



Laboratório de
Tecnologia da Informação
Aplicada

Controle de Versão usando Git

Marcelo Augusto Cordeiro

marcelo.augusto.cordeiro@gmail.com



Laboratório de
Tecnologia da Informação
Aplicada

CENTRO DE
PESQUISA
CATI
SEPIN/MCT



Agenda

- O que é e por que usar Controle de Versão
- Por que usar o Git
- Configurações
- Comandos Básicos
- Branching
- Remote Branches
- Desfazendo Mudanças
- Reescrevendo a História
- Projetos OpenSource (GitHub)



O que é Controle de Versão

O controle de versão é um sistema que registra as mudanças feitas em um arquivo ou em um conjunto de arquivos ao longo do tempo de forma que você possa recuperar versões específicas.

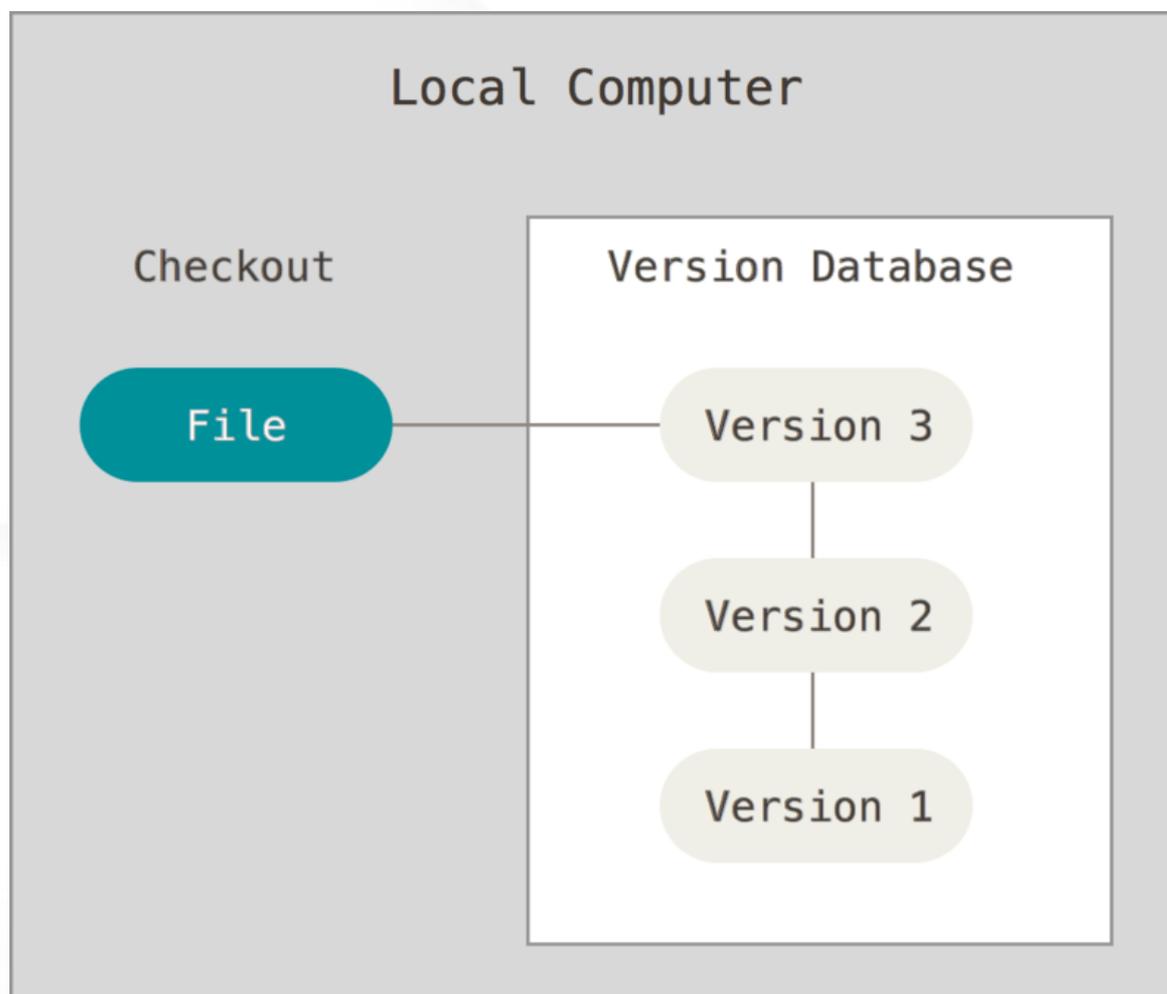
Pro Git, Scott Chacon e Ben Straub



Por que usar Controle de Versão

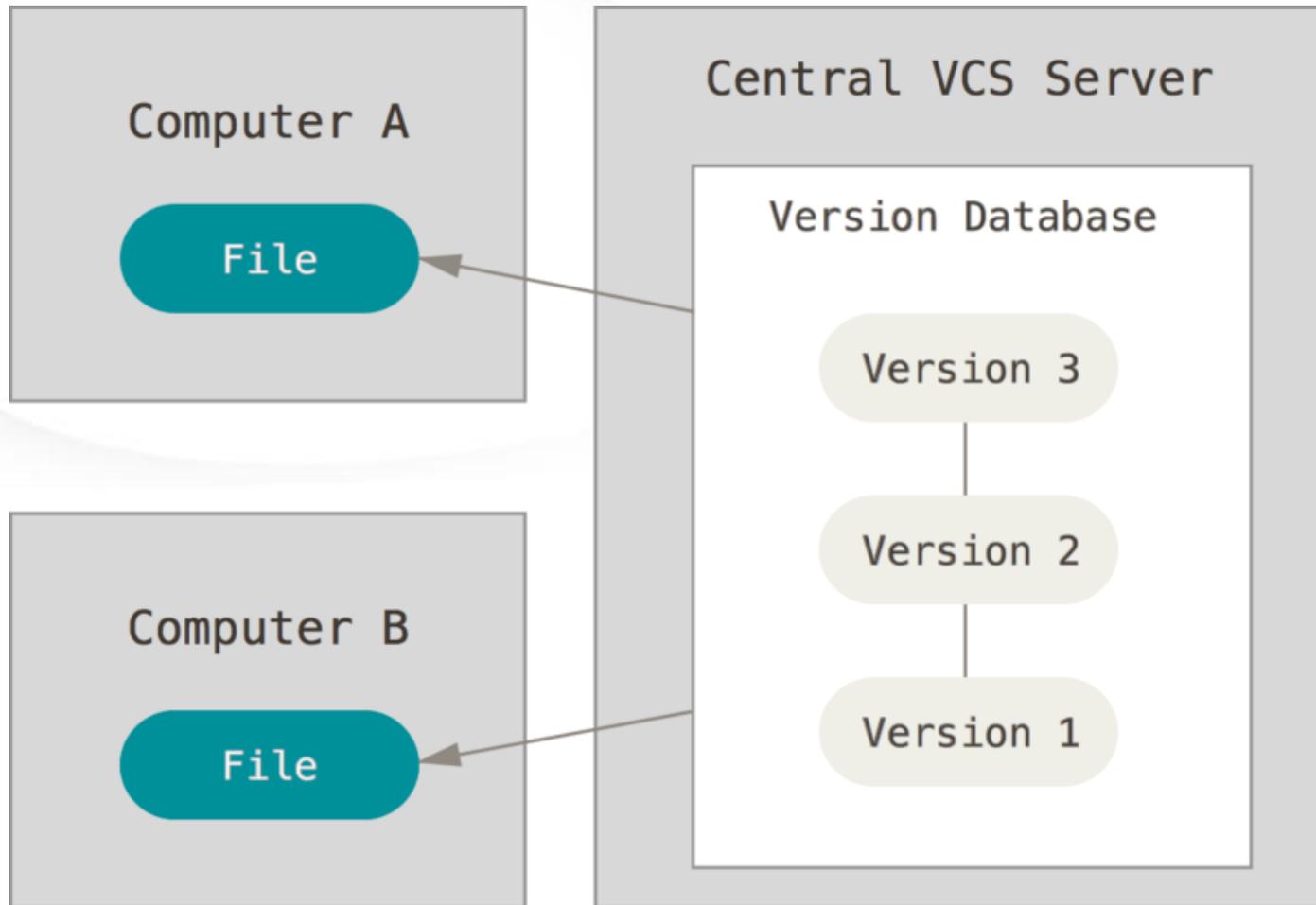
- Compartilhamento e versionamento de qualquer arquivo;
- Trabalho em paralelo com fácil controle das modificações;
- Possibilidade de trabalhar em “equipes” enormes.

Controle de Versão Local

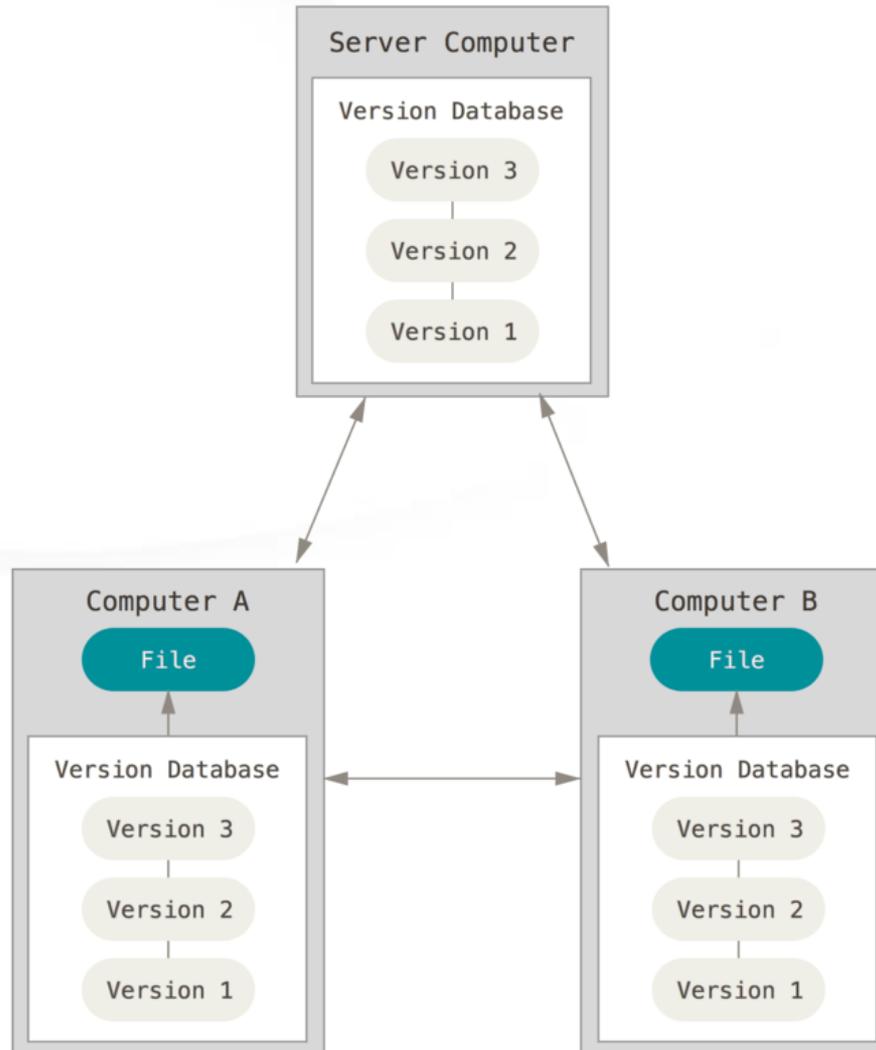




Controle de Versão Centralizado



Controle de Versão Distribuído





Por que usar o Git?

- Implementação de Branches extremamente leve;
- Comunidade OpenSource do GitHub;
- “Portfólio” para computação.



Configurando o Git

- `git config --global user.name "Nome Sobrenome"`
- `git config --global user.email email@gmail.com`
- `git config --global core.editor "notepad"`

- `git config --list`

- `git help comando`



Criando um Repositório

- `git init`
- `git clone https://github.com/user/Project.git`



Comandos Básicos

Crie um diretório git

Working Dir.

Staging Area

.git Dir.



Comandos Básicos

Crie um arquivo qualquer .txt

Working Dir.



File.txt

Staging Area

.git Dir.



Comandos Básicos

```
git add File.txt
```

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



Comandos Básicos

```
git status
```

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



Comandos Básicos

```
git commit -m "Adicionei File.txt"
```

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



File.txt



Comandos Básicos

Edite o arquivo File.txt

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



File.txt



Comandos Básicos

git status

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



File.txt



Comandos Básicos

```
git add *
```

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



File.txt



Comandos Básicos

Edite novamente File.txt

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



File.txt



Comandos Básicos

```
git add *.txt
```

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



File.txt



Comandos Básicos

```
git commit -m "Altereí File.txt"
```

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



File.txt



Comandos Básicos

git log

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



File.txt



Comandos Básicos

git status

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



File.txt



Comandos Básicos

Delete o arquivo File.txt

Working Dir.

Staging Area

.git Dir.



File.txt



File.txt



Comandos Básicos

git status

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



File.txt



Comandos Básicos

```
git add File.txt
```

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



File.txt



Comandos Básicos

```
git commit -m "Deletei File.txt"
```

Working Dir.



File.txt

Staging Area



File.txt

.git Dir.



File.txt



Comandos Básicos

```
git rm File.txt
```

Working Dir.

Staging Area

.git Dir.



Git Ignore

- Arquivo .gitignore
- <https://www.gitignore.io/>
- <https://github.com/github/gitignore>

```
1 # ===== #
2 # Unity generated #
3 # ===== #
4 Temp/
5 Obj/
6 UnityGenerated/
7 /Library/
8
9 # ===== #
10 # Visual Studio / MonoDevelop generated #
11 # ===== #
12 ExportedObj/
13 *.svd
14 *.userprefs
15 *.csproj
16 *.pidb
17 *.suo
18 *.sln
19 *.user
20 *.unityproj
21 *.booproj
22
23 # ===== #
24 # OS generated #
25 # ===== #
26 .DS_Store
27 .DS_Store?
28 .*
29 .Spotlight-V100
30 .Trashes
31 Icon?
32 ehthumbs.db
33 Thumbs.db
```



Outros Comandos

- `git commit -a -m "Também adiciona as mudanças"`
- `git commit --amend -m "Altera o último commit"`
- `git diff`



Aliases

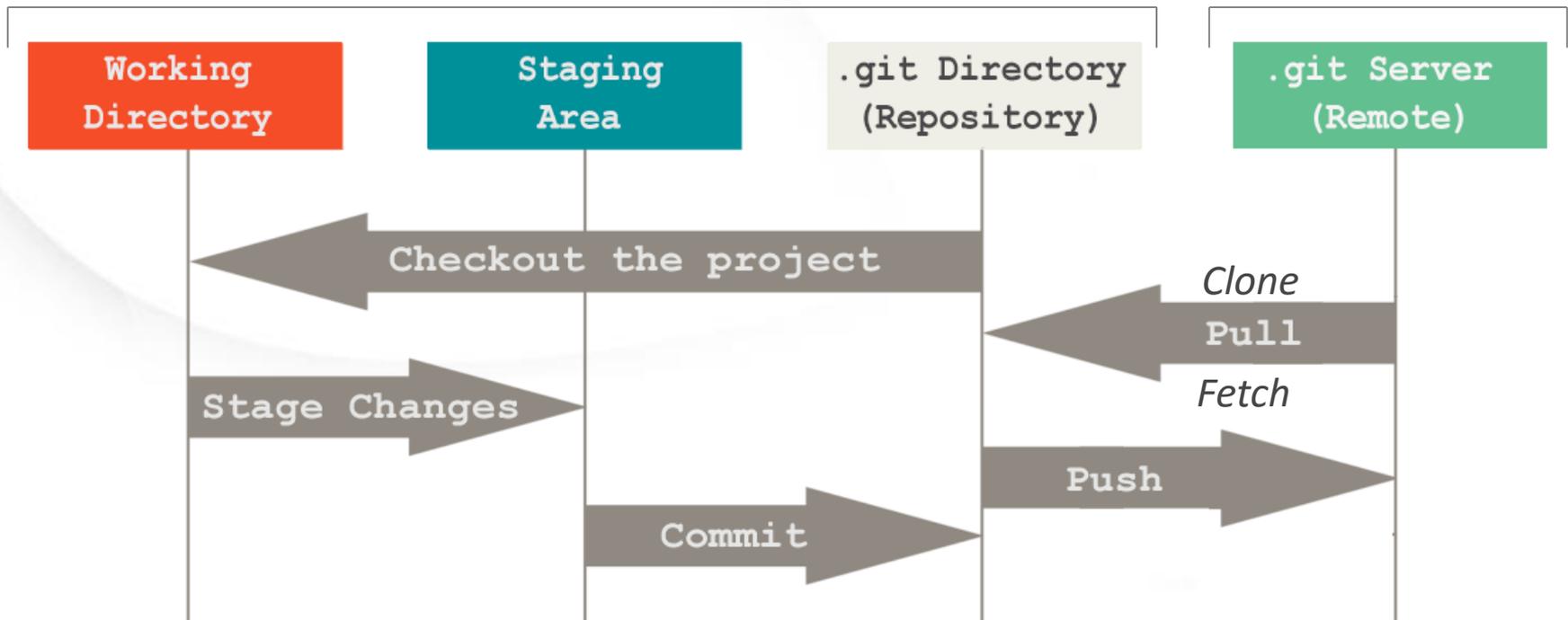
- `git config --global alias.cm commit`
- `git config --global alias.unstage 'reset HEAD --'`
- `git config --global alias.last 'log -1 HEAD'`



Workflow

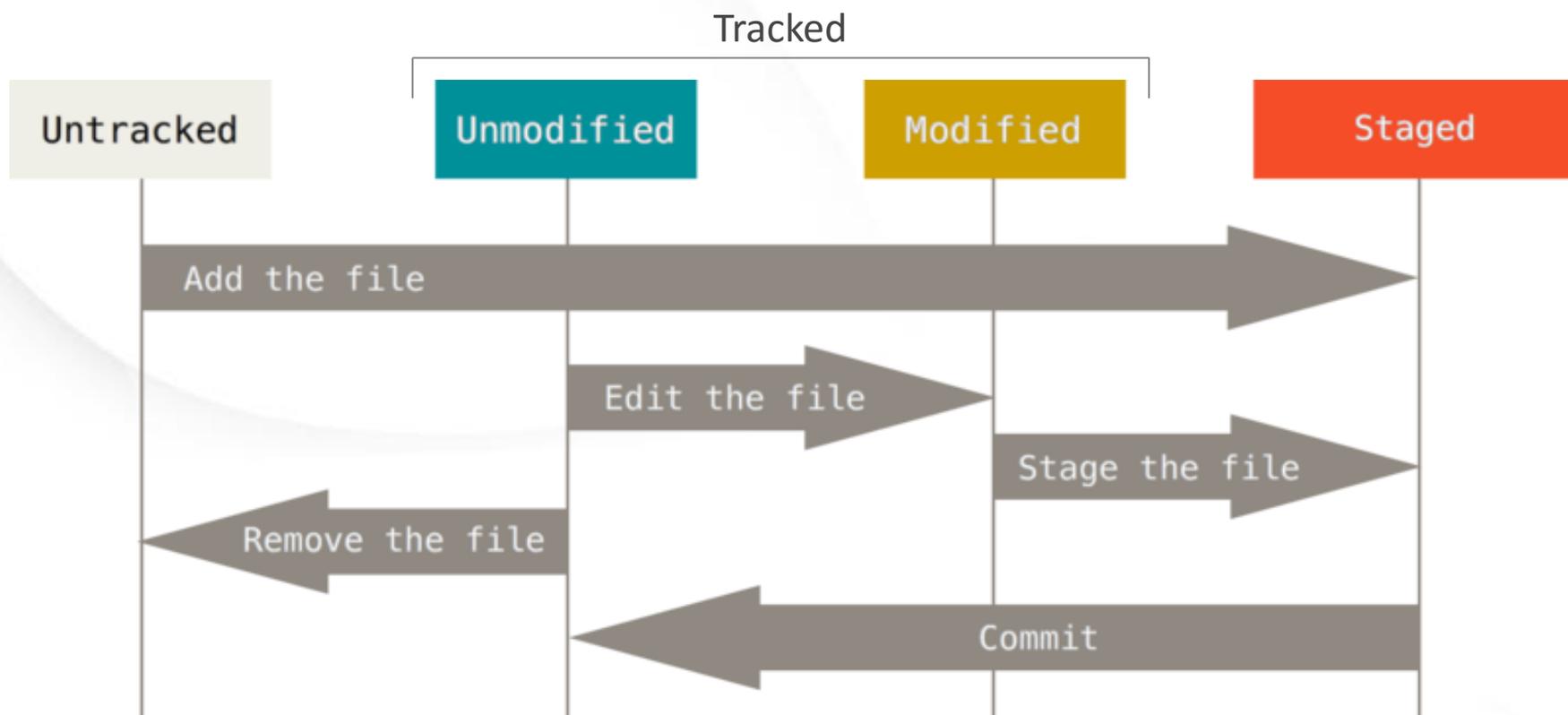
Local

Remote





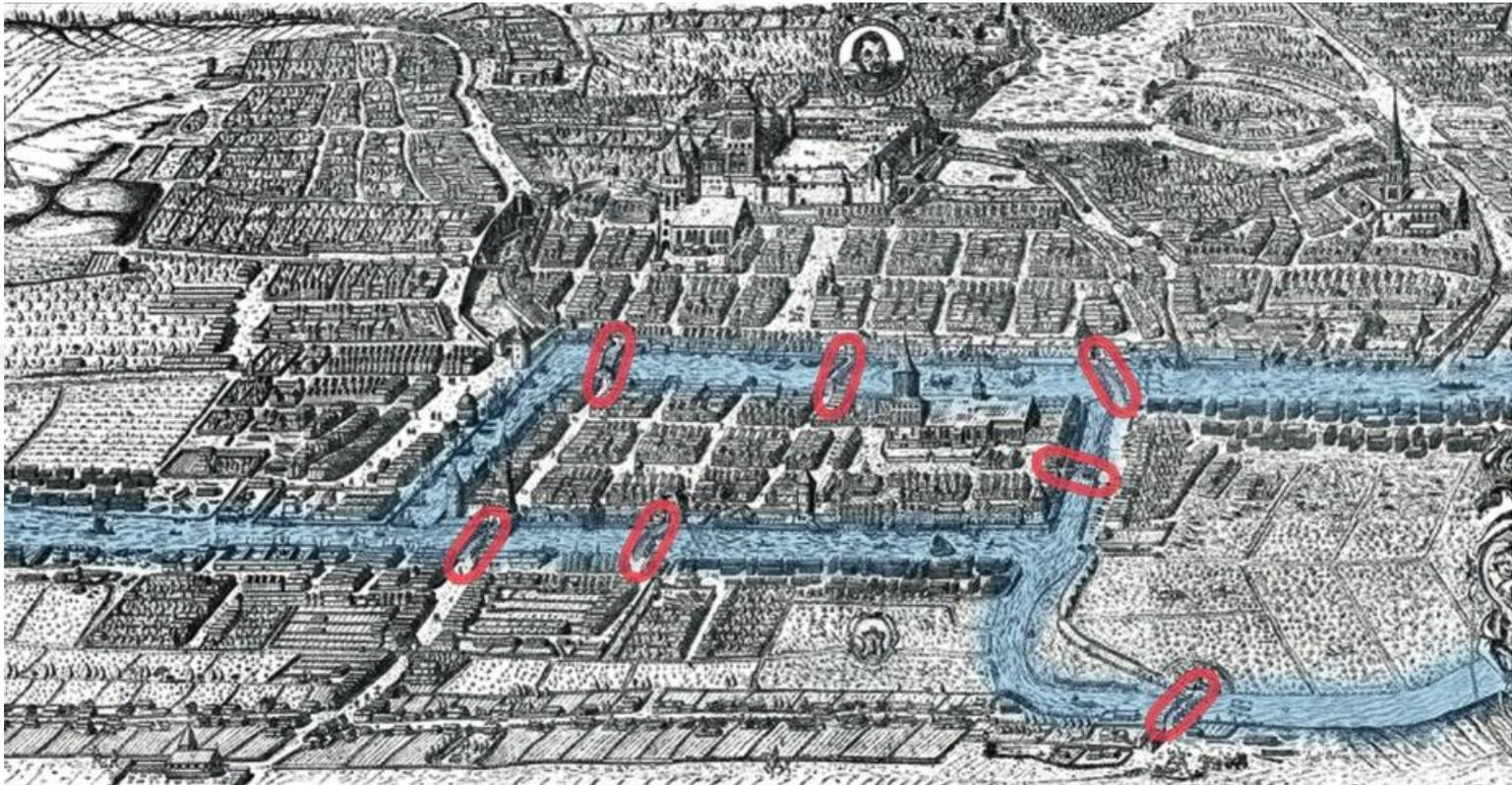
Ciclo de vida de um arquivo

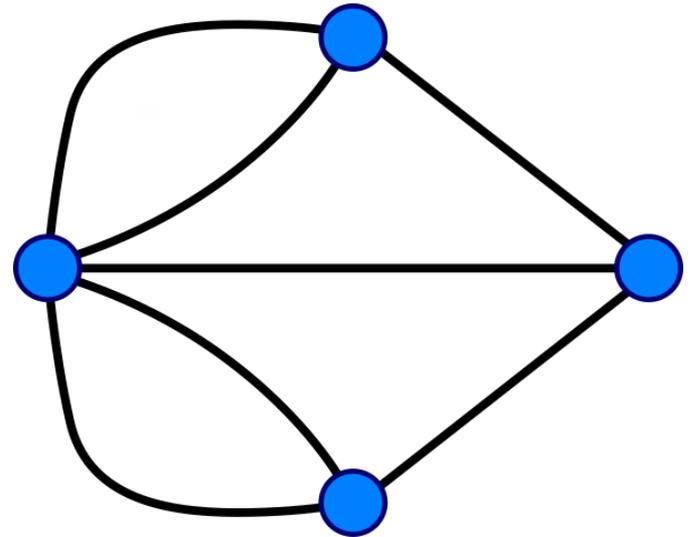
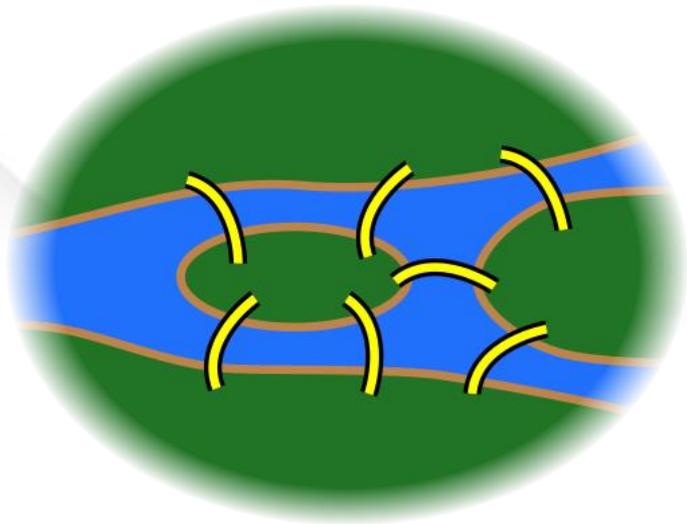


Branching

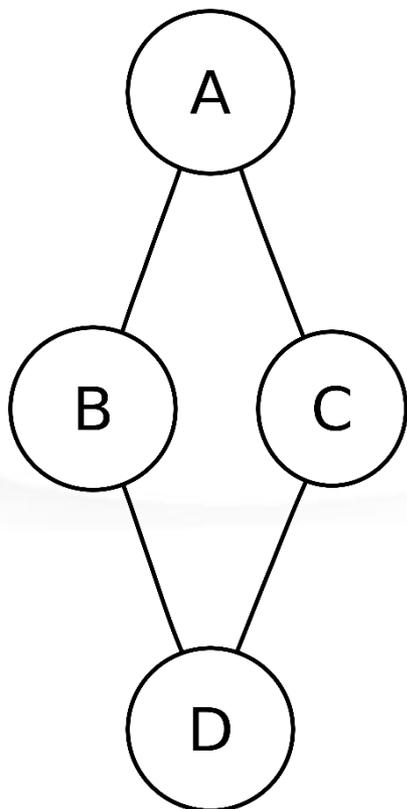


Laboratório de
Tecnologia da Informação
Aplicada

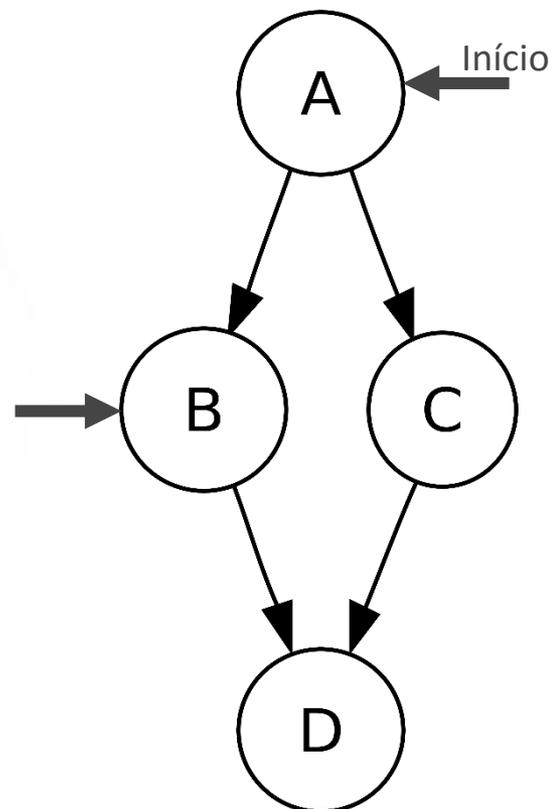




Grafos



Não Direcionado



Direcionado



Branching

Criar um *branch* significa dizer que você vai divergir da linha principal de desenvolvimento e continuar a trabalhar sem bagunçar essa linha principal.

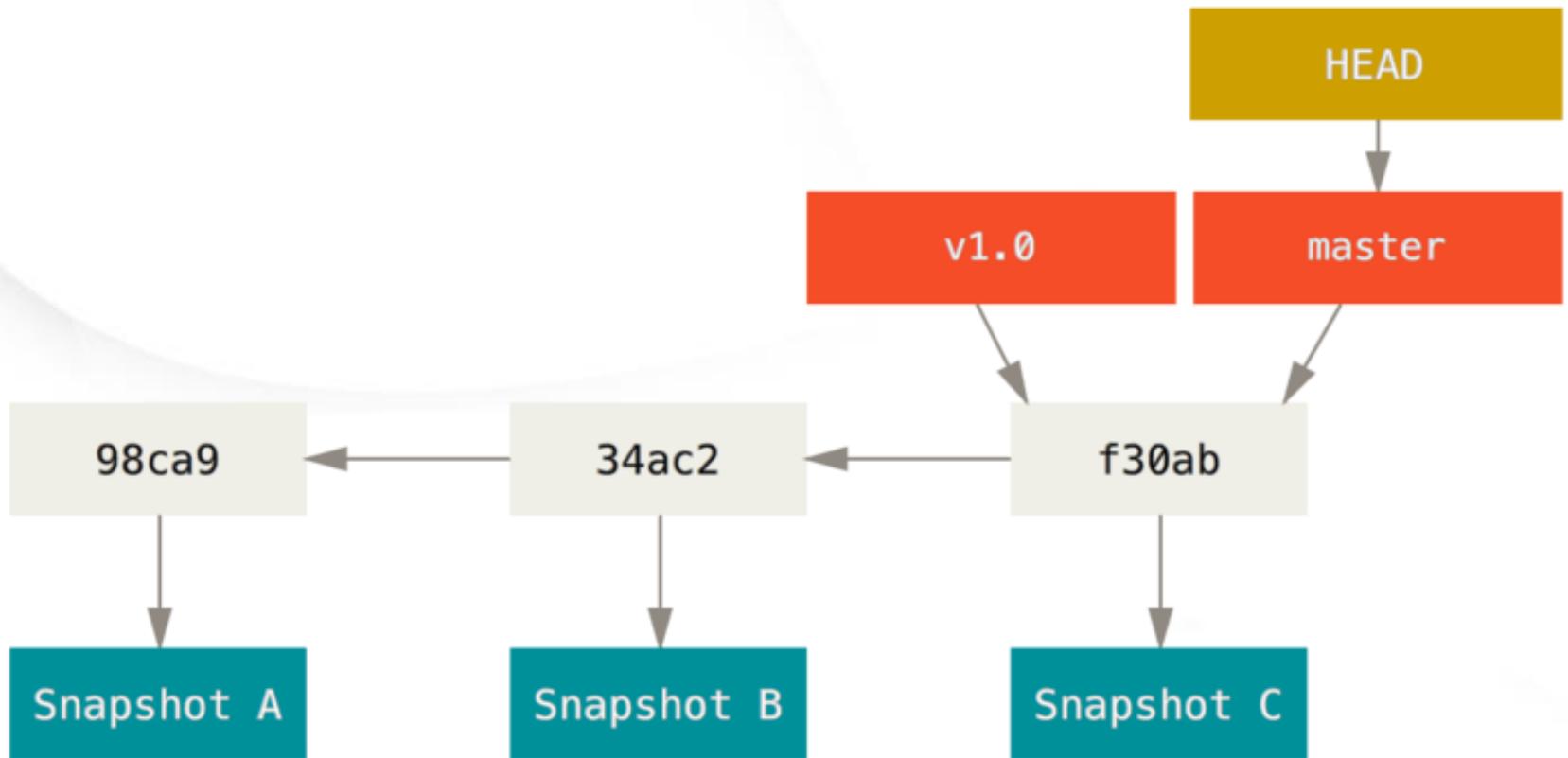
A forma como o Git cria *branches* é inacreditavelmente leve, fazendo com que as operações sejam praticamente instantâneas e a alternância entre os *branches* seja tão rápida quanto.

O Git incentiva um fluxo de trabalho no qual se fazem *branches* e *merges* com frequência, até mesmo várias vezes ao dia.

Pro Git, Scott Chacon e Ben Straub



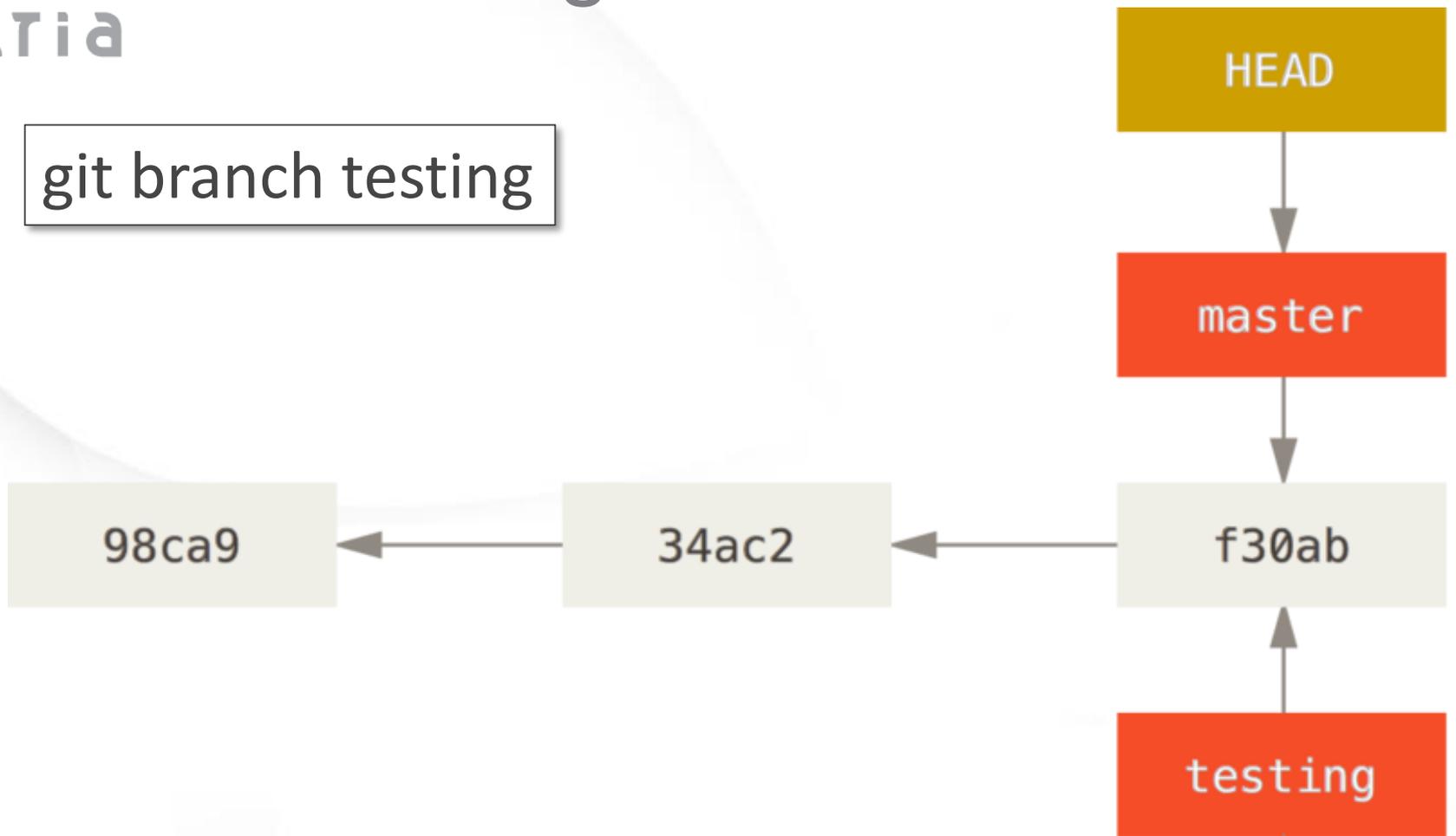
Branching





Branching

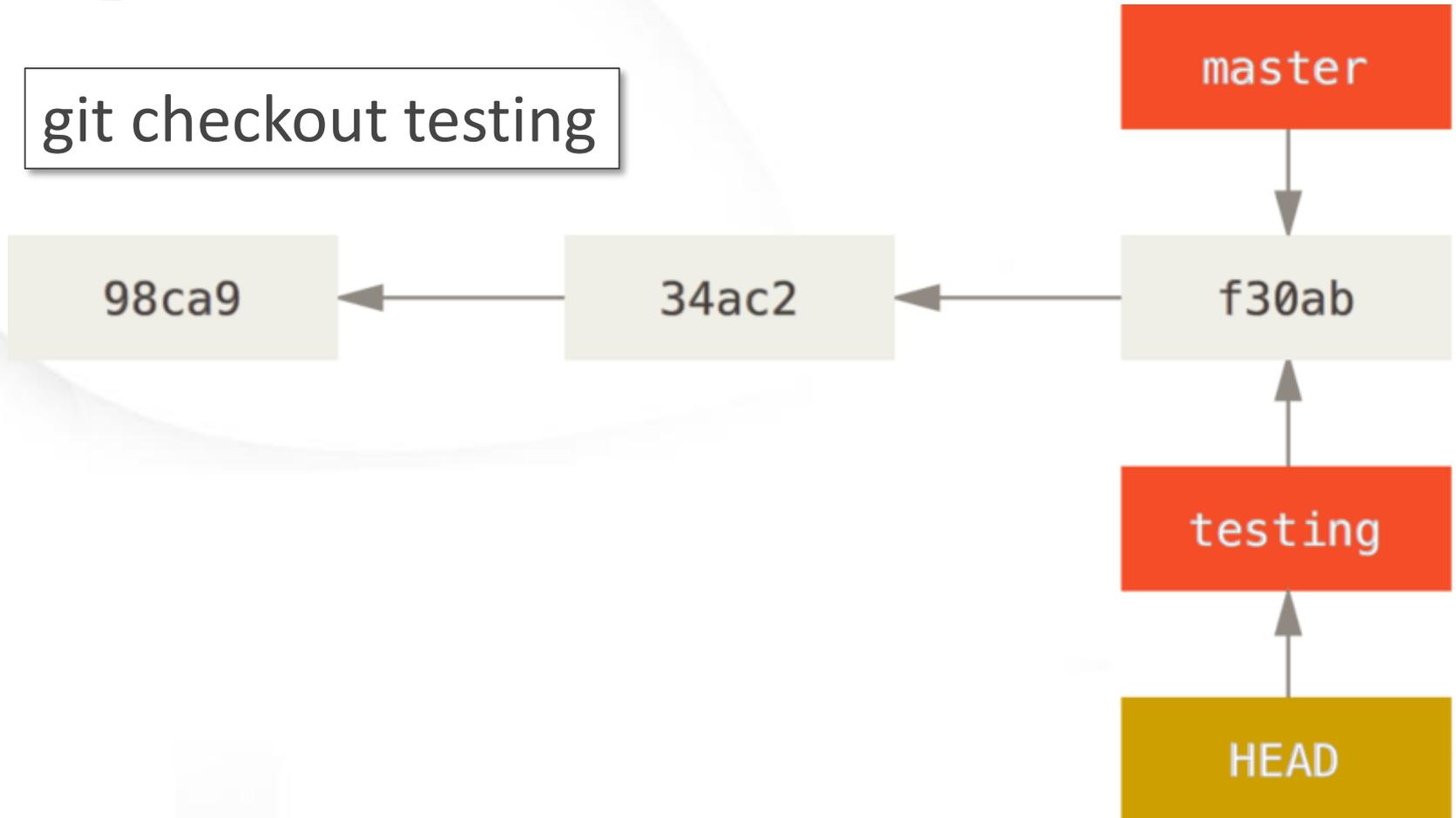
git branch testing





Branching

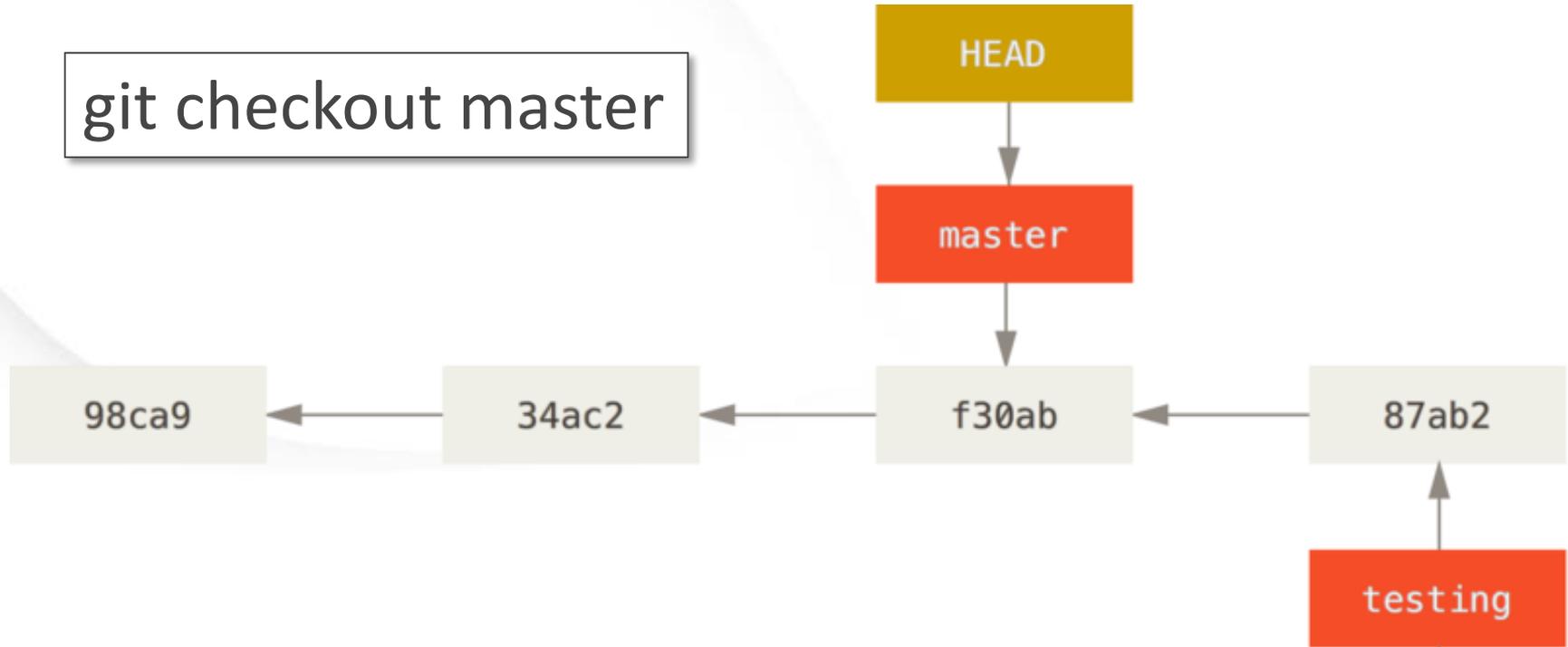
git checkout testing





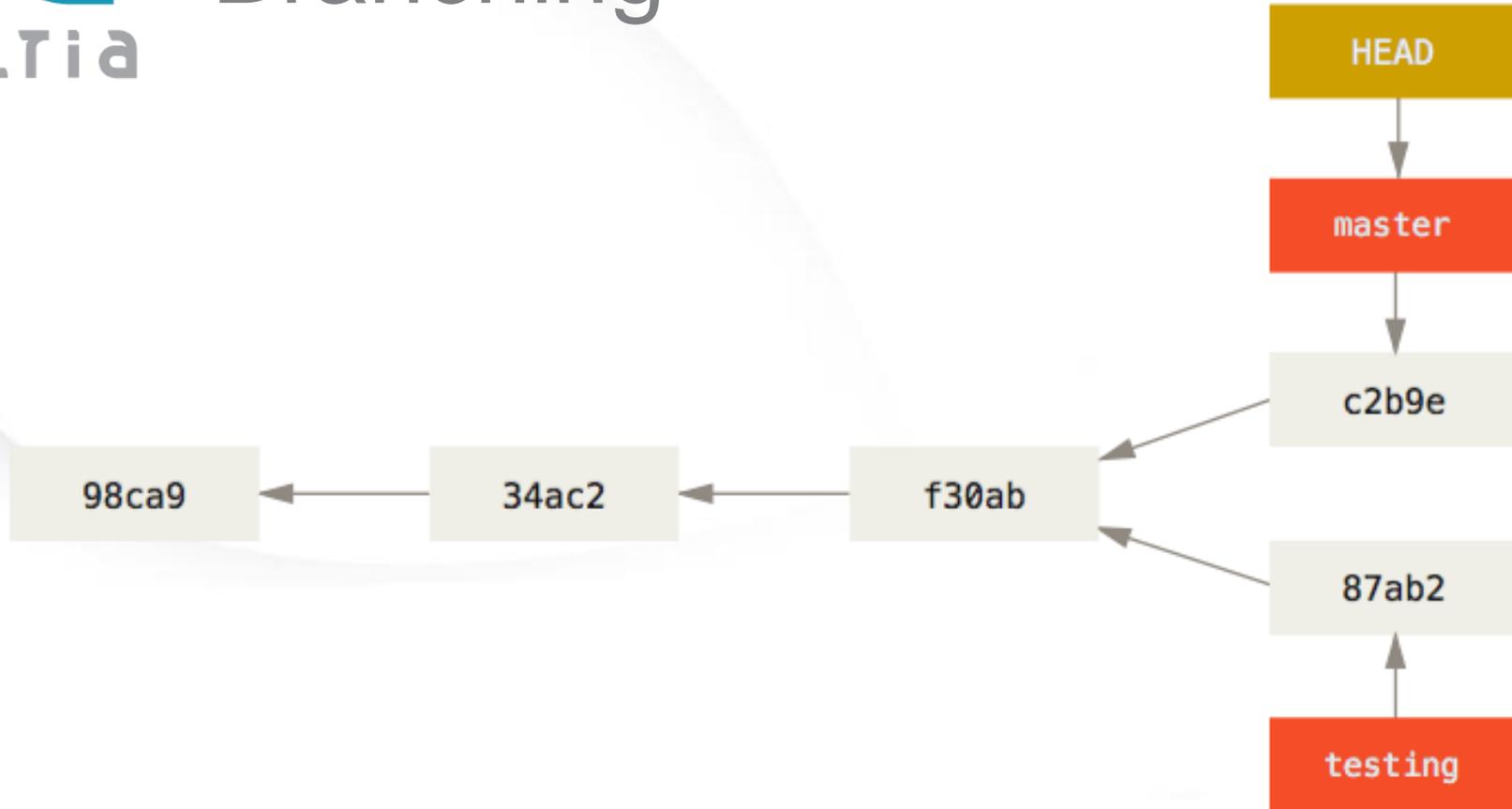
Branching

git checkout master





Branching

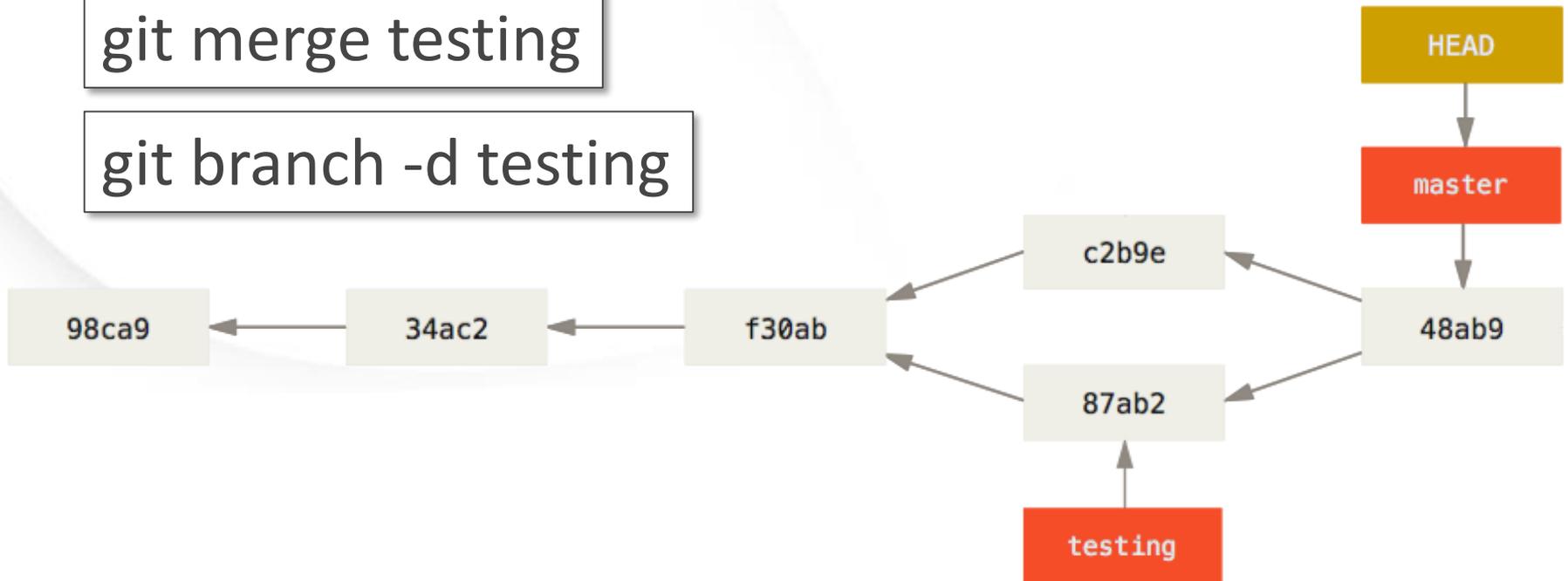




Branching

git merge testing

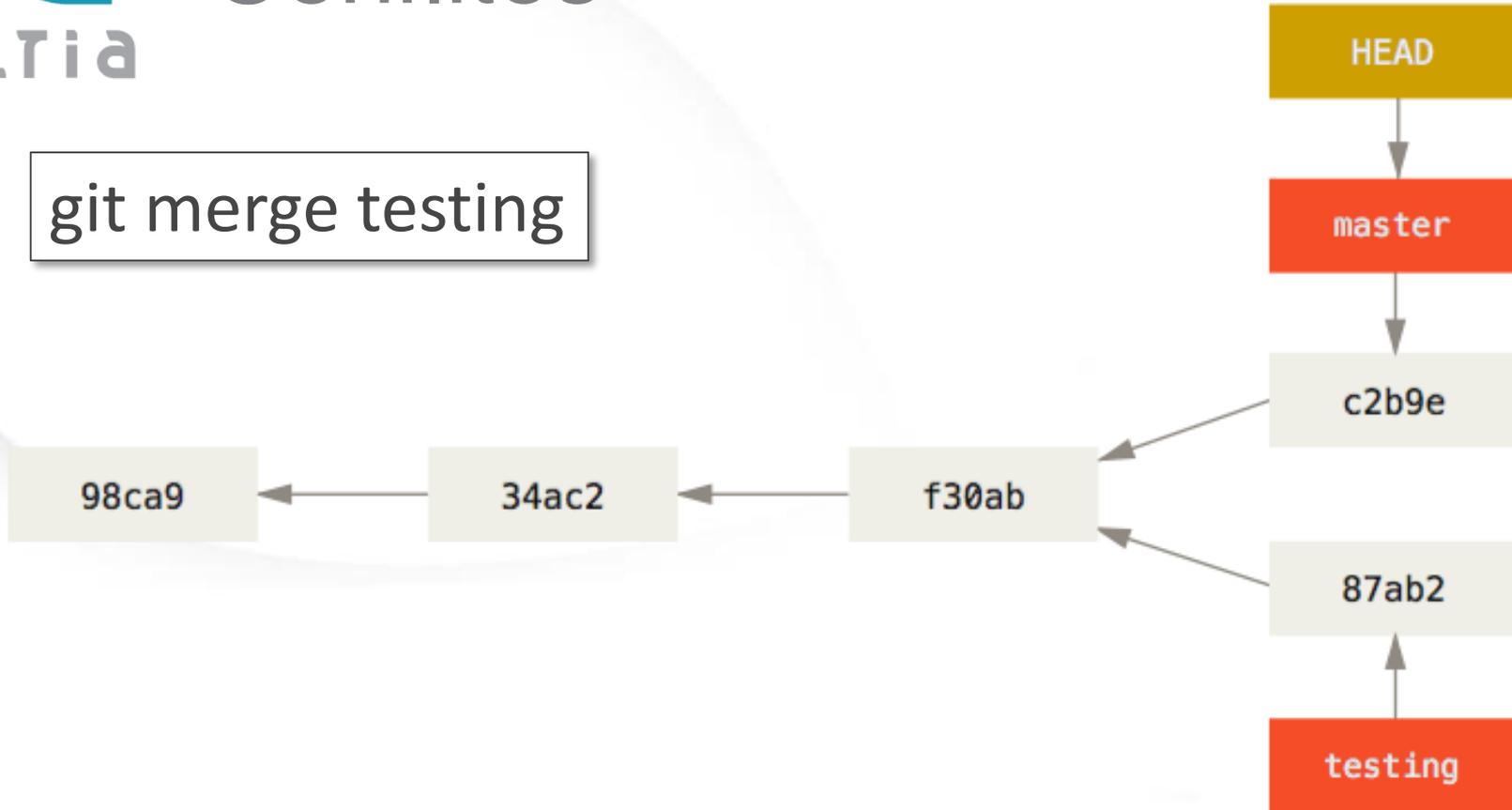
git branch -d testing





Conflitos

git merge testing





Conflitos

```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Exemplo (master)
$ git merge testing
Auto-merging main.html
CONFLICT (content): Merge conflict in main.html
Automatic merge failed; fix conflicts and then commit the result.
```

```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Exemplo (master|MERGING)
$ |
```



Conflitos

```
1 <html>
2   <head>
3     <link type="text/css" rel="stylesheet" media="all" href="style.css" />
4   </head>
5   <body>
6 </body>
7 </html>
```

```
1 <html>
2   <head>
3     <link type="text/css" rel="stylesheet" media="all" href="estilo.css" />
4   </head>
5   <body>
6 </body>
7 </html>
```



Conflitos

```
1 <html>
2   <head>
3 <<<<<< HEAD|
4     <link type="text/css" rel="stylesheet" media="all" href="estilo.css" />
5 =====
6     <link type="text/css" rel="stylesheet" media="all" href="style.css" />
7 >>>>>> testing
8   </head>
9   <body>
10  </body>
11 </html>
```



Conflitos

```
1 <html>
2   <head>
3     <link type="text/css" rel="stylesheet" media="all" href="style.css" />
4   </head>
5   <body>
6 </body>
7 </html>
```



Conflitos

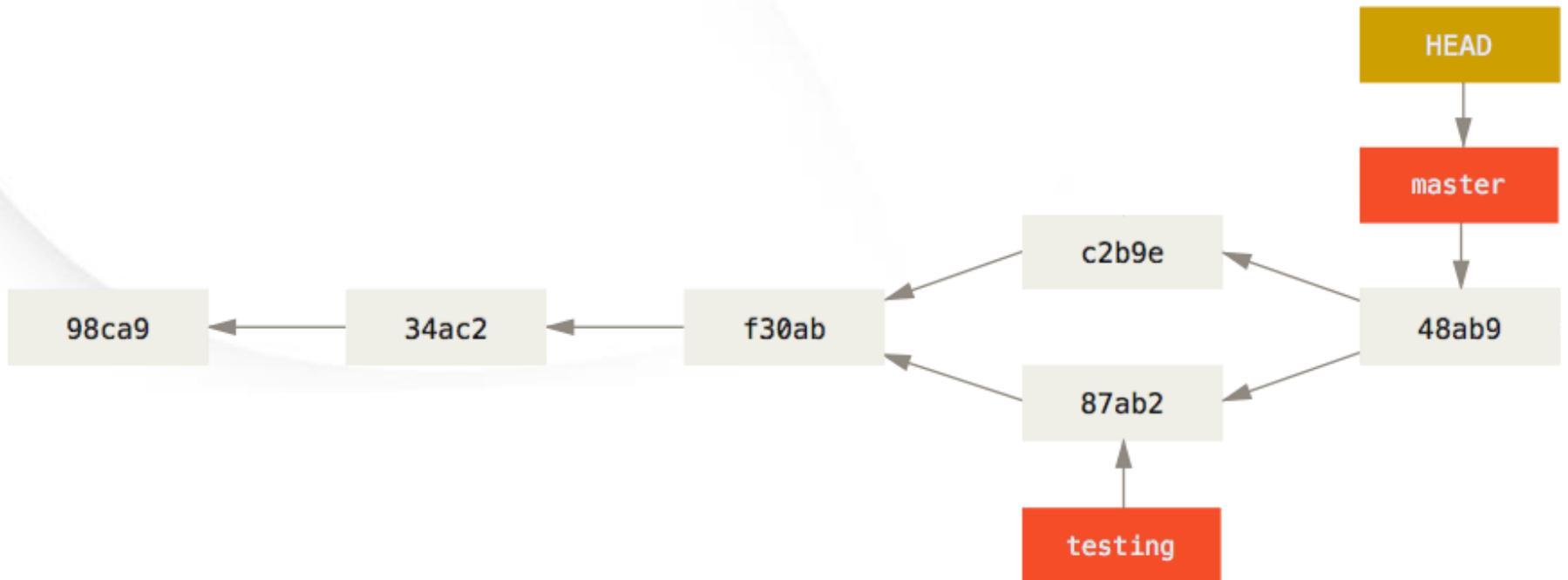
```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Exemplo (master|MERGING)
$ git add main.html
```

```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Exemplo (master|MERGING)
$ git commit -m "Merged com testing, conflito resolvido"
[master 0df50af] Merged com testing, conflito resolvido
```

```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Exemplo (master)
$ |
```



Conflitos





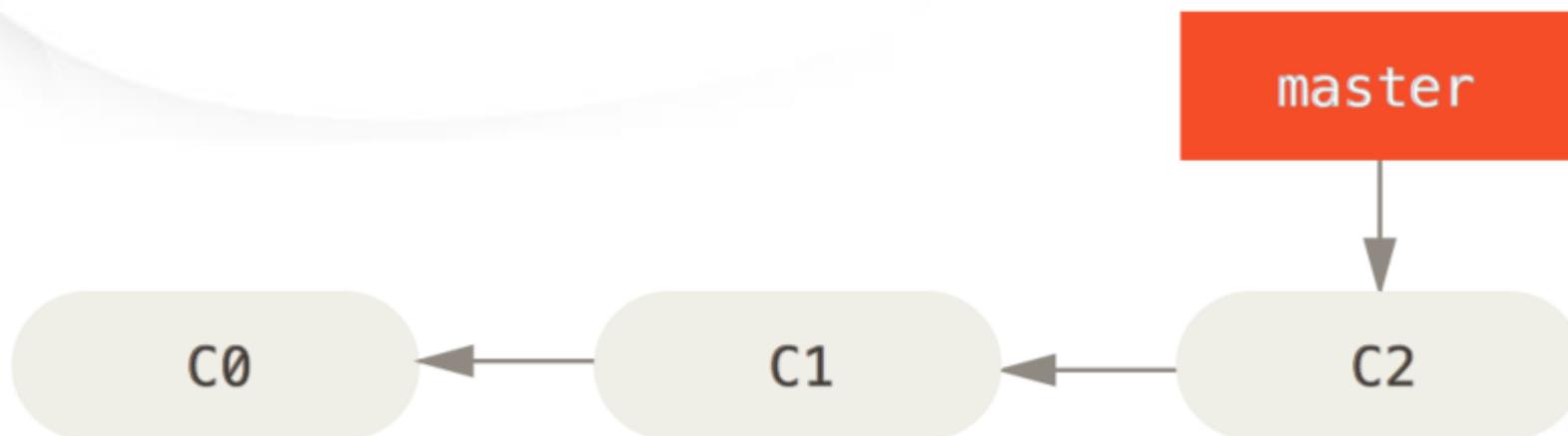
Tags

- `git tag -a v1.0 -m "Versão 1.0"`
- `git tag -a v2.0 -m 98ca9 "Versão 2.0"`
- `git push --tags`



Branching - Exercício

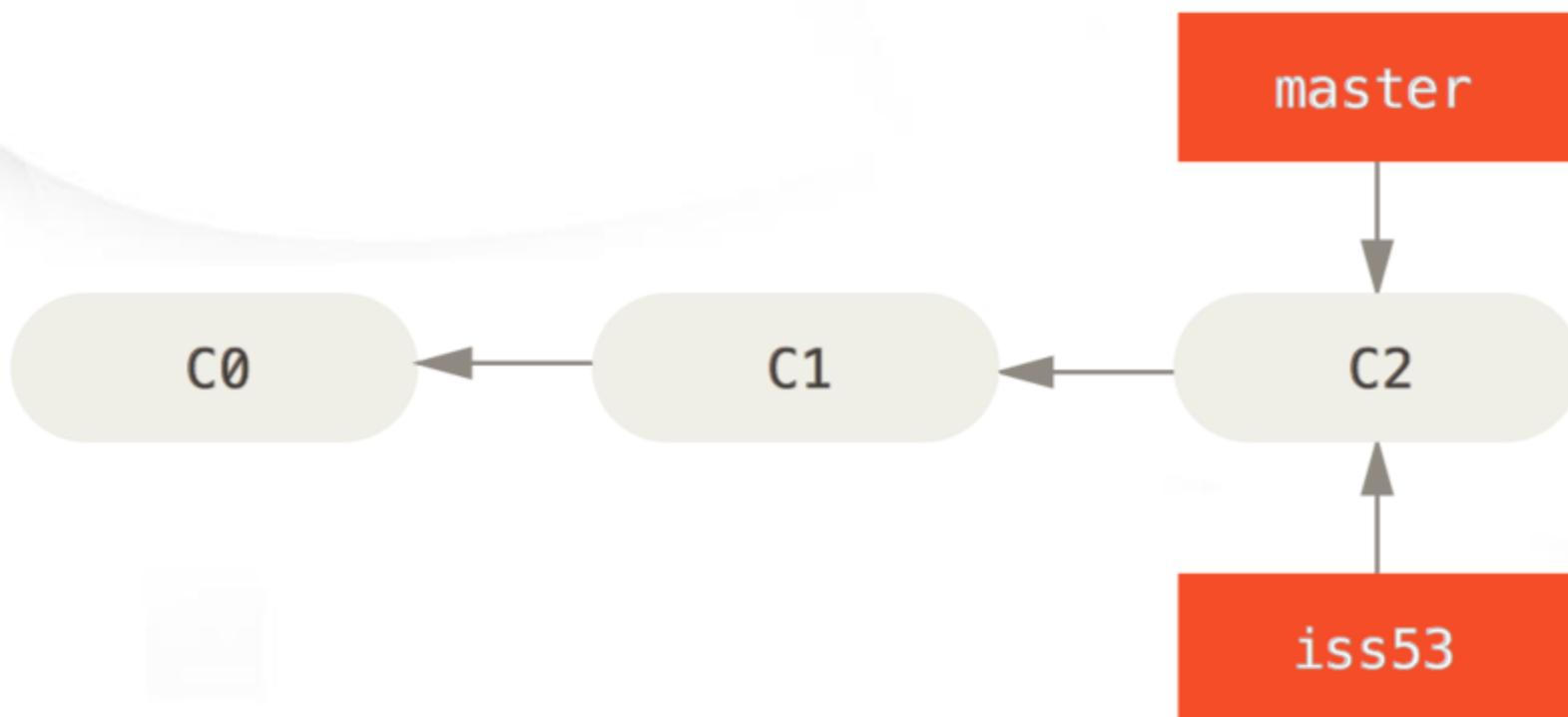
Suponha que você está trabalhando em um site. Inicialize e configure o diretório, crie um arquivo e faça três commits com ele, alterando algo no arquivo entre cada commit.





Branching - Exercício

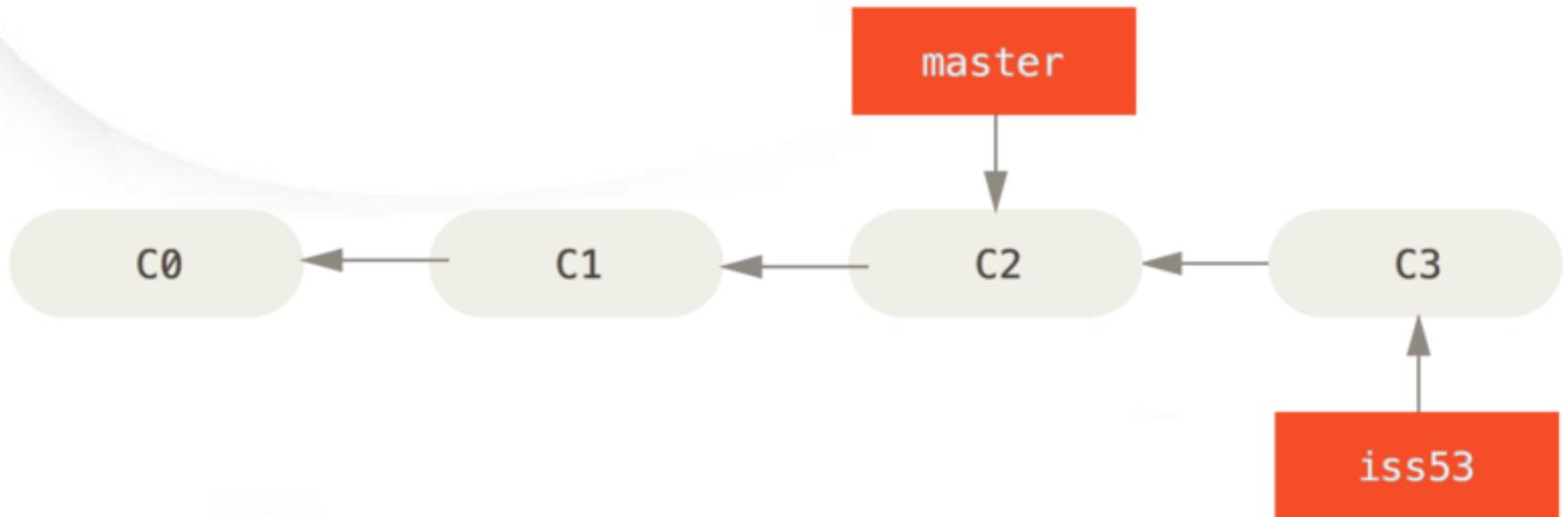
Agora você decide resolver o Issue #53, portanto crie um branch para ele (*iss53*).





Branching - Exercício

Crie outro arquivo, e adicione-o em um commit para o branch *iss53*.

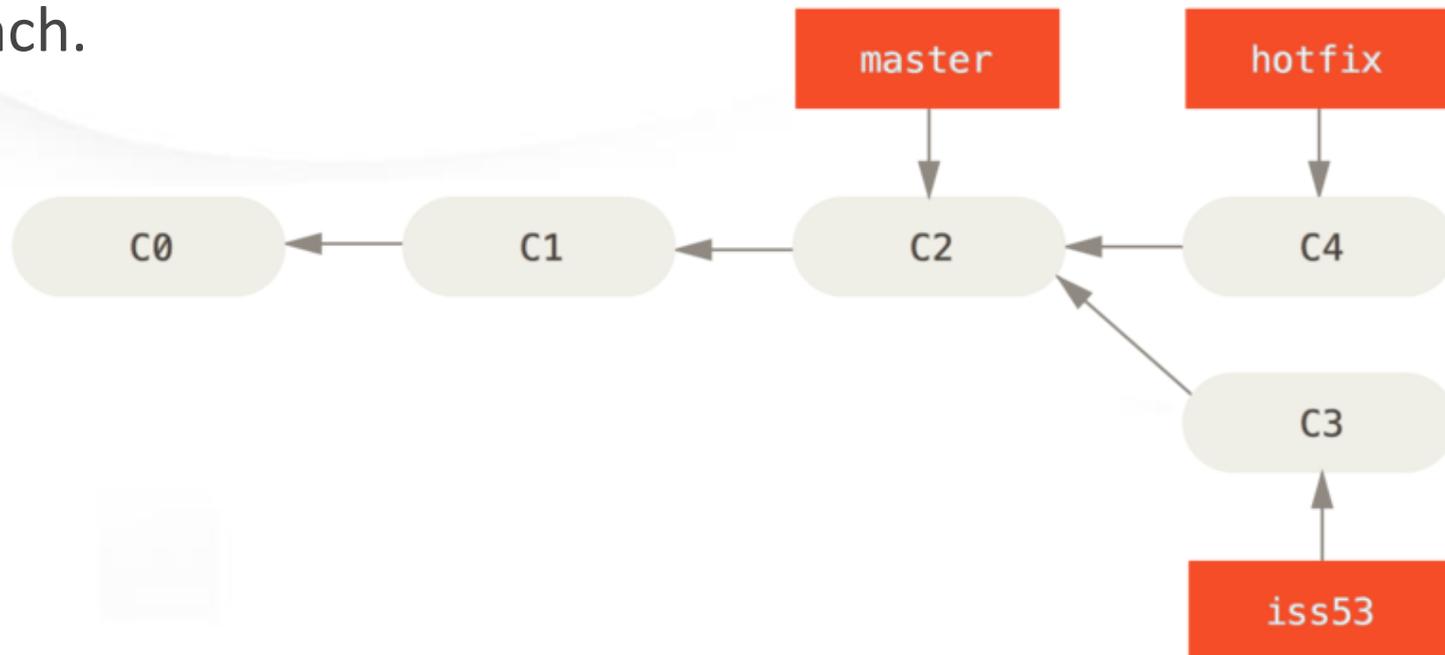




Branching - Exercício

Encontraram um problema urgente no site que precisa ser resolvido antes de qualquer outra coisa.

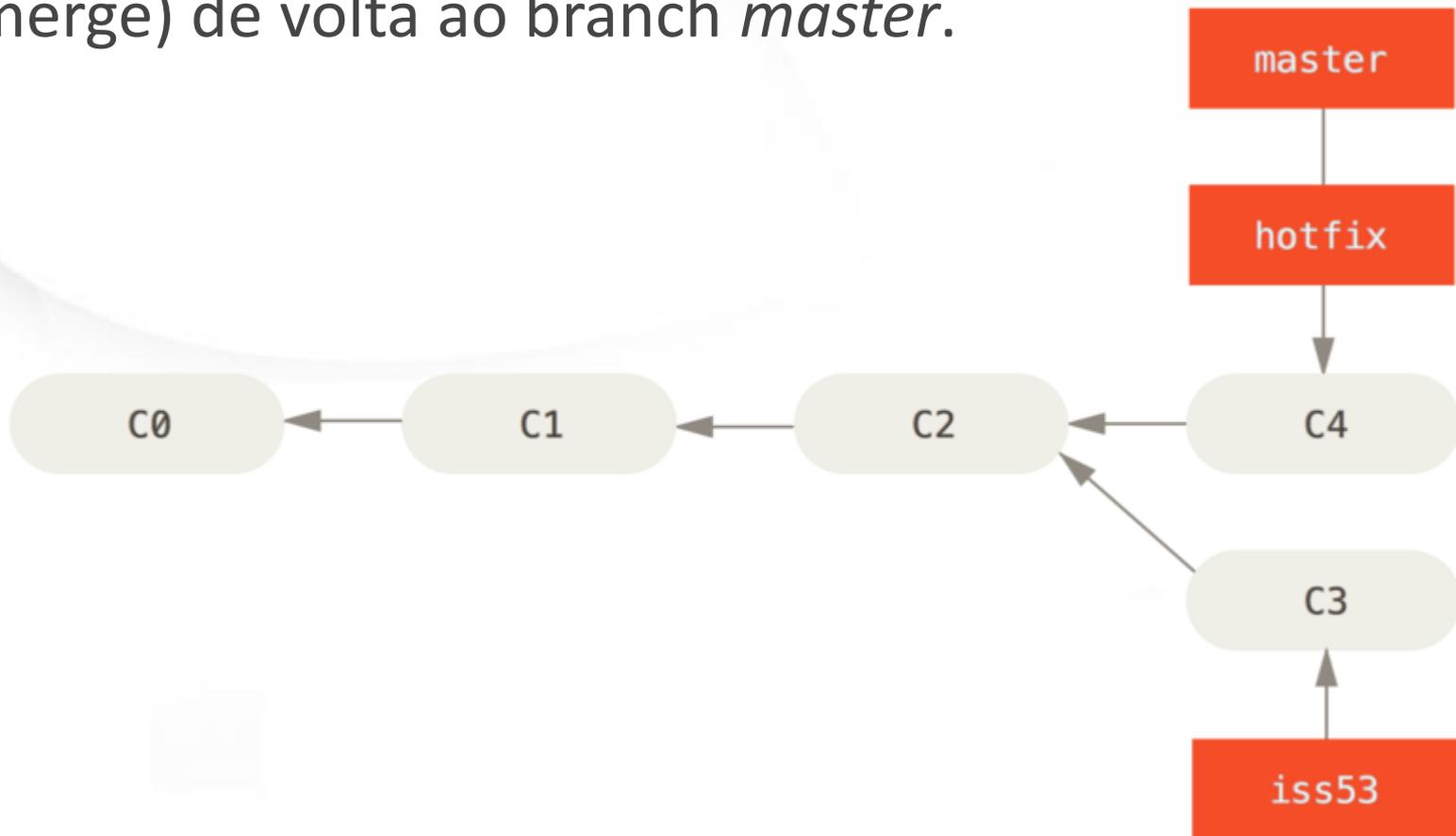
Crie um novo branch para resolver o problema (*hotfix*), crie um novo arquivo e adicione-o em um commit para esse novo branch.





Branching - Exercício

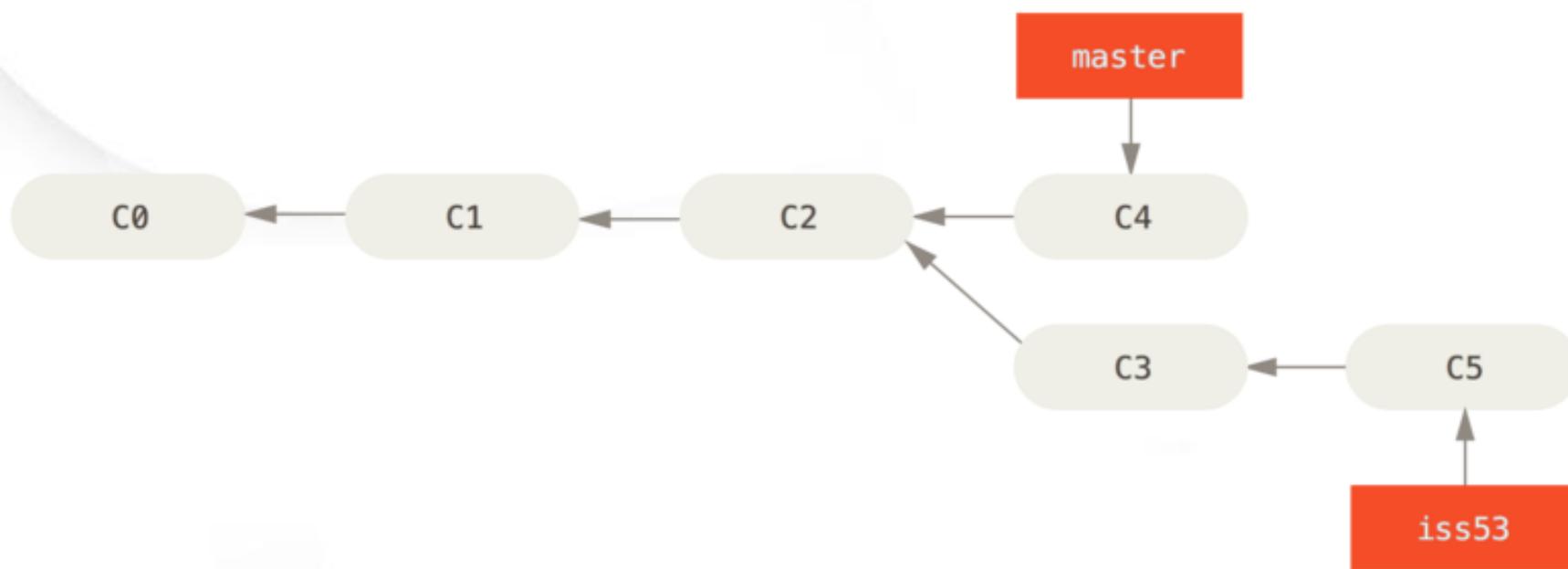
O problema urgente foi resolvido! Incorpore ele (merge) de volta ao branch *master*.





Branching - Exercício

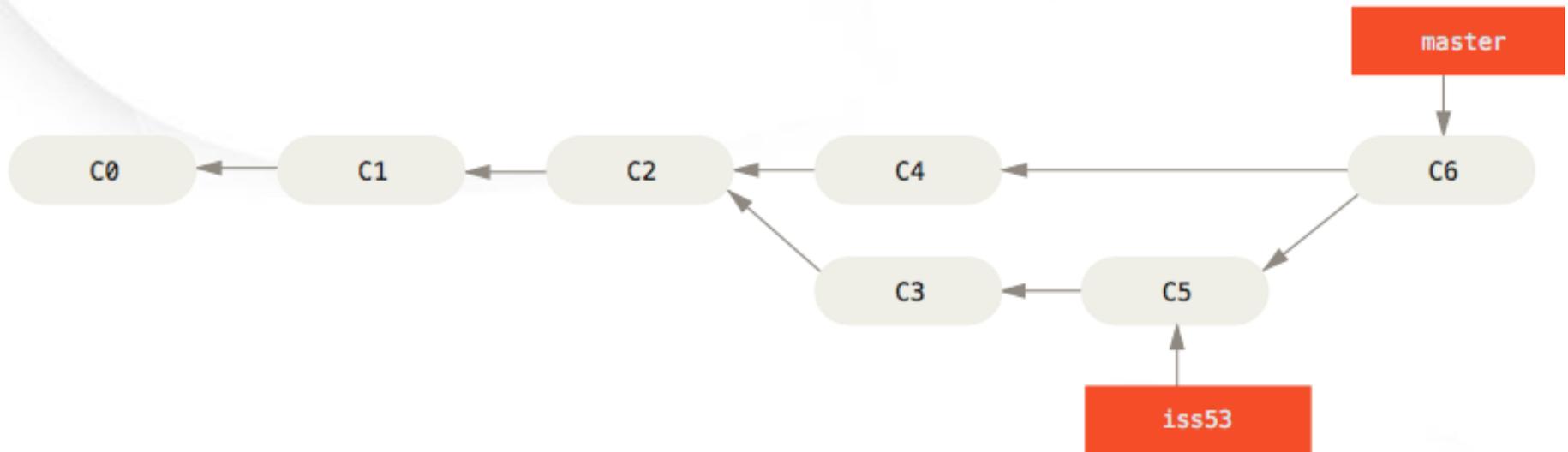
Exclua o branch *hotfix* e volte a trabalhar no Issue #53. Altere algum arquivo e faça um commit.





Branching - Exercício

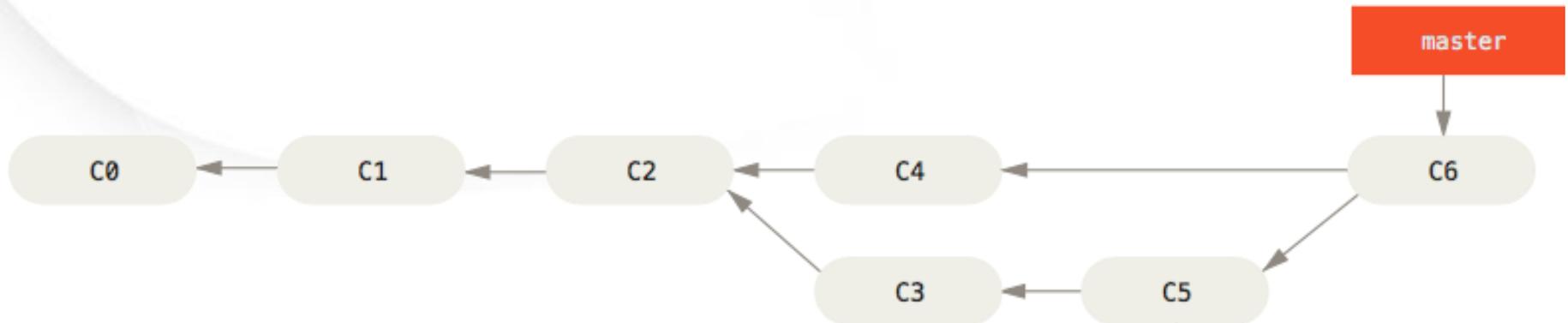
O Issue #53 foi resolvido! Incorpore as mudanças de volta ao branch *master*.





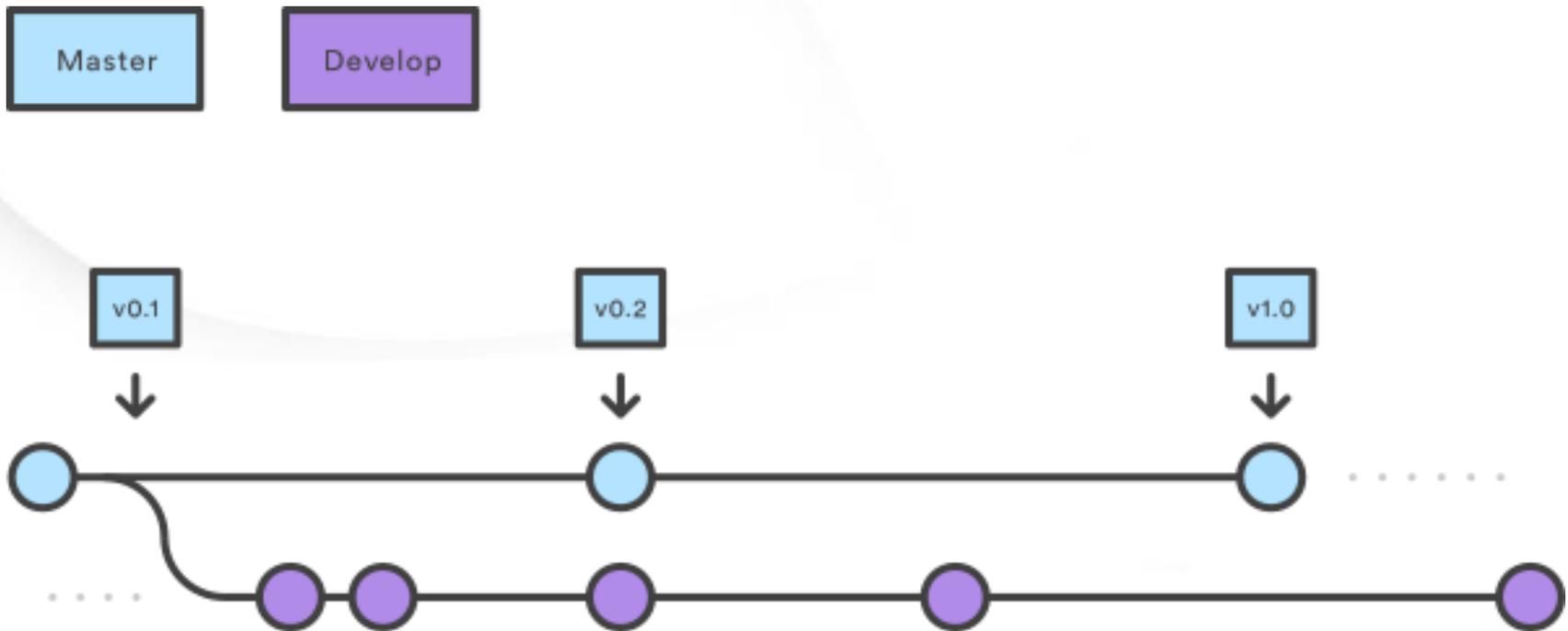
Branching - Exercício

Exclua o branch *iss53*.



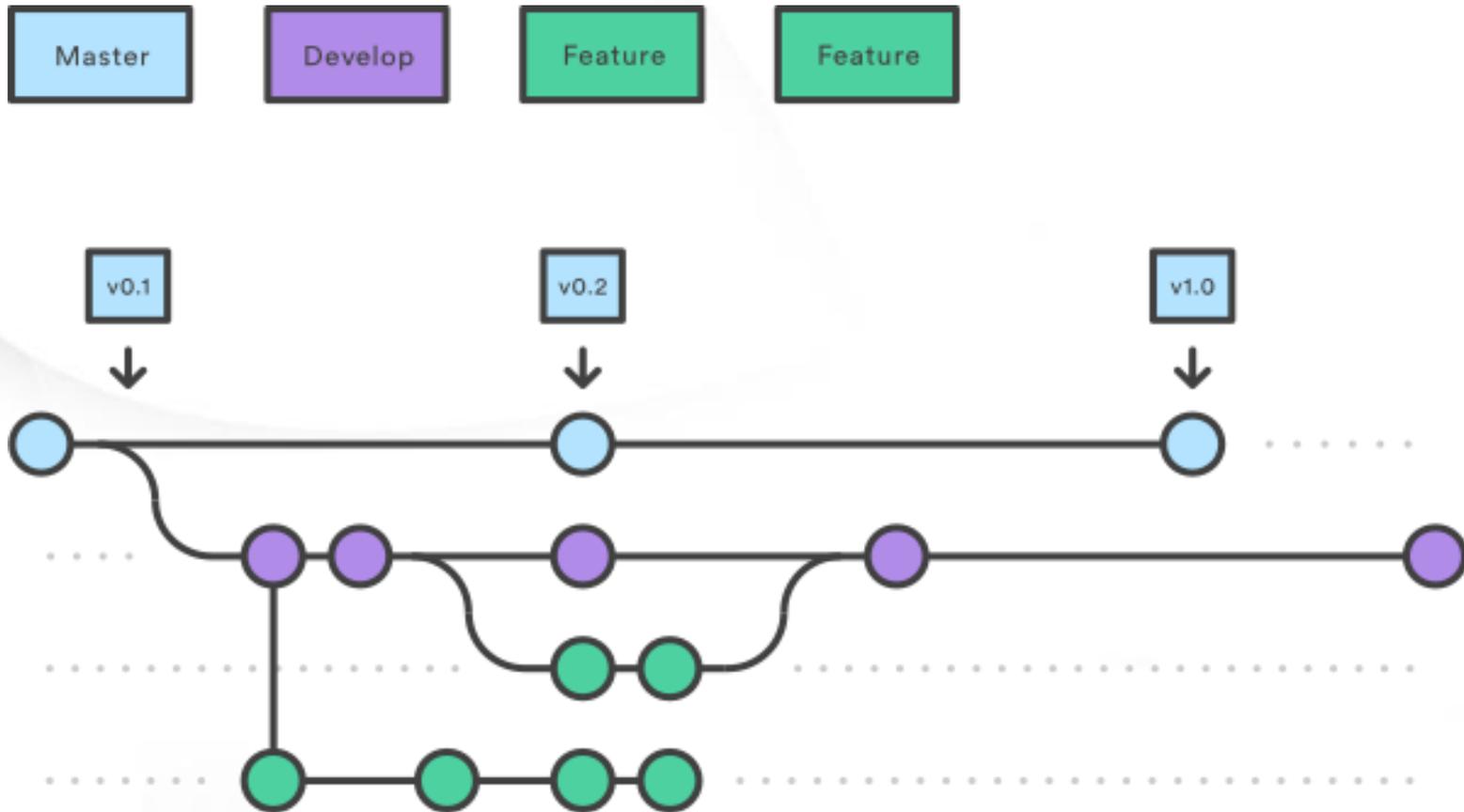


Branch Workflow



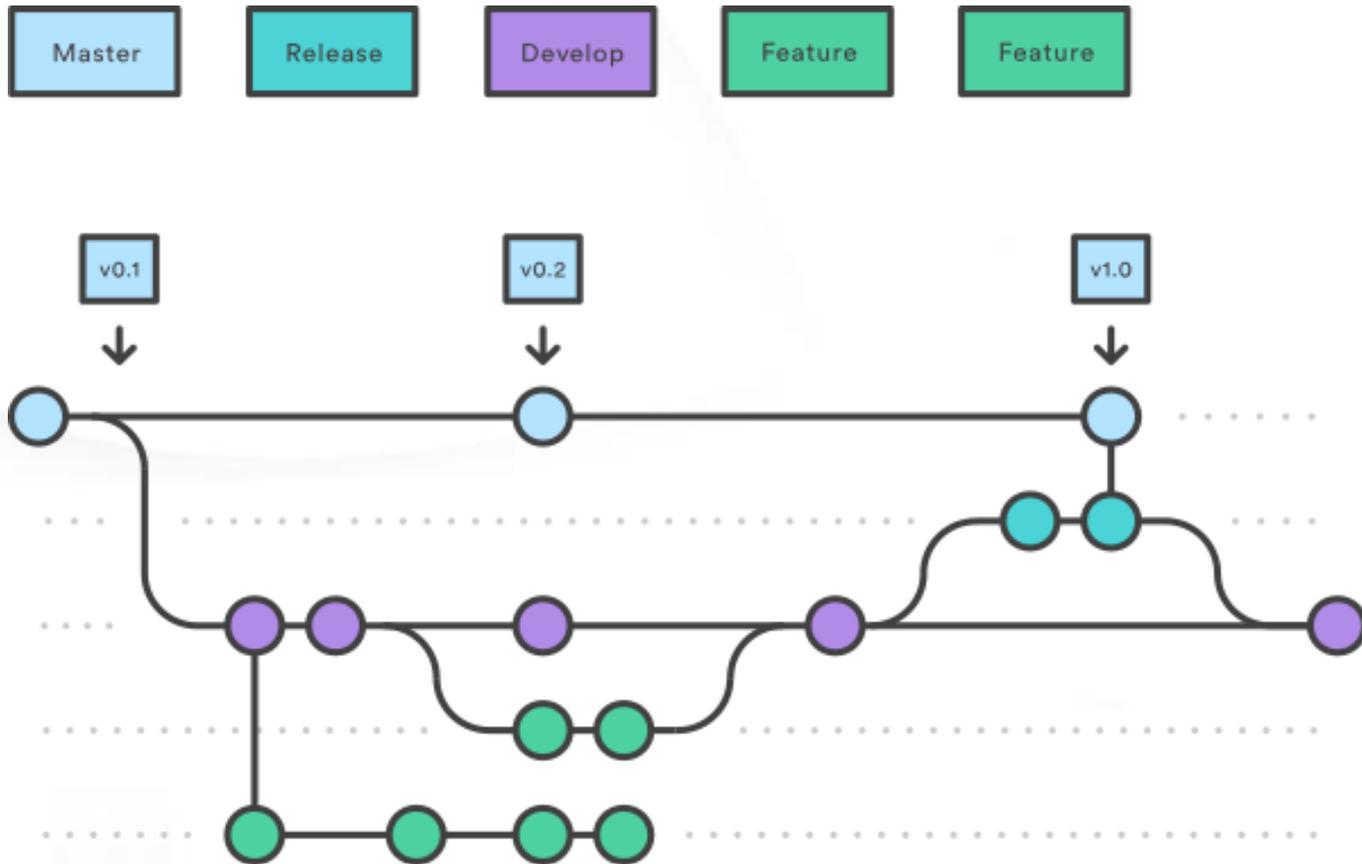


Branch Workflow



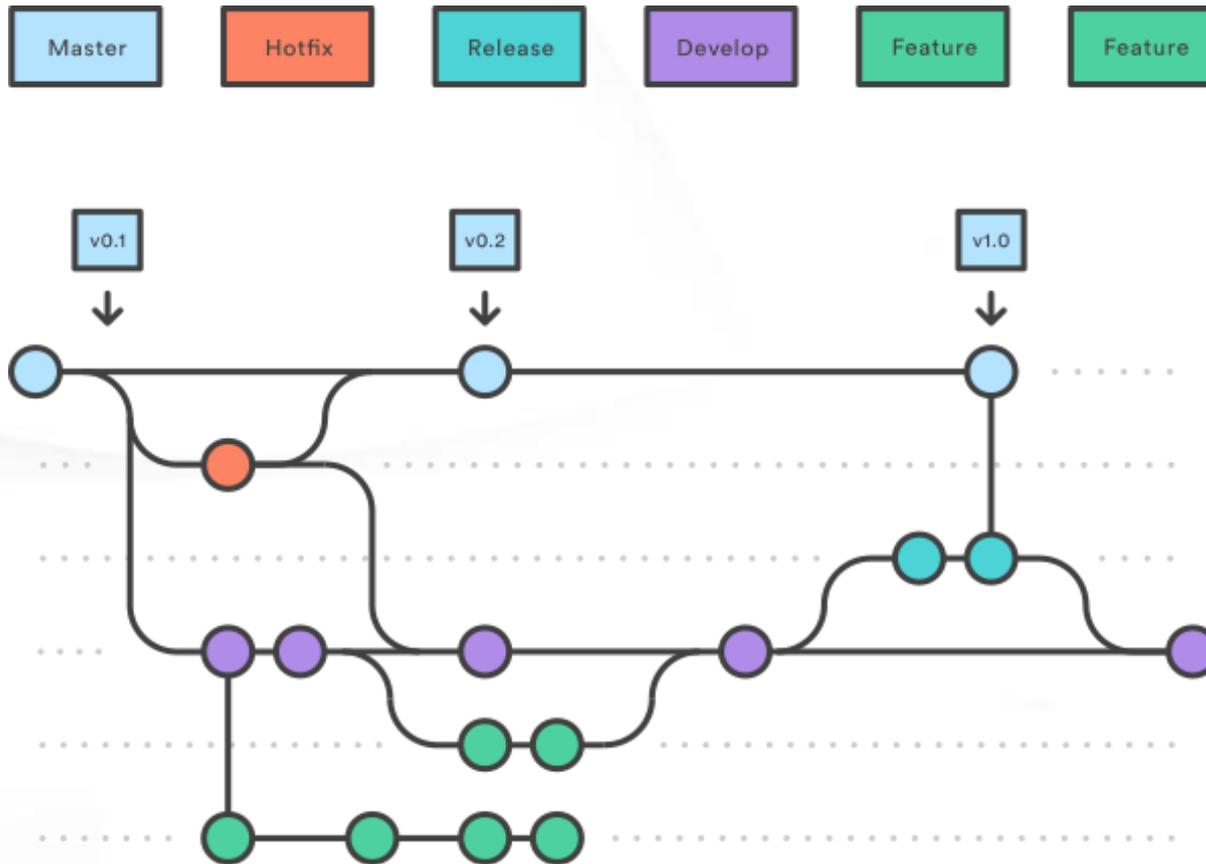


Branch Workflow





Branch Workflow



Remote Branches



Laboratório de
Tecnologia da Informação
Aplicada



Autenticação - HTTPS

- Não seguro: `git config credential.helper store`
- Windows: `git config credential.helper wincred`

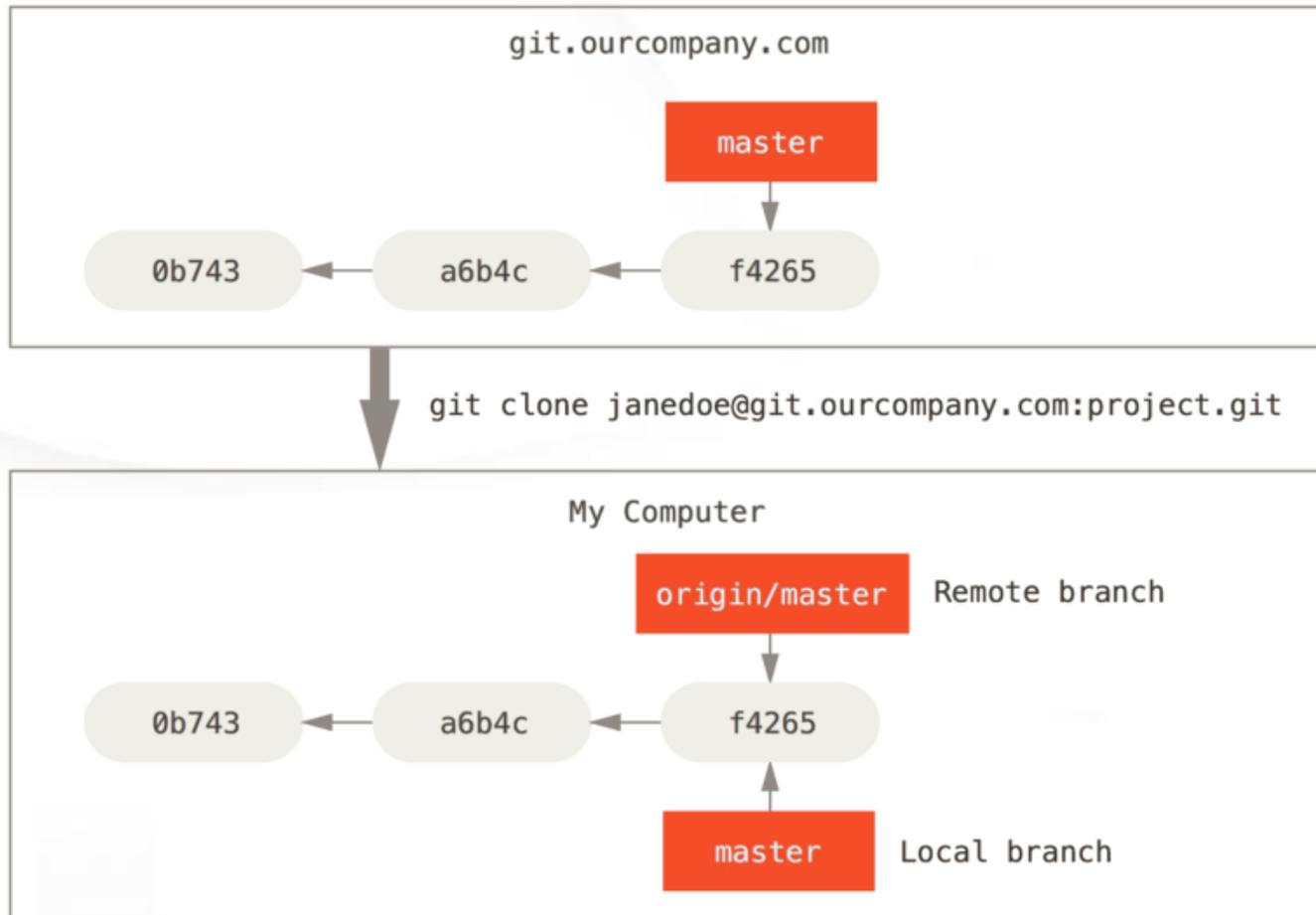


Autenticação - SSH

- Checar se já existe uma SSH key;
 - Criar uma SSH key;
- Adicionar a SSH key ao ssh-agent;
- Adicionar a SSH key ao GitHub;
- <https://help.github.com/articles/generating-an-ssh-key/>

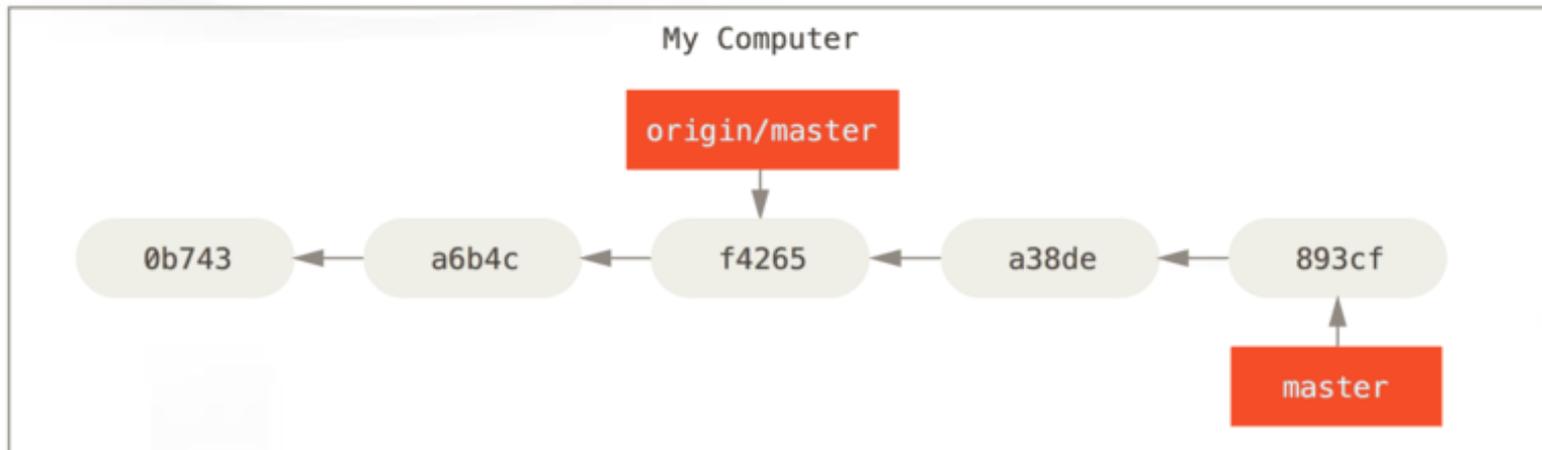
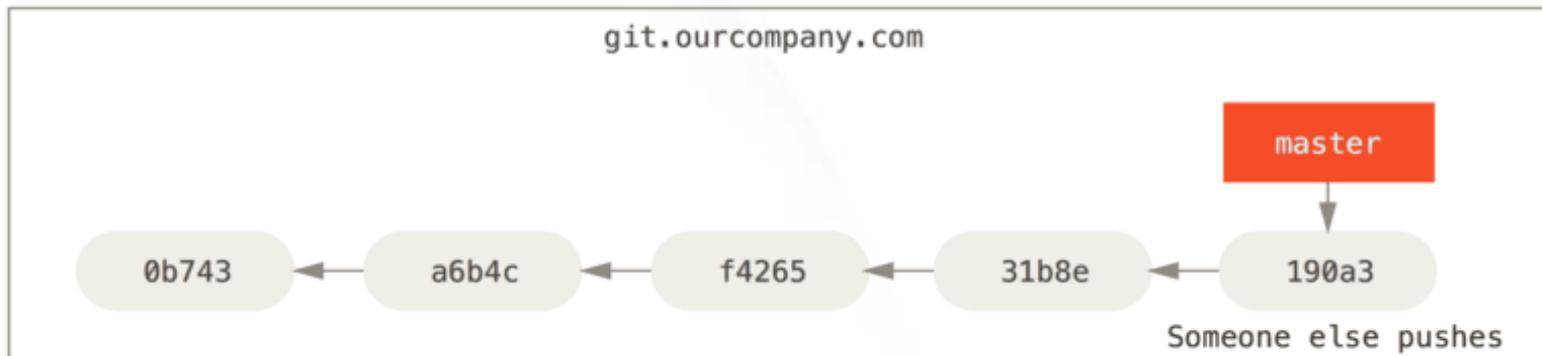


Remote Branches



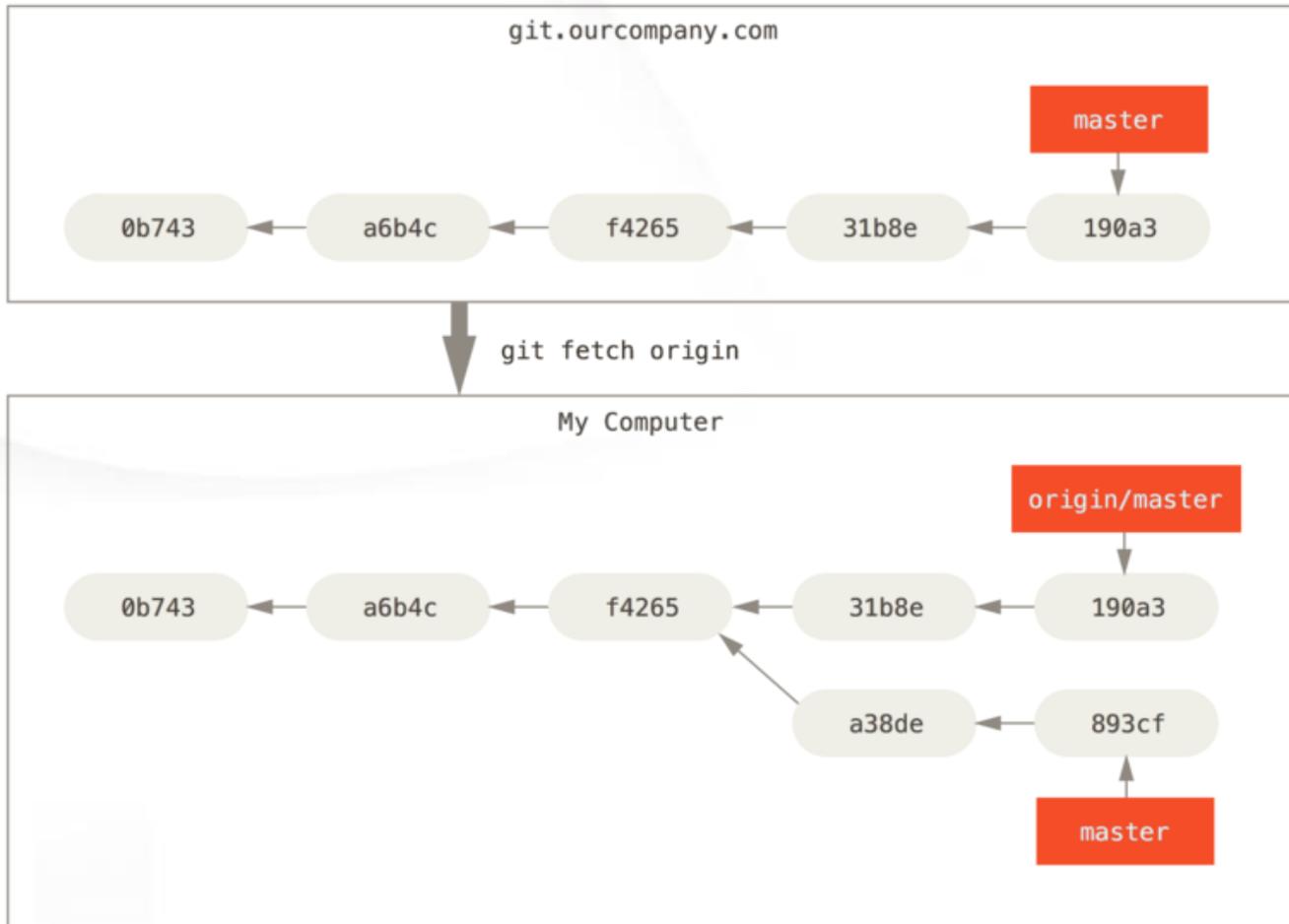


Remote Branches



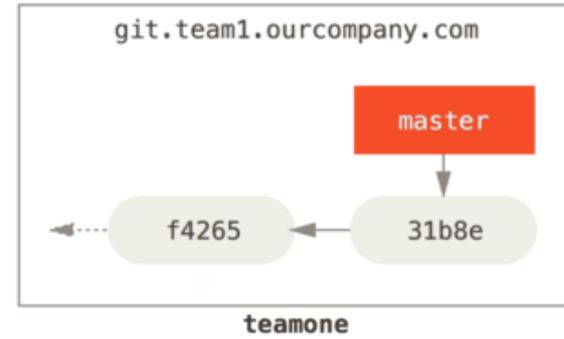
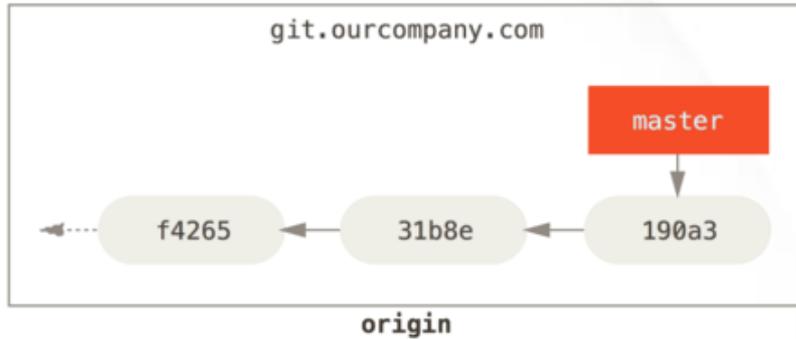


Remote Branches

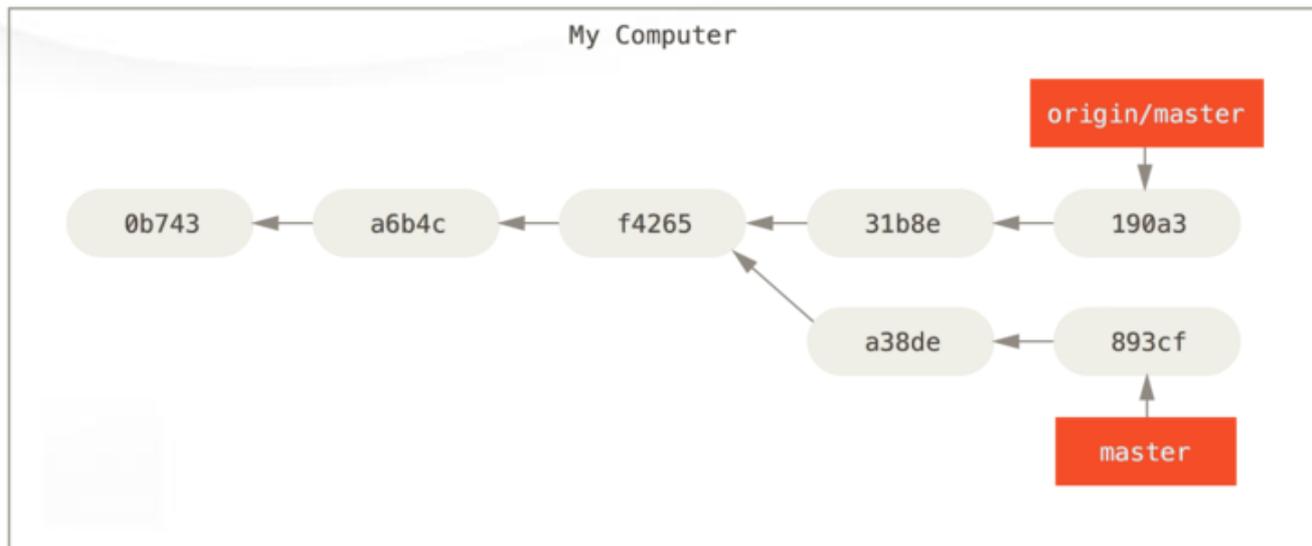




Remote Branches

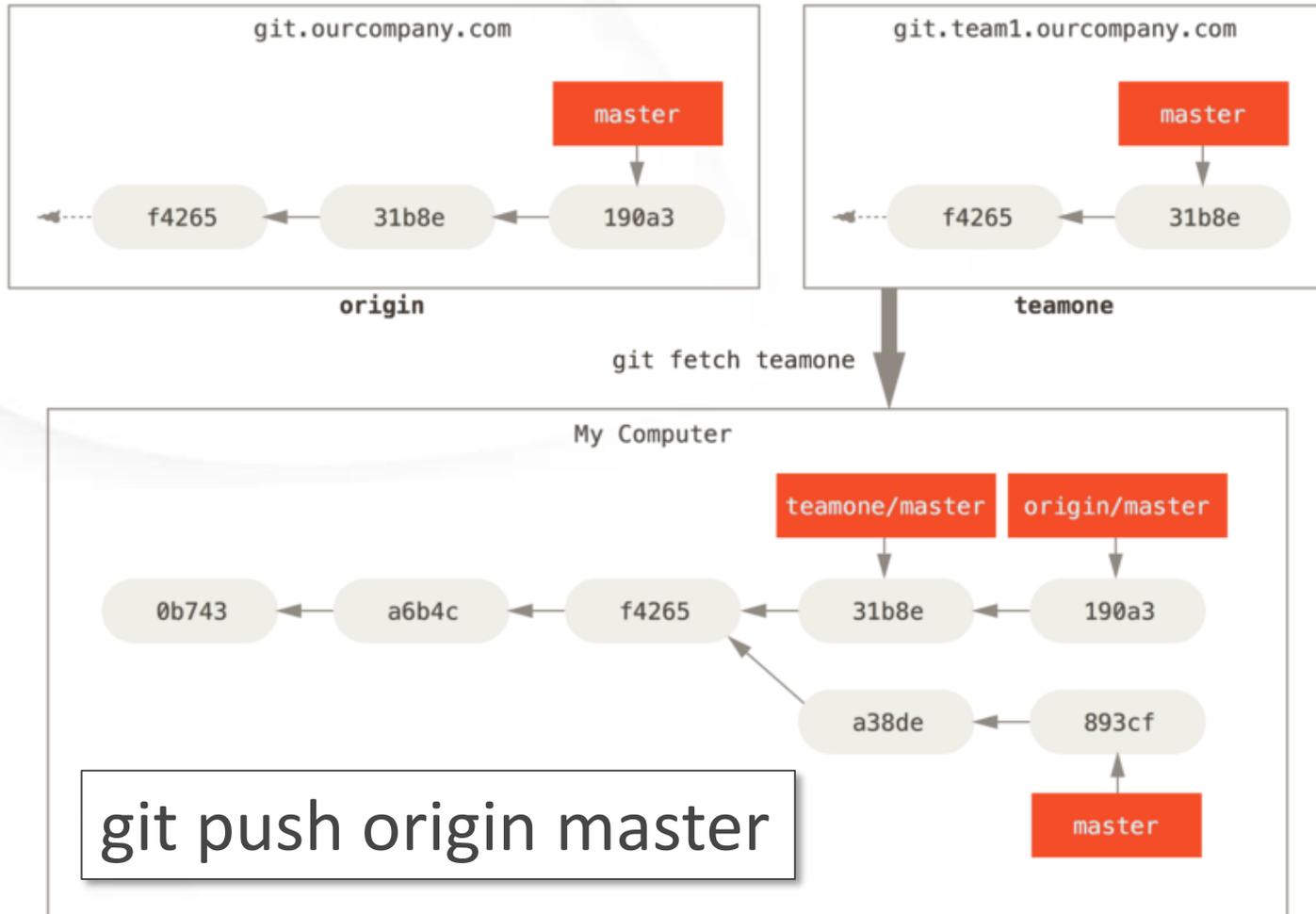


```
git remote add teamone git://git.team1.ourcompany.com
```





Remote Branches



Desfazendo Mudanças



Laboratório de
Tecnologia da Informação
Aplicada



Checkout

- Checkout arquivos;
- Checkout commits; —————> Read-Only
- Checkout branches.

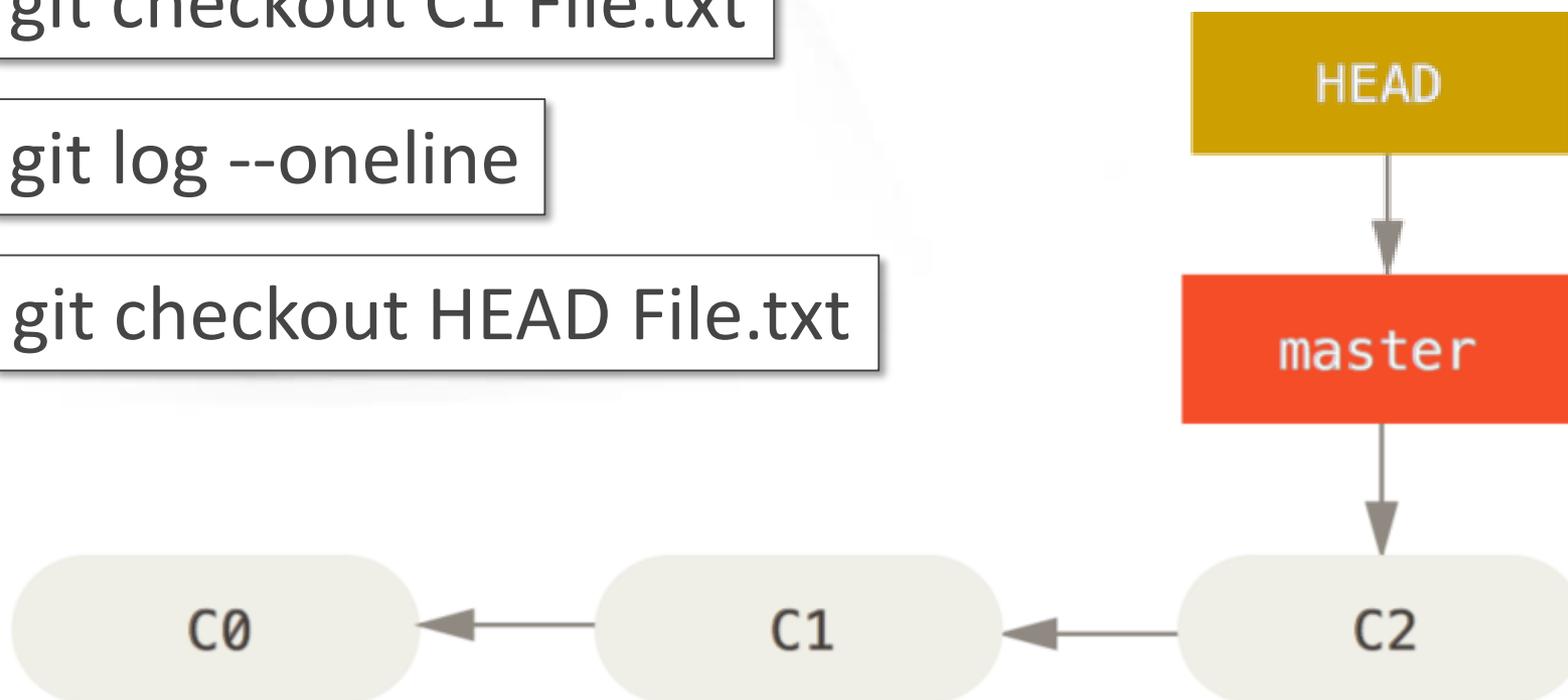


Checkout - arquivo

```
git checkout C1 File.txt
```

```
git log --oneline
```

```
git checkout HEAD File.txt
```

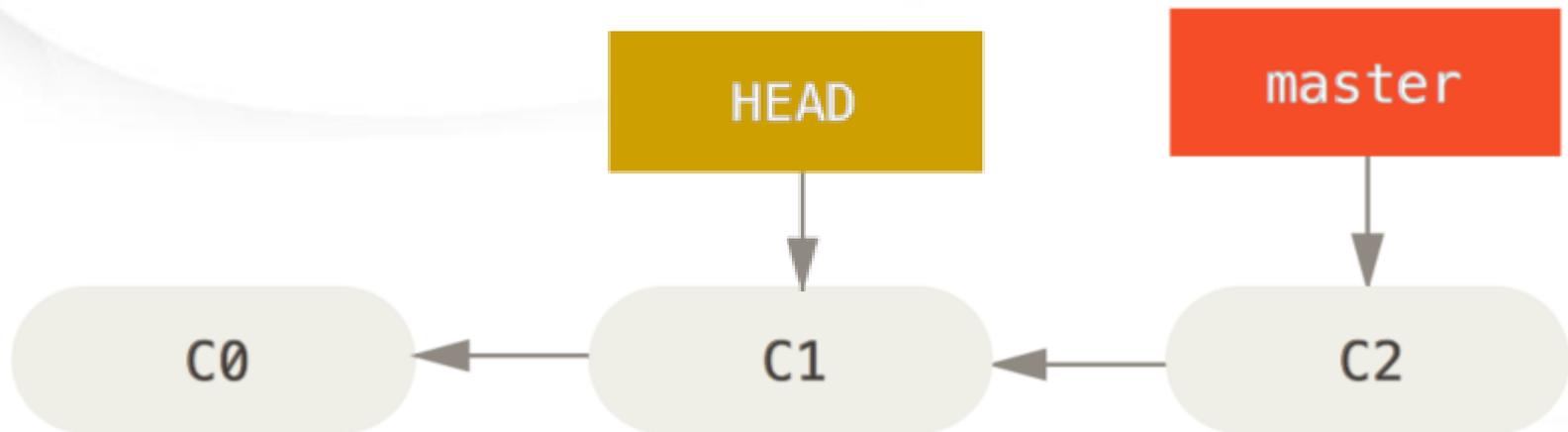




Checkout - commit

```
git checkout C1
```

```
git checkout master
```



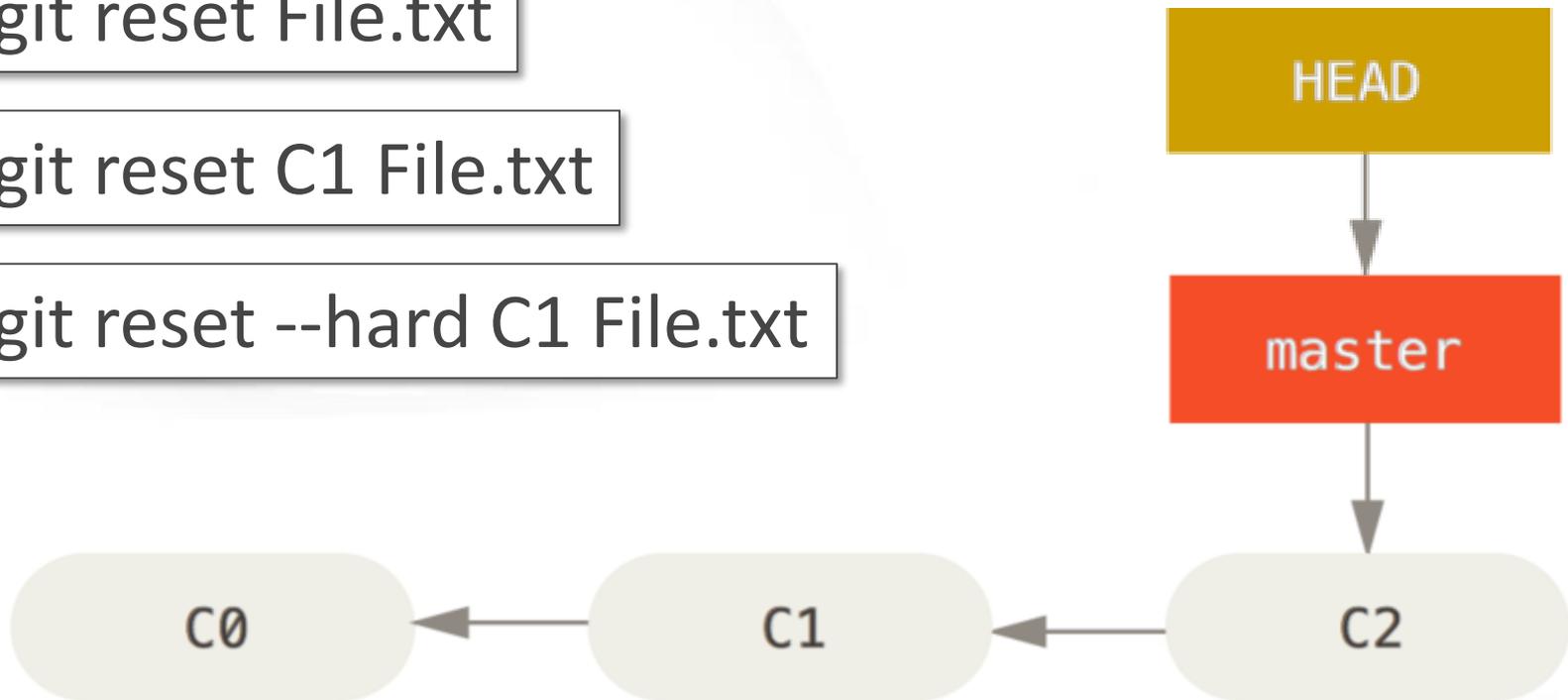


Reset - arquivo

```
git reset File.txt
```

```
git reset C1 File.txt
```

```
git reset --hard C1 File.txt
```

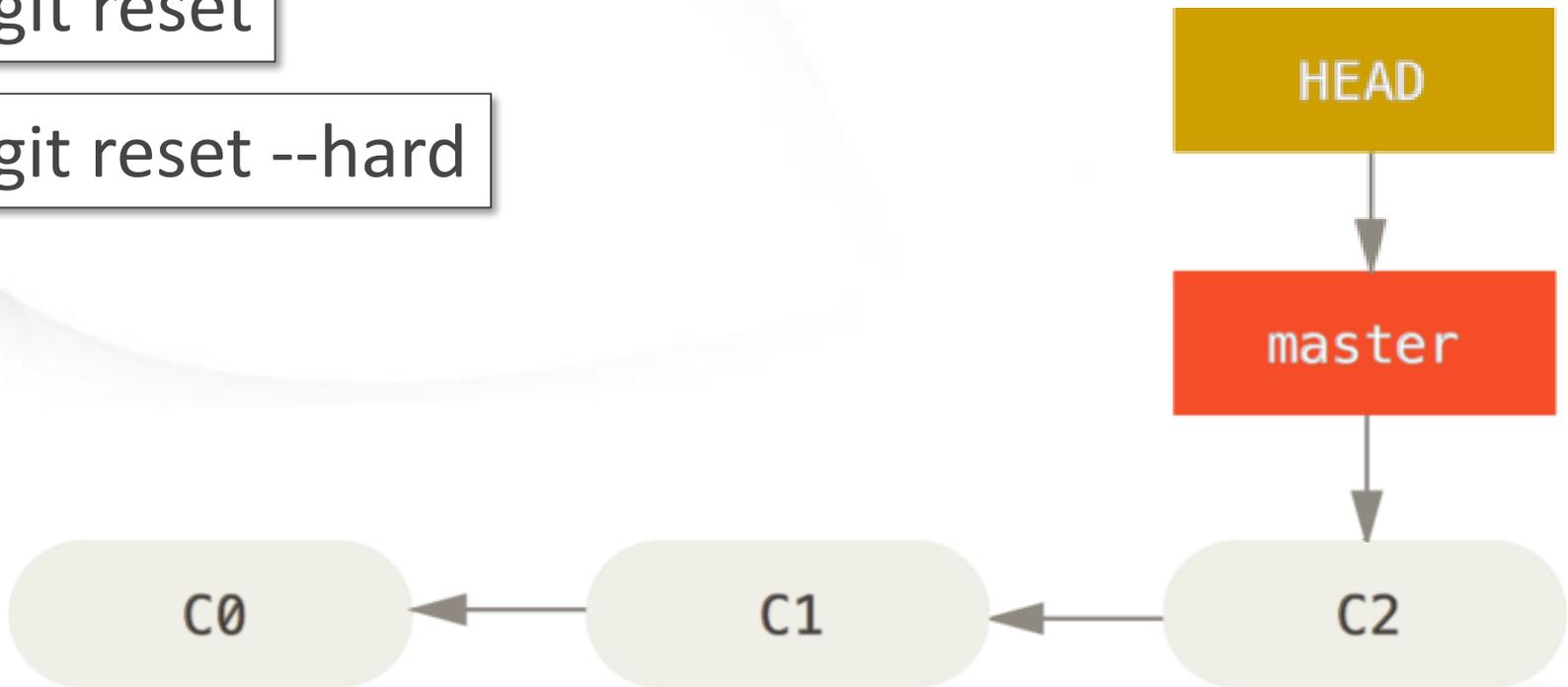




Reset - commit

git reset

git reset --hard

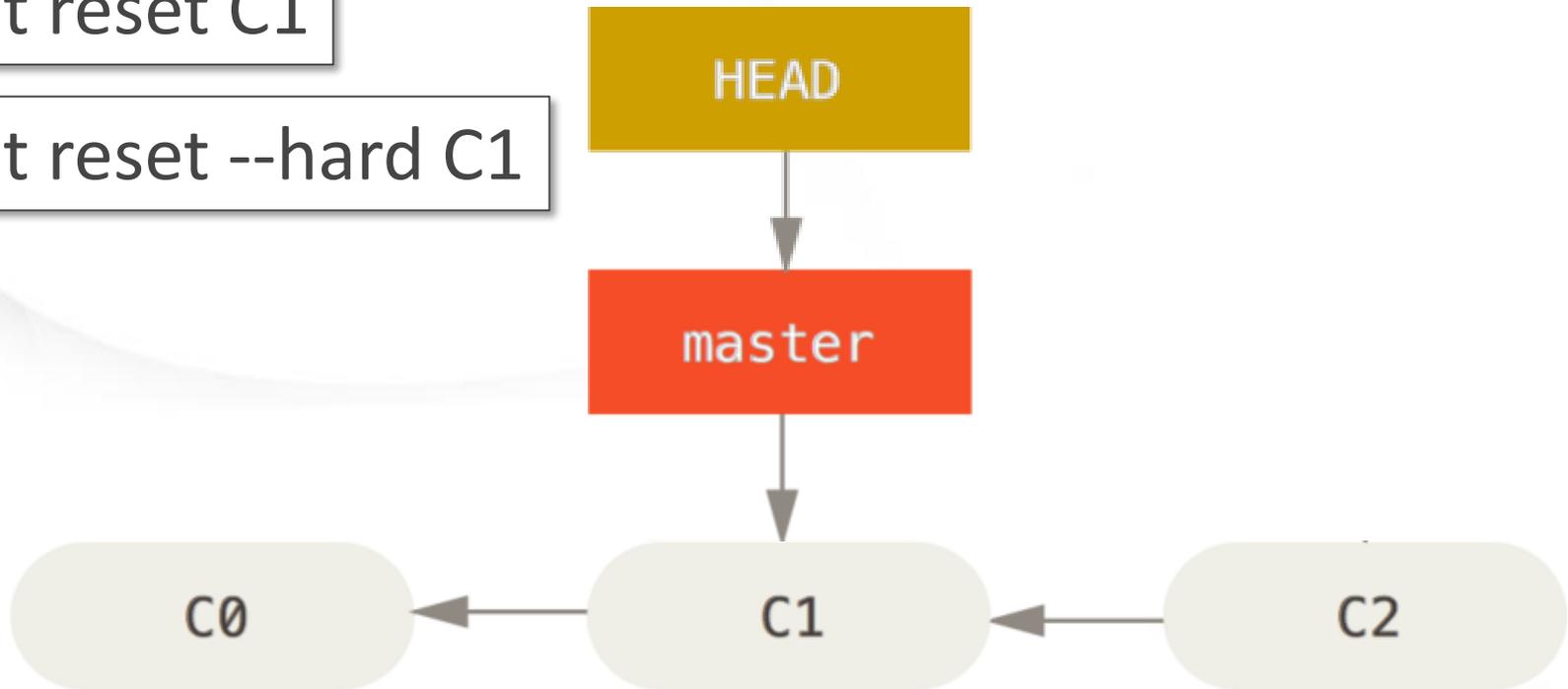




Reset - commit

```
git reset C1
```

```
git reset --hard C1
```

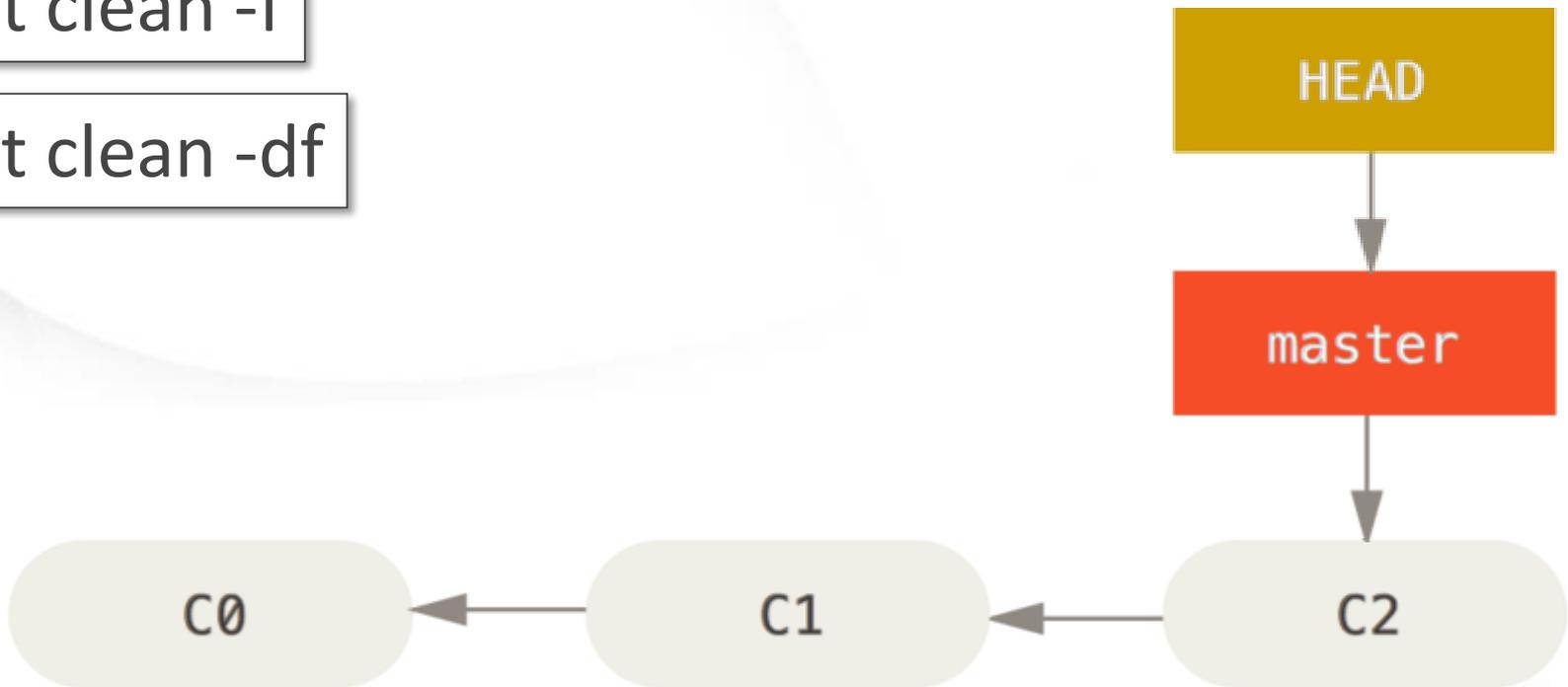




Clean

```
git clean -f
```

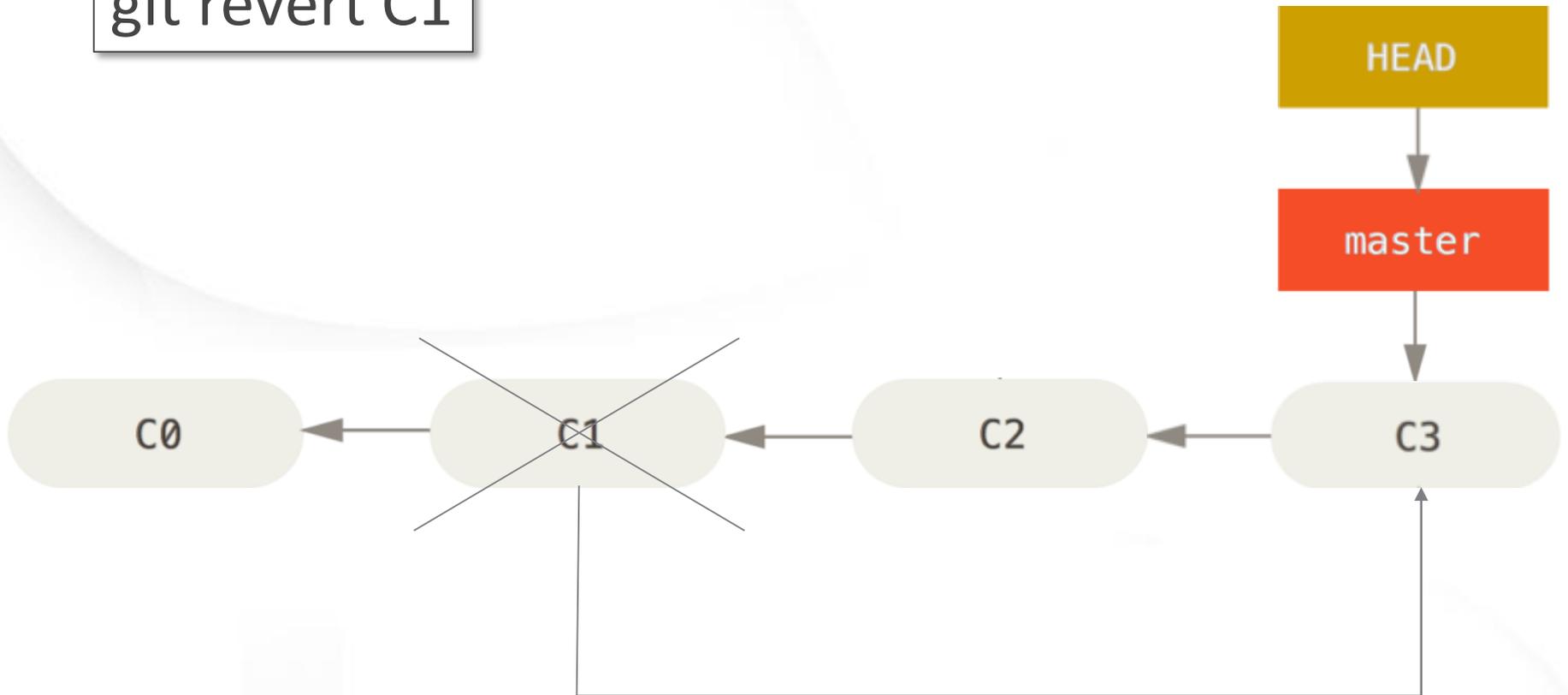
```
git clean -df
```





Revert

```
git revert C1
```



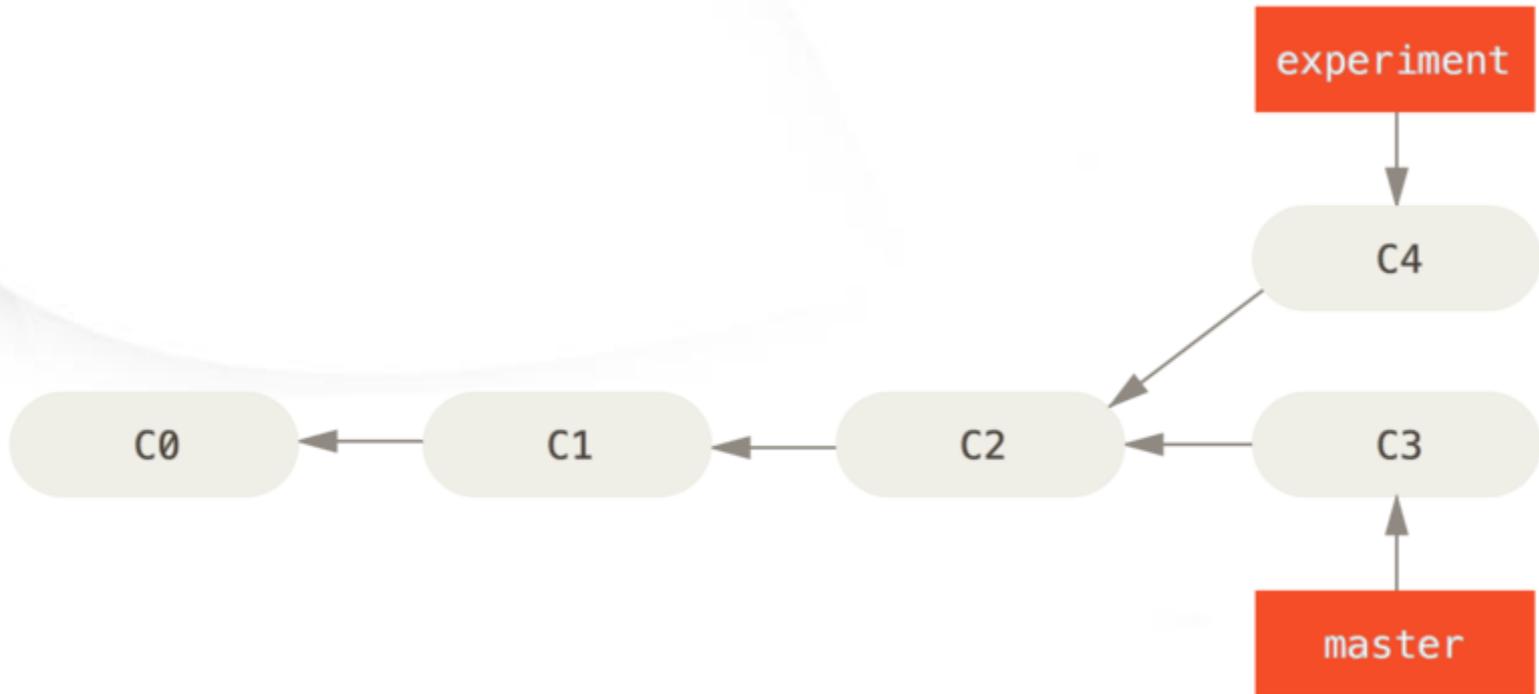
Reescrevendo a História



Laboratório de
Tecnologia da Informação
Aplicada



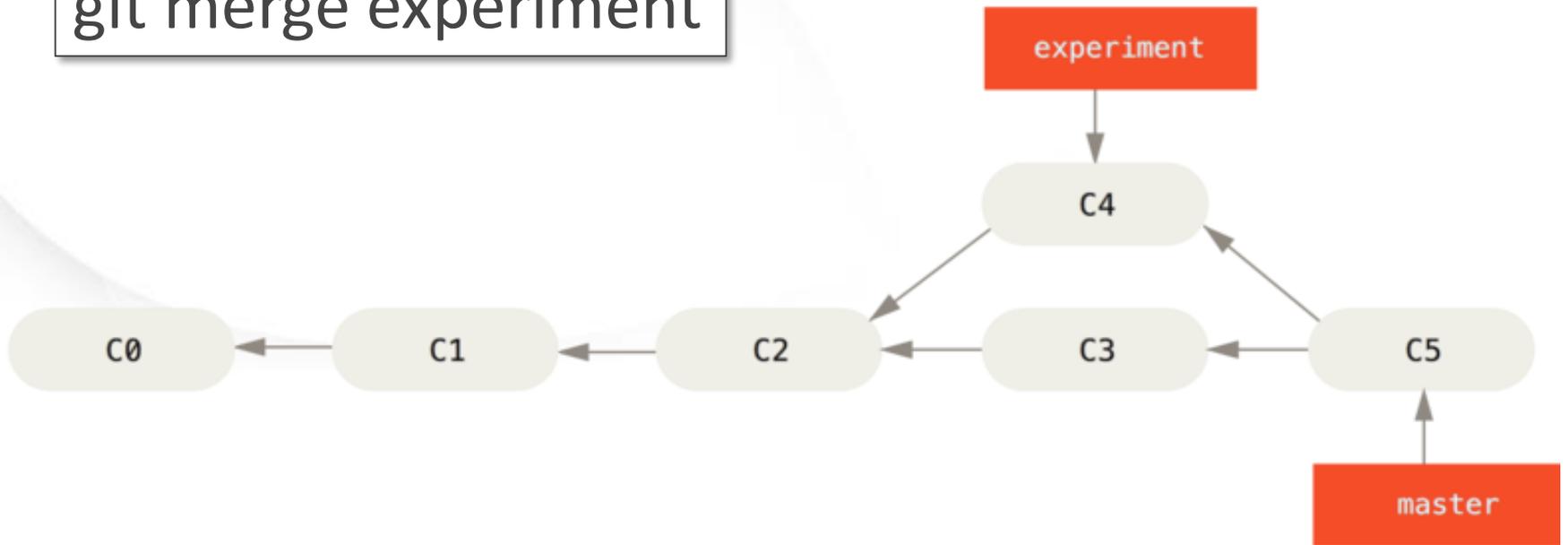
Rebase





Rebase

git merge experiment

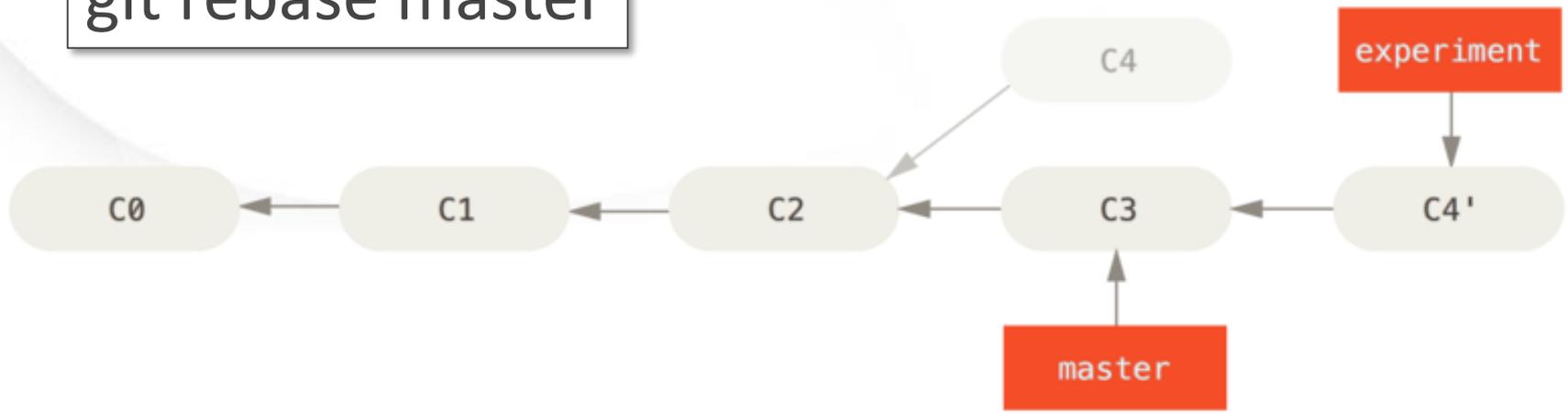




Rebase

git checkout experiment

git rebase master





Interactive Rebase

```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Rebase (master)
$ git log --oneline
09ae0f0 Alterei File 3
7b7ef05 Adicionei File 3 (não vazo)
d70276e Alterei File 2
bce5966 Alterei File 1
1d0ab2f Adicionei File 1 e 2 (vazios)
```

```
git rebase -i HEAD~4
```



Interactive Rebase

```
1 |pick bce5966 Alterei File 1
2 |pick d70276e Alterei File 2
3 |pick 7b7ef05 Adicionei File 3 (não vazo)
4 |pick 09ae0f0 Alterei File 3
5
6 # Rebase 1d0ab2f..09ae0f0 onto 1d0ab2f (4 command(s))
7 #
8 # Commands:
9 # p, pick = use commit
10 # r, reword = use commit, but edit the commit message
11 # e, edit = use commit, but stop for amending
12 # s, squash = use commit, but meld into previous commit
13 # f, fixup = like "squash", but discard this commit's log message
14 # x, exec = run command (the rest of the line) using shell
15 # d, drop = remove commit
16 #
17 # These lines can be re-ordered; they are executed from top to bottom.
18 #
19 # If you remove a line here THAT COMMIT WILL BE LOST.
20 #
21 # However, if you remove everything, the rebase will be aborted.
22 #
23 # Note that empty commits are commented out
```



Interactive Rebase

```
1 |edit d70276e Alterei File 2
2 |pick bce5966 Alterei File 1
3 |reword 7b7ef05 Adicionei File 3 (não vazio)
4 |squash 09ae0f0 Alterei File 3
5
6 # Rebase 1d0ab2f..09ae0f0 onto 1d0ab2f (4 command(s))
7 #
8 # Commands:
9 # p, pick = use commit
10 # r, reword = use commit, but edit the commit message
11 # e, edit = use commit, but stop for amending
12 # s, squash = use commit, but meld into previous commit
13 # f, fixup = like "squash", but discard this commit's log message
14 # x, exec = run command (the rest of the line) using shell
15 # d, drop = remove commit
16 #
17 # These lines can be re-ordered; they are executed from top to bottom.
18 #
19 # If you remove a line here THAT COMMIT WILL BE LOST.
20 #
21 # However, if you remove everything, the rebase will be aborted.
22 #
23 # Note that empty commits are commented out
```



Interactive Rebase

```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Rebase (master)
$ git rebase -i HEAD~4
Stopped at d70276ec9af276040e6b093179cf8d21de4ebc3a... Alterei File 2
You can amend the commit now, with

    git commit --amend

Once you are satisfied with your changes, run

    git rebase --continue
```



Interactive Rebase

```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Rebase (master|REBASE-i 1/4)
$ git status
interactive rebase in progress; onto 1d0ab2f
Last command done (1 command done):
  edit d70276e Alterei File 2
Next commands to do (3 remaining commands):
  pick bce5966 Alterei File 1
  reword 7b7ef05 Adicionei File 3 (n3o vaziao)
  (use "git rebase --edit-todo" to view and edit)
You are currently editing a commit while rebasing branch 'master' on '1d0
ab2f'.
  (use "git commit --amend" to amend the current commit)
  (use "git rebase --continue" once you are satisfied with your changes)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working director
y)

        modified:   File2.txt

no changes added to commit (use "git add" and/or "git commit -a")
```



Interactive Rebase

```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Rebase (master|REBASE-i 1/4)
$ git commit -a --amend
[detached HEAD 6420681] Alterei File 2 duas vezes
Date: Thu Jan 28 03:08:39 2016 -0200
1 file changed, 1 insertion(+)
```

```
1 Alterei File 2 duas vezes|
2
3 # Please enter the commit message for your changes. Lines starting
4 # with '#' will be ignored, and an empty message aborts the commit.
5 #
6 # Date:      Thu Jan 28 03:08:39 2016 -0200
7 #
8 # interactive rebase in progress; onto 1d0ab2f
9 # Last command done (1 command done):
10 #   edit d70276e Alterei File 2
11 # Next commands to do (3 remaining commands):
12 #   pick bce5966 Alterei File 1
13 #   reword 7b7ef05 Adicionei File 3 (não vazio)
14 # You are currently editing a commit while rebasing branch 'master' on '1d0ab2f'.
15 #
16 # Changes to be committed:
17 #   modified:   File2.txt
18 #
```



Interactive Rebase

```
git rebase --continue
```

```
1 |Adicionei File 3 (não vazio)
2
3 # Please enter the commit message for your changes. Lines starting
4 # with '#' will be ignored, and an empty message aborts the commit.
5 #
6 # Date:      Thu Jan 28 03:09:27 2016 -0200
7 #
8 # interactive rebase in progress; onto 1d0ab2f
9 # Last commands done (3 commands done):
10 #   pick bce5966 Alterei File 1
11 #   reword 7b7ef05 Adicionei File 3 (não vazio)
12 # Next command to do (1 remaining command):
13 #   squash 09ae0f0 Alterei File 3
14 # You are currently editing a commit while rebasing branch 'master' on '1d0ab2f'.
15 #
16 # Changes to be committed:
17 #   new file:   File3.txt
18 #
```



Interactive Rebase

```
1 # This is a combination of 2 commits.
2 # The first commit's message is:
3
4 Adicionei File 3 (não vazio)
5
6 # This is the 2nd commit message:
7
8 Alterei File 3
9
10 # Please enter the commit message for your changes. Lines starting
11 # with '#' will be ignored, and an empty message aborts the commit.
12 #
13 # Date:      Thu Jan 28 03:09:27 2016 -0200
14 #
15 # interactive rebase in progress; onto 1d0ab2f
16 # Last commands done (4 commands done):
17 #   reword 7b7ef05 Adicionei File 3 (não vazio)
18 #   squash 09ae0f0 Alterei File 3
19 # No commands remaining.
20 # You are currently editing a commit while rebasing branch 'master' on '1d0ab2f'.
21 #
22 # Changes to be committed:
23 #   new file:   File3.txt
24 #
```



Interactive Rebase

```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Rebase (master|REBASE-i 1/4)
$ git rebase --continue
[detached HEAD 9fbca85] Adicionei File 3 (não vazio)
Date: Thu Jan 28 03:09:27 2016 -0200
1 file changed, 1 insertion(+)
create mode 100644 File3.txt
[detached HEAD af20506] Adicionei File 3 (não vazio)
Date: Thu Jan 28 03:09:27 2016 -0200
1 file changed, 1 insertion(+)
create mode 100644 File3.txt
Successfully rebased and updated refs/heads/master.
```

```
Marcelo@GS70-MARCELO MINGW64 ~/Desktop/Rebase (master)
$ git log --oneline
af20506 Adicionei File 3 (não vazio)
258ff28 Alterei File 1
6420681 Alterei File 2 duas vezes
1d0ab2f Adicionei File 1 e 2 (vazios)
```

Projetos OpenSource



Laboratório de
Tecnologia da Informação
Aplicada



- Fundado em 2008;
- Sede em São Francisco, Califórnia, EUA;
- 415 funcionários;
- Mais de 10 milhões de usuários.



Fork

Se você quiser contribuir para um projeto existente para o qual você não tem permissão de *push*, GitHub incentiva a utilização de *forks* do projeto.

Um *fork* significa que o GitHub fará uma cópia do projeto que é inteiramente sua: ficará hospedado na sua conta de usuário.

Pro Git, Scott Chacon e Ben Straub



Pull requests

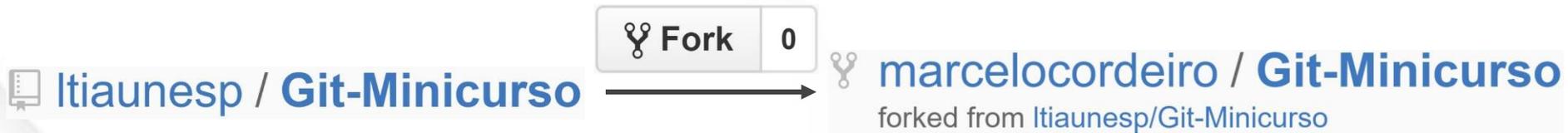
Através do *pull request* os mantenedores do projeto não precisam se preocupar com a adição de usuários como colaboradores para dar-lhes acesso de escrita.

As pessoas podem criar um *fork* de um projeto e fazer um *push* nele, e o mantenedor do projeto principal pode fazer um *pull* dessas mudanças, adicionando-as como remotos e fazendo um merge no seu projeto.

Pro Git, Scott Chacon e Ben Straub



Exemplo



New pull request

- git request-pull destino origem



Exemplo

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).

base fork: **Itiaunesp/Git-Minicurso** ▼ base: **master** ▼ ... head fork: **marcelocordeiro/Git-Minic...** ▼ compare: **master** ▼

✗ Can't automatically merge. Don't worry, you can still create the pull request.

Create pull request Discuss and review the changes in this comparison with others. 

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).

base fork: **Itiaunesp/Git-Minicurso** ▼ base: **master** ▼ ... head fork: **marcelocordeiro/Git-Minic...** ▼ compare: **master** ▼

✓ Able to merge. These branches can be automatically merged.

Create pull request Discuss and review the changes in this comparison with others. 



Hands on!

<https://github.com/ltiaunesp/Git-Minicurso>



DUVIDAS???

Obrigado!

Marcelo Augusto Cordeiro

marcelo.augusto.cordeiro@gmail.com



Laboratório de
Tecnologia da Informação
Aplicada

CENTRO DE
PESQUISA
CATI
SEPIN/MCT