Norwegian Meteorological Institute

GitHub & advanced R

Kajsa Parding, Lene Østvand, Hans Olav Haugen

14.09.2022

Presentation for Bangladesh Meteorological Institute

What is GitHub?

http://www.github.com

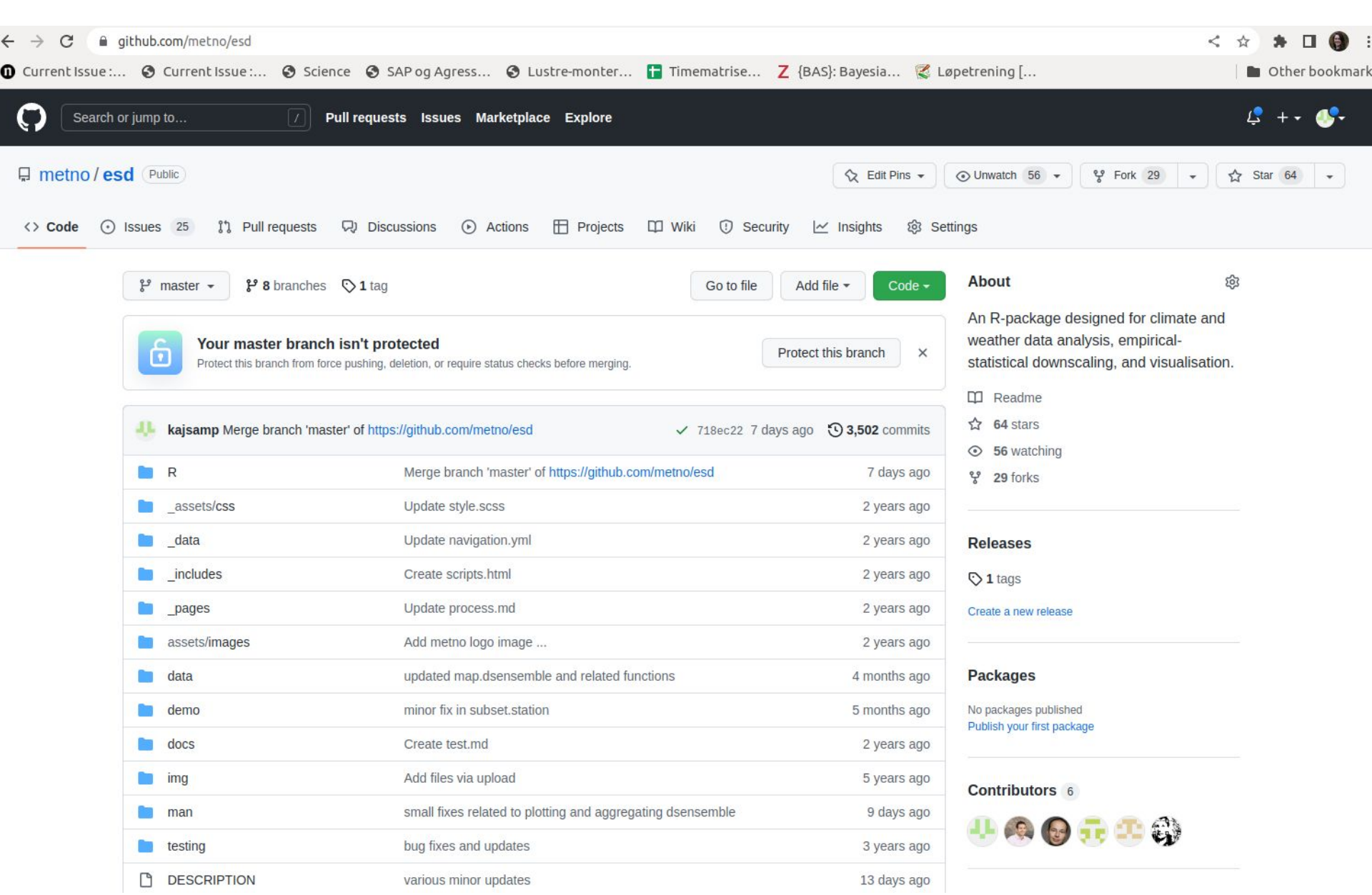
GitHub is a website and tool to store and share code.

You can share code and data with one or several collaborators, but it will also be open to the world.

The location where you put your code on GitHub is called a repository.

GitHub is host to many R-packages, including the ‘esd’ package.

https://github.com/metno/esd



Norwegian Meteorological Institute

https://github.com/metno/BMD

Exercises

1. Register and log in at https://github.com

2. Download code and documents from

   https://github.com/metno/BMD

3. Upload a script of your own (that you are comfortable sharing).

Working with GitHub locally

You can access GitHub via the terminal window

Basic commands

`clone`	download a repository to your computer
`pull`	update the downloaded repository
`add`	add the changes that you have made locally
`commit`	prepare the added changes to be uploaded
`push`	upload your local version to the Github repository

Exercises

1. Make a directory called git on your computer

```
mkdir git
cd git
```

2. Clone the BMD repository

```
clone https://github.com/metno/BMD.git
```

3. Update the local version of the BMD repository

```
pull BMD
```

GitHub resources

Tutorial
https://docs.github.com/en/get-started/quickstart/hello-world

GitHub book
https://git-scm.com/book/en/v2

R-training

https://github.com/metno/BMD/tree/master/Exercises

Writing functions in R

A function is a set of instructions, like a recipe.
R comes with many built-in functions.

```
## print: Print input
print('Print this input!')

## seq: Create sequence from 2 to 12
x <- seq(2,12)
print(x)

## sum: Calculate the sum of the sequence x
print(sum(x))

## paste: Attach character strings to each other
print(paste("Hello", "world!", sep=" "))
```

Writing functions in R

The basic syntax of a function in R looks like this:

```
function_name <- function(input1, input2, …) {
    function body
}
```

Writing functions in R

The basic syntax of a function in R looks like this:

```
function_name <- function(input1, input2, …) {
    function body
}
```

```
## Example 1: calculate a*b and print the results
f1 <- function(a = 3, b = 6) {
   result <- a * b
   print(result)
}

f1(53, 67)
```

Writing functions in R

The basic syntax of a function in R looks like this:

```
function_name <- function(input1, input2, …) {
    function body
}
```

```
## Example 2: calculate a*b and return the results
f2 <- function(a = 3, b = 6) {
   result <- a * b
   return(result)
}

y <- f2(53, 67)
```

Exercises

1. Write a function that prints the phrase: “Hello world!”

2. Write a function that attach two words together and print them. Get your function to say “Hello world!”.

3. Write a function that calculates the mean value of two numbers.

4. Write a function that reads a file and extracts some of the data.

5. Change the function from step 4 to also produce a plot.

6. Find a script that you have written. Make a part of the code into a function and use it in the script.

Using ‘for’ to create loops

Loops can be used to repeat tasks with minor variations. The basic syntax of the for-loop looks like this:

```
for(index in vector) {
  loop body
}
```

Using ‘for’ to create loops

Loops can be used to repeat tasks with minor variations. The basic syntax of the for-loop looks like this:

```
for(index in vector) {
  loop body
}

## Example 1
for(x in 1:10) {
  print(x)
}
```

Using ‘for’ to create loops

Loops can be used to repeat tasks with minor variations. The basic syntax of the for-loop looks like this:

```
for(index in vector) {
  loop body
}

## Example 2 - simple for loop that prints years
years <- seq(2021, 2050)
for(y in years) {
  txt <- paste(“The year was”, y)
  print(txt)
}
```

Using ‘for’ to create loops

You can put a loop inside a function, or a function inside a loop.

```
## Example 3. Loop in function
myforloop <- function(start, end) {
  for(i in start:end) {
    print(i)
  }
}
```

Using ‘for’ to create loops

You can put a loop inside a function, or a function inside a loop.

```
## Example 4. Loops in functions
mysum(x) {
  xsum <- 0
  for(xi in x) {
    xsum <- xsum + xi
  }
  return(xsum)
}

y <- c(34,56,87,98,65)
mysum(y)
sum(y)
```

Using ‘if’ statements

Use if statements to choose what to do based on the input.

```
if(TRUE) {
   print(“The statement is true.”)
} else {
   print(“The statement is false.”)
}
```

Using ‘if’ statements

Use if statements to choose what to do based on the input.

```
# Specify the value of x
x <- 0

if(x < 0) {
  print("x is lower than zero.")
} else {
  print("x is higher or equal to zero.")
}
```

Using ‘if’ statements

Use if statements to choose what to do based on the input.

```
file <- '/path/to/file/filename.nc'

if(file.exists(file)) {
   print("The file exists. Opening file.")
   x <- retrieve(file)
} else {
   print("The file does not exist.")
}
```


Norwegian
Meteorological
Institute