



Apresentação



Gerência de Configuração e de Mudança

Prof. MSc. Marcos Miguel

marcos.a.miguel@gmail.com

<https://sites.google.com/site/marcosamiguel>

Quem sou eu?

- Mestre em Ciência da Computação pela UFJF
 - MBI (Master in Business Intelligence) pela UFJF
 - Bacharel em Sistemas de Informação pelo Centro de Ensino Superior de Juiz de Fora
 - Diretor de desenvolvimento da Projetus TI
 - Professor do Centro de Ensino Superior de JF (CES)
-
- Experiência na área de Ciência da Computação, com ênfase em software, atuando principalmente nos seguintes temas: Java, Groovy, Jax-ws, Jax-RS, Web 2.0, Struts, Spring, Flex, Delphi, SQL-Server, MySQL, PostgreSQL, Qualidade de Software, Refatoração, Testes, Metodologias Ágeis e Estimativa de Esforço/Tempo

Plano de Ensino

- Gerência de configuração de software.
- Gerência de Versão.
- Gerência de Mudanças.
- Utilização de ferramentas de controle de versões e para gerência de mudanças.

Objetivos

- Desenvolver nos estudantes os conhecimentos e habilidades necessárias para o gerenciamento de configurações e de mudanças em projetos de software, utilizando as ferramentas, práticas e estratégias adequadas de acordo com a natureza do projeto, bem como a familiarização com a prática e aplicação de ferramentas utilizadas no mercado.

Conteúdo Programático

1. Gerência de configuração de software.
2. Gerência de Versão.
3. Ferramentas de Controle de Versão.
4. Gerência de Mudanças.
5. Ferramentas de Controle de Mudanças.
6. Integração contínua
7. Ferramentas de Integração Contínua.
8. Estudos de Casos

Procedimentos Didáticos

Aulas expositivas e práticas. Avaliações escritas.
Trabalhos em grupo. Recursos de projeção e laboratório
de informática. Trabalhos de pesquisa. Seminários.

Avaliações

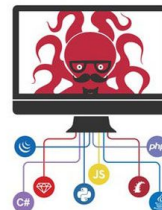
- 2 Provas de 30 pts.
- 1 Trabalho 40 pts.
- 2ª Chamada deverá ser requisitada na secretaria e somente de uma das atividades. É realizada ao final do período e contempla a matéria TODA

Bibliografia

- KOSCIANSKI, A., SOARES, M., S. **Qualidade de Software**. 2ª Edição. Novatec. 2007.
- AQUILES, Alexandre, FERREIRA, Rodrigo. **Controlando versões com Git e GitHub**. Casa do Código, 2014.
- BOAGLIO, Fernando. **Jenkins - Automatize tudo sem complicações**. Casa do Código, 2016.



Controlando vers es com
Git e GitHub



Casa do
C digo

ALEXANDRE
FERREIRA

Jenkins

Automatize tudo sem
complica es



Casa do
C digo

FERNANDO BOAGLIO

Bibliografia Complementar

- TORRES, Joaquim. **Gestão de produtos de software - Como aumentar as chances de sucesso do seu software**. Casa do Código, 2015.
- TORRE, S. **Aprender com os erros: o erro como estratégia de mudança**. Porto Alegre: Artmed, 2007.
- HIRAMA, Kechi. **Engenharia De Software - Qualidade e Produtividade com Tecnologia**. Elsevier, 2011.
- MICROSOFT, Time de Suporte Microsoft Modern Apps, **Desenvolvimento efetivo na plataforma Microsoft Como desenvolver e suportar software que funciona**. Casa do Código, 2016.
- Romero, Daniel. **Containers com Docker - Do desenvolvimento à produção**. Casa do Código, 2015.

Frequência

- Disciplina de 2 créditos
 - 30 aulas
 - Frequência mínima de 75%

Dúvidas?



Mudanças no desenvolvimento de software

- Mudanças:

- não são analisadas antes de ser feitas
- não são registradas antes de ser implementadas
- não são relatadas àqueles que precisam saber
- ou não são controladas de forma que melhore a qualidade e reduza erros



Gerenciamento de Configuração e mudanças

- Durante o desenvolvimento do software queremos saber:
 - O que mudou e quando? **(controle de versão)**
 - Por que mudou? **(controle de mudanças)**
 - Quem fez a mudança? **(auditoria de configuração)**
 - Podemos reproduzir esta mudança? **(auditoria de configuração)**

Gerenciamento de Configuração e mudanças

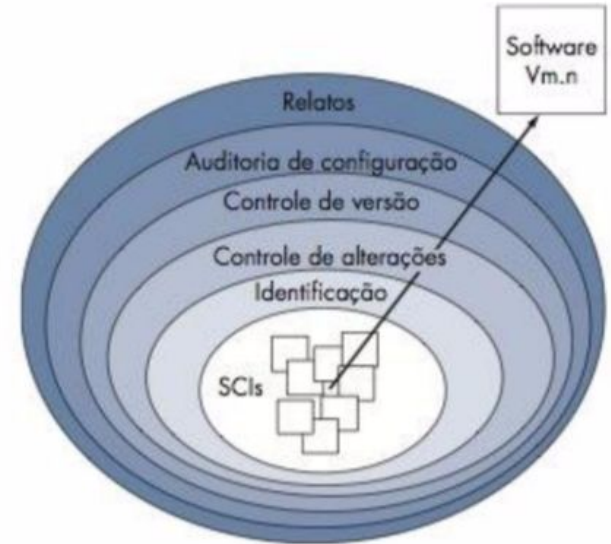
- Coordenar o desenvolvimento para minimizar a confusão!

*“Arte de **identificar**, **organizar** e **controlar modificações** no software que está sendo criado, maximizando a produtividade e reduzindo os erros” [Pressman, 2011]*

Gerenciamento de Configuração e mudanças

As atividades são desenvolvidas para:

- Identificar a alteração
- Controlar a alteração
- Assegurar que a alteração seja implementada corretamente
- Relatar as alterações aos outros interessados



O que é uma Configuração?

Configuração de um sistema é uma coleção de **versões específicas** de itens de configuração (hardware, firmware ou software) que são combinados de acordo com procedimentos específicos de construção para servir a uma finalidade particular.

Item de configuração: Elemento unitário ou um grupo de elementos para efeito de controle de versão.

- Código
- Documentação
- Diagramas, planos, ferramentas, casos de teste e etc

Todos os documentos que podem ser úteis para a evolução futura do sistema

Baseline

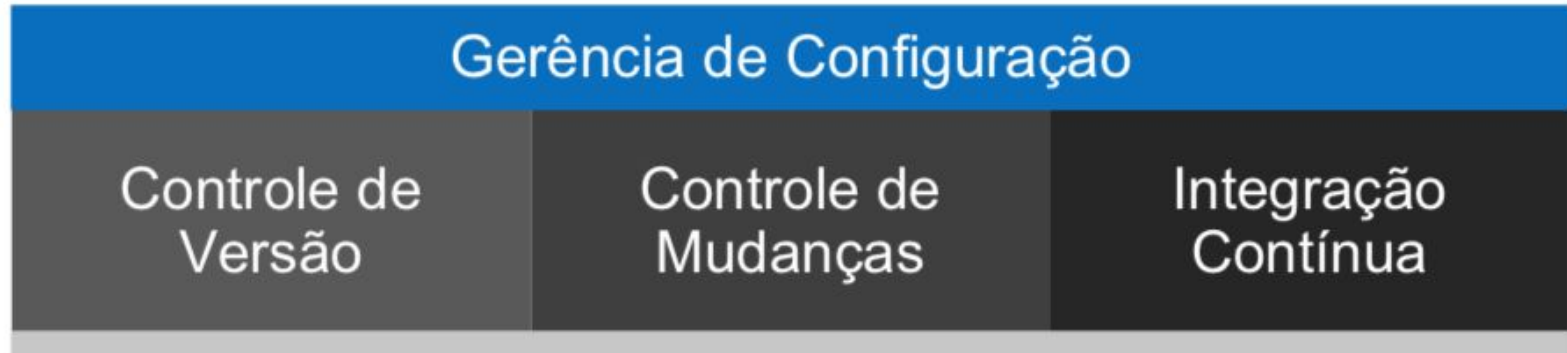
Baseline: configuração formalmente aprovada que servirá de referência para desenvolvimento posterior

A configuração do software em um ponto discreto no tempo

Quando um conjunto de artefatos de software se torna um item de configuração?



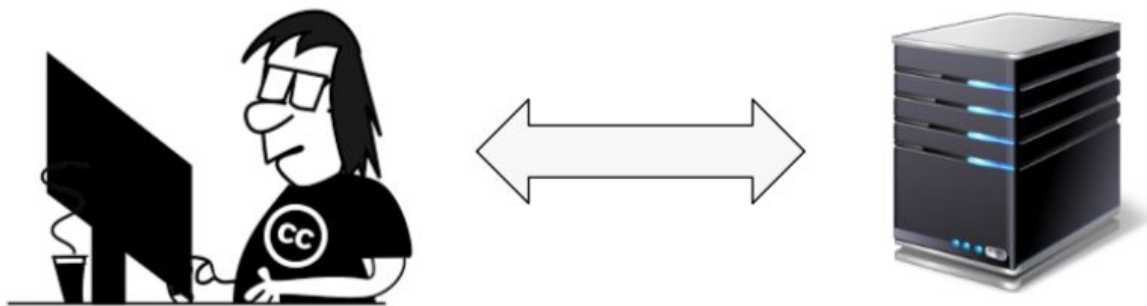
Qual a importância do controle de versões?



Exemplo 1

Você precisa editar um site hospedado em um servidor:

- 1) Você faz o download via FTP
- 2) Faz as alterações necessárias
- 3) Envia os arquivos alterados para o servidor via FTP



Exemplo 1

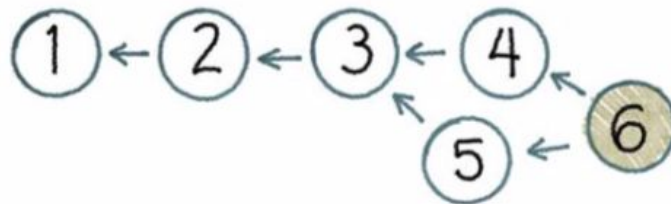
Agora outro desenvolvedor também deve fazer alterações no site...

- 1) Baixa o mesmo arquivo junto com você
- 2) Edita e manda para o servidor depois de você
- 3) Sobrescreve suas alterações!!!



Controle de Versões

- Principais objetivos do Controle de Versões do Sistema [Sink, 2011]
 - Trabalhar simultaneamente
 - Evitar conflitos em alterações
 - Manter versões





git

Visão geral

O que é Git?

- Sistema de Controle de Versão Distribuído
- Criado por Linus Torvalds (2005)
- Auxiliar no Desenvolvimento do Linux

O que é Github?

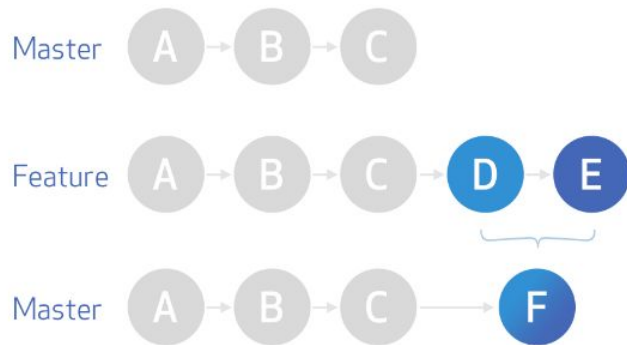
- Serviço de Web Hosting compartilhado para projetos que usam o controle de versionamento Git



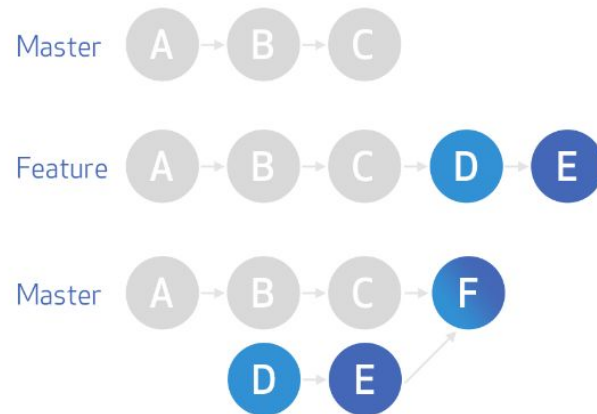
Companies & Projects Using Git



Controle de Versões Distribuído



Squash and merge: D + E combined into F



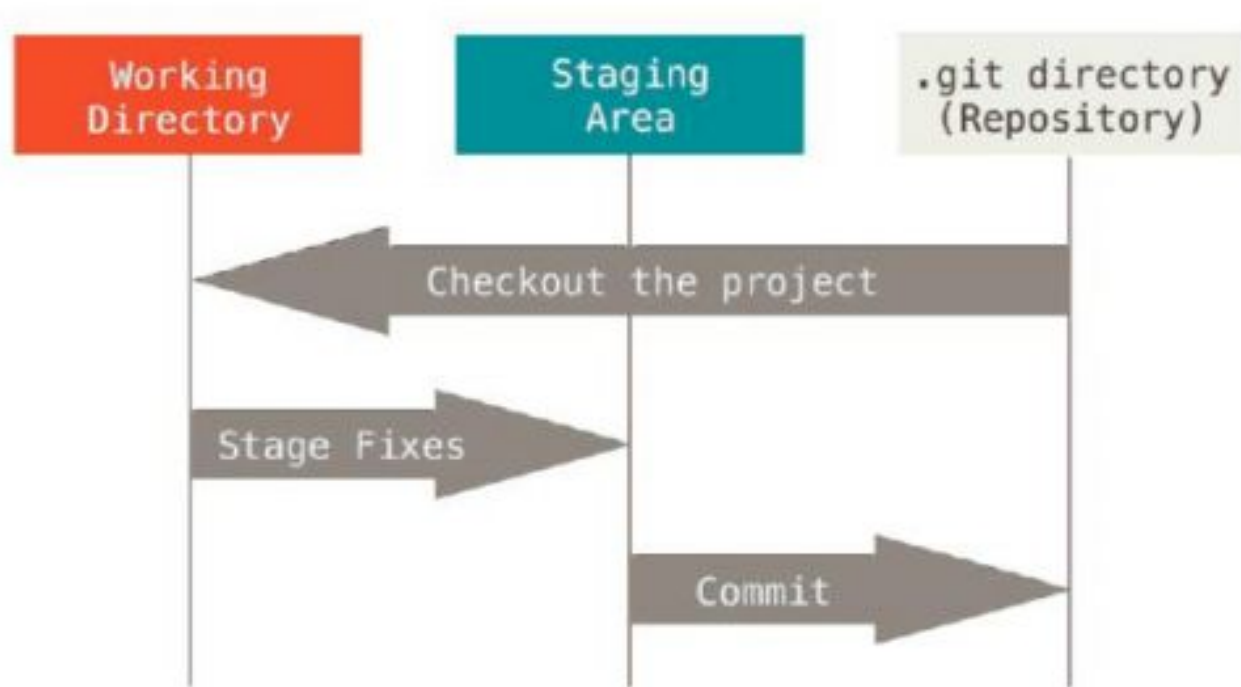
Merge commit: D + E added to Master via F

Instalando Git

- Windows - <https://git-for-windows.github.io/>
- Linux - Package management tool (Ubuntu)
apt-get install git
- Mac <http://sourceforge.net/projects/git-osx-installer/>



Estados do Git



Configurações básicas

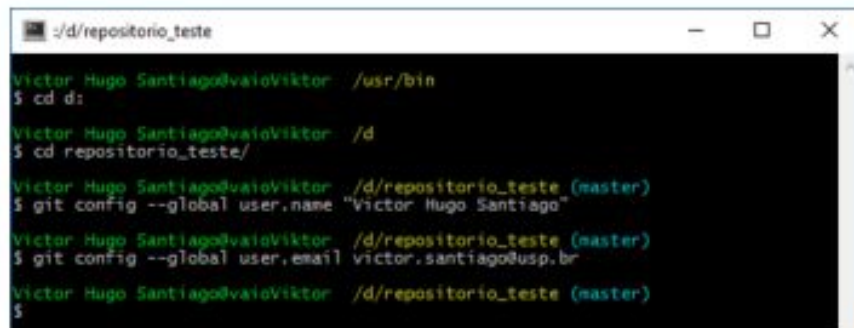
É importante nos identificarmos para o Git, execute os comandos a seguir no terminal com os seus dados:

```
git config -l
```

```
git config --global <variavel> <parâmetro>
```

```
git config --global user.name Nome
```

```
git config --global user.email meu@email.com
```

A terminal window titled "/d/repositorio_teste" with standard window controls. The terminal shows a series of commands and their outputs. The prompt is "Victor Hugo Santiago@vaioViktor". The commands and outputs are: 1. "\$ cd /" followed by "Victor Hugo Santiago@vaioViktor /usr/bin". 2. "\$ cd /d" followed by "Victor Hugo Santiago@vaioViktor /d". 3. "\$ cd repositorio_teste/" followed by "Victor Hugo Santiago@vaioViktor /d/repositorio_teste/". 4. "\$ git config --global user.name 'Victor Hugo Santiago'" followed by "Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)". 5. "\$ git config --global user.email victor.santiago@usp.br" followed by "Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)". 6. The prompt "\$" appears again, followed by "Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)".

```
Victor Hugo Santiago@vaioViktor /usr/bin
$ cd /
Victor Hugo Santiago@vaioViktor /d
$ cd repositorio_teste/
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git config --global user.name "Victor Hugo Santiago"
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git config --global user.email victor.santiago@usp.br
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$
```

Precisou de ajuda?

```
git help
```

```
git help <parâmetro>
```

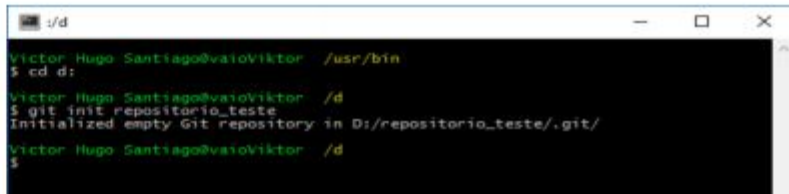
Iniciando seu repositório

Para transformar um diretório local em um repositório Git, basta abrir o diretório via terminal e executar o comando:

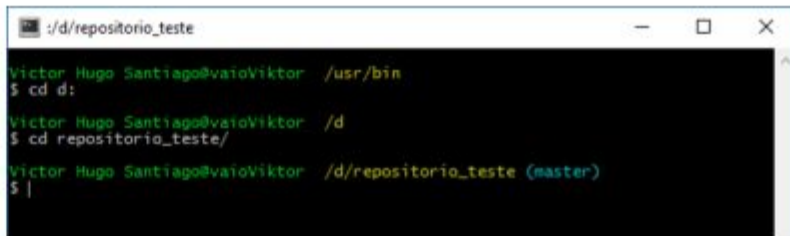
`git init`

ou

`git init repositorio_teste`



```
Victor Hugo Santiago@vaioViktor /usr/bin
$ cd d:
Victor Hugo Santiago@vaioViktor /d
$ git init repositorio_teste
Initialized empty Git repository in D:/repositorio_teste/.git/
Victor Hugo Santiago@vaioViktor /d
$
```

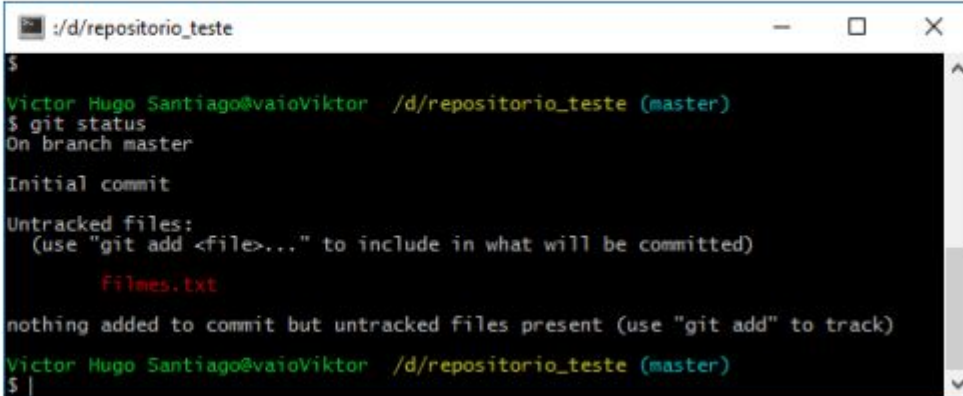


```
Victor Hugo Santiago@vaioViktor /usr/bin
$ cd d:
Victor Hugo Santiago@vaioViktor /d
$ cd repositorio_teste/
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ |
```

Rastreando com Git

Supondo que adicionamos o arquivo `filmes.txt` no diretório “repositorio_teste”

git status

A terminal window titled ':/d/repositorio_teste' with standard window controls. The terminal shows the output of the 'git status' command. The prompt is 'Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)'. The command '\$ git status' is entered. The output indicates an initial commit on the master branch with one untracked file, 'filmes.txt', shown in red. It suggests using 'git add' to track the file. The prompt returns to '\$ |' at the bottom.

```
:/d/repositorio_teste
$
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git status
On branch master

Initial commit

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

       filmes.txt

nothing added to commit but untracked files present (use "git add" to track)
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ |
```

Rastreando com Git

Para que um arquivo seja rastreado, devemos executar o seguinte comando:

`git add filmes.txt` ou
`git add .` (para incluir todos os arquivos)

```
:/d/repositorio_teste

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git add filmes.txt

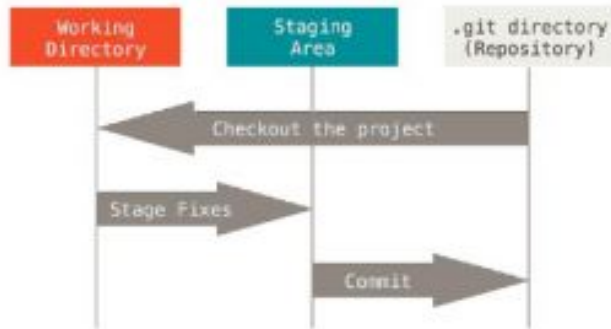
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   filmes.txt

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$
```



Gravando um arquivo no repositório (*commit*)

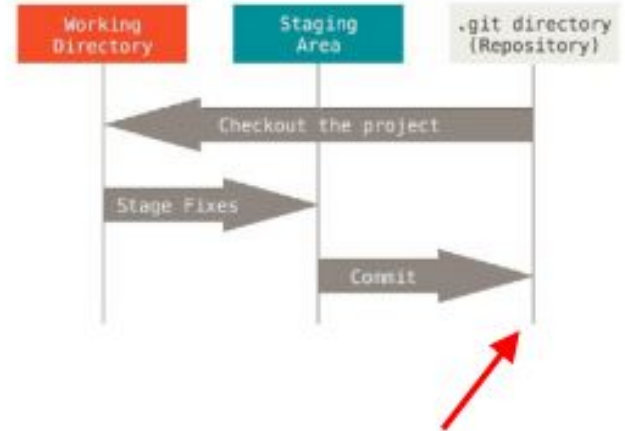
Para gravar as mudanças no repositório, execute o comando:

```
git commit -m "Arquivo inicial de filmes para download"
```

```
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git commit -m "Arquivo inicial de filmes para download"
[master (root-commit) 67aa6a7] Arquivo inicial de filmes para download
1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 filmes.txt

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git status
On branch master
nothing to commit, working directory clean

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$
```



Gravando um arquivo no repositório (*commit*)

Para visualizar o histórico de commits

`git log`

Para mostrar as alterações de um commit

`git show [commit]`

Commit deve ser especificado pela chave

```
~/d/repositorio_teste
Victor Hugo Santiago@vaioViktor: /d/repositorio_teste (master)
$ git log
commit 67aa6a743b3159f3dac92b5fb6fd98185e702013
Author: Victor Hugo Santiago <victor.santiago@usp.br>
Date: Sun Mar 6 18:55:45 2016 -0300

    Arquivo inicial de filmes para download

Victor Hugo Santiago@vaioViktor: /d/repositorio_teste (master)
$
```

```
commit 67aa6a743b3159f3dac92b5fb6fd98185e702013
Author: Victor Hugo Santiago <victor.santiago@usp.br>
Date: Sun Mar 6 18:55:45 2016 -0300

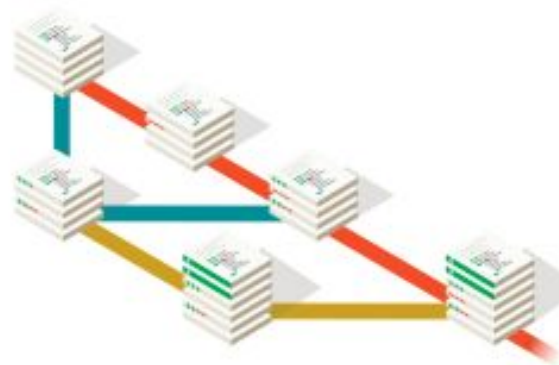
    Arquivo inicial de filmes para download

diff --git a/filmes.txt b/filmes.txt
new file mode 100644
index 0000000..e69de29
```

Curiosidade: o identificador do *commit* é um número de verificação de integridade de dados (*checksum*) criado a partir do conteúdo do arquivo usando a função de *hash* SHA-1

Branching (criando ramificações no repositório)

- *Branch* é uma ramificação do repositório
- Alterações (*commits*) ocorrem na *branch*
- Muito útil para trabalhos colaborativos
- *Branchs* de desenvolvimento facilitam o controle



Para listar as branches do nosso repositório, devemos executar o comando:

git branch

Por que ***master** ?

```
:/d/repositorio_teste

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git branch
* master

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ |
```

Branching (criando ramificações no repositório)

Como listar todas as *branches*?

git branch -v (- r para branch remota, ver depois)

Como criar uma *branch*?

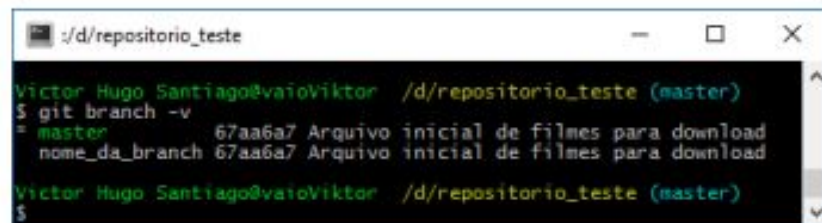
git branch nome_da_branch

Modo mais rápido de criar e deixá-la como corrente:

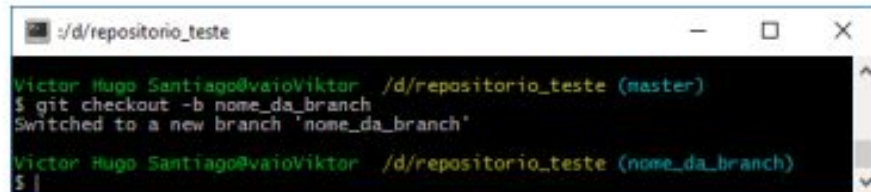
git checkout -b nome_da_branch

E se fosse para criar um *branch* remota? (ver depois)

git push origin nome_da_branch

A terminal window titled ':/d/repositorio_teste' showing the command 'git branch -v' and its output. The output lists the 'master' branch with its commit hash '67aa6a7' and a description 'Arquivo inicial de filmes para download'.

```
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git branch -v
* master        67aa6a7 Arquivo inicial de filmes para download
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$
```

A terminal window titled ':/d/repositorio_teste' showing the command 'git checkout -b nome_da_branch' and its output. The output indicates that a new branch 'nome_da_branch' has been created and the user has switched to it.

```
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git checkout -b nome_da_branch
Switched to a new branch 'nome_da_branch'
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (nome_da_branch)
$
```

Apagando uma branch

Apagar uma *branch* em seu repositório local

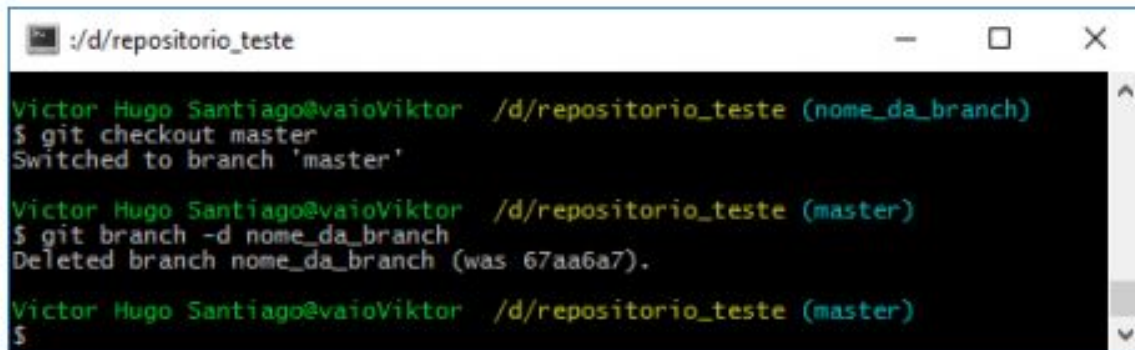
```
git branch -d nome_da_branch
```

Apagar uma *branch* remota (ver depois)

```
git push origin --delete nome_da_branch
```

Como mudo de *branch*?

```
git checkout nome_da_branch
```



```
:/d/repositorio_teste

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (nome_da_branch)
$ git checkout master
Switched to branch 'master'

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git branch -d nome_da_branch
Deleted branch nome_da_branch (was 67aa6a7).

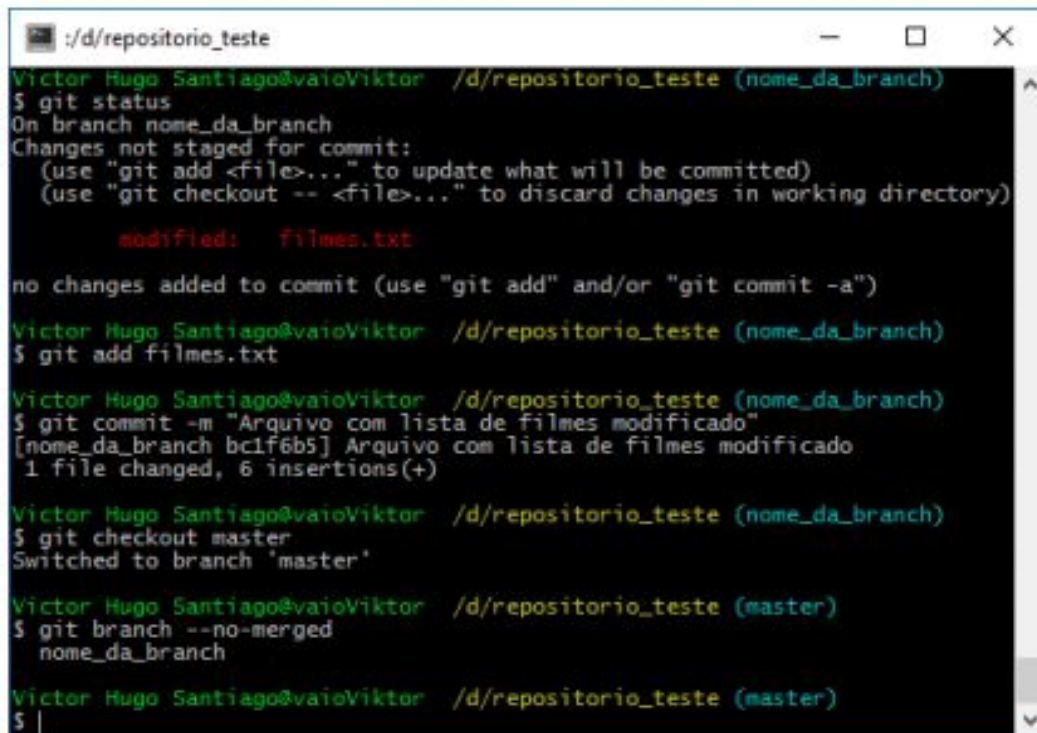
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$
```

Mesclando branches

Supondo que estamos na branch *master*, podemos verificar as *branches* ainda não mescladas da seguinte forma:

```
git branch --no-merged
```

Temos uma branch com modificações em relação a *branch master*!



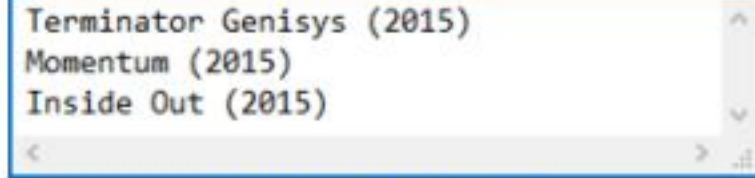
```
:/d/repositorio_teste
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (nome_da_branch)
$ git status
On branch nome_da_branch
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   filmes.txt

no changes added to commit (use "git add" and/or "git commit -a")
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (nome_da_branch)
$ git add filmes.txt
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (nome_da_branch)
$ git commit -m "Arquivo com lista de filmes modificado"
[nome_da_branch bclf6b5] Arquivo com lista de filmes modificado
1 file changed, 6 insertions(+)
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (nome_da_branch)
$ git checkout master
Switched to branch 'master'
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git branch --no-merged
        nome_da_branch
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$
```

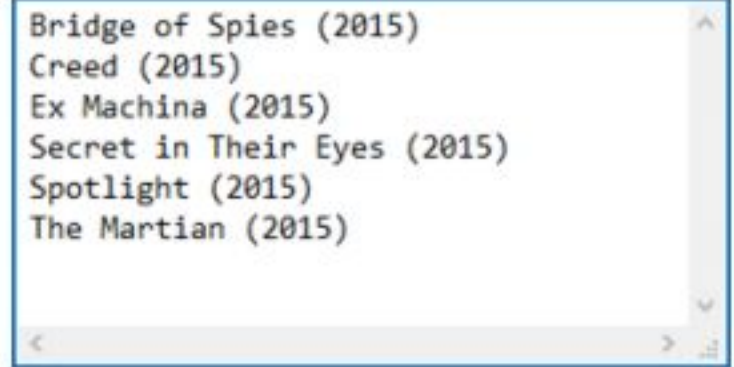
Mesclando branches

Para esclarecer:

A screenshot of a text editor window with a blue border. It contains three lines of text: "Terminator Genisys (2015)", "Momentum (2015)", and "Inside Out (2015)". The editor has a scrollbar on the right and navigation arrows at the bottom.

```
Terminator Genisys (2015)
Momentum (2015)
Inside Out (2015)
```

filmes.txt (master)

A screenshot of a text editor window with a blue border. It contains five lines of text: "Bridge of Spies (2015)", "Creed (2015)", "Ex Machina (2015)", "Secret in Their Eyes (2015)", and "Spotlight (2015)". The editor has a scrollbar on the right and navigation arrows at the bottom.

```
Bridge of Spies (2015)
Creed (2015)
Ex Machina (2015)
Secret in Their Eyes (2015)
Spotlight (2015)
```

filmes.txt (nome_da_branch)

Queremos mesclar a branch “nome_da_branch” com “master”!

Será que isso resultará em um conflito?

Mesclando branches

Para mesclar podemos usar um comando como esse:

```
git merge nome_da_branch -m "Mesclando com a branch master"
```

A terminal window titled ':/d/repositorio_teste' with standard window controls. The terminal shows a user switching to the 'master' branch and then attempting to merge 'nome_da_branch'. The merge fails due to a conflict in 'filmes.txt'.

```
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (nome_da_branch)
$ git checkout master
Switched to branch 'master'

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git merge nome_da_branch -m "Mesclando com a branch master"
Auto-merging filmes.txt
CONFLICT (content): Merge conflict in filmes.txt
Automatic merge failed; fix conflicts and then commit the result.

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master|MERGING)
$ |
```

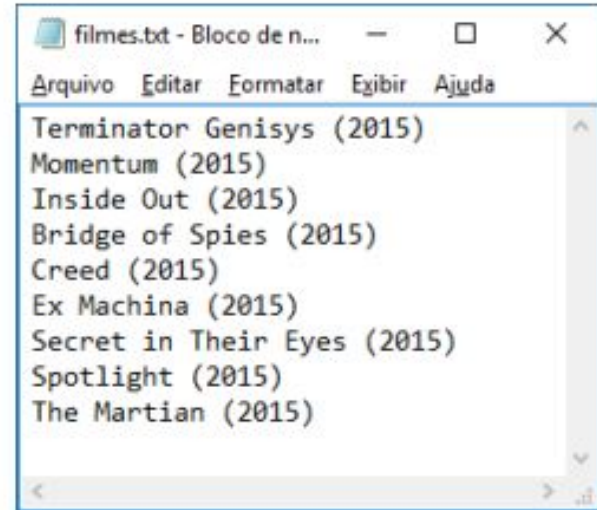
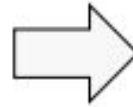
Mesclando branches

Resultado:



```
filmes.txt - Bloco de notas
Arquivo  Editar  Formatar  Exibir  Ajuda
|<<<<<<< HEAD
Terminator Genisys (2015)
Momentum (2015)
Inside Out (2015)
=====
Bridge of Spies (2015)
Creed (2015)
Ex Machina (2015)
Secret in Their Eyes (2015)
Spotlight (2015)
The Martian (2015)
>>>>>>> nome_da_branch
```

Após alterações:



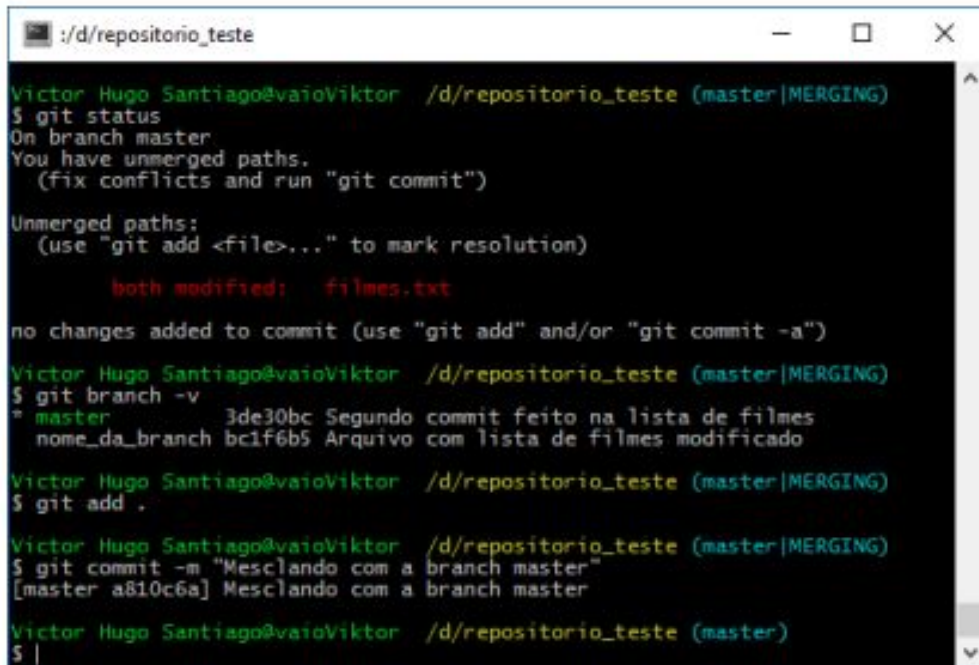
```
filmes.txt - Bloco de n...
Arquivo  Editar  Formatar  Exibir  Ajuda
Terminator Genisys (2015)
Momentum (2015)
Inside Out (2015)
Bridge of Spies (2015)
Creed (2015)
Ex Machina (2015)
Secret in Their Eyes (2015)
Spotlight (2015)
The Martian (2015)
```


Resolvendo conflitos

Faça as alterações no arquivo mesclado e depois use os comando:

```
git add .
```

```
git commit -m "mensagem"
```

A terminal window titled ':/d/repositorio_teste' with standard window controls. It shows a sequence of Git commands and their outputs. The user is on the 'master' branch in a 'MERGING' state. They run 'git status', which reports unmerged paths for 'filmes.txt'. They then run 'git add .' and 'git commit -m "Mesclando com a branch master"', which successfully resolves the conflict and creates a new commit. The terminal text is as follows:

```
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master|MERGING)
$ git status
On branch master
You have unmerged paths.
  (fix conflicts and run "git commit")

Unmerged paths:
  (use "git add <file>..." to mark resolution)

        both modified:   filmes.txt

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

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master|MERGING)
$ git branch -v
* master          3de30bc Segundo commit feito na lista de filmes
  nome_da_branch bc1f6b5 Arquivo com lista de filmes modificado

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master|MERGING)
$ git add .

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master|MERGING)
$ git commit -m "Mesclando com a branch master"
[master a810c6a] Mesclando com a branch master

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ |
```


Ignore arquivos com *.gitignore*

Basta criar um arquivo chamado **.gitignore** no diretório principal do nosso projeto com os nomes dos arquivos ou extensões que queremos ignorar.

Ex.:

```
*.log  
imagens/*.bmp
```

Em projetos Java, arquivos **.class**, **.jar** e **.war** são exemplos de arquivos que devem ser ignorados.

Detalhe importante: o **.gitignore** também envolverá com o projeto.

Assim, o que precisamos fazer?

Visualizar logs dos commits

git log

```
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git log
commit a810c6ac6961da052cfcc31a3c4a7a6f60298154
Merge: 3de30bc bc1f6b5
Author: Victor Hugo Santiago <victor.santiago@usp.br>
Date: Sun Mar 6 19:44:57 2016 -0300

    Mesclando com a branch master

commit 3de30bc338e4403fad91e933f73ad58041b7c471
Author: Victor Hugo Santiago <victor.santiago@usp.br>
Date: Sun Mar 6 19:32:34 2016 -0300

    Segundo commit feito na lista de filmes

commit bc1f6b5e1b80633cd6ac3b4b50393538b674897b
Author: Victor Hugo Santiago <victor.santiago@usp.br>
Date: Sun Mar 6 19:27:37 2016 -0300

    Arquivo com lista de filmes modificado

commit 67aa6a743b3159f3dac92b5fb6fd98185e702013
Author: Victor Hugo Santiago <victor.santiago@usp.br>
Date: Sun Mar 6 18:55:45 2016 -0300

    Arquivo inicial de filmes para download

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ |
```

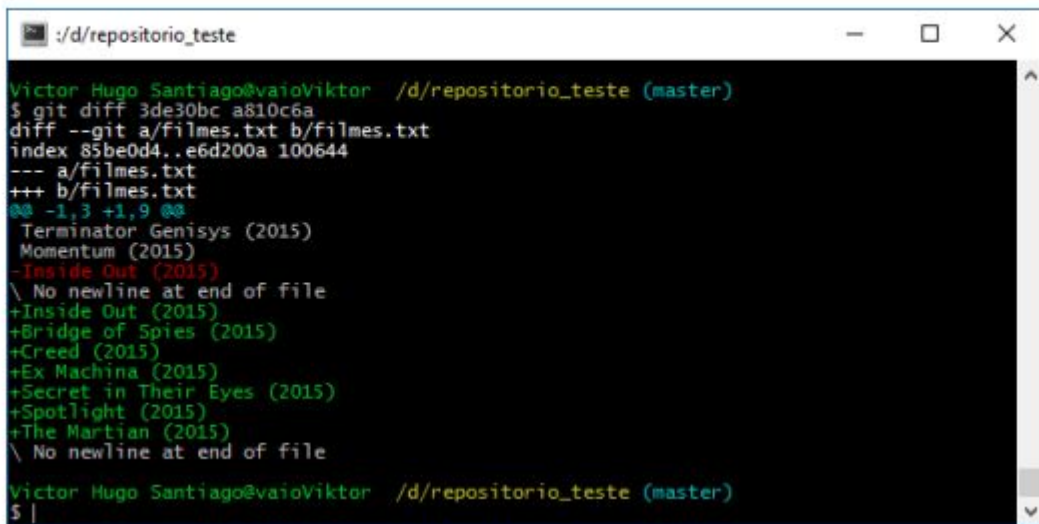
git log --graph --oneline

```
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git log --graph --oneline
* a810c6a Mesclando com a branch master
|
| * bc1f6b5 Arquivo com lista de filmes modificado
| * | 3de30bc Segundo commit feito na lista de filmes
|/
* 67aa6a7 Arquivo inicial de filmes para download

Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ |
```

Verificar diferenças entre commits

```
git diff 3de30bc a810c6a
```

A terminal window titled ':/d/repositorio_teste' with standard window controls. The terminal shows the execution of 'git diff 3de30bc a810c6a'. The output indicates a difference in 'a/filmes.txt' and 'b/filmes.txt' with a line count change from -1,3 to +1,9. The diff shows several lines of movie titles being added, with '-Inside Out (2015)' in red indicating a deletion and '+Inside Out (2015)' in green indicating an addition. The window has a scrollbar on the right side.

```
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ git diff 3de30bc a810c6a
diff --git a/filmes.txt b/filmes.txt
index 85be0d4..e6d200a 100644
--- a/filmes.txt
+++ b/filmes.txt
@@ -1,3 +1,9 @@
Terminator Genisys (2015)
Momentum (2015)
-Inside Out (2015)
\ No newline at end of file
+Inside Out (2015)
+Bridge of Spies (2015)
+Creed (2015)
+Ex Machina (2015)
+Secret in Their Eyes (2015)
+Spotlight (2015)
+The Martian (2015)
\ No newline at end of file
Victor Hugo Santiago@vaioViktor /d/repositorio_teste (master)
$ |
```

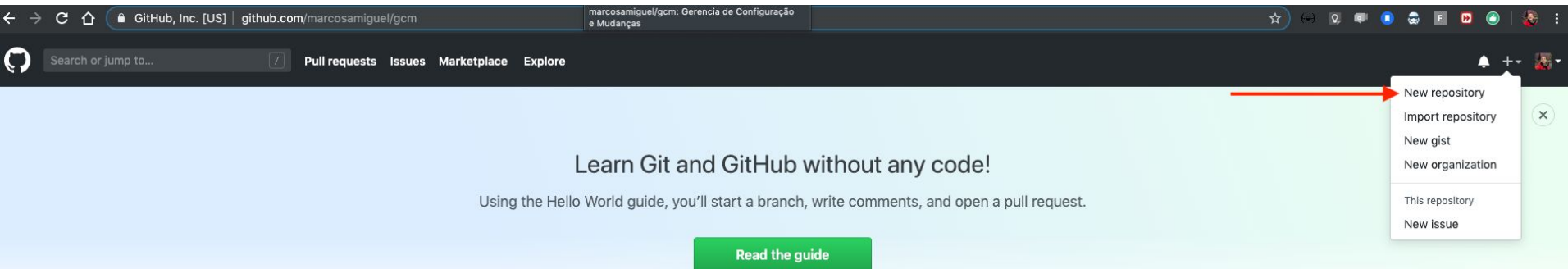


GitHub



Criando repositório

Crie sua conta no github: <https://github.com/>



The screenshot shows the GitHub homepage. The browser's address bar displays 'github.com/marcosamiguel/gcm'. The GitHub navigation bar includes links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. A red arrow points to the '+' icon in the top right corner, which has opened a dropdown menu. The menu contains the following options: 'New repository' (highlighted), 'Import repository', 'New gist', 'New organization', 'This repository', and 'New issue'.

Learn Git and GitHub without any code!

Using the Hello World guide, you'll start a branch, write comments, and open a pull request.

[Read the guide](#)

- New repository
- Import repository
- New gist
- New organization
- This repository
- New issue

Criando repositório

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner



marcosamiguel ▾

Repository name *

teste1



Great repository names are short and memorable. Need inspiration? How about **curly-octo-guacamole**?

Description (optional)



Public

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



Private

You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.



Initialize this repository with a README

This will let you immediately clone the repository to your computer.

Add .gitignore: **None** ▾

Add a license: **None** ▾



Create repository



Gravando dados em projeto Git

1. Abra o Terminal .
2. Altere o diretório de trabalho atual para o seu projeto local.
3. Inicialize o diretório local como um repositório Git.

```
$ git init
```

4. Adicione os arquivos em seu novo repositório local. Isso os prepara para o primeiro commit.

```
$ git add .
```

```
# Adds the files in the local repository and stages them for commit. To unstage a file, use 'git reset HEAD YOUR-FILE'.
```

Gravando dados em projeto Git

5. Commit os arquivos que você colocou no seu repositório local.

```
$ git commit -m "Primeiro commit"
```

```
# Commits the tracked changes and prepares them to be  
# pushed to a remote repository. To remove this commit  
# and modify the file, use 'git reset --soft HEAD~1'  
# and commit and add the file again.
```

6. Na parte superior da página Configuração rápida do repositório do GitHub, clique para copiar o URL do repositório remoto.

The screenshot shows the GitHub repository page for 'marcosamiguel / gcm'. At the top, there are buttons for 'Unwatch', 'Star', and 'Fork'. Below these are tabs for 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. The main content area shows 'Gerencia de Configuração e Mudanças' and 'Manage topics'. A red horizontal line highlights the 'Clone or download' button. A red arrow points from the text in step 6 to this button. Another red arrow points from the 'Clone with HTTPS' dialog to the 'Clone or download' button. The dialog shows the URL 'https://github.com/marcosamiguel/gcm' and buttons for 'Open in Desktop' and 'Download ZIP'. Below the dialog, there is a table of files and commits.

File	Commit
README.md	Initial commit
curriculum.html	versao 1.0.0020 - Github
instrucoes.txt	parte 2



Gravando dados em projeto Git

8. No Terminal, adicione o URL para o repositório remoto onde seu repositório local será enviado.

```
$ git remote add origin remote repository URL
```

Ex: `git remote add origin https://github.com/marcosamiguel/gcm.git`

9. Suba as alterações no seu repositório local para o GitHub.

```
$ git push -u origin master
```

```
# Pushes the changes in your local repository up to the remote repository you specified as the origin
```

Clonado dados em projeto Git

1. Abra o Terminal .
2. Crie um diretório de trabalho para seus projetos. Ex: c:\dev
3. Selecione o projeto no GitHub

The screenshot shows the GitHub interface for the repository 'marcosamiguel / gcm'. A red arrow points from the third step of the list ('Selecione o projeto no GitHub') to the repository name. Another red arrow points from the '10 commits' link to the 'Clone with HTTPS' dialog box. The dialog box provides the URL 'https://github.com/marcosamiguel/gcm.' and options to 'Open in Desktop' or 'Download ZIP'.

marcosamiguel / gcm

Unwatch 1 Star 1 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

Gerencia de Configuração e Mudanças

Manage topics

10 commits 1 branch 0 releases 1 contributor

Branch: master New pull request

Create new file Upload files Find file Clone or download

miguel parte 2

README.md	Initial commit
curriculum.html	versao 1.0.0020 - Github
instrucoes.txt	parte 2

README.md

Clone with HTTPS

Use Git or checkout with SVN using the web URL.

[https://github.com/marcosamiguel/gcm.](https://github.com/marcosamiguel/gcm)

Open in Desktop Download ZIP



Clonado dados em projeto Git

4. Clone o Projeto

```
git clone URL do Projeto
```

```
Ex: git clone https://github.com/marcosamiguel/gcm.git
```

Criando uma nova branch no GitHub

1. No github cria a branch

The screenshot shows the GitHub interface for the repository 'marcosamiguel / gcm'. At the top, there are navigation tabs: Code, Issues (0), Pull requests (0), Projects (0), Wiki, Security, Insights, and Settings. Below these, the repository name and statistics are displayed: 10 commits, 1 branch, 0 releases, and 1 contributor. A red horizontal line separates the repository header from the content area. The content area shows the 'Switch branches/tags' dropdown menu, which is open, displaying the current branch 'master' and a list of branches. The 'Create branch: versao1 from 'master'' option is highlighted in blue. Two red arrows point to this option. The commit history table is visible in the background, showing the latest commit 'Initial commit' 27 minutes ago, and previous commits 'versao 1.0.0020 - Github' and 'parte 2'.

marcosamiguel / gcm

Unwatch 1 Star 1 Fork 0

<> Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

Gerencia de Configuração e Mudanças Edit

Manage topics

10 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find File Clone or download

Switch branches/tags

versao1

Branches Tags

Create branch: versao1 from 'master'

Latest commit 2e13615 27 minutes ago

Initial commit	6 months ago
versao 1.0.0020 - Github	6 months ago
parte 2	27 minutes ago

README.md

gcm

Gerencia de Configuração e Mudanças



Criando uma nova branch no github

2. Clone o projeto

```
git clone -b {branch} {url}
```

```
git clone -b versao1 https://github.com/marcosamiguel/gcm.git
```

3. Modifique ou adicione arquivos commit e suba o projeto

```
git add .
```

```
git commit -m "Primeiro commit da branch"
```

```
git remote add origin https://github.com/marcosamiguel/gcm.git
```

```
git push -u origin versao1
```



Mergeando com master

1. Dentro da Branch (ex: versao1) criada, faça todo o processo de commit ao push
2. Faça o merge :

```
git checkout master → troca da branch atual para a master e vice versa
```

```
git merge versao1 → faz o merge da versao1 com a master
```

```
git commit -m "merge da alteracao" → Caso tenha conflitos
```

```
git push -u origin master → sobe as alterações
```



Tutorial GIT

Doc umentação Completa: <https://git-scm.com/doc>

Book: <https://git-scm.com/book/en/v2>

Merge:

<https://git-scm.com/book/pt-br/v1/Ramifica%C3%A7%C3%A3o-Branching-no-Git-B%C3%A1sico-de-Branch-e-Merge>



Utilitários Git

- `git grep -n <expressão>` ⇒ Busca uma expressão em todo o repositório do projeto.
- `git log > log.txt` ⇒ Cria um log das operações do git
- `git reflog` ⇒ Registro das referências ao HEAD e branches dos últimos meses
- `git show HEAD@{5}` ⇒ Mostra a última mudança de um Commit {5} por exemplo.
- `git show master@{yesterday}` ⇒ Como a master estava ontem por exemplo.
- `git log --pretty=format:'%h %s' --graph` ⇒ Referência gráfica do commit através de sua ancestralidade

github
SOCIAL CODING

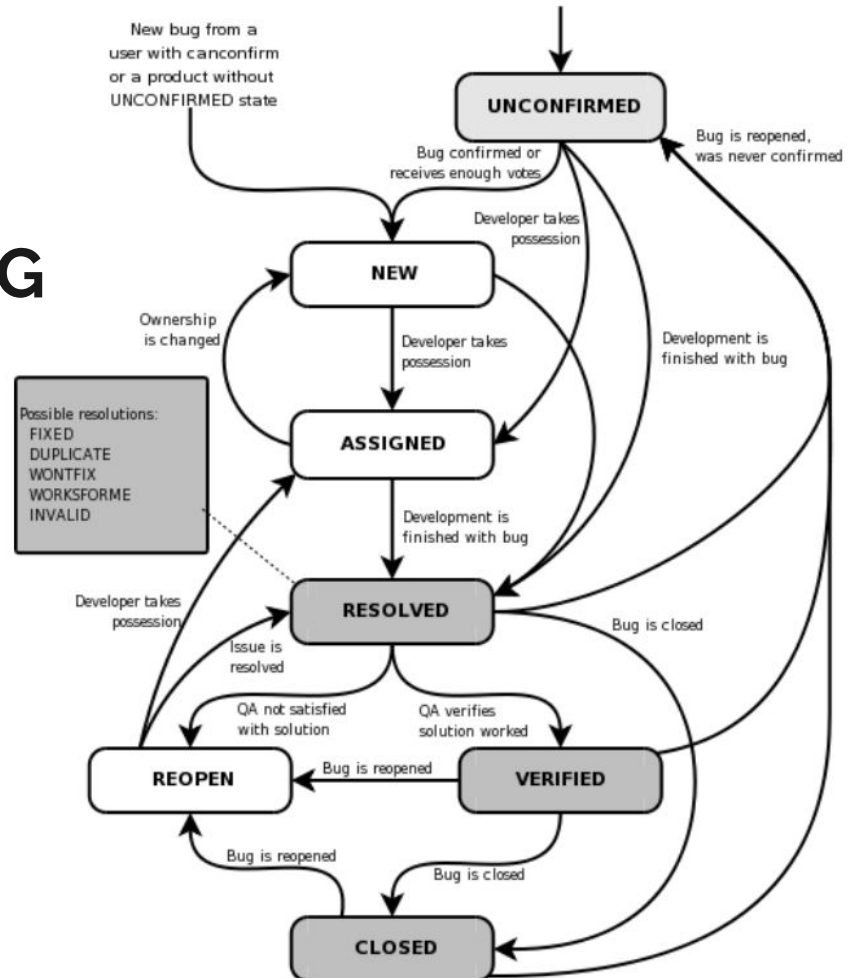


<https://github.com/>



<https://www.mantisbt.org/>

Ciclo de Vida do BUG





Ciclo de Vida do BUG

- Unconfirmed (Não Confirmado): Registro do Defeito criado mas não atribuído a ninguém.
- New (Novo): Quando um defeito é identificado e cadastrado na primeira vez. Seu estado é dado como Novo.
- Assigned (Atribuído): Após o testador ter cadastrado o defeito, seus Gerente verifica se esse defeito é realmente novo e então atribui o defeito para que o desenvolvedor e seu time possa corrigi-lo. Seu estado é dado como Atribuído.
- Resolved (Resolvido): Defeito foi resolvido pelo desenvolvedor.
- Reopen (Reaberto): Se o defeito ainda existir mesmo após o mesmo ter sido corrigido pelo desenvolvedor, o testador muda o estado para 'Reaberto', Então, o defeito inicia todo o ciclo de vida novamente.
- Closed (Fechado): Uma vez que o defeito foi corrigido, testado e verificado que o Defeito não mais existe, o testador muda o estado para 'Fechado'. Este estado informa que o defeito está corrigido, testado e aprovado.
- Verified (Verificado ou Validado): O testado testa novamente a correção após ele ser corrigido pelo desenvolvedor, se o defeito não for mais localizado no Software, então, o testador informa que o defeito foi consertado e muda o estado para "Validado" ou "Verificado".

Ciclo de Vida do BUG

O ciclo de vida do defeito ou bug ocorre sempre que um defeito é identificado durante o ciclo de vida da aplicação e termina quando o defeito tem seu estado alterado para 'fechado' após ser confirmada a sua correção.

- Os bugs podem ser reportados:
 - Pelo cliente ao utilizar o sistema
 - Pela equipe de QA (Qualidade e testes)
 - Pelos próprios desenvolvedores ao testarem um rotina



INTEGRAÇÃO CONTÍNUA



Jenkins



INTEGRAÇÃO CONTÍNUA

- Integração Contínua (CI - Continuous Integration) é uma prática dentro da área de engenharia de software que tem por objetivo realizar de forma automatizada compilações parciais ou totais do(s) software(s) que está(ão) sendo desenvolvido(s), testes unitários, testes integrados e outras atividades que a ferramenta em questão possa executar de forma automática para garantir a qualidade do produto.
- Vale a pena ressaltar que essa prática surgiu dentre os estudos de metodologias ágeis e é uma prática que vem se difundindo cada vez mais dentro do mercado haja visto que essa ferramenta consegue destacar problemas antes de que seja realizada uma release do produto.

Tutorial de Instalação do Jenkins

Entre no site do Jenkins em jenkins.io: <https://jenkins.io/>



Jenkins

Build great things at any scale

The leading open source automation server, Jenkins provides hundreds of plugins to support building, deploying and automating any project.

Documentation

Download

Download Jenkins 2.117 for:

Arch Linux

Docker

FreeBSD

Gentoo

Mac OS X

OpenBSD

openSUSE

Red Hat/Fedora/CentOS

Ubuntu/Debian

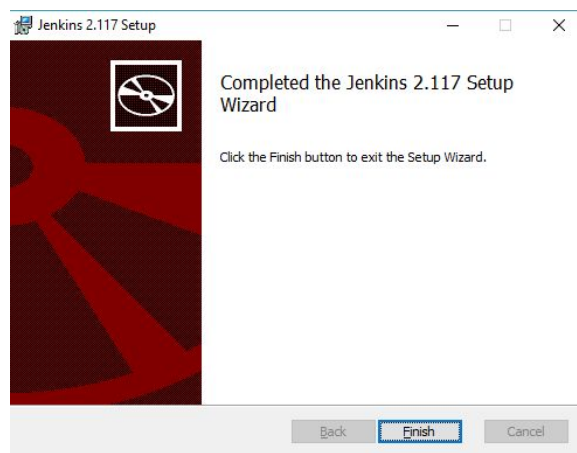
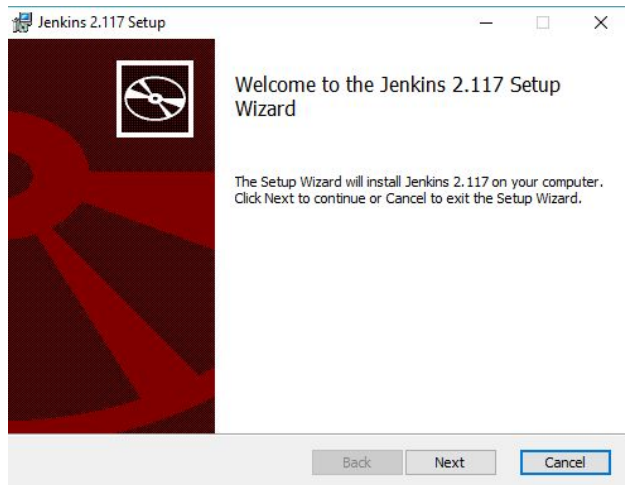
OpenIndiana Hipster

Windows

Generic Java package (.war)

Tutorial de Instalação do Jenkins

Descompacte o arquivo jenkins-(versão).zip e execute o instalador jenkins.msi e ao abrir a tela de bem-vindo, clique em Next, e siga os procedimentos até o final.





Tutorial de Instalação do Jenkins

Configuração inicial do Jenkins

Após o término da instalação, o endereço `http://localhost:8080/` vai abrir no browser e então uma tela de desbloqueio do Jenkins será exibida, com o caminho da senha inicial.

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

```
C:\Program Files (x86)\Jenkins\secrets\initialAdminPassword
```

Please copy the password from either location and paste it below.

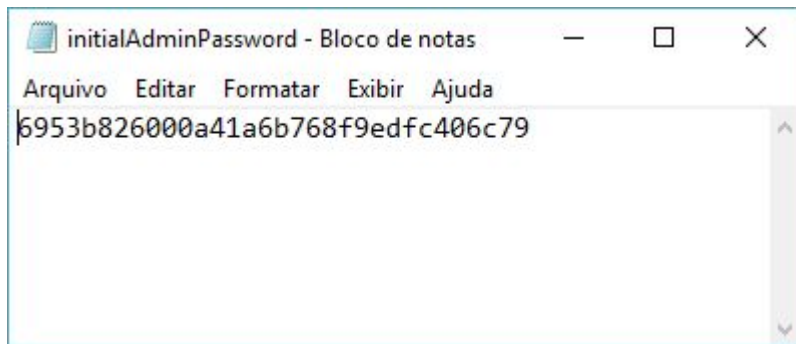
Administrator password

.....

Continue

Tutorial de Instalação do Jenkins

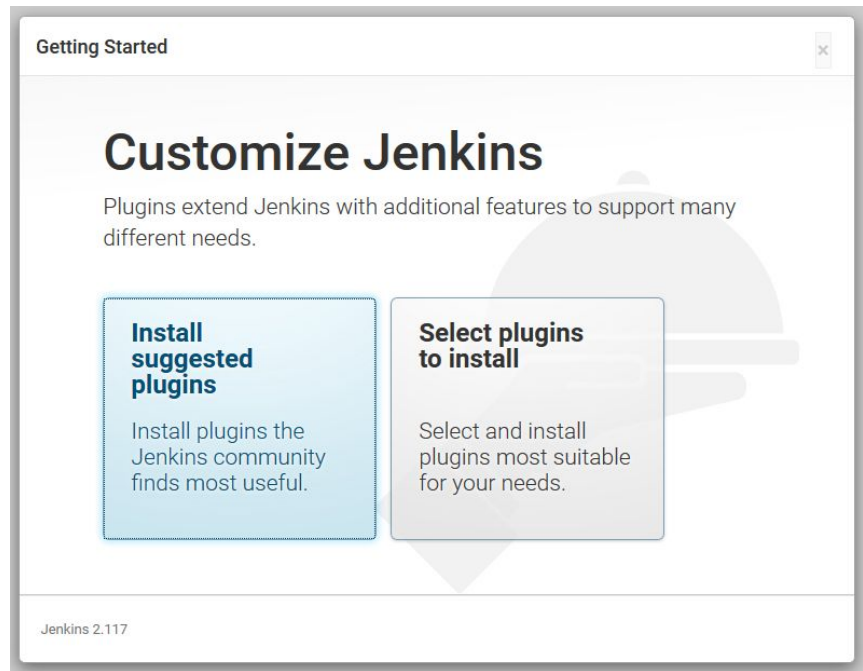
Abra o arquivo initialAdminPassword no Bloco de Notas e copie a senha.





Tutorial de Instalação do Jenkins

Digite a senha no campo Administrator password e clique em Continue. Em Customize Jenkins clique em Install suggested plugins, para que o Jenkins instale o plugins mais usados.





Tutorial de Instalação do Jenkins

Após a instalação dos plugins, será preciso criar o primeiro usuário administrador, para isso preencha o formulário em Create First Admin User e por fim clique em Save and Finish.

Getting Started

Create First Admin User

Nome de usuário:

Senha:

Confirmar a senha:

Nome completo:

Endereço de e-mail:

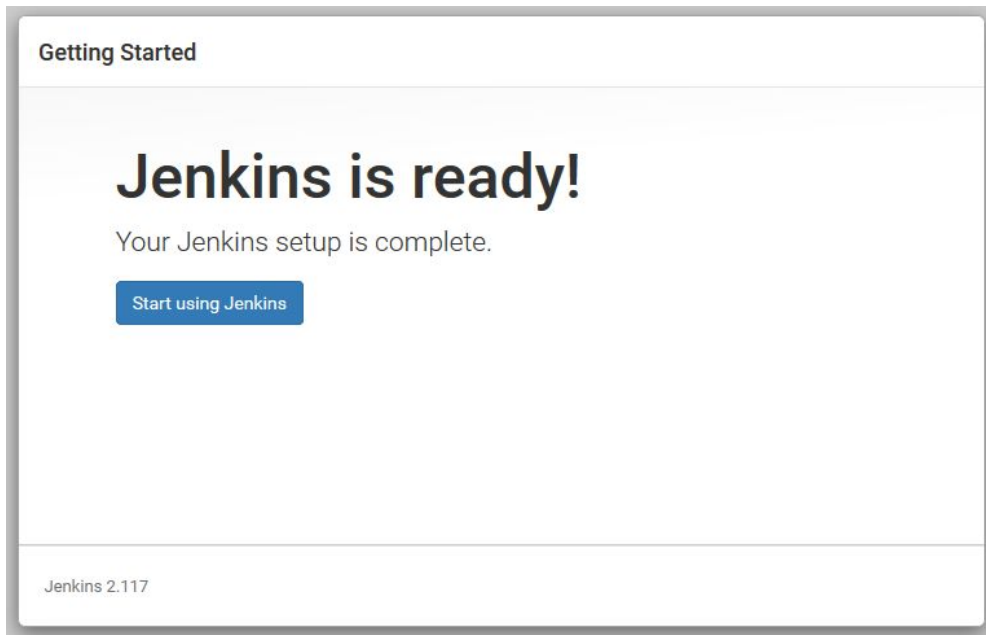
Jenkins 2.117

[Continue as admin](#)[Save and Finish](#)



Tutorial de Instalação do Jenkins

Agora o Jenkins está pronto para uso! O próximo passo é criar os Jobs e instalar os plugins adicionais.





Tutorial de Instalação do Jenkins

Documentação oficial:

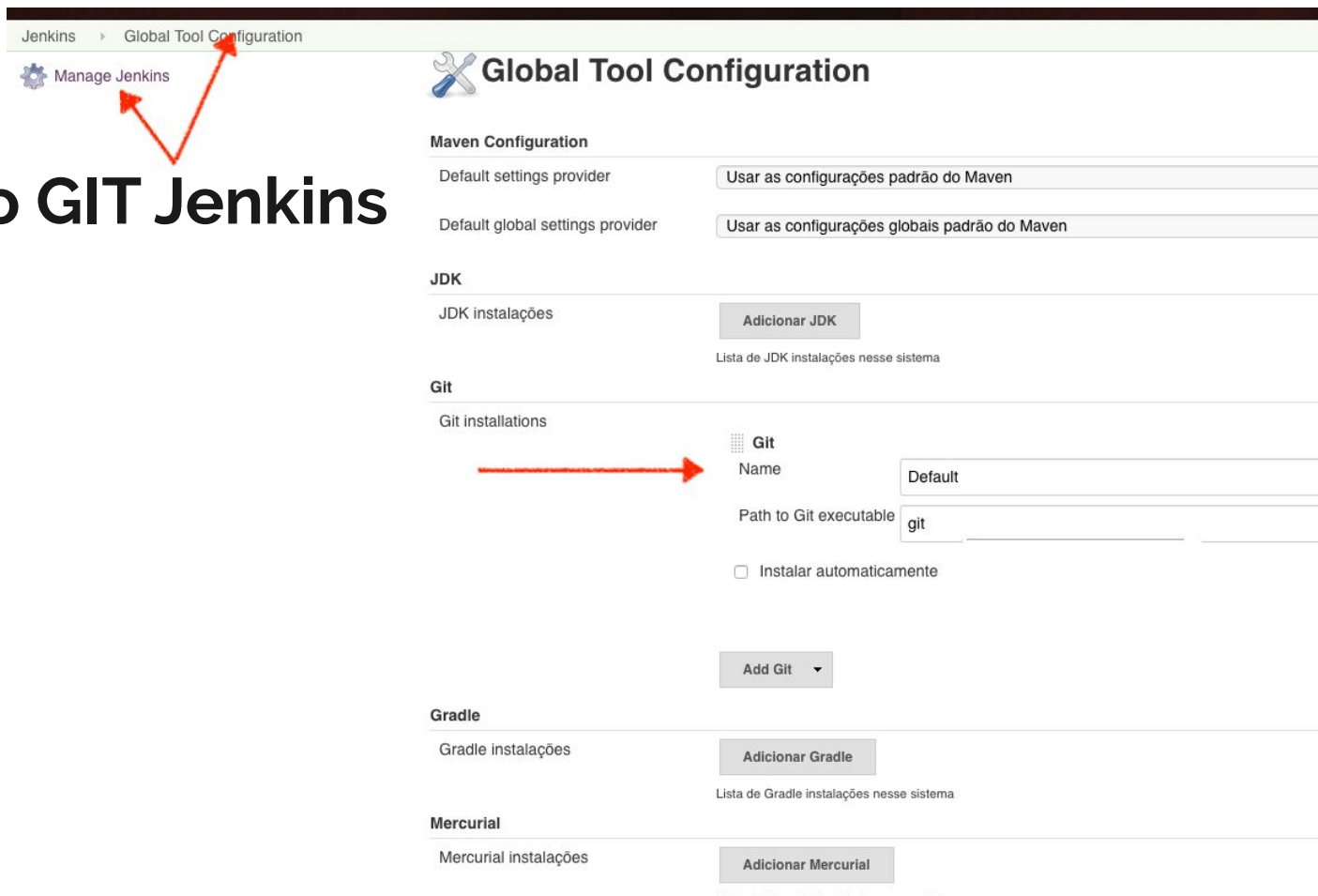
<https://jenkins.io/doc/book/installing/>

<https://jenkins.io/doc/tutorials/build-a-java-app-with-maven/>

Adicionais:

https://pt.wikiversity.org/wiki/Github_-_Jenkins

Configurando GIT Jenkins



The screenshot shows the Jenkins web interface. At the top, the breadcrumb navigation reads 'Jenkins > Global Tool Configuration'. Below this, there is a 'Manage Jenkins' link with a gear icon. A red arrow points from the 'Manage Jenkins' link to the 'Global Tool Configuration' page. The page title is 'Global Tool Configuration' with a wrench and screwdriver icon. The page is divided into sections for different build tools: Maven Configuration, JDK, Git, Gradle, and Mercurial. The 'Git' section is currently active, showing 'Git installations'. A red arrow points from the 'Git' section header to the 'Git' installation table. The table has columns for 'Name' and 'Path to Git executable'. The first entry has 'Name' set to 'Default' and 'Path to Git executable' set to 'git'. There is a checkbox for 'Instalar automaticamente' which is unchecked. Below the table is an 'Add Git' button with a dropdown arrow. The other sections (Maven, JDK, Gradle, Mercurial) each have a list of installations and an 'Adicionar' (Add) button.

Jenkins > Global Tool Configuration

Manage Jenkins

Global Tool Configuration

Maven Configuration

Default settings provider: Usar as configurações padrão do Maven

Default global settings provider: Usar as configurações globais padrão do Maven

JDK

JDK instalações: Adicionar JDK

Lista de JDK instalações nesse sistema

Git

Git installations

Git
Name: Default
Path to Git executable: git
<input type="checkbox"/> Instalar automaticamente

Add Git

Gradle

Gradle instalações: Adicionar Gradle

Lista de Gradle instalações nesse sistema

Mercurial

Mercurial instalações: Adicionar Mercurial

Configurando Maven Jenkins

Jenkins > Global Tool Configuration

Ant

Ant instalações

Adicionar Ant

Lista de Ant instalações nesse sistema

Maven

Maven instalações

Adicionar Maven

Maven

Nome

☒ Instalar automaticamente

Instalar a partir do Apache

Versão

Adicionar instalador

Adicionar Maven

Lista de Maven instalações nesse sistema

NodeJS

NodeJS instalações...

Docker

Docker instalações

Adicionar Docker




Lista de Docker instalações nesse sistema

Criando um novo Job

Enter an item name

xx| exemplo

Required field

-  **Construir um projeto de software free-style**
Esta é a central de funcionalidades do Jenkins. Ele construirá seu projeto, e você pode combinar qualquer SCM com qualquer sistema de software.
-  **Construir um projeto maven**
Construir um projeto maven. Jenkins tira vantagem de seus arquivos POM e reduz drasticamente a configuração. Ainda é um trabalho em andamento.
-  **Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflow).

 **Jenkins**

Jenkins ▶

 [Novo job](#)

 [Usuários](#)

 [Histórico de compilações](#)

 [Relacionamento entre projetos](#)

 [Verificar arquivo digital](#)

 [Gerenciar Jenkins](#)

 [Minhas views](#)

 [Open Blue Ocean](#)

 [Lockable Resources](#)

 [Credentials](#)

 [New View](#)

Fila de builds

Nenhum build na fila.

Criando um novo Job

General

Build Triggers

Advanced Project Options

Pipeline

☐ Desabilitar builds

☐ Período de silêncio

☐ Dispare builds remotamente (exemplo, a partir dos scripts)

Advanced Project Options

Pipeline

Definition

Pipeline script

Script

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

```
node {
  stage("checkout") {
    git url: "https://github.com/marcosamiguel/exemplo.git"
    def MAVEN_PLUGIN = 'maven-jenkins'
    def mvnHome = tool MAVEN_PLUGIN
    mvn = "${mvnHome}/bin/mvn"
  }
  stage("build") {
    def BUILD = 'clean package -U'
    sh "${mvn} ${BUILD}"
  }
  stage("test") {
    def JUNIT = '**/target/surefire-reports/TEST-*.xml'
    junit JUNIT
  }
}
```

☒ Use Groovy Sandbox

[Pipeline Syntax](#)

Salvar

Aplicar



Criando um novo Job

```
node {  
  stage("checkout") {  
    git url: "https://github.com/marcosamiguel/exemplo.git"  
    def MAVEN_PLUGIN = 'maven-jenkins'  
    def mvnHome = tool MAVEN_PLUGIN  
    mvn = "${mvnHome}/bin/mvn"  
  }  
  stage("build") {  
    def BUILD = 'clean package -U'  
    sh "${mvn} ${BUILD}"  
  }  
  
  stage("test") {  
    def JUNIT = '**/target/surefire-reports/TEST-*.xml'  
    junit JUNIT  
  }  
  
  stage("deploy") {  
    def JAR = 'target/*.jar'  
    archive JAR  
  }  
}
```

Job - Pipeline Sintaxe

General

Build Triggers

Advanced Project Options

Pipeline

☐ Desabilitar builds

☐ Período de silêncio

☐ Dispare builds remotamente (exemplo, a partir dos scripts)

Advanced Project Options

Pipeline

Definition

Pipeline script

Script

```
1 node {
2   stage("checkout") {
3     git url: "https://github.com/marcosamiguel/exemplo.git"
4     def MAVEN_PLUGIN = 'maven-jenkins'
5     def mvnHome = tool MAVEN_PLUGIN
6     mvn = "${mvnHome}/bin/mvn"
7   }
8   stage("build") {
9     def BUILD = 'clean package -U'
10    sh "${mvn} ${BUILD}"
11  }
12  stage("deploy") {
13    def JAR = 'target/*.jar'
14    archive JAR
15  }
16 }
```

☒ Use Groovy Sandbox

[Pipeline Syntax](#)

Salvar

Aplicar

[Back](#)[Snippet Generator](#)[Declarative Directive Generator](#)[Declarative Online Documentation](#)[Steps Reference](#)[Global Variables Reference](#)[Online Documentation](#)[IntelliJ IDEA GDSL](#)

Overview

This **Snippet Generator** will help you learn the Pipeline Script code which can be used to define various steps. Pick a step you are interested in from the list, configure it, click **Generate Pipeline Script**, and you will see a Pipeline Script that would call the step with that configuration. You may copy and paste the whole statement into your script, or pick up just the options you care about. (Most parameters are optional and can be omitted in your script, leaving only the required values.)

Steps

Sample Step **bat: Windows Batch Script**Batch Script **nota.bat****Generate Pipeline Script**

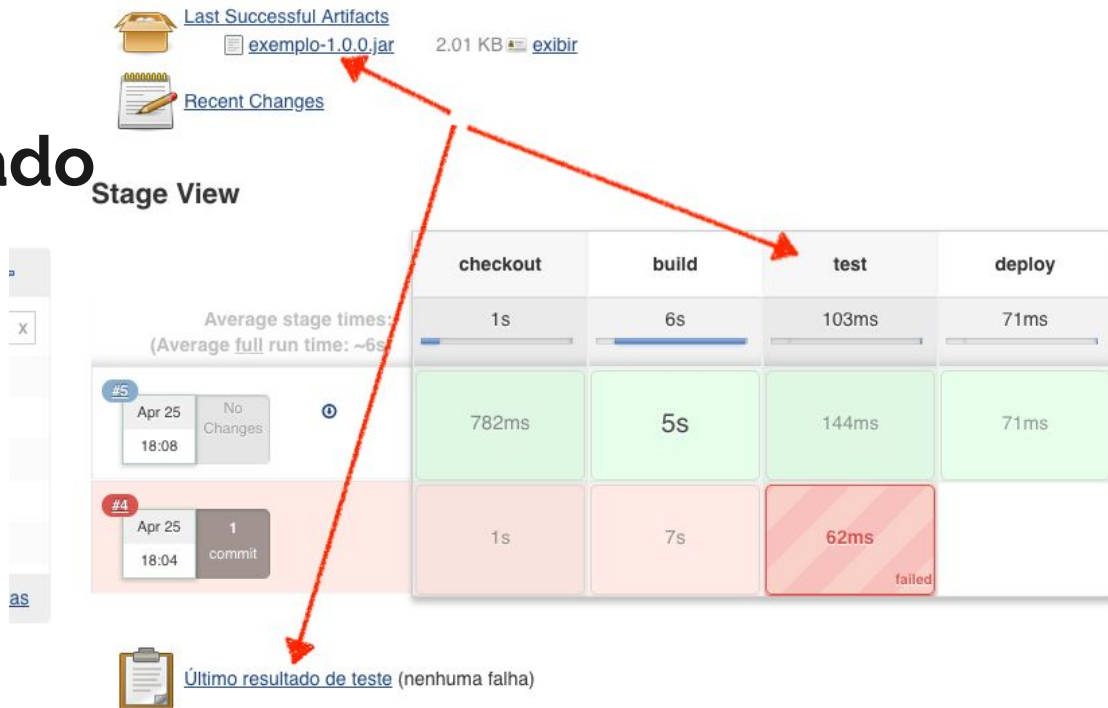
bat 'nota.bat'

Global Variables

There are many features of the Pipeline that are not steps. These are often exposed via global variables, which are not supported by the snippet generator. See the [Global Variables Reference](#) for details.

Job - Pipeline Sintaxe

Jenkins Resultado



Links permanentes

- [Último build \(#5\), 17 segundos atrás](#)
- [Último build estável \(#5\), 17 segundos atrás](#)
- [Último build bem sucedido \(#5\), 17 segundos atrás](#)
- [Último build que falhou \(#4\), 3 minutos 59 segundos atrás](#)
- [Último build que falhou \(#4\), 3 minutos 59 segundos atrás](#)



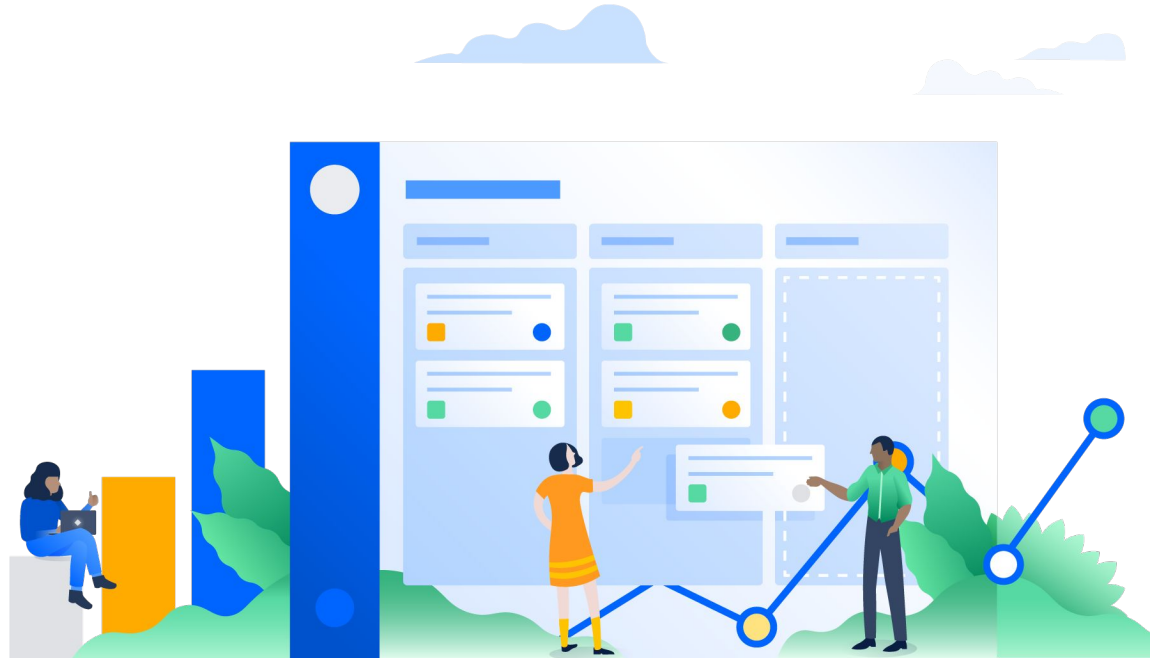
Jira - BugTracker e Scrum



Jira - BugTracker e Scrum


<https://br.atlassian.com/software/jira>

<https://marcosmiguel.atlassian.net/>



Criando um Projeto

← → ↻ 🏠 <https://marcosmiguel.atlassian.net/secure/BrowseProjects.jspa> ☆ Anônima




Jira Software


- ☆
- 🔍
- +
- 📁 Projetos
- 📊 Painéis de controle
- 📋 Itens e filtros
- ⚙️ Configurações do...

Projetos


Projetos sugeridos



Projeto 1 - GCM
Projeto de Software




Time 3 - Scrum
Projeto de Software



Projeto 2 - GCM
Projeto de Software







TESTAR GRATUITAMENTE



Configurar uma central de atendimento para

📋 Fila de solicitações

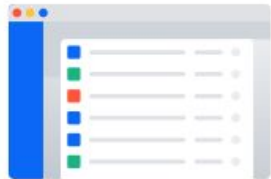
🔍 Todos os tipos ▼

Nome ↑	Chave ↕	Tipo	Lead ↕	URL	Marcado com estrelas
 Projeto 1 - GCM	P1G	 Software	 Marcos Miguel		...
 Projeto 2 - GCM	P2G	 Software	 Marcos Miguel		...

Criando o Projeto



Criando o Projeto



Controle de bugs (anteriormente Desenvolvimento de software básico)

Gerencie uma lista de tarefas e erros de desenvolvimento Ótimo para equipes que não precisam de um quadro

Alterar template

Passo 1

Criar projeto

Passo 2

Nome

teste

> Avançado

Criar

Passo 3


Escolha um template clássico

Os templates clássicos têm todos os recursos e tecnologias que você espera. Eles são criados e gerenciados centralmente por um administrador do Jira, com o uso de esquemas.

Todos os tipos ▾

Texto


[Importe seu trabalho](#)



Kanban

Monitore o trabalho em um fluxo contínuo para ter equipes ágeis Ideal para equipes que controlam o volume de trabalho com um backlog.

O que há nisso? Selecionar




Scrum

Gerencie histórias, tarefas e fluxos de trabalho para uma equipe do Scrum Para equipes que entregam trabalho regularmente.

O que há nisso? Selecionar

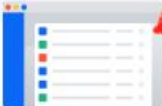
TESTAR GRATUITAMENTE



Central de atendimento para clientes externos

Reúna e responda rapidamente a relatórios de bugs e solicitações de suporte dos seus clientes.


Saiba mais Testar o Jira Service Desk



Controle de bugs (anteriormente



Gestão de projetos



Acompanhamento de tarefas



Projeto 1 - GCM
Projeto de software

Projetos / Projeto 1 - GCM

Itens abertos

Dar feedback

Voltar para o projeto



[Pesquisa avançada](#)

Criando Ticket/Tarefa

Itens abertos

Meus itens abertos

Reportado por mim

Todos os itens

Itens abertos

Itens concluídos

Visualizado recentemente

Criado recentemente

Resolvido recentemente

Atualizado recentemente

[Ver todos os filtros](#)



Nenhum item foi encontrado em "Itens abertos"

Selecione um filtro diferente ou crie um novo item

[Atualizar](#)

Criar item

Importar itens

Configurar campos ▾

Projeto*



Projeto 1 - GCM (P1G)



Tipo de item*



Bug



Alguns tipos de itens estão indisponíveis devido a configuração de campo incompatíveis e/ou associações de fluxo de trabalho.

Resumo*

Componentes

Nenhum

Descrição

Style ▾

B

I

U

A

▾

³A ▾



≡

≡



+

▾




☐ Criar outro

Criar

Cancelar

Relatórios Gerenciais e Configurações



Projeto 1 - GCM
Projeto de software

Projetos / Projeto 1 - GCM

Itens abertos

Pesquisa avançada

[Dar feedback](#)

Nenhum item foi encontrado em "Itens abertos"

Selecione um filtro diferente ou crie um novo item

[Atualizar](#)

Relatórios

Configurações do sistema



 **Time 1 - Scrum**
Projeto de software

 **quadro T1S**
Painel

 **Backlog**

 Sprints ativos

 Relatórios

 Versões

 Itens e filtros

 Páginas

 Componentes

 Adicionar Item

 Configurações do...

Projetos / Time 1 - Scrum / quadro T1S

Backlog

 Share 



Somente meus itens

Recentemente atualizados

quadro Sprint 1 1 pendência
16/mai/19 8:03 PM • 30/mai/19 8:03 PM

VERSÕES
ÉPICOS



☒ Refatorar pagina cadastro de aluno

 T1S-1 

Backlog 0 pendências

Criar sprint

Seu backlog está vazio.

+ Criar item

Jira - SCRUM





Time 1 - Scrum
Projeto de software



quadro T1S
Painel



Backlog



Sprints ativos



Relatórios



Versões



Itens e filtros



Páginas



Componentes



Adicionar Item



Configurações do...

Projetos / Time 1 - Scrum / quadro T1S

quadro Sprint 1



5 dias restantes

Completar sprint



Somente meus itens

Recentemente atualizados

ITENS PENDENTES

EM ANDAMENTO

ITENS CONCLUÍDOS

Refatorar pagina cadastro de aluno



T1S-1 DM

Jira - SCRUM -> Sprint



Time 3 - Scrum
Projeto de software



Voltar para o projeto

Relatórios

Todos os relatórios

AGILIDADE



quadro T3S
Painel



Gráfico de burndown

Gráfico de burnup

Relatório do sprint

Gráfico de velocidade

Diagrama de fluxo cumulat...

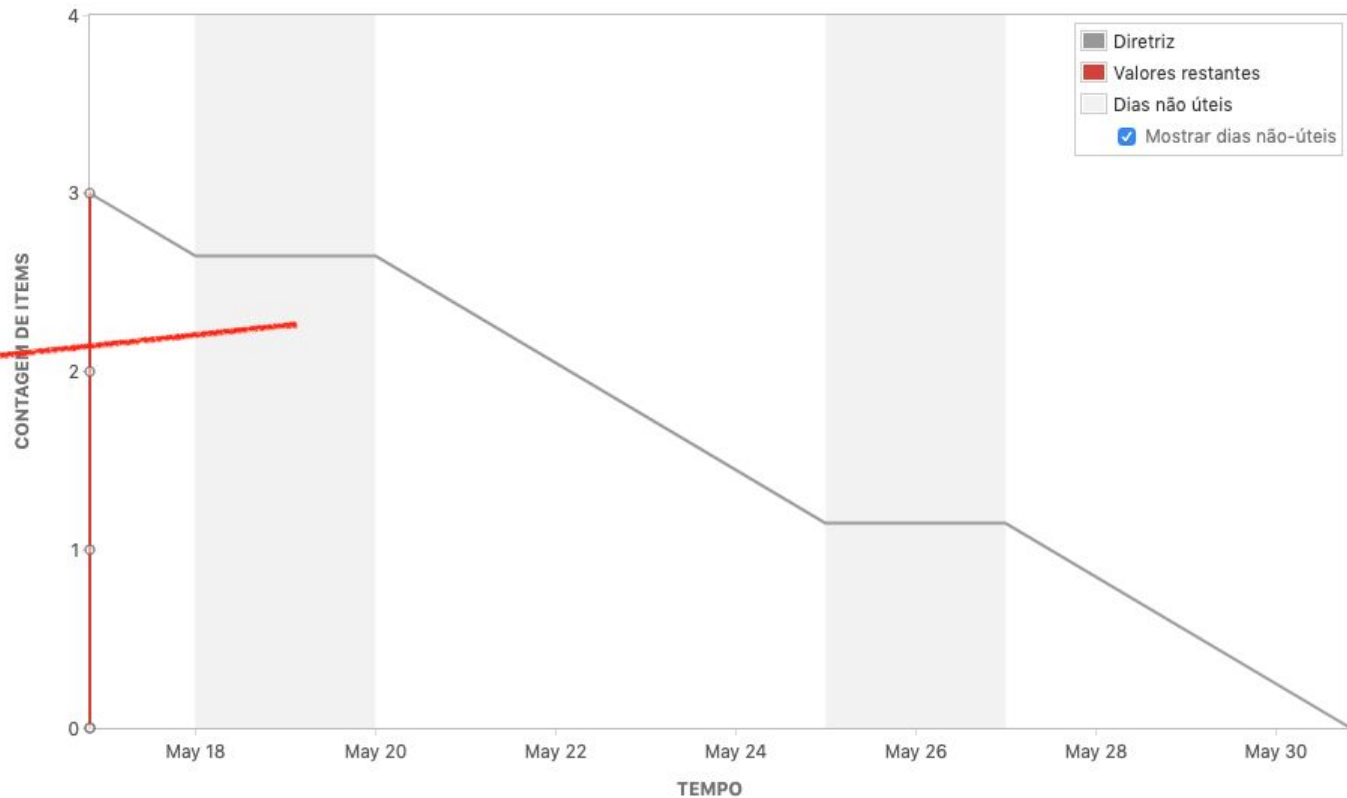
Relatório da versão

Relatório epic

Gráfico de controle

Projetos / Time 3 - Scrum / quadro T3S / Relatórios

Gráfico de burndown





Time 3 - Scrum
Projeto de software



Voltar para o projeto

Relatórios

Todos os relatórios

AGILIDADE



quadro T3S
Painel



Gráfico de burndown

Gráfico de burnup

Relatório do sprint

Gráfico de velocidade

Diagrama de fluxo cumulat...

Relatório da versão

Relatório epic

Gráfico de controle

Gráfico de burndown de e...

Projetos / Time 3 - Scrum / quadro T3S / Relatórios

Relatório de Sprint



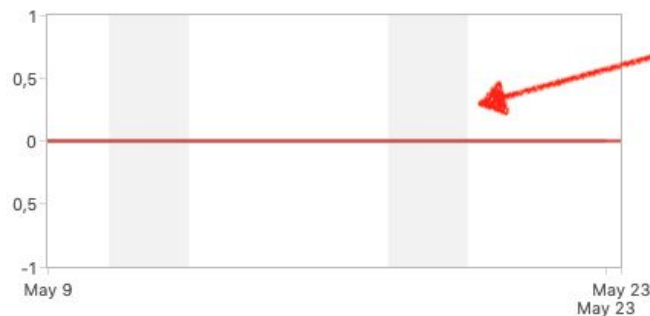
Sobre a regularidade de se manter um aumento excessivo de escopo.

[Ocultar esta informação](#)

Sprint 1



Sprint fechado, finalizado em [Marcos Miguel](#) 09/mai/19 10:56 AM - 23/mai/19 7:58 PM [Linkada páginas](#)



Relatório de Estado

* Item adicionada ao sprint depois do tempo de início

Items concluídas

[Visualizar no Navegador de Items](#)

Chave	Resumo	Tipo de item	Prioridade	Status	Story Points (-)
T3S-5 *	Teste	Bug	↑ Medium	ITENS CONCLUÍDOS	-



Time 3 - Scrum
Projeto de software



Voltar para o projeto

Relatórios

Todos os relatórios

AGILIDADE



quadro T3S
Painel



Gráfico de burndown

Gráfico de burnup

Relatório do sprint

Gráfico de velocidade

Diagrama de fluxo cumulati...

Relatório da versão

Relatório epic

Gráfico de controle

Projetos / Time 3 - Scrum / quadro T3S / Relatórios

Diagrama de fluxo cumulativo

