

Git

Sistemas de controlo de versões

O que é um sistema de controlo de versões.

- Um sistema de controlo de versões é um sistema de desenvolvimento que **regista todas as alterações que são efectuadas aos ficheiros** de um projeto.
- Porque é importante:
 - Como designer gráfico imaginem que pretendem guardar as **diferentes versões de uma imagem ou de um layout**. Em vez de terem ficheiros espalhados com nomes que ninguém entende um sistema de gestão de versões é uma melhor solução.

Vantagens de utilizar um sistema de controlo de versões

- Um historial de todos os ficheiros de um projecto. Se algo de mal acontecer pode-se sempre reverter para uma versão anterior.
- Permite criar diferentes ramos (branches) do mesmo projeto de forma a que diferentes pessoas possam trabalhar de forma concorrente em diferentes ideias sem corromper o código dos restantes.
- Os diferentes ramos podem ser depois mesclados (merged) e os potenciais conflitos entre versões serem resolvidos sem problemas.
- Rastreabilidade de todas as alterações efectuadas.

Desenvolvimento sem riscos

- Um sistema de gestão de versões permite um desenvolvimento de software com menos riscos.
- Nenhuma equipa profissional aceitaria desenvolver um projeto sem um sistema de rastreabilidade e gestão de versões.

RCS - Sistema local baseado numa base de dados

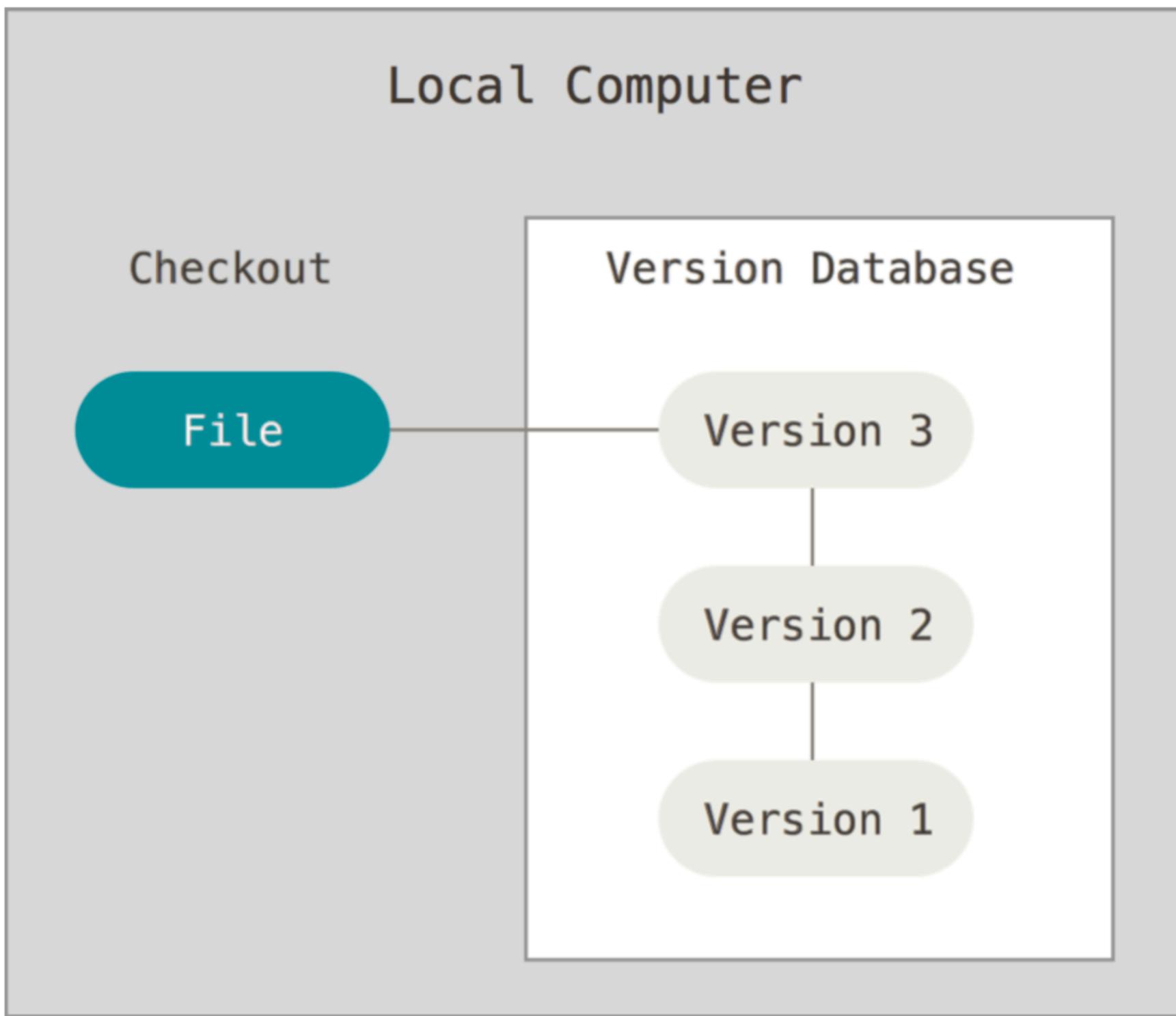


Figure 1. Local version control.

Sistemas Centralizados de Gestão de Versões.

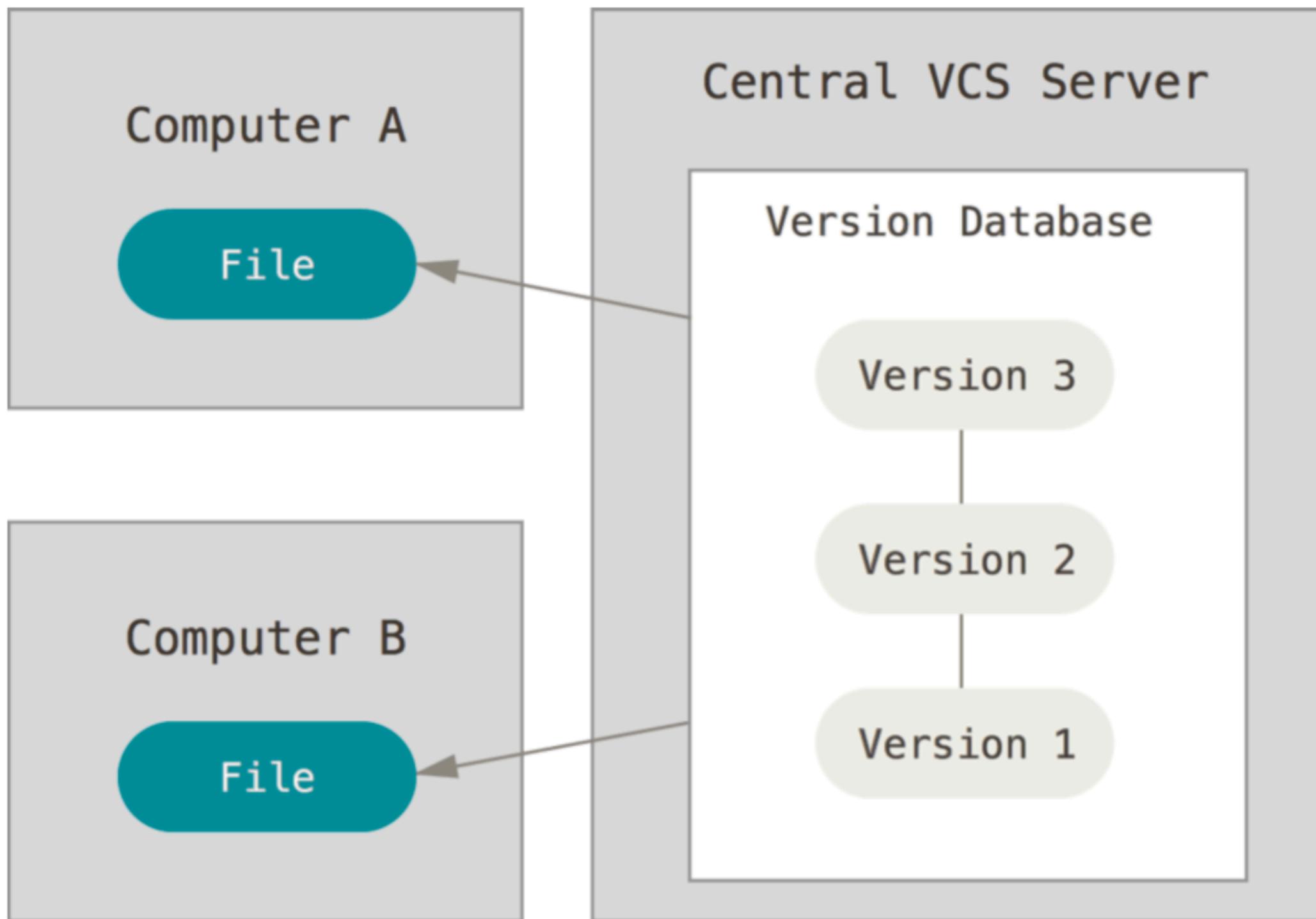


Figure 2. Centralized version control.

**Qual o problema
destes dois
sistemas?**

Single point of failure

SISTEMAS DISTRIBUÍDOS DE CONTROLO DE VERSÕES

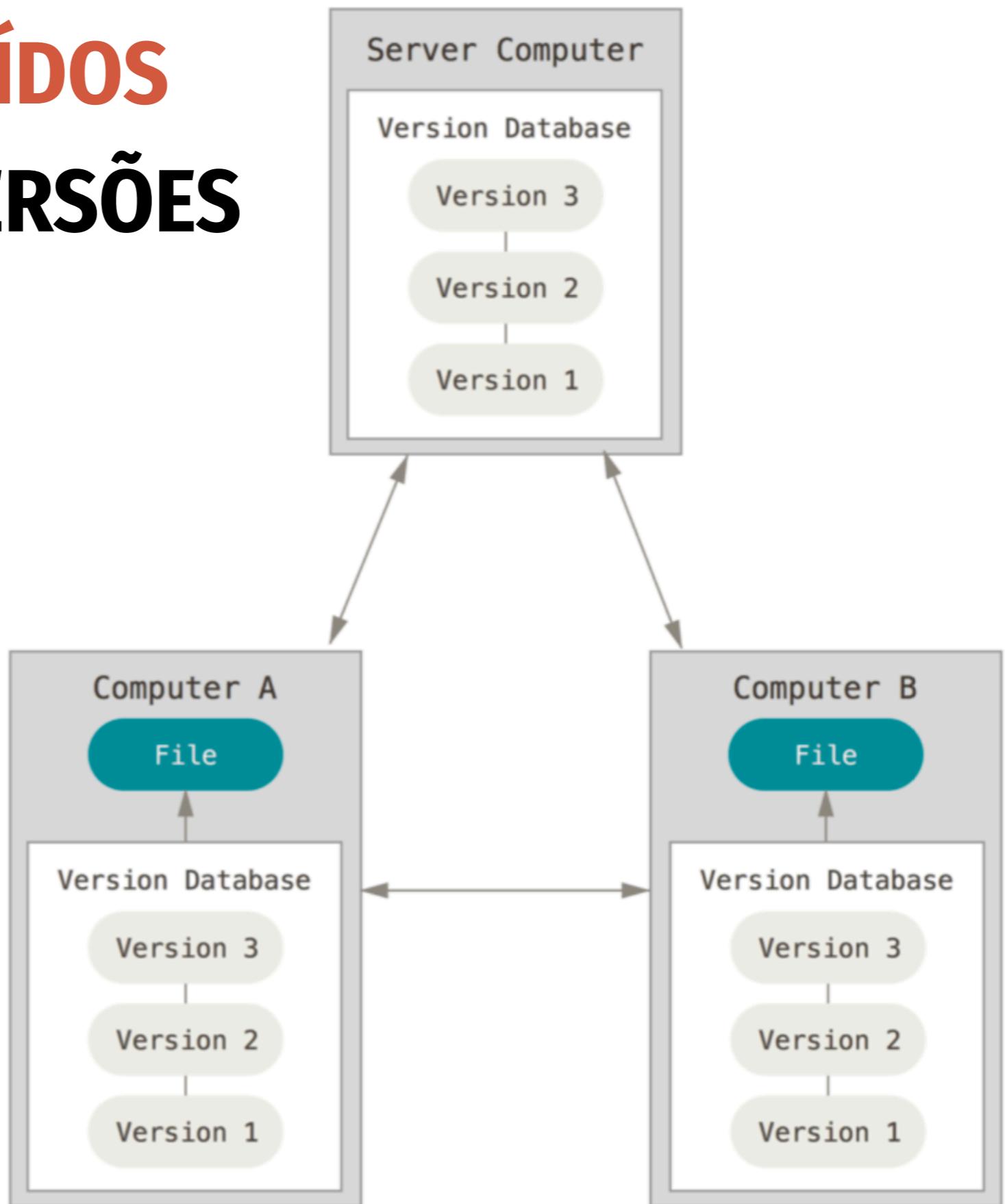


Figure 3. Distributed version control.

Enter Git

A história do Git (breve)

- Git surgiu no meio da discórdia:
- Até 2002 o **kernel de linux** foi desenvolvido um pouco ad-hoc.
- Entre 2002 e 2005 começaram a utilizar um produto comercial chamado BitKeeper para fazer a gestão de versões.
- E em 2005 o BitKeeper quiz que a comunidade open source pagasse uma licença comercial para utilizar o BitKeeper, o que levou **Linus Torvalds** a criar o **git** como alternativa.

O kernel é o conjunto de processos mais básicos do sistema operativo linux que fazem a gestão de todo o sistema. <https://www.kernel.org/>



O git pode ser obtido a partir de
<https://git-scm.com/>

Principais características do Git

- Distribuído
 - Velocidade
 - Design simples
 - Suporte para desenvolvimento não linear
 - Capaz de suportar projetos gigantescos

não tem um ponto único de falha

suporta múltiplos branches

Linux, Google, FB, Microsoft,...
todos usam git.

Como funciona o git - Snapshots vs Deltas

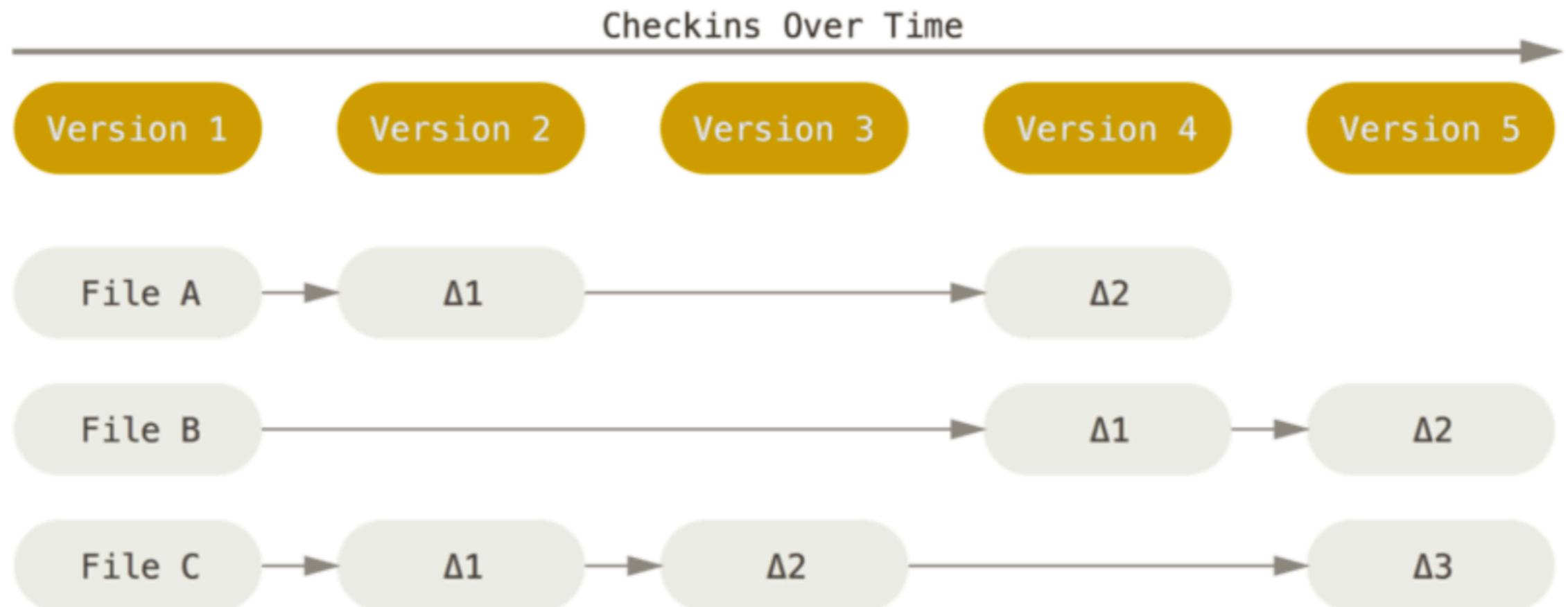


Figure 4. Storing data as changes to a base version of each file.

Como funciona o git - Snapshots vs Deltas

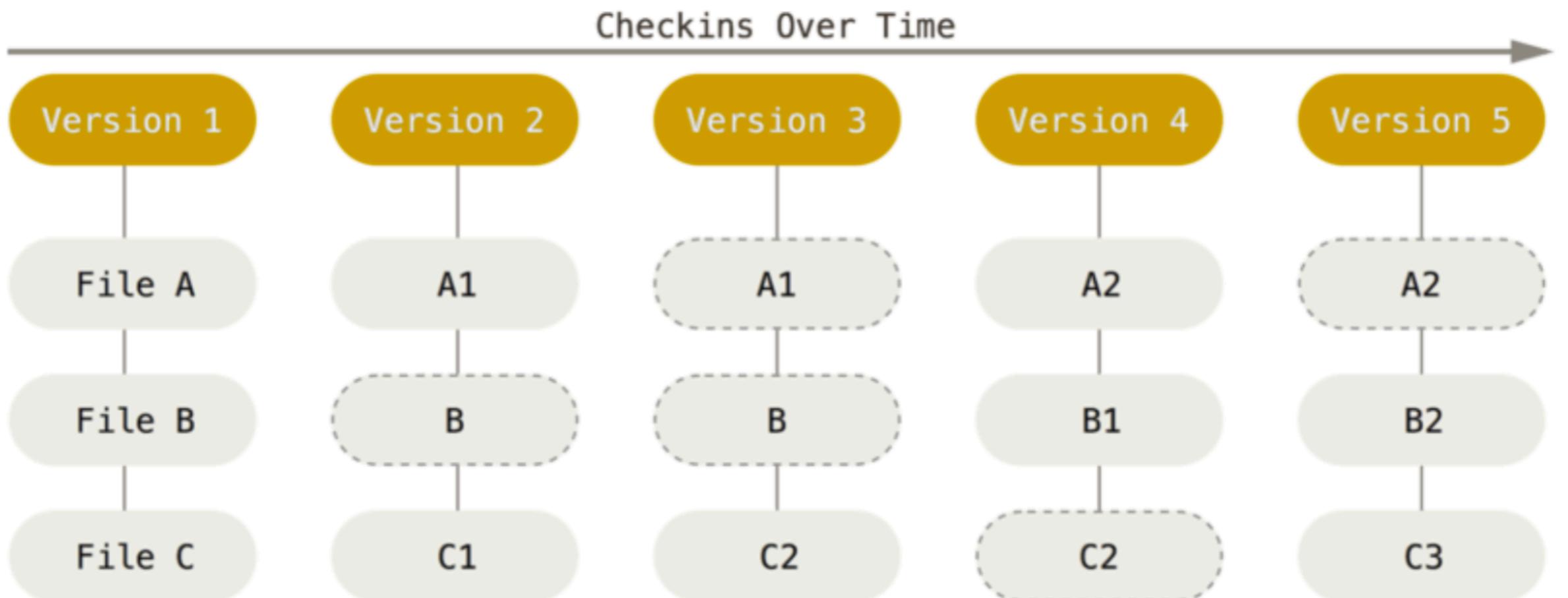


Figure 5. Storing data as snapshots of the project over time.

Importante

Três estados de um repositório Git

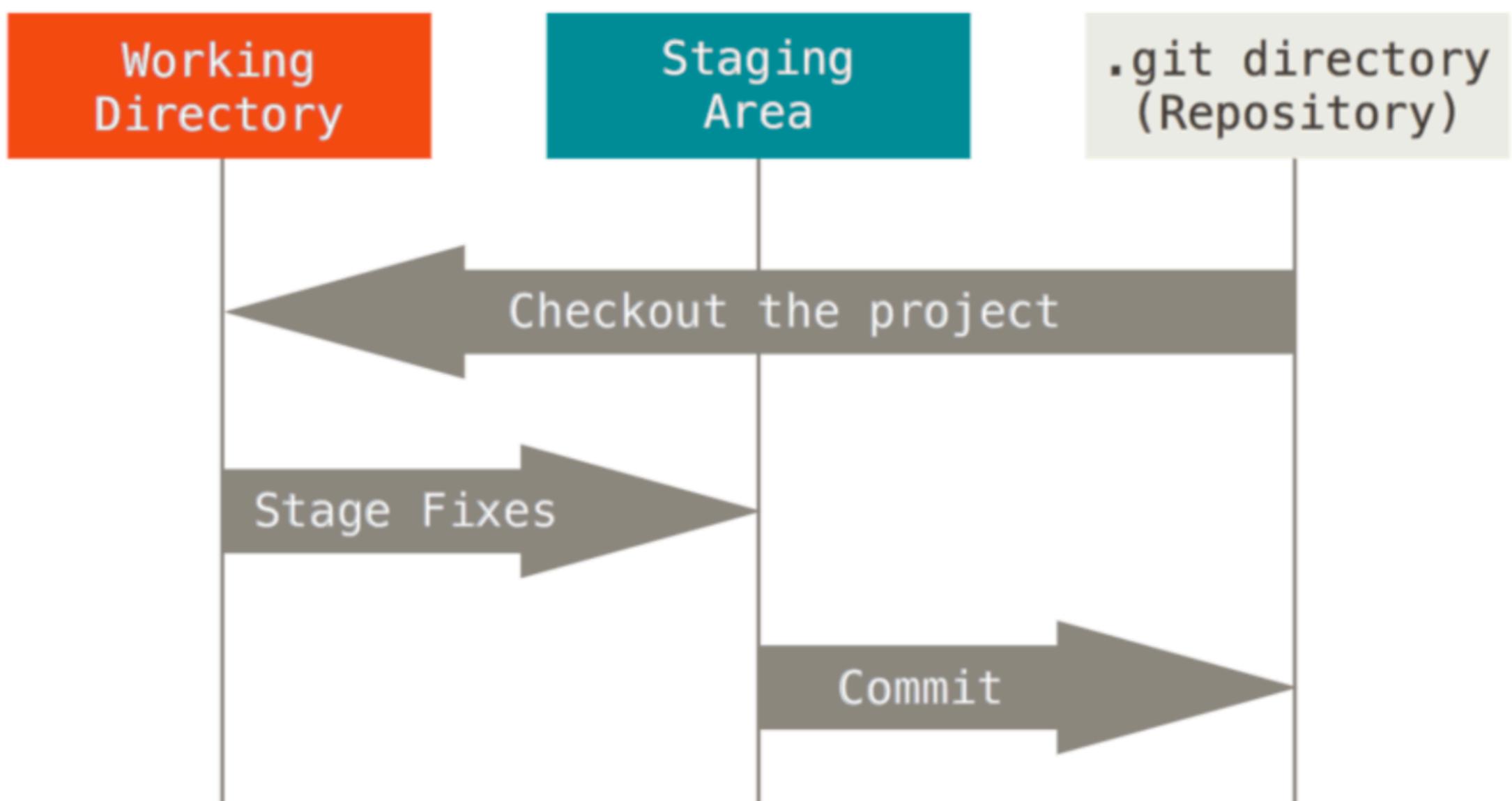


Figure 6. Working tree, staging area, and Git directory.

a directria **.git**

- a directria **.git** é onde o **git** guarda todos os metadados e a base de dados dos objectos do nosso projecto.
- Esta directria é o que é copiado para o nosso computador quando clonamos um repositório remoto.

Em muitos OSes directrias cujo nome comece por um ponto (.) normalmente são escondidas do utilizador.

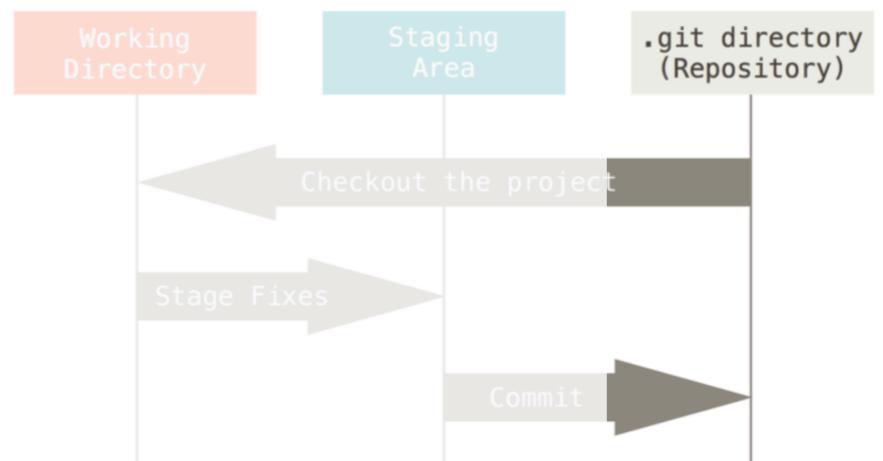


Figure 6. Working tree, staging area, and Git directory.

a directoria de Trabalho

- é um checkout de um snapshot do repositório.
- Estes ficheiros são retirados da base de dados e colocados na directoria para serem editados.

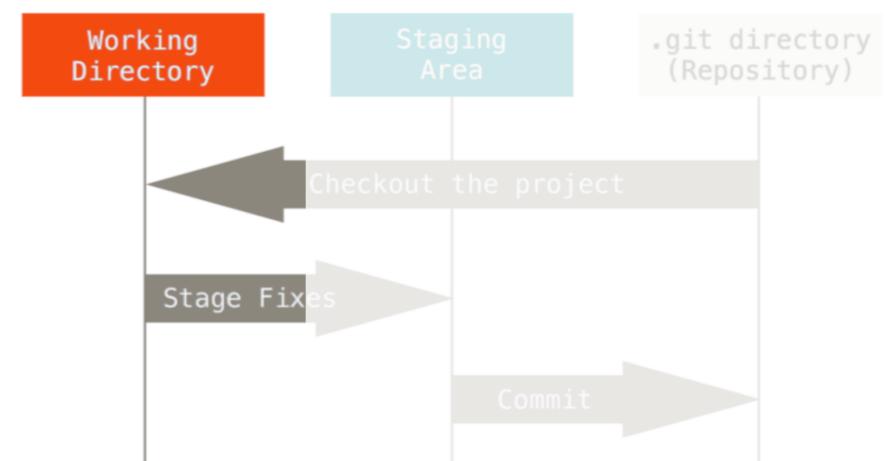


Figure 6. Working tree, staging area, and Git directory.

a directória de **Staging**

- é na prática um ficheiro que guarda informação sobre o que deve ser incluído na próxima submissão de alterações (**commit**)

Pode-se pensar que o processo de submeter alterações para um repositório é feito em duas etapas: adicionam-se as alterações à **Staging Area** e depois faz-se **Commit**.

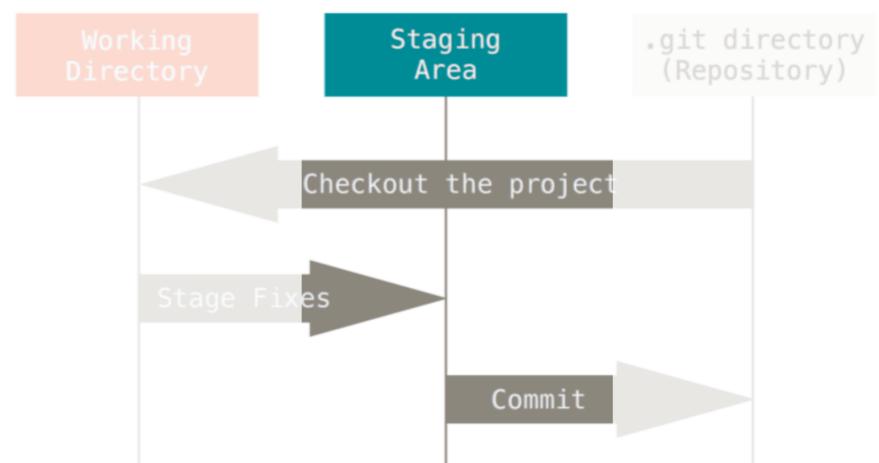


Figure 6. Working tree, staging area, and Git directory.

Instalar o Git

Instalar o Git.

- Há várias formas de instalar o git.
- A forma oficial é instalar as ferramentas a partir de <https://git-scm.com/>
 - Isto instalará as ferramentas todas que funcionam nos terminais de texto.
- Uma alternativa é instalar o software GitHub Desktop a partir de
<https://desktop.github.com/>

O GitHub é um serviço de hosting de repositórios que se baseia em Git.

O GitHub Desktop é uma interface visual para os comandos git e para os repositórios do GitHub.

[github.com - Criar uma conta](#)



Why GitHub? Enterprise Explore Marketplace Pricing

Search GitHub



Sign in

Sign up

Built for developers

GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside 31 million developers.

Username

Pick a username

Email

you@example.com

Password

Create a password

Make sure it's more than 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more](#).

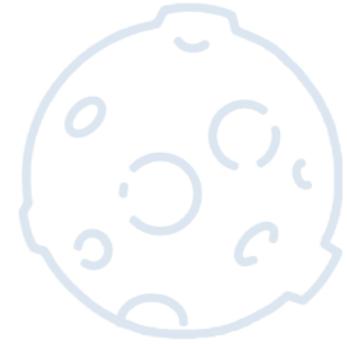
[Sign up for GitHub](#)

By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy statement](#). We'll occasionally send you account related emails.



Let's get started!

Add a repository to GitHub Desktop to start collaborating



Filter



Your Repositories

- sixhat/CasaMusical
- sixhat/ChessInfluenceViewer
- sixhat/dave.complexity
- sixhat/davtwitter
- sixhat/laggos
- sixhat/mailun
- sixhat/sixhat.github.com
- sixhat/SO1718Classes
- sixhat/swipe
- sixhat/theseus
- sixhat/utils.dave.p5.js



Clone a Repository from the Internet...



Create a New Repository on Your Hard Drive...



Add an Existing Repository from Your Hard Drive...



ProTip! You can drag & drop an existing repository folder here to add it to Desktop

Create a New Repository

Name

Description

Local Path

Initialize this repository with a README

Git Ignore

License

Current Repository **LabApps**

Current Branch **master**

Publish repository
Publish this repository to GitHub

Changes History

0 changed files

No local changes

You have no uncommitted changes in your repository! Here are some friendly suggestions for what to do next.



Publish your repository to GitHub
This repository is currently only available on your local machine. By publishing it on GitHub you can share it, and collaborate with others.

Always available in the toolbar for local repositories or

Open the repository in your external editor
Configure which editor you wish to use in [preferences](#)

Repository menu or

Show in Finder

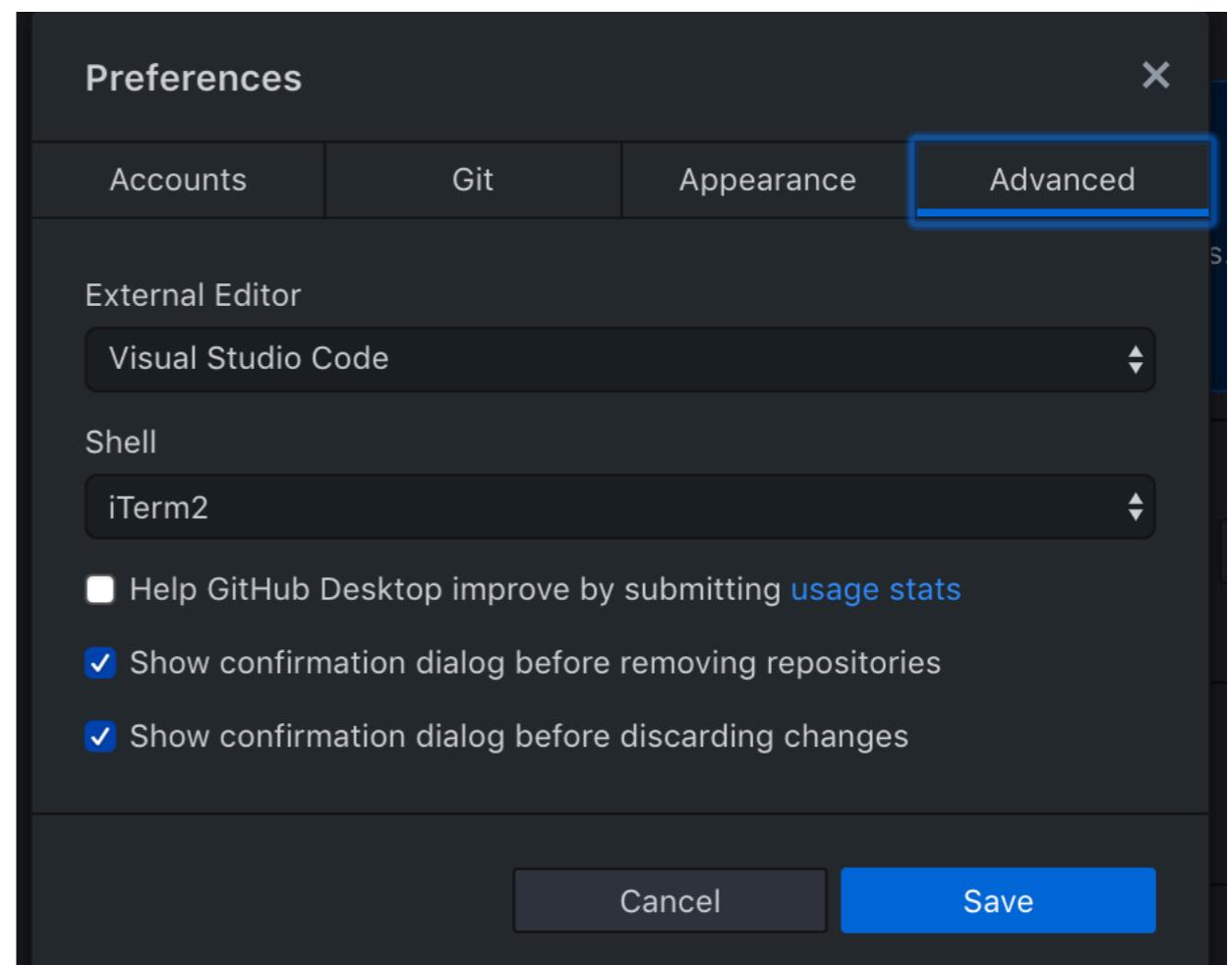
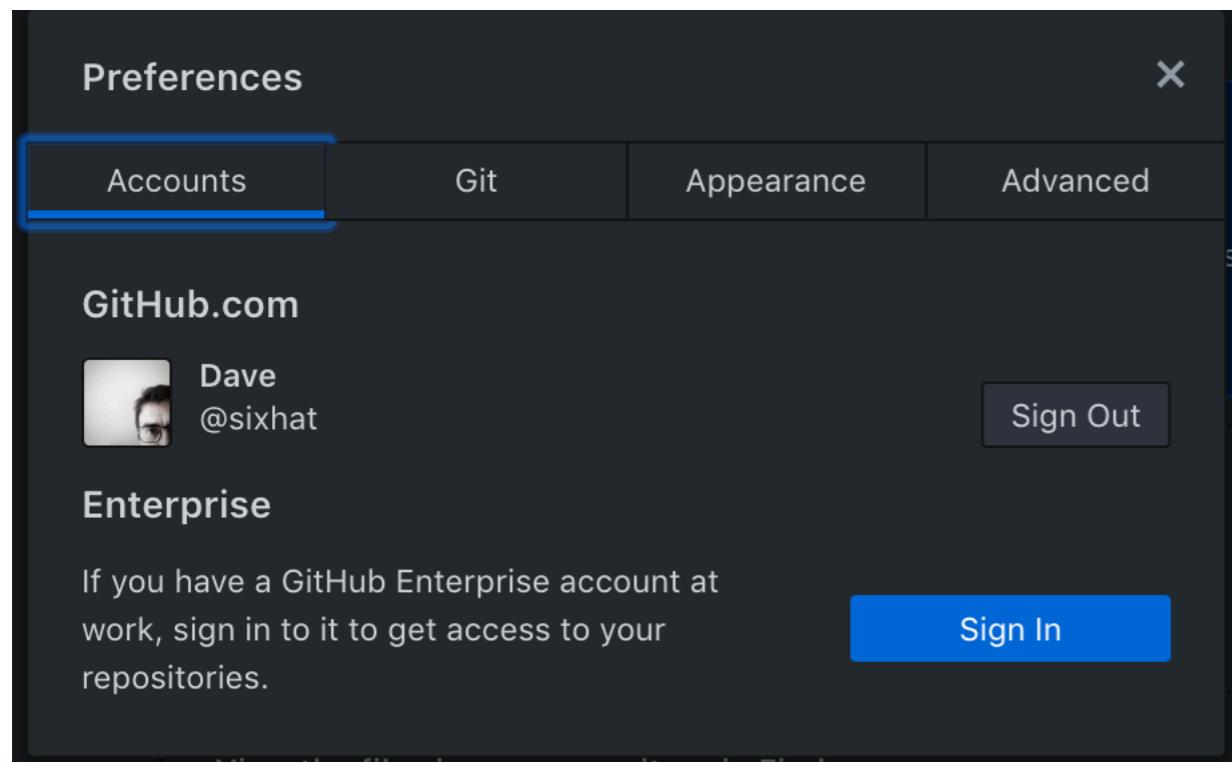
Commit to master

Committed just now

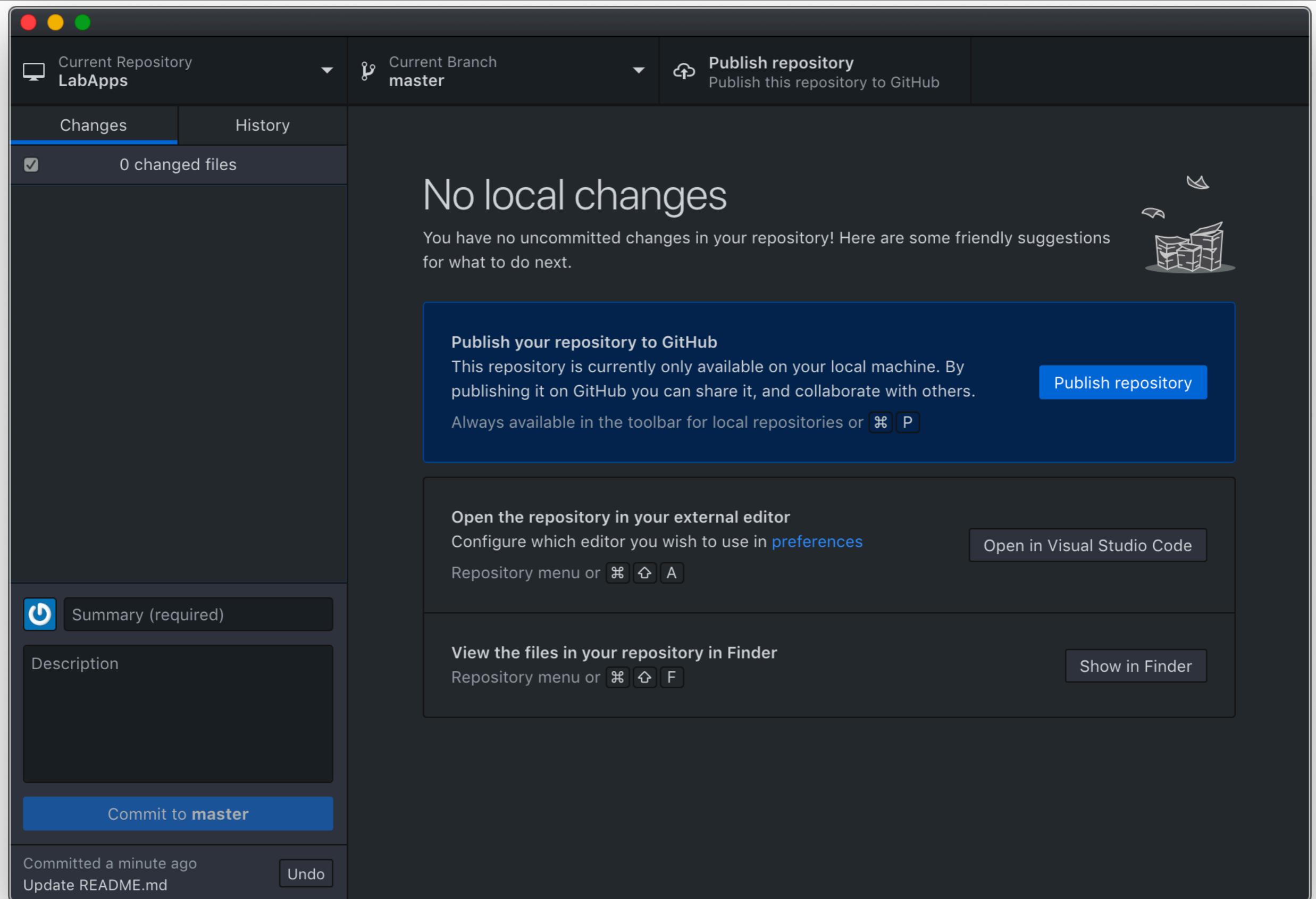
Initial commit

Undo

Configurar as vossas preferências



Publicar no github.com o primeiro repositório.



Current Repository **LabApps**

Current Branch **master**

Publish repository
Publish this repository to GitHub

Changes History

0 changed files

No local changes

You have no uncommitted changes in your repository! Here are some friendly suggestions for what to do next.

Publish your repository to GitHub
This repository is currently only available on your local machine. By publishing it on GitHub you can share it, and collaborate with others.

Always available in the toolbar for local repositories or  

Open the repository in your external editor
Configure which editor you wish to use in [preferences](#)

Repository menu or  

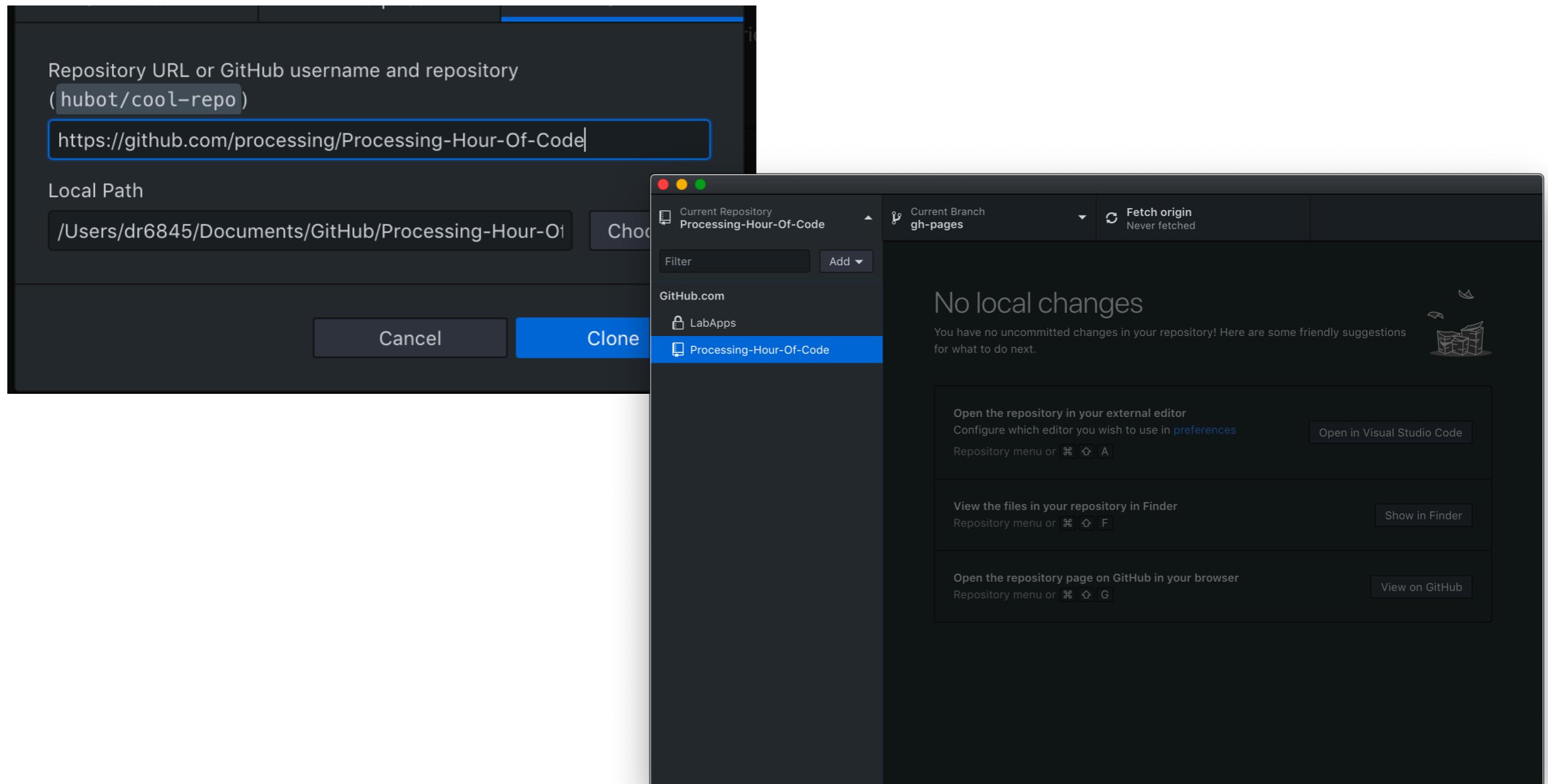
View the files in your repository in Finder
Repository menu or  

Commit to master

Committed a minute ago
Update README.md 

Clonar um repositório do GitHub.

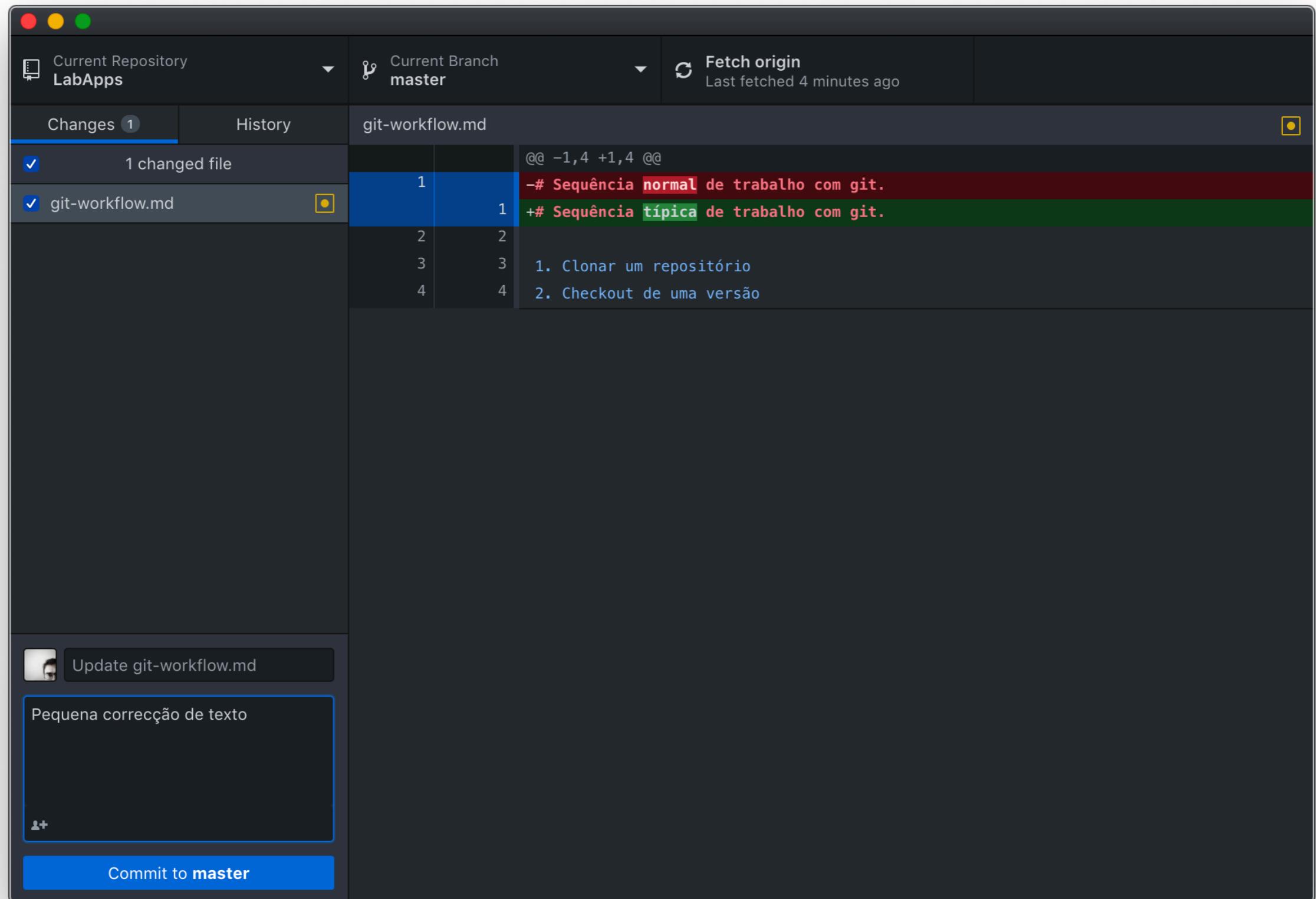
- se utilizarem o terminal
git clone <https://github.com/sixhat/LabApps>



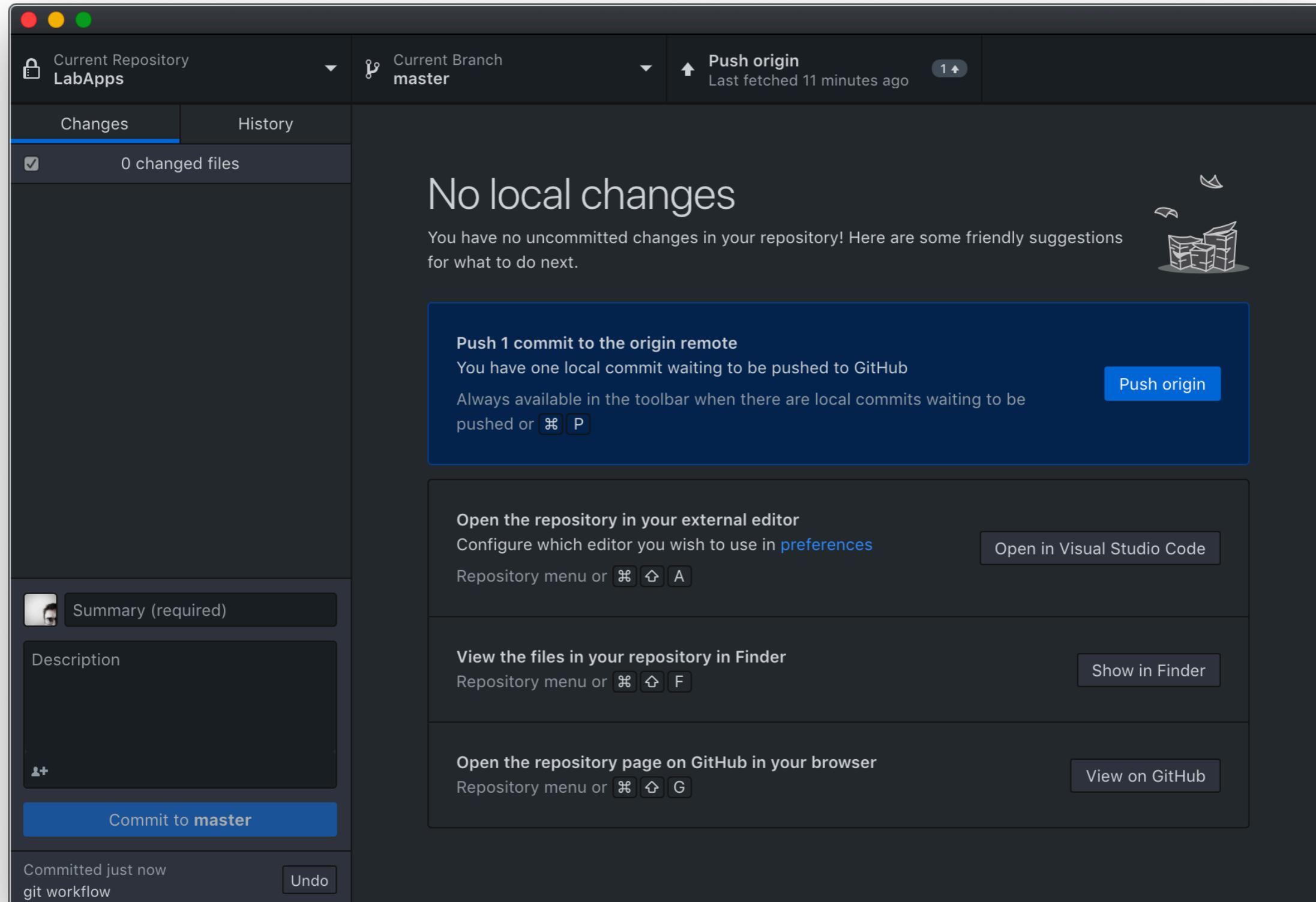
Editar localmente.

- Quando se tem um repositório local podemos fazer alterações.
- Um repositório **git** não é só para ficheiros texto. Pode conter também ficheiros noutrios formatos.
- Depois da edição (VSCode, Photoshop, etc..) o Github Desktop vai mostrar que há alterações prontas para serem adicionadas e vai colocar esses ficheiros no Stage para se poder fazer commit.

Submeter (Commit)



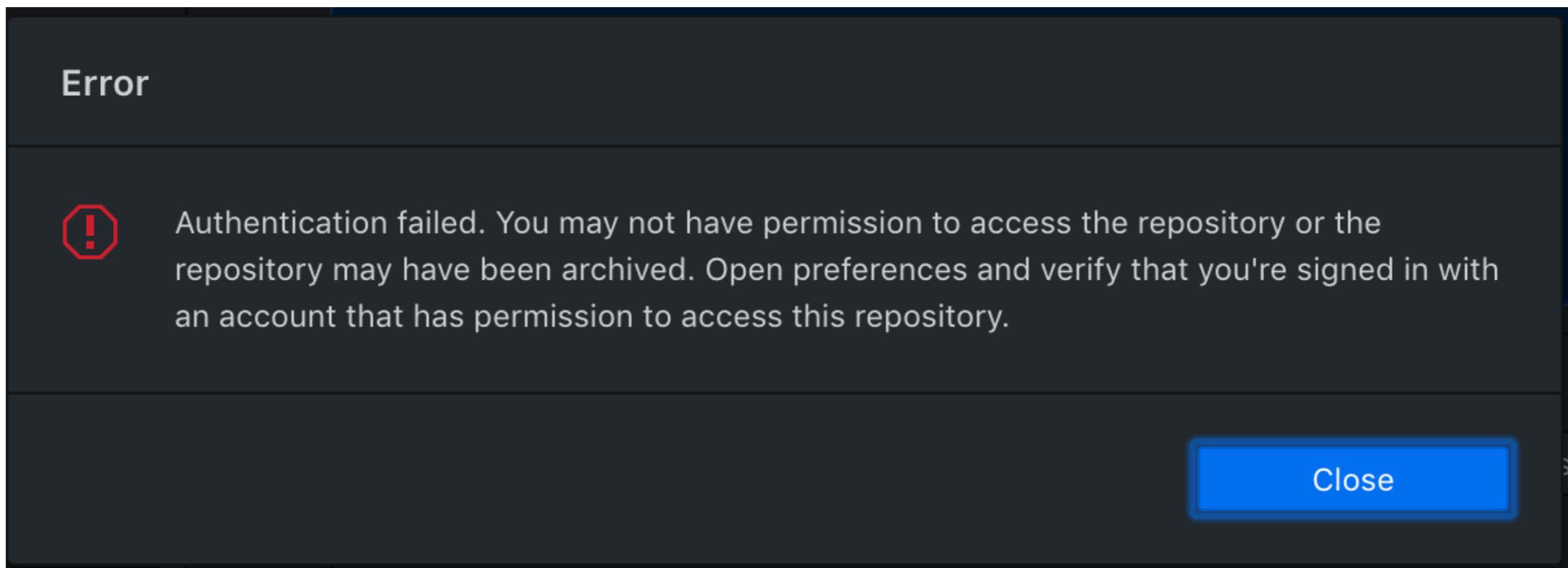
Push para o servidor do GitHub.



Como fazer push para um servidor do qual não somos colaboradores?

- O sistema vai impedir o vosso acesso.

no vosso projecto devem adicionar ao repositório todos os autores que participam para evitar esta situação.



Como contribuir para um repositório do qual não somos membros?

- Primeiro precisamos de uma copia do projecto para o qual queremos contribuir, na nossa área do *github*.
isto é conseguido através de um **fork** na página do projeto de origem.
- Depois do **fork** podemos fazer uma cópia local no nosso computador com o **clone**.
- O processo agora é normal. No fim das nossas edições podemos fazer **push** para a nossa área no *github*.
- No fim é preciso fazer um **pull request**. Um pedido aos membros do repositório original para reverem e integrarem as nossas alterações se concordarem com elas.
Mais informação de como os PR funcionam pode ser encontrada <https://help.github.com/en/articles/about-pull-requests>

Exercício

- Façam fork do repositório <https://github.com/sixhat/LabApps> fork
- Façam clone para o vosso computador. clone
- Escrevam o nome dos alunos do vosso grupo no ficheiro README.md (editar)
- Publiquem para o vosso repositório commit e push
- Façam um Pull Request para eu integrar as vossas alterações. pull request