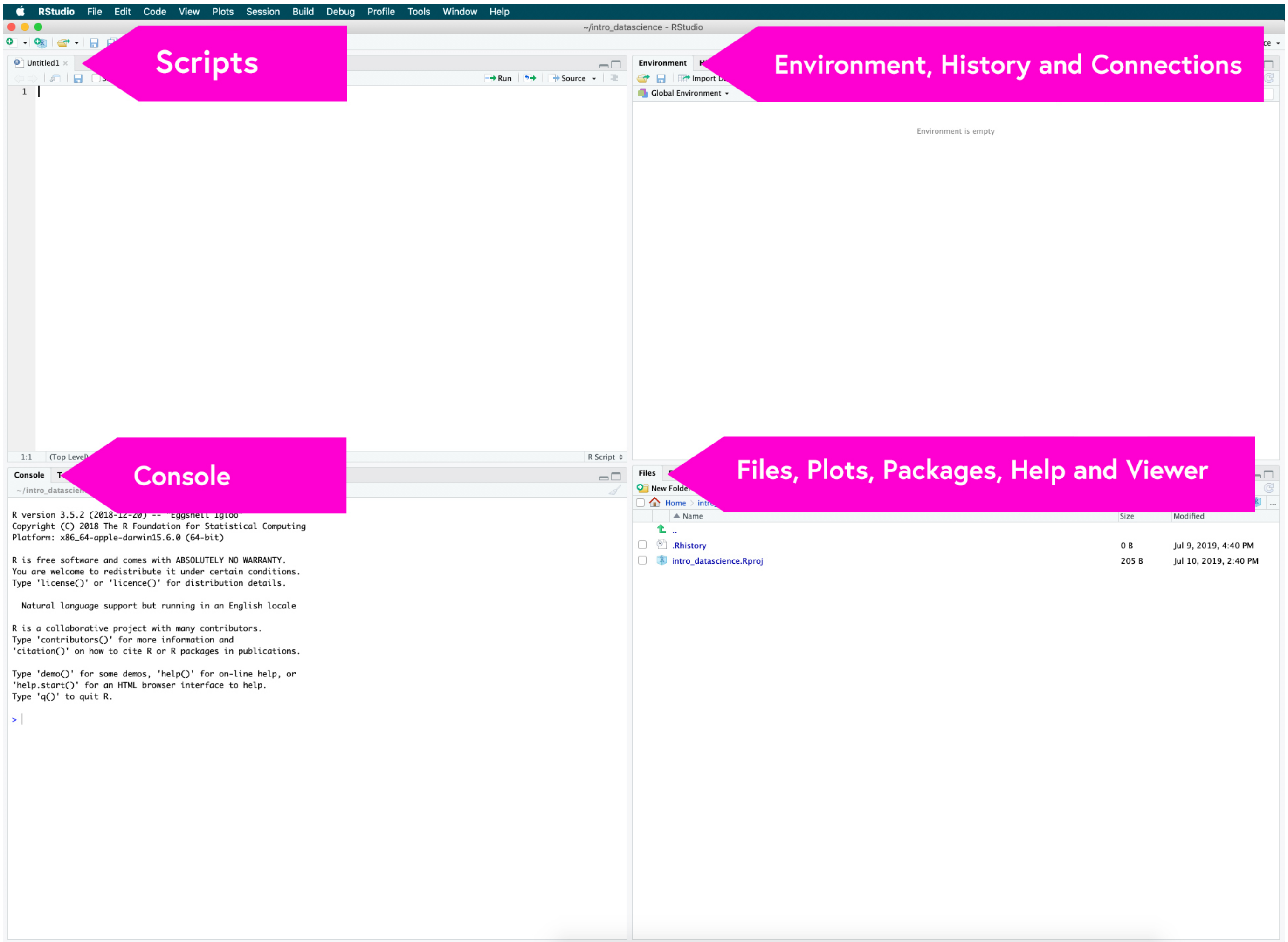
(https://ugc.futurelearn.com/uploads/assets/f3/cd/f3cdae1c-b009-4f72-b5de-1752a6d58e1b.jpg)

Once R has finished downloading, open the **'.pkg'** and then work through the setup prompts. Once the setup is complete, download the latest version of RStudio (https://www.rstudio.com/products/rstudio/download/#download). Then, open the **'.dmg'** file and then follow the prompts.

Check your installation!

Well done, you've installed R and RStudio on your computer. Open RStudio to check that everything has installed correctly and to explore the RStudio interface. The RStudio interface should be divided into **four** panes:

- **scripts** pane - the place where you write and save your R code.
- tab for the **console** - the place where your code gets run. You can write code directly in here, but you might forget the code you write, so it's a good idea to write your code in the **scripts** pane.
- tab for **Files**, **Plots**, **Packages**, **Help**, and a **Viewer** - files are listed here, just like a 'Finder' in Mac OS X or 'My Computer' in Windows.
- tab for **Environment**, **History** and **Connections** - things you create in R that are a result of code being run, such as data, objects and models are listed in the **Environment**.



```
R version 3.5.2 (2018-12-20) -- "Eggshell Igloo"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
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.

> 
```

View a larger version of the image of the RStudio interface. (https://ugc.futurelearn.com/uploads/assets/d1/93/d1931bcb-f20b-4f92-b58e-d12b9276ca4e.jpg)

Give it a go!

Continue to explore RStudio by making your way through this exercise - using R as a calculator. At the prompt of your **console** in RStudio, run the following code chunk:

```
1999 * 2 / 1000

(39 + 13 + 2) / 4

cos(pi)
```

Store the results of your computation as an object using the left arrow `<-` (The shortcut in RStudio is **Alt + -**).

Again, at the prompt of your **console** in RStudio, run the following code chunk:

```
my_variable <- 7 * 8
```

What’s it doing?

This is read as `my_variable` gets the value of `7*8` . In your RStudio the object called `my_variable` should be listed in the **Environment pane**.

Print the value of `my_variable` again by typing `my_variable` at the console.

Naming conventions are important!

Names of objects are important. You want them to be descriptive, and if you have multiple words in a name you need a way of dealing with that.

The convention we will use in this course is *snake_case*, where we separate multiple words by an underscore, and use all lower case. The most important thing though is to be consistent with your names.

Names are **case sensitive** and their **spelling** matters, otherwise R will not be able to correctly interpret the result.

Try running the following code:

```
my_Variable

my_varíable
```

Why does R give you an error?

R gives an error for the first line of code because there was an upper case `V` instead of a lower case `v`. Similarly, the second is an error because there was an accented `í` instead of an unaccented `i`.

Using functions

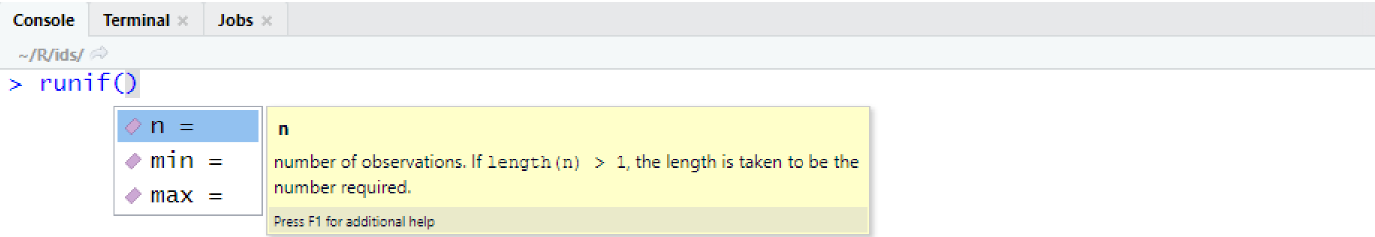
Most computations are performed by using *functions*. Functions take some input (arguments) and return an output. As an example, let's use R's built-in random number generator function `runif()` to generate 10 random numbers between 0 and 1.

If you type the number 10 as the first argument you will get 10 random numbers between 0 and 1.

```
runif(10)
```

```
##  [1] 0.52776777 0.70749135 0.10957173 0.74972638 0.72023903 0.07794937
##  [7] 0.01379601 0.98275840 0.40175891 0.44119403
```

If you type `runif()` in the RStudio console and then press **TAB** on your keyboard, a floating tooltip will appear that contains the names of the inputs to the function.



```
> runif()
n =
min =
max =
```

n

number of observations. If `length(n)` > 1, the length is taken to be the number required.

Press F1 for additional help

View a larger version of the image of the floating tooltip. (https://ugc.futurelearn.com/uploads/assets/a7/3c/a73cfdf5-7875-4cba-a71e-06577cdfb334.png)

You can be more explicit by specifying each input:

```
runif(n = 10, min = 0, max = 1)
```

```
##  [1] 0.53035215 0.31007780 0.03649348 0.10824814 0.26512682 0.33867215
##  [7] 0.71020387 0.26197506 0.33093167 0.01652366
```

By changing the values of the arguments, we can finally generate our numbers between 0 and 10.

```
runif(10, min = 0, max = 10)
```

```
##  [1] 4.1950369 6.2642353 8.3307466 1.3785580 7.0478302 8.9962923 0.4970724
##  [8] 1.8792311 4.4243016 0.4300972
```

And you can save the result to an object using the `<-` operator.

```
y <- runif(10, min  = 0, max = 10)

y
```

```
##  [1] 1.7155393 9.4846947 6.8817908 3.9466444 8.2147286 3.3620823 0.9818874
##  [8] 5.7382736 6.5781956 4.5633179
```

Computing summaries

So far, you've explored the basics of using R as a calculator and creating objects and calling functions.

Consider expanding your R vocabulary by using the following functions to compute the **mean** using the `mean()` function, the **variance** using the `var()` function, and the **range** using the `range()` function of the object `y` you just created. Again, you can do this from the **Console**.

Remember, if you are not sure how to use the function, type the name of it in the console and press **TAB** on your keyboard.

Tell us how you went

Within the **Comments**, consider sharing with other learners how you've gone with your early investigation of using R and computing summaries, and then respond to one or more of the following:

- **You may have noticed the numbers you generated at your R console are different from the ones presented earlier. Why is that?**
- **Look up the help file for the function `set.seed` by typing a question mark in front of it at the console: `?set.seed`.**
- **How could you use it to ensure you get the same random numbers?**

Don't forget to contribute to the discussion by reviewing the comments made by other learners, making sure you provide constructive feedback and commentary.

Remember you can also **'Like'** comments or follow other learners throughout the course.