# Introduction to R

Part 1

Stefan Wyder
stefan.wyder@uzh.ch

Universität Zürich^UZH

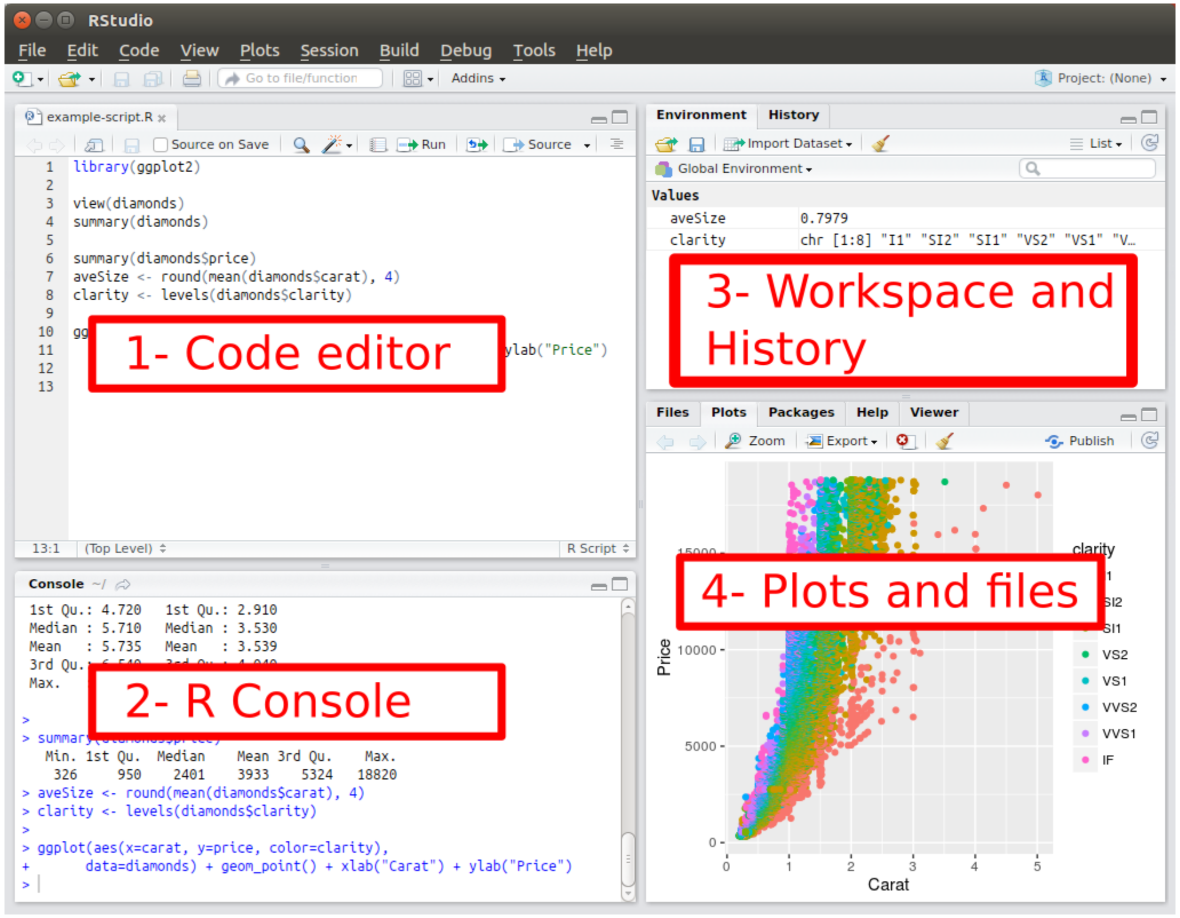
URPP
Evoluton

# R Studio

http://www.rstudio.com/

**Editor**
for writing longer pieces of code.

**R Console**
If you type code here, it is evaluated so that you get an answer



```
library(ggplot2)

view(diamonds)
summary(diamonds)

summary(diamonds$price)
aveSize <- round(mean(diamonds$carat), 4)
clarity <- levels(diamonds$clarity)

gg
                                                  ylab("Price")
```

```
aveSize     0.7979
clarity     chr [1:8] "I1" "SI2" "SI1" "VS2" "VS1" "V…
```

```
1st Qu.: 4.720   1st Qu.: 2.910
Median : 5.710   Median : 3.530
Mean   : 5.735   Mean   : 3.539
3rd Qu.
Max.

>
> summary
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    326     950    2401    3933    5324   18820
> aveSize <- round(mean(diamonds$carat), 4)
> clarity <- levels(diamonds$clarity)
>
> ggplot(aes(x=carat, y=price, color=clarity),
+        data=diamonds) + geom_point() + xlab("Carat") + ylab("Price")
>
```

will tell you things about objects (data objects and functions) in the workspace

Will display files, plots, packages, and help information

## Material & Links

http://milchmolch.github.io/R_Tutorial/

# Key points

- Assign values to variables `a <- 5`
- The primary data type in R is the vector
- Make a vector `c(1, 3, 3)`

- Extract values and subsections from data

```
vector[2]
dataframe[Row, Column]
dataframe$Height
datafr[datafr$Heigth > 10 & datafr$Weight < 5,]
subset(dataframe, Heigth > 10 & Weight < 5)
```

- All indices start with 1 (not 0)
- Be aware of vector recycling: how R handles vectors of unequal length

```
c(4,6) > c(2,4,6,8)
[1]  TRUE  TRUE FALSE FALSE
```

# Atomic classes

| | **Examples** |
|---|---|
| character | "a"  "URPP"<br>'a' 'URPP' |
| numeric (real or decimal) | 21  5.5 |
| integer | 2**L** |
| logical | TRUE  FALSE |

# Data structures

| | Examples |
|---|---|
| vector/ atomic vector | c(1, 2, 3)<br>c(TRUE, TRUE, FALSE, FALSE)<br>c("URPP", "Evolution", "UZH") |
| list | "Container"<br>list(1, "a", TRUE, "UZH") |
| matrix | Every column is of the same class<br><br>1 4 6 3 5<br>3 4 2 4 2<br>5 3 6 2 6 |
| data frame | "Spreadsheet table"<br>Every column can have a different class<br><br>"Joe"  blond  green  173<br>"Susan"  black  brown  168 |
| factors | categorical data -  fixed and known set of possible values (e.g. female, male)<br>ordered or unordered |

## Data Structures 2

| Dimensions | Homogeneous | Heterogeneous |
|---|---|---|
| 1-D | vector | list |
| 2-D | matrix | data frame |

# Data Structures 3

- Find class an object belongs to — class()
  internal data type — typeoff()

- Convert data type
  as.numeric(), as.character(), as.factor(), as.data.frame(), ...

## Clean Code

**Reproducibility:** Our end goal is not just to "do stuff" but to do it in a way that anyone can easily and exactly replicate our workflow and results.

**Readability** - Write programs for humans to read

Ideally your scripts should be short and readable, anyone should be able to pick them up and understand what it does.

- Apply **consistent** style (indentation/spacing, names, ...)
- **Functions** enable easy reuse within a project
- Break down problem into bite size pieces
  Corresponding with a single operation, single function
- Tell us what your function is doing, not how

# What makes a good function

- Functions enable easy reuse within a project

- It's short
  not longer than 1-2 screens

- Performs a single operation
  Break down problem into bite size pieces
  Corresponding with a single operation, single function

- Uses intuitive names Calc_Average_Per_Gene()
  Tell us what your function is doing, not how