

5ª EDIÇÃO

Git

Sistema de controle
de versão

Luiza
<CODE>

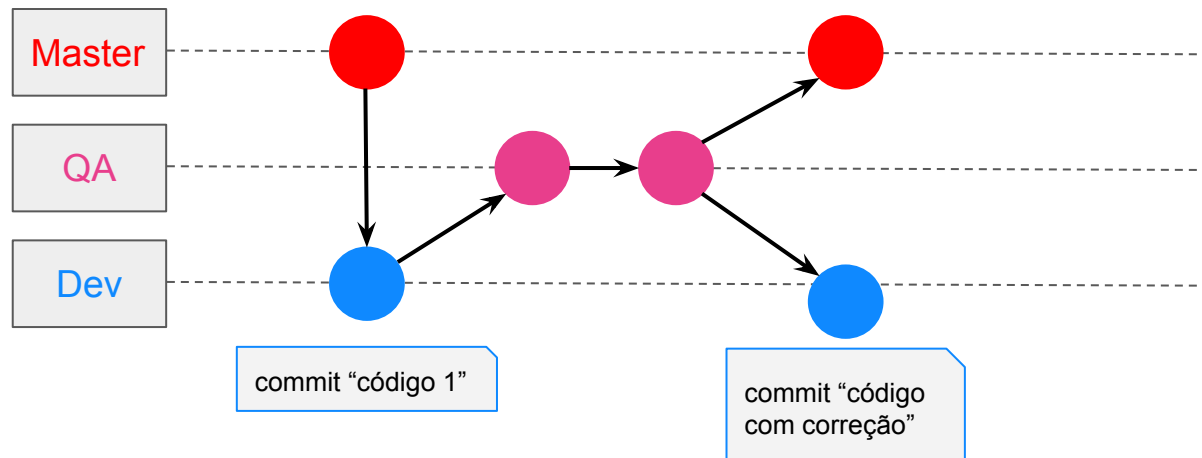
Luizalabs
magalu

6. Branch

6. Branch

6.1 - O que é branch no git?

Uma **ramificação** no **git** é um **ponteiro** para as **alterações** feitas nos **arquivos** do projeto. É útil em situações nas quais você deseja adicionar um novo recurso ou corrigir um erro, gerando uma nova **ramificação** garantindo que o **código instável** não seja **mesclado nos arquivos do projeto principal**. Depois de concluir a atualização dos códigos da ramificação, você pode mesclar a ramificação com a principal, geralmente chamada de master.



6. Branch

6.2 - Git Branch

O comando **git branch** permite **criar, listar, renomear e excluir ramificações**. Ele não permite alternar entre as ramificações ou reunir um histórico bifurcado de novo.

6.3 - Comandos Git Branch

git branch - Listar todas as ramificações no seu repositório. Isso é sinônimo de git branch --list.

git branch <nome-branch> - Criar uma nova ramificação chamada .

git branch -d <nome-branch> - Excluir a ramificação especificada. Esta é uma operação “segura” em que o Git impede que você exclua a ramificação se tiver mudanças não mescladas.

git branch -D <branch> - Forçar a exclusão da ramificação especificada, mesmo que ela tenha mudanças não mescladas.

git branch -m <branch> - Renomear a ramificação atual para .

6. Branch

6.4 - O que é git checkout?

O **git checkout** é uma forma de **alternar** entre **versões** de **arquivos**, **commits** ou **branches**. Ele tem diferentes formas de usar, mas seus dois principais usos são: trocar de branch ou restaurar arquivos.

Quando você está em um branch e quer trocar para outro você pode usá-lo. Um outro uso, é quando você precisa voltar para uma versão específica de um arquivo ou restaurar apenas aquele arquivo para um commit específico.

6.3 - Comandos Git Checkout

git checkout <nome-do-branch> - Altera a branch que estará no seu *Working Directory*.

git checkout -b <nome-do-branch> - Esse comando, além de criar, já altera para o branch criado. Ele é uma versão resumida de criar e depois trocar de branch, como a sequência de comandos abaixo:

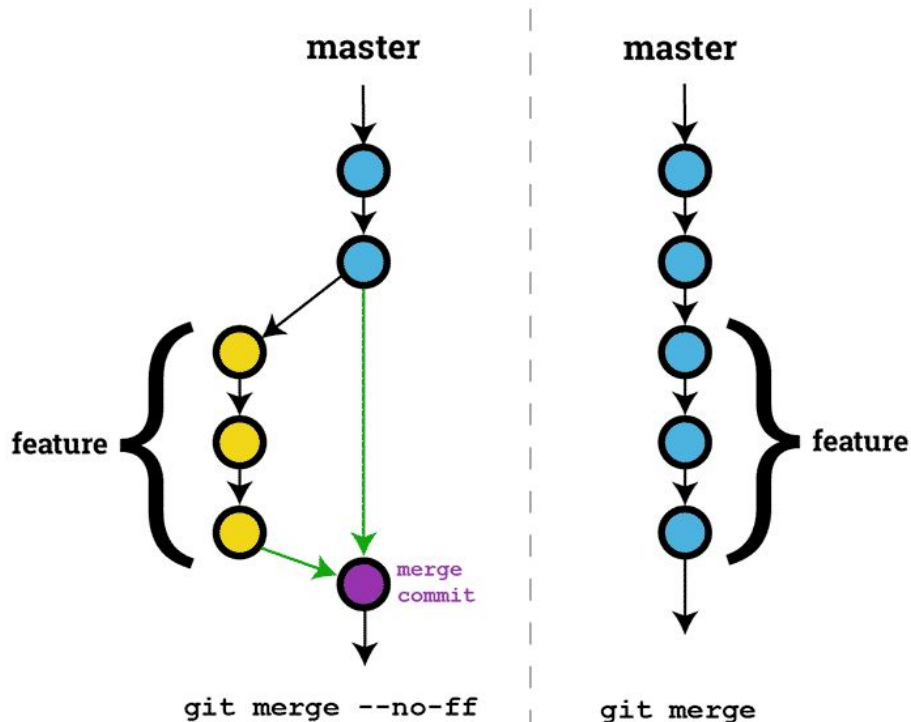
git branch <nome-do-branch>

git checkout <nome-do-branch>

7. Merge

7. Merge

Outra maneira de integrar mudanças em branches e resolver divergências

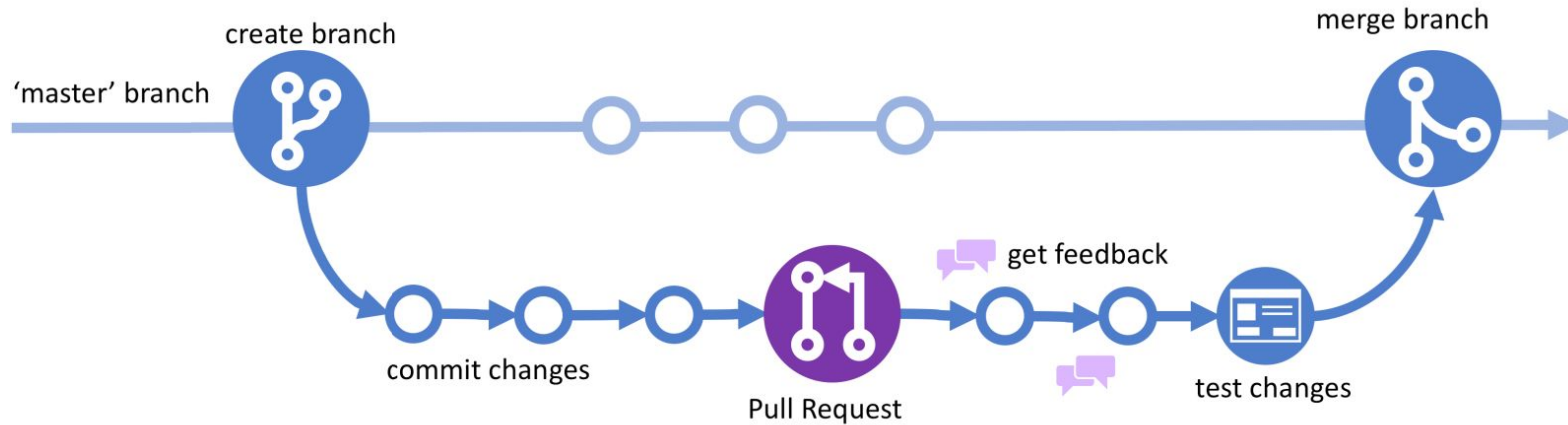


<https://www.hostingadvice.com/wp-content/uploads/2014/12/git-merge1.gif>

8. Pull Request

8. Pull Request

GitHub Flow



8. Pull Request


← ↻ 🏠 🔒 <https://github.com/luizalabs/tutorial-python-brasil/pull/15>

<> Code • Issues 🔗 **Pull requests** ▶ Actions 📁 Projects 📖 Wiki 🛡 Security 📈 Insights

Fix assincrono.md, avancadas.md e breaker.md #15




Merged cassiobotaro merged 3 commits into `luizalabs:main` from `vitorStein:main` on 24 Oct 2021


💬 Conversation 1 🔗 Commits 3 📄 Checks 0 📄 Files changed 3

 **vitorStein** commented on 24 Oct 2021 Contributor 😊 ...

No description provided.

📄 **vitorStein** added 3 commits 9 months ago

- 🔗  Fix assincrono.md 4c0fd39
- 🔗  Fix avancadas.md cd37ec0
- 🔗  Fiz breaker.md 9e4503f

🔗  cassiobotaro merged commit 957d01f into `luizalabs:main` on 24 Oct 2021

9. Conflitos

9. Conflitos

```
index.html — carparts-website_conflict (git: main)
13 <div id="navigation">
14 <ul>
15 <<<<<<< HEAD
16 <li><a href="index.html">Home</a></li>
17 <li><a href="about.html">About Us</a></li>
18 <li><a href="product.html">Product</a></li>
19 <li><a href="imprint.html">Imprint</a></li>
20 =====
21 <li><a href="returns.html">Returns</a></li>
22 <li><a href="faq.html">FAQ</a></li>
23 >>>>>>> develop
24 </ul>
25 </div>
```

Versão da
branch
atual

Versão
remota

10. Diff

10. Diff

Permite visualizar as mudanças realizadas no repositório:

git diff

```
MINGW64 /c/projetos/tutorial-python-brasil (my-branch2)
$ git diff
diff --git a/README.md b/README.md
index 7416216..191df8b 100644
--- a/README.md
+++ b/README.md
@@ -10,6 +10,8 @@ Abordaremos conceitos como integração com serviços externos, integração com
Vamos tentar nos preparar para situações ruins que possam acontecer e garantir que nosso sistema s
+nova alteracao
+
## O que veremos?
- Integração com serviços externos
```


11. Stash

11. Stash

Faz um backup das mudanças que ainda não foram “comitadas”, retornando o repositório ao último estado “limpo” (correspondente ao último commit).

git stash

git stash list

git stash pop

git stash drop

git stash apply

12. Reset, Revert e Checkout

12. Reset

Permite *resetar* a referência atual da **head** para um estado específico. (apagando o que aconteceu nesse meio tempo do historico)

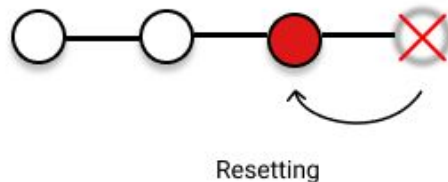
É possível resetar o estado de arquivos específicos, bem como de uma branch inteira.

git reset <hash> -- [paths]

git reset --hard

git reset HEAD <file-to-reset>

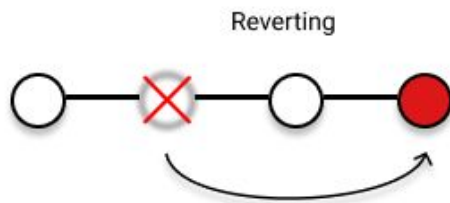
git reset HEAD~<commits-to-reset>



12. Revert

Permite *reverter* um commit, gerando um novo commit que conta esse revert (ou seja fica no historico)

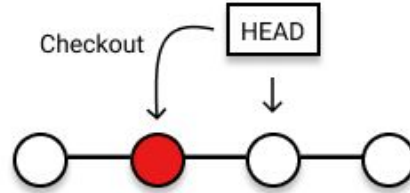
git revert <hash>



12. Checkout

Permite desfazer alterações locais.

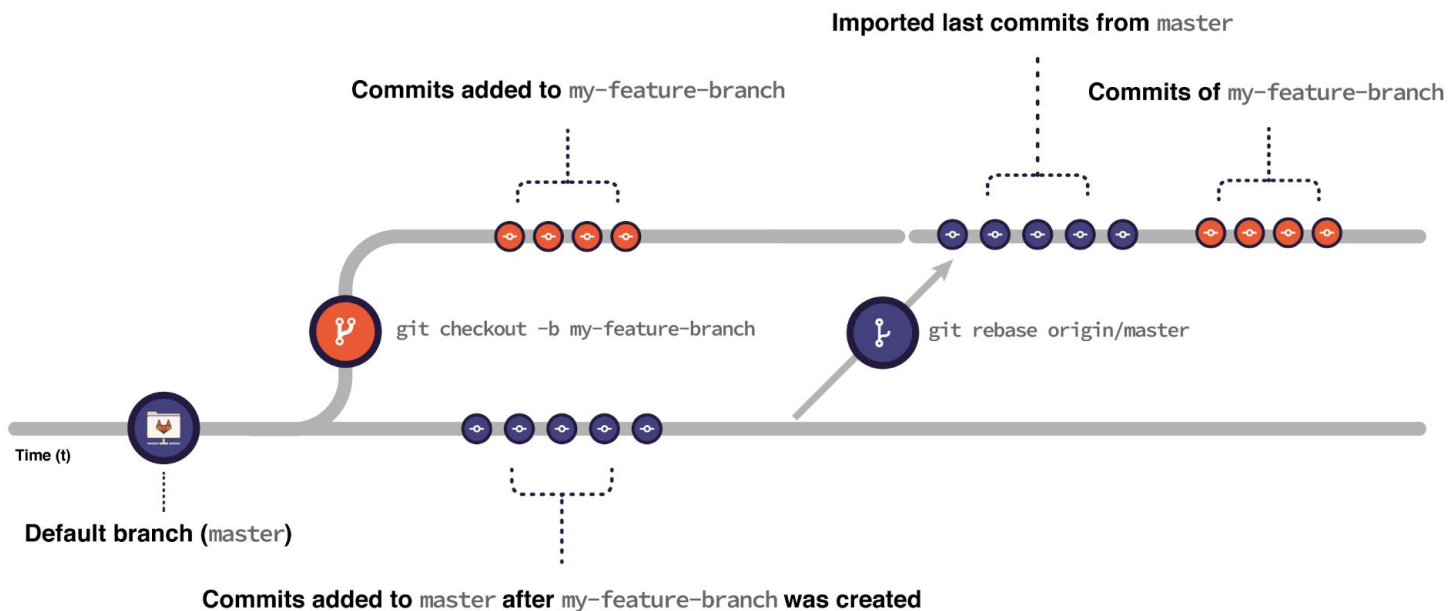
git checkout -- [paths]
git checkout -- .



13. Rebase

13. Rebase

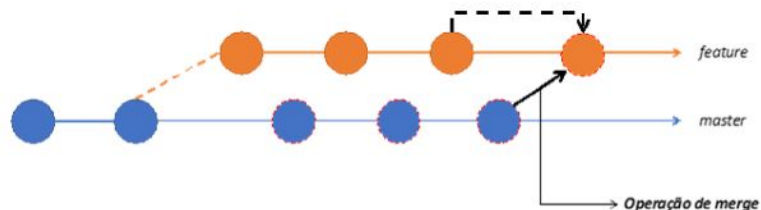
Uma maneira de integrar mudanças em branches e resolver divergências



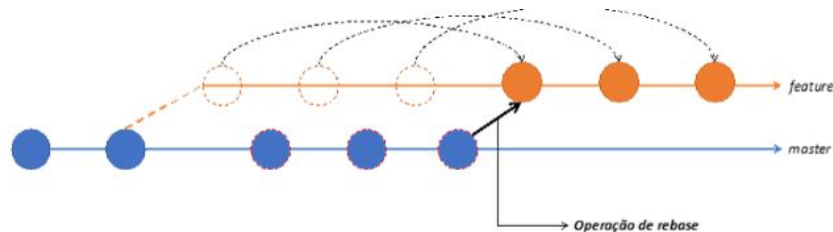
https://docs.gitlab.com/ee/topics/git/img/git_rebase_v13_5.png

13. Rebase

Merge:



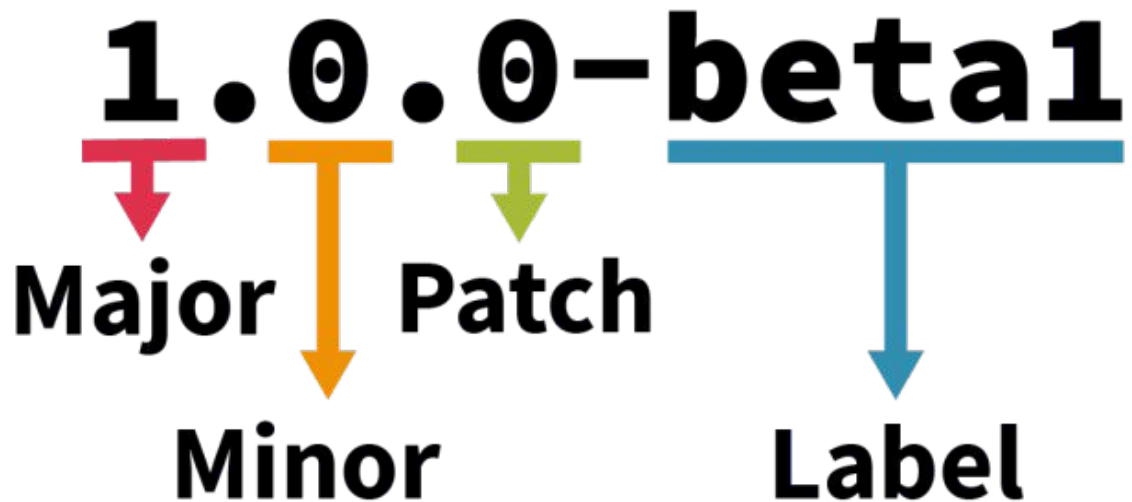
Rebase:



<https://www.treinaweb.com.br/blog/git-merge-e-git-rebase-quando-usa-los>

14. Tags

14. Tags



<https://i0.wp.com/n8d.at/wp-content/uploads/2016/10/semantic-version.png?resize=720%2C349&ssl=1>

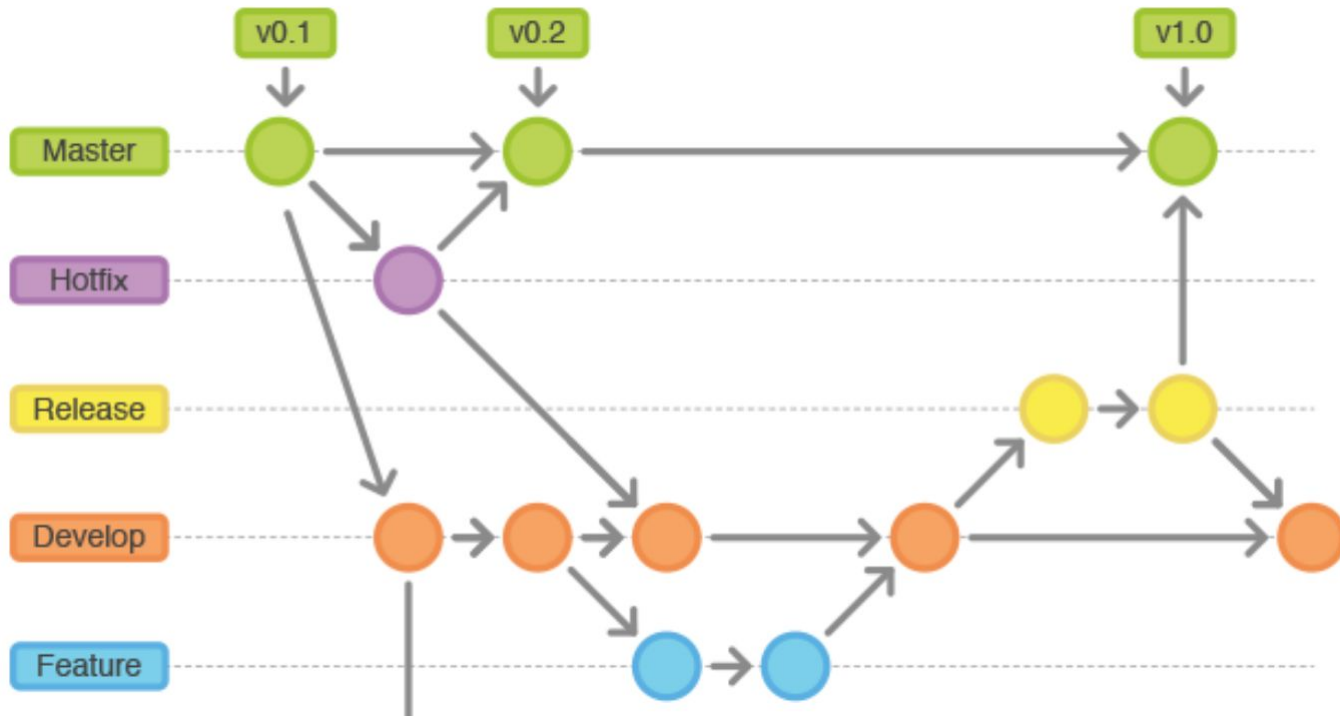
Listar: **git tag**

Criar: **git tag <tag>**

Criar com mensagem: **git tag -a <tag> -m <msg>**

Enviar para remoto: **git push origin <tag>**

14. Tags





Perguntas?

Magalu



#VemSerFeliz