

Guia de uso Git / GitHub

Nome: Luan Martins Corrêa Data: 08/11/2021

O que é Git?

Git é um sistema de versionamento de código, atualmente é o mais utilizado no mundo por suas funcionalidades, desempenho e segurança.

Veremos a seguir um breve guia de uso, abordando suas funções básicas.

Guia de uso - Git / GitHub

1. Instalação

O primeiro passo é instalarmos o Git em nosso computador. Existem pequenas diferenças na instalação de acordo com o sistema operacional. Nesse guia, a fim de deixa-lo mais sintetizado iremos seguir utilizando o Windows.

Instalando o Git

- 1. Baixar o instalador mais recente (gitforwindows.org)
- 2. Seguir o passo a passo do instalador, atentando-se ao assistente de configuração.
- 3. Abrir o prompt de comando / terminal / Git Bash (de acordo com a opção escolhida).
- 4. Executar os seguintes comandos para configurar com seu usuário, para associar seus *commits*

```
$ git config --global user.name "Seu Nome" $ git config
--global user.email "exemplo@exemplo.com"
```

2. Criação de repositório

Novo Repositório

Para criar um novo repositório executamos o comando git init.

Obter Repositório

Para obter um repositório já existente de maneira local executamos o comando:

```
git clone /caminho/para/repositorio
```

Caso deseje usar um servidor remoto, executa-se o comando:

```
git clone usuário@servidor:/caminho/para/repositorio
```

3. Fluxo de trabalho

Os repositórios locais são distribuídos em 3 "árvores":

- 1. Working Directory: Contém os arquivos
- 2. Index: Funciona como área temporária
- 3. **HEAD:** Aponta para o ultimo commit

4. Adicionar / Commit

Para adicionar arquivos ao Index, usa-se os comandos:

```
git add <arquivo>
git add *
```

Para confirmar as mudanças e efetuar o *commit* executamos o comando:

```
git commit -m "comentário"
```

Feito isto, o arquivo agora está no HEAD.

5. Enviar alterações

Com as alterações já no HEAD, agora enviaremos para o repositório remoto com o seguinte comando:

```
git push origin master
```

master pode ser alterado para qualquer ramo (*branch*) existente, que para onde serão envias suas alterações.

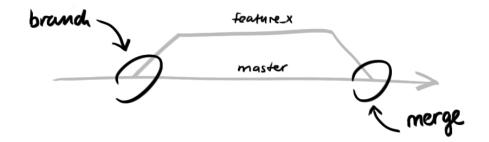
Caso seu repositório não tenha sido clonado de outro existente, você pode conecta-lo a um servidor remoto com o comando:

```
git remote add origin <servidor>
```

Em seguinda, já é possível enviar as alterações para esse servidor.

6. Ramificações (branches)

As *branches* são ramificações usadas para separar ou desenvolver funcionalidades aparte, de maneira a isola-las das outras. A branch principal é a master. Você pode criar novas, e quando conveniente pode mescla-las (*merge*) novamente a branch principal.



- 1. Criando nova *branch*: git checkout -b novabranch
- 2. Retornando a master: git checkout master
- 3. Removendo *branch*: git branch -d novabranch
- 4. Enviar branch para repositório remoto: git push origin <novabranch>

7. Atualizar (pull) / Mesclar (merge)

Para atualizar uma nova versão no repositório local executamos o comando:

```
git pull
```

Seguindo, para mesclar uma branch a sua branch ativa (ex: master) usa-se o comando:

```
git merge <branch>
```

O git tentará mesclar automáticamente. Em casos específico isso acaba não sendo possível, sendo necessário ajustar manualmente possíveis conflitos.

Você pode pré-visualizar as alterações usando o comando:

```
git diff <branch origem> <branch destino>
```

Depois de concluído basta marcarmos como *merged*, usando os comandos:

```
git add <arquivo>
git merge <branch>
```

8. Rotular (tag)

Para garantir o controle e organização do versionamento, recomenda o uso dos rótulos para os *releases* do código/software.

Para criar um novo rótulo executamos o comando:

```
git tag 1.0.0 1b3e1d64ff
1b3e1d64ff é o id do commit, para obtê-lo usa-se: git log.
```

9. Sobrescrever alterações

Para sobrescrever as alterações locais em caso de possíveis erros, usamos o comando:

```
git checkout -- <arquivo>
```

Assim é substituido as alterações com o conteúdo mais recendo de HEAD, mantendo alterações a adições no Index.

Em caso de necessidade de remover todas alterações e commits locais, basta recuperar o histórico mais recente do servidor para a *branch* principal da seguinte maneira:

```
git fetch origin
git reset --hard origin/master
```