WEB ACADEMY

Tópicos Fundamentais

Prof.ª Catarina Costa

Agenda

01 GC

02 Git

"FINAL".doc

GC

Já perdeu uma versão anterior do arquivo do projeto e precisou dela?

Alguém já modificou indevidamente um arquivo e o original não poderia ter sido perdido?

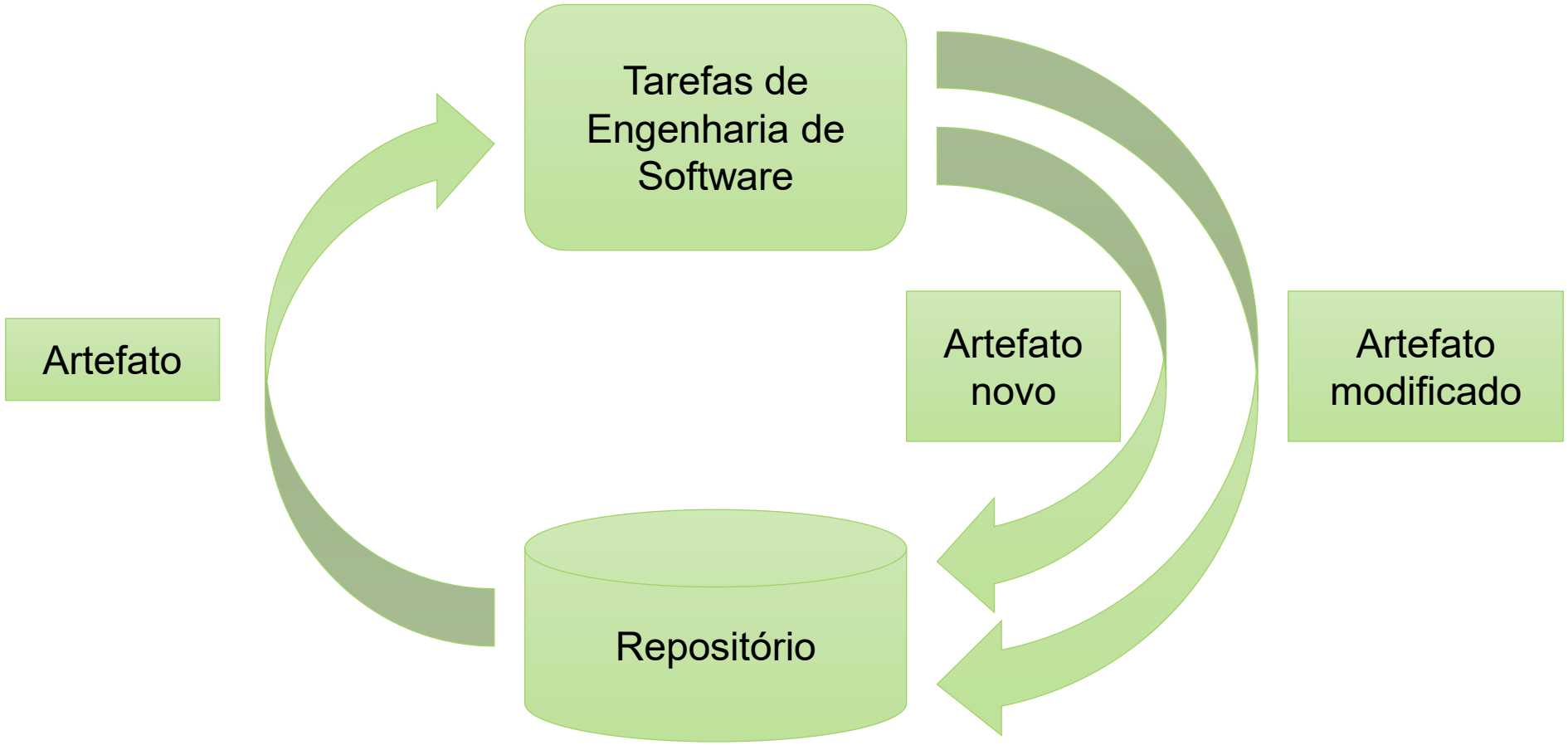
Tem dificuldade em saber quem modificou o que em um projeto?

É sobre isso...

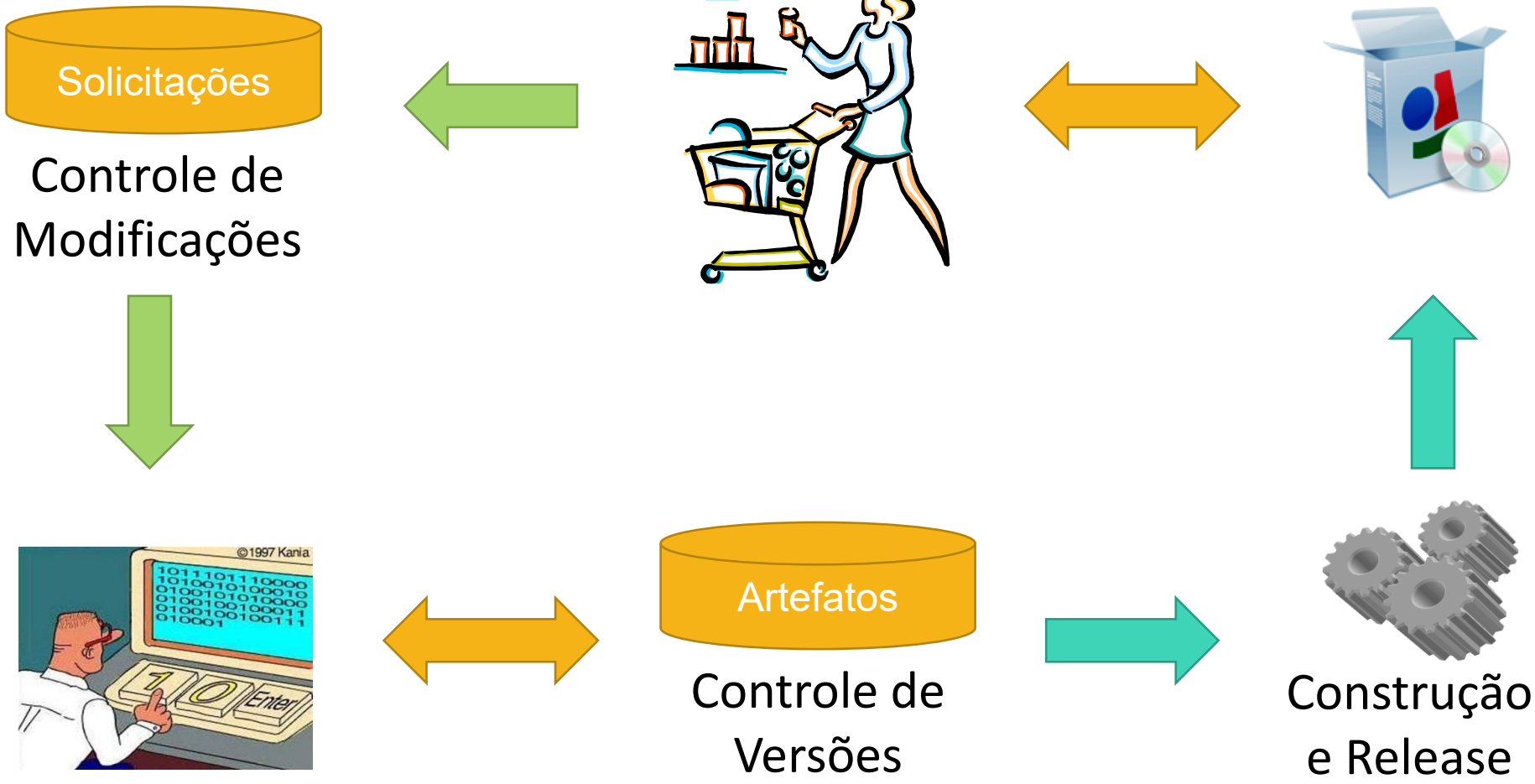
- A Engenharia de Software...
  - Abordagem disciplinada para o desenvolvimento de software
  - Grande diversidade de metodologias

- Ponto em comum nas metodologias:
  - refinamentos sucessivos de **artefatos**

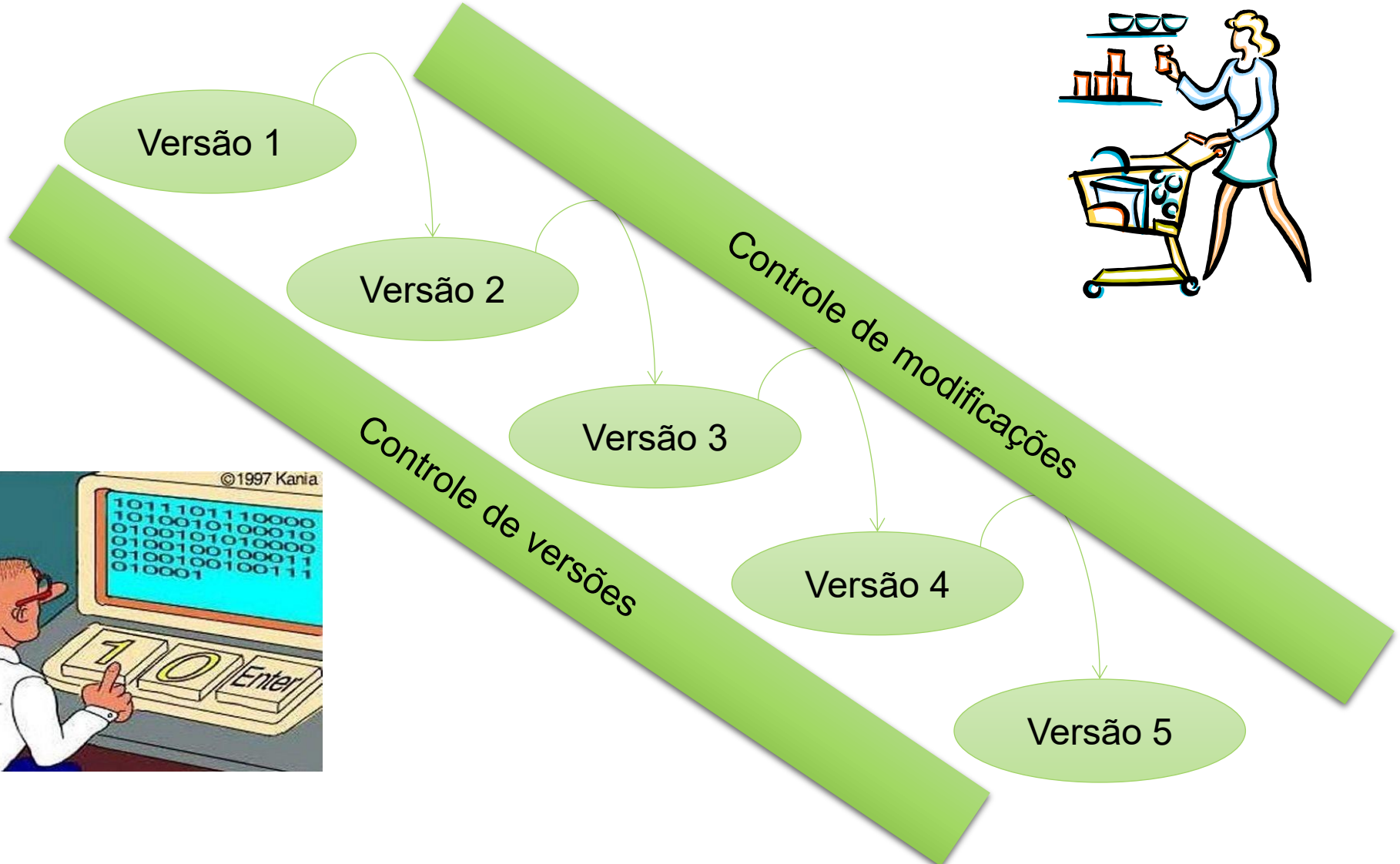
Mas onde ficam esses artefatos?



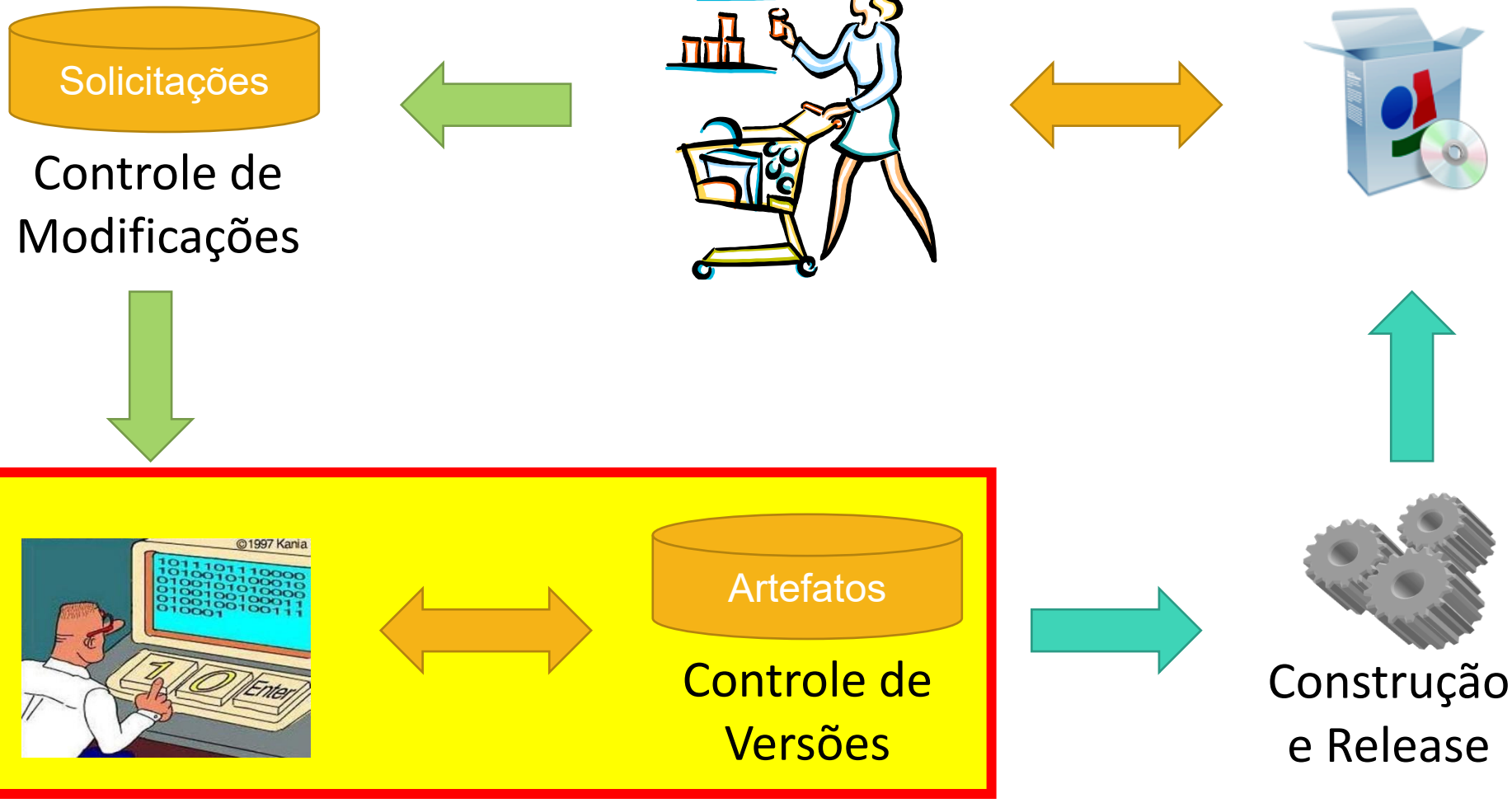
Sistema de Gerência de Configuração



Sistema de Gerência de Configuração



Sistema de Gerência de Configuração



Revisões

Gerações do iMac (1998 – 2013)

Variantes

Hatchback

Honda Civic

Sedan

Coupe

Cooperação (versões rascunho)

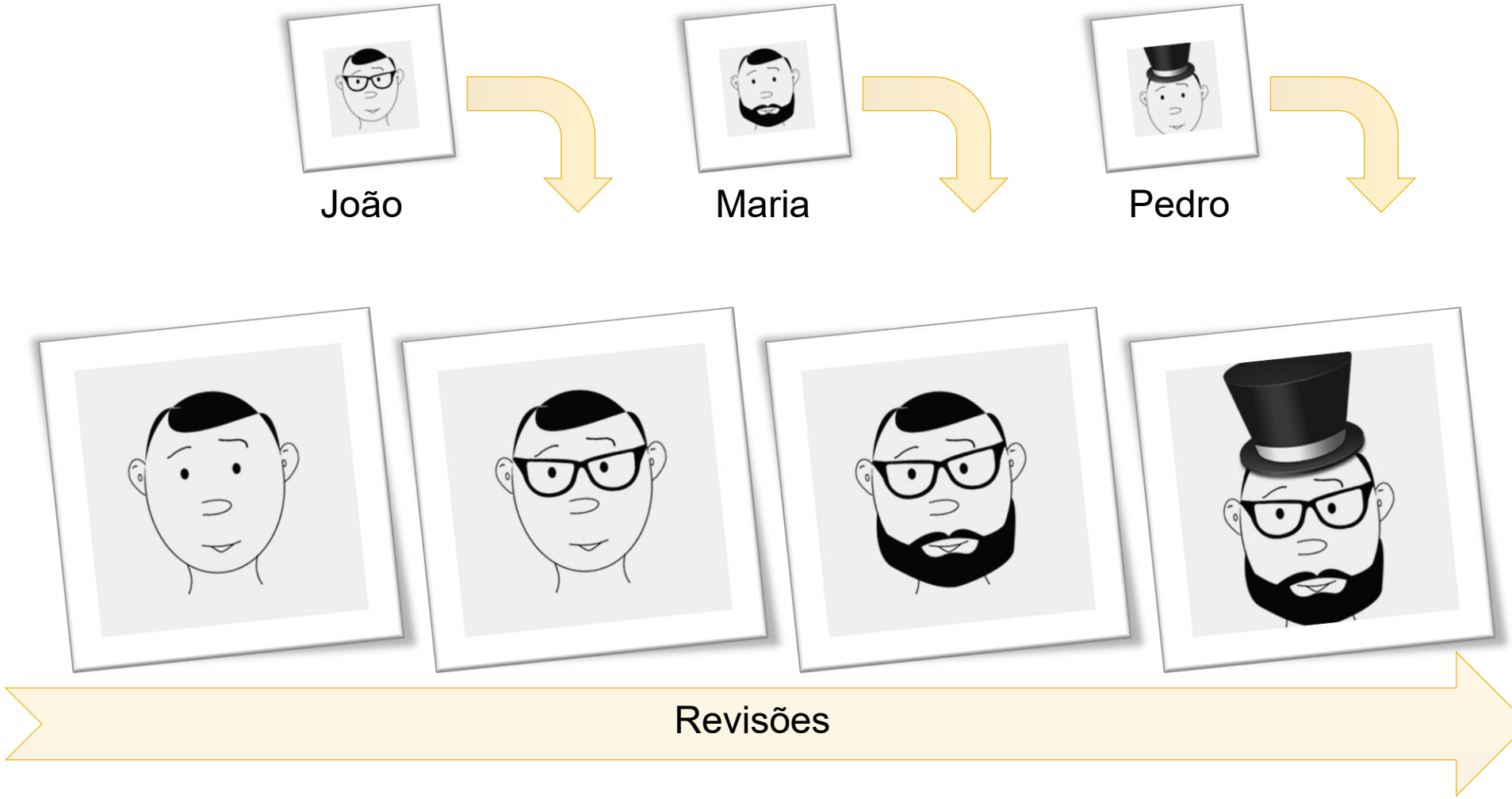
Versão base

Espaço de trabalho do Pedro

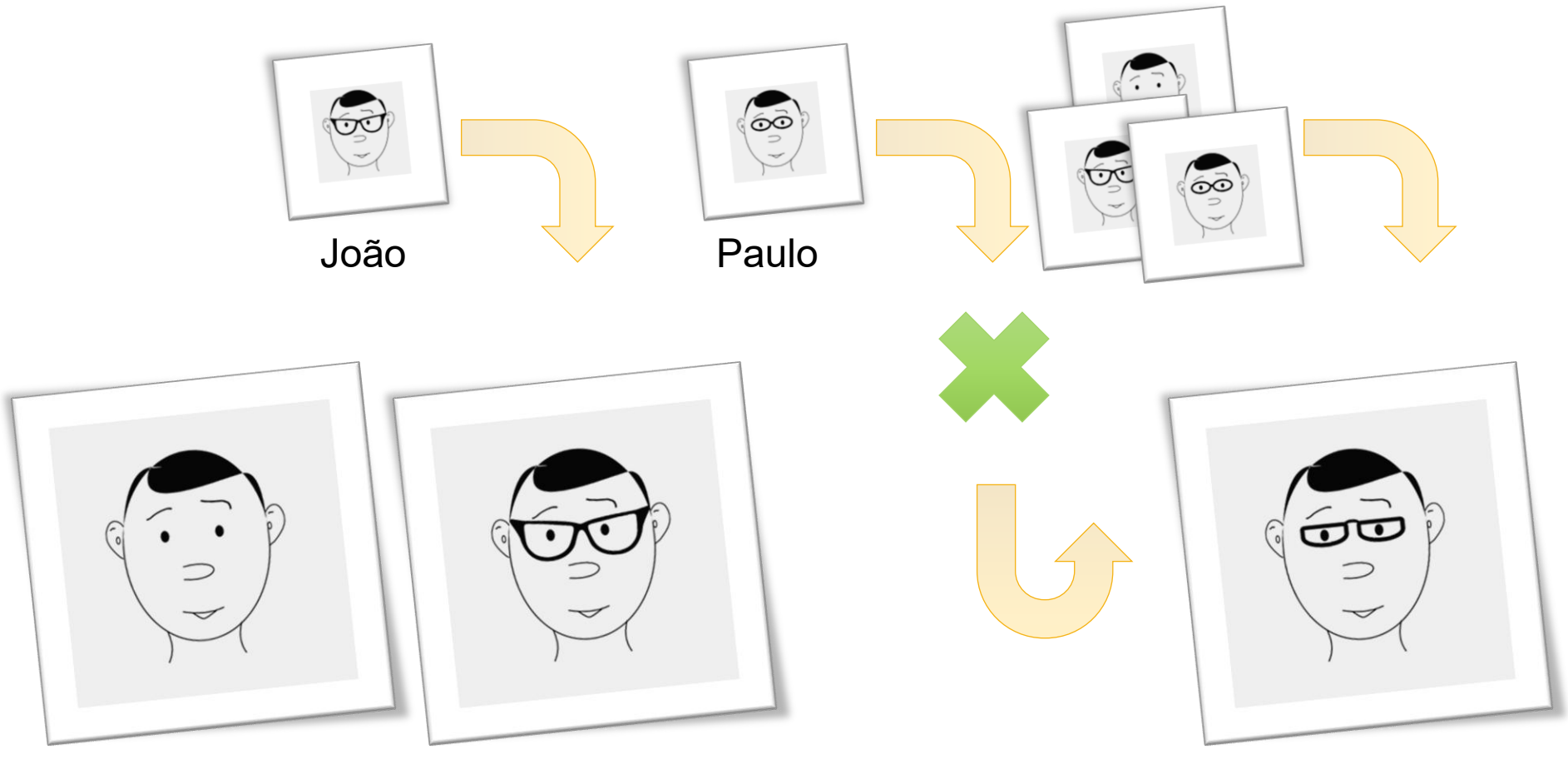
Espaço de trabalho da Maria

Espaço de trabalho do João

Versões de rascunho podem ser combinadas (operação de *merge*)

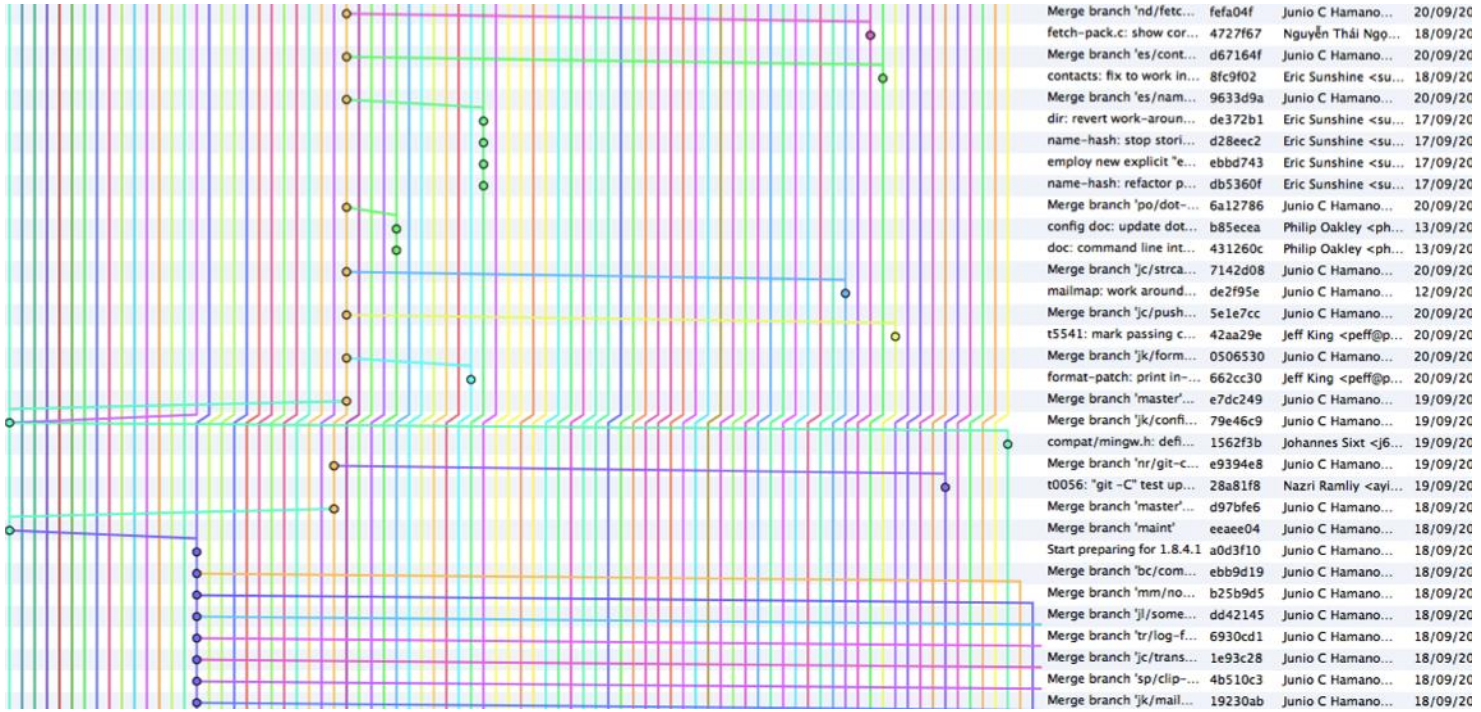


Conflitos podem ocorrer durante o *merge*



Versões no mundo real

- Infinidade de revisões e variantes juntas (sem contar versões rascunho)



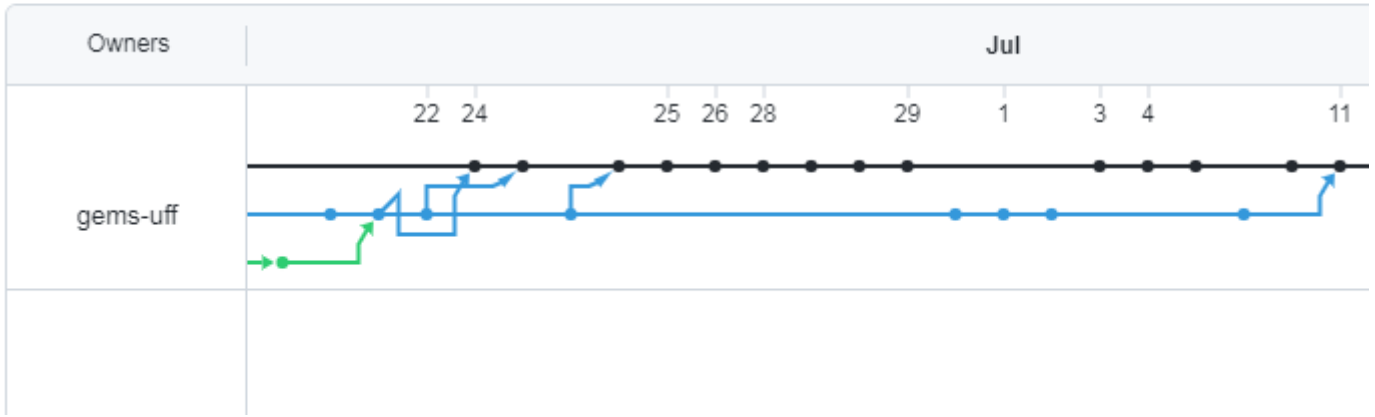
Mas afinal, para que servem versões?

- Sincronizar equipes
- Reproduzir configurações passadas
- Explorar possibilidades
- Isolar desenvolvedores
- Customizar produtos
- Rastrear a introdução de bugs
- Entender a evolução de software (MSR)
- Auditar mudanças
- Etc.

Ramos + Merges

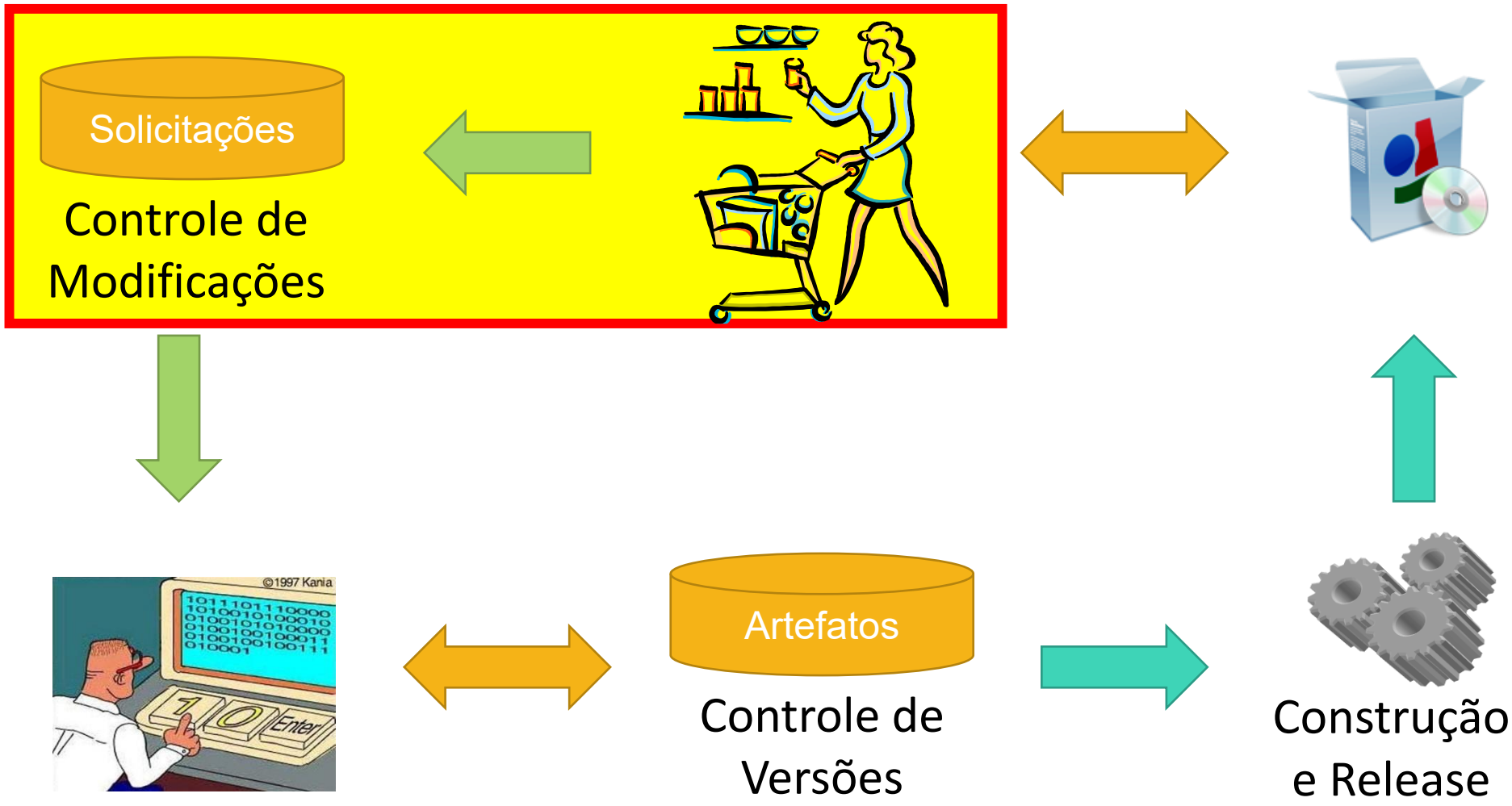
Network graph

Timeline of the most recent commits to this repository and its network ordered by most recently pushed to.



Principal sistema de controle de versão open-source

Sistema de Gerência de Configuração



Baseline

- Configuração revisada e aprovada que serve como base para uma próxima etapa de desenvolvimento e que somente pode ser modificada via processo formal de GCS
- São estabelecidas ao final de cada fase de desenvolvimento

Controle de modificações

- Tarefas
  - Solicitação de modificação
  - Classificação da modificação
  - Análise da modificação
  - Avaliação da modificação
  - Implementação da modificação
  - Verificação da modificação
  - Geração de baseline

Controle de modificações

Inability to restore some items from trash #2369

Closed **dstillman** opened this issue 15 days ago · 2 comments

dstillman commented 15 days ago • edited — Member

https://forums.zotero.org/discussion/94771/failed-to-restore-to-the-library

> [JavaScript Error: "Error: getNotes() cannot be called on items of type 'note'" {file: "chrome://zotero/content/xpcom/data/item.js" line: 2250}]

(Not sure if this error is from this. We can get a Debug ID if we're not sure.)

Regression from `48a3235` ?

dstillman added the Bug label 15 days ago

dstillman assigned **AbeJellinek** 15 days ago

Controle de modificações

✓ Fix error restoring top-level notes & attachments

Browse files

Fixes #2369

master

AbeJellinek committed 14 days ago — 1 parent 003b3db — commit b54466f089b8933025f2f0fbf6ea56f24eea602e

Showing 1 changed file with 3 additions and 2 deletions.

Split | Unified

5 chrome/content/zotero/zoteroPane.js

```
@@ -1946,14 +1946,16 @@ var ZoteroPane = new function()
1946 1946                               }
1947 1947
1948 1948                               let parent = this.itemsView.getRow(row).ref;
     1949 +                             let children = [];
     1950 +                             if (!parent.isNote()) children.push(...parent.getNotes(true));
     1951 +                             if (!parent.isAttachment()) children.push(...parent.getAttachments(true));
1949 1952
1950 1953                               if (isSelected(parent)) {
1951 1954                                       if (parent.deleted) {
1952 1955                                               parent.deleted = false;
1953 1956                                               await parent.save();
1954 1957                                       }
1955 1958
1956      -                                      let children = [...parent.getNotes(true), ...parent.getAttachments(true)];
1957 1959                                       let noneSelected = !children.some(isSelected);
1958 1960                                       for (let child of Zotero.Items.get(children)) {
1959 1961                                               if ((noneSelected || isSelected(child)) && child.deleted) {
@@ -1963,7 +1965,6 @@ var ZoteroPane = new function()
```

Exemplo de ferramentas de controle de modificações

- Livre
  - Github
  - Bugzilla
  - Mantis
  - Redmine
  - Trac
- Comercial
  - ClearQuest (IBM Rational)
  - JIRA (Atlassian)
  - StarTeam (Borland)
  - Synergy/Change (Telelogic)
  - TeamTrack (Serena)
  - Team Foundation Server (Microsoft)

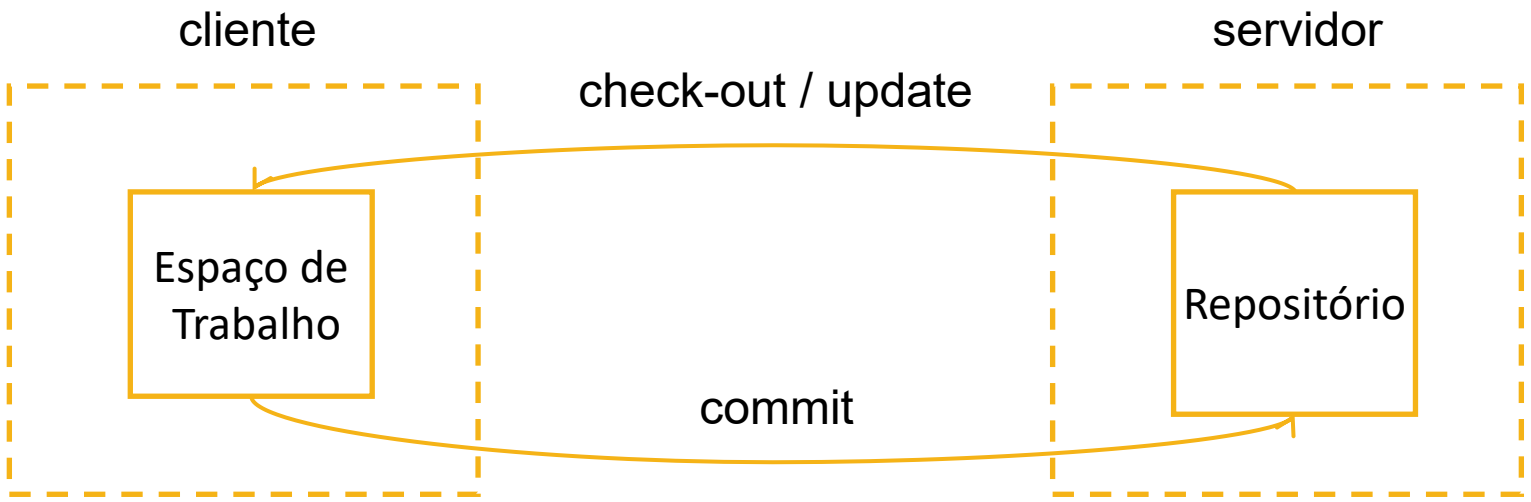
Agenda

01 GC

02 Git

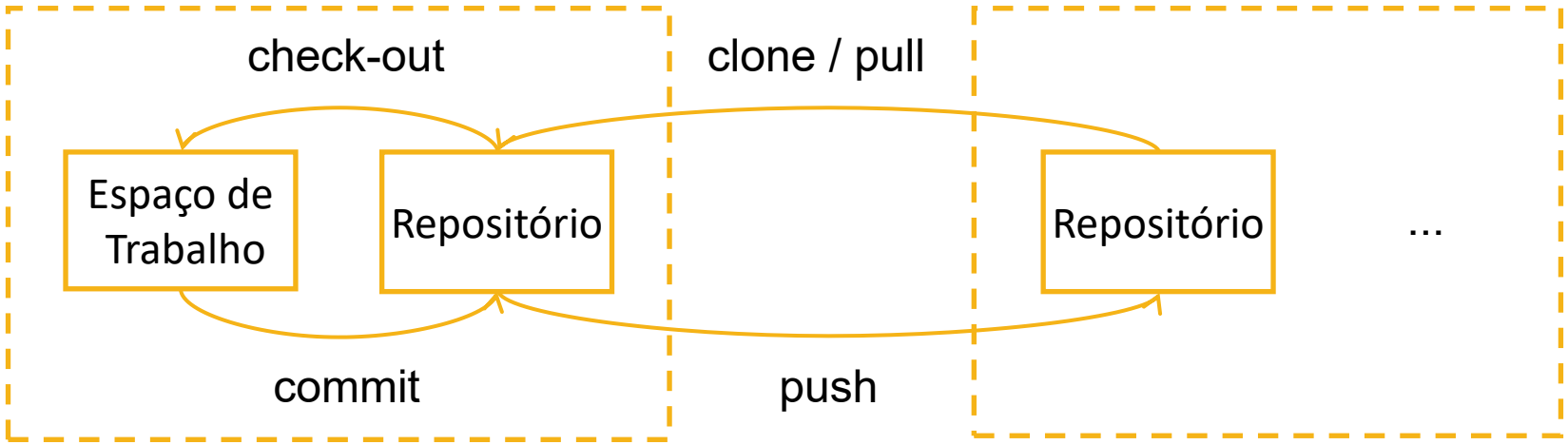
Centralizado

- Anos 80/90 – Sistemas cliente-servidor
  - CVS (1986)
  - Subversion (2000)



Distribuído

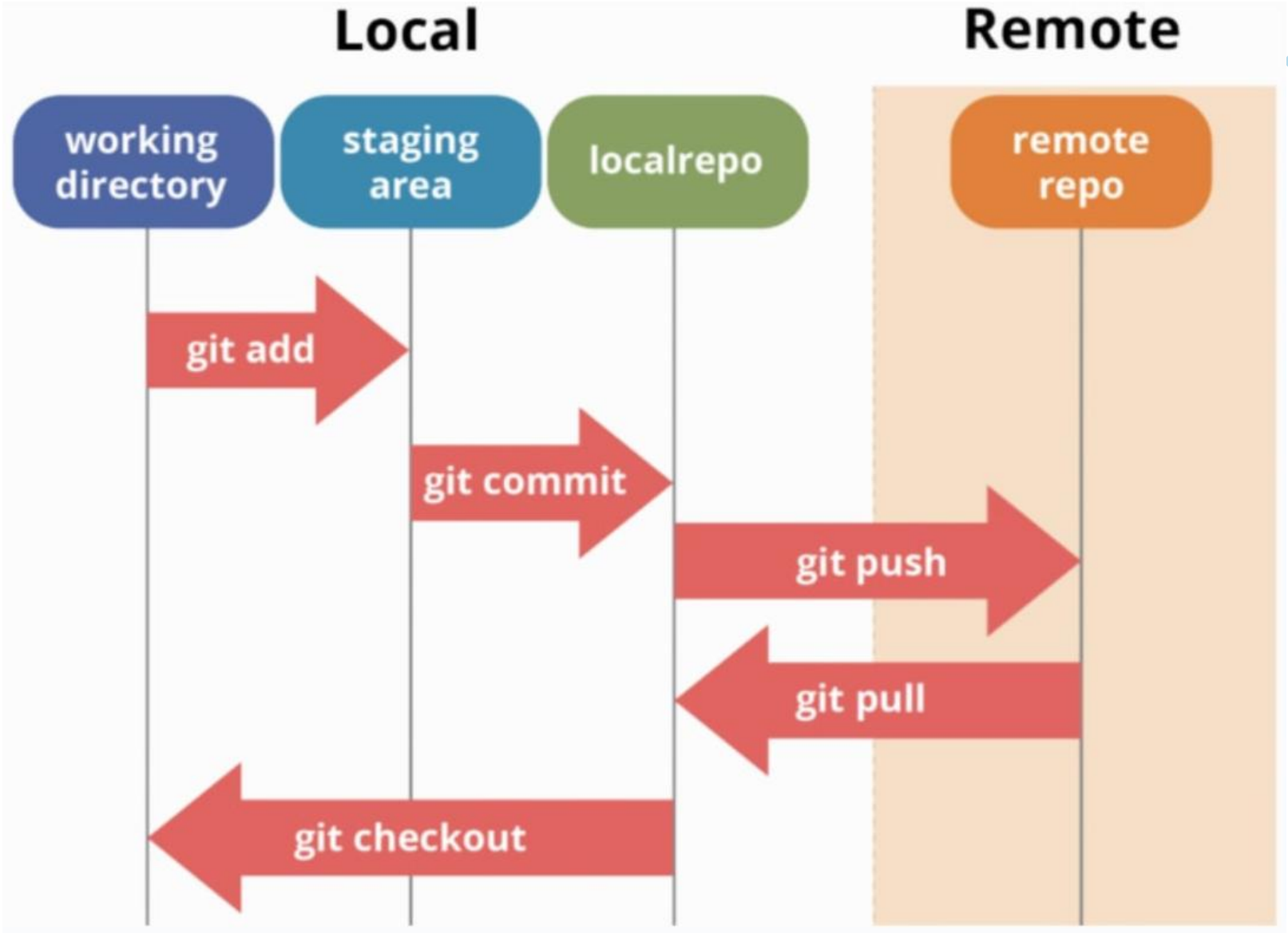
- Anos 2000 – Sistemas peer-to-peer
  - Git (2005)
  - Mercurial (2005)



Conceitos básicos: *staging area*

- Área onde são colocados os arquivos que pretendemos enviar para o repositório

*Área de preparação


Local
Remote
working directory
staging area
localrepo
remote repo
git add
git commit
git push
git pull
git checkout

Conceitos básicos: *commit* id

- Cada sistema de controle de versão usa uma estratégia diferente para identificar *commits*
  - Número sequencial por arquivo (CVS)
  - Número sequencial por repositório (Subversion)
  - Hash (Git e Mercurial)

Conceitos básicos: *apelidos*

- A versão base do seu espaço de trabalho
  - *HEAD*
- O ramo principal do seu repositório
  - *main*
- O repositório do qual seu repositório foi clonado
  - *origin*

Conceitos básicos: *help*!

- git help
  - Oferece ajuda geral sobre o git
- git help <comando>
  - Oferece ajuda sobre um comando específico do git
- Demais comandos dão dicas do que pode ser feito (leia com atenção as saídas dos comandos!)

Conceitos básicos: *quem sou eu?*

- git config --global user.name <seu nome>
  - Configura o nome do usuário
- git config --global user.email <seu email>
  - Configura o email do usuário

Repositório local

- git init <nome>
  - Cria um repositório Git no diretório
- git add --all
  - Adiciona um arquivo (ou todos) na *staging area* para ser enviado ao repositório no próximo *commit*
- git commit –m “mensagem”
  - Envia os arquivos que estão na *staging area* para o repositório

Repositório remoto

- git clone <url>
  - Cria um repositório local copiando o histórico de um repositório remoto
- git pull
  - Atualiza o repositório local e o espaço de trabalho em relação a um repositório remoto
- git push
  - Atualiza o repositório remoto em relação ao repositório local

Interface gráfica

- É possível fazer todos esses passos de forma visual
- Dentre várias ferramentas, temos...

Inspecionando mudanças

- git status
  - Inspeciona o espaço de trabalho
- git show
  - Inspeciona um *commit*
- git diff
  - Compara o espaço de trabalho com a staging area ou com alguma versão do repositório

Repositório local com ramos

- git branch --all -v
  - Lista os ramos existentes no repositório
- git branch <nome do ramo>
  - Cria um ramo à partir da versão indicada no HEAD
- git branch –d <nome do ramo>
  - Remove um ramo
- git checkout <commit id ou nome do ramo>
  - Troca a versão base do espaço de trabalho
- git merge <nome do ramo>
  - Combina um ramo com o ramo corrente

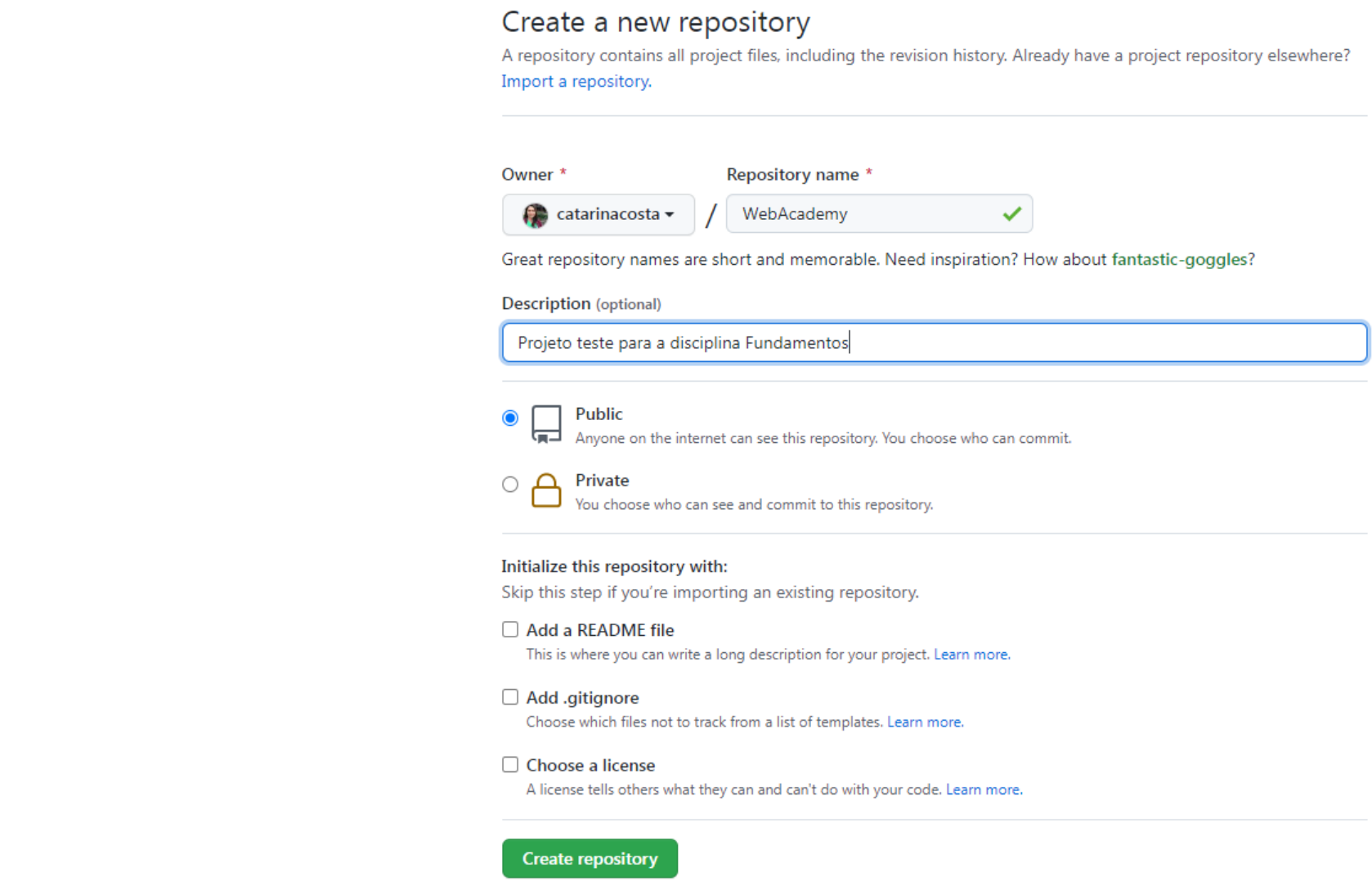
Múltiplos repositórios remotos

- git remote -v
  - Listar os repositórios remotos cadastrados
- git remote add <nome> <url>
  - Adiciona um novo repositório remoto
- git remote remove <nome>
  - Remove um repositório remoto existente

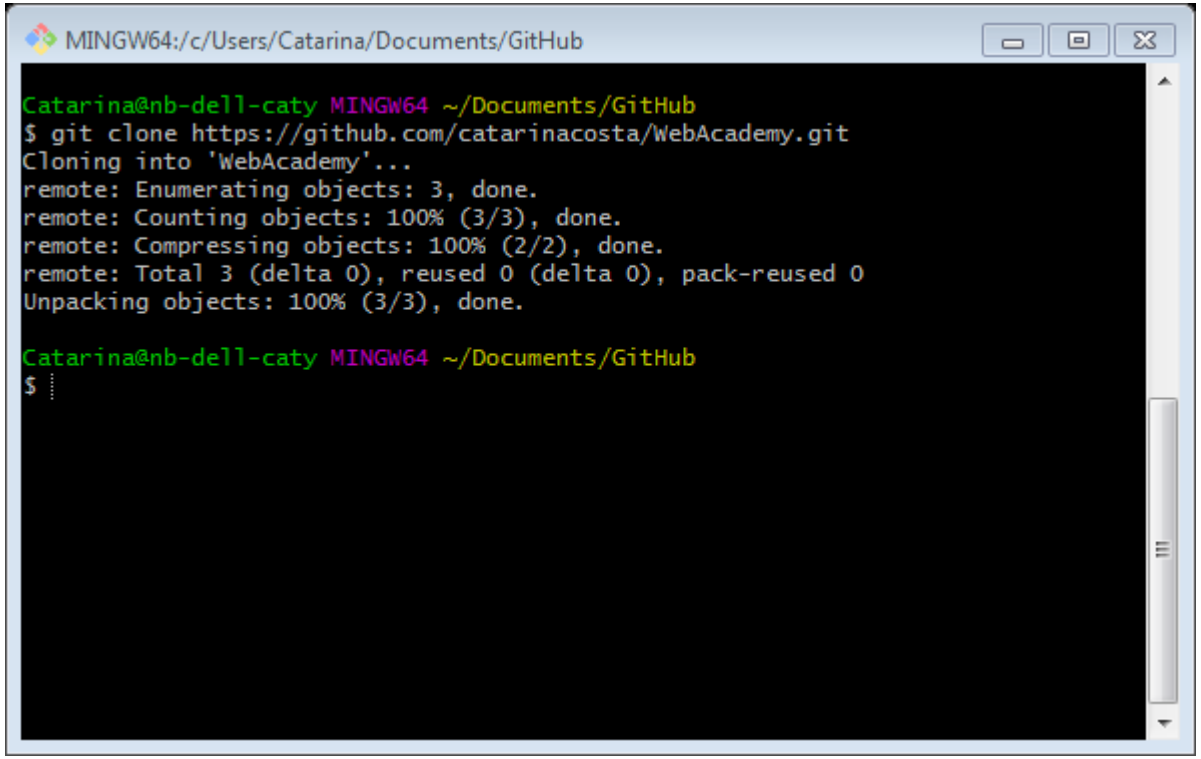
Serviço de hospedagem

- Há diversos serviços de hospedagem de repositórios Git na Internet
- Dentre eles, vamos praticar com...

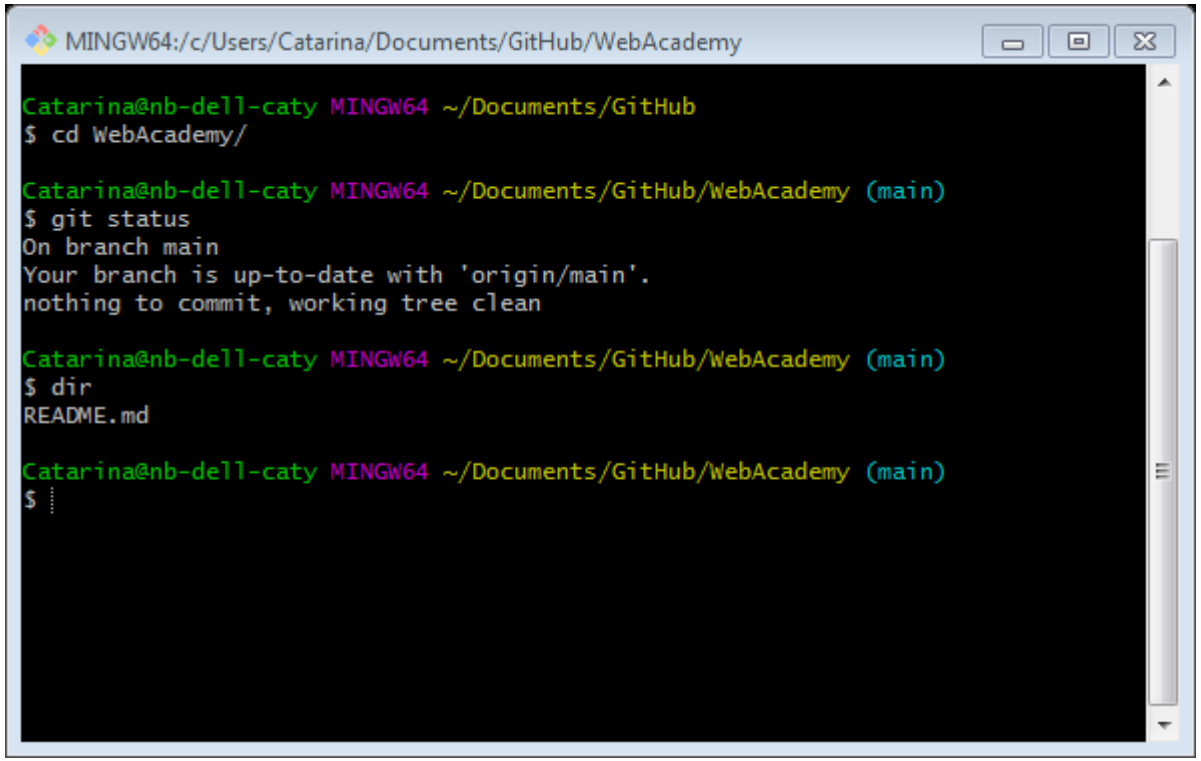
Bora criar nosso repositório



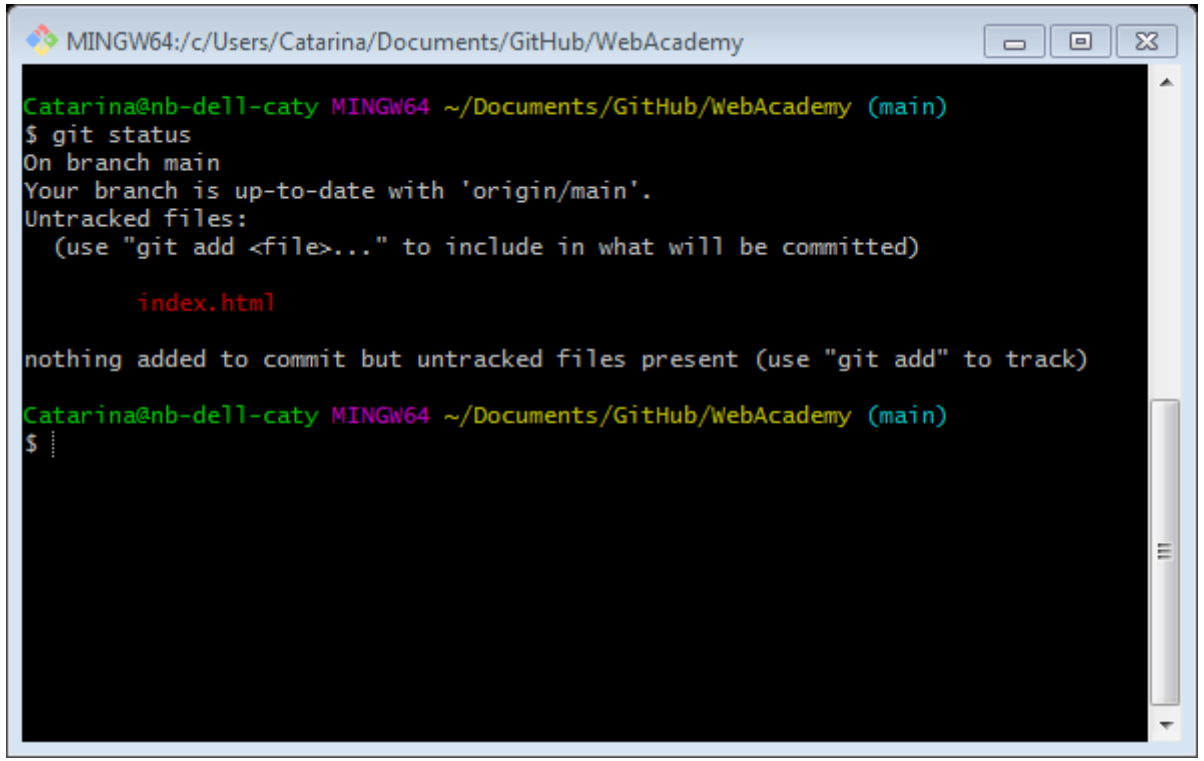
Bora criar nosso repositório



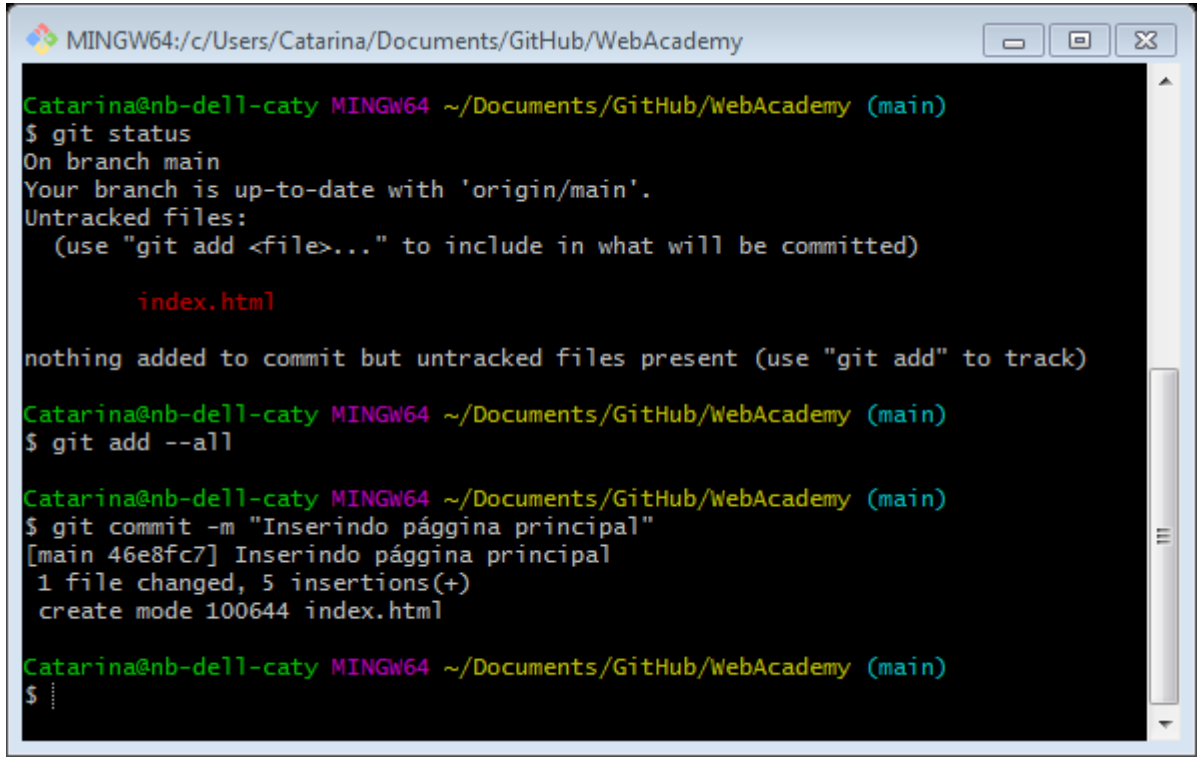
Verificando o que tem no repo



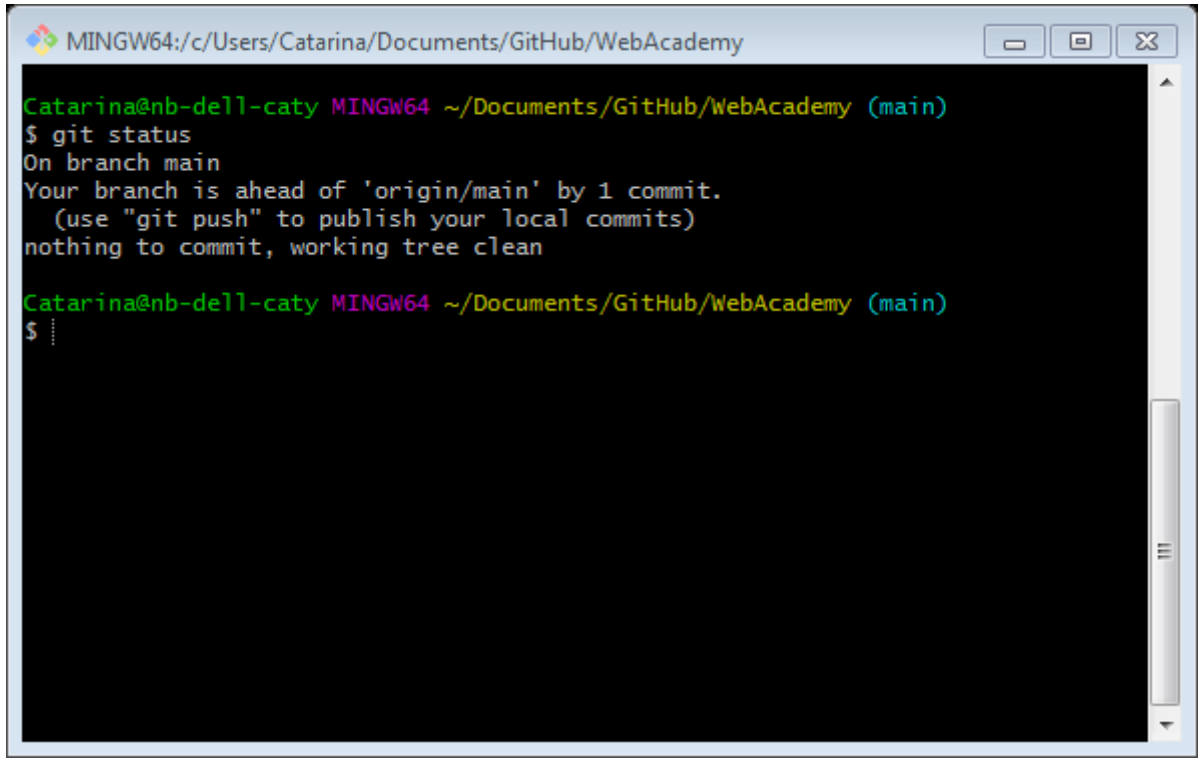
Verificando a alteração realizada



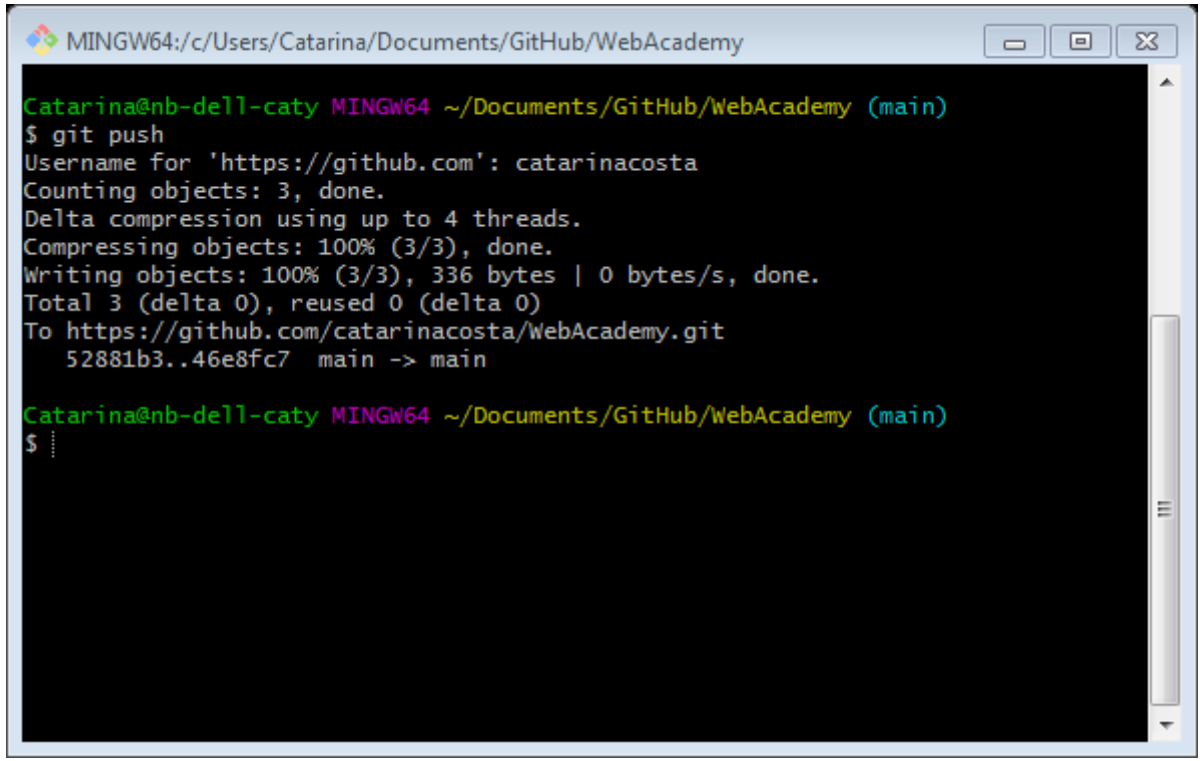
Adicionando e commitando



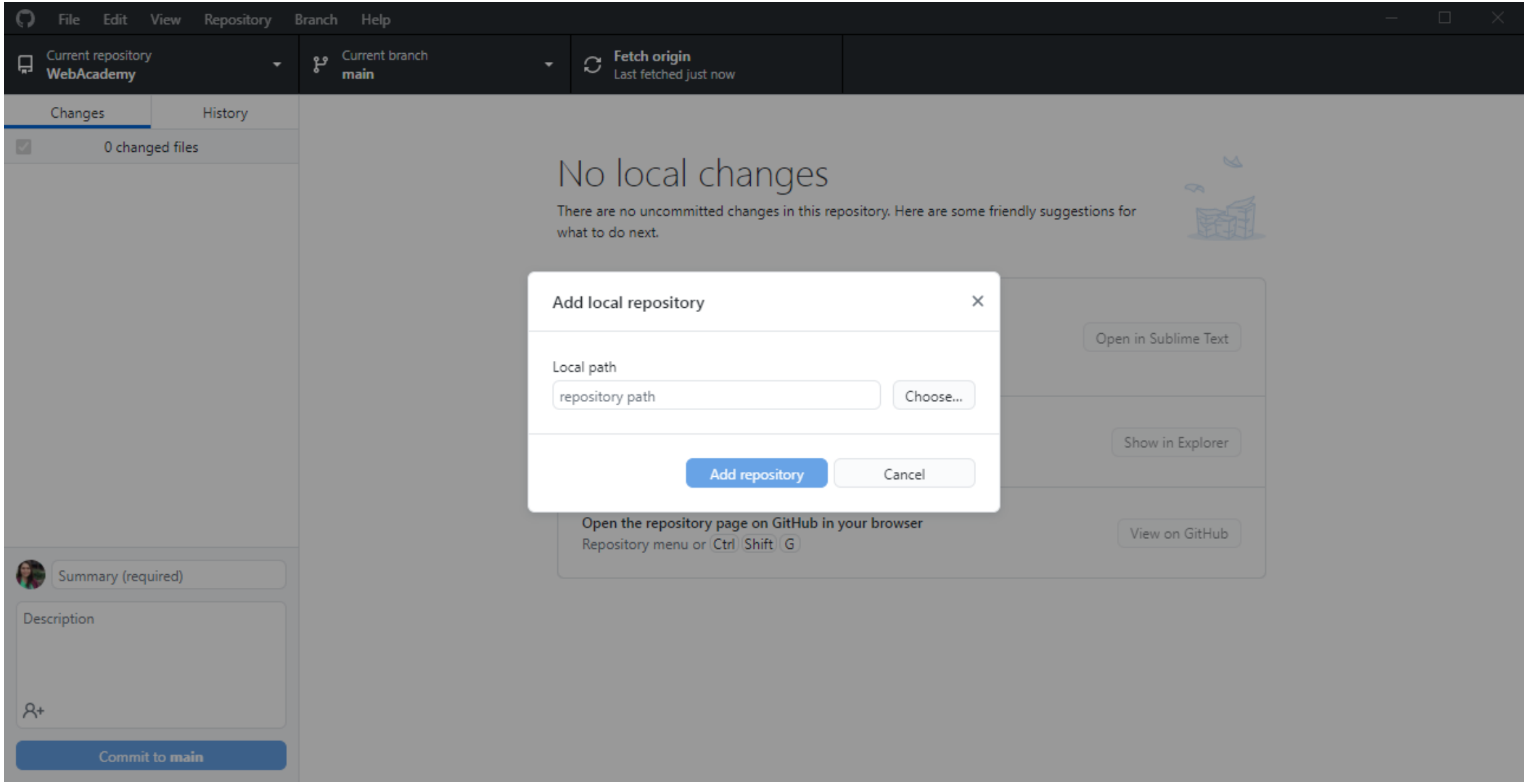
Verificando o status



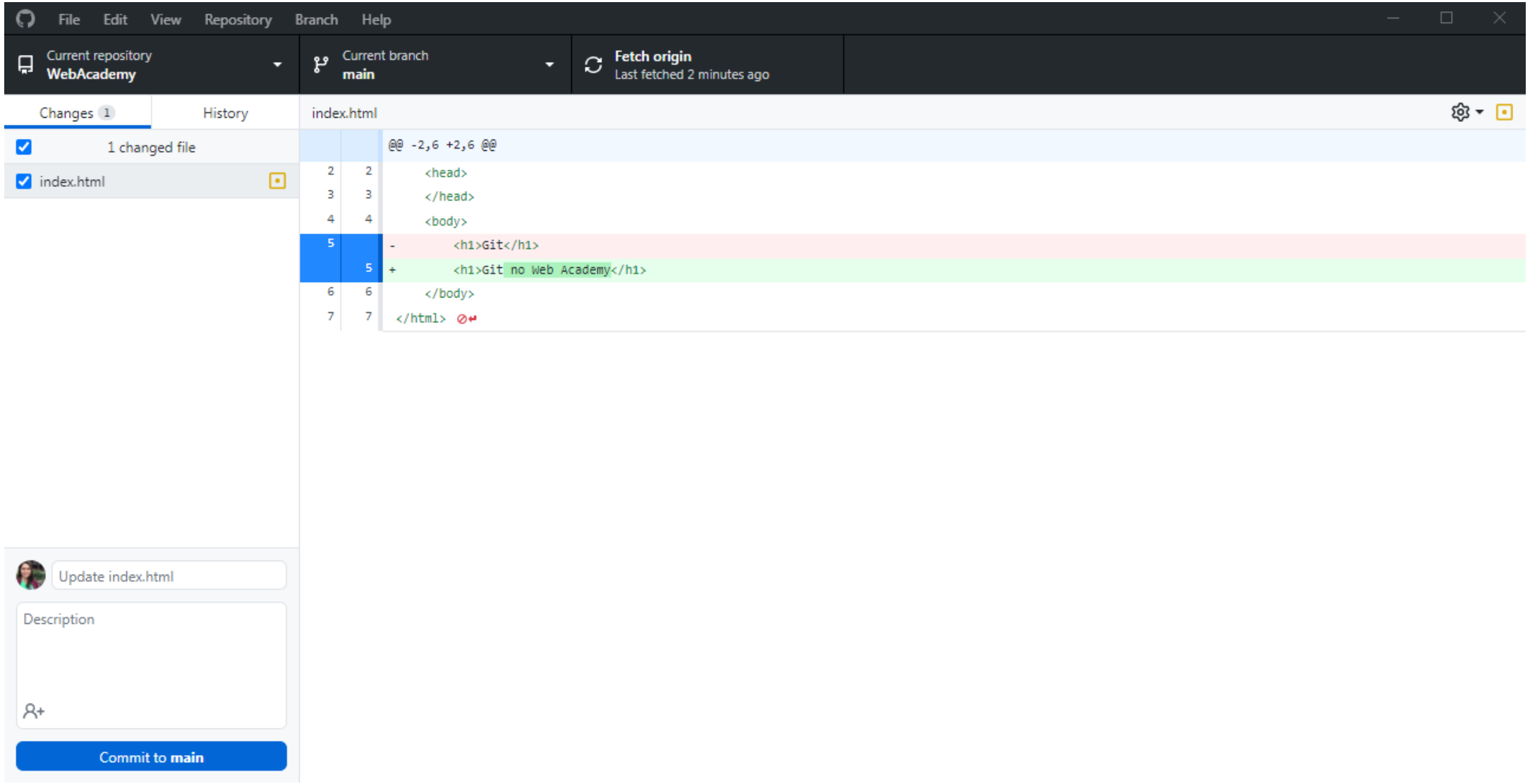
Bora criar nosso repositório



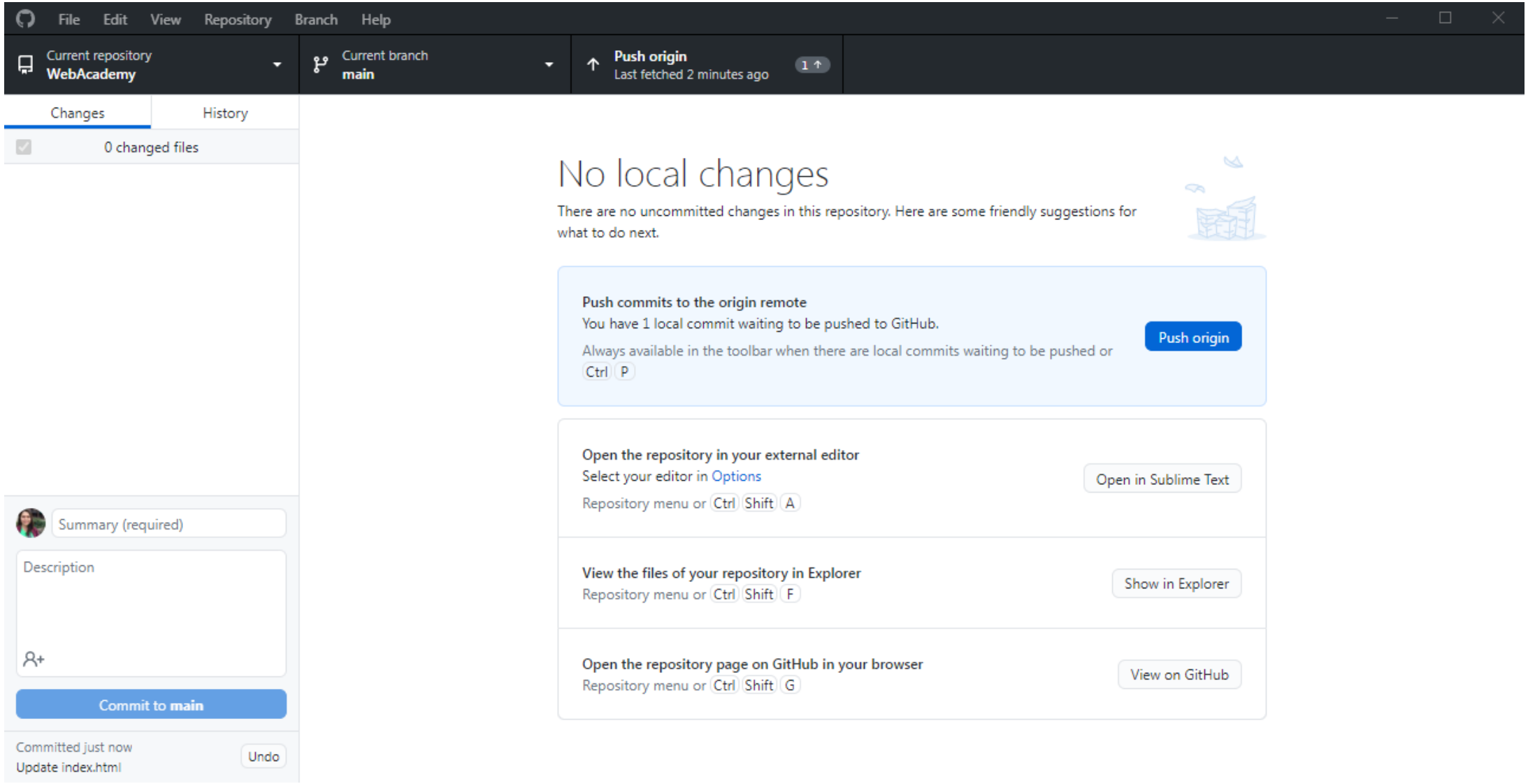
Abrir no Github Desktop



Alterar / Commit / Push



Alterar / Commit / Push



Criar o ramo e mudar para a nova ramificação

```
Catarina@nb-dell-caty MINGW64 ~/Documents/GitHub/WebAcademy2 (main)
$ git branch feature
```

```
Catarina@nb-dell-caty MINGW64 ~/Documents/GitHub/WebAcademy2 (main)
$ git checkout feature
Switched to branch 'feature'
```

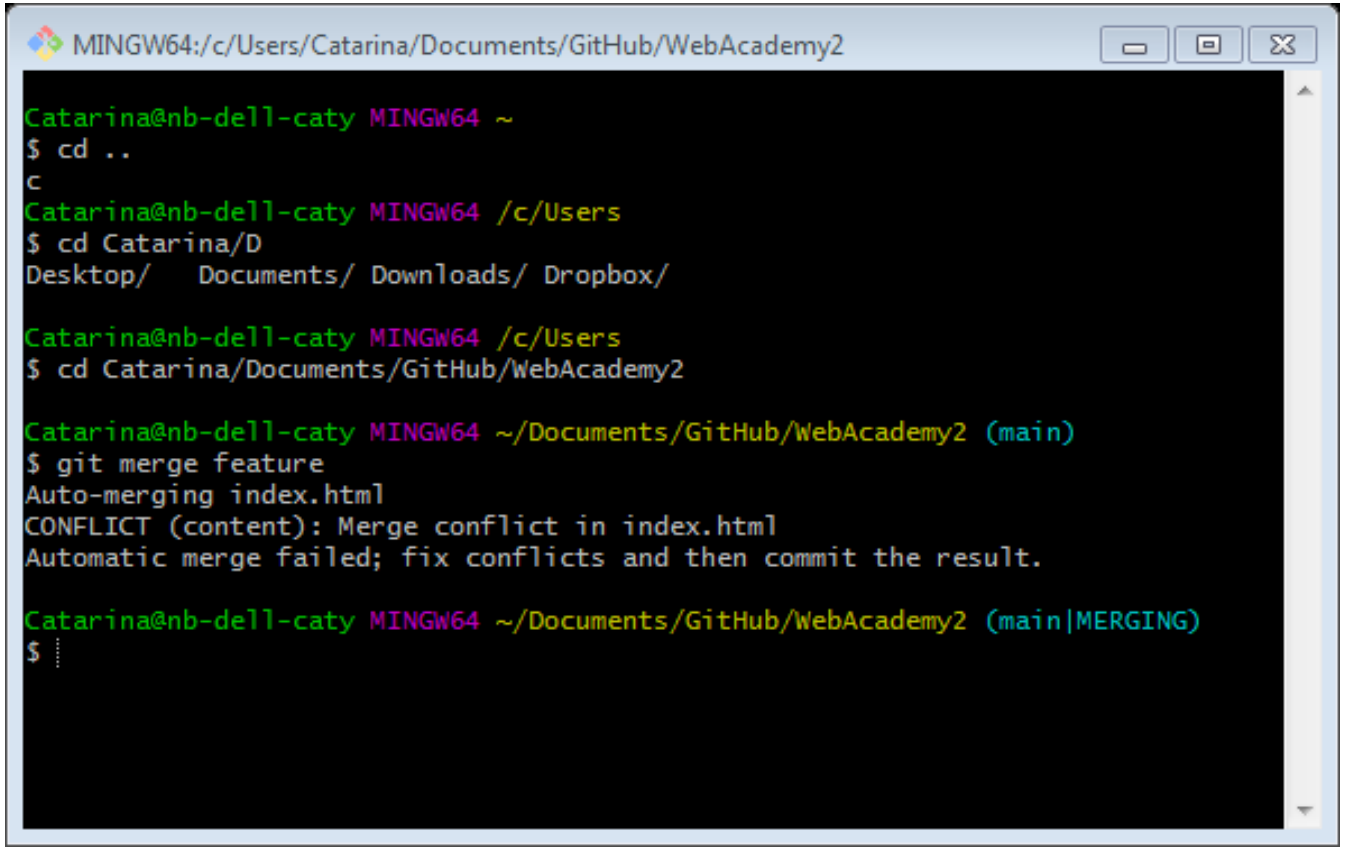
```
Catarina@nb-dell-caty MINGW64 ~/Documents/GitHub/WebAcademy2 (feature)
$ git status
On branch feature
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   index.html
```

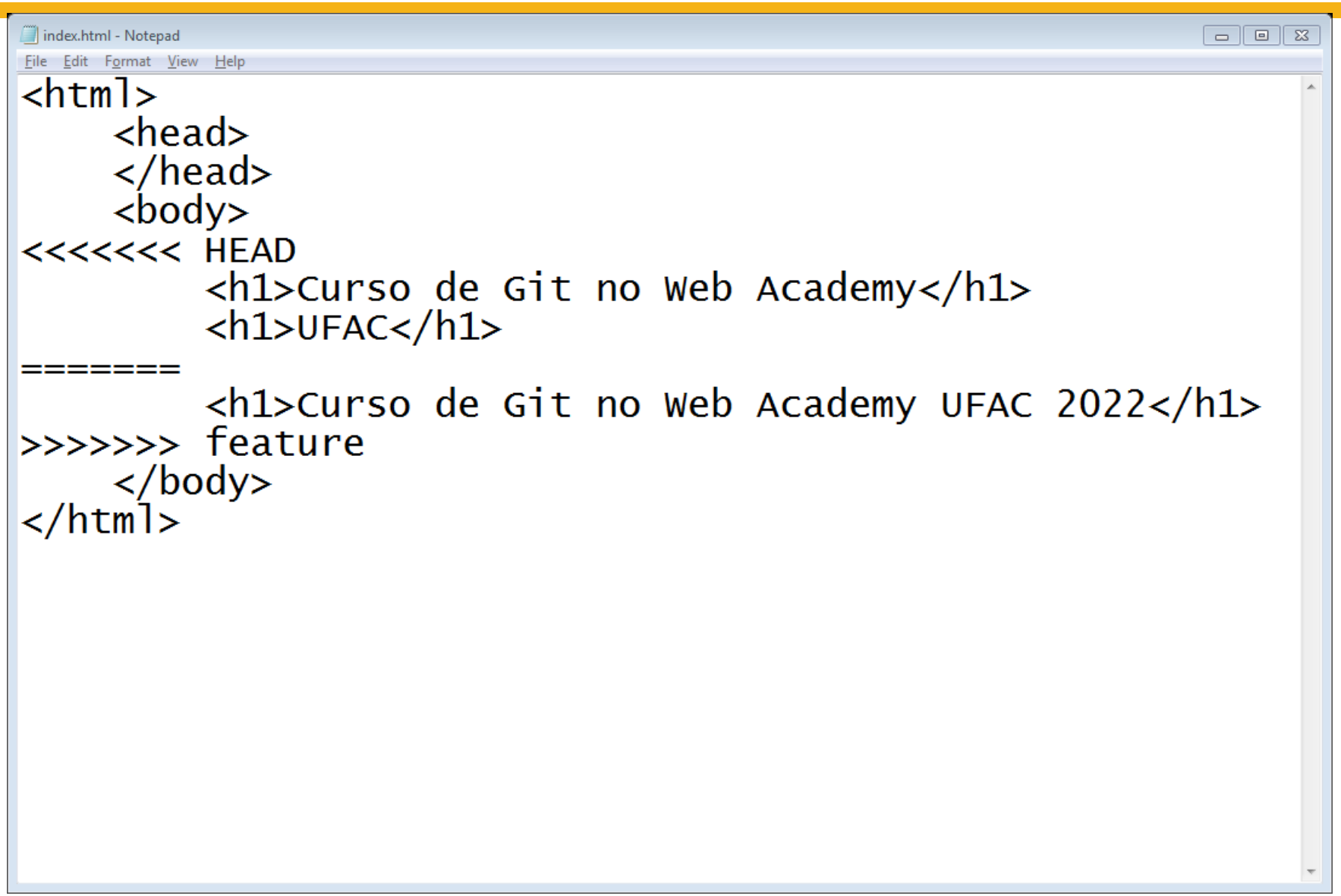
Para mudar de ramo – git checkout – voltando para o main

```
Catarina@nb-dell-caty MINGW64 ~/Documents/GitHub/WebAcademy2 (feature)
$ git checkout main
Switched to branch 'main'
Your branch is up-to-date with 'origin/main'.
```

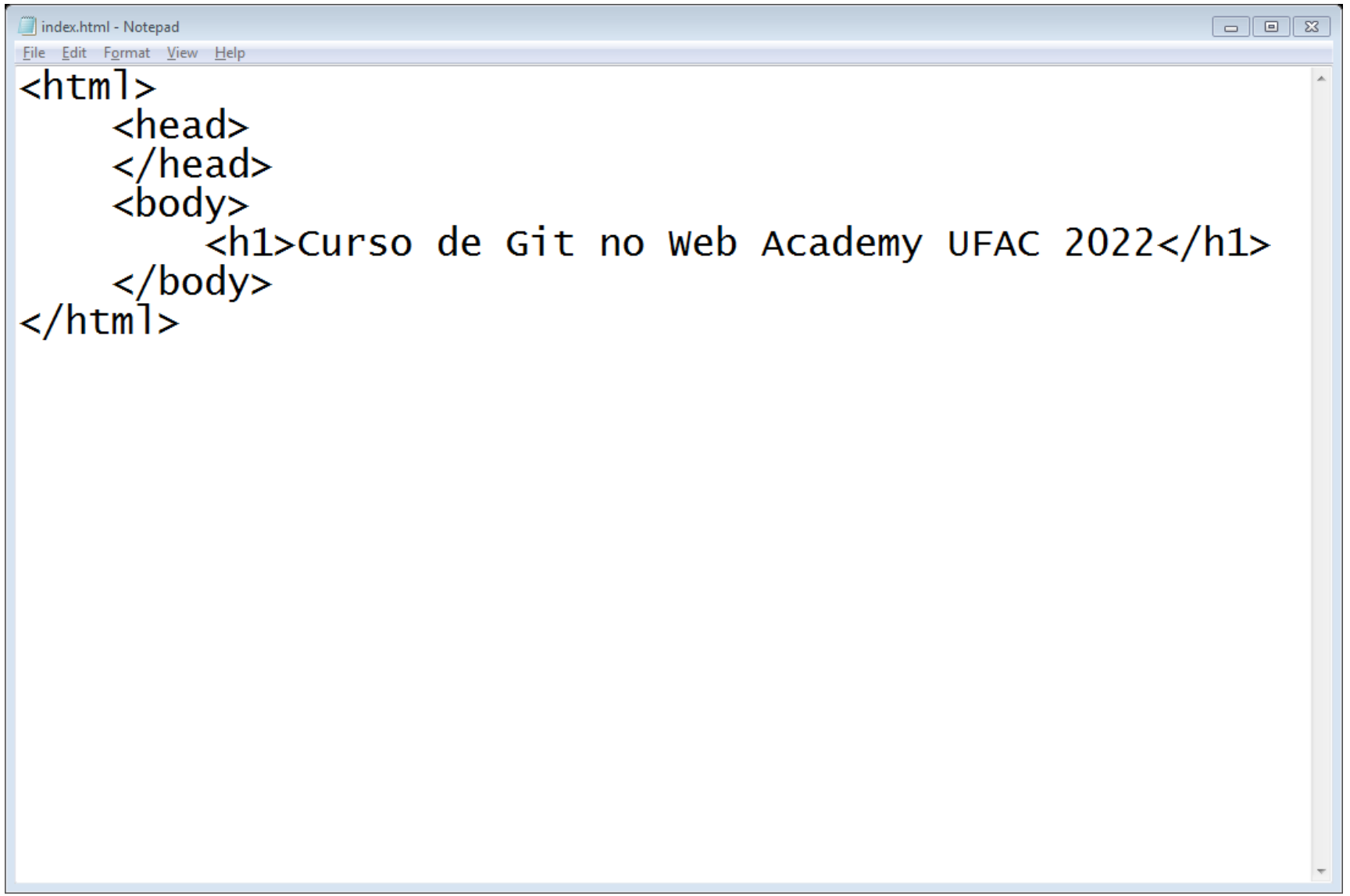
Tentando fazer o merge -> encontrando conflito



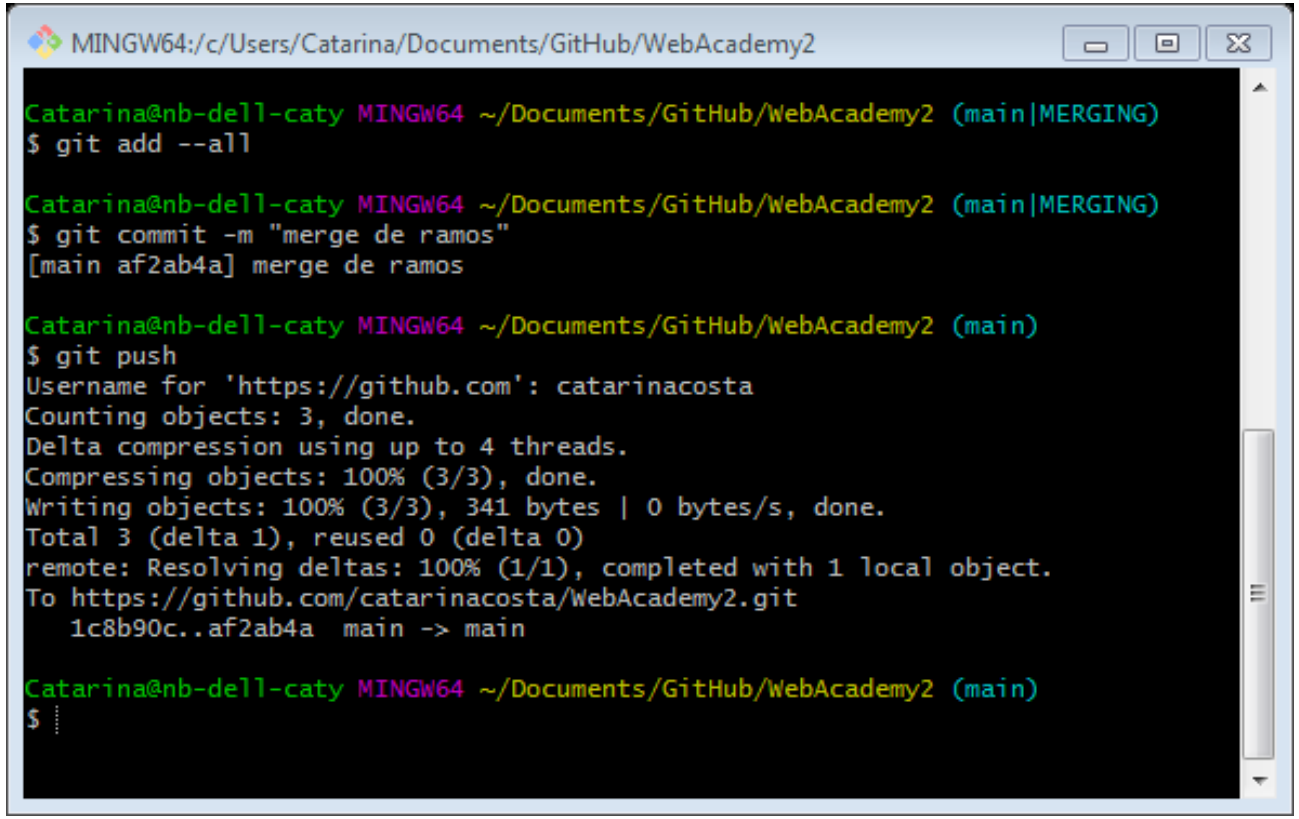
Resolvendo o conflito no editor

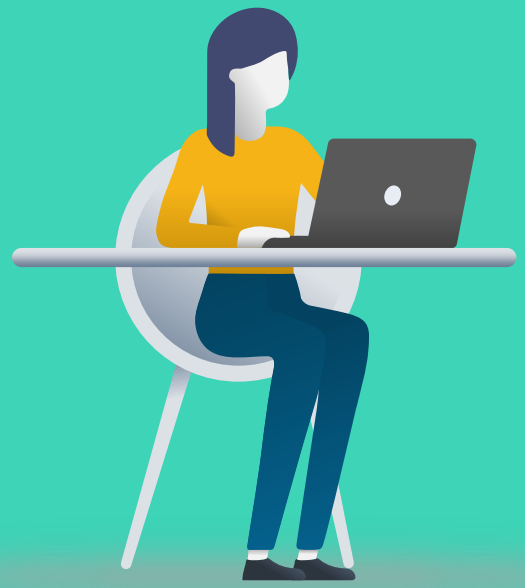


Resolvendo o conflito – o que foi escolhido pra ficar e ser commitado



Enviando a solução (fazendo o merge) para o repositório remoto





DICAS DE LEITURA +:

Leituras Git
https://engsoftmoderna.info/capAp.html
https://git-scm.com/book/en/v2

ATIVIDADE +:

Instalar o Git
Instalar Github desktop
Git bash (ou outro)