

Trabalho Prático Individual Final - Git

Desenvolvimento Ágil de Software

Desenvolvimento de Software

Pedro Marques



Índice

Trabalho Prático Individual Final - Git

Introdução	2
1. Criação das Branches para a utilização do modelo GitFlow.	3
2. Níveis de acesso no GitHub	4
3. Revisão de código antes de um pull request	5
4. Adicionar GitIgnore	6
5. Controlar as versões	7-9

Introdução

Devemos começar por criar um repositório local com os seguintes comandos:

```
MINGW64:/c/Users/marques/projetofinal

marques@DESKTOP-F1F0C9I MINGW64 ~
$ mkdir projetofinal

marques@DESKTOP-F1F0C9I MINGW64 ~
$ cd projetofinal

marques@DESKTOP-F1F0C9I MINGW64 ~/projetofinal
$ git init
Initialized empty Git repository in C:/Users/marques/projetofinal/.git/
```

Depois de criado o repositório local devemos acessar o GitHub e criar o nosso repositório em: **Your repositories >>** 

Assim que concluído, para ligar o nosso repositório local ao nosso repositório do remoto devemos digitar o seguinte comando:

```
marques@DESKTOP-F1F0C9I MINGW64 ~/projetofinal (develop)
$ git remote add origin https://github.com/DAS-ProjetoFinal/projetofinal.git
```

1. Criação das Branches para ser possível a utilização do modelo GitFlow.

Para criar todas as branches necessárias e configurar o modelo gitflow devemos começar por digitar o seguinte comando:

```
marques@DESKTOP-F1F0C9I MINGW64 ~/projetofinal (master)
$ git flow init
No branches exist yet. Base branches must be created now.
Branch name for production releases: [master]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/marques/projetofinal/.git/hooks]
```

Quando digitado o comando é recomendado pressionar **ENTER** até ao fim da configuração. Após o procedimento, estaremos por diante de 2 branches: Master e Develop.

2. Níveis de acesso no GitHub

Para criar níveis de acesso no GitHub, teremos de criar uma organização e associar o nosso repositório à organização.

Para criar uma organização devemos acessar

Your organizations >> New Organization

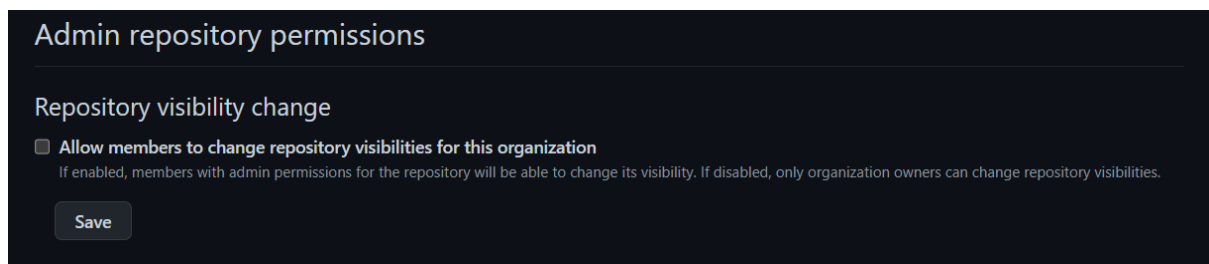
Quando criada devemos transferir o nosso repositório para a nova organização

Your Repositories >> (O repositório) >> Settings >> Transfer ownership

Pronto! Agora é só acessar a nossa nova organização e ir a:

Settings >> Member Privileges

E dar untick da seguinte opção:

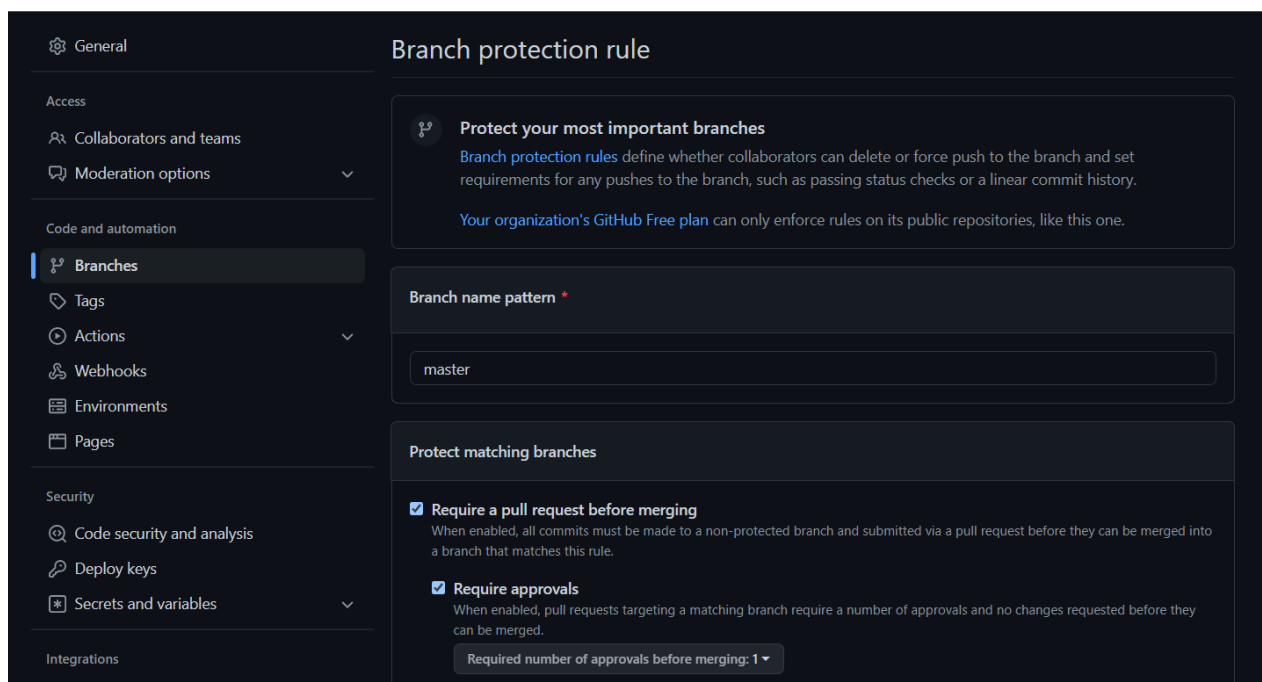


3. Revisão de código antes de um pull request

Para tornarmos obrigatória a revisão de código antes de um pull request é necessário acessarmos:

Your Organization >> (A organização que contém o repositório) >> Repositories >> (O repositório) >> Settings >> Branches >> [Add rule](#)

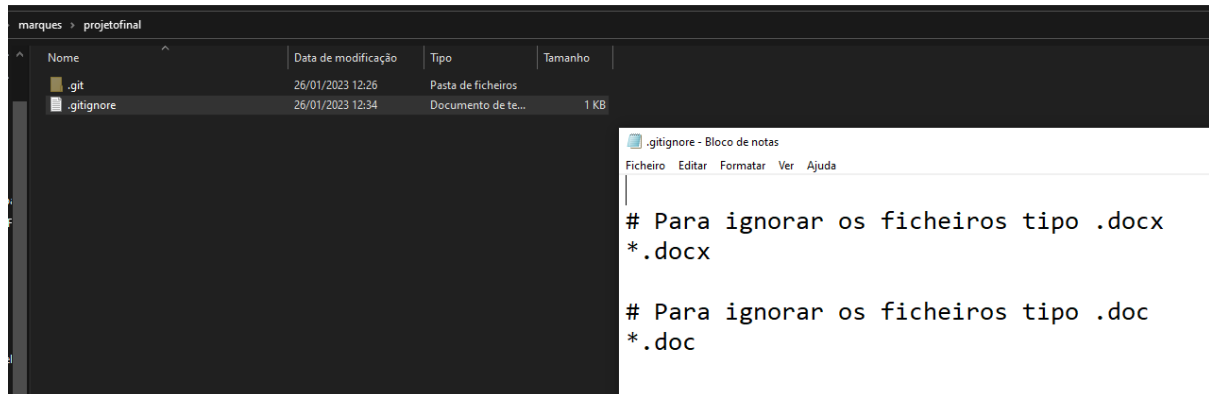
Para aplicar a regra a todas as branches é só aplicá-la na branch master visto que todas as branches provêm desta mesma. Devemos dar tick à regra Require a pull request before merging e Require approvals (escolhendo quantos aprovals queremos na revisão de código para que o pull request seja aprovado)



4. Adicionar Gitignore

Para adicionar um ficheiro `.gitignore` começamos por criá-lo no repositório local e abrir como documento de texto para poder escrever quais ficheiros deseja ignorar.

Para ignorar ficheiros `.docx` e `.doc` devemos escrever o seguinte no ficheiro texto:



Assim feito, vamos agora proceder ao commit e dar push para a branch master

Para isso escrevemos estas linhas de comando:

```
marques@DESKTOP-F1F0C9I MINGW64 ~/projeto-final (master)
$ git add .gitignore

marques@DESKTOP-F1F0C9I MINGW64 ~/projeto-final (master)
$ git commit -m ".gitignore adicionado para ignorar ficheiros .docx e .doc"
[master 9850145] .gitignore adicionado para ignorar ficheiros .docx e .doc
1 file changed, 6 insertions(+)
create mode 100644 .gitignore
```

```
marques@DESKTOP-F1F0C9I MINGW64 ~/projeto-final (master)
$ git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 463 bytes | 463.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/pedrogpmarques/projeto-final
* [new branch]      master -> master
```

5. Controlar versões

A. Carregamento inicial do relatório (template inicial) na branch develop

Para carregarmos o relatório na branch **develop** é simples. Só precisamos dar checkout à branch develop e proceder com os seguintes comandos:

```
marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git add TrabalhoPraticoIndividualFinalGit.pdf

marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git commit -m "Versão inicial do relatório"
[develop 7762b90] Versão inicial do relatório
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 TrabalhoPraticoIndividualFinalGit.pdf

marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git push origin develop
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 48.29 KiB | 24.14 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote: This repository moved. Please use the new location:
remote: https://github.com/DAS-ProjetoFinal/projetoFinal.git
To https://github.com/pedrogpmarques/projetoFinal
3dcabdf..7762b90 develop -> develop
```


B. Alterações no próprio branch develop com os devidos comentários

No ponto B, repetimos basicamente o processo do ponto A, a única diferença é que são dadas alterações em cada commit dado, ao invés de ser adicionado um ficheiro.

Por fim devemos dar o comando:

git push origin develop

para que tudo seja “atualizado” no repositório remoto.

```
marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git add TrabalhoPraticoIndividualFinalGit.pdf

marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git commit -m "Adicionada Introdução"
[develop 442f6ec] Adicionada Introdução
1 file changed, 0 insertions(+), 0 deletions(-)

marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git add TrabalhoPraticoIndividualFinalGit.pdf

marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git commit -m "Adicionado o 1º Ponto"
[develop 06fb688] Adicionado o 1º Ponto
1 file changed, 0 insertions(+), 0 deletions(-)

marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git add TrabalhoPraticoIndividualFinalGit.pdf

marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git commit -m "Adicionado o 2º Ponto"
[develop 5cb15a8] Adicionado o 2º Ponto
1 file changed, 0 insertions(+), 0 deletions(-)

marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git add TrabalhoPraticoIndividualFinalGit.pdf

marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git commit -m "Adicionado o 3º Ponto"
[develop f3b03b1] Adicionado o 3º Ponto
1 file changed, 0 insertions(+), 0 deletions(-)

marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git add TrabalhoPraticoIndividualFinalGit.pdf

marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git commit -m "Adicionado o 4º Ponto"
[develop 47c6bc9] Adicionado o 4º Ponto
1 file changed, 0 insertions(+), 0 deletions(-)

marques@DESKTOP-F1F0C9I MINGW64 ~/projetoFinal (develop)
$ git push origin develop
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 8 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (9/9), 279.27 KiB | 19.95 MiB/s, done.
Total 9 (delta 5), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (5/5), completed with 1 local object.
remote: This repository moved. Please use the new location:
remote: https://github.com/DAS-ProjetoFinal/projetoFinal.git
To https://github.com/pedrogpmarques/projetoFinal
06fb688..47c6bc9 develop -> develop
```

C. Release que contribua para o branch master

Para criar uma release devemos proceder com os seguintes comandos:

```
marques@DESKTOP-F1F0C9I MINGW64 ~/projetofinal (develop)
$ git flow release start REL_1.0 develop
Switched to a new branch 'release/REL_1.0'

Summary of actions:
- A new branch 'release/REL_1.0' was created, based on 'develop'
- You are now on branch 'release/REL_1.0'

Follow-up actions:
- Bump the version number now!
- Start committing last-minute fixes in preparing your release
- When done, run:

    git flow release finish 'REL_1.0'
```

git add TrabalhoPraticoIndividualFinalGit.pdf

git commit -m “Adicionado ponto A B e C”

git flow release finish ‘REL_1.0’ -k

git pull origin master

git push origin master

D. Criar uma Hotfix

```
marques@DESKTOP-F1F0C9I MINGW64 ~/projetofinal (master)
$ git flow hotfix start RELEASE1.0.1 develop
Branches 'develop' and 'origin/develop' have diverged.
And local branch 'develop' is ahead of 'origin/develop'.
Switched to a new branch 'hotfix/RELEASE1.0.1'

Summary of actions:
- A new branch 'hotfix/RELEASE1.0.1' was created, based on 'develop'
- You are now on branch 'hotfix/RELEASE1.0.1'

Follow-up actions:
- Start committing your hot fixes
- Bump the version number now!
- When done, run:

    git flow hotfix finish 'RELEASE1.0.1'
```

git add TrabalhoPraticoIndividualFinalGit.pdf

git commit -m “Adicionado ponto D”

git flow hotfix finish ‘RELEASE1.0.1’

git push origin master