6) Git and GitHub

Agenda

- What is Git
- How to install Git
- Basic Git usage
- Intro to GitHub

When you work on some project and you **code some new improvements**, you might later discover these improvements were **not that good**, so you'd like to **go back to the old code.** Unfortunately, the "undo" button doesn't work anymore and your good old code is lost forever...

Wouldn't it be great if you could have a **historic archive** of your code?

Well... you can! Not just *can*, every serious programmer **should** have it. And it's called Git.

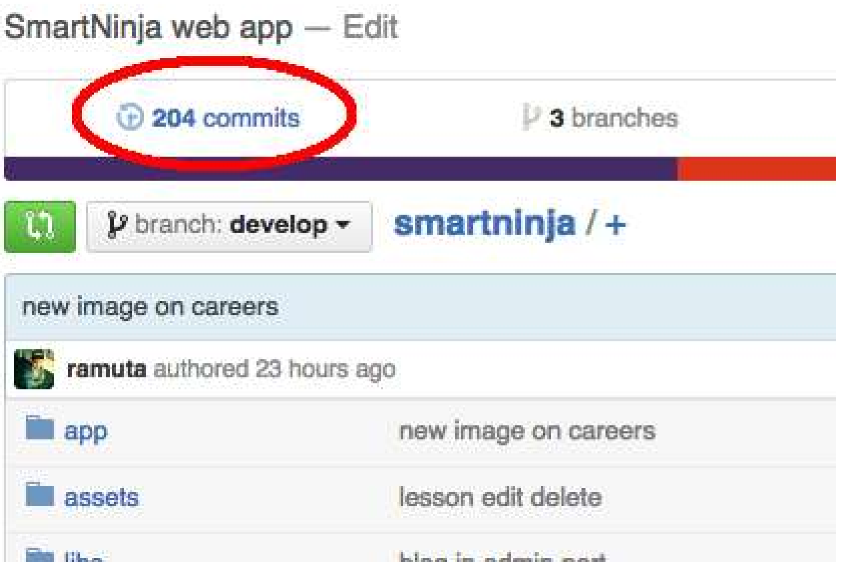
How does Git look like?

There's a website called GitHub where you can upload and save your code. Let's take a look for example a code for an ex SmartNinja forum: https://github.com/smartninja/forum (https://github.com/smartninja/forum). If you click for example on `main.py` you're able to see some of the forum code.

> *This is an old SmartNinja forum code that we don't use anymore.*

This is the latest version of the code. So where can we find the previous versions? After all, we said that Git provides you with a historic code archive (in Git parlance it's called a *repository*).

Go back to https://github.com/smartninja/forum (https://github.com/smartninja/forum) and click on "commits":



Now you can see a list of all the changes ever made in the project. If you click on one, you can see what exactly was changed - what was added (marked green) and what was deleted (marked red).

Why should you use Git

Besides being a historic code archive, there are also **other benefits** with using Git:

- **multiple programmers** at once can work on a single project,
- GitHub serves as a **code storage** for you - in case your computer disk breaks and you loose all the data on it,
- the projects you have on GitHub serve as sort of a **programming CV** for you. Having a GitHub account with many projects on it can **increase** your chances of getting a **coding job**.

Let's learn how to use Git and GitHub.

Prerequisites

Before you can start using Git and GitHub, make sure that you have the following two things set up:

1. GitHub account

If you haven't already, create yourself an account on GitHub.com (https://github.com/).

2. Git installed on your computer

Git is a program on your computer, that will help you create a git repository for each of your projects and then upload these repositories to GitHub.

If you've already installed Git, skip this step.

If not, go to http://git-scm.com/downloads (http://git-scm.com/downloads) and download Git (when installing just click "next" and use the default settings).

Two ways of using Git

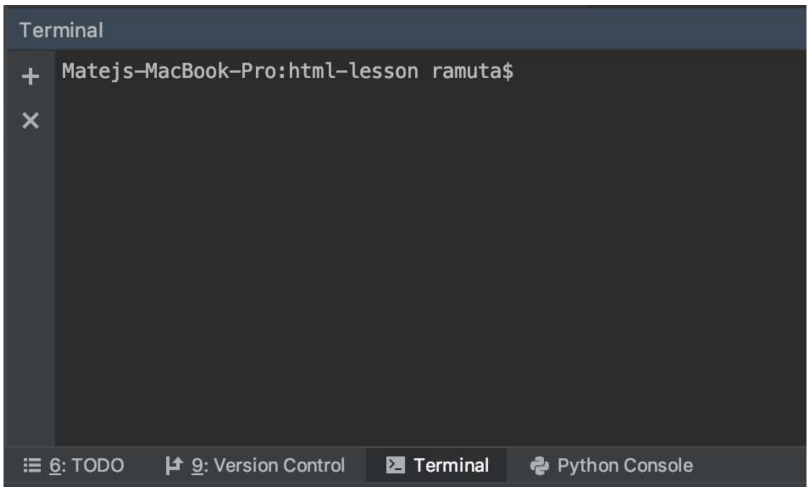
You can use Git via a command line (Terminal) or visually through a graphical user interface.

Let's first learn to use Git via a Terminal (if something doesn't work, skip to the GUI part).

Git via the Terminal

1. Open the Fakebook project in PyCharm

Open the Fakebook project (or any other) in PyCharm. In the bottom of PyCharm you'll see the Terminal tab. Click on it and the Terminal will open.

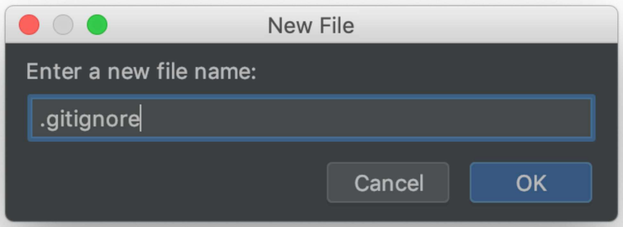


```
Matejs-MacBook-Pro:html-lesson ramuta$
```

You'll use the Terminal to enter git commands in it.

2. Create a .gitignore file

Create a new file in the Fakebook project. Name the file `.gitignore`:



Make sure the file name starts with a **dot** `.`.

> *Fun fact: Files that start with a dot are considered as hidden files.*

Enter the following line in the `.gitignore` file:

```
.idea
```

This means that Git will ignore the `.idea` folder and it will not save it into the Git archive.

3. Set your name and email

Enter your name and email, so that Git can know who was the person that was changing the code. Enter these two commands (one after another) into the Terminal (use your real name and email):

```
git config --global user.name "John Doe"

git config --global user.email johndoe@example.com
```

Do this only once. Git will remember these settings for all of your projects.

4. Initialize a Git repository

Run the following command:

```
git init
```

This will initialize an empty Git repository (code archive) for your Fakebook project.

5. Add and commit code to your Git repository

Currently the Git archive is empty, so we need to add some code to it. Type the following command:

```
git commit -am "your message"
```

The `"your message"` should be a short message describing your changes. If you've added an image to your project, write the message like this:

```
git commit -am "image added"
```

For the first commit people usually write:

```
git commit -am "initial commit"
```

Now we have our code saved in our Git repository on our computer. But we haven't finished yet. Next we want to push our code to GitHub, so we can see it online.

6. Create a GitHub repository

1. Go to GitHub.com (make sure you're logged in).
2. In the right side of the navigation bar (near your username), click on + (plus) and select "New repository". This is basically a code repository for some coding project of yours.
3. As a project name put `fakebook` (don't use any space in the name).
4. You can add some description if you want, but leave the rest as it is.
5. Click on "Create repository".

7. Add a link to your GitHub repository

Now you have to let your fakebook project on your computer (in PyCharm) know the URL of your GitHub repository, so you can upload your code there.

Go back to PyCharm and type this command in the terminal:

```
git remote add origin https://github.com/your-username/fakebook.git
```

Make sure to add the right URL in this command (the above example is not correct!). Instead of `your-username` it should be your GitHub username. Make sure the URL has `.git` at the end.

You can check if you've entered the right URL using this command:

```
git remote -v
```

8. Push (upload) to GitHub

Now you can push your local Git repository to GitHub:

```
git push origin master
```

The Terminal will ask you to enter your GitHub username and password. Be aware that the password will not be visible in the Terminal, but the Terminal will recognize what you type in.

If everything went well, you should be able to see your Fakebook code on GitHub. Check it out!

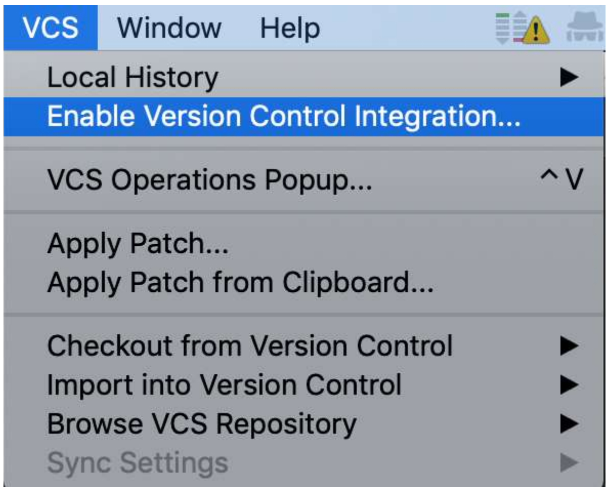
Git via GUI

Instead of using a command line (via the Terminal), you can use PyCharm's visual tools and push your code to GitHub by clicking :)

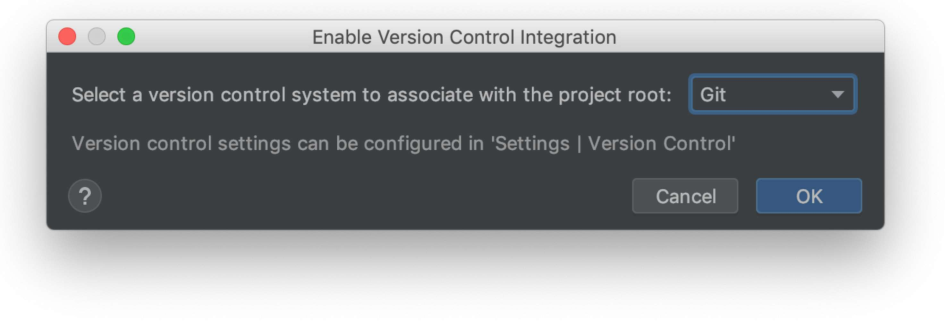
Let's learn how to do it on a new project. You can use one of your previous homeworks (like Boogle) or create a new PyCharm project and some file in it (for example `index.html`).

1. Enable version control integration

First go to the menu bar, find VCS and click on "Enable Version Control Integration".

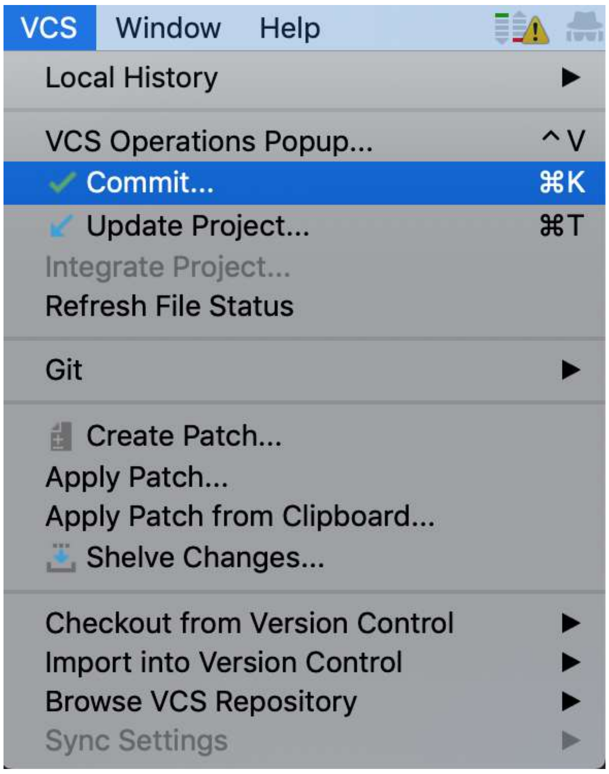

VCS
Window
Help
Local History
Enable Version Control Integration...
VCS Operations Popup... ^V
Apply Patch...
Apply Patch from Clipboard...
Checkout from Version Control
Import into Version Control
Browse VCS Repository
Sync Settings

A new window will appear. Select Git in it and click OK:

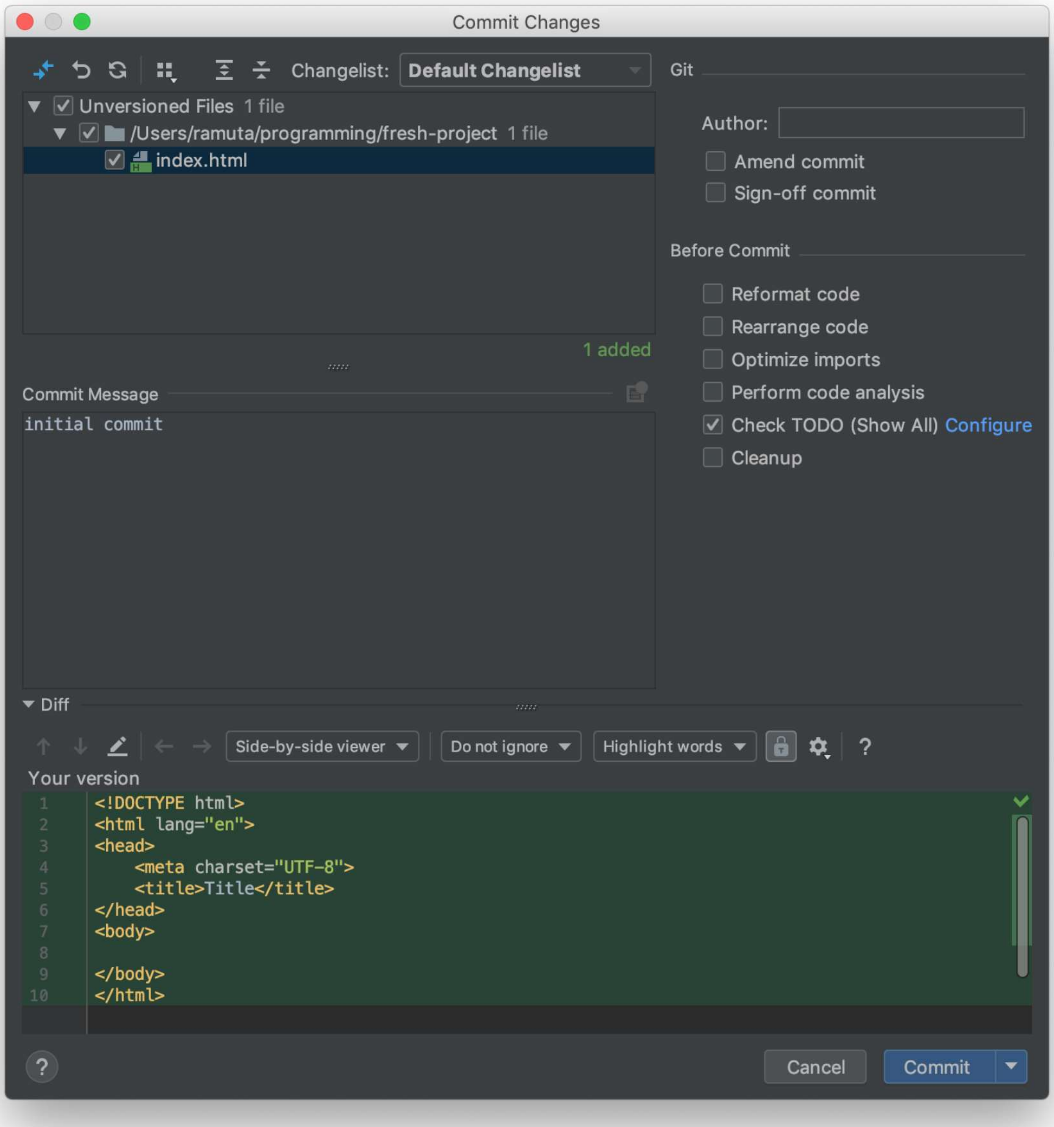

Enable Version Control Integration
Select a version control system to associate with the project root:
Git
Version control settings can be configured in 'Settings | Version Control'
?
Cancel
OK

2. Commit your changes

Click on the VCS tab in the menu bar again. You can see that new options appeared. Click on **Commit**:



A new window will appear:



Commit Message: initial commit

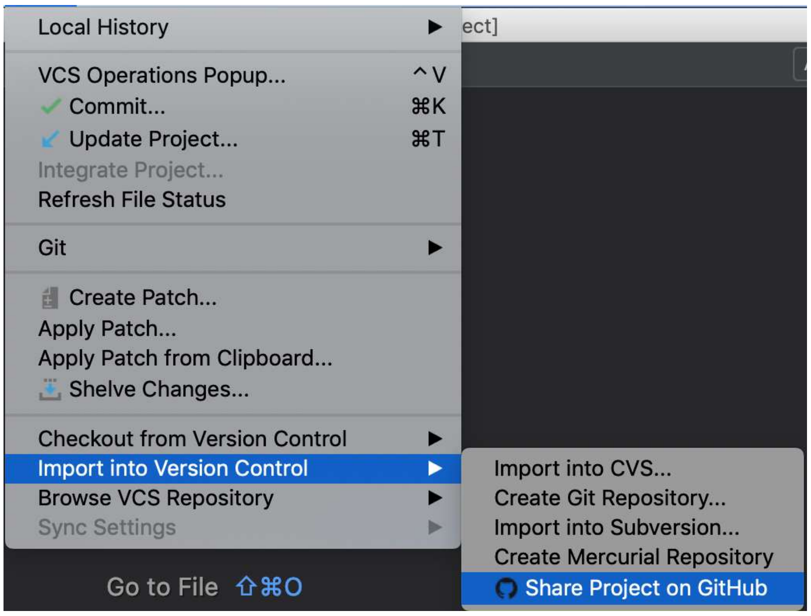
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>

</body>
</html>
```

Check the files that you want to commit and enter the **commit message.** Then click on **Commit button.**

3. Push to GitHub

The last step is to push your code to GitHub.

Go to the menu bar again, click on CVS and select Import into version control --> Share project on GitHub.



Enter your username and password and follow the next steps that the process will demand from you.

If everything went okay, you can see the changes on your GitHub account.

Q&A

Any question?

If there's enough time after Q&A, students can start working on their homework.

Homework 6.1

Add a Git repository for every project that you've created so far in this course and upload it to GitHub.

Bonus: Other GUI tools

> *Do this at home*

If you'd like, you can check out these two Git tools that have a graphical user interface.

SourceTree

Instead of using Git through a command line (terminal/bash/shell), you can use it via graphic user interface. For this, you need SourceTree. Read a short tutorial here (https://www.smartninja.at/blog/how-to-use-git-with-sourcetree-1463228948177).

GitKraken

Another Git GUI tool is GitKraken. You can download and try it out here: https://www.gitkraken.com/ (https://www.gitkraken.com/).

You can also take a look at this GitKraken tutorial: https://www.youtube.com/watch?v=f0y_xCeM1Rk (https://www.youtube.com/watch?v=f0y_xCeM1Rk).

Bonus: GIT and command line tutorials

- Atlassian tutorial: https://www.atlassian.com/git/tutorials/ (https://www.atlassian.com/git/tutorials/)
- Learn how to navigate in the Terminal: https://www.codecademy.com/en/courses/learn-the-command-line (https://www.codecademy.com/en/courses/learn-the-command-line)
- Git YouTube tutorials: https://www.youtube.com/playlist?list=PL6gx4Cwl9DGAKWClAD_iKpNC0bGHxGhcx (https://www.youtube.com/playlist?list=PL6gx4Cwl9DGAKWClAD_iKpNC0bGHxGhcx)
- Git cheatsheet: https://dev.to/digitalocean/how-to-use-git-a-reference-guide-6b6 (https://dev.to/digitalocean/how-to-use-git-a-reference-guide-6b6)