



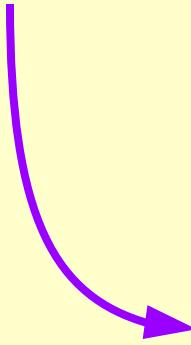
Git e Github

para iniciantes



Alysson Ajackson

Roteiro

- 
- i. Conhecendo o GIT
 - ii. Vocabulário: expressões ~~estranhas~~ e alguns comandos
 - iii. Entendendo um ambiente de desenvolvimento com GIT
 - iv. Criando um repositório
 - v. Alterações e commits
 - vi. Enviando e recebendo (remotes)
 - vii. Conhecendo o GitHub
 - viii. Criando um repositório
 - ix. Compartilhando seu código

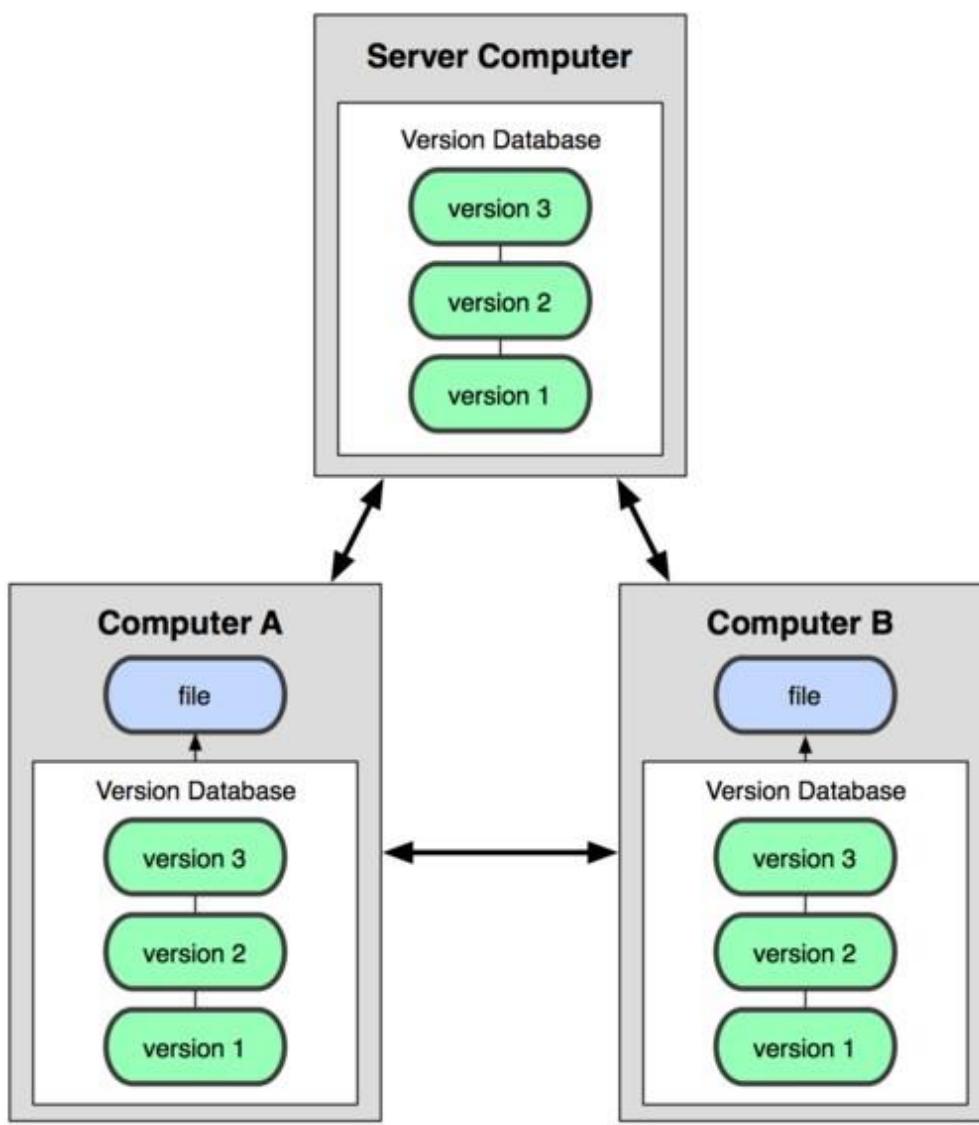
Mas o que é o GIT?

Conhecendo o GIT

Controle de versão?

“O controle de versão é um sistema que registra as mudanças feitas em um arquivo ou um conjunto de arquivos ao longo do tempo de forma que você possa recuperar versões específicas.”

Distribuído?



fonte: git-scm.com

[Vocabulário básico]

- **Repositório git:** Pasta contendo arquivos de um projeto, cujas modificações nesses arquivos são trackeadas pelo GIT.
- **Commit:** cada “salvamento” de uma revisão.
- **Merge:** Junção de 2 “versões” do código.
- **Pull:** Obter código de outro repositório [remoto], fazendo merge com o seu repositório atual.

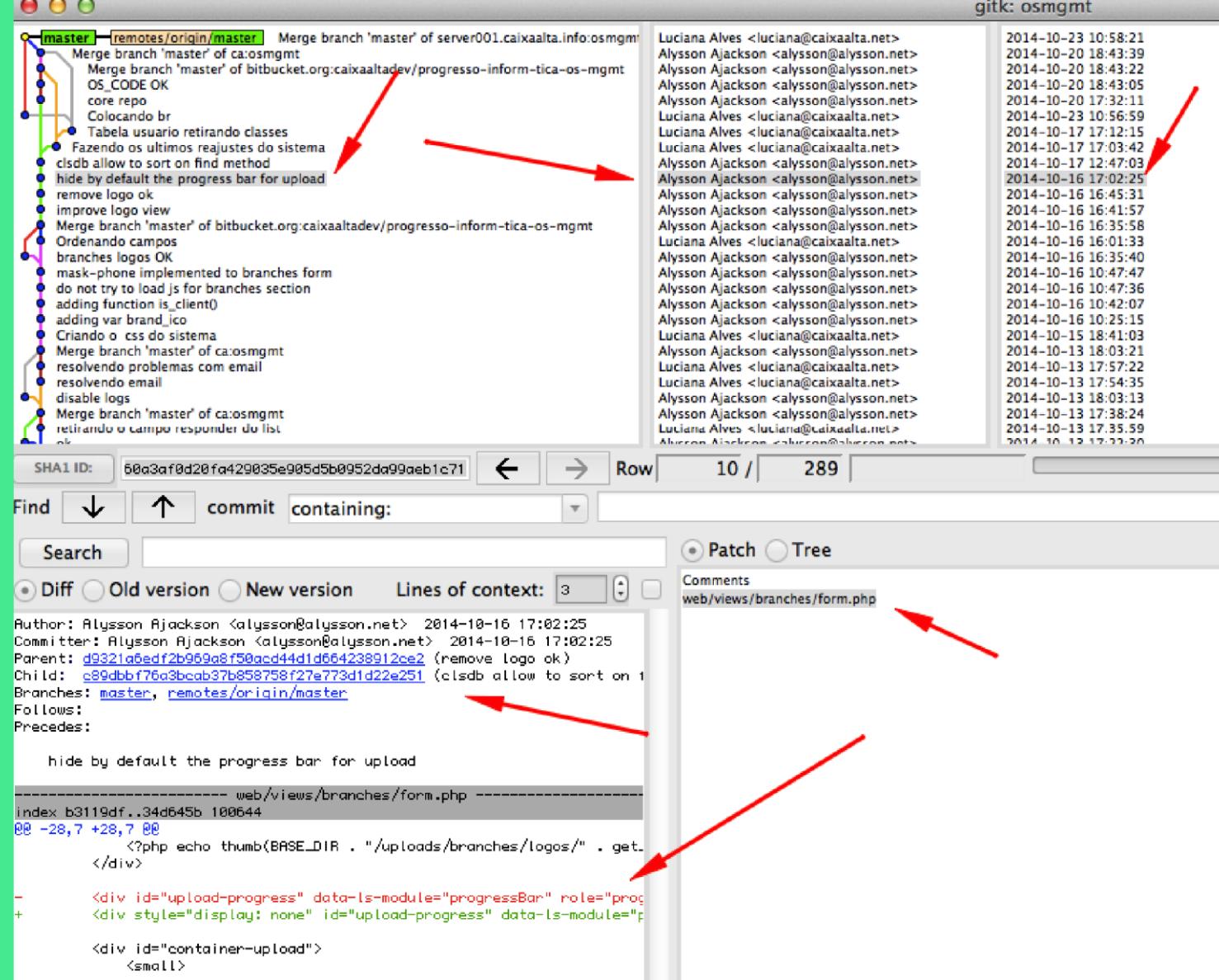
- **Push:** Enviar seu código atual para outro repositório [remoto].

Como o GIT funciona?

Snapshots. E não diferenças [somente].

A cada commit seu, o git tira uma foto do seu projeto!

Árvore de commits



GIT tem integridade!

“Todo no Git tem seu checksum (valor para verificação de integridade) calculado antes que seja armazenado e então passa a ser referenciado pelo checksum. Isso significa que é impossível mudar o conteúdo de qualquer arquivo ou diretório sem que o Git tenha conhecimento.”

fonte: git-scm.com

GIT tem integridade!

“O mecanismo que o Git usa para fazer o checksum é chamado de hash SHA-1, uma string de 40 caracteres composta de caracteres hexadecimais (0-9 e a-f) que é calculado a partir do conteúdo de um arquivo ou estrutura de um diretório no Git.”

Um hash SHA-1 parece com algo mais ou menos assim:

24b9da6552252987aa493b52f8696cd6d3b00373

fonte: git-scm.com

Autor dos commits [não repúdio]

Antes de fazer um commit, é necessário configurar nome e e-mail do usuário (autor).

- (1) `git config --global user.name "Alysson Ajackson"`
- (2) `git config --global user.email "alysson@caixaalta.net"`

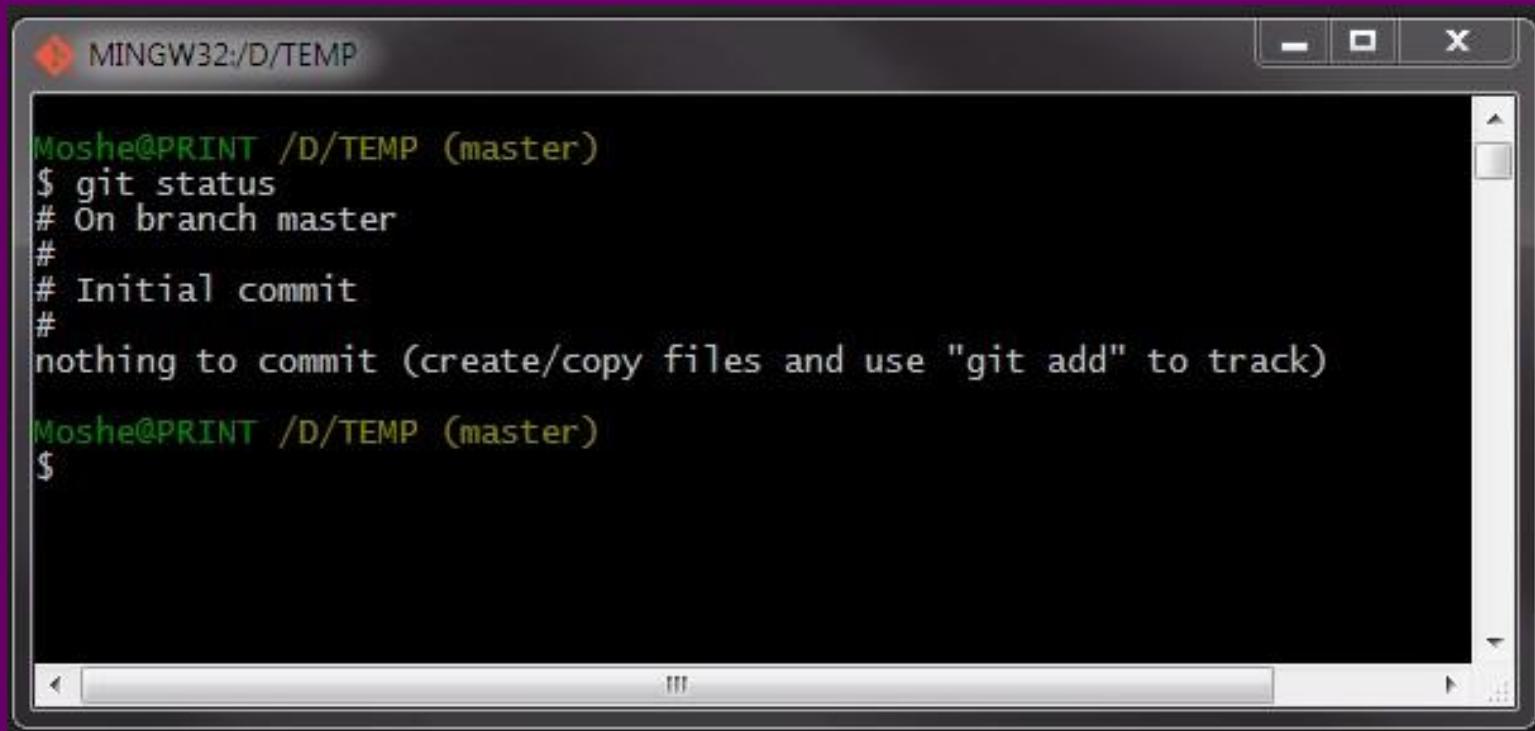
Autenticação e Autenticidade

- O servidor GIT restringe o acesso aos repositórios;
- A autenticação é feita por meio de par de chaves [SSH] ou usuário e senha [HTTPs];
- O Github, por exemplo, permite esses 2 tipos de autenticação.
 - SSH: indicado para computadores não públicos (suas chaves ficam ligadas a ele)
 - HTTP: pode ser utilizado em qualquer computador, sem prévias configurações

Vantagens de se utilizar GIT

- ✓ Backups!
- ✓ Controle de versões
- ✓ Integração
- ✓ Agilidade
- ✓ Segurança
- ✓ Não repúdio
- ✓ Não há perda de dados
- ✓ Fácil configuração de um novo ambiente
- ✓ Multiplataforma

Como utilizar o GIT

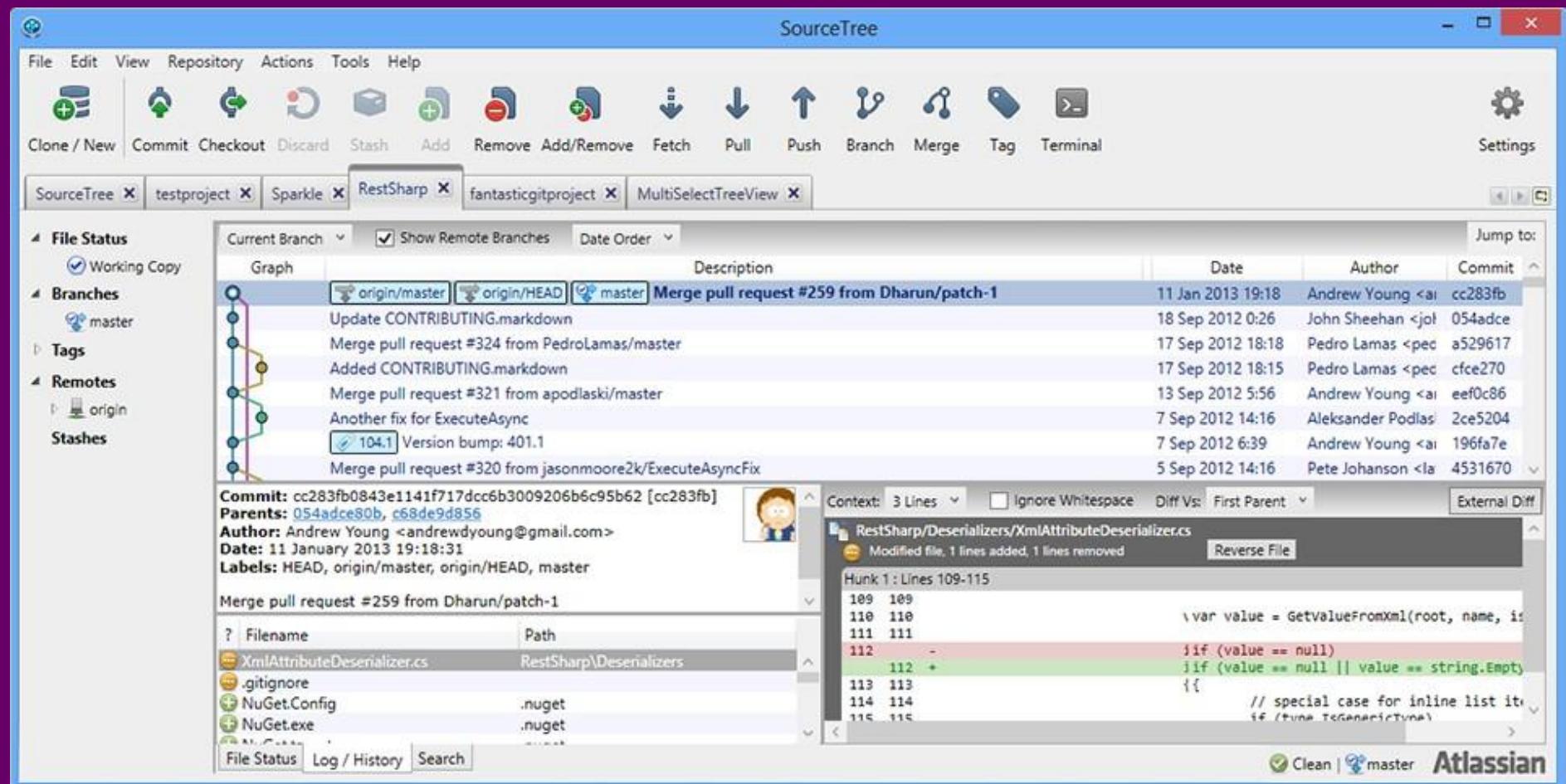


A screenshot of a terminal window titled "MINGW32:/D/TEMP". The window contains the following text:

```
Moshe@PRINT /D/TEMP (master)
$ git status
# On branch master
#
# Initial commit
#
nothing to commit (create/copy files and use "git add" to track)

Moshe@PRINT /D/TEMP (master)
$
```

Opção 1 – Terminal (linhas de comando)



Opção 2 – SourcetreeApp (for Windows and MAC OS) - by bitbucket.org

git - SmartGit/Hg 4.6 (evaluation until Aug 4)

git - Log for .

Review

Pull Sync

Check Out Add Tag Add Branch Merge Cherry-Pick Revert Rebase HEAD to Reset

Directories x git (master) Branches x Commits (895) Details x

HEAD (master) develop master = origin Local Branches origin (5) - https://github.com Tags (474) Stashes Lost Heads

Message Date Author

develop origin master Merge branch 'maint' 01:43 AM Junio C Hamano

git-remote-mediawiki: un-brace file handles in binmode calls Yesterday 12:14 PM Matthieu Moy

[origin/maint] Update draft release notes to 1.8.3.3 01:43 AM Junio C Hamano

Merge branch 'rr/diffcore-pickaxe-doc' into maint 01:41 AM Junio C Hamano

Merge branch 'cr/git-work-tree-sans-git-dir' into maint 01:41 AM Junio C Hamano

Merge branch 'fc/do-not-use-the-index-in-add-to-index' into maint 01:40 AM Junio C Hamano

Merge branch 'dm/unbash-subtree' into maint 01:39 AM Junio C Hamano

Merge branch 'jc/core-checkstat' into maint 01:39 AM Junio C Hamano

Merge branch 'jc/t5551-posix-sed-bre' into maint 01:37 AM Junio C Hamano

Merge branch 'vv/help-unknown-ref' into maint 01:37 AM Junio C Hamano

Merge branch 'rs/empty-archive' into maint 01:36 AM Junio C Hamano

Merge branch 'rh/merge-options-doc-fix' into maint 01:36 AM Junio C Hamano

Merge branch 'an/diff-index-doc' into maint 01:35 AM Junio C Hamano

Merge branch 'cm/qitweb-project-list-persistent-cqi-fix' into maint 01:31 AM Junio C Hamano

Junio C Hamano, 07/04/2013 01:43 AM

Merge branch 'maint'

Files (1) x Name Modification git-config.txt Modified

Changes of git-config.txt x

Default behavior is to replace at most one line. This replaces all lines matching the key (and optionally the value_regex).

--add:: Adds a new line to the option without altering any existing values. This is the same as providing '^\$' as the value_regex in '--replace-all'.

--get:: Get the value for a given key (optionally filtered by a regex matching the value). Returns error code 1 if the key was not found and error code 2 if multiple key values were found.

--get-all:: Like get, but does not fail if the number of values for the key is not exactly one.

--get-regexp:: Like --get-all, but interprets the name as a regular expression and writes out the key names. Regular expression matching is currently case-sensitive and done against a canonicalized version of the key in which section and variable names are lowercased, but subsection

Default behavior is to replace at most one line. This replaces all lines matching the key (and optionally the value_regex).

--add:: Adds a new line to the option without altering any existing values. This is the same as providing '^\$' as the value_regex in '--replace-all'.

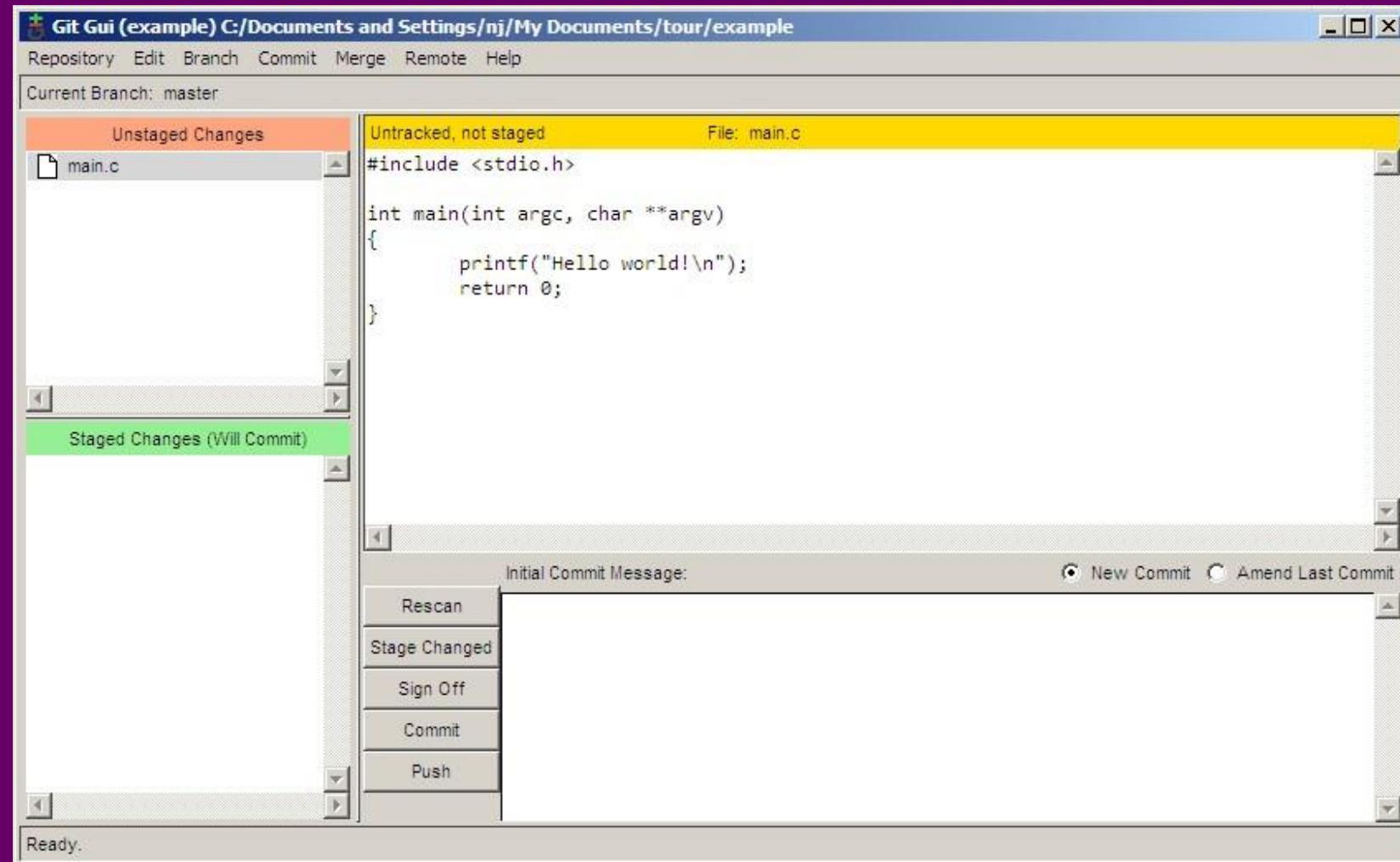
--get:: Get the value for a given key (optionally filtered by a regex matching the value). Returns error code 1 if the key was not found and the last value if multiple key values were found.

--get-all:: Like get, but does not fail if the number of values for the key is not exactly one.

--get-regexp:: Like --get-all, but interprets the name as a regular expression and writes out the key names. Regular expression matching is currently case-sensitive and done against a canonicalized version of the key in which section and variable names are lowercased, but subsection

Ready

Opcão 3 – SmartGit – For Windows, Linux and MAC OS (free for non-commercial use)



The screenshot shows the GitHub for Windows application interface. On the left is a sidebar with a list of repositories: GitHub, Akavache, atom, GitPad, libgit2, libgit2sharp, octokit.net (which is selected and highlighted in blue), ReactiveUI, SeeGit, Windows, windows.github.com, Enterprise, and tutorial. The main area displays a list of commits under the 'History' tab for the 'master' branch. The commits are:

- Merge pull request #481 from octokit/sort-is-actually-case-sensitive 18 days ago by Phil Haack
- cache enum label 18 days ago by Brendan Forster
- test fields 18 days ago by Brendan Forster
- removed redundant attributes from issue search 18 days ago by Brendan Forster
- if the attribute cannot be found, lowercase the enum 18 days ago by Brendan Forster
- bugfix - the State value when searching for issues i... 22 days ago by Brendan Forster
- Merge pull request #479 from octokit/fix-async-na... 22 days ago by Phil Haack

On the right, a detailed view of the merge commit for pull request #481 is shown. The commit message is "Merge pull request #481 from octokit/sort-is-actually-case-sensitive". It was made by Phil Haack (aad94fe) and includes the note "searching for issues hits a case-sensitive field". The file being edited is "Octokit.Tests.Integration\Clients\SearchClientTests.cs". The code changes are as follows:

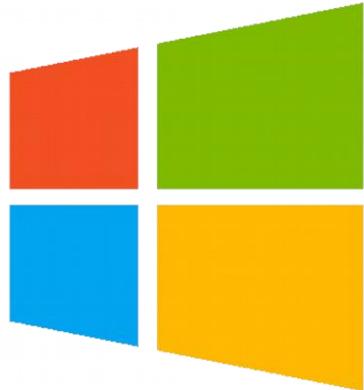
```
@@ -54,4 +54,16 @@ public class SearchClientTests
54
55     Assert.NotEmpty(repos.Items);
56 }
57 +
58 [Fact]
59 public async Task SearchForOpenIssues()
60 {
61     var request = new SearchIssuesRequest("phone");
62     request.Repo = "caliburn-micro/caliburn.micro";
63     request.State = ItemState.Open;
64 +
65     var issues = await _githubClient.Search.SearchIssues(request);
66 +
67     Assert.NotEmpty(issues.Items);
68 }
```

The status bar at the bottom indicates "Sync 12" and "Revert".

**Um ambiente de
desenvolvimento
o com GIT**

Prática!

Obtendo e instalando o GIT



<http://git-scm.com/downloads>

Obtendo e instalando o GIT no Linux



\$ sudo apt-get install git

\$ sudo yum install git ...

Conhecendo o terminal

cd → mudar de pasta

ls → mostrar conteúdo da pasta

pwd → em que pasta estou?

mkdir “projeto” → cria pasta “projeto”

rm x → apaga arquivo x

rm -r y → apaga pasta y

cd [tab][tab] #mostra opções [no caso, pastas] possíveis



Funciona para qualquer comando no linux

Workflow básico do GIT

- (1) Você modifica arquivos no seu diretório de trabalho.
- (2) Você seleciona os arquivos, adicionando snapshots deles para sua área de preparação .
- (3) Você faz um commit , que leva os arquivos como eles estão na sua área de preparação..

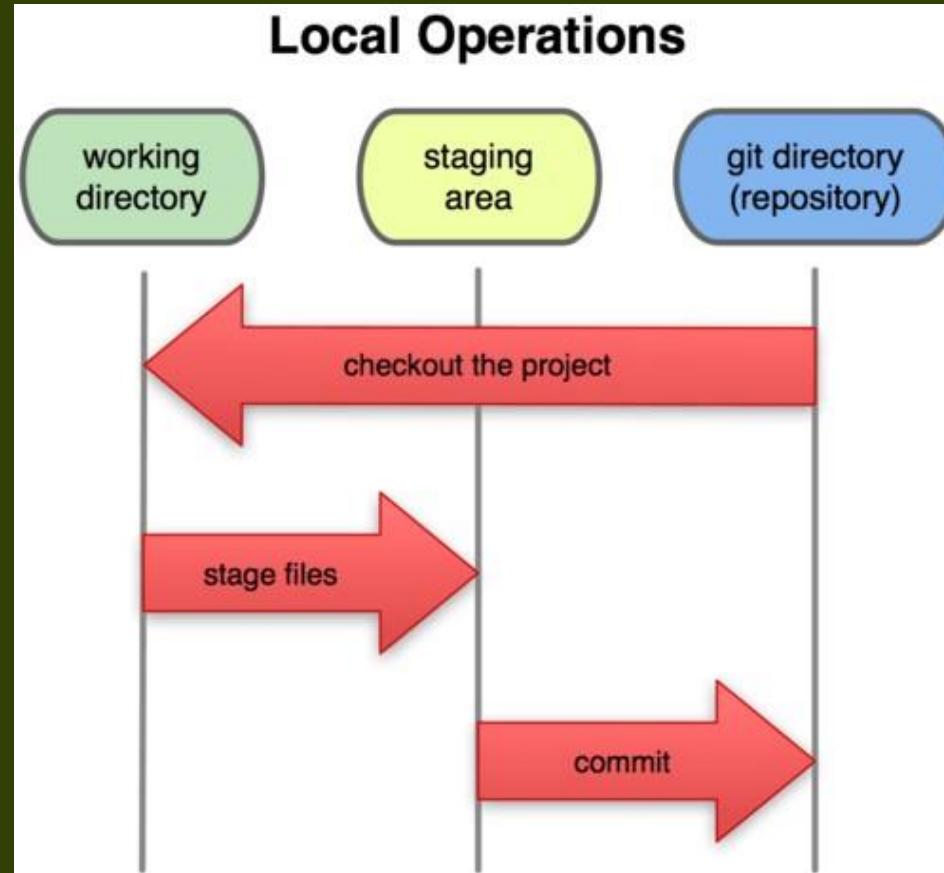
(armazena informações de como será seu próximo commit)

Hello world.



Criando o seu primeiro
repositório

Os 3 estados do GIT



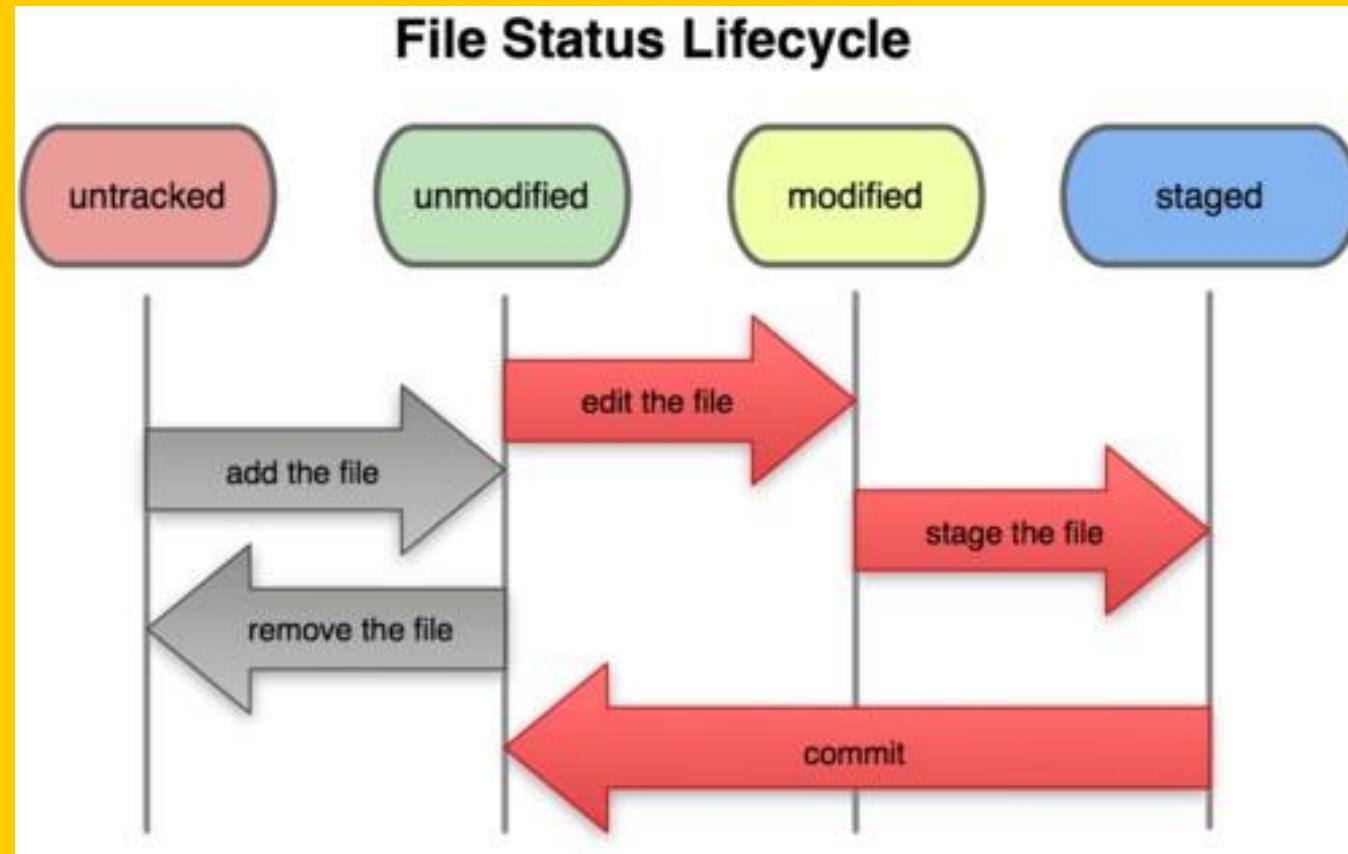
Ignorando arquivos

- (1) Crie um arquivo `.gitignore`, na raiz do seu projeto
- (2) Indique os arquivos a serem ignorados pelo GIT.

Ex.:

```
/config/database.php #ignora o arquivo que contém senha e dados de acesso ao BD  
/uploads #ignora a pasta uploads que está na raiz  
*.bk #ignora quaisquer arquivos com a extensão .bk
```

Ciclo de vida dos status de um arquivo



```
git commit -m \  
“Iniciando projeto”
```



fazendo o seu primeiro
commit

Criando um repositório

- (1) Criar pasta
- (2) Acessar a pasta via terminal → cd /meu/projeto
- (3) git init [enter]
- (4) Adicione arquivos
- (5) Faça commits

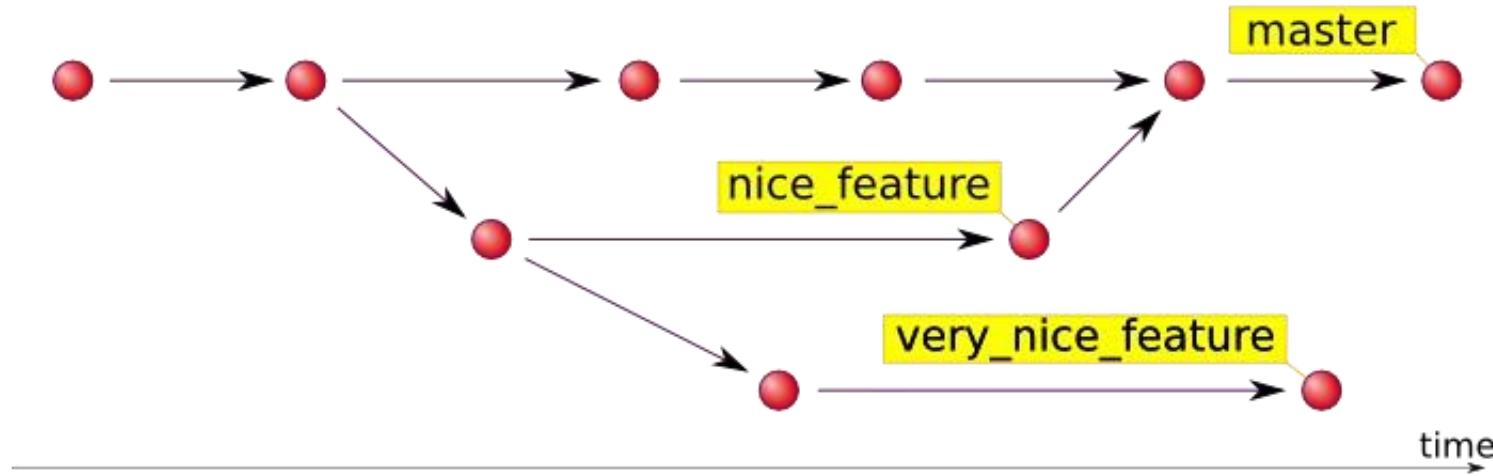
Alterações e commits

- (1) Crie arquivo(s) em seu projeto
- (2) git status
- (3) git add [nome_do_arquivo]
- (4) git status
- (5) git commit -m “mensagem de commit”

Alterações e commits

- (1) Altere o conteúdo do(s) arquivo(s) em seu projeto
- (2) `git status`
- (3) `git diff`
- (4) `git add [nome_do_arquivo]`
- (5) `git status`
- (6) `git commit -m "mensagem de commit"`

Trabalhando com branches



Desfazendo alterações

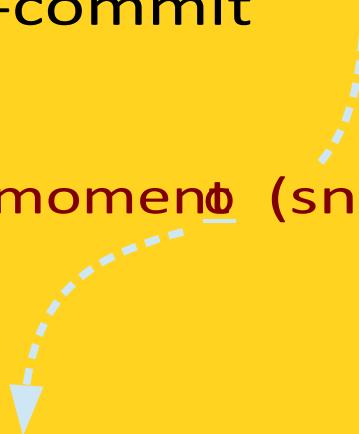
(1) `git checkout -- arquivo`

```
# desfaz as alterações feitas (ainda não  
arquivo indicado
```

Alterando entre versões

(1) `git checkout hash-do-commit`

retorna o repositório ao momento (snapshot) indicado pelo
hash do commit



/* Você pode indicar um nome de arquivo nesse comando,
para voltar a versão de um arquivo específico */

Criando um branch

- (1) `git checkout -b "funcionalidade"`
- (2) `git status` #perceba que o branch mudou!
- (3) Crie arquivos ou faça alterações
- (4) Faça commits (`git add...git commit`)
- (5) `git status` #perceba que os commits estão OK
- (6) `git checkout master` #voltamos ao branch principal

Merge! [branches locais]

- (1) git checkout “funcionalidade” #agora sem o -b
- (2) git status #perceba que o branch mudou!
- (3) Crie arquivos ou faça alterações
- (4) Commits (git add...git commit) #funcionalidade OK
- (5) git status #perceba que os commits estão OK
- (6) git checkout master #voltamos ao branch principal
- (7) git merge “funcionalidade” “master” #merge OK

Configurando atalhos [terminal]

- `git config alias.s status#git status` virou 'git s'
- `git config alias.c commit#git status` virou 'git c'
- ...

Repositórios remotos [servidor GIT]



Atlassian



github
SOCIAL CODING





Search GitHub

Explore Gist Blog Help



alyssonajackson

+ ▾ ⌂ ⚙ ⌚

Owner



alyssonajackson ▾

Repository name

project-test-01



Great repository names are short and memorable. Need inspiration? How about [psychic-octo-tyrion](#).

Description (optional)

this is a test project

 **Public**

Anyone can see this repository. You choose who can commit.

 Private

You choose who can see and commit to this repository.

 Initialize this repository with a README

This will allow you to git clone the repository immediately. Skip this step if you have already run git init locally.

Add .gitignore: **None** ▾Add a license: **None** ▾**Create repository**



Quick setup — if you've done this kind of thing before

Set up in Desktop

or

HTTP

SSH

git@github.com:alyssonajackson/project-test-01.git



We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
touch README.md  
git init  
git add README.md  
git commit -m "first commit"  
git remote add origin git@github.com:alyssonajackson/project-test-01.git  
git push -u origin master
```



...or push an existing repository from the command line

```
git remote add origin git@github.com:alyssonajackson/project-test-01.git  
git push -u origin master
```



Code

Issues

Pull Requests

Wiki

Pulse

Graphs

Settings

Configurando suas chaves [RSA] para acesso via SSH

- (1) No terminal, digite: ssh-keygen [enter]
- (2) [enter] para confirmar o local de criação da chave
- (3) Digite uma passphrase para essa chave (se deixar em branco não será necessário digitar a senha toda vez que quiser enviar ou receber código do github)
- (4) Par de chaves criado!



alyssonajackson

Profile

Account settings

Emails

Notification center

Billing

SSH keys

Security

Applications

Repositories

Organization membership

Need help? Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH Problems](#)

SSH Keys

[Add SSH key](#)

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

- alysson@home-winxp.pub**
ab:ed:f3:0c:39:66:c9:d8:4d:a7:f8:f0:36:2c:4f:6f
Added on Mar 17, 2012 — ⓘ No recent activity [Delete](#)
- alysson@home-toshiba-win7**
20:75:e6:6f:34:44:64:16:90:d0:c4:d0:03:64:44:fc
Added on Oct 24, 2012 — ⓘ No recent activity [Delete](#)
- alysson@home-toshiba-debian**
f9:48:21:e9:a3:24:8c:c2:2c:93:01:e3:e3:e9:44:4c
Added on Mar 13, 2013 — Last used on Sep 1, 2014 [Delete](#)
- www-data@host02.constru.it**
26:8f:a2:aa:d6:2c:24:50:5a:36:9d:04:39:5a:74:f1
Added on Jul 22, 2014 — Last used on Jul 22, 2014 [Delete](#)
- faculdadeunibhlab19-temp**
72:ee:ef:13:e0:44:d7:f3:83:03:e7:ff:10:38:64:42
Added on Oct 24, 2014 — Last used on Oct 24, 2014 [Delete](#)
- alyssons-macbook-air**
64:5f:85:58:91:14:6f:6d:83:2f:10:bb:92:dc:84:18
Added on Oct 26, 2014 — Never used [Delete](#)

Add an SSH Key

Title

Identificador da chave (ex: nome do pc)

Key

Cole sua chave aqui

Add key

Chaves adicionadas.

Volte à tela do seu repositório no GitHub.

A opção clonar via SSH deve estar habilitada.

Adicionando o repositório do github como remoto!

Voltando ao terminal...

- (1) git remote -v #não temos remoto algum
- (2) git remote add origin "url-do-repo-github[enter]"
- (3) git remote -v #temos um repositório remoto!

Enviando e recebendo código seu PC↔ GitHub

Ainda no terminal...

- (1) git pull origin master #atualizando o branch master [local], a partir do origin
- (2) git push origin master #enviando o branch master p/ origin

Lidando com conflitos!

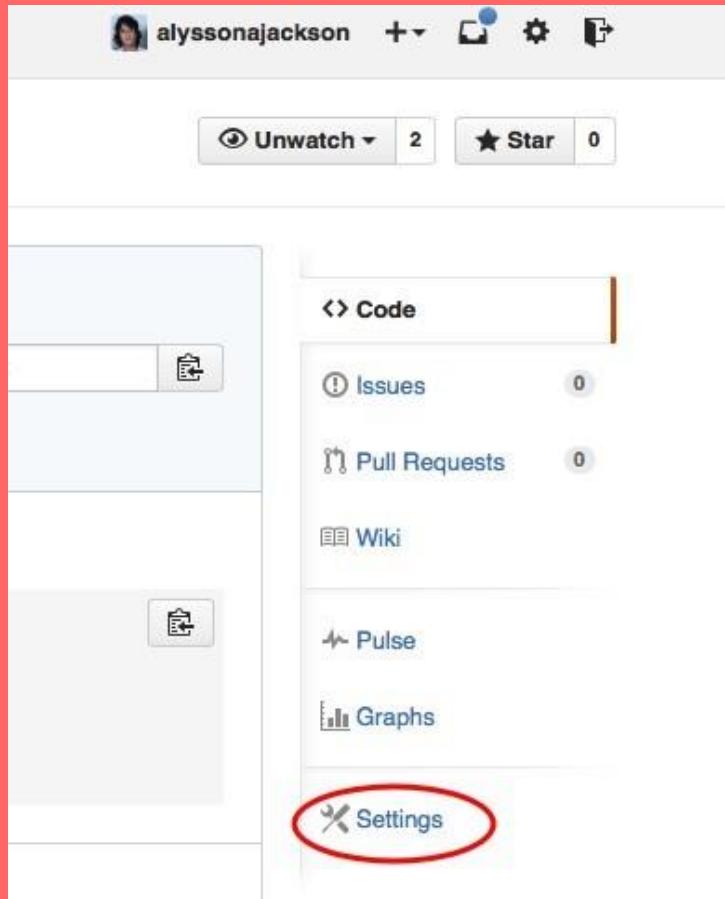
```
1 <<<<< HEAD
2
3 Here is the original change.
4 =====
5 Here is the modified change.
6 >>>>> 58326c301d09b58f3ac23d616e73f7b478424cc5
7
```

Quando isso ocorrer:

Basta reajustar o arquivo, fazer o commit, e pronto!

Você já pode dar push normalmente!

Trabalhando com colaboradores [github]



Trabalhando com colaboradores [github]

The screenshot shows a GitHub repository page for 'alyssonajackson / project-test-01'. The left sidebar has options: Options, **Collaborators** (circled in red), Webhooks & Services, and Deploy keys. The main area is titled 'Collaborators' and says 'This repository currently has no collaborators.' It features a 'Type a username' input field and an 'Add collaborator' button. A red arrow points from the 'Type a username' field to the 'Add collaborator' button. The top right shows 'Unwatch' (2 notifications), 'Star' (0 stars), and other repository stats.

Exercício

- Criar um projeto adicionar colaboradores,
- Membro cria um resumo da interface github
- Crie resumo dos comandos mais usados e
- Os resumos devem estar dentro da pasta resumo, deve-se também criar um ramo para cada resumo e depois fazer merge com máster

Enviar um link do repositório para a.sousajose@gmail.com com assunto engenharia de software