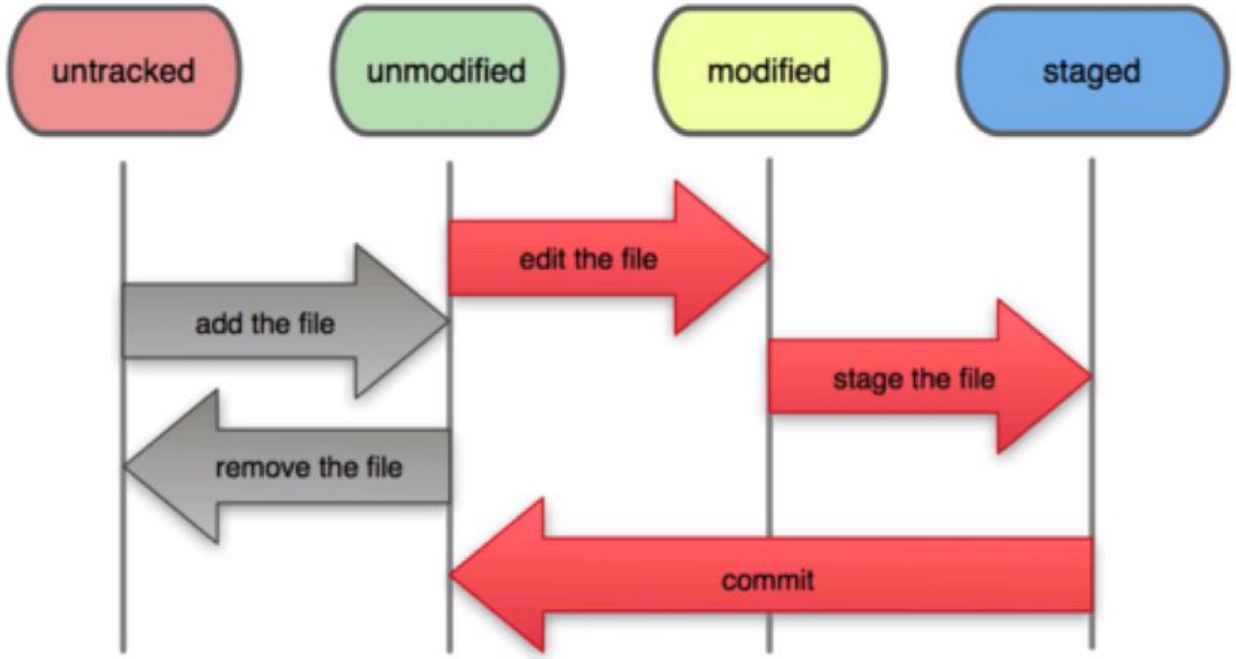
UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL

Gerência de Configuração de Software

Aula 08 – Git (Parte 2)

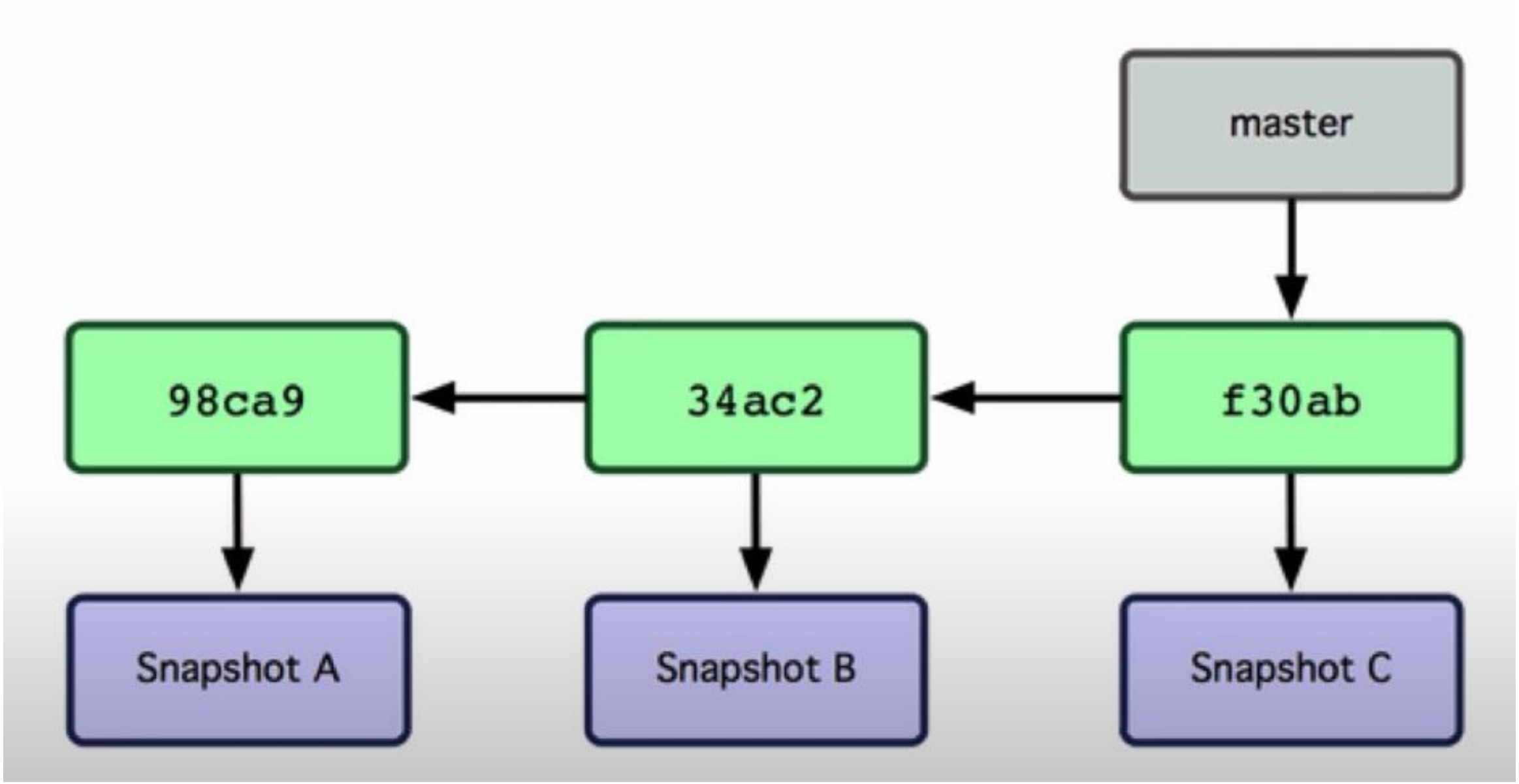
Prof. Dr. Awdren de Lima Fontão
awdren.fontao@ufms.br

Ciclo de vida do status dos arquivos

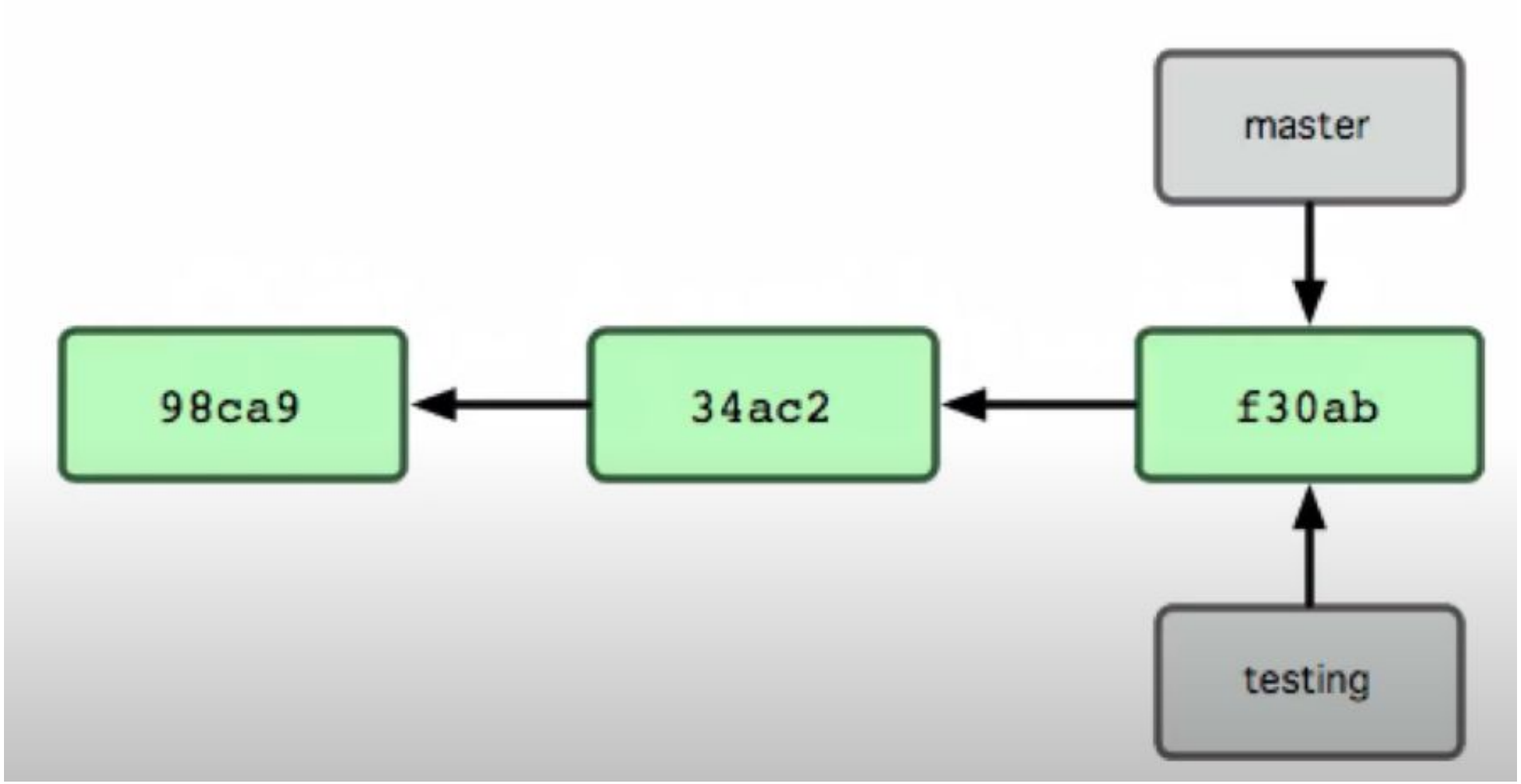


O que é um branch e por que usar?

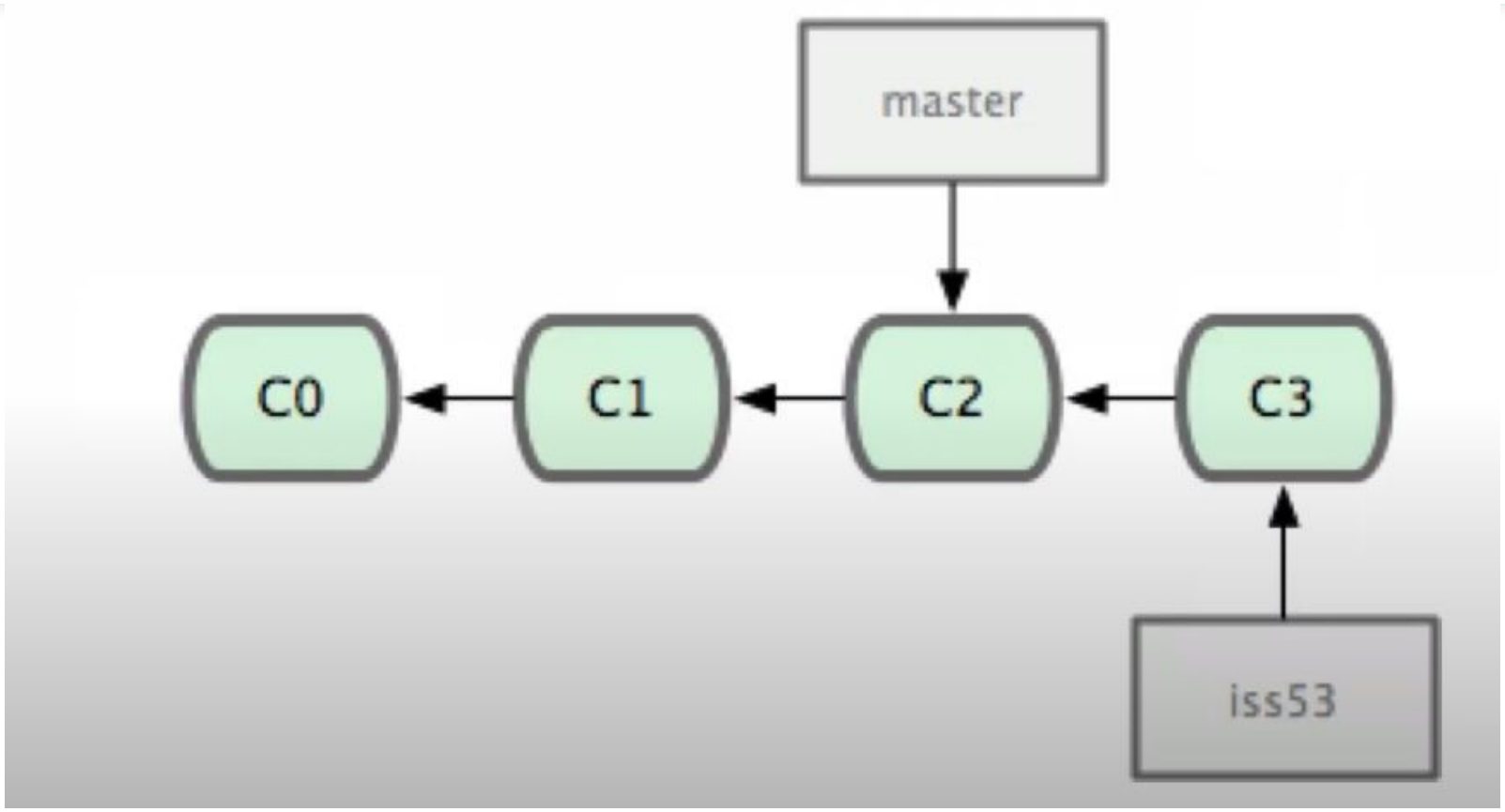
- Um ponteiro móvel pra um commit


master
98ca9
34ac2
f30ab
Snapshot A
Snapshot B
Snapshot C

Branchs separados



Branchs em diferentes commits



Vantagens

- Modificar os ICs sem alterar a master;
- “Desligar” facilmente;
- Permite várias pessoas trabalhando ao mesmo tempo;
- Evitar conflitos.

Criando um branch

git branch testing

git checkout –b testing

```
$ git checkout -b testing
Switched to a new branch 'testing'
```

Depois verifique: git branch

```
$ git branch
  master
* testing
```

Movendo entre branch

```
$ git checkout testing
Switched to branch 'testing'
```

```
$ git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.
```

Apagando uma branch

```
$ git branch -D testing
Deleted branch testing (was a30ba6c).
```

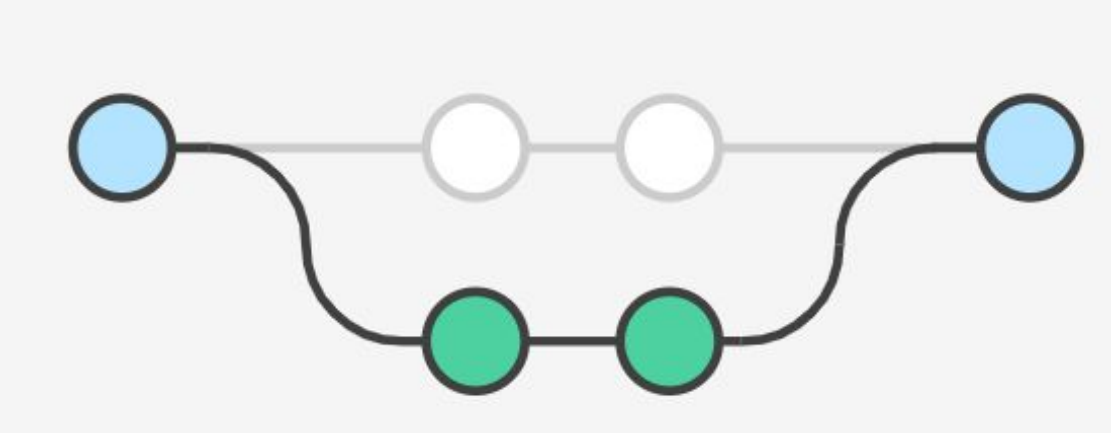
```
$ git branch
* master
```

Renomeando uma branch

```
$ git branch -m OLD-BRANCH-NAME NEW-BRANCH-NAME
$ git fetch origin
$ git branch -u origin/NEW-BRANCH-NAME NEW-BRANCH-NAME
$ git remote set-head origin -a
```

Unindo branches

- Merge
- Rebase

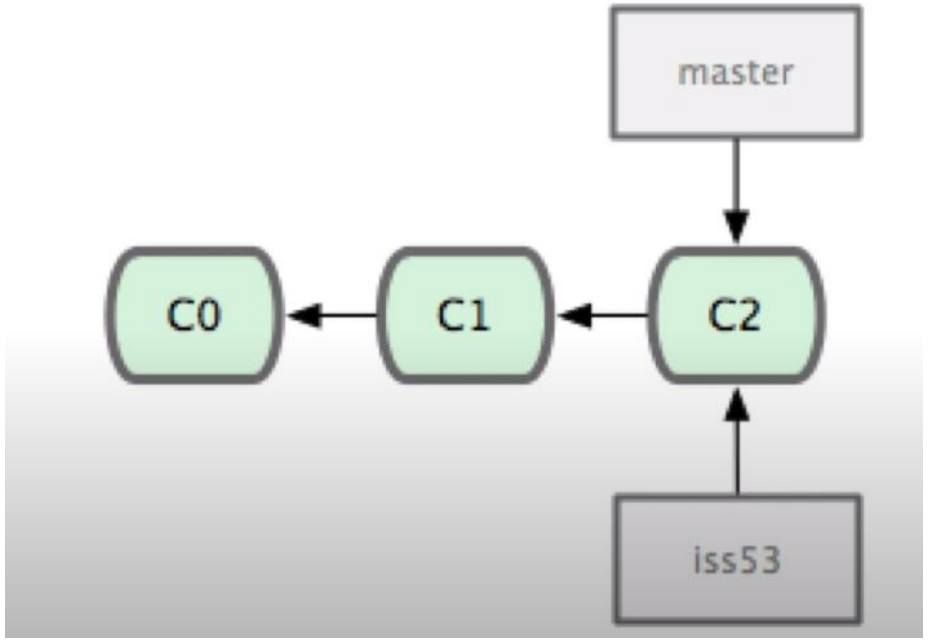


A primeira coisa a ser entendida sobre o git rebase é que resolve o mesmo problema que o git merge. Ambos os comandos são desenvolvidos para integrar alterações de uma ramificação para outra—eles só fazem isto de formas muito diferentes.

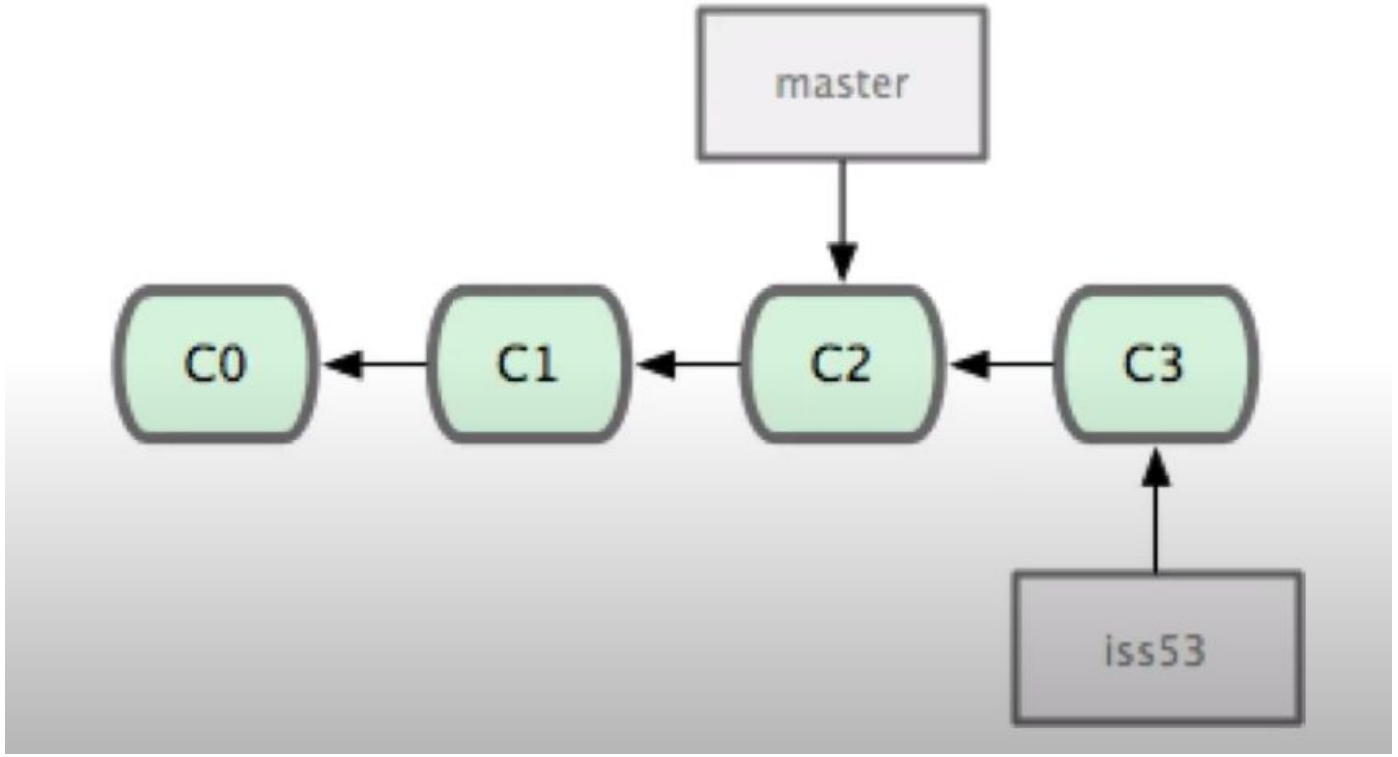
Merge

- Serve pra mesclar duas branchs
- Caso aconteça algum conflito, você precisará resolver manualmente

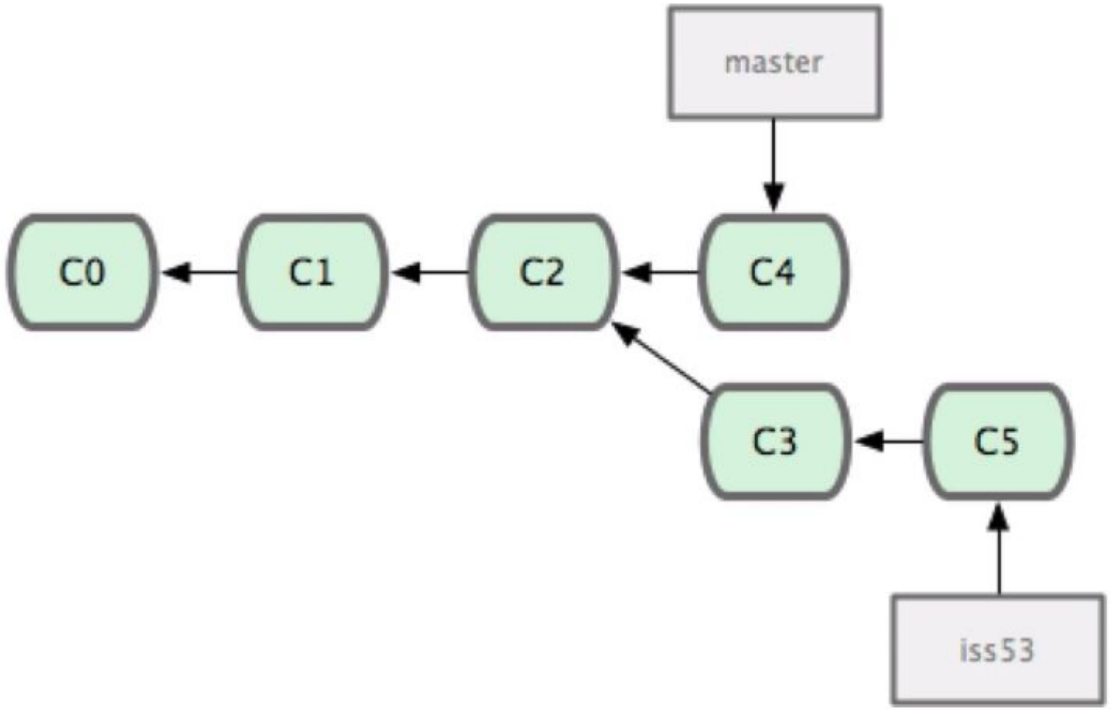
Estado inicial



Um commit na nova branch



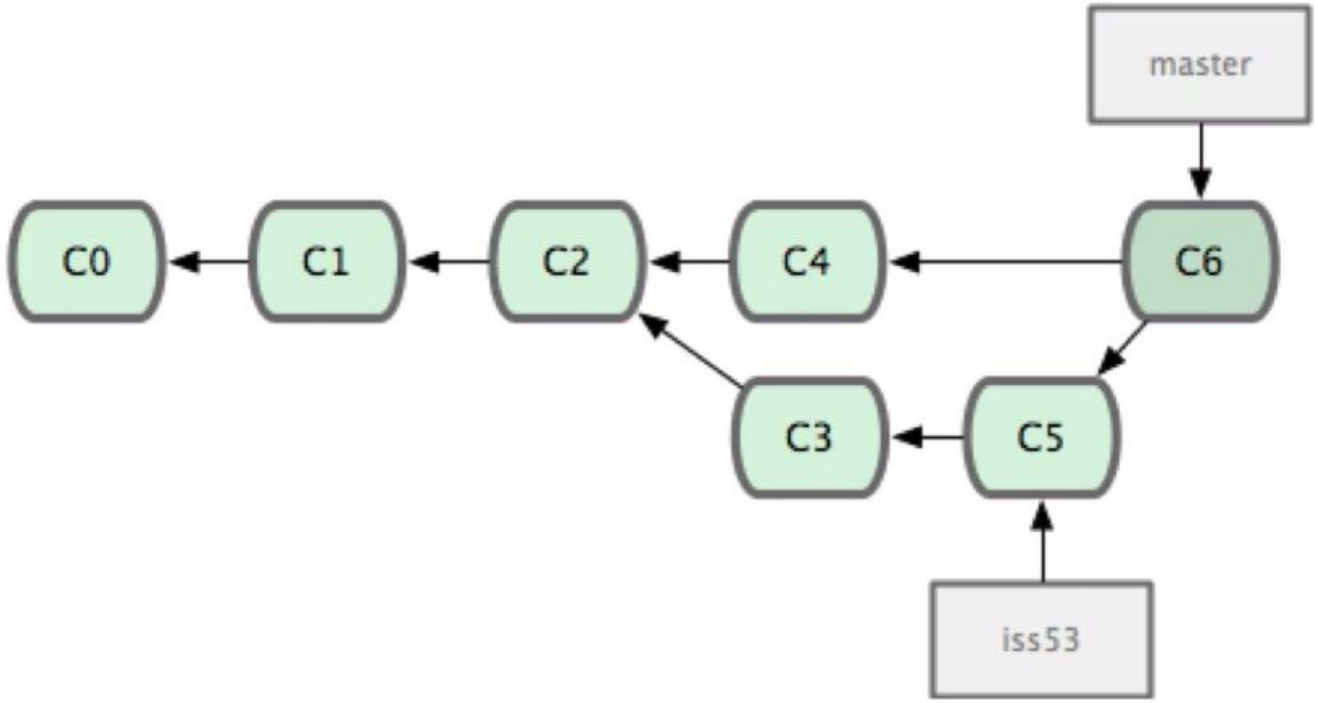
Um commit pelo branch iss53



Fazendo o merge

```
git checkout master
git merge iss53

git merge master iss53
```



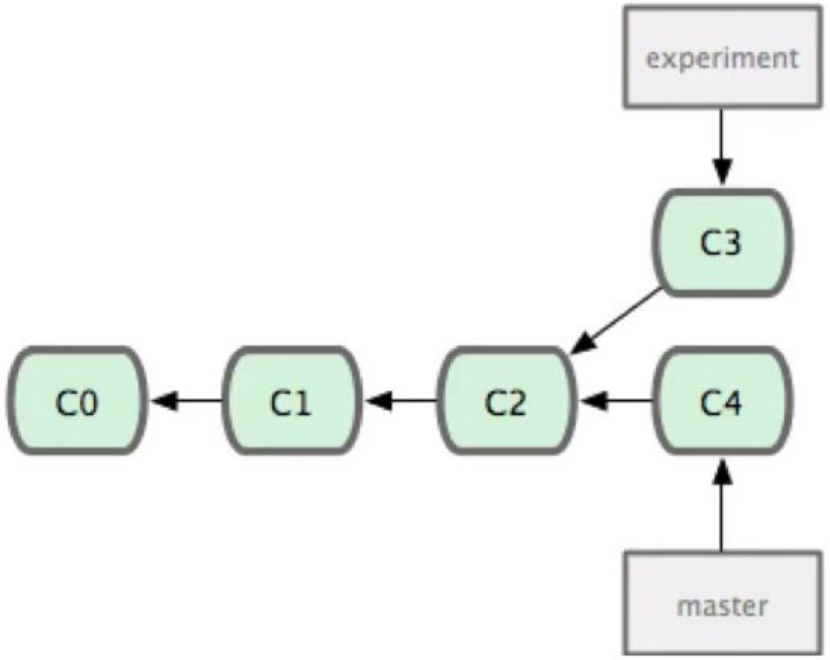
Vantagens e desvantagens

- Vantagens
  - Operação que não destroi commits
  - Não modifica o histórico
- Desvantagens
  - Precisa de um commit extra
  - Histórico poluído – “forma diamante”, árvore com muitas linhas

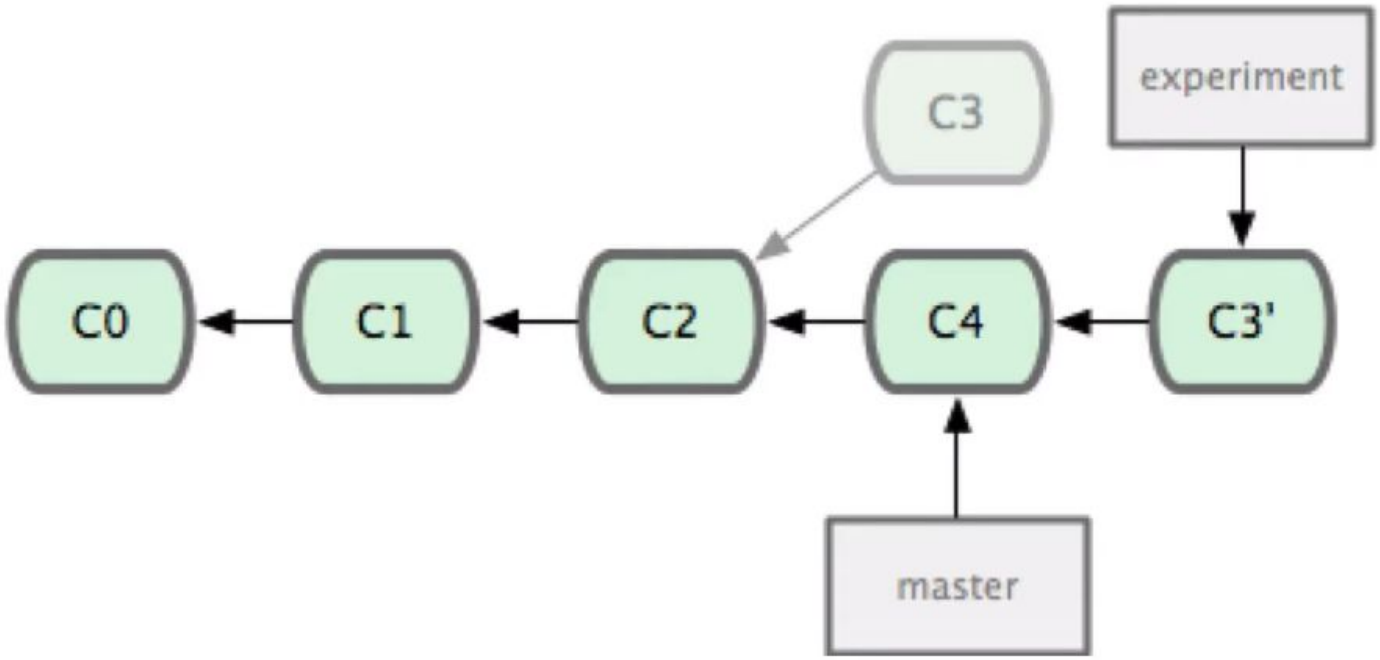
Rebase

- Na prática, precisamos dele quando estamos trabalhando na branch e sabemos que o repositório master foi atualizado por outros desenvolvedores e precisamos ter nosso branch atualizado, concorda?
- Não é nada legal ficarmos trabalhando na branch com o código de 2 dias atrás.
  - Podemos ter problemas e sérios conflitos aqui na hora do merge, o melhor é que se for ter conflito que seja pequeno e de imediato.

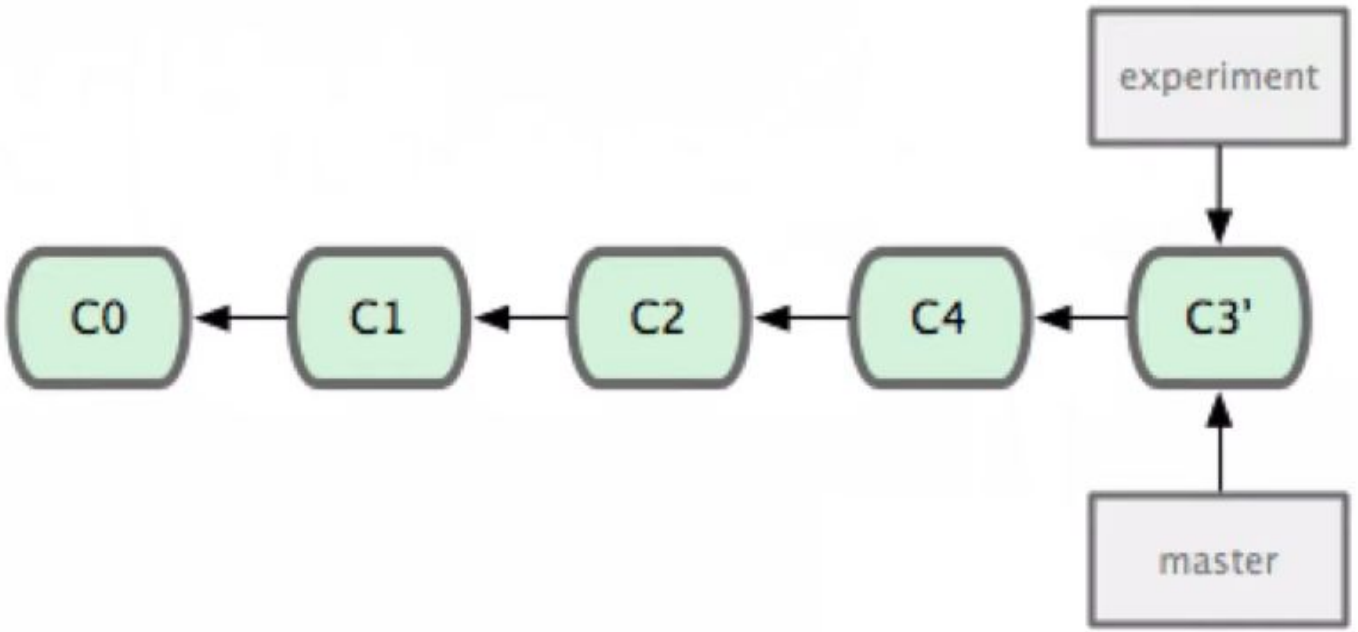
Estado inicial

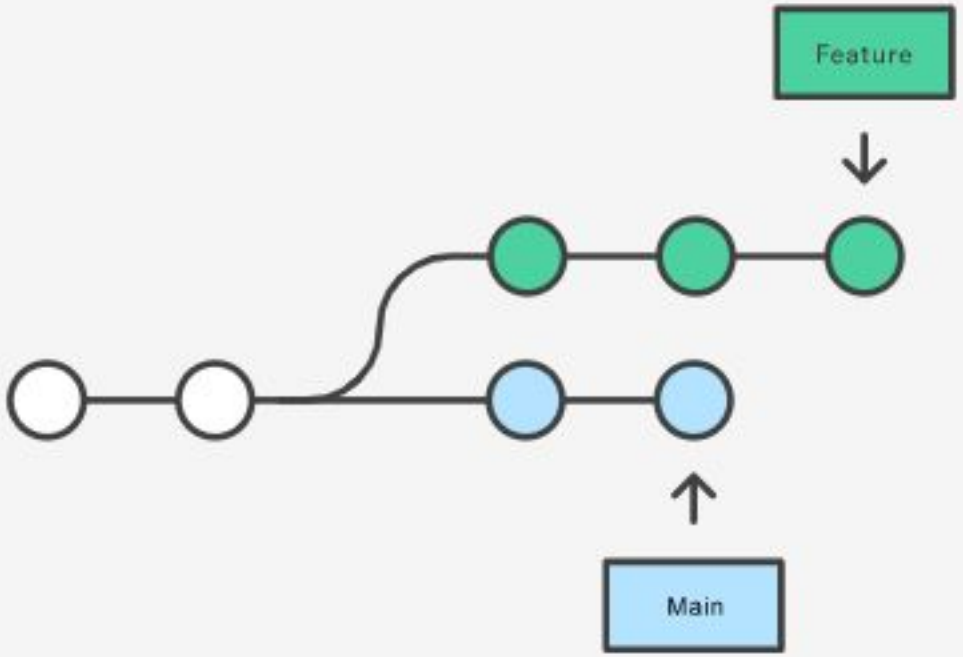


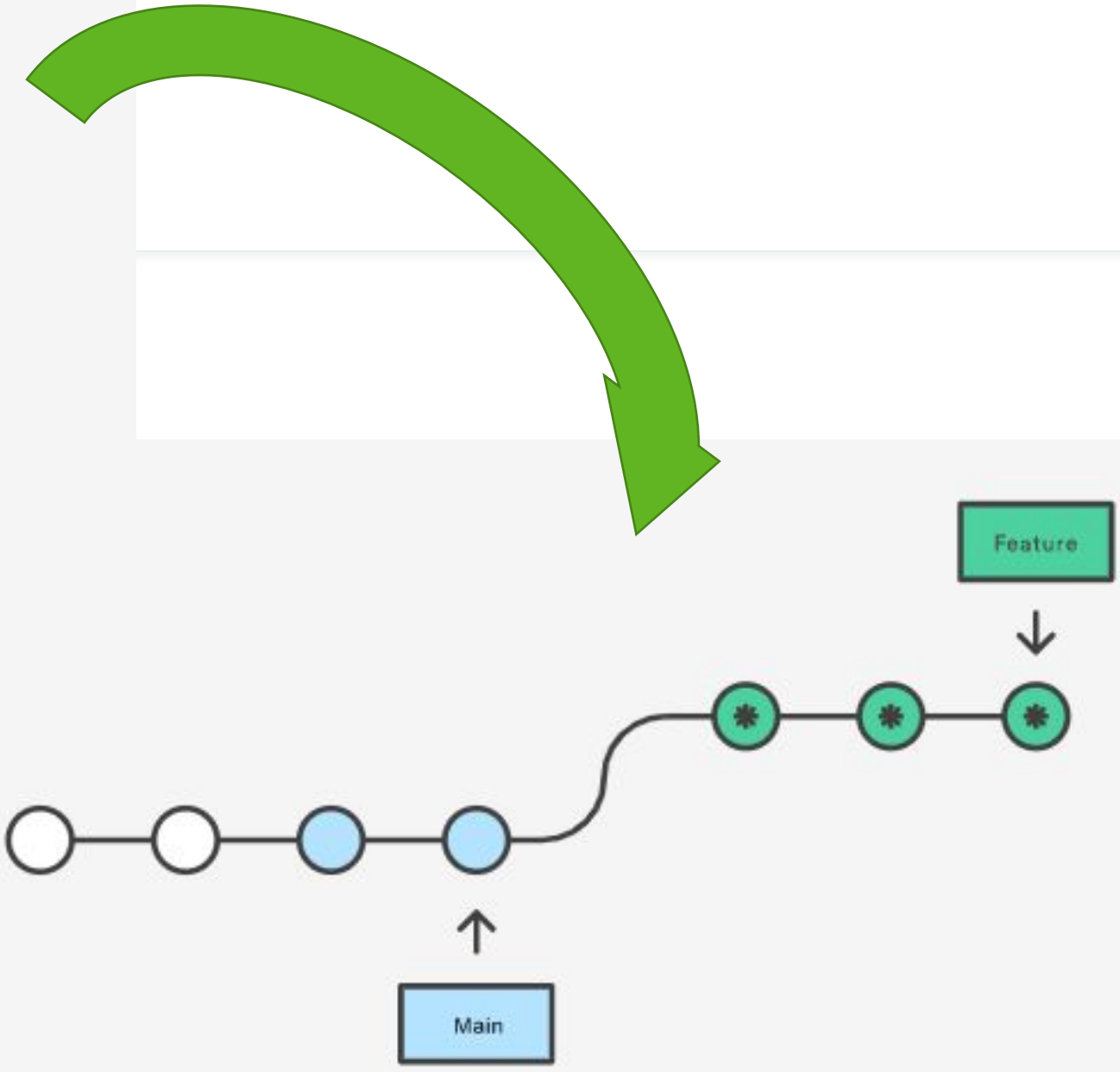
Durante o processo



Final do rebase







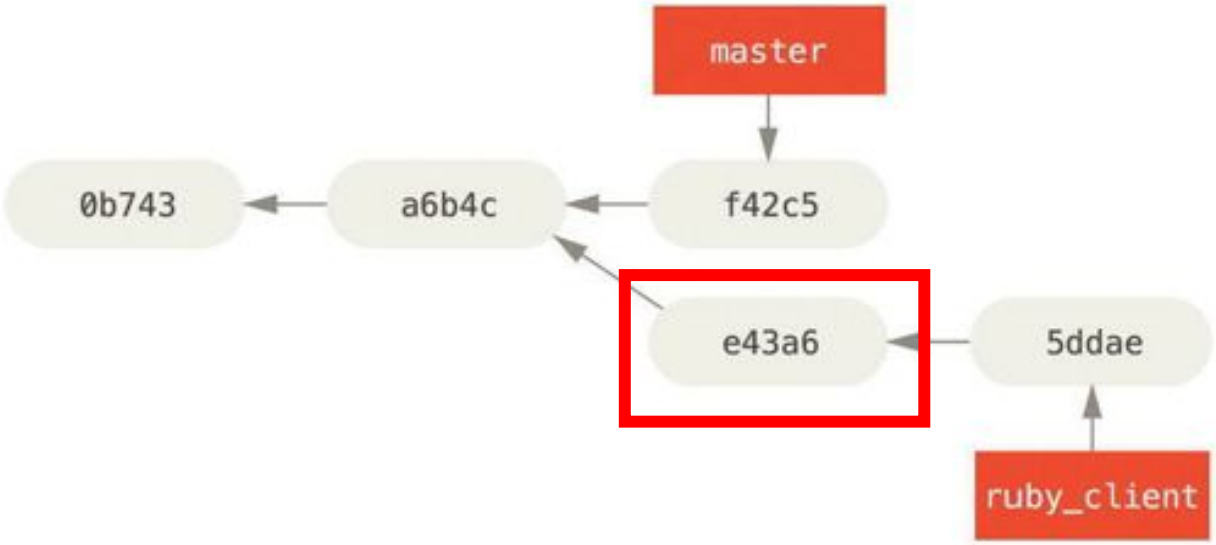

```
git checkout feature
git rebase main
```

rebase

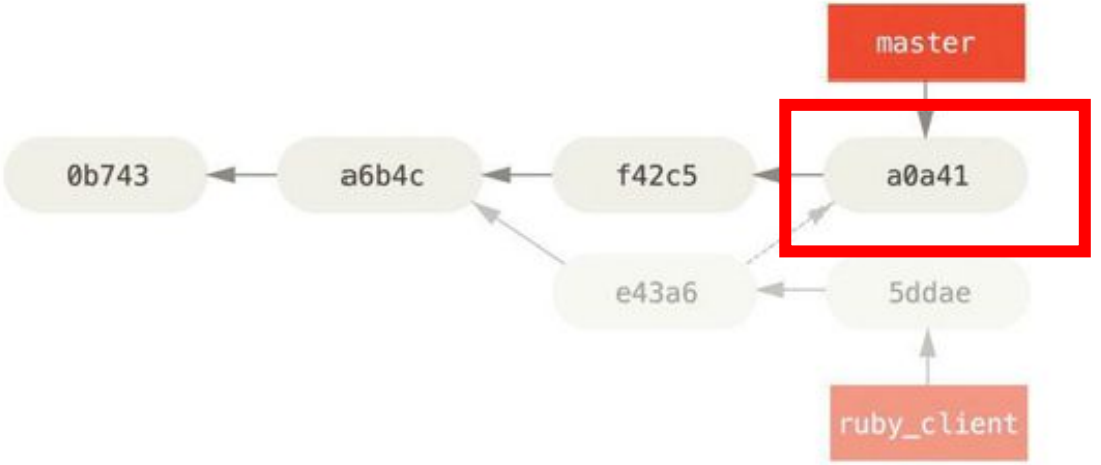
- Vantagens
  - Evita commit extra
  - Histórico linear
- Desvantagens
  - Perda da ordem cronológica – muda o histórico
  - git pull --rebase

git cherry-pick

- O comando git cherry-pick é usado para pegar a alteração introduzida em um único commit do Git e tentar reintroduzi-lo como um novo commit no branch que você está atualmente.
- Isso pode ser útil para obter apenas um ou dois commits de uma ramificação individualmente, em vez de mesclar na ramificação que recebe todas as alterações.



git cherry-pick e43a6



Pode remover a branch e os commits que você não quer deixar disponíveis para puxar.

git revert

- Uma grande feature e subi pra produção :S
  - Quebrou a entrega
- Preciso reverter e não “resetar” pra poder entender melhor.
- O comando git revert é essencialmente um git cherry-pick reverso. Ele cria um novo commit que aplica exatamente o oposto da alteração introduzida no commit que você está direcionando, essencialmente desfazendo-o ou revertendo-o.

git log

git show <identificador do commit>

git revert <identificador do commit>

git stash

- Rascunho
- O comando git stash é usado para armazenar temporariamente o trabalho não commitado para limpar seu diretório de trabalho sem ter que commitar o trabalho finalizado em uma ramificação
  - Por exemplo, quero mudar de branch…

- git stash
- git stash apply
- git stash list
- git stash clear

.gitignore

- Especifica ICs para não serem rastreados intencionalmente;
  - Por exemplo, senhas, arquivos que o próprio sistema gera (. DSTORE no MAC)
- É um arquivo no diretório, com padrões que queremos que não sejam considerados.
- https://github.com/github/gitignore

Esqueci de incluir arquivos
git commit --amend

- git commit -m ‘initial commit’
- git add forgotten_file
- git commit --amend

Unstaging um IC

- git add
- git status

- git reset HEAD <file>

Desfazendo a alteração em um arquivo modificado

- git status
- git checkout -- <file>

Depurando

- O Git tem alguns comandos que são usados para ajudar a depurar um problema em seu código. Isso varia desde descobrir onde algo foi introduzido até descobrir quem o introduziu.

git shortlog

```
awdrenfontao@MacBook-Pro-de-Awdren awdren.github.io % git shortlog
Awdren Fontão (5):
      Adding files
      Adding information about journal article
      Changing title
      Updating informations
      Update index.html
```

- O comando git shortlog é usado para resumir a saída do git log. Serão necessárias muitas das mesmas opções que o comando git log, mas em vez de listar todos os commits, ele apresentará um resumo dos commits agrupados por autor.

Exercício 1

- Configure um colaborador para seu repositório;
- Essa pessoa deve clonar o repositório e fazer um commit nele;
- Atualize seu repositório com o novo commit
- Invertam os papeis

Exercício 2

- Crie um conflito no seu repositório
- Seu colaborador deve editar o mesmo arquivo que você
- Você deve fazer o pull, resolver o conflito e fazer o push das alterações
- Seu colaborador deve ver que a sua alteração foi adicionada junto à alteração dele.
- Invertam os papeis.

Exercício 3 - rebase

Faça um commit na master/main;

Faça um commit numa outra branch;

Facá o rebase da branch com a master;

Veja a ordem dos commits;

*Mesmo que tenha conflitos o histórico é preservado.