Mini-workshop LaTeX and Git

Daniel Pereira
Armando Nolasco Pinto

Optical Networks
DETI, University of Aveiro
February 13, 2019

Introduction to LaTeX

Intro

Initial notions

- What is it?
  - A document preparation system for high-quality typesetting.
- It has some advantages when compared to Office platforms:
  - Free.
  - Easy reference and citation management.
  - Potent mathematical writing.
  - Very commonly used in science and engineering.
  - It's as cross-platform as you can get.

Initial notions

- What is needed for it to work?
  - A TEX distribution (MiKTeX, MacTex, etc).
  - Some text editor: WinEdt, Texmaker, TeXworks, TeXShop, Overleaf and Sharelatex (online editor).
- What is handy to have?
  - Citation manager (JabRef, Mendeley, Bibdesk or other).
  - A decent PDF reader (Sumatra PDF, Foxit Reader, Adobe Acrobat or other).
- Whenever you have any doubts, Google `en.wikibooks.org/wiki/LaTeX`.

What is a command

- A command has the following structure:

```
\commandname[option1, option2]{argument1}{argument2}
```

- Examples:

```
\documentclass[11pt]{report}
\usepackage[utf8x]{inputenc}
```

- The \usepackage command includes packages in the document, these packages give meaning to a few commands. Example:

`\usepackage{amsmath}` allows for equation writing.

What is an environment

```
\begin{environment}
 ...ambient content...
\end{environment}
```

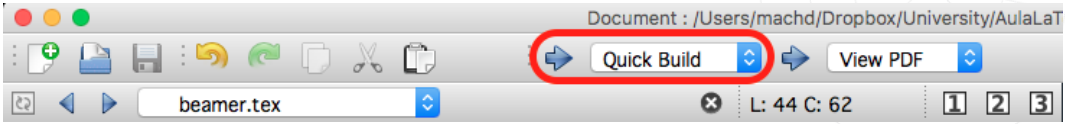
- There is plenty of code that only functions inside a specific environment. Example:

```
\begin{document}
 ...document content...
\end{document}
```

How to start a document

Offline LaTeX compilation

- The code to be compiled should be in a .tex file.
- Compilation can be done with a .tex editor or in the command line.



- When using an offline compiler, save the .tex file and run the compiler inside a folder, LaTeX generates a bunch of support files.

Offline LaTeXcompilation

- This is what a LaTeXcode looks like.

```
1  \documentclass{report}
2  \usepackage[utf8]{inputenc}
3
4  \begin{document}
5  Isto é um documento com uma linha de texto.
6  \end{document}
```

Isto é um documento com uma linha de texto.

- Compilation usually returns a .pdf file.

Simple text editing

bold, *italics*, underline, colourful

- \textbf{Bold text}
- \textit{Italicised text}
- \underline{Underlined text}
- \textcolor{Colourname}{Colourful text}
  - Colour names can be found here→en.wikibooks.org/wiki/LaTeX/Colors.

Paragraphs, line breaks and sections

- \\ Breaks the line, doesn't start a new paragraph.
- \par Breaks the line and starts a new paragraph.
- \chapter{Chapter name} Starts a chapter.
- \section{Section name} Starts a section.
  \subsection{Subsection name},
  \subsubsection{Subsubsection name}
- All of these are numbered, writting a * before the {} suppresses this.

Titles, authors and tables of contents

- All of these can be generated automaticaly by LaTeX, their appearance depends on the template.
- In this case, you need to give LaTeX the necessary information, in the preamble write:
  - \title{Title}
  - \author{Author or Authors}
  - \date{Date}
- For the title to appear you need to use the\maketitle command, usually right after: \begin{document}
- You may be given a template where the title is defined explicitly, in that case just alter the corresponding text.
- To generate the index just write the command \tableofcontents, usually right after the \maketitle.

Changing the language of the document

- Some compilers have a spell checker, set it to the language you are using.

- The language of the document can be changed with the babel package.

```
\documentclass{report}
\usepackage[utf8]{inputenc}
\usepackage[portuguese]{babel}

\title{Isto é um título}
\author{Eu escrevi isto}
\date{\today}

\begin{document}
\maketitle
\tableofcontents
\chapter{Isto é um capítulo}
\section{Isto começa uma secção}
\subsection*{Esta subsecção não é numerada}
Isto é um documento com uma linha de texto.
\chapter{Isto é outro capítulo}
\end{document}
```

Making tables

Tables - the hard way

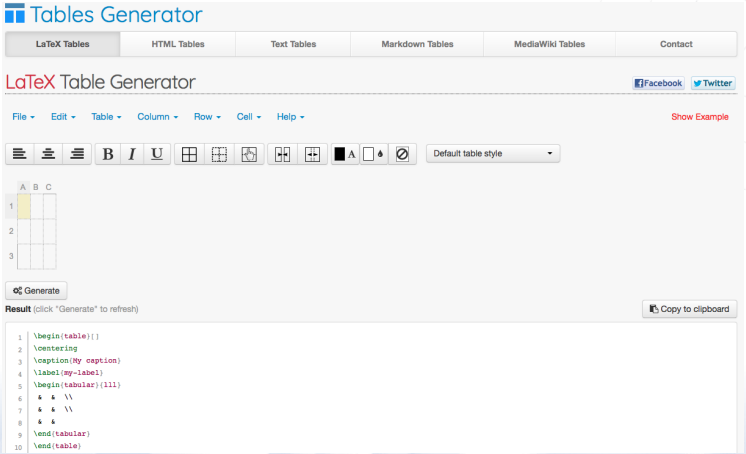
- You need to use the {table} environment.
- You need to use the {tabular} environment.
- You need to set the column alignment and if you want to have vertical lines between them.
- You have to set the horizontal lines you want.

```
\begin{table}[]
\begin{tabular}{c|cl}
cell1 & cell2 & cell3 \\ \hline
cell4 & cell5 & cell6
\end{tabular}
\end{table}
```

- You can declare merged cells, partial horizontal and vertical lines, this easily becomes way too complex.

Tables - the easy way

- Use this website www.tablesgenerator.com.



What are *floats*?

- You may have noticed a blank space in the previous code.
- With the information inside the [ ], LaTeX decides where it will draw the table.

```
1  \begin{table}[]
2  \centering
3  \caption{My caption}
4  \label{my-label}
5  \begin{tabular}{lll}
6   &  &  \\
7   &  &  \\
8   &  &
9  \end{tabular}
10 \end{table}
```

Types of *float*

- There are multiple types of *floats*:
  - H - Draws the *float* exactly where it is declared, may deform the text.
  - h - Draws the *float* close to where it is declared, this avoids deforming the text.
  - t - Draws the float at the top of the page in which it is declared.
  - b - Draws the *float* at the bottom of the page in which it is declared.
  - p - Draws the float in a page restricted to *floats*.
- The {figure} environment also uses floats *floats*.
- Use the package {float}

Figures and images

How to declare an image

- Use the {graphicx} package.
- Images need to be inside a folder where LaTeX knows it should look.

```
\graphicspath{ {pathtofolder1}{pathtofolder2} }
```

- Images should be declared inside the {figure} environment.

```
\begin{figure}[float]
\centering
\includegraphics[figure alterations]{imagename}
\end{figure}
```

- PNG, JPG, PDF are all acepted. Other file types are as well, check google in case of doubts.
- Multiple properties can be altered, check en.wikibooks.org/wiki/LaTeX/Importing_Graphics

Lists and enumerations

How to make a list

- The {itemize} environment generates unnumbered lists.
- The {enumerate} environment generates numbered lists.
- Nested lists are very much possible.
- Items are identified by the \item command.

```
\begin{itemize}
\item First item of the unnumbered list
\begin{enumerate}
\item First item of the numbered sublist
\item Second item of the numbered sublist
\end{enumerate}
\item Second item of the unnumbered list
\end{itemize}
```

Equations and other math topics

Math environments

- Use the {amsmath} package.
- $equation$ generates an inline equation, can be included in the middle of a sentence.
- $$equation$$ generates a separated, centred equation.
- The {equation} environment generates numbered equations, this is the best option.

```
\begin{equation}
equation
\end{equation}
```

- A blank line inside a math environment causes a compilation error!

Greek letters and other special symbols

- You need to use the letter names in english.
  - \alpha writes α.
  - \beta writes β.
  - etc
- There are arrows and mathematical symbols
  - \rightarrow writes $\rightarrow$.
  - \simeq writes $\simeq$.
  - etc
- All of these symbols can only be used in a math environment.
- Check the list here en.wikibooks.org/wiki/LaTeX/Mathematics

Fractions, parentheses and square roots

- Inside a math environment, it's declared as:

```
\frac{numerator}{denominator}
```

- You can have a parentheses with necessary size to envelop the fraction:

```
\left(\frac{numerator}{denominator}\right)
```

- This method for parentheses works with [, { e ".".
- Using \left.something\right) causes only the right parenthesis to be drawn.
- Having a mismatched number of \left or \right causes a compilation error!
- Roots envelop the whole radicand:

```
\sqrt[index]{radicand}
```

Superscripts, subscripts, vectors and accents

- The ^ symbol puts things in superscript, this is how you write powers.

basis^{exponent} $\Rightarrow \text{basis}^{\text{exponent}}$

- The _ symbol puts things in subscript, this is how you write indices.

basis_{subscript} $\Rightarrow \text{basis}_{\text{subscript}}$

- Vectors are declared by the $\backslash vec\{\}$ command.

$\backslash vec\{v\} \Rightarrow \vec{v}$

- For more, see en.wikibooks.org/wiki/LaTeX/Mathematics

Referencing content

What is a reference?

- To call, by a number, some equation, figure or table.
- There are 3 different commands for this:
  - \label{identificationtext}
  - \ref{identificationtext}
  - \eqref{equationidentificationtext}
- You can call the reference before and after it appears in the text.
- LaTeXdeals with the pesky problem of numbering.

Referencing equations

- Just add a label to the equation:

```
\begin{equation}\label{labeltext}
equation content
\end{equation}
```

- You then call the reference with the \eqref command:

"As demonstrated in relation \eqref{labeltext}..."

- This command is made especially for equations, the reference appears between parenthesis.

Referenciar tabelas e figuras

- The figure/table needs to have a caption.
- Just add a label to the figure/table.

```
\begin{table}[]
\caption{legend}
\label{labeltext}
\begin{tabular}{c|cl}
Table content...
\end{tabular}
\end{table}
```

- You then call the reference with the \ref command.
- Usually, table captions are placed above the table.

Referenciar tabelas e figuras

- The figure/table needs to have a caption.
- Just add a label to the figure/table.

```
\begin{figure}[float]
\centering
\includegraphics[...]{imagename}
\caption{legend}
\label{labeltext}
\end{figure}
```

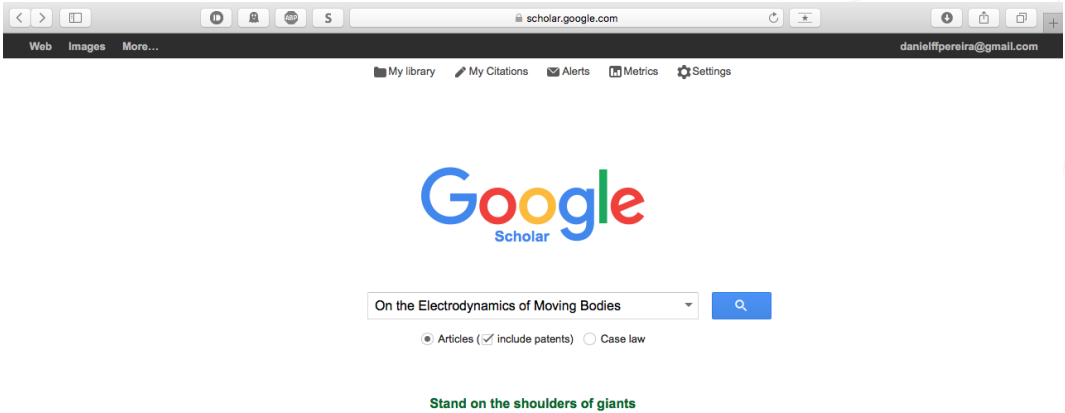
- You then call the reference with the \ref command.

Bibliographies and citations

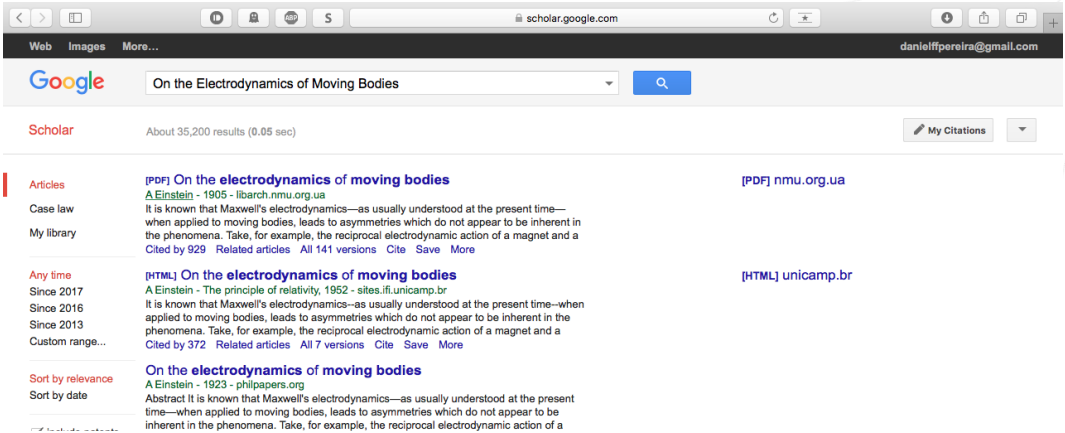
How to make a bibliography

- Easiest way is to have a .bib file.
- This file can be made by hand or with a reference management software (JabRef, Mendeley, Bibdesk or other).
- I'll show you how to do it by hand.
- Generate a .bib file, somehow, by changing the extension of a .txt created with notepad, for example.
- Go get the reference text and copy it into the .bib file.

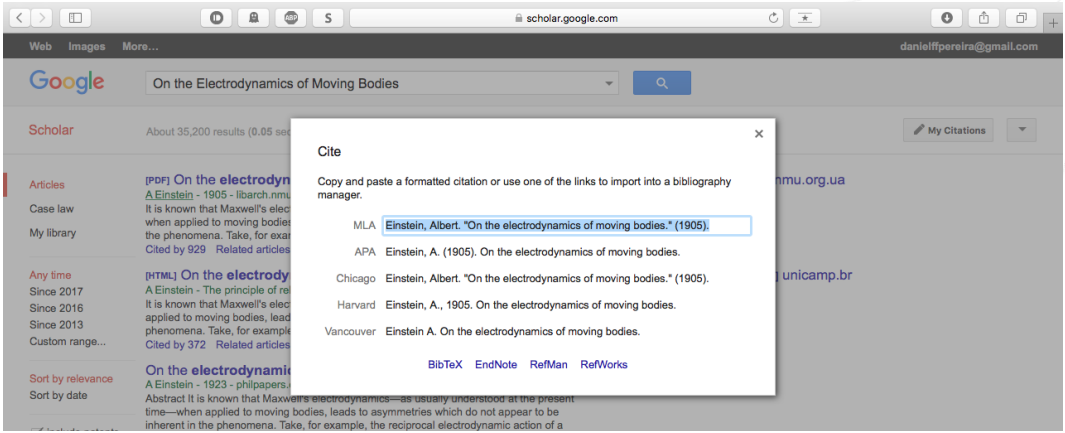
Where to get references



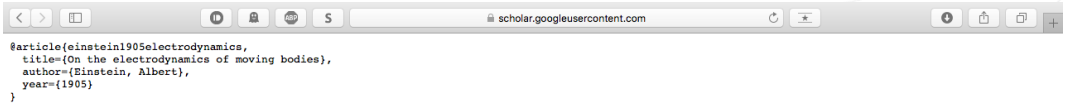
Where to get references



Where to get references



Where to get references



Understanding the citation text

```
@article{einstein1905electrodynamics,
  title={On the electrodynamics of moving bodies},
  author={Einstein, Albert},
  year={1905}
}
```

- Different publications want different formats.

```
@typeofsource{citetext,
  title={Source title},
  author={Authors},
  year={Publication year}
}
```

How to insert the bibliography in the document

- After preparing a .bib file, you need to feed it to LaTeX.

```
\bibliography{bibliography}
```

- There are different styles of bibliographies, they change the way things are presented.

```
\bibliographystyle{plain}
```

- By default, LaTeXonly includes cited sources in the bibliography, if you want uncited sources to be included, use the code:

```
\nocite{*}
```

How to cite a source

- After having included the bibliography in the document, this is cited with the \cite{citetext} command.
- If you wish to cite multiple sources at the same time, do:

\cite{citetext1,citetext2,citetext3,...}

- For more, see en.wikibooks.org/wiki/LaTeX/Bibliography_Management

This concludes the LaTeX mini-workshop

Introduction to Git

Intro

Initial notions

- What is it?
  - A database control and sharing system.
- GitHub is a very popular option, it's free and open. Create an account on GitHub.
- You need to install the git distribution.
  - Windows: `gitforwindows.org`
  - Mac: `sourceforge.net/projects/git-osx-installer/files/`
  - Linux: run the following code in the console (this should work for most distros)

```
sudo apt-get update
sudo apt-get install git
```

- You should use a Git client:
  - GitKraken: `www.gitkraken.com/git-client`
  - GitHub Desktop: `desktop.github.com`

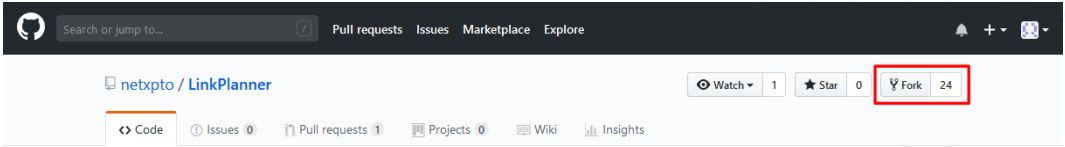
What is a repository

- A repository is a data structure that:
  - Stores a set of files and/or a directory structure.
  - A historical record of the changes to those files.
- The main repository lives somewhere in a server.
- You can **clone** a copy of the repository to your PC.
- Changes are made locally to the cloned repository can be made permanent by **committing** to it.
- Changes can then be **pushed** to the external repository.
- If you are working on another computer, you can then **pull** the changes from the external repository.

Forking repositories

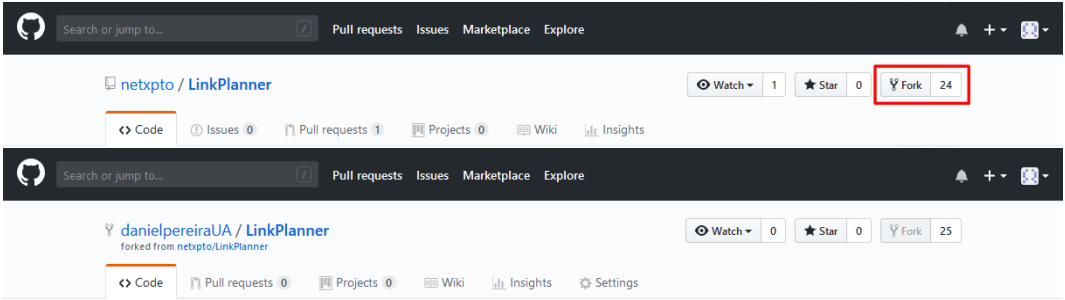
What the fork?

- A fork is a copy of another repository.
- In the GitHub website, navigate to the repository you want to fork.



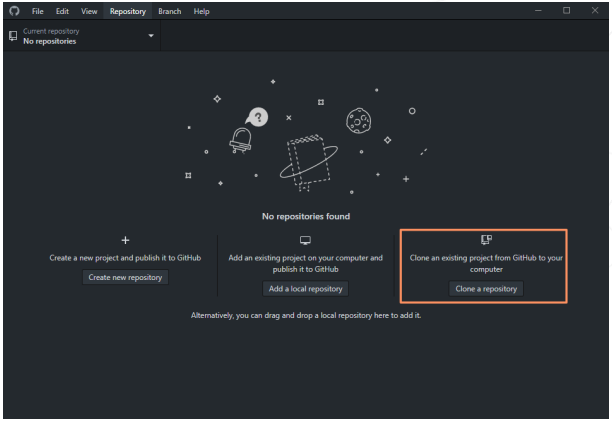
How to fork a repository

- A fork is a copy of another repository.
- In the GitHub website, navigate to the repository you want to fork.



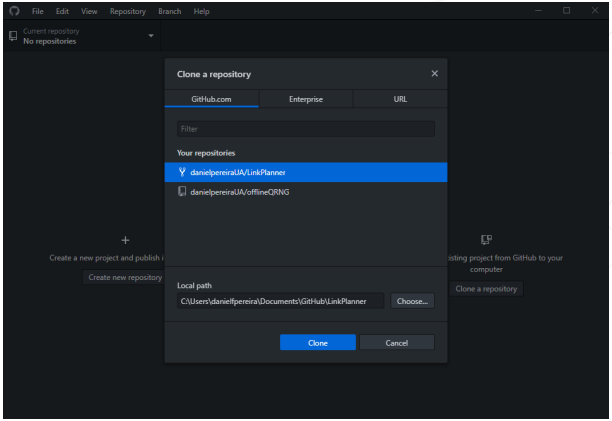
How to clone your fork

- This is not the only way to do it, but it is the easiest.
- In the GitHub Desktop app, choose to *clone a repository*.



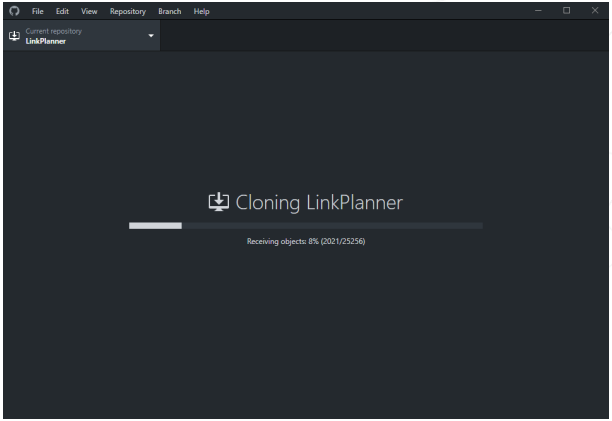
How to clone your fork

- This is not the only way to do it, but it is the easiest.
- In the GitHub Desktop app, choose to *clone a repository*.



How to clone your fork

- This is not the only way to do it, but it is the easiest.
- In the GitHub Desktop app, choose to *clone a repository*.
- Then you just have wait while it downloads, may take a while.

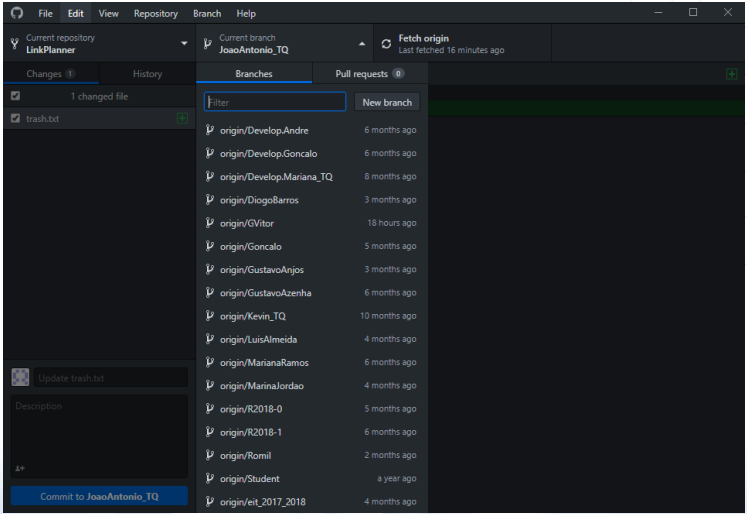


Working inside your fork

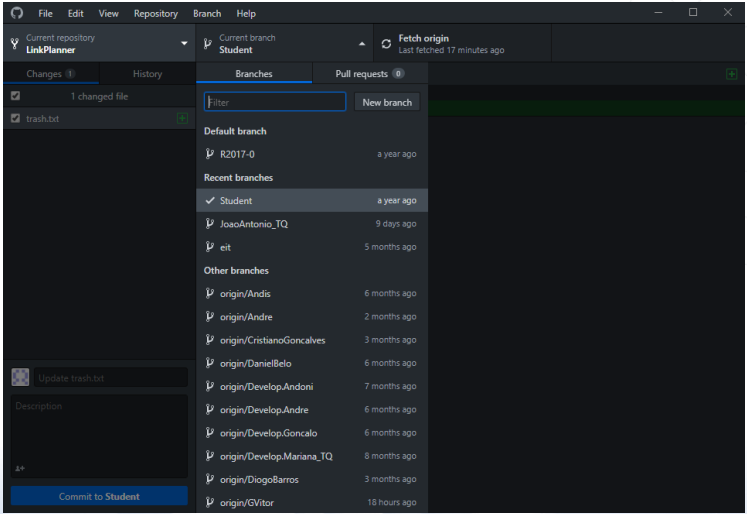
Branches

- What is a branch?
  - You can see it as a split of a repository inside it.
  - While a fork is to another account, a branch remains in the same account.
  - Allows code to be tested before it is included in the main branch.
- You won't have to worry about branches much in this class, only that you work on the branch allotted to you.

Branches

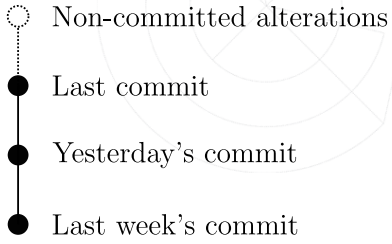


Branches

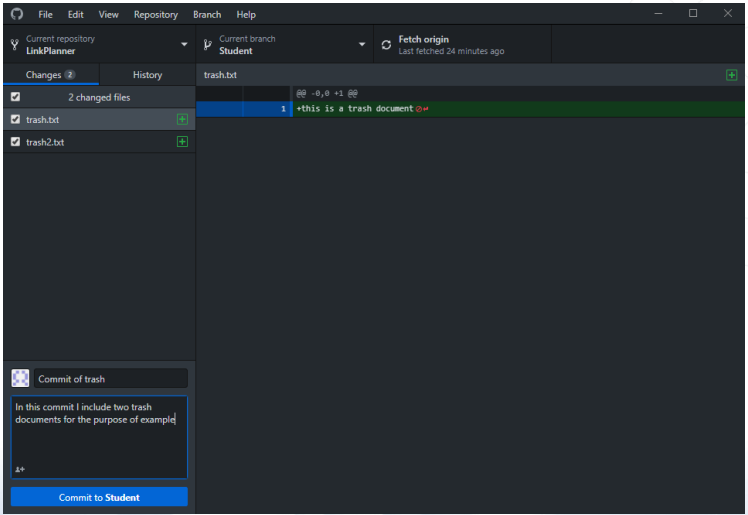


Committing, pushing and pulling

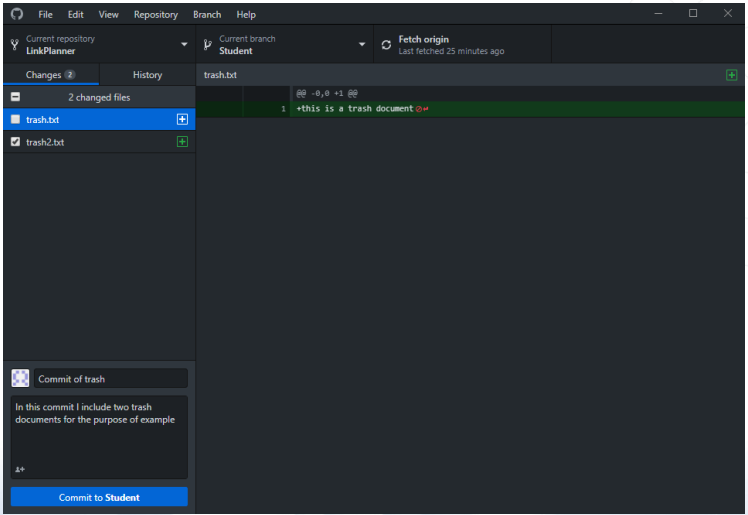
- Alterations made on your **clone** (that lives on your computer) can be made "official" by *committing* to them.
- You can discard changes by *checking out* the version of the latest commit. You can even **check out** a version of a file from any previous commit.
- The alterations you make this way are local to your machine, you need to **push** them to your "cloud" repository.
- If you wish to work on your repository on another machine, you will need to **pull** the latest version from the "cloud" repository.



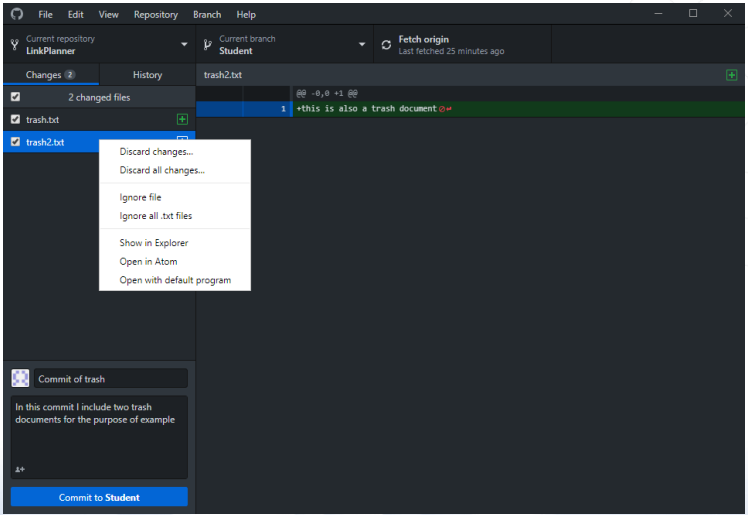
Committing, pushing and pulling



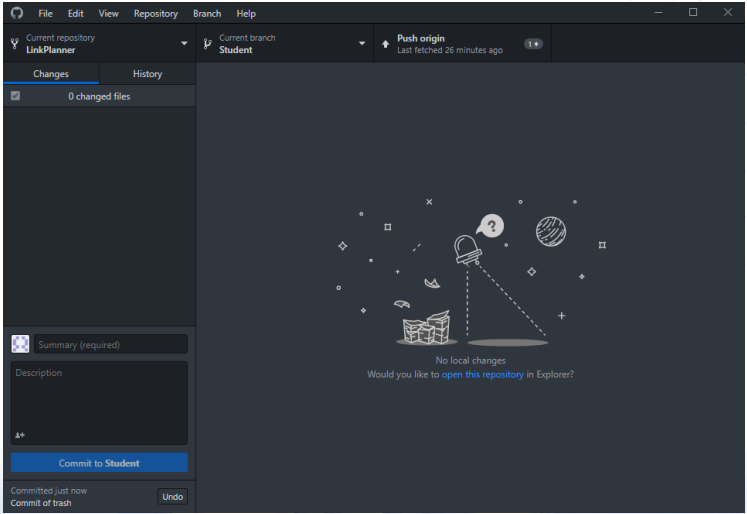
Committing, pushing and pulling



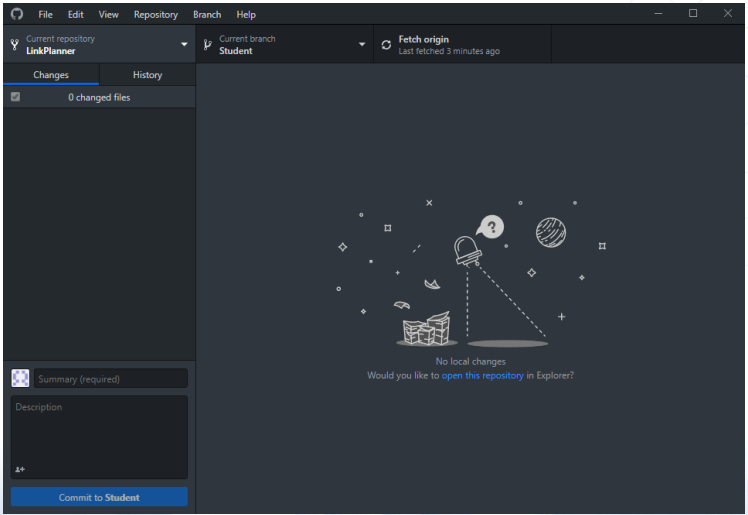
Committing, pushing and pulling



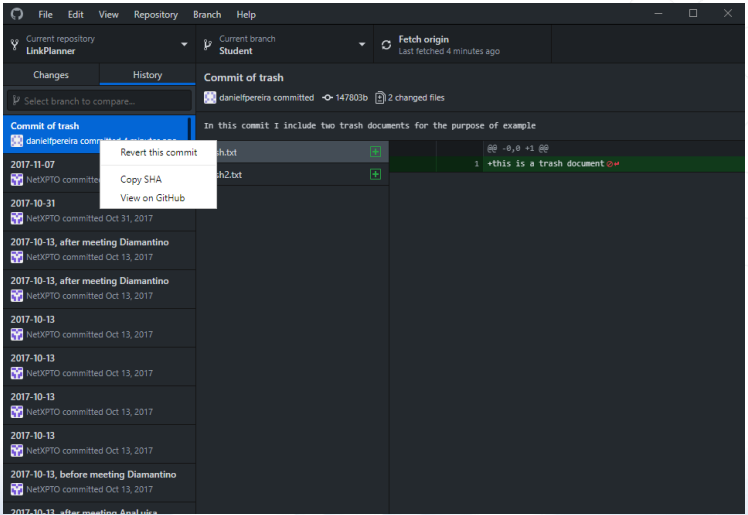
Committing, pushing and pulling



Committing, pushing and pulling



Committing, pushing and pulling

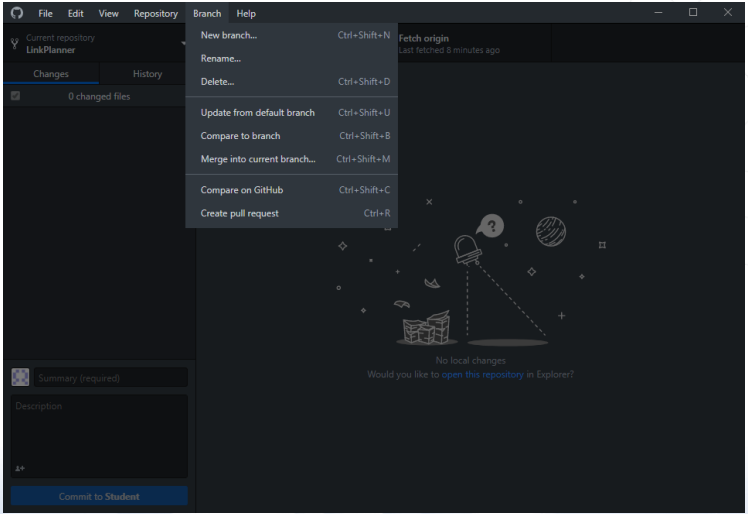


Communicating between forks

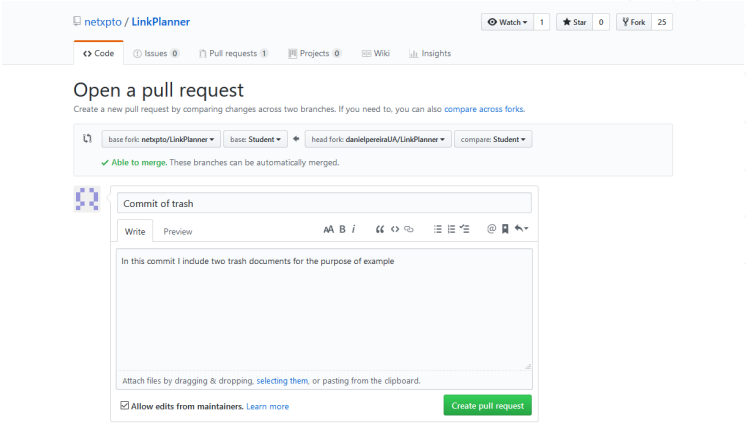
Pull requests

- The alterations you made and pushed to your account only live in your fork.
- If you want to share them with someone else (for example the owner of the original repository) you need to open a pull request.

Pull requests



Pull requests

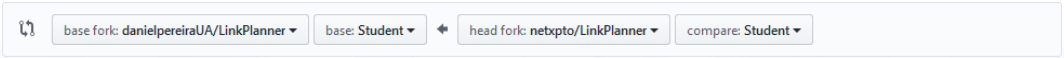


Pull requests

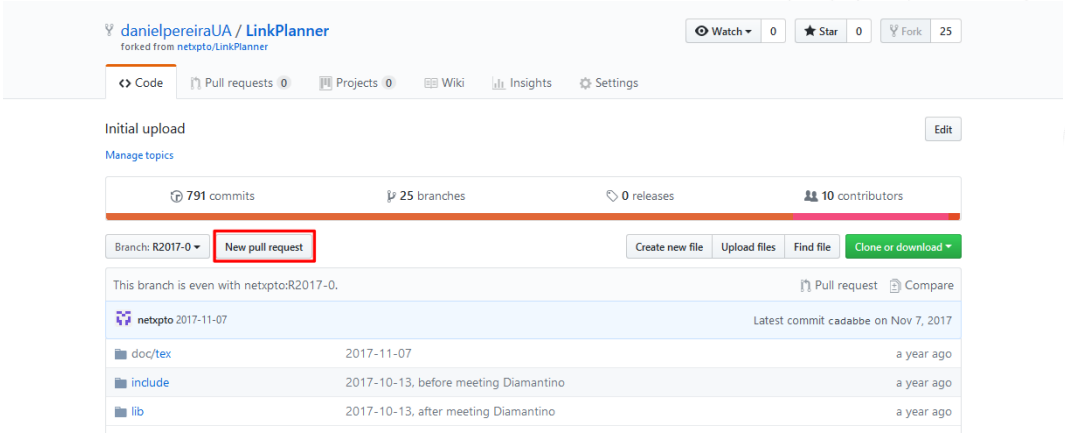
- The alterations you made and pushed to your account only live in your fork.
- If you want to share them with someone else (for example the owner of the original repository) you need to open a pull request.
- The owner of the repository you are requesting the pull to needs to approve it before it actually happens.

Pull requests

- The alterations you made and pushed to your account only live in your fork.
- If you want to share them with another fork of the same repository (for example original repository) you need to open a pull request.
- The owner of the repository you are requesting the pull to needs to approve it before it actually happens.
- Now say you want to update your fork from another fork of the same repository (for example, from the original repository).
- You do the reverse of what you did previously.
- Create a pull request from the fork you want to pull from into your fork.



Pull requests

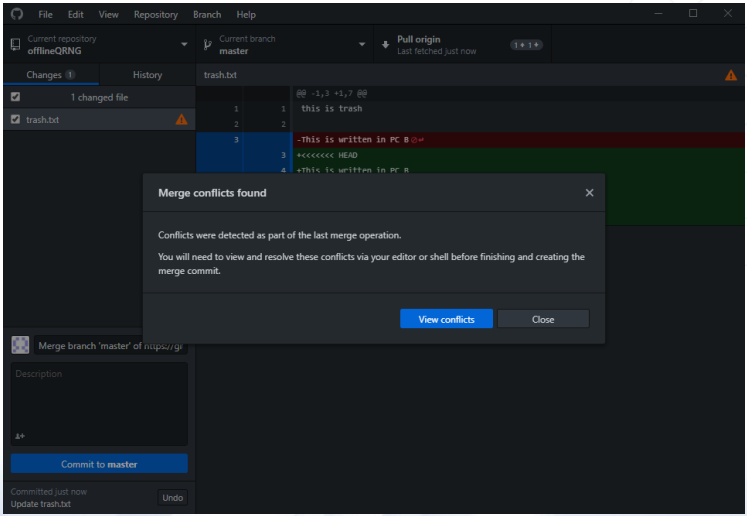


What if things go wrong?

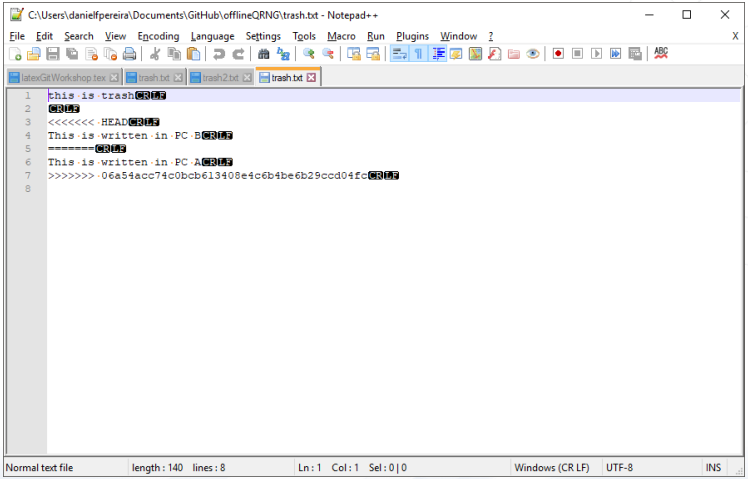
Conflicts

- A conflict arises when:
  - Change a file on PC A, push it to the cloud.
  - Change the same file on PC B before pulling the changes made on PC A.
  - When you then try to pull/push the changes made on PC A/B, you will have a conflict.
- Git knows you made changes on both machines, it evens know what changes you made in which.
- It needs you to tell it what changes to accept and what changes to discard.
- This is called merging.

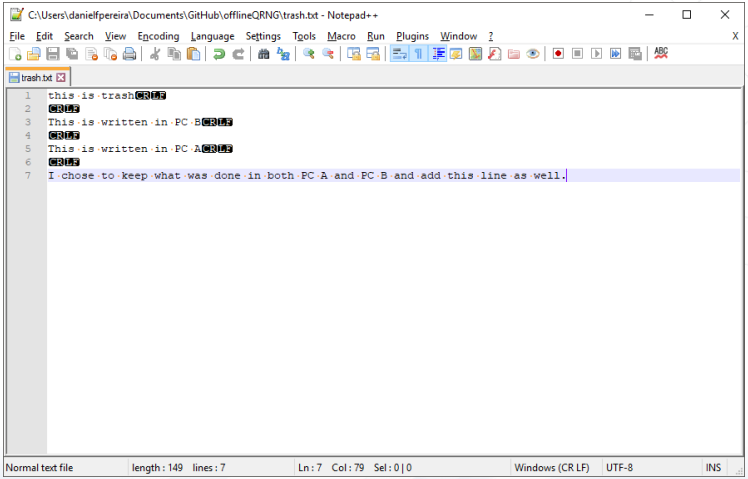
Conflicts



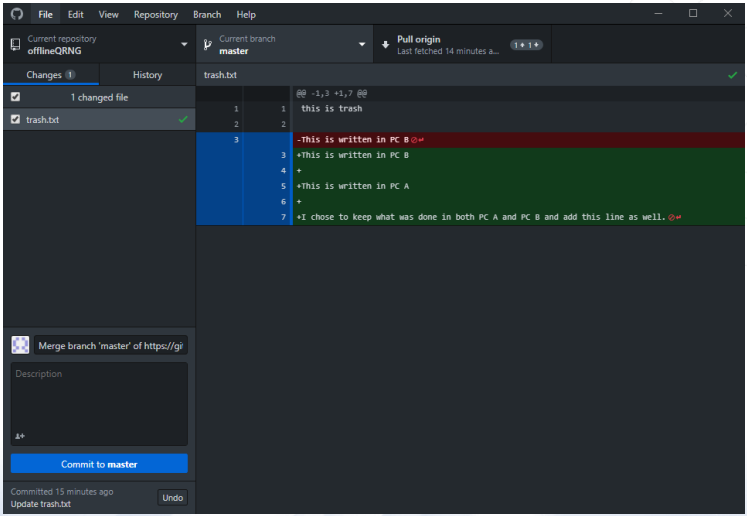
Conflicts



Conflicts

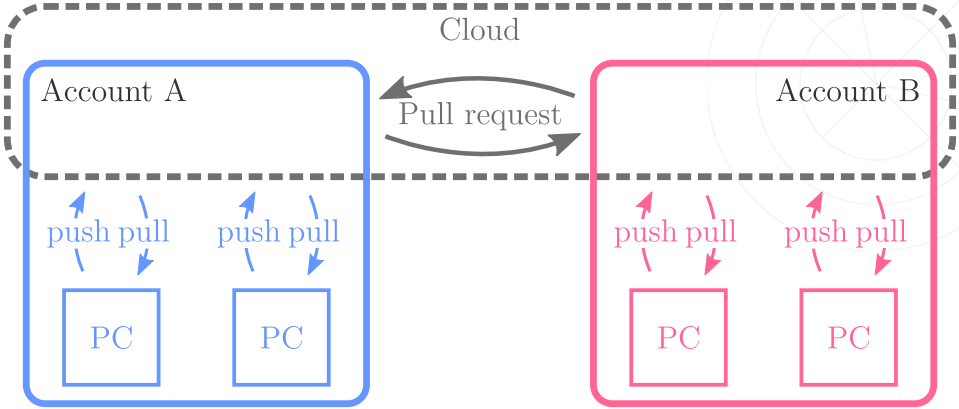


Conflicts



Summary

Topology of communications



This concludes the Git mini-workshop

The end!