**O que são branches?**

terça-feira, 20 de junho de 2023

15:12

* + As branchs são ramificações feitas através de outros desenvolvedores para trabalhar no projeto sem que utilize a branch master/main
  + Nunca deve ser utilizada a branch master/main por que se feita uma modificação definitiva na master e for perdida, é mais complicado de se recuperar
  + Branchs são as divisões do projeto em versões diferentes
  + Após a finalização do projeto as branchs são unidas para ter o código-fonte final ( Merge )
  + Branch dev - features mais novas adicionadas nessa branch
  + Branch staged - é a branch onde entrarão em testes

**Criando uma branch - git branch**

terça-feira, 27 de junho de 2023

14:26

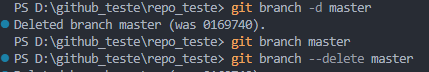
* Visualizando branchs disponíveis - **git branch**
* Criando branchs - **git branch nome**
* Quando criamos uma branch ela parte de qual branch você criou.
  + **Ex:** Criamos uma branch dev a partir da main, a branch dev vai ter tudo que a branch main tem e o que você adicionar nessa nova branch.

**Deletando uma branch - git branch -d**

terça-feira, 27 de junho de 2023

14:35

* + Utiliza-se a flag **-d ou --delete**
  + Deve-se usar o comando git branch -d nome
  + É pouco usado, pois serve de histórico
  + Usado geralmente quando criamos errado a branch



Mudando de branch - git checkout nomebranch

terça-feira, 27 de junho de 2023

14:39

* Utiliza-se a flag **-b**
* Também é utilizado para dispensar mudanças de um arquivo
* As alterações que não foram commitadas vão juntas para essa nova branch
* O comando **git checkout -b nomebranch** - cria uma nova branch e entra nela

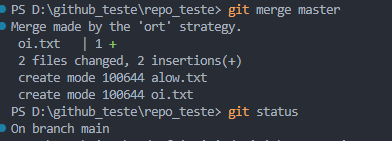
Unindo branchs - git merge nomebranch

terça-feira, 27 de junho de 2023

14:56

* Deve-se usar na branch que queremos adicionar o conteúdo
* Utiliza-se o comando **git merge <nome>**
* Também pode ser utilizado para atualizar uma branch atrasada

**Trazendo conteúdo de uma branch master para a main**



**Utilizando uma stash - git stash**

terça-feira, 27 de junho de 2023

15:14

* Funciona como se estivesse jogando o código no lixo, ou seja as modificações feitas seram salvas juntos com o **git stash**
* A branch após o stash volta para o sua versão original do repositório remoto
* Os commits também são perdidos



**Recuperando um stash - git stash apply**

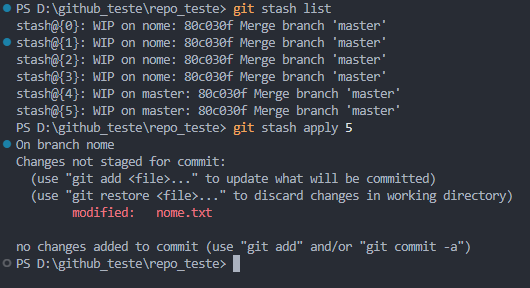
terça-feira, 27 de junho de 2023

15:39

* Deve-se usar o comando **git stash list** para verificar a lista de stash
* Para recuperar devemos utilizar o comando **git stash apply numerostash**
* Para verificar no terminal as modificações feitas nessa stash podemos usar o comando

**git stash show -p numerostash**

**Recuperando stash 5**

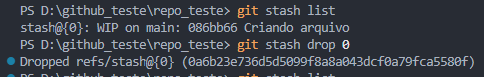


**Removendo as stash- git stash clear ou git stash drop numerostash**

terça-feira, 27 de junho de 2023

15:44

* Para excluir todas as stash devemos usar o comando **git stash clear**
* Para excluir uma stash específica devemos usar o comando **git stash drop numerostash**



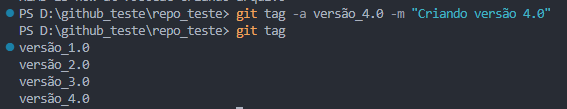
**Criando tags - git tag -a <nome> -m "<msg>"**

terça-feira, 27 de junho de 2023

16:07

* Serve para criar tags e salvar algo na branch
* Diferente do stash a tag serve como um checkpoint da branch
* Utilizado para demarcar estágios do desenvolvimento de algum recurso, funciona como se fosse um ponto de recuperação
* Salva em versões
* Elas funcionam como branchs de um branchs praticamente

**Criando uma tag**



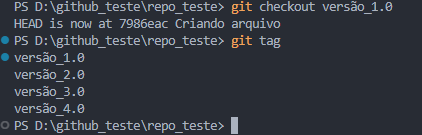
**Alterando entre tags - git show ou git checkout**

terça-feira, 27 de junho de 2023

16:19

* Usamos o comando **git show nome** - para ver as modificações da tag
* Usamos o comando **git checkout nome** - para alterar entre as tags

**Alterando entre tags**



**Enviando tags ao repositório - git push origin nome**

terça-feira, 27 de junho de 2023

16:28

* As tags podem ser compartilhadas entre outros devs, quando enviada para o repositório online
* Deve-se usar o comando **git push origin <nometag>**
* Para enviar todas as tags devemos usar **git push origin --tags**