`git` for Youngsters

Book 1

Figure 1: `git` for Youngsters, book 1

#	Description
1	Setup git
2	Register and sign in
3	Install Discord
4	Create a repository
5	Monologue
6	Fairytale on `master`

#	Description
7	'Hello world' on `master`

Contents

Forword

This is a book about `git` and Codeberg for youngsters. `git` is a version control manager. Codeberg (https://codeberg.org) is website that allows one to use `git` with multiple people. This book teaches how to use `git` and Codeberg

About this book

This book is a bit sloppy. There are typosd and th*e layout a*lso `is not alw`**ays pretty**. Because this book is hosted on Codeberg, everyone can help at making this book a bit less sloppy.

1. Setup git

In this chapter, we are going to install the program to work with `git`.

`git` is a version control system

1.1 Exercise 1

First, we check if `git` is already installed.

1.1.1 Exercise 1 for Linux or Mac

If you have Linux or Mac, open a Terminal and type:

```
git --help
```

If you get the manual of `git`, go to exercise 3

`git` is created by the same programmer that made Linux

1.1.2 Exercise 1 for Windows

If you have Windows, press the Windows key (usually between the left CTRL and left ALT). Type 'git bash'. If the program 'Git Bash' shows up, it is installed and you can go to exercise 3.

1.2 Exercise 2

We need to install `git`

1.2.1 Exercise 2 for Linux

If you have Linux, start a Terminal and do

```
sudo apt install git
```

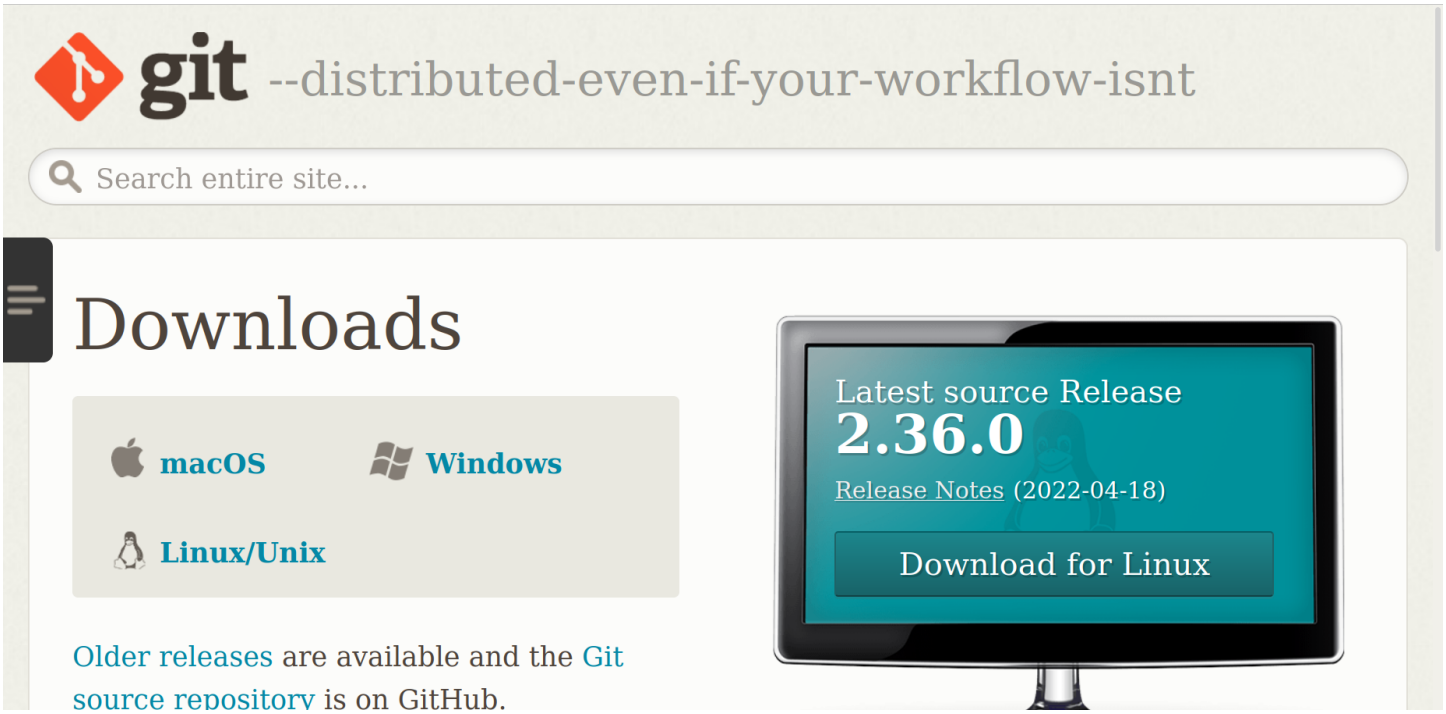
Linux is easiest :-)

Type in the root password and `git` is installed.

Go to exercise 1 to check that it worked.

1.2.2 Exercise 2 for Windows

If you have Windows or Mac, install Git Bash at https://git-scm.com/downloads.



When installing, use all default settings.

Go to exercise 1 to check that it worked.

1.3 Exercise 3

We need to tell `git` who we are.

Start Git Bash or a terminal and tell `git` who you are:

```
git config --global user.name [username]
git config --global user.email [email]
```

where `[username]` is your Codeberg username, and `[email]` is the email you set in your Codeberg profile. As an example:

```
git config --global user.name richelbilderbeek
git config --global user.email richel@richelbilderbeek.nl
```

You only need to do this once

1.4 Final exercise

In the Terminal or Git Bash, prove you told `git` who you are:

Type

```
git config --global user.name
```

and your Codeberg username should appear.

Type

```
git config --global user.email
```

and your Codeberg email should appear.

As an example, this is how it can look like:

```
$ git config --global user.name
richelbilderbeek
$ git config --global user.email
richel@richelbilderbeek.nl
```

Luckily, this only needs to be done once

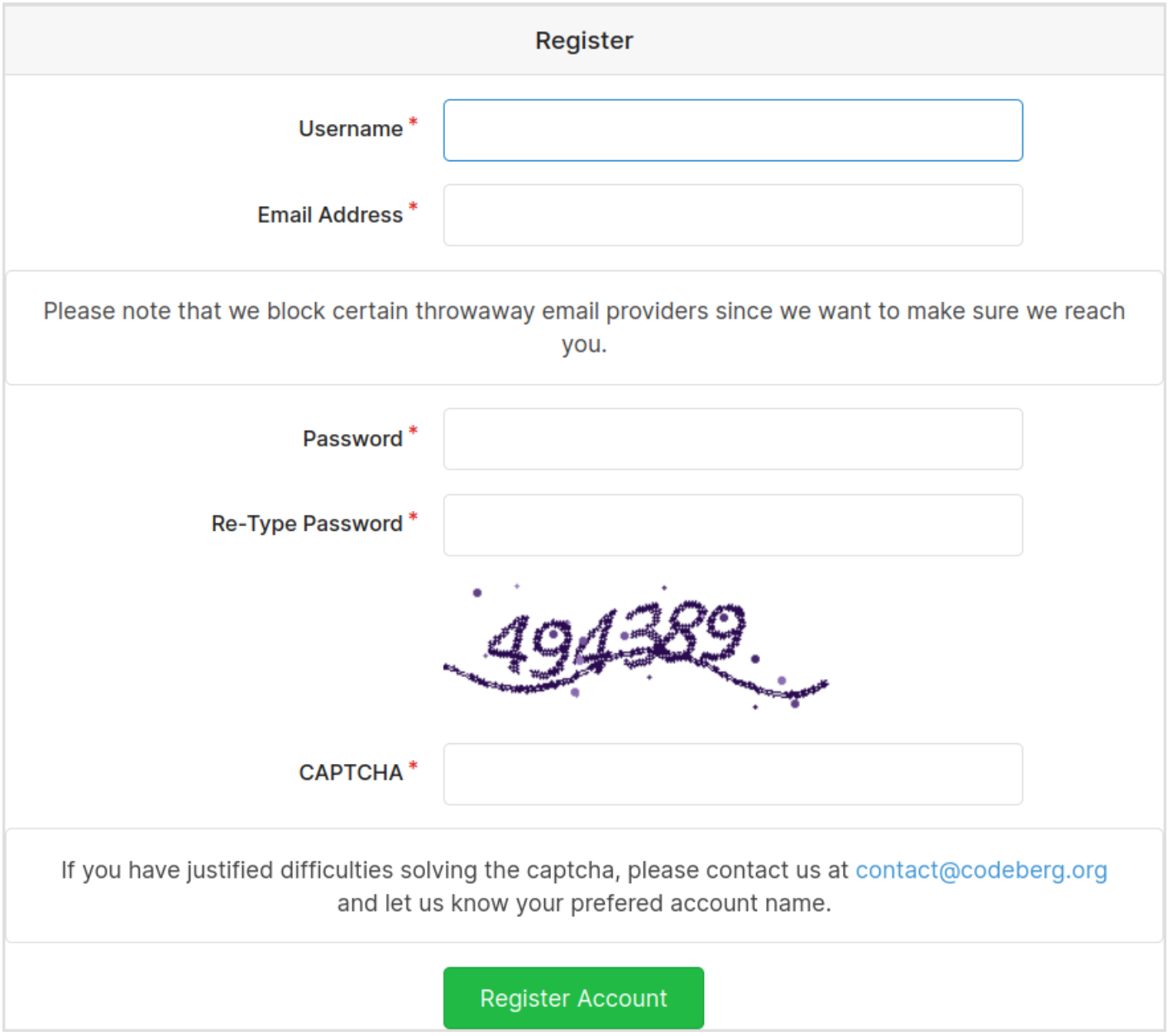
2. Register and sign in

In this chapter, we will register and sign in to Codeberg.

Codeberg hosts git repositories for free

2.1 Exercise 1

To register, go to https://codeberg.org/ and click on 'Register'.

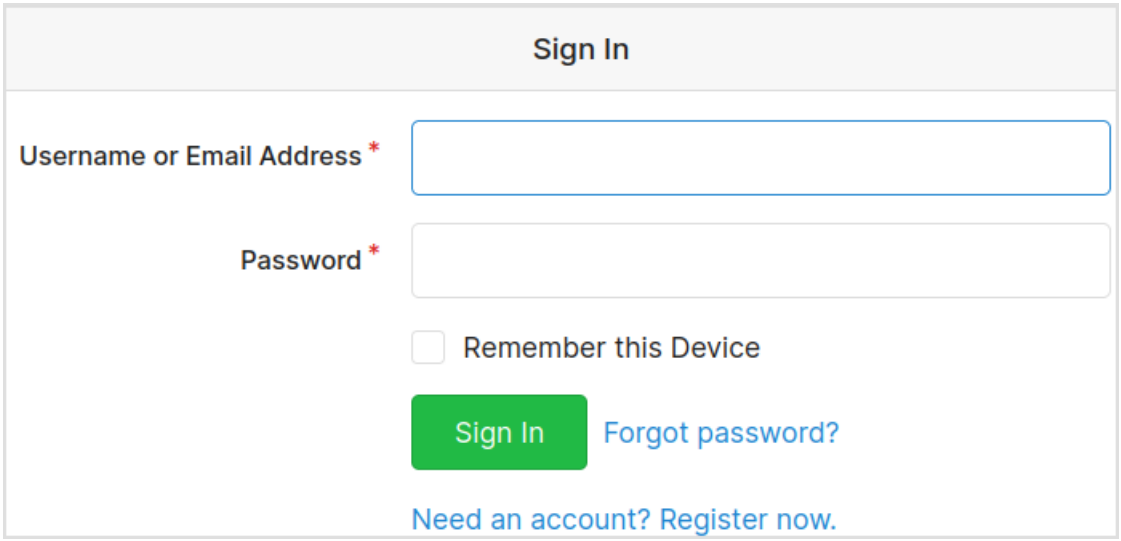


Do so.

Of course, Codeberg needs to know how cool I am

2.2 Exercise 2

Go to https://codeberg.org/ and click on 'Register' to sign in:

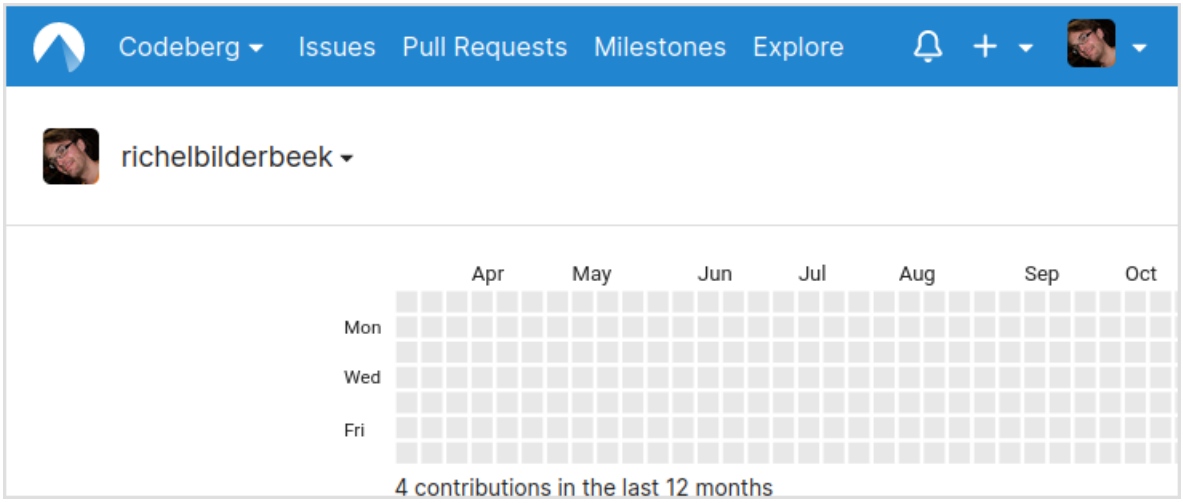


Do so, you will be taken to the Dashboard.

Of couse, Codeberg is happy to let me in

2.3 Final exercise

Register and sign in. This will take you to the Dashboard.



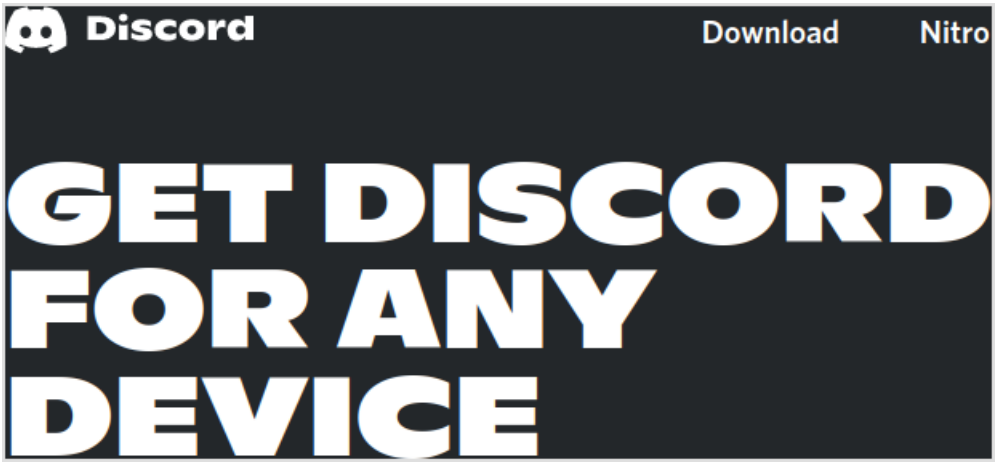
3. Install Discord

In this chapter, we install Discord

With Discord, we can talk to one another online

3.1 Exercise 1

Go to the Discord homepage at https://discord.com/download



The Discord homepage

Discord is used mostly by games and Twitch streamers

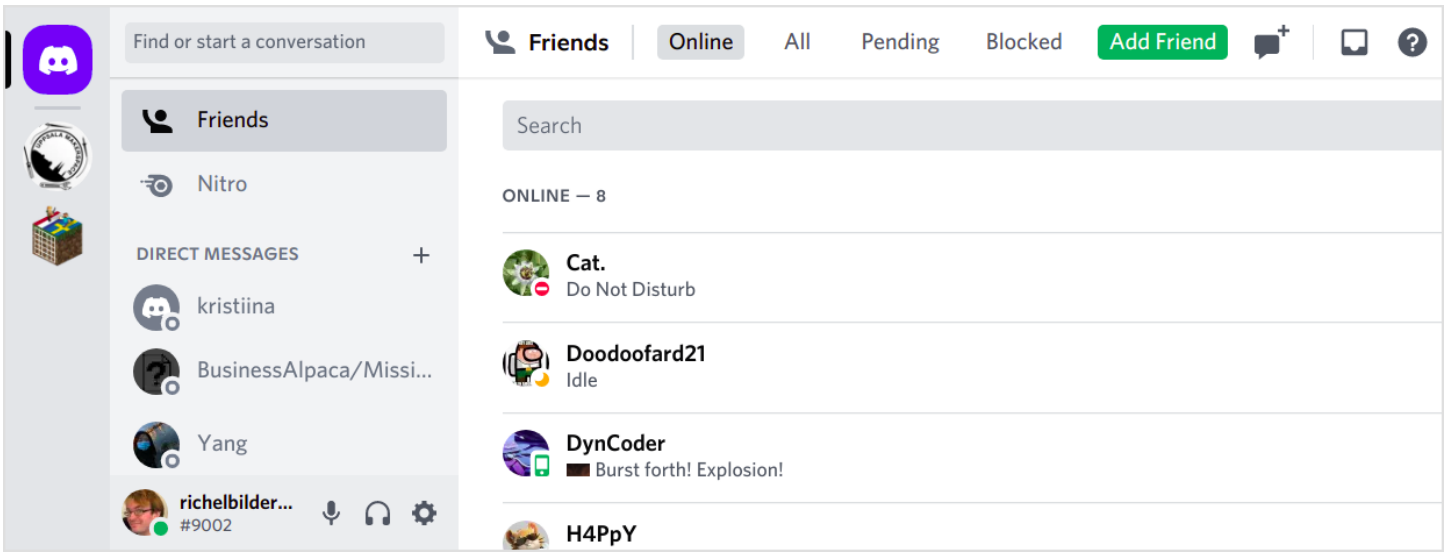
3.2 Exercise 2

At the Discord homepage, click in ‘Install’ to install Discord.

Installing Discord is easy

3.3 Exercise 3

Start Discord. You will see the Discord dashboard:



The Discord dashboard

After a while, you have many cool Discord servers here

Discord is big, take your time to look around

3.4 Exercise 4

Time to add a Discord friend.

Click on 'Add Friend' and fill in a username from someone you know, for example `richelbilderbeek#9002`.

Hopefully, you have picked a good Discord username yourself

3.5 Exercise 5

Time to go to our Discord server.

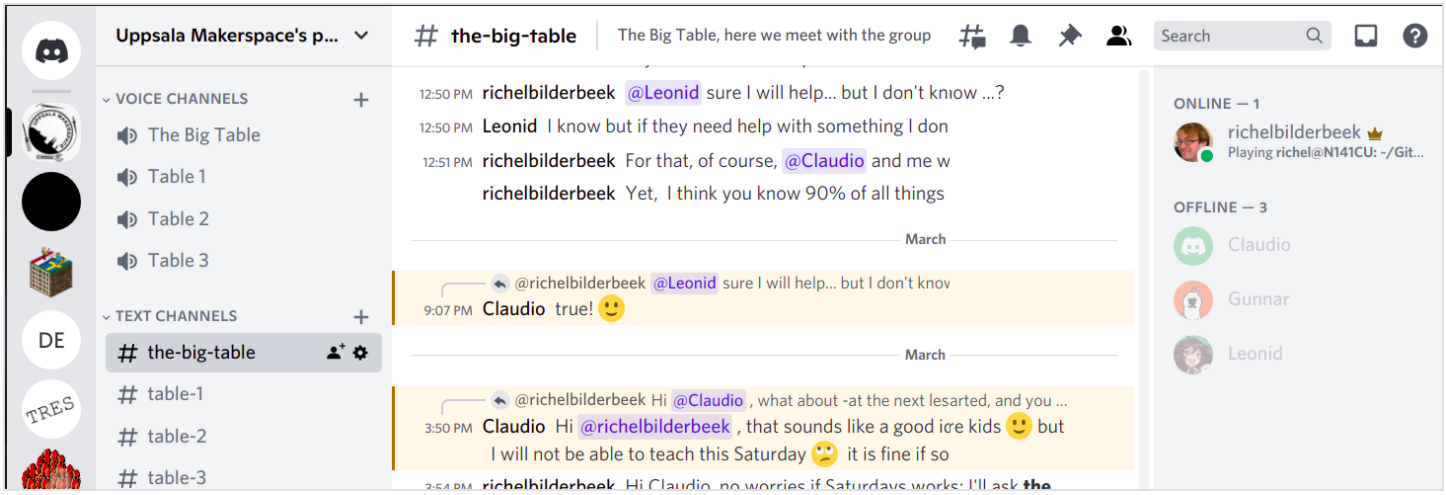
Ask your Discord friend to invite you to the Discord server.

You will get a PM with an Invite link. Click that link.

One can also use a direct Invite link to the server

3.6 Final exercise

On our Discord server, say something at 'The Big Table' channel.



Our Discord server

A Discord server can do many things, take your time to look around

---

Of course, I will say the coolest thing!

---

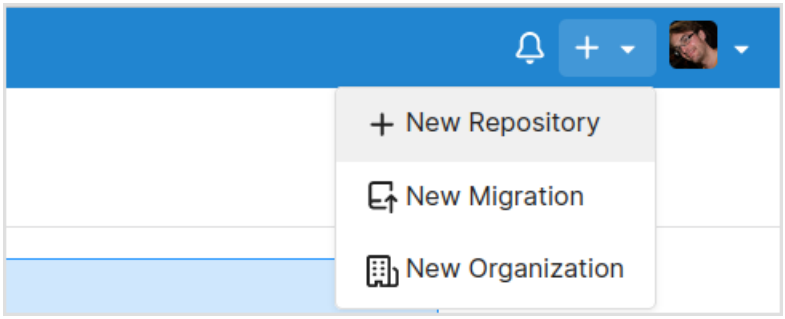
4. Create a repository

In this lesson, we are going to create our own repository

A repository is a website that works with `git`

4.1. Exercise 1

Login to Codeberg. Then, on the dashboard, click ‘New repository’:

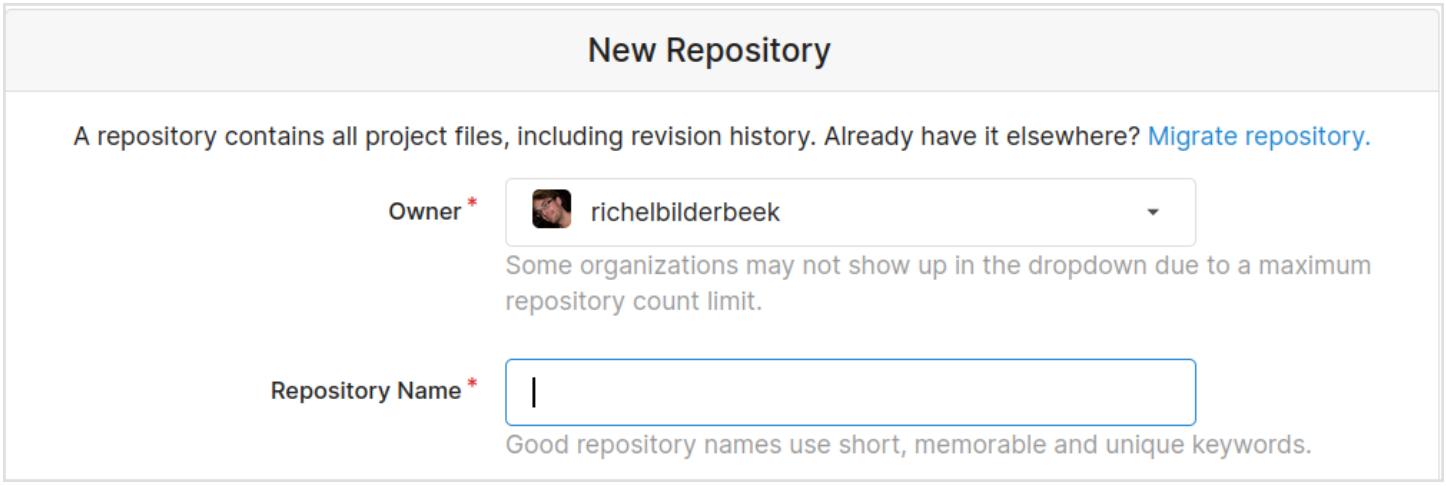


This will take you to the ‘New repository’ screen.

The dashboard shows all your repostories

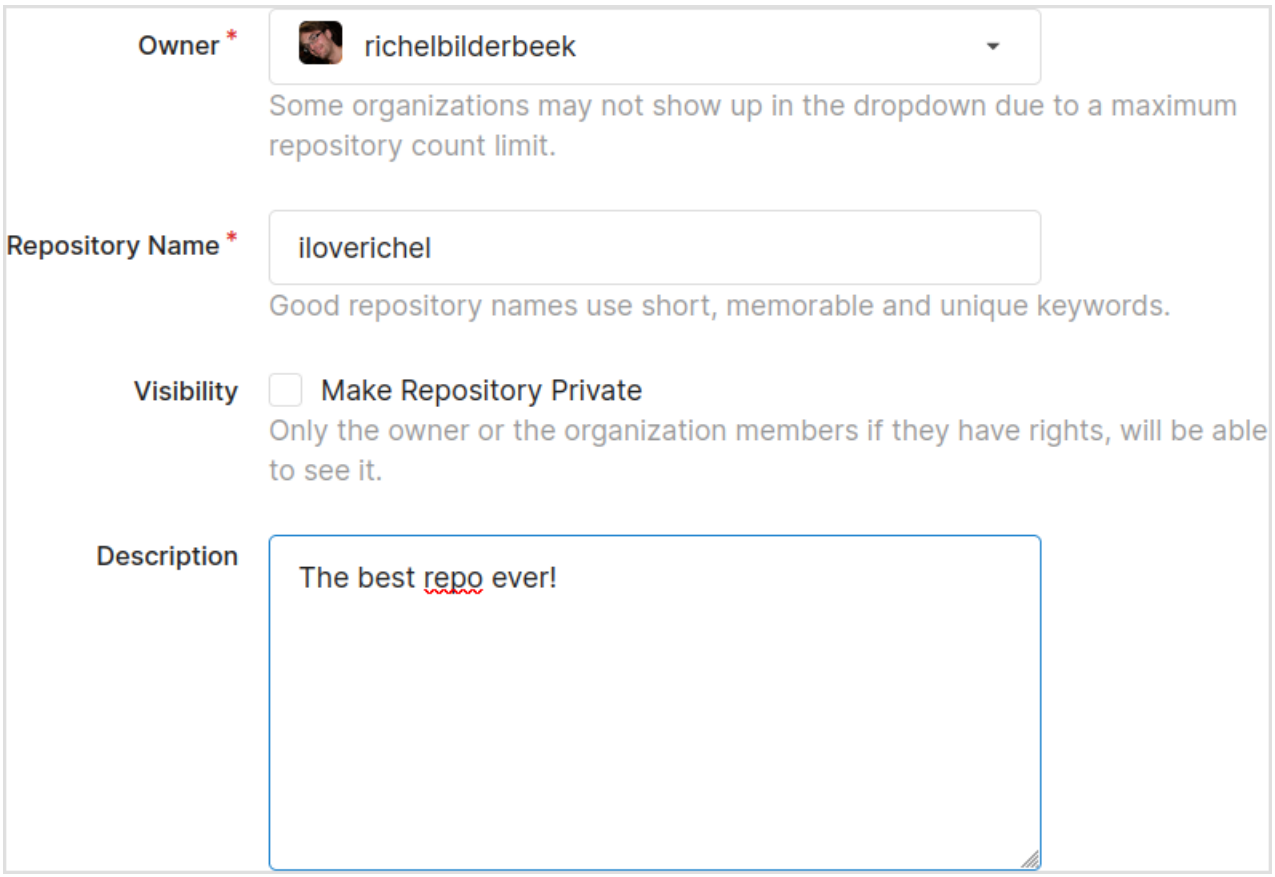
4.2. Exercise 2

This is the 'New repository' screen:

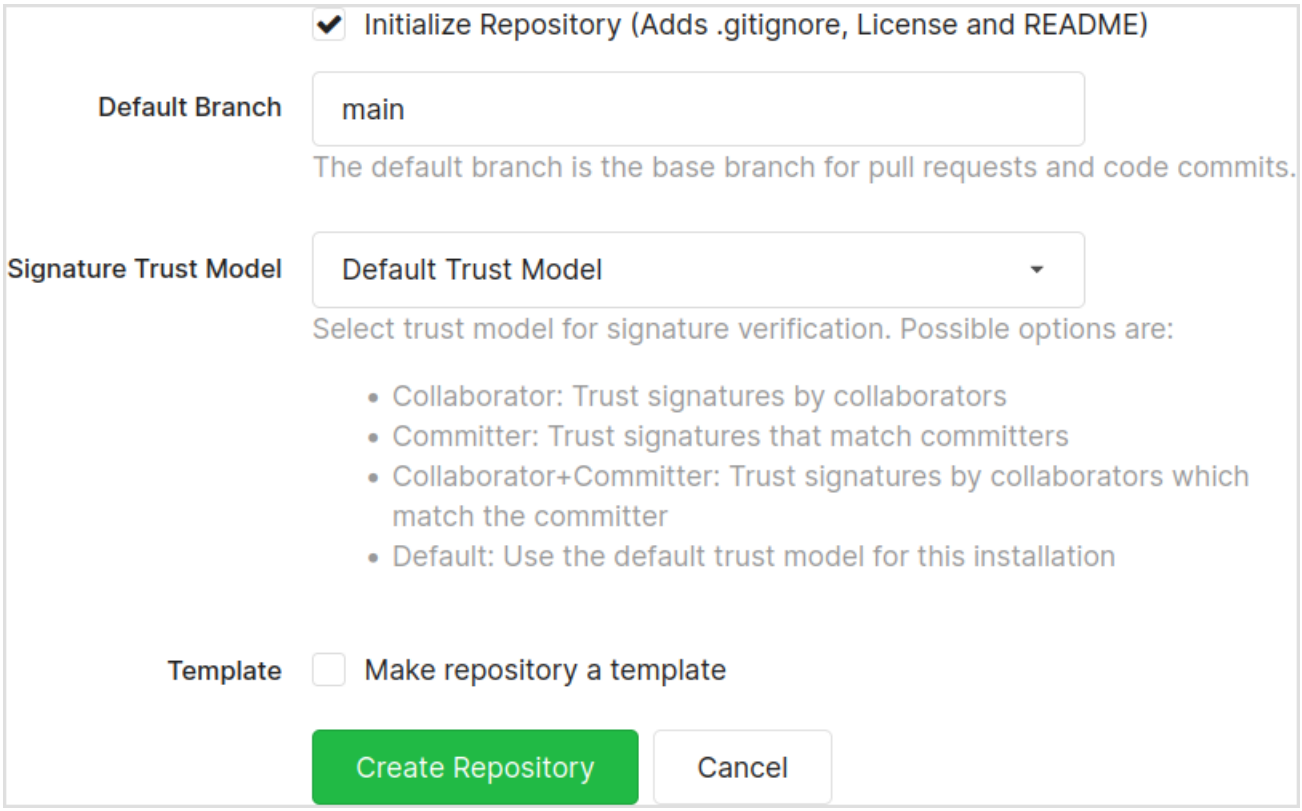


Fill in the 'Repository name', which will be the -duh!- name of the repository. In the case below, `iloverichel` is used.

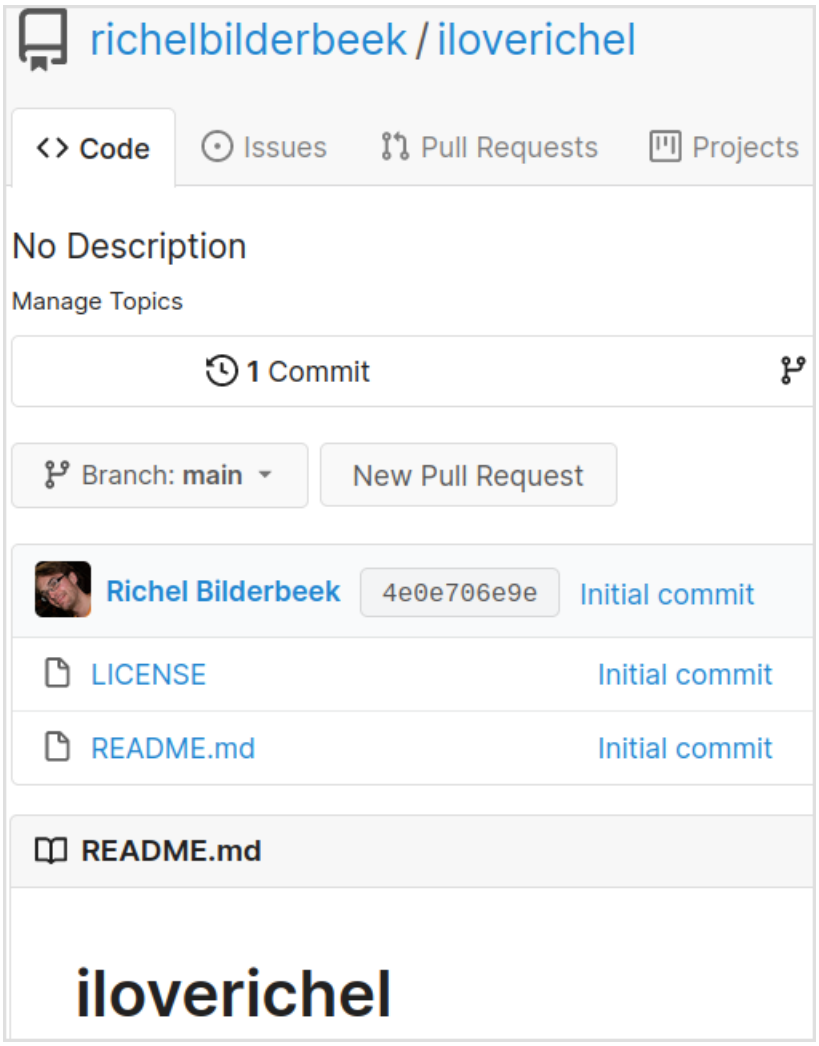
Maybe `richeliscrazy` is a better name



Click on ‘Initialize repository (Adds .gitignore, License and README)’. This step is optional, but we want this :-) .



Now you have your new repo!



The website text shown, is from `README.md`

4.3. Final exercise

Create a new repository without looking at this booklet.

This should be easy!

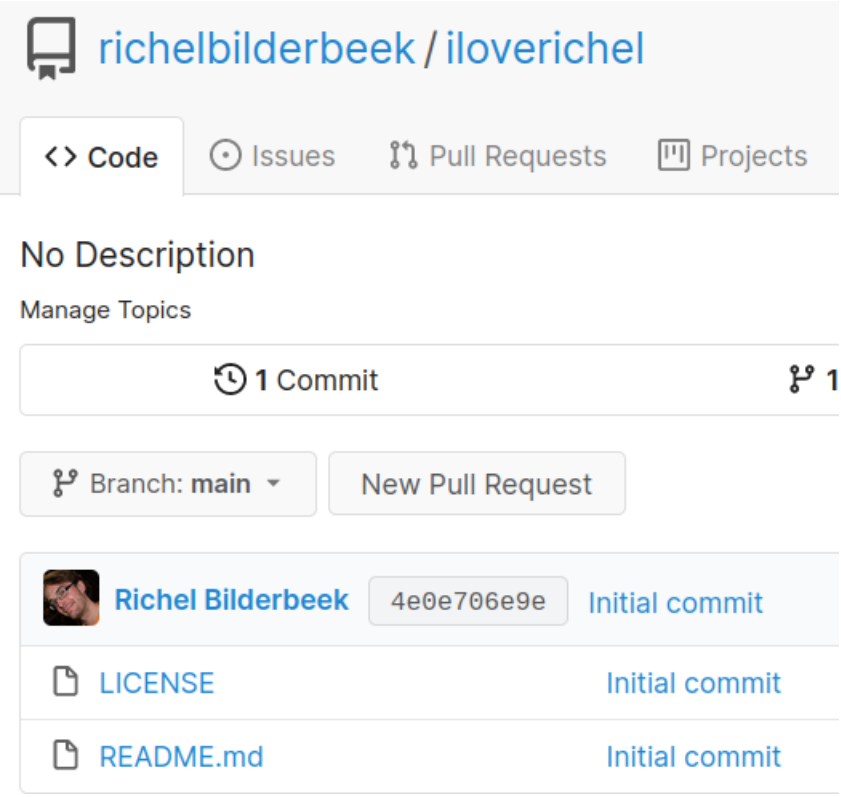
5. Monologue

A monologue is when you talk to someone without letting them talk back

Monologues are my favorite!

In this chapter, we are going to use the basic `git` workflow by writing a monologue.

5.1 Have a Codeberg repository



In this chapter, we will use an example user, which is `richelbilderbeek` and and example repository name, which is `ilovericheI`.

A repository is a website that works with `git`

5.2 Exercise 1

Create a Codeberg page for your repository.

'repository' is often shortened to 'repo'

5.3 Cloning a repository

To clone your repository, in Git Bash, or a Linux/Mac terminal, do:

```
git clone https://codeberg.org/richelbilderbeek/ilovericheL.git
```

```
richel@N141CU:~$ git clone https://codeberg.org/richelbilderbeek/ilovericheL.git
```

This will download your online repository to a folder on your harddrive:

```
richel@N141CU:~$ git clone https://codeberg.org/richelbilderbeek/ilovericheL.git
Cloning into 'ilovericheL'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (4/4), 11.60 KiB | 1.93 MiB/s, done.
richel@N141CU:~$
```

5.4 Exercise 2

Clone your repository.

Star Wars 'The Clone Wars' was not about `git`

5.5 Navigate into the folder

After cloning a repo, we need to navigate into the folder where the repository is. Use `cd` and the repository name (in this example `iloverichel`):

`cd iloverichel`

```
richel@N141CU:~$ cd iloverichel/
richel@N141CU:~/iloverichel$
```

`cd` means 'Change directory'

To make sure you are in that folder, use `ls`:

`ls`

```
richel@N141CU:~$ git clone https://codeberg.org/richelbilderbeek/iloverichel.git
Cloning into 'iloverichel'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (4/4), 11.60 KiB | 1.93 MiB/s, done.
richel@N141CU:~$ cd iloverichel/
richel@N141CU:~/iloverichel$ ls
LICENSE  README.md
richel@N141CU:~/iloverichel$
```

`ls` means 'List'

One file you will see is called `README.md`.

5.6 Exercise 3

Navigate into the folder where your repo is cloned.

Use `cd` to change the directory/folder

5.7 Use `git status`

`git status` is very helpful. You can always use `git status` in a repo folder:

```
git status
```

`git status` gives many hints how to use `git`.

5.8 Exercise 4

Run `git status` in a repo folder. If you get an error message, you did not navigate into a repo folder.

My `git status` is `Cool`

5.9 Edit `README.md`

`README.md` contains the front page of your GitHub repo.

`README.md` is in screamcase, due to history

`.md` is short for 'Markdown', which is a markup language

HTML is a famous markup language

To edit `README.md`, you can open it:

- using a file browser (e.g. Microsoft File Manager, Apple Files, or Nautulus)
- using Git Bash or a terminal

In Git Bash or the terminal, do:

```
[name of your editor] README.md
```

where `[name of your editor]` is the name of your editor, for example `notepad`, `mousepad`, `edit`, `gedit`, `vim`, `emacs`.

There are many text editors!

This is, for example:

```
notepad README.md
```

Now your editor will open up and you can modify the README.md file.

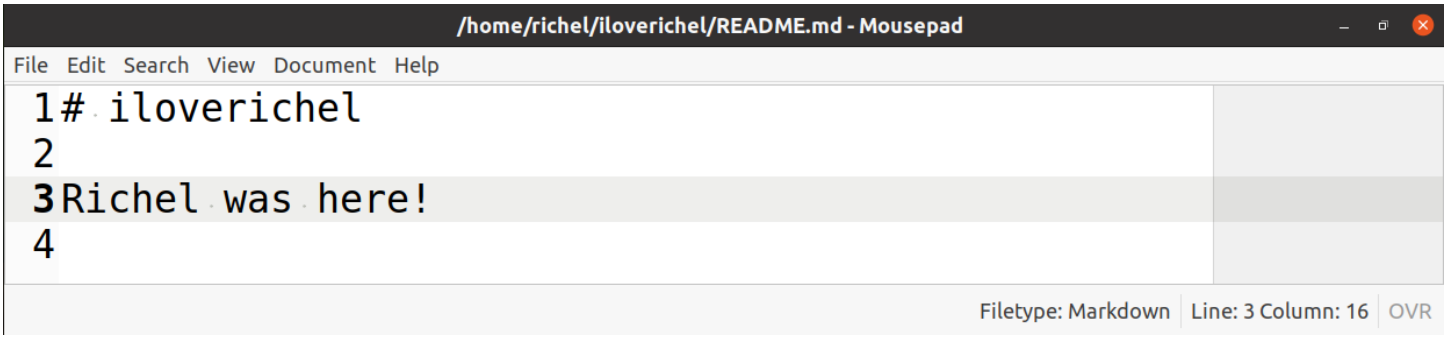
The text will be similar to:

```
# iloverichel
```

Change the text in any way, as if having a monologue:

```
# iloverichel

Richel was here!
```



Save the file and close the editor.

You will be taken back to Git Bash or terminal:

```
richel@N141CU:~/iloverichel$ mousepad README.md
richel@N141CU:~/iloverichel$
```

5.10 Exercise 5

Edit your `README.md`.

5.11 `add`, `commit`, `push`

Now come the most commonly used three `git` commands.

`git` can do a lot of things

Tell `git` you want to 'stage' (whatever that is) all files you've changed:

```
git add .
```

Don't forget the dot (`.`) at the end!

Next, tell `git` what your changes were:

```
git commit -m "Improved monologue"
```

Between the "s you describe what you did.

Good programmers have commit messages that tells clearly what they did

Finally, upload your work to Codeberg:

```
git push
```

If you did it correctly, your Codeberg repository will have your new text.

If you are asked for:

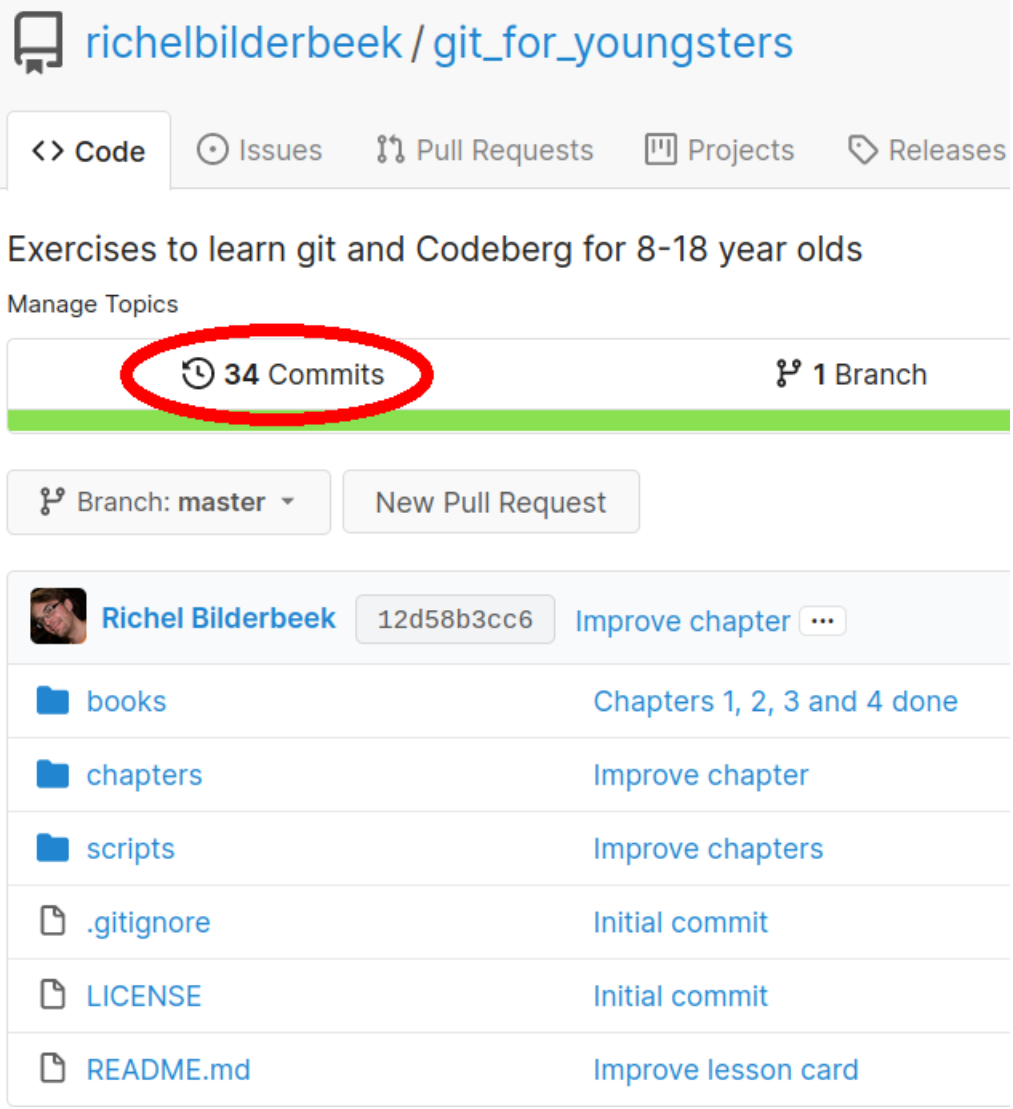
- a password, type it in
- another password, type it in again
- an access token, see chapter 'Register and login'

`git` needs to be sure who you are

After this, do another round of `add`, `commit`, `push` :-)

Congratulations, you just did the full workflow!

5.11 Exercise 6



Add 5 commits by, for 5 times, adding a sentence and then `add`, `commit`, `push`.

Tip: with the arrow key upwards, you can get back previous commands!

It is handy that Git Bash knows my previous commands!

---

6. Fairytale on `master`

In this chapter, we are going to use the basic `git` workflow by writing a fairytale with a team.

A fairytale is a story in which anything can happen

6.1 Exercise 1

On Codeberg, create a new repository.

Easy peasy!

6.2 Adding Collaborators

A Collaborator is someone in your team

To add Collaborators, in your repo, click on 'Settings' (at the top right), then click on the 'Collaborators' tab.

Add one or more Collaborators. Collaborators are people you work together with. They too can now modify the text.

6.3 Exercise 2

Add at least 1 Collaborator.

Add me, I am the best!

6.4 Basic workflow

You already know this

Clone the repository on your computer:

`git clone [repo_url]`

For example,

`git clone https://codeberg.org/richelbilderbeek/ilovericheL`

Navigate into the folder of that repository:

`cd [repo_name]`

For example,

`cd ilovericheL`

Modify the `README.md`, for example

`notepad README.md`

Fairytales usually start with 'Once upon a time'

Fairytales usually end with 'And they lived happily ever after'

Do `add`, `commit`, `push`:

```
git add .
git commit -m "Improve"
git push
```

6.5 Exercise 3

Each team member must

- add a sentence to the fairytale
- `add`, `commit` and `push` it

If something goes wrong, go to the next paragraph.

The story is unimportant here

6.6 `git pull`

If at `git push` you get a long text similar to below, you need to update your local repository, see below that :-)

```
richel@N141CU:~/codeberg/git_for_youngsters$ git push
To https://codeberg.org/richelbilderbeek/git_for_youngsters.git
 ! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'https://codeberg.org/richelbilderbeek/git_
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

If you read the text, you can read you should do a `git pull`

The reason for this text is, when you work together, it can be that the version on your computer is outdated: a Collaborator can have `push`ed a new version as well. To update, do:

```
git pull
```

Tip: you can never do `git pull` too often

Tip: when in doubt, do `git pull`

6.7 Exercise 4

At the same time, each team member must

- add a sentence to the fairytale
- `add`, `commit` and `push` it
- get the message that suggests to do `git pull`
- do a `git pull`

If you get an error message like below, move on to the next paragraph.

```
CONFLICT (content): Merge conflict in README.md
```

If your team works smart, there will be few merge conflicts

6.8 Merge conflict

When two people work on different files, or on different parts of the same file, `git` can easily combine the changes. However, when `git` is not sure what to do, it will give you a merge conflict.

For example, assume `README.md` is:

```
To be
or not to be
```

One person changes this to:

```
To be
a dinosaur
or not to be
```

The second person, however, changes his/her version of `README.md` to:

```
To be a crocodile
or not to be
```

When the first person pushes his/her work, there is no problem. A problem will arise when the second person tries to push (which fails), then pulls: a merge conflict. This looks similar to this:

```
richel@N141CU:~/codeberg/git_for_youngsters$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 294 bytes | 294.00 KiB/s, done.
From https://codeberg.org/richelbilderbeek/git_for_youngsters
   7879d89..5e32784  master     -> origin/master
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.
```

The second person, with the merge conflict, needs to solve this. This can be done by editing `README.md`. It will look similar to:

```
<<<<<<<<<<<<
To be
a dinosaur
============
To be a crocodile
>>>>>>>>>>>>
or not to be
```

The `<<`s and `==`s and `>>`s are nothing special: they are just text and can be deleted like any other text.

The second person now needs to **think** and change the text to whatever is best, for example, to:

```
To be a crocodile or dinosaur
or not to be
```

This is a change, hence a `add`, `commit` and `push` needs to be done.

6.9 Exercise 5

Work together on a fairytale, with 5 commits per person. Also, each person must have solved one merge conflict. If you get no merge conflicts, **cause** them.

We work on a fairytale, as that 'always works' (unlike code)

---

7. ‘Hello World’ on `master`

In this chapter, we are going to use the basic `git` workflow by writing a simple program with a team.

A fairytale is a story in which anything can happen

7.1 Exercise 1

On Codeberg, create a new repository.

Easy peasy!

7.2 Adding Collaborators

A Collaborator is someone in your team

To add Collaborators, in your repo, click on 'Settings' (at the top right), then click on the 'Collaborators' tab.

Add one or more Collaborators. Collaborators are people you work together with. They too can now modify the code.

7.3 Exercise 2

Add at least 1 Collaborator.

Add me, I am the best!

7.4 Clone the repo

You already know this

Clone the repository on your computer:

`git clone [repo_url]`

For example,

`git clone https://codeberg.org/richelbilderbeek/iloverichel`

7.5 Exercise 3

Clone your repo

7.6 Processing

Start Processing.

Below is a short Processing program:

```
void draw()
{

}
```

You can run it by clicking on 'Run'.

7.7 Exercise 4

- Save the Processing program above in the folder of your repository
- Upload it

7.8 Modifying code

Navigate into the folder of your repository:

`cd [repo_name]`

For example,

`cd iloverichel`

Here, as usual, you can do `add`, `commit`, `push` and `pull`:

```
git add .
git commit -m "Improve"
git push
git pull
```

With Processing, you can edit the code.

7.9 Exercise 3

Make something pretty together!

Each team member must

- add one line of code at a time
- `add`, `commit` and `push` it
- at least 5 commits per team member

Below is some example code:

```
void setup()
{
  size(256,256);
}

void draw()
{
  fill(mouseX, mouseY, mouseX + mouseY);
  ellipse(mouseX, mouseY, 50, 50);
  fill(mouseY, mouseX, 255);
  ellipse(mouseY, mouseX, 50, 50);
}
```