



# Introdução a Web GIT - Versionando

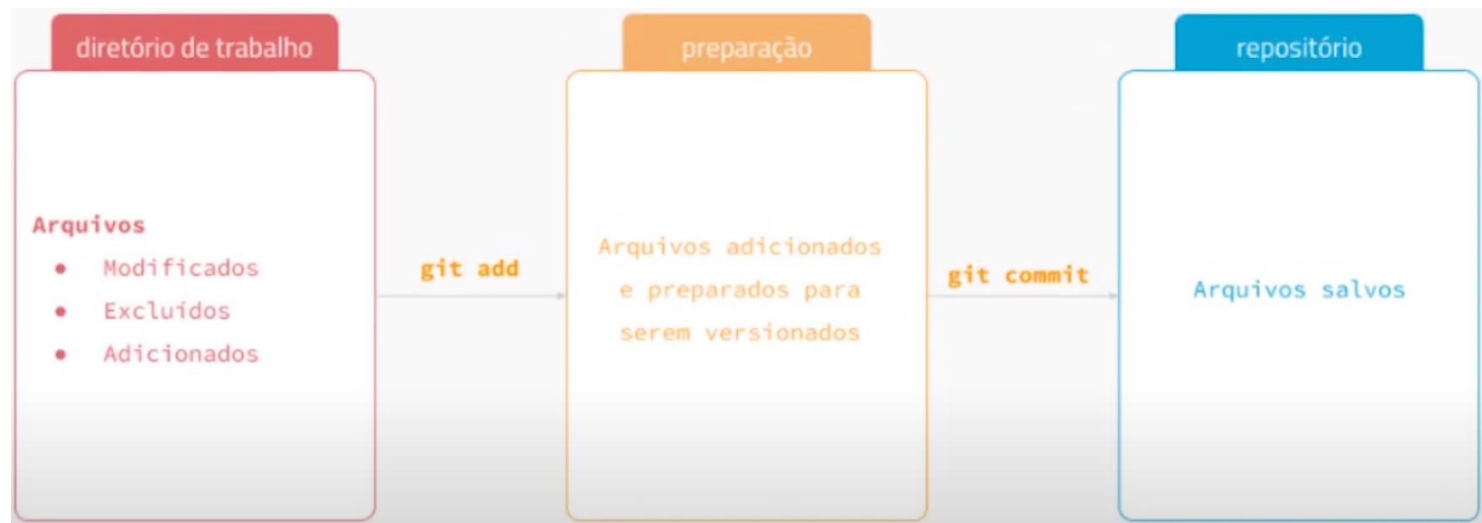


Prof: Sergio Hermenegildo  
FEMASS

# GIT: Conceitos básicos

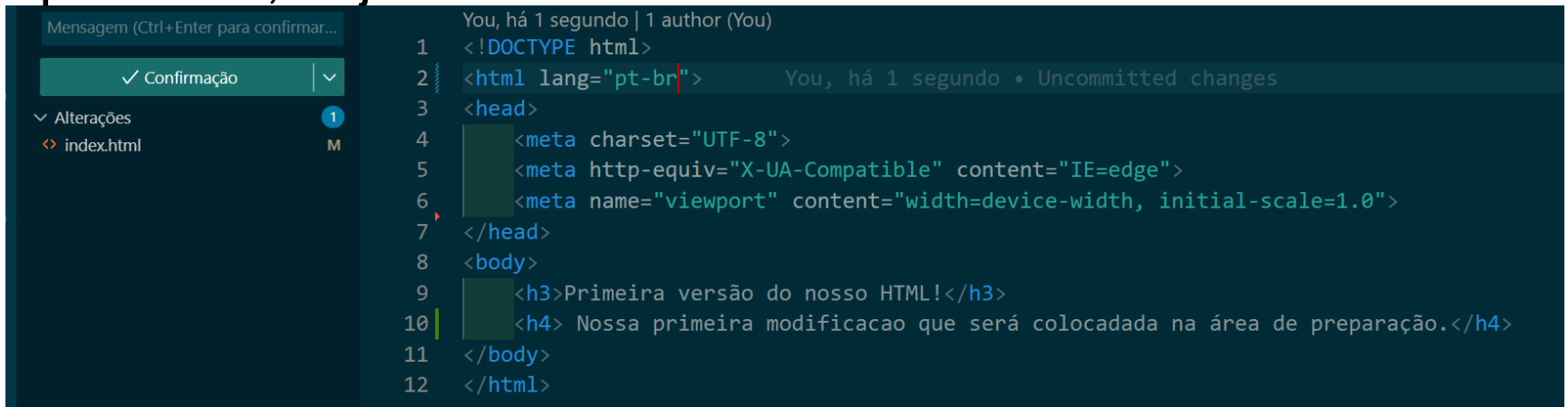
Muito bem, agora que já criamos nosso repositório e fizemos nosso primeiro commit, vejamos todos os passos normais que fazemos no dia a dia da codificação.

Relembrando, nós temos 3 locais, o nosso diretório de trabalho, nossa staging área e nosso repositório



# GIT: Versionando

Então, quando fazemos uma modificação naquele arquivo que já havíamos criado, informações novas aparecem, vejamos.



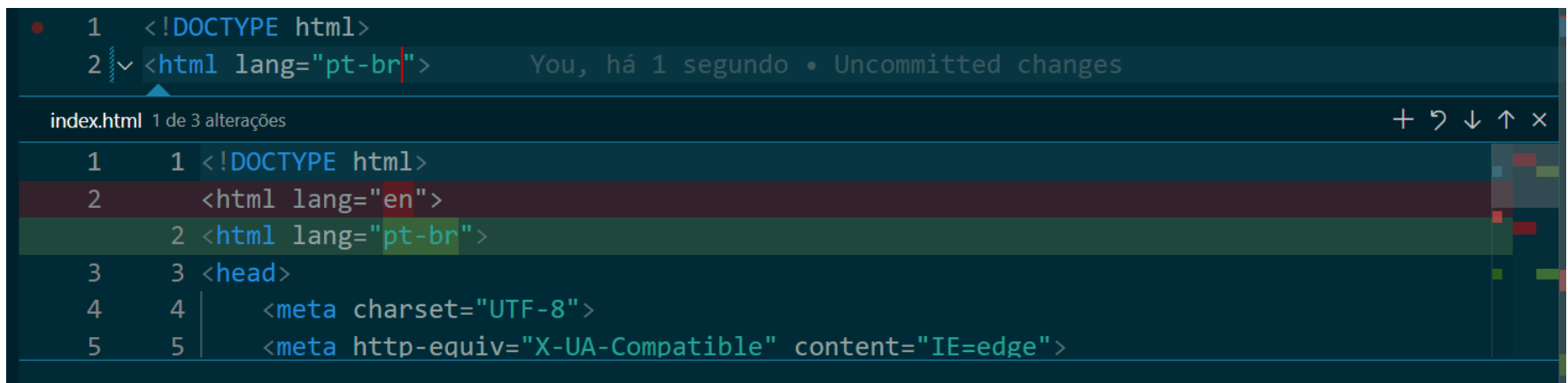
```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7 </head>
8 <body>
9   <h3>Primeira versão do nosso HTML!</h3>
10  <h4> Nossa primeira modificacao que será colocadada na área de preparação.</h4>
11 </body>
12 </html>
```

Ao adicionar a linha 10, é gerado uma linha verde, enquanto que a alteração da linha 2 gera uma linha azul e a exclusão da linha de título da nossa página HTML gera um triângulo vermelho.

# GIT: Versionando

Com esse código de cores, fica fácil para percebermos o que foi mudado no decorrer da nossa programação.

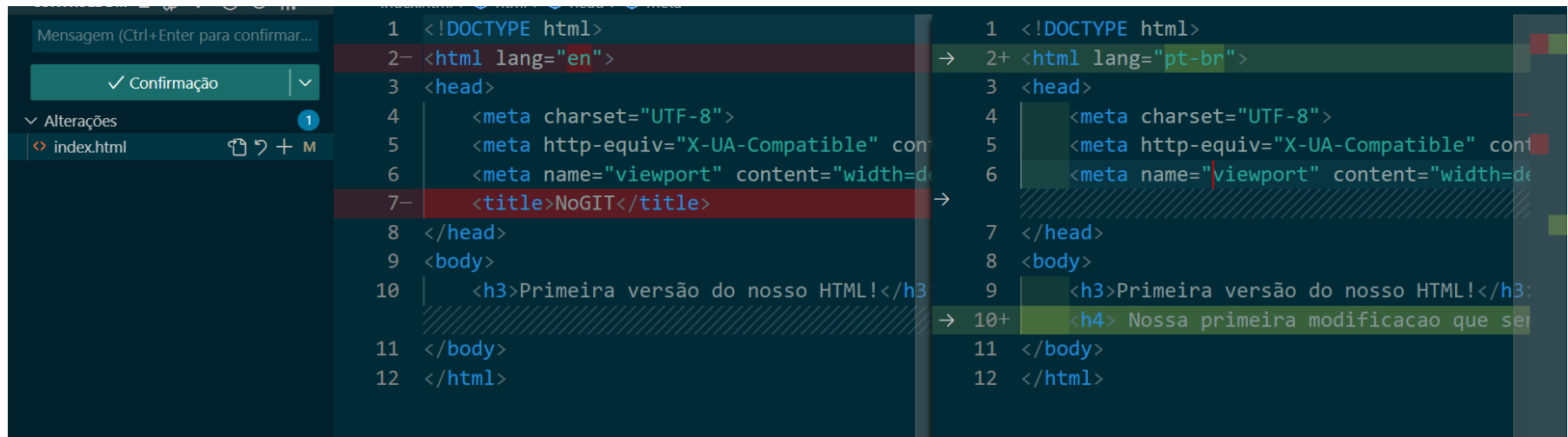
Mas, ainda tem mais, basta clicarmos na linha azul para que ele abra, abaixo da linha, as modificações ocorridas. Percebam os ícones no lado direito, através deles, podemos retornar ao que era antes (setinha de retorno), ou colocar o arquivo na nossa área de preparo (+)



```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

# GIT: Versionando

Beleza, mas talvez aquela visão não seja a mais adequada quando estamos verificando muitas mudanças, então para isso podemos ter uma visão do arquivo lado a lado com todas suas modificações, e para isso, basta clicarmos em cima do nome do nosso arquivo.



```
1 <!DOCTYPE html>
2- <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7- <title>NoGIT</title>
8 </head>
9 <body>
10  <h3>Primeira versão do nosso HTML!</h3>
11 </body>
12 </html>
```

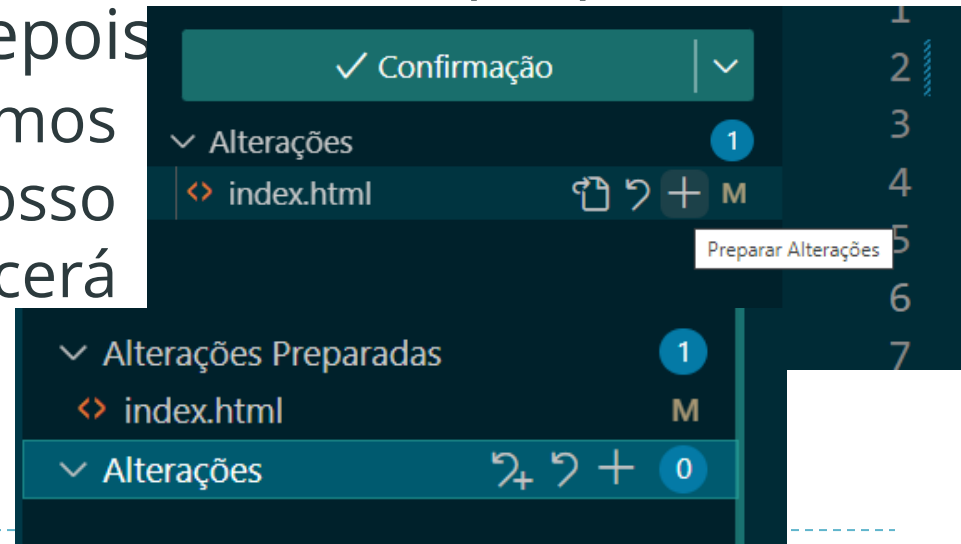
```
1 <!DOCTYPE html>
2+ <html lang="pt-br">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7 </head>
8 <body>
9   <h3>Primeira versão do nosso HTML!</h3>
10+  <h4> Nossa primeira modificacao que sera feita</h4>
11 </body>
12 </html>
```

# GIT: Versionando

Digamos que eu tenha desistido de apagar o título da nossa página, basta para isso, clicar em cima da seta que está na área da direita da mesma linha que excluimos, para que ela volte para nosso arquivo.

Legal, estou satisfeito com minha alteração, irei colocar o arquivo na minha área de preparo para efetivar o commit depois.

Basta então clicarmos no (+) do lado do nosso arquivo e ele aparecerá listado nas alterações\_preparadas

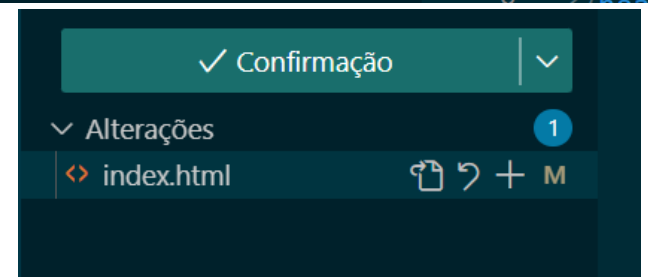
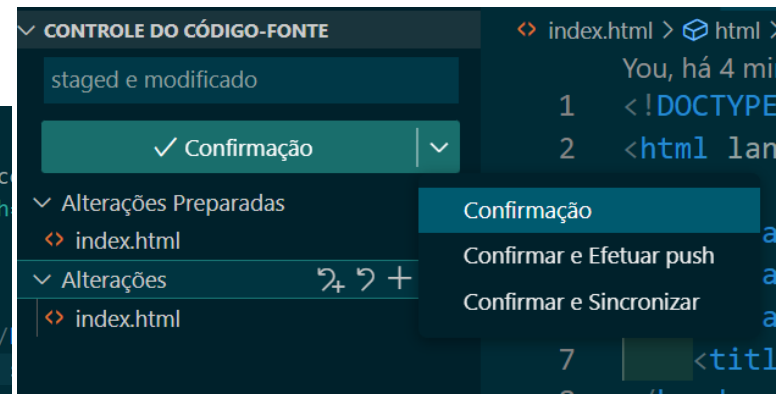


# GIT: Versionando

Notem que, mesmo vocês já tendo posto o arquivo na sua área de preparo, vocês ainda podem implementar quaisquer tipo de modificações no mesmo, mas atentem que se fizermos o commit agora, apenas o que está na área de preparo irá para o repositório, ficando a última alteração apenas no nosso diretório de trabalho. Veremos essa diferença no GITHUB.



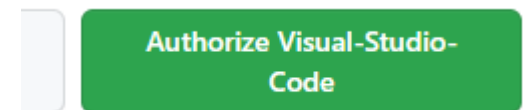
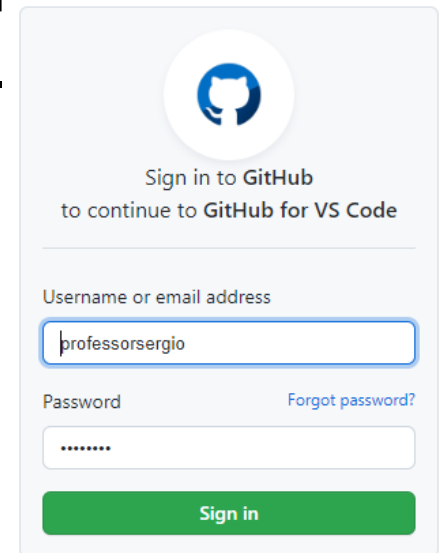
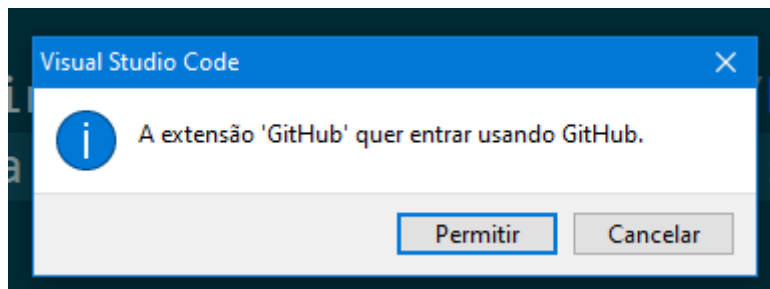
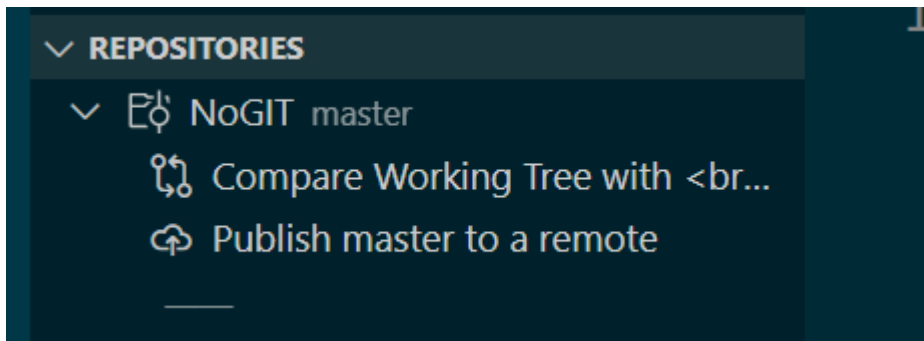
The screenshot shows the VS Code interface. On the left, the Explorer sidebar has two sections: 'Alterações Preparadas' (Staged Changes) with one file 'index.html' marked with an 'M' (modified) icon, and 'Alterações' (Changes) with one file 'index.html' also marked with an 'M' icon. The main editor shows the content of 'index.html', which is an HTML document. The code includes a head section with meta tags for charset, http-equiv, and viewport, and a title 'NoGIT'. The body section contains two paragraphs: 'Primeira versão do nosso HTML!' and 'Nossa primeira modificação que...'. The bottom status bar indicates 'index.html 1 de 1 alteração'.



# GIT: Versionando

Muito bem, temos então um ótimo momento para fazermos nossa primeira publicação para o GITHUB.

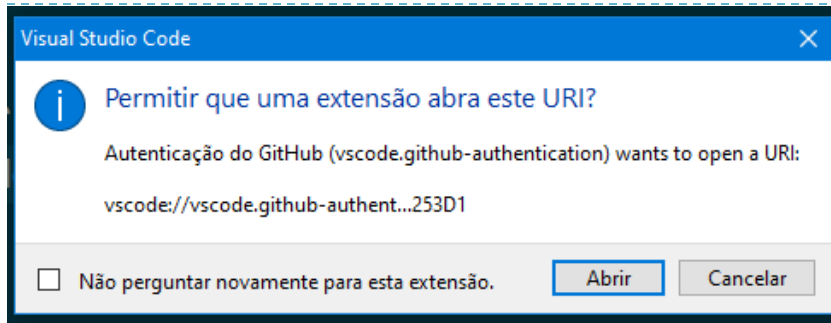
Para isso vamos na sessão repositories e selecionamos Publish master to a remote.



you will redirect to



# GIT: Versionando

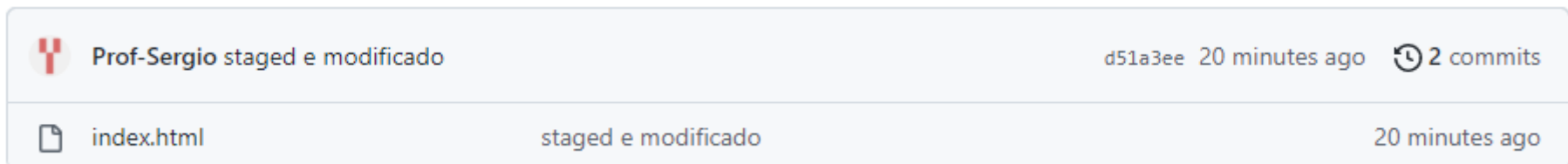


Continuem com as autorizações e escolham o repositório público.



Com isso, nosso arquivo index.html veio para o GITHUB na versão que estava no nosso repositório.

Vamos conferir....



# GIT: Versionando

Vejam, a última alteração que fizemos, colocando çã em modificação, não ficou refletida aqui por conta de não termos efetivado o commit dessa alteração.

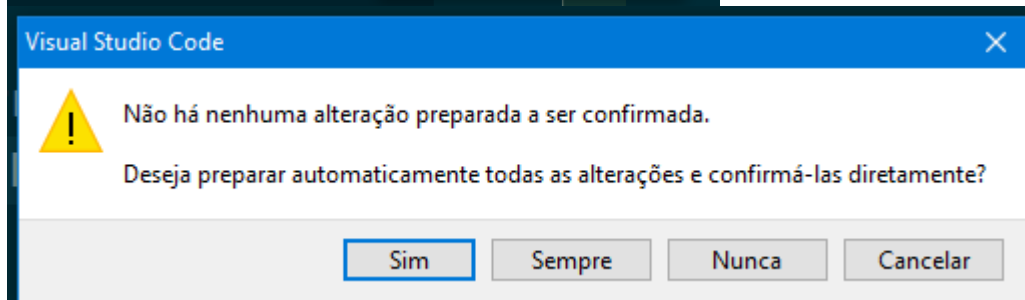
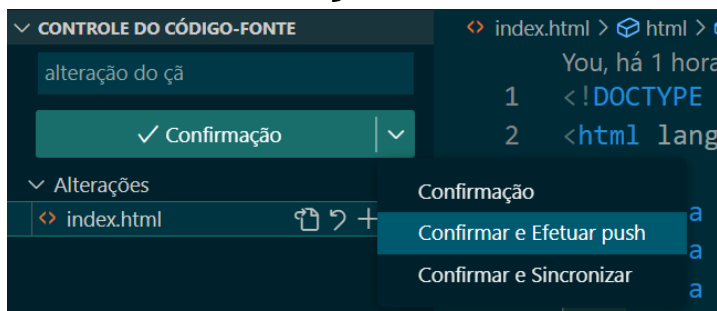
13 lines (13 sloc) | 391 Bytes

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>NoGIT</title>
8  </head>
9  <body>
10     <h3>Primeira versão do nosso HTML!</h3>
11     <h4> Nossa primeira modificacao que será colocadada na área de preparação.</h4>
12 </body>
13 </html>
```

# GIT: Versionando

Poxa vida, fizemos mudanças, não vamos abandoná-las... Vamos efetivar o commit, só que desta vez, vamos efetivar o push para o GITHUB (não esqueçam da mensagem do commit). Ele vai informar que não há nada na nossa área de preparação e pergunta se pode prosseguir de forma automática.

E voilá... Basta atualizar a página do GITHUB para ver as modificações



13 lines (13 sloc) | 393 Bytes

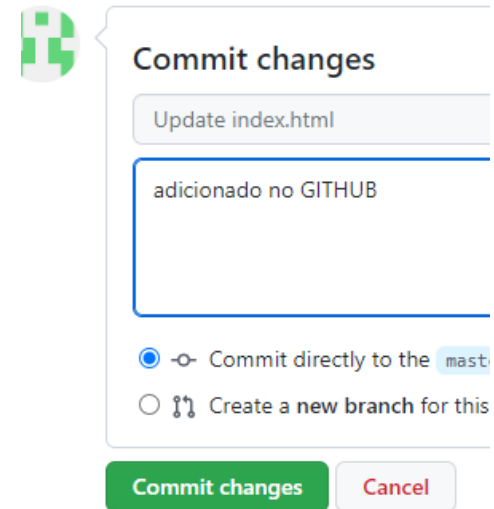
```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" cc
6   <meta name="viewport" content="width=
7   <title>NoGIT</title>
8 </head>
9 <body>
10   <h3>Primeira versão do nosso HTML!</t
11   <h4> Nossa primeira modificação que s
12 </body>
13 </html>
```

# GIT: Versionando

Nós também podemos editar nosso código dentro do próprio GITHUB, para isso, basta selecionarmos o ícone do lápis e editar.

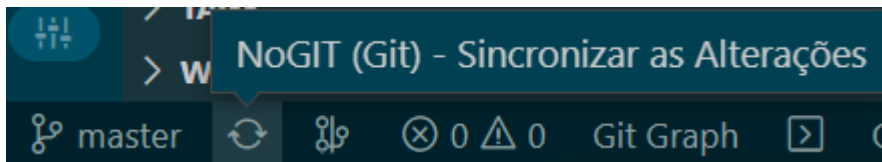
Vamos fazer a inclusão de uma linha, descer e efetivar o commit.

```
10      <h3>Primeira versão do nosso HTML!</h3>
11      <h4> Nossa primeira modificação que será c
12      <h5> Editado no GITHUB </h5>
13  </body>
```



The image shows a GitHub 'Commit changes' dialog box. It has a title bar with a green and white icon. Below the title, there is a text input field containing 'Update index.html'. Below that, there is a larger text area containing 'adicionado no GITHUB'. At the bottom, there are two radio buttons: the first is selected and labeled 'Commit directly to the master branch', and the second is labeled 'Create a new branch for this'. At the very bottom, there are two buttons: 'Commit changes' (green) and 'Cancel' (grey).

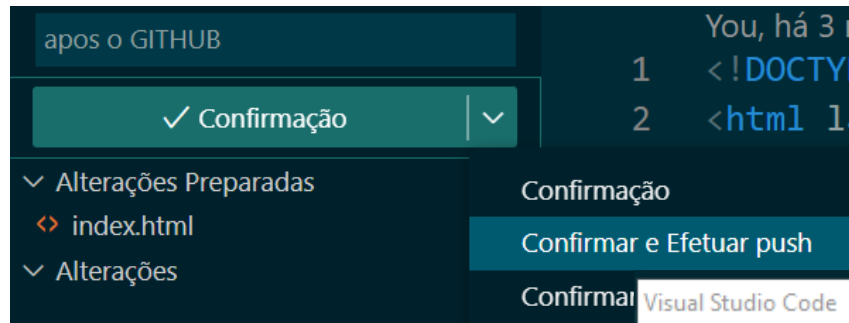

Voltando para o VSC, basta clicarmos no ícone de atualização (canto inferior esquerdo), e nossa linha surgirá



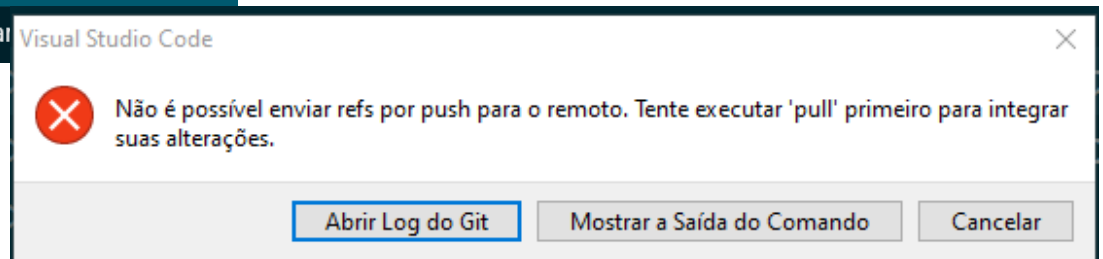
# GIT: Versionando

Mas o que iria acontecer se fizéssemos um commit no GITHUB e outro commit no VSC, e tentássemos mandar a atualização do VSC para o GITHUB?

```
<h3>Primeira versão do nosso HTML! (alterada no GITHUB após commit do VSC)</h3>
<h4> Nossa primeira modificação que será colocada na área de preparação.</h4>
<h5> Editado no GITHUB </h5>
body>
html>
```



Ele não permitirá pois o repositório distribuído está com modificações.



# GIT: Versionando

Isso é o que acontece quando trabalhamos em equipe e um outro desenvolvedor já subiu sua versão.

Para resolver isso, iremos abrir um terminal no VSC (control+shift+') e digitar git pull. Com isso, ele fará o merge automático visto não existir conflito. Depois basta sincronizar.

```
10 <h3>Primeira versão do nosso HTML! (alterada no GITHUB após commit do VSC)</h3>
11 <h4> Nossa primeira modificação que será colocada na área de preparação.</h4>
12 <h5> Editado no GITHUB (e editado antes do VSC)</h5>
13 </body>
```

PROBLEMAS SAÍDA CONSOLE DE DEPURAÇÃO TERMINAL GITLENS cmd +

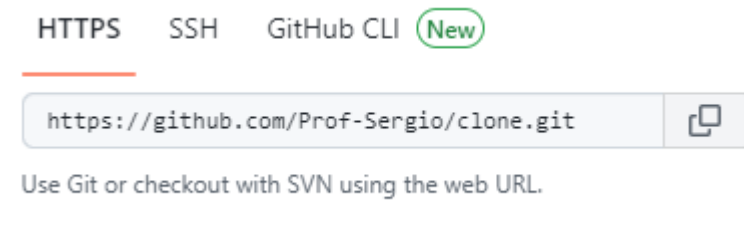
```
C:\Users\Sergio\Dropbox\femass\web\fontes\Clonagem>git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 714 bytes | 47.00 KiB/s, done.
From https://github.com/Prof-Sergio/Clonagem
 86ef069..f2bae52 master    -> origin/master
Auto-merging index.html
Merge made by the 'ort' strategy.
 index.html | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

# GIT: Clonando

---

Agora que já vimos como trabalhamos com um repositório, vejamos como clonar um.

Obtenha o link do repositório a ser clonado;



Crie um novo diretório no seu computador que receberá a clonagem.

Abra o terminal e digite `git clone <endereço>`  
`https://github.com/Prof-Sergio/clone.git`

E com isso finalizamos a clonagem, tendo no seu computador todo o código do projeto.



# GIT: Exercício

---

1. Criem um novo diretório na HD local, e nesse diretório, criem um novo repositório.
2. Efetivem a criação de um novo arquivo e o coloquem na área de preparo, após isso o alterem, verificando todas as suas modificações.
3. Façam o Commit e disponibilizem o código no GITHUB.
4. Alterem o código no GITHUB
5. Atualizem no VSC
6. Façam a clonagem do meu repositório  
<https://github.com/Prof-Sergio/clone.git>

