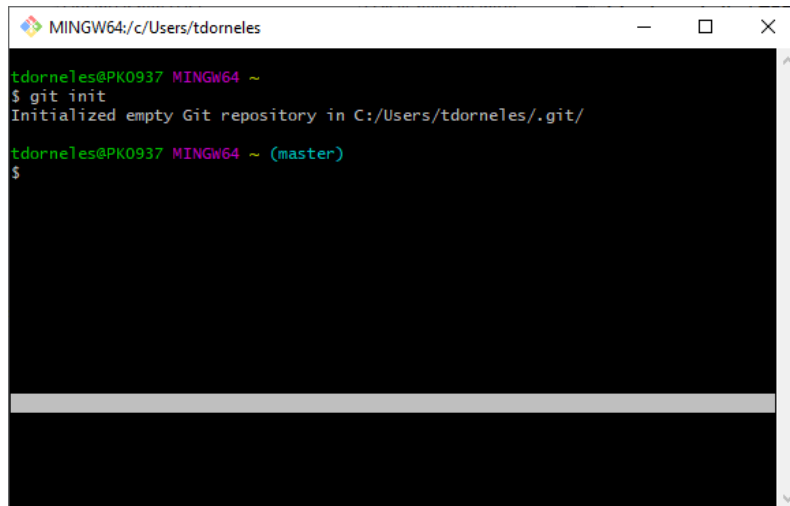


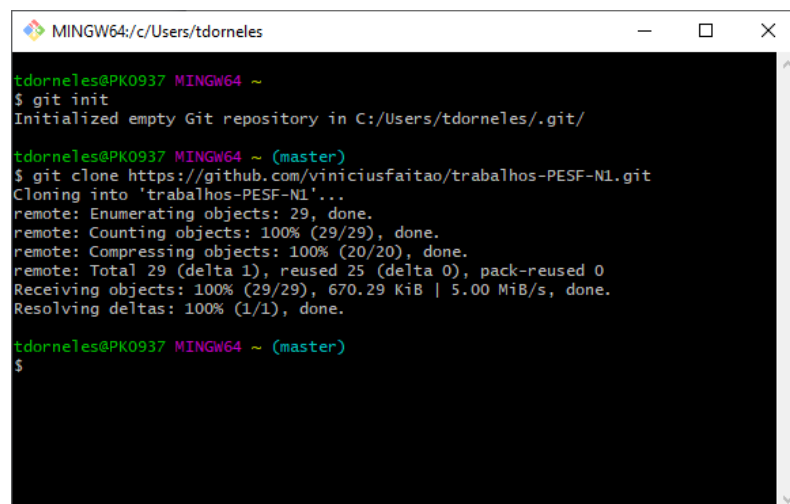
# Tutorial Git/GitHub

1. O Git é um sistema de controle de versão de código/documentos, para iniciarmos precisamos começar um diretório, para isso acontecer utilizamos o comando ***git init***, ele iniciará o repositório localmente na sua maquina chamada ***/.git***, por padrão ele já cria a branch **master**, branch é como se fosse uma ramificação em controle de versão e gerenciamento.



```
MINGW64/c/Users/tdorneles
tdorneles@PK0937 MINGW64 ~
$ git init
Initialized empty Git repository in C:/Users/tdorneles/.git/
tdorneles@PK0937 MINGW64 ~ (master)
$
```

2. Depois podemos clonar um repositório, como o do GitHub, assim usamos o comando ***git clone <link do repositório>***, ele irá literalmente clonar o repositório que está dentro do GitHub para conseguir realizar o versionamento do teu código sem precisar estar conectado na internet.



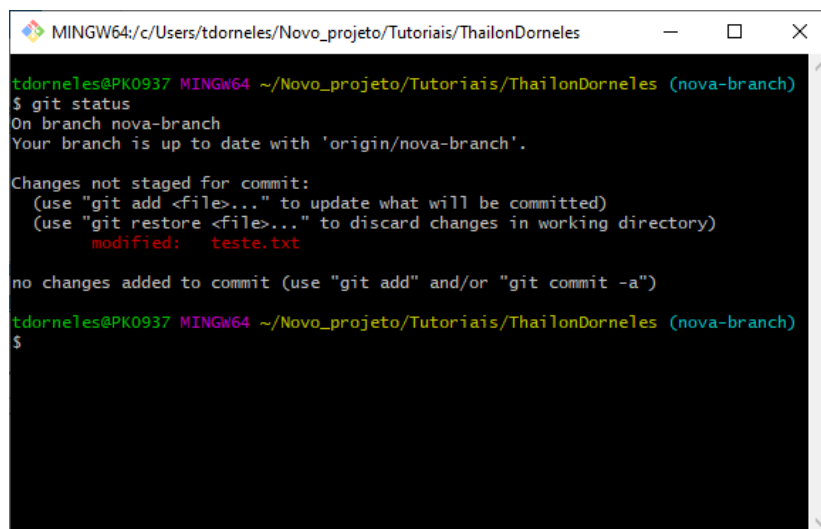
```
MINGW64/c/Users/tdorneles
tdorneles@PK0937 MINGW64 ~
$ git init
Initialized empty Git repository in C:/Users/tdorneles/.git/

tdorneles@PK0937 MINGW64 ~ (master)
$ git clone https://github.com/viniciusfaitao/trabalhos-PESF-N1.git
Cloning into 'trabalhos-PESF-N1'...
remote: Enumerating objects: 29, done.
remote: Counting objects: 100% (29/29), done.
remote: Compressing objects: 100% (20/20), done.
remote: Total 29 (delta 1), reused 25 (delta 0), pack-reused 0
Receiving objects: 100% (29/29), 670.29 KiB | 5.00 MiB/s, done.
Resolving deltas: 100% (1/1), done.

tdorneles@PK0937 MINGW64 ~ (master)
$
```

3. Temos o comando ***git pull <repositório remote>***, ele basicamente faz o mesmo que o clone porém ele busca somente o que tiver de diferente do repositório remoto e do que tu tem na tua maquina já, se não tiver nada então ele traz tudo.

4. Para criar um branch podemos utilizar o comando ***git branch <nome da branch>*** podemos também mudar de branch e navegar nelas utilizando o comando ***git checkout -b <nome da branch>***.
5. Usamos o código ***git remote add origin <link do repositório>***, conseguimos nos conectar no repositório do github por exemplo e fazer um push com nossos commits lá para dentro.
6. Antes de conectarmos no github temos 3 estágios para realizarmos nossos commits, sempre quando um arquivo for criado na pasta ele ficará com um Status ***Untracked Files***, ou seja, ele é um arquivo desconhecido pelo git, o git não está controlando a versão do mesmo. Para verificar esse Status podemos usar o ***git status***, ele vai listar todos os arquivos novos e que sofreram modificações.



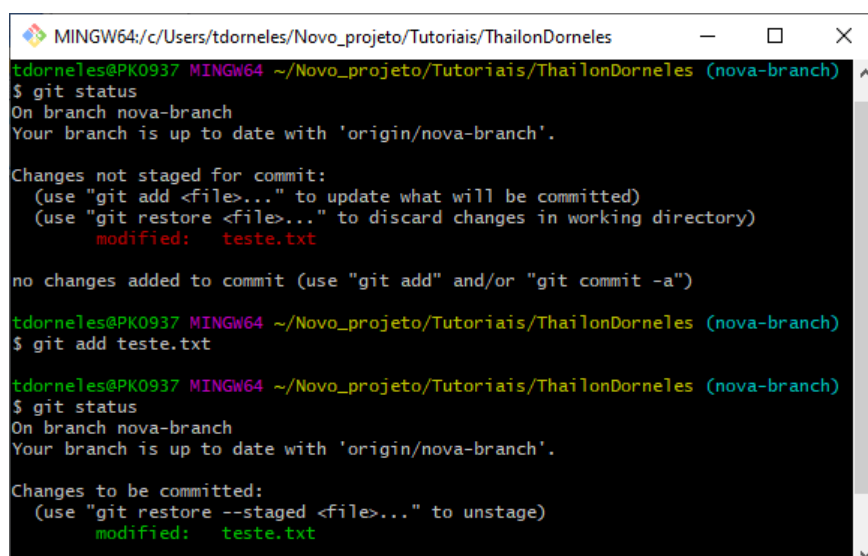
```
MINGW64:/c:/Users/tdorneles/Novo_projeto/Tutoriais/ThailonDorneles
tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$ git status
On branch nova-branch
Your branch is up to date with 'origin/nova-branch'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   teste.txt

no changes added to commit (use "git add" and/or "git commit -a")

tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$
```

7. Para nosso segundo passo temos que adicionar estes arquivos, transformá-los no status Changes to be committed, ou seja os arquivos adicionados a este status já estão prontos para serem commitados, e farão parte do repositório quando executarmos o commit. O código que utilizamos para adicionar é ***git add <nome arquivo>***.



```
MINGW64:/c:/Users/tdorneles/Novo_projeto/Tutoriais/ThailonDorneles
tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$ git status
On branch nova-branch
Your branch is up to date with 'origin/nova-branch'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   teste.txt

no changes added to commit (use "git add" and/or "git commit -a")

tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$ git add teste.txt

tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$ git status
On branch nova-branch
Your branch is up to date with 'origin/nova-branch'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   teste.txt
```

8. Terceiro estágio é commitar o arquivo, utilizamos o comando ***git commit -m "descrição do commit"***, ele irá efetivamente adicionar a versão do arquivo que você adicionou para o repositório, logo quando utilizarmos o ***git status***, não vai mais ficar nada pendente.

```
MINGW64:/c/Users/tdorneles/Novo_projeto/Tutoriais/ThailonDorneles
tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$ git commit -m "mais um teste"
[nova-branch f2da8b3] mais um teste
1 file changed, 1 insertion(+)

tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$ git status
On branch nova-branch
Your branch is ahead of 'origin/nova-branch' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$

tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$

tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$

tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$
```

9. Para colocarmos no repositório do github utilizamos o ***git push -u origin <branch criada>***, ele vai “empurrar” todo teu repositório que está local para o remoto lá no github, atualizando todos os códigos.

```
MINGW64:/c/Users/tdorneles/Novo_projeto/Tutoriais/ThailonDorneles
tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$ git status
On branch nova-branch
Your branch is ahead of 'origin/nova-branch' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$ git push -u origin nova-branch
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 449 bytes | 224.00 KiB/s, done.
Total 5 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:viniciusfaitao/trabalhos-PESF-N1.git
   347874e..f2da8b3  nova-branch -> nova-branch
Branch 'nova-branch' set up to track remote branch 'nova-branch' from 'origin'.

tdorneles@PK0937 MINGW64 ~/Novo_projeto/Tutoriais/ThailonDorneles (nova-branch)
$
```

10. Os **pull requests** como o nome já diz, ele traz todos os **push** que foram feitos para o repositório e logo após irá realizar um **merge** que vai fundir o repositório “local” com o repositório do github e vai trazer qualquer erro que tiver. Na segunda imagem ele verifica os conflitos entre as branch e permite fazer o merge

