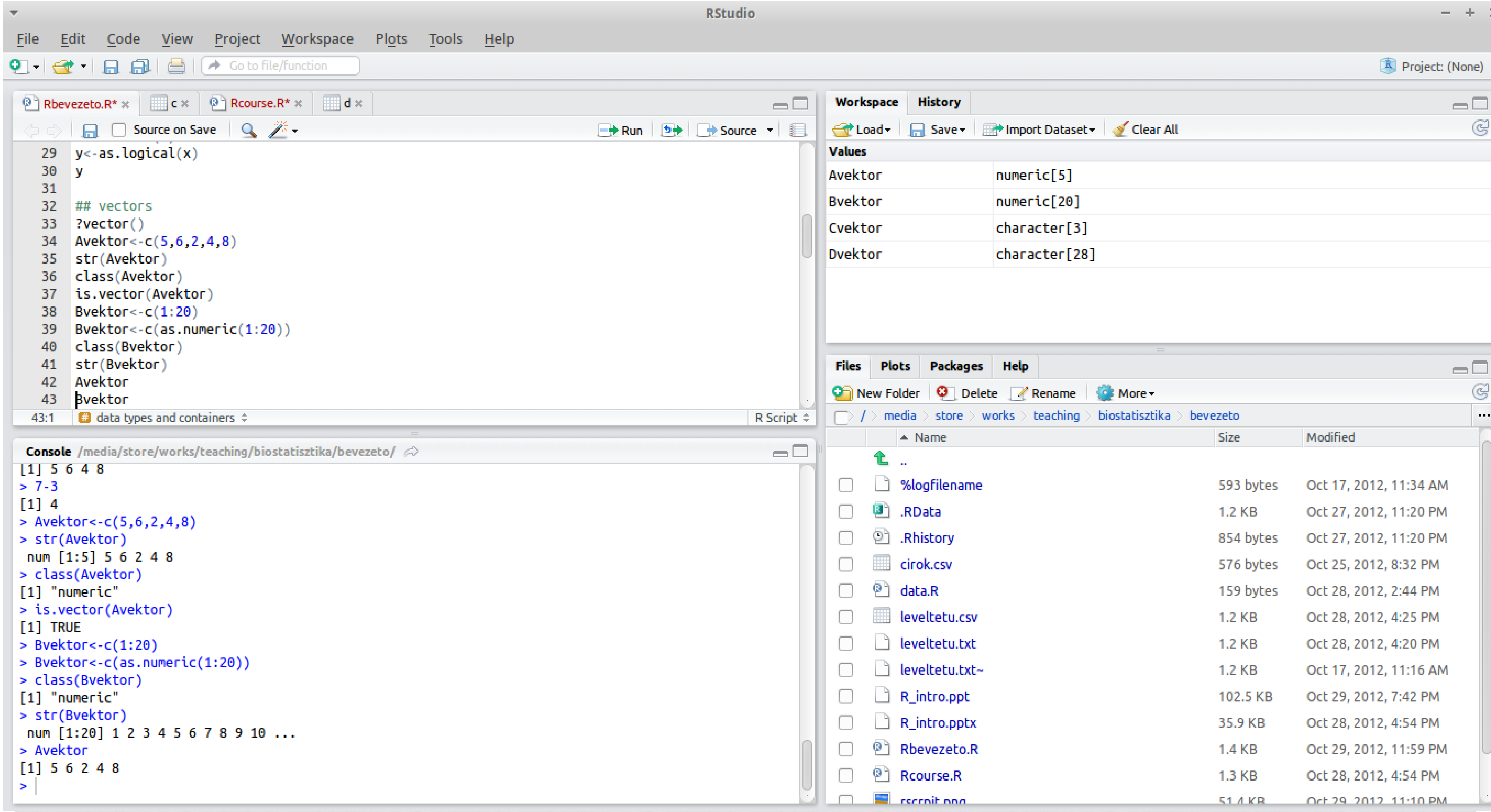
INTRODUCTION TO R

RSTUDIO



```
y<-as.logical(x)
y

## vectors
?vector()
Avektor<-c(5,6,2,4,8)
str(Avektor)
class(Avektor)
is.vector(Avektor)
Bvektor<-c(1:20)
Bvektor<-c(as.numeric(1:20))
class(Bvektor)
str(Bvektor)
Avektor
Bvektor
```

```
[1] 5 6 4 8
> 7-3
[1] 4
> Avektor<-c(5,6,2,4,8)
> str(Avektor)
 num [1:5] 5 6 2 4 8
> class(Avektor)
[1] "numeric"
> is.vector(Avektor)
[1] TRUE
> Bvektor<-c(1:20)
> Bvektor<-c(as.numeric(1:20))
> class(Bvektor)
[1] "numeric"
> str(Bvektor)
 num [1:20] 1 2 3 4 5 6 7 8 9 10 ...
> Avektor
[1] 5 6 2 4 8
> 
```

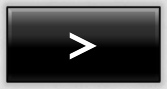
Values

Avektor	numeric[5]
Bvektor	numeric[20]
Cvektor	character[3]
Dvektor	character[28]

	Name	Size	Modified
	..		
□	%logfilename	593 bytes	Oct 17, 2012, 11:34 AM
□	.RData	1.2 KB	Oct 27, 2012, 11:20 PM
□	.Rhistory	854 bytes	Oct 27, 2012, 11:20 PM
□	cirok.csv	576 bytes	Oct 25, 2012, 8:32 PM
□	data.R	159 bytes	Oct 28, 2012, 2:44 PM
□	leveltetu.csv	1.2 KB	Oct 28, 2012, 4:25 PM
□	leveltetu.txt	1.2 KB	Oct 28, 2012, 4:20 PM
□	leveltetu.txt~	1.2 KB	Oct 17, 2012, 11:16 AM
□	R_intro.ppt	102.5 KB	Oct 29, 2012, 7:42 PM
□	R_intro.pptx	35.9 KB	Oct 28, 2012, 4:54 PM
□	Rbevezeto.R	1.4 KB	Oct 29, 2012, 11:59 PM
□	Rcourse.R	1.3 KB	Oct 28, 2012, 4:54 PM
□	rscript.png	51.4 KB	Oct 29, 2012, 11:10 PM

THE CONSOLE

- Input, oputput
- Prompt = where the computer waits for the input



```
>
```

- Syntax = grammar of a programming language
- [1]
- , = .   (0.8, 16.521)
- Ékezetes betűk

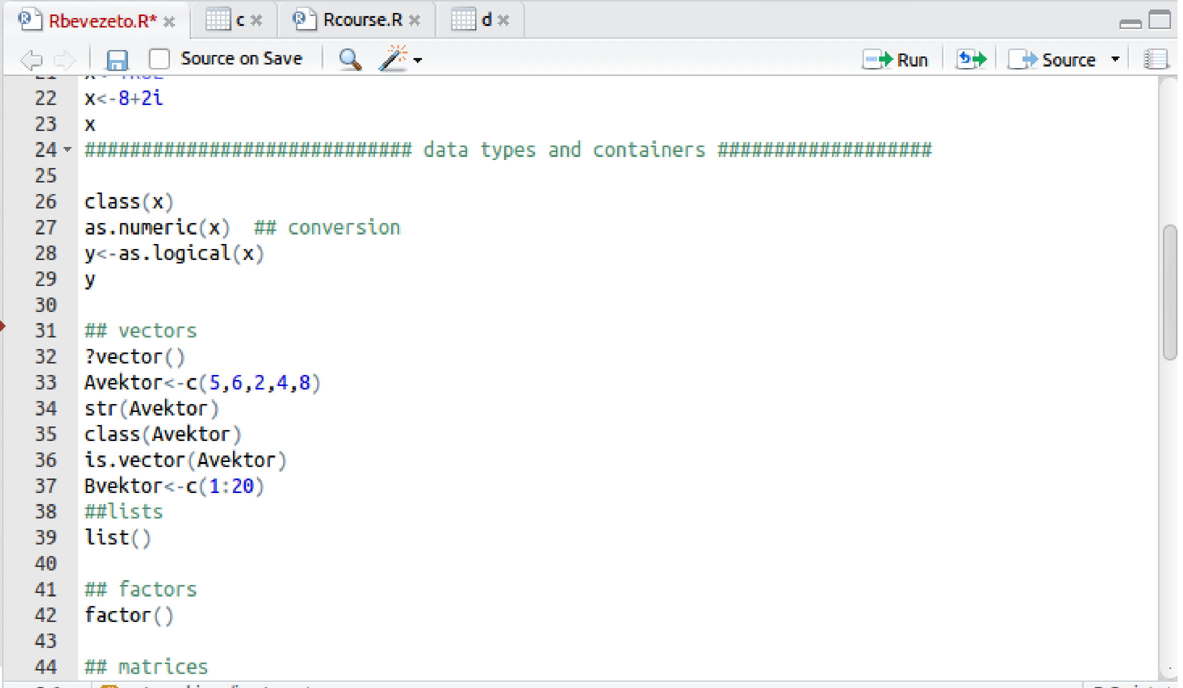
AN R SCRIPT

- setwd = set working directory (program reads and writes files here)
- Let’s set our working directory!
- getwd = where are we?
- Open/close/save
- Run - ctrl+enter
- # comment, titles
- Structuring the code
- An example

```
setwd()
getwd()
```

menu: tools/setwd

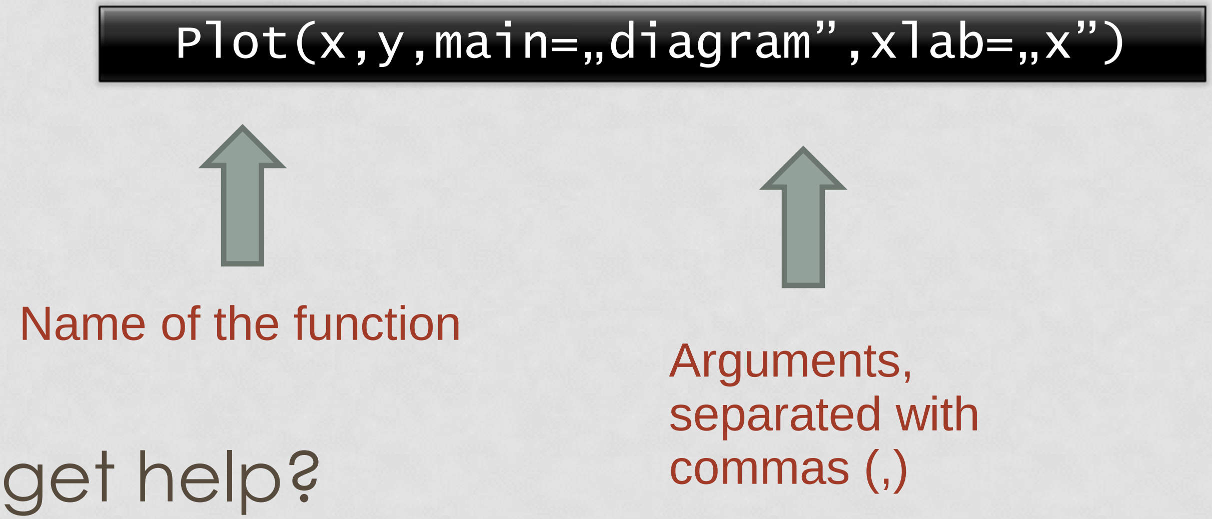
```
dir()
lists the files
in your working
directory
```



```
x<-8+2i
x
############################# data types and containers ###################

class(x)
as.numeric(x)  ## conversion
y<-as.logical(x)
y

## vectors
?vector()
Avektor<-c(5,6,2,4,8)
str(Avektor)
class(Avektor)
is.vector(Avektor)
Bvektor<-c(1:20)
##lists
list()

## factors
factor()

## matrices
```

BUILT IN FUNCTIONS, HELP

- What is a function in this context?



```
Plot(x,y,main=„diagram”,xlab=„x”)
```

Name of the function

Arguments, separated with commas (,)

- How to get help?

```
?plot
```

```
min(), max(),
mean(), print(),
ls(), dir(), str()….
```

- And there are many-many more..
- Moreover, we can define our own functions

R OBJECTS – CLASSES = TYPES

- What is an object in R?

- What's the difference?
- Size – memory usage
- Possible operations

- Features
- Capacity
- dimensions

Types of variables

• Complex	**2+i**
• Numeric	**8.435**
• Integer	**5**
• Logical	**TRUE**
• String	**„hello"**

Types of data containers

• Vectors	1D
• Lists	1D
• Factors	1D
• Matrixes	2D
• Data frames	anyD

```
summary()
str()
class()
is.matrix()
as.list()
```

```
class() –
gives the class
of the object

is.numeric()
is.logical()
– Test if the
object is
numeric, logical,
etc…

as.numeric()
as.complex()
– conversion
```

OPERATORS

Mathematical:

- + -
- × /
- ^
- %% - modulo

Logical:

- <,>,<=,>=,==,!=
- **! – not**
- **& - and**
- **| - or**

Assignment:

<-

VECTORS

- Can contain only objects from the same class (integer or string, etc..)
- Can contain vectors as elements
- Fast operations
- Low memory usage

`c()`

- How to get an element?

  indexing

`Avektor[1]` – simple selection for one element

`Avektor[2:3]` – selecting a sequence of elements (e.g. 2-3th or 40-80th)

`Avektor[Avektor>3]` – logical index, if something is true for an element, than it's selected

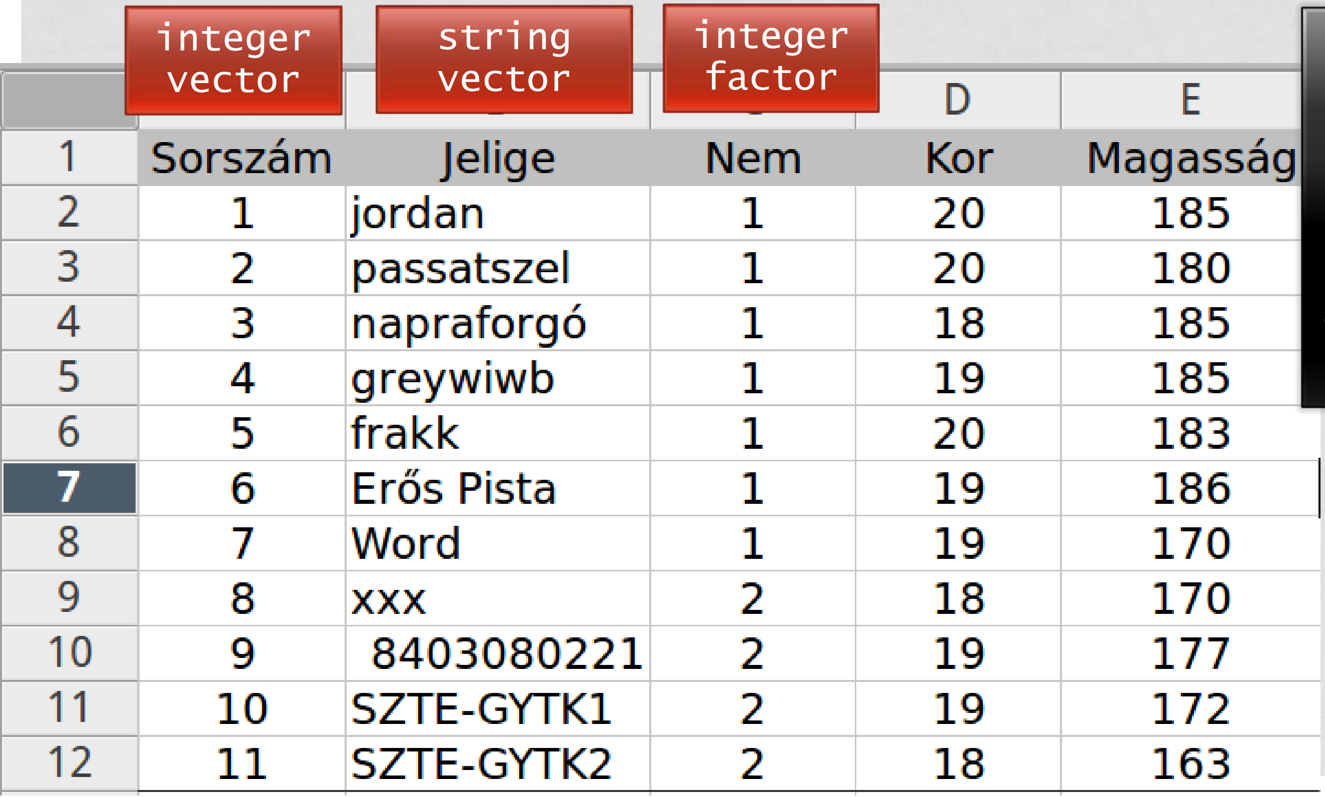
LISTS

- Can contain objects from different classes
- Even lists can be elements of other lists (embeded lists)
- Memory consuming
- Very useful!

```
list()

is.list()
as.list()
```

FACTORS

- For categorical data
- Integer e.g. grades
- Logical (TRUE - FALSE)
- String („M" – „F", „red"-"black"-"blue")



integer vector

string vector

integer factor

				D	E
1	Sorszám	Jelige	Nem	Kor	Magasság
2	1	jordan	1	20	185
3	2	passatszel	1	20	180
4	3	napraforgó	1	18	185
5	4	greywiwb	1	19	185
6	5	frakk	1	20	183
7	6	Erős Pista	1	19	186
8	7	Word	1	19	170
9	8	xxx	2	18	170
10	9	8403080221	2	19	177
11	10	SZTE-GYTK1	2	19	172
12	11	SZTE-GYTK2	2	18	163

```
factor()

factor(c(4,5,2,1,3,1))

is.factor()
as.factor()
```

MATRIXES

- Vectors with dimension attribute
- 2 dimensional tables
  with m rows, n columns -> m×n
- Each element has a „coordinate” created from the row and column number
- Columnwise creation in R

```
matrix()
dim()

cbind() –
columnbind,
binds vectors
together, they
have the same
number of
elements

rbind() –
rowbind
```

$$A_{(mxn)} = \begin{bmatrix} a11 & a12 & . & . & . & a1n \\ a21 & a22 & . & . & . & a2n \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ am1 & am2 & . & . & . & amn \end{bmatrix}$$

$$M = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -4 & -1 & -1 & -1 & -1 & 2 & 2 & 2 & 2 \\ 4 & -2 & -2 & -2 & -2 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & -1 & 0 & 1 & -1 & -1 & 1 \\ 0 & -2 & 0 & 2 & 0 & 1 & -1 & -1 & 1 \\ 0 & 0 & 1 & 0 & -1 & 1 & 1 & -1 & -1 \\ 0 & 0 & -2 & 0 & 2 & 1 & 1 & -1 & -1 \\ 0 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \end{pmatrix}$$

DATA FRAMES

- Usually we import them from files
- .csv ; = ,
- .txt

```
read.csv()
```

```
read.table()
```

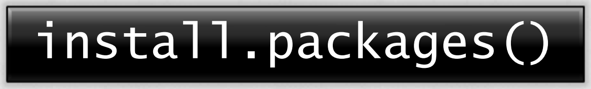
- Can have any number of dimensions
- How to imagine a multidimensional data frame?

1. **Plant abundance data (3 species)**
2. **In several quadrats (here 2)**
3. **Every month (2)**
4. **Every year**
5. Many sample sites

```
{{{2,3,4},{6,1,3}},{{1,5,4},{1,2,6}}}
```

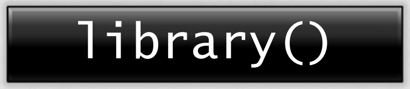
PACKAGES

- contain groups of functions for spacial purposes
- Install



```
install.packages()
```

- load



```
library()
```

HISTOGRAMS

`hist()`

BAR CHARTS

```
barplot()
```

```
barplot(table())
```