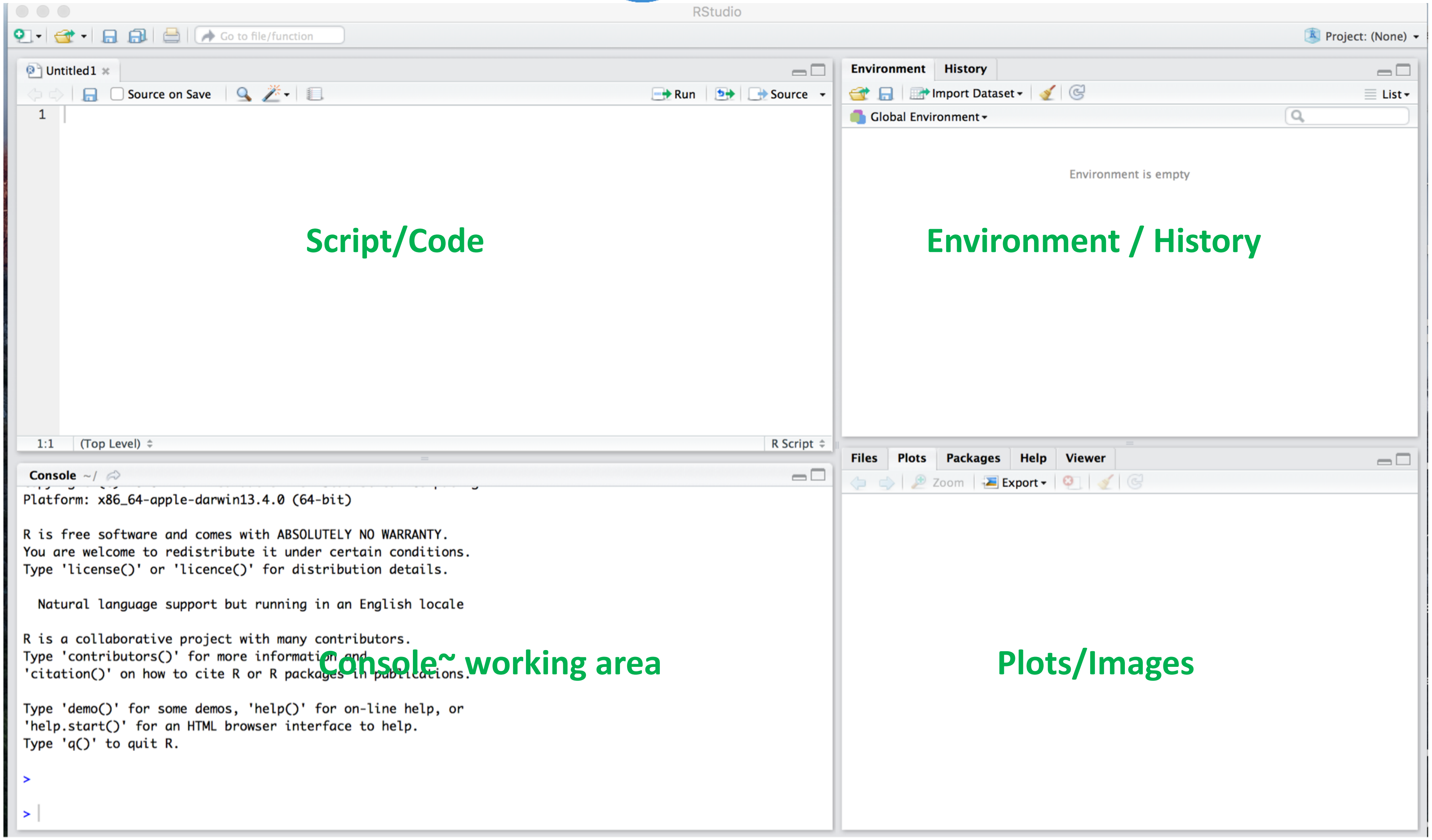
Introduction to R

By

Joubert Fayette
Ravin Poudel
Garrett Lab



```
Platform: x86_64-apple-darwin13.4.0 (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.

>

>
```

Calculating

> This symbol is called a prompt and is an invitation to put R to work.

Before creating 'objects', you can do simple calculations

```
7+2
5/2
6*9
1000-89
12/pi
rnorm(10) # default with mean 0, sd 1, 0-1
```

Creating objects

X <- 43 # This makes 43 the contents of the object X

# the arrow made from two characters <- is called the **gets arrow**

# A **pound sign** indicates a comment

X # Entering an object name prompts R to return the contents

x <- 23 **# R is case sensitive, so X and x are different objects**

Note: Variable names

- Variable names **should not begin** with numbers or symbols
  eg. 1x, %x
- Variable names **should not** contain blank spaces
  eg. Use back.pack, and not back pack

Creating objects

```
X <- 43 # This makes 43 the contents of the object X
# the arrow made from two characters <- is called the gets arrow
# A pound sign indicates a comment
X # Entering an object name prompts R to return the contents
x <- 23 # R is case sensitive, so X and x are different objects

X = 43 # the arrow or equal sign can be used to assign values

# Suppose you would like to create a combination of values
Y <- c(3,2,6) # Y contains 3, 2, and 6, using concatenation function c
Y # See for yourself
Y2 <- c(3,X,x)
Y2
Y3 <- c(Y,Y2)
Y3 # From here out, please check the contents of objects regularly
```

Vectors

Vectors are **one dimensional.**

```
# We have created vectors using c()
# c is the concatenation function
X <- c(4,2,7)
# Or using seq
X2 <- seq(3,6)
# We can join vectors together using cbind or rbind
X <- c(1,2,3)
Y <- c(7,8,9)
Z <- cbind(X,Y) # column bind
Z
Z2 <- rbind(X,Y) # row bind
Z2
```

Properties of a Vector

```
# For information about data structure
str(X)

# length of a vector
length(X)

#check if an object is a vector
is.vector(X)
```

Matrices

Matrices are two dimensional and contain elements of same category
Commonly used for mathematical and statistical analysis

```
# We can create matrices
help(matrix)
Mat1 <- matrix(1,ncol=3,nrow=4)
Mat1
X <- c(1,2,3)
Y <- c(7,8,9)
Z <- c(X,Y)
Z

Mat2 <- matrix(Z,ncol=2,byrow=F)
Mat2
```

Matrices

```
#check the properties of a matrix

length(Mat2)   # total number of elements

dim(Mat2)

str(Mat2)

attributes(Mat2)
```

Subscripting I

```
# Suppose you would like to work with a subset of an object
# Handy and powerful tool that can be used with any object type
X <- c(4,2,7) # We just made this
X[1]     #  Takes the first entry in the vector X, with the index
indicated using square brackets

Z15 <- X[1]  # The first entry in X can be put in another object Z15
X[3]
X[2:3] # We can also take more than one entry

X[c(2,3)] # This does the same thing

X[c(1,3)] # What does this do?

#Logical subscripting
X[X>2]
```

Subscripting II

```
Mat2 <- matrix(Z,ncol=2,byrow=F)  #Default
Mat2[1,2]      # 2-D matrices require two subscript dimensions
Mat2[2:3,1:2] # 1st subscript gives rows, then 2nd gives columns
Mat2[,2] # If we want all rows or all columns, the corresponding
subscript can be left off

#Negative subscripting
Mat2[,-2] # What happens here?
```

Question: Remove 2nd and 3rd rows in Mat2

```
Mat2[-c(2,3),]
```