



Git e Github

Criação de um repositório Git

1 - Acessar o Git Bash e criar um novo diretório:

```
$ mkdir celso
```

2 - Acessar o diretório criado:

```
$ cd celso
```

3 - Criar um arquivo qualquer, por exemplo, index.html com nano:

```
$  
nano index.html
```

4 - Fazer com que um repositório do Git seja inicializado neste diretório:

```
$ git init
```

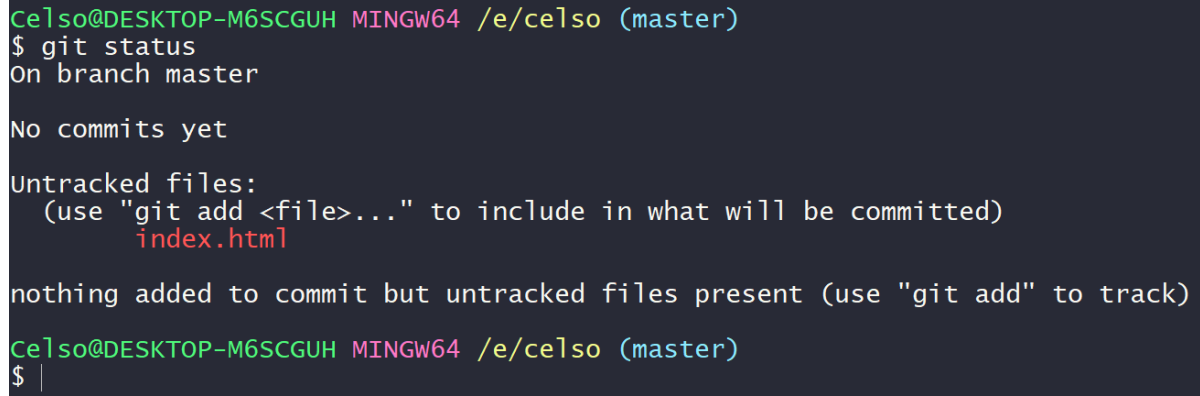
Você terá como resultado a seguinte tela:

```
Celso@DESKTOP-M6SCGUH MINGW64 /e/celso  
$ git init  
Initialized empty Git repository in E:/celso/.git/  
Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)  
$
```

5 - Exibir o status do repositório:

\$ git status

O resultado deverá se parecer com o da imagem abaixo:



```
Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        index.html

nothing added to commit but untracked files present (use "git add" to track)
Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ |
```

On branch master, quer dizer que estamos no ramo principal do nosso repositório.

No commits yet, significa que ainda não fizemos nenhum commit em nosso repositório.

Untracked files, exibe uma lista com os arquivos que não estão sendo monitorados, neste caso somente o **index.html**.

Na última linha temos a informação que não há nada para “comitar”, mas que há arquivos não monitorados que podem ser adicionados ao monitoramento, neste caso somente o **index.html**.

Parabéns!! Você já criou o seu primeiro repositório Git.

Configurações iniciais do repositório Git

1 - Para atribuir um nome de usuário e e-mail para um repositório, utilize os seguintes comandos:

```
$ git config --local user.name "Seu nome"
```

```
$ git config --local user.email "seu email"
```

Marcando arquivos para serem monitorados pelo Git

1 - Ativando monitoramento em um arquivo específico:

```
$ git add index.html
```

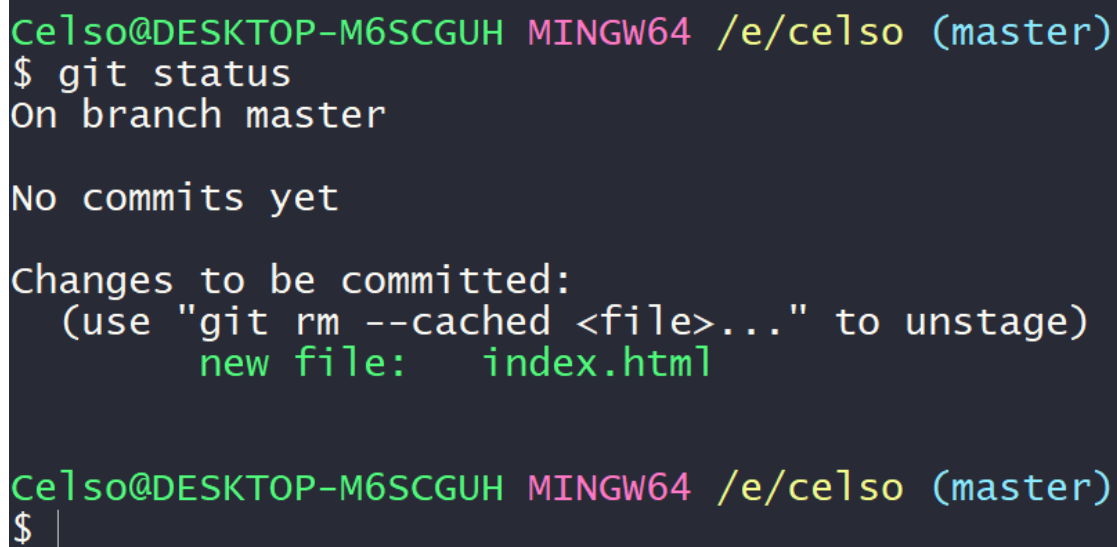
2 - Caso você tenha vários arquivos no repositório e queira que todos sejam monitorados utilize o comando:

```
$ git add .
```

3 - Verificando se os arquivos foram adicionados ao monitoramento:

```
$ git status
```

O resultado do comando acima deverá ser similar à imagem abaixo:

A terminal window with a dark background and light-colored text. The prompt is 'Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)'. The command '\$ git status' is entered. The output shows 'On branch master', 'No commits yet', and 'Changes to be committed: (use "git rm --cached <file>..." to unstage)'. A new file 'index.html' is listed. The prompt is repeated at the bottom with a cursor after the '\$'.

Changes to be committed, indica quais são os arquivos que ainda não foram salvos no histórico.

4 - Efetuar a gravação das mudanças no Git:

```
$ git commit -m "Mensagem qualquer"
```

Observação: A mensagem é obrigatória. Ela deve descrever as mudanças na atualização do repositório. Procure utilizar frases curtas.

O resultado do comando acima deverá ser similar ao da imagem abaixo:

```
Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ git commit -m "Criação do arquivo index.html"
[master (root-commit) 7ccd12e] Criação do arquivo index.html
1 file changed, 2 insertions(+)
create mode 100644 index.html

Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ |
```

5 - Verificar se ainda há alguma atualização para salvar:

\$ git status

O resultado abaixo indica que não há mais arquivos monitorados a serem atualizados e que não há nenhum novo arquivo não monitorado:

```
Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ git status
On branch master
nothing to commit, working tree clean

Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ |
```

Atualizando as alterações em nossos repositórios

1 - Abra o arquivo index.html e faça alguma alteração nele, salve e execute o comando abaixo:

\$ git status

O resultado do comando deverá ser similar à imagem abaixo:

```
Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ git status
On branch master
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:   index.html

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

Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ |
```

A tela acima nos informa que o arquivo index.html, que é monitorado pelo Git, sofreu alterações desde a última atualização e que devemos utilizar o comando git add para adicioná-lo novamente a lista de arquivos que serão atualizados.

2 - Adicionado os arquivos alterados à lista de commits futuros:

\$ git add index.html

Observação: lembre-se que você poderia utilizar o ponto no lugar do nome do arquivo.

Verifique novamente o status e o resultado deverá ser similar ao da imagem abaixo:

```
Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   index.html

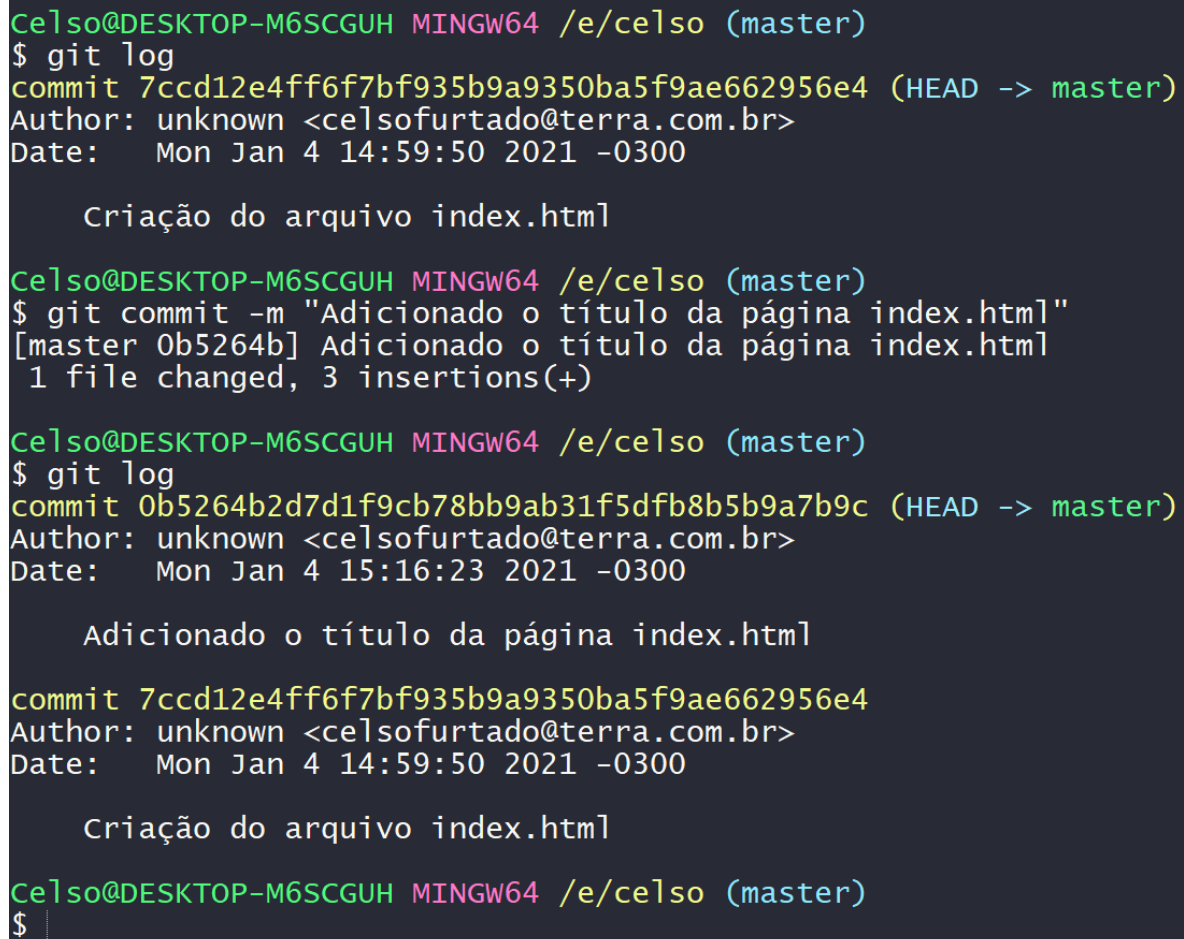
Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ |
```

Visualização do histórico de alterações do repositório

1 - Para visualizar o histórico de alterações de um repositório digite o comando abaixo:

\$ git log

O resultado deverá ser similar ao da imagem abaixo:

A terminal window with a dark background and light-colored text. It shows the execution of the 'git log' command. The output displays the commit hash '7ccd12e4ff6f7bf935b9a9350ba5f9ae662956e4', the author 'unknown <celsofurtado@terra.com.br>', and the date 'Mon Jan 4 14:59:50 2021 -0300'. The commit message is 'Criação do arquivo index.html'. Below this, the 'git commit' command is shown with the message 'Adicionado o título da página index.html'. The output of the commit shows the new commit hash '0b5264b2d7d1f9cb78bb9ab31f5dfb8b5b9a7b9c', the same author and date, and the commit message. The final part of the screenshot shows the 'git log' command being executed again, but the output is partially cut off.

```
Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ git log
commit 7ccd12e4ff6f7bf935b9a9350ba5f9ae662956e4 (HEAD -> master)
Author: unknown <celsofurtado@terra.com.br>
Date:   Mon Jan 4 14:59:50 2021 -0300

    Criação do arquivo index.html

Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ git commit -m "Adicionado o título da página index.html"
[master 0b5264b] Adicionado o título da página index.html
1 file changed, 3 insertions(+)

Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ git log
commit 0b5264b2d7d1f9cb78bb9ab31f5dfb8b5b9a7b9c (HEAD -> master)
Author: unknown <celsofurtado@terra.com.br>
Date:   Mon Jan 4 15:16:23 2021 -0300

    Adicionado o título da página index.html

commit 7ccd12e4ff6f7bf935b9a9350ba5f9ae662956e4
Author: unknown <celsofurtado@terra.com.br>
Date:   Mon Jan 4 14:59:50 2021 -0300

    Criação do arquivo index.html

Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$
```

O histórico é exibido da atividade mais recente para a mais antiga. Na imagem acima, vemos que a alteração mais recente é o “Adicionando o título da página index.html”. Neste commit, um arquivo foi alterado e foram efetuadas três alterações.

2 - Exibir o log de forma resumida:

\$ git log --oneline

O resultado deverá ser similar ao da imagem abaixo:

```
Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ git log --oneline
0b5264b (HEAD -> master) Adicionado o título da página index.html
7ccd12e Criação do arquivo index.html

Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ |
```

Neste caso, estamos vendo somente as mensagens do commit.

3 - Visualizar as alterações de um commit:

\$ git log -p

O resultado deverá ser similar ao da imagem abaixo:

```
Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ git log -p
commit 0b5264b2d7d1f9cb78bb9ab31f5dfb8b5b9a7b9c (HEAD -> master)
Author: unknown <celsofurtado@terra.com.br>
Date: Mon Jan 4 15:16:23 2021 -0300

    Adicionado o título da página index.html

diff --git a/index.html b/index.html
index 90531a4..29680be 100644
--- a/index.html
+++ b/index.html
@@ -1,2 +1,5 @@
 <html>
+ <head>
+ <title>Testando Git</title>
+ </head>
</html>

commit 7ccd12e4ff6f7bf935b9a9350ba5f9ae662956e4
Author: unknown <celsofurtado@terra.com.br>
Date: Mon Jan 4 14:59:50 2021 -0300

    Criação do arquivo index.html

diff --git a/index.html b/index.html
new file mode 100644
index 0000000..90531a4
--- /dev/null
+++ b/index.html
@@ -0,0 +1,2 @@
+<html>
+</html>

Celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ |
```

O conteúdo em verde representa o que foi adicionado ao arquivo.

Exclua algum conteúdo do arquivo index.html, faça o commit novamente e rode o git log -p novamente.

O resultado deverá ser similar ao da imagem abaixo:

```
$ git log -p
commit 9382e483f23499e3ec905dba3f278470e60b268f (HEAD -> master)
Author: Prof. Celso Furtado <celsofurtado@terra.com.br>
Date: Mon Jan 4 15:35:09 2021 -0300

    Remoção do título da página index.html

diff --git a/index.html b/index.html
index 29680be..d25b3f3 100644
--- a/index.html
+++ b/index.html
@@ -1,5 +1,4 @@
<html>
  <head>
-  <title>Testando Git</title>
  </head>
</html>

commit 0b5264b2d7d1f9cb78bb9ab31f5dfb8b5b9a7b9c
Author: unknown <celsofurtado@terra.com.br>
Date: Mon Jan 4 15:16:23 2021 -0300

    Adicionado o título da página index.html

diff --git a/index.html b/index.html
index 90531a4..29680be 100644
--- a/index.html
+++ b/index.html
@@ -1,2 +1,5 @@
<html>
+ <head>
+ <title>Testando Git</title>
+ </head>
</html>

commit 7ccd12e4ff6f7bf935b9a9350ba5f9ae662956e4
Author: unknown <celsofurtado@terra.com.br>
Date: Mon Jan 4 14:59:50 2021 -0300

    Criação do arquivo index.html

diff --git a/index.html b/index.html
new file mode 100644
index 0000000..90531a4
--- /dev/null
+++ b/index.html
@@ -0,0 +1,2 @@
+<html>
+</html>

celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)
$ |
```

Neste exemplo, vemos que no último commit foi removido uma linha do arquivo index.html, que era responsável pelo título da página. As remoções aparecem em vermelho e com sinal negativo (-).

Mais opções de filtros podem ser encontrados neste link: <https://devhints.io/git-log>

Ignorar arquivos

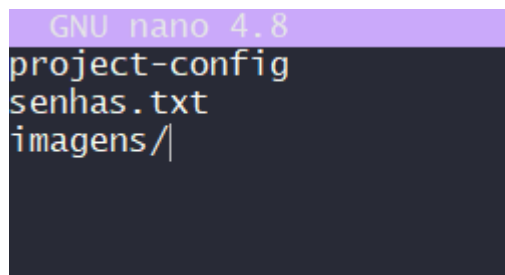
Caso tenhamos alguns arquivos e/ou diretório que não desejamos que seja monitorado pelo Git, devemos utilizar o seguinte recurso:

1 - Crie um arquivo chamado `.gitignore` na raiz do seu repositório:

```
$ nano .gitignore
```

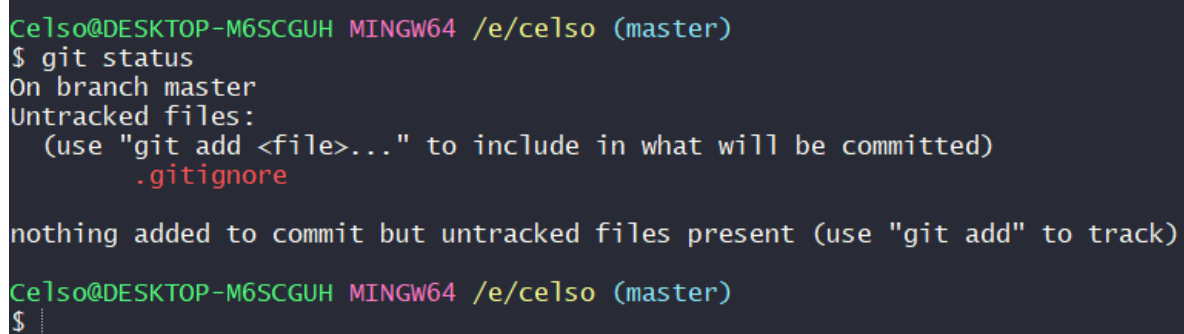
Observação: Os arquivos que começam com um “ponto” são ocultos, então utilize o parâmetro `-a` ao comando `ls` para vê-los.

2 - Adicione, no arquivo `.gitignore`, os nomes dos arquivos que deseja ignorar, um por linha, conforme a imagem abaixo:

A screenshot of a terminal window showing the GNU nano 4.8 text editor. The editor is open to a file named .gitignore. The content of the file is: project-config, senhas.txt, and imagens/. The text is displayed in a light blue font on a dark background. The cursor is at the end of the third line, after the slash in 'imagens/'.

Na imagem acima, as duas primeiras linhas representam arquivos que deverão ser ignorados pelo git. A terceira linha representa um diretório chamado `imagens`, como você pode ver, para informar ao Git que se trata de um diretório, basta colocar uma barra à direita do nome.

3 - Rode o comando `git status` novamente e observe o resultado:

A screenshot of a terminal window showing the output of the git status command. The prompt is 'celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)'. The command '\$ git status' is entered. The output is: 'On branch master', 'Untracked files:', '(use "git add <file>..." to include in what will be committed)', and '.gitignore'. Below this, it says 'nothing added to commit but untracked files present (use "git add" to track)'. The prompt is then 'celso@DESKTOP-M6SCGUH MINGW64 /e/celso (master)' and '\$'.

Como podemos ver, o arquivo **project-config** não está sendo mais monitorado e nem sugerido para monitoramento. E o **.gitignore**? Ele deverá ser monitorado, pois é comum alterarmos o seu conteúdo constantemente e desejamos sempre tê-lo atualizado, então, adicione-o com o `git add`.

Conectar repositório local com o Github

Antes de começar, precisamos de uma conta no Github, então acesse o link abaixo e crie a sua conta:

<https://github.com/>

Depois de criar a sua conta, crie um repositório, que será responsável por receber uma cópia do nosso repositório local. Vamos chamar o repositório de ds1-teste.

Com o comando git status, certifique-se que não haja nada para comitar e execute os seguintes comandos:

1 - Alterar o nome do branch atual que é master para main:

\$ git branch -M main

2 - Adicionar o repositório remoto:

\$ git remote add origin <https://github.com/celsofurtado/ds1-teste>

3 - “Empurrar” o repositório local (main) para o remoto (origin)

\$ git push origin main

Dica para Markdown:

<https://medium.com/@raullestevess/github-como-fazer-um-readme-md-bonit%C3%A3o-c85c8f154f8>

“Puxar” o repositório remoto para o local

1 - Criar o repositório local:

\$ git init

2 - Determinar quem é o repositório remoto (origin):

\$ git remote add origin <endereço do remoto>

3 - Agora é só puxar:

\$ git pull origin main