ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

SCHERMAN PARIS – 202027594

Tutorial Git - GitHub

PORTO ALEGRE

2021

Tutorial Git (GitHub) para iniciantes

Nesse tutorial estou considerando que o usuário já possui um conta no Git e GitHub, saiba como acessar os repositórios. Também foi considerado que o usuário tenha Git instalado em sua máquina.

Serão listados os comandos mais utilizados por equipes de desenvolvimento de sistemas.

1º PASSO: Como usar o GIT

Com o GIT instalado, configurado em seu dispositivo, podemos explorar alguns conceitos e iniciarmos a utilização.
Os comandos GIT podem ser utilizados em qualquer serviço que permita do versioanamento GIT.

- Bitbucket (Atlasian)
- Space (JetBrains)
- GitHub (Microsoft)
- entre outros

Criar/configurar/verificar um repositório

 # git init <directory> : Com esse comando você transforma qualquer diretório em um repositório GIT;

 # git clone <directory> : Para clonar um repositório, em um ambiente GIT, basta executar esse comando. Ele fará o download de todas as pastas e arquivos armazenados no repositório (‘**<directory>’**);

Obs.: você também pode utilizar esse comando para clonar um repositório local.

exe:

 # git clone /path/to/local/repository

Comandos básicos:

Qualquer alteração realizada em algum arquivo, ou novo arquivo que deverá ser enviado para o repositório, precisará ser enviado para o arquivo índice.

 # git add . : Com esse comando você pode adicionar todos os arquivos/alterações listadas pelo "*git status*" sem precisar dar "*git add <nome_do_arquivo>*" um por um.

 # git add <nome_do_arquivo> : esse comando adiciona o arquivo apontado para ser enviado para o repositório.

 # git status : com esse comando é possível listar todos os arquivos que estão pendentes para submissão, e estão sendo rastreados pelo Git;

> **Arquivo '.gitignore'** : nesse arquivo podem ser adicionados todos os arquivos e/ou pasta que deverão ser ignoradas na submissão para o repositório remoto, ou seja, estes arquivos ou pastas não serão rastreados pelo Git.

 # git commit -m "mensagem_ou_descrissão_do_envio" : esse comando é responsável por preparar as alterações a serem enviadas para a *branch* onde o Git está sendo apontado, criando pequenas histórias documentadas com as alterações que estão sendo realizadas, em seu ambiente local.

***OBS.:** o comando "**git commit**" não funciona se não houver envio de arquivos pelo "**git add**"*

 # git push : é o comando responsável pelo envio dos arquivos "*commitados*" para o ambiente remoto.

Para realizar o envio para o ambiente remoto, em uma branch (local) e que ainda não obteve comunicação com o repositório remoto, então precisará rodar o comando:

 # git push -u origin "branch-nome" : com esse comando, a branch local será criada no ambiente remoto, criará um relacionamento entre eles, e enviará as alterações salvas no commit.

Branchs

As ***barnchs,*** ou ramificações, possibilita que várias pessoas atuem paralelamente no mesmo projeto. Assim cada um poderá realizar seu trabalho sem "atrapalhar" o desenvolvimento do outro.

 # git branch “nome-da-branch” : Com esse comando você criará uma branch em seu ambiente local, depois você precisará realizar o comando ***# git push -u origin “branch-nome”***

 # git branch ou **git branch --list :** com esses comandos você poderá listar todas as branchs existentes no ambiente.

 # git branch -d “nome-da-branch” : Esse comando permite que você delete uma branch;

 # git checkout “nome-da-ramificação” : Com esse comando, você poderá alterar o “apontamento” do seu ambiente local para a branch selecionada;

 # git checkout -b “nome-da-branch” : Com esse comando você poderá criar uma branch e já alterar diretamente para ela.

 # git pull : Quando você dá “checkout” em uma branch remota, precisará dar esse comando para que seu ambiente local possua os arquivos/pastas existentes no repositório remoto.

Adicional

 # git stash : Com esse comando você salva as condições atuais do diretório ativo, e do indice. As alterações salvas com esse comando podem ser vistas com o comando **# git stash list.**

 # git fetch : Permite realizar download dos objetos e *refs* do repositório;

 # git prune : Esse comando é responsável pela remoção de todos os objetos inacessíveis armazenados.