Chapter 2: Creating Variables

Dr Megan L. Wood

2022-11-30

Chapter Overview:

1. **What is good code?**
2. **Writing code in RMarkdown**
3. **Creating your first variable**
4. **The dataframe**
5. **Creating variables cont.**

What is good code?

Good code is:

- well-annotated (using comments)
- clearly and consistently formatted
- reproducible
- sensibly named (including variables)

Another analyst (or future you) should be able to receive your script, understand what is going on and why and be able to run it.

Let's get coding!

Writing Code in RMarkdown

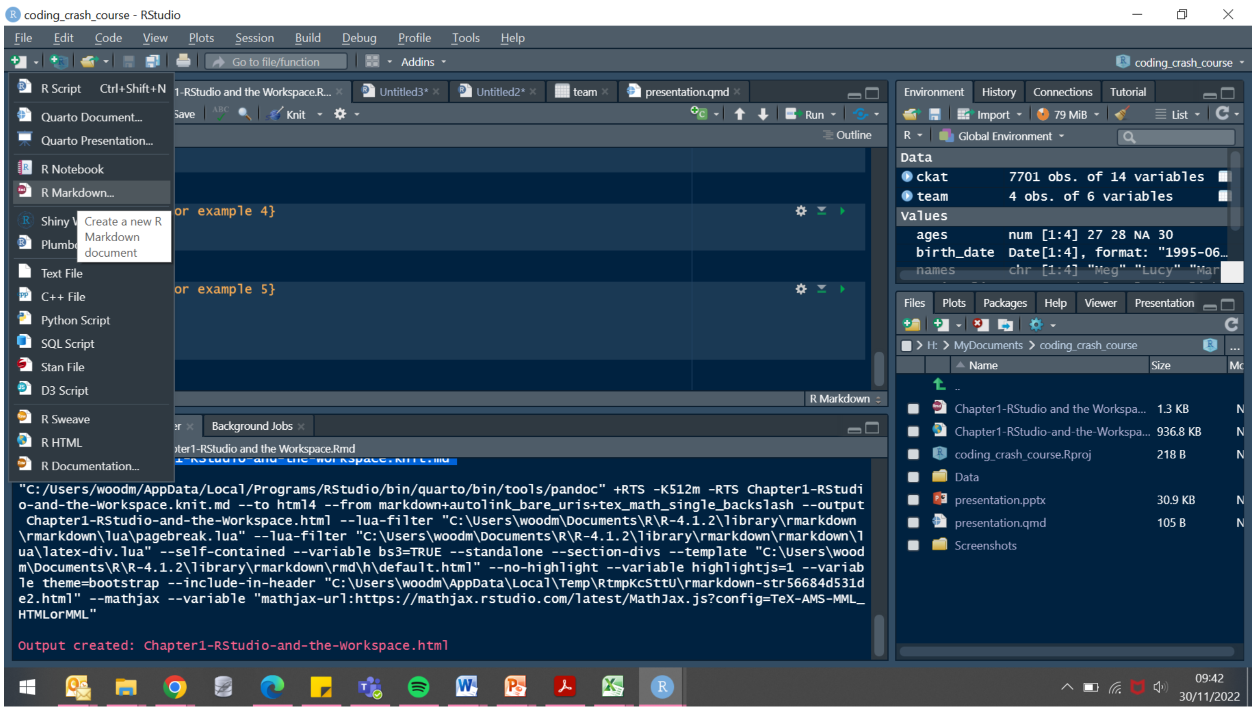
Now, let's get started in creating your first script!

Go to: File –> NewFile –> RMarkdown…

Markdown is a special type of script that can be used to create reports or word documents that includes code and output. This is easier than using a standard script, as it encourages you to annotate your work appropriately. It also means you can write up any results from your analysis directly in the script where it is being run!

Markdown is slightly different in that it uses **code chunks**.

The below is an example of a code chunk:



```
or example 4}
or example 5}
```

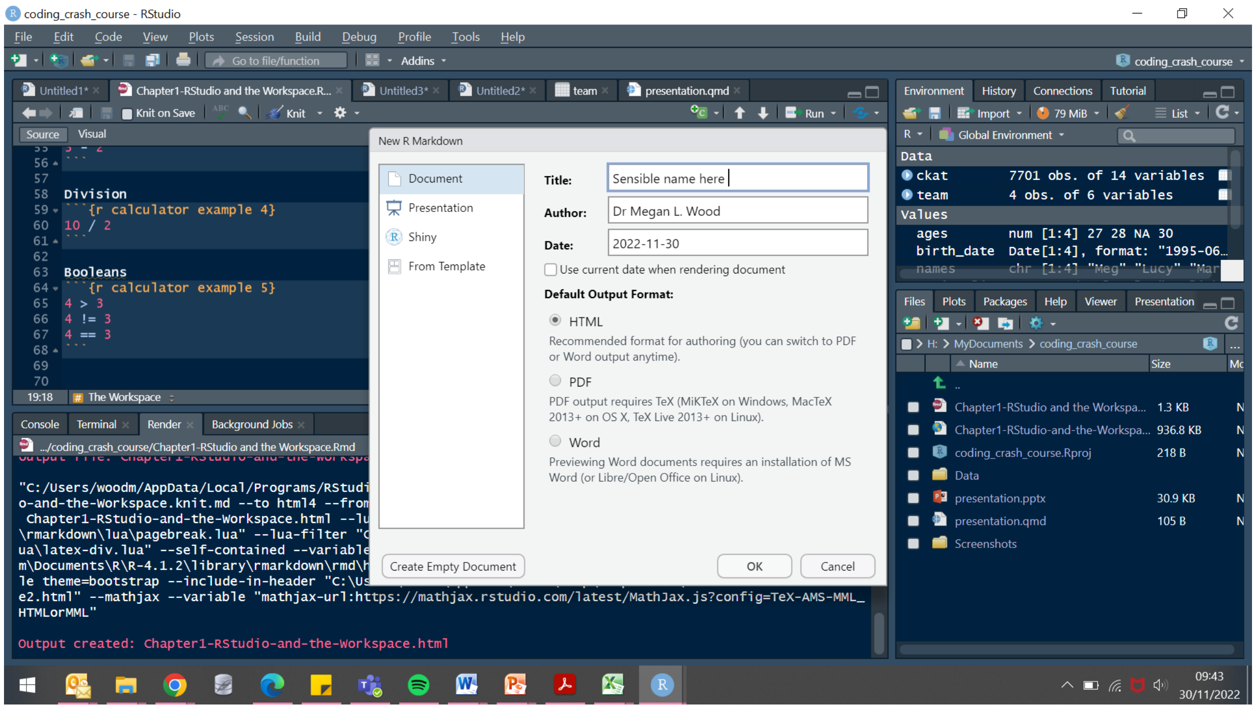
```
"C:/Users/woodm/AppData/Local/Programs/RStudio/bin/quarto/bin/tools/pandoc" +RTS -K512m -RTS Chapter1-RStudi
o-and-the-Workspace.knit.md --to html4 --from markdown+autolink_bare_uris+tex_math_single_backslash --output
 Chapter1-RStudio-and-the-Workspace.html --lua-filter "C:\Users\woodm\Documents\R\R-4.1.2\library\rmarkdown
\rmarkdown\lua\pagebreak.lua" --lua-filter "C:\Users\woodm\Documents\R\R-4.1.2\library\rmarkdown\rmarkdown\l
ua\latex-div.lua" --self-contained --variable bs3=TRUE --standalone --section-divs --template "C:\Users\wood
m\Documents\R\R-4.1.2\library\rmarkdown\rmd\h\default.html" --no-highlight --variable highlightjs=1 --variab
le theme=bootstrap --include-in-header "C:\Users\woodm\AppData\Local\Temp\RtmpKcSttU\rmarkdown-str56684d531d
e2.html" --mathjax --variable "mathjax-url:https://mathjax.rstudio.com/latest/MathJax.js?config=TeX-AMS-MML_
HTMLorMML"

Output created: Chapter1-RStudio-and-the-Workspace.html
```

```
Data
ckat        7701 obs. of 14 variables
team        4 obs. of 6 variables
Values
  ages        num [1:4] 27 28 NA 30
  birth_date  Date[1:4], format: "1995-06…
  names       chr [1:4] "Meg" "Lucy" "Mar…
```

Name	Size
..	
Chapter1-RStudio and the Workspa...	1.3 KB
Chapter1-RStudio-and-the-Workspa...	936.8 KB
coding_crash_course.Rproj	218 B
Data	
presentation.pptx	30.9 KB
presentation.qmd	105 B
Screenshots	

Figure 1: image_here.


coding_crash_course - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Untitled1*
Chapter1-RStudio and the Workspace.R...
Untitled3*
Untitled2*
team
presentation.qmd
Knit on Save
Knit
Run
Source Visual
19:18
The Workspace
Console Terminal Render Background Jobs
.../coding_crash_course/Chapter1-RStudio and the Workspace.Rmd
Environment History Connections Tutorial
Global Environment
Files Plots Packages Help Viewer Presentation
09:43
30/11/2022

~~~
Division
```{r calculator example 4}
10 / 2
~~~

Booleans
```{r calculator example 5}
4 > 3
4 != 3
4 == 3
```
```

New R Markdown

- Document
- Presentation
- Shiny
- From Template

Title: Sensible name here

Author: Dr Megan L. Wood

Date: 2022-11-30

☐ Use current date when rendering document

Default Output Format:

● HTML
Recommended format for authoring (you can switch to PDF or Word output anytime).

○ PDF
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

○ Word
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

Create Empty Document | OK | Cancel

Figure 2: image_here.
```

```r
fruit <- c("apple", "pear", "orange", "banana") # create fruit variable
```

Anything written within a chunk will be treated as code. As you can see, we can use commenting within a chunk to explain exactly what we are doing. This is useful when we start to create more complex chunks of code.

To create a code chunk:

Three backticks { r a sensible name for your chunk } Three backticks

The whole chunk can then be executed using the "Play" button on the right-hand side. We can also run individual parts of the chunk by highlighting the bit we want to run and clicking "Run".

Have a go creating a code chunk as below:

```r
print("Hello World!")
```

Creating your first variable

We are going to firstly create a "string" variable

The standard format for creating a variable:

object <- **function**

Think of the arrow as *is created from*

String variables

String variables contain characters - this can be names, categories etc.

```r
names <- c("Marc", "Sally", "Zayn", "Maryam")
```

Numeric variables

These take a numeric format (e.g., 4, 10) but can also include decimal points too (e.g., 6.4325)

Notice here that we can include missing data by using the **NA** placeholder.

```r
ages <- c(27, 28, NA, 30)
```

The Dataframe

A dataframe is a two-dimensional data structure that takes the form of rows and columns. We can combine the two individual variables we have just created into a single dataframe.

```r
team <- data.frame(names, ages)
```

Let's have a look at the team dataframe that we have just created.

```r
team
```

Creating variables cont.

We can also create new variables from existing ones.

```r
# Add in a new variable to existing df
team$years_since_graduation <- c(0, 1, NA, 2)

# Create a new variable from existing variable
team$age_at_graduation <- team$ages - team$years_since_graduation
```

Date data

As well as strings and numeric variables, R also recognises dates as unique variable types.

```r
# Create a date variable
birth_date <- as.Date(c("1995-06-28", "1994-10-04", NA, "1992-05-03"))

# Add to existing df
team$birth_date <- birth_date
```

Factors

These are variables which use numbers or strings to represent different groups of data (e.g., experimental vs. control; male vs. female; nationality). Often, factors are preferred when comparing groups in statistical modelling:

```r
# Create a new variable called "nationality"
nationality <- c(1, 1, 2, 1)
# Create as a factor and give labels
nationality <- factor(nationality, levels = 1:2, labels = c("English", "Scottish"))

# Add the factor back into the dataframe
team <-data.frame(team, nationality)
```

rbind and cbind

These are useful functions to add in additional rows or columns to an existing dataframe.

To add a new row, we need to add values for all the variables in the dataframe (in the correct order!).

```r
new_team_member <- c("Kris", 32, 3, 27, "1990-05-23", "Scottish")
```

We can then "bind" the rows and overwrite the existing "team" dataframe.

```r
team <- rbind(team, new_team_member)
```

Similarly, we can use this function to add new columns using **cbind**. In the same way, we can add all the values for each row of our new column to provide information of each participant's degree.

```
degree <- c("Psychology", "Sports Science", "Psychology", "Linguistics", "Medicine", "Neuroscience")
```

Now let's take another look at our dataframe. It should look something like this:

```
team
```

```
##    names ages years_since_graduation age_at_graduation birth_date nationality
## 1   Marc   27                      0                27 1995-06-28     English
## 2  Sally   28                      1                27 1994-10-04     English
## 3   Zayn <NA>                   <NA>              <NA>       <NA>    Scottish
## 4 Maryam   30                      2                28 1992-05-03     English
## 5   Kris   32                      3                27 1990-05-23    Scottish
```