

# Repositórios GIT

Allinger Lima Medeiros  
Fernando Benedito

Laboratório de Sistemas Distribuídos Inteligentes (LSDi)  
Universidade Federal do Maranhão (UFMA)  
<http://www.lsd.ufma.br>

Janeiro de 2018



# Objetivo

Descrever os tipos de repositórios, a estrutura padrão dos repositórios de código e/ou documentação, o fluxo de trabalho adotado e introduzir os comandos do git que irão permitir a execução desse fluxo de trabalho.

Para alcançar esse objetivo, esta apresentação foi organizada com as seguintes seções:

- Código/Documentação
  - Estrutura dos repositórios do LSDi;
  - Fluxo de trabalho;
  - Comandos git;
- Artigos
  - Overleaf;
  - Git;



# Sumário

- 1 Código/Documentação
  - Introdução
  - Configurações iniciais
  - Branches
  - Trabalhando localmente
  - Trabalhando com o repositório de origem
  - Integração
  - Tutoriais em vídeo
- 2 Artigos
  - Overleaf
  - Repositórios git







## Criar repositório (Administrador do Laboratório)

Para criar o repositório de um projeto (Administrador do Laboratório):

- criar usuário e grupo do projeto no LDAP
- `ssh <dali>@lsdi.ufma.br`
- `mkdir /gitroot/<repositório do seu projeto>`
- `sudo chown -R <usuario>:<grupo> /gitroot/<repositório do seu projeto>`
- `sudo chmod -R g+ws /gitroot/<repositório do seu projeto>`



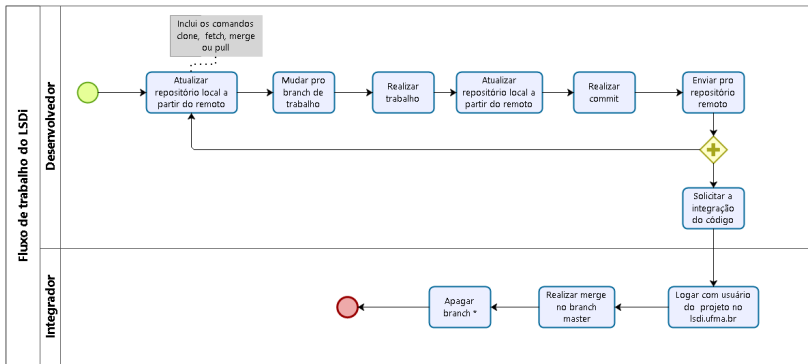
# Criar repositório (Usuário ou Integrador)

Para criar o repositório de um projeto (Usuário ou Integrador):

- `ssh <usuario do projeto>@lsdi.ufma.br`
- `cd /gitroot/<repositório do seu projeto>`
- `git init --shared=group`  
`--template=/usr/share/git-core/projeto && git reset --hard`
- `git add *`
- `git commit -m "Commit Inicial"`



# Fluxo de trabalho



# Clonar repositório

Para clonar o repositório para seu computador:

- 1 Acesse no terminal a pasta na qual deseja manter sua cópia do repositório
- 2 `git clone <usuario>@lsdi.ufma.br:/gitroot/<repositório do seu projeto>`
- 3 digite a senha do seu usuário no sistema do laboratório

Provavelmente você executará esse procedimento apenas uma vez. No caso de já ter clonado o repositório em momento anterior, não deixe de atualizar o seu repositório com o comando `git pull` ou `git fetch` antes de iniciar um novo fluxo de trabalho.





# Configurar usuário

Antes de realizar outras operações com o git como commits, é necessário configurar nome e email do usuário git. Isso serve para que o git salve o autor de cada commit. Para tanto bastam os dois comandos a seguir:

- 1 `git config --global user.name "<Nome>"`
- 2 `git config --global user.email "<nome@lsdi.ufma.br>"`



# Branches

Branches mantém ramificações diferentes de um mesmo repositório e permitem que novas funcionalidades possam ser desenvolvidas separadamente sem que modificações feitas em um branch interfiram com as outras.

O branch principal de um repositório é chamado de master e provavelmente esse é o branch ativo no repositório que você copiou. Ele deve conter apenas o que já foi testado e/ou ratificado. O ideal é que cada usuário possua seu próprio branch.

## Importante

**NÃO FAÇA MODIFICAÇÕES DIRETAMENTE NO BRANCH MASTER!**



# Listando os branches

Listar o branch ativo e os branches visíveis localmente:

- `git branch`

Listar todos os branches:

- `git branch -a`

Após clonar o repositório, o primeiro comando mostrará apenas o branch que está ativo no repositório de origem (no servidor do laboratório), mas o segundo mostrará todos os branches presentes no repositório de origem.



# Trocando o branch ativo

Para trocar o branch ativo por uma dos branches listados (locais e remotas):

- `git checkout <nome do branch>`

Para criar um branch e mudar para ele imediatamente:

- `git checkout -b <nome do branch>`



# Commits

Commits são versões diferentes de um branch.

Deve-se criar um commit sempre que forem feitas modificações no repositório.

É possível voltar a um commit anterior.

Para ver uma lista com os commits feitos:

- git log



# Salvando modificações localmente

Antes de salvar alguma modificação você deve trocar o branch ativo caso ele seja o master.

O git mantém o controle de versão apenas dos arquivos que o usuário especificar.

Caso novos arquivos sejam criados e se deseje que o git acompanhe a mudanças feitas nesses arquivos é necessário explicitar isso ao git.

Adicionar um arquivo a lista de arquivos versionados ou área de seleção:

- `git add <nome do arquivo>`

Adicionar todos os arquivos da pasta atual e suas subpastas:

- `git add *`



# Salvando modificações localmente

Para salvar o estado atual dos arquivos em uma versão deve-se criar um novo commit:

- `git commit -m "<mensagem>"`

A mensagem deve conter explicitamente que projetos esse commit está modificando.

Antes de subir essas modificações para o branch master do servidor do laboratório você deve resolver eventuais conflitos.



# Baixando as modificações feitas no repositório de origem

Baixar as modificações de todas os branches (não realiza a integração com o repositório local):

- git fetch

Baixar as modificações feitas no branch ativo e realizar a integração:

- git pull

Baixar as modificações feitas em todos os branches visíveis localmente e realizar a integração:

- git pull -all





# Baixando as modificações feitas no repositório de origem

O comando merge adiciona as modificações feitas em uma branch a outra, caso haja conflitos o git os acusará:

- `git merge <nome do branch>`

Listará as ferramentas que podem ser utilizadas para resolução de conflitos:

- `git mergetool -tool-help`

Abre a ferramenta para resolução de conflitos especificada:

- `git mergetool -tool=<tool>`



# Salvando modificações remotamente

Você pode então subir esse novo commit para o servidor do laboratório:

- `git push origin <nome do branch>`

esse comando sobe todos os commits do branch atual para o repositório de origem.

## Nota

Nos repositórios de código e e documentação, não é possível realizar o push diretamente para o branch ativo no repositório de origem.



# Integração com o master

Caso a modificação/funcionalidade desenvolvida no seu branch esteja pronta para ser adicionada ao branch master, dois passos devem ser seguidos.

- 1 Realizar o merge localmente e resolver eventuais conflitos;
- 2 Solicitar ao integrador que o merge de sua branch com o master seja feito no servidor do laboratório;



# Resolução local de conflitos

Primeiro é preciso garantir que seu master esteja atualizado:

- git checkout master
- git pull

Então volte para a sua branch e tente o merge com o master

- git checkout <branch>
- git merge master

O git acusará caso hajam conflitos. Você deve resolvê-los por exemplo editando os arquivos onde foram detectados conflitos. Os tutorias em vídeo que seram apresentados posteriormente demonstram a resolução de conflitos utilizando editores de texto e IDEs.



# Integrando modificações no repositório remoto (Desenvolvedor)

Primeiro você deve fazer o push da sua branch:

- `git commit -m "<mensagem>"` (caso não tenha sido feito após a resolução de conflitos)
- `git push`

Então solicite ao Integrador que o merge com o master no repositório remoto seja feito.



# Integrando modificações no repositório remoto (Integrador)

Logue com o usuário do projeto/grupo no servidor do LSDi e faça o merge:

- `ssh <projeto>@lsdi.ufma.br`
- `cd /gitroot/<repositório do seu projeto>`
- `git merge <branch>`



# Tutoriais Eclipse

- Clonando Repositório: <https://youtu.be/LbtiXtHunJs>
- Enviando projeto para o repositório remoto:  
<https://youtu.be/3cYJhrAl7Mw>
- Importando projeto de um repositório git remoto:  
<https://youtu.be/ouTziuQULGM>
- Resolvendo conflitos entre versões diferentes de um código no mesmo branch: <https://youtu.be/23HHJjJGQRg>
- Criando e alternando entre branches:  
<https://youtu.be/D-PCjPSsWo8>
- Realizando merge entre branches diferentes:  
<https://youtu.be/fU-Vz0D4hck>



# Tutoriais Linha de Comando

- Vídeo completo: <https://youtu.be/1MY9sx9QCsc>
- Clone <https://youtu.be/1MY9sx9QCsc?t=35s>
- Salvando (add, commit e push)  
<https://youtu.be/1MY9sx9QCsc?t=1m5s>
- Merge com o master  
<https://youtu.be/1MY9sx9QCsc?t=2m30s>
- Resolução de conflitos  
<https://youtu.be/1MY9sx9QCsc?t=3m20s>
- Pull das modificações remotas  
<https://youtu.be/1MY9sx9QCsc?t=3m30s>





# Tutoriais Android Studio/Jetbrains

- Vídeo completo: <https://youtu.be/zrcDNFSJEIY>
- Clone <https://youtu.be/zrcDNFSJEIY?t=1s>
- Salvando (add, commit e push)  
<https://youtu.be/zrcDNFSJEIY?t=1m35s>
- Pull das modificações remotas  
<https://youtu.be/zrcDNFSJEIY?t=3m>
- Merge e resolução de conflitos  
<https://youtu.be/zrcDNFSJEIY?t=3m35s>



# Sumário

- 1 Código/Documentação
  - Introdução
  - Configurações iniciais
  - Branches
  - Trabalhando localmente
  - Trabalhando com o repositório de origem
  - Integração
  - Tutoriais em vídeo
- 2 Artigos
  - Overleaf
  - Repositórios git



# Artigos

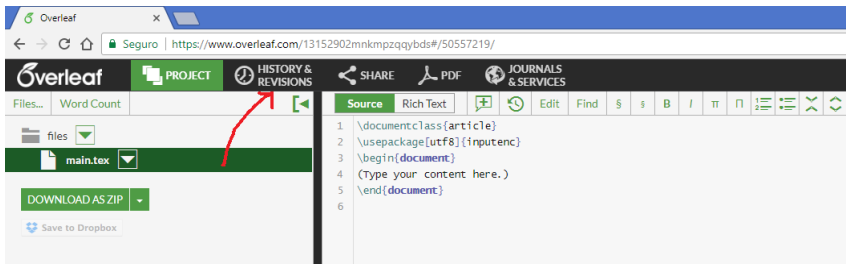
Duas opções:

- Overleaf: <https://www.overleaf.com/>
- Git



# Overleaf - Revisões

- É possível criar revisões através do link HISTORY & REVISION.



# Overleaf - Revisões

- Adicione um nome à revisão e clique sobre o botão ADD LABEL.

152902mnkmpzqybdsh/50557219/

SHARE PDF JOURNALS & SERVICES

Source Rich Text

```
1 \documentclass{article}
2 \usepackage[utf8]{inputenc}
3 \begin{document}
4 (Type your content here.)
5 \end{document}
6
```

### Versions

Looking for more? Full project history is available on our [Pro plan](#) and above.

versao1.0 **ADD LABEL**

**Project created**  
saved [about 12 hours ago](#) [compare](#) [restore](#)

Overleaf automatically saves the latest version of your project.  
You can also label and save versions here yourself for future reference.

# Overleaf - Revisões

- Após a adição da revisão, três opções são exibidas do lado direito: compare, restore e delete.

I52902mnkmpzqqybdS#/50557219/

SHARE PDF JOURNALS & SERVICES

Source Rich Text

```

1 \documentclass{article}
2 \usepackage[utf8]{inputenc}
3 \begin{document}
4 (Type your content here.)
5 \versao 2.0
6 \versao 3.0
7 \end{document}
8

```

### Versions

Looking for more? Full project history is available on our [Pro plan and above](#).

label current version ADD LABEL

<b>versao2.0</b> saved by Allinger Medeiros <i>less than a minute ago</i>	compare	restore	delete
<b>versao1.0</b> saved by Allinger Medeiros <i>about a minute ago</i>	compare	restore	delete
<b>Project created</b> saved <i>about 12 hours ago</i>	compare	restore	

Overleaf automatically saves the latest version of your project.  
You can also label and save versions here yourself for future reference.



# Overleaf - Comparação

- Exemplo de comparação entre revisões.

The screenshot displays the Overleaf web interface in 'COMPARE' mode. The top navigation bar includes 'PROJECT', 'HISTORY & REVISIONS', and 'COMPARE'. The left sidebar shows a file tree with 'main.tex' selected. The main area is split into two panels: 'Current version' and 'Compared version (read only)'. The 'Current version' panel shows the source code of 'main.tex' with lines 5 and 6 highlighted in blue. Line 5 is 'versao 2.0' and line 6 is 'versao 3.0'. The 'Compared version' panel shows the same code but with line 5 as '\end{document}' and line 6 as an empty line. A blue arrow points from the highlighted line 6 in the 'Current version' to the corresponding line 6 in the 'Compared version'.

```
1 \documentclass{article}
2 \usepackage[utf8]{inputenc}
3 \begin{document}
4 (Type your content here.)
5 versao 2.0
6 versao 3.0
7 \end{document}
```

```
1 \documentclass{article}
2 \usepackage[utf8]{inputenc}
3 \begin{document}
4 (Type your content here.)
5 \end{document}
6
```

# Overleaf - Histórico

- Podem ser visualizadas, ao clicar no link Show File History, alterações realizadas nas últimas 24 horas sobre o arquivo corrente/ativo.

verleaf.com/13152902mnkmpzqqybd#50557219/

HISTORY & REVISIONS SHARE PDF JOURNALS & SERVICES

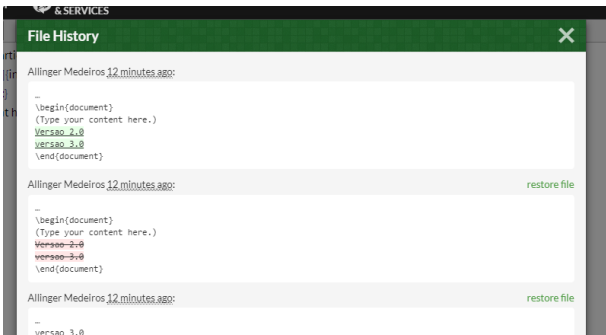
Source Rich Text Edit Find \$ s B / π Π 1/2 ≡ ≡

```
1 \documentclass{article}
2 \usepackage{utf8}[inputenc]
3 \begin{document}
4 (Type your content here.)
5 Versao 2.0
6 versao 3.0
7 \end{document}
8 |
```



# Overleaf - Histórico

- No canto superior direito de cada alteração registrada, existe um link para restaurar uma versão anterior do arquivo.

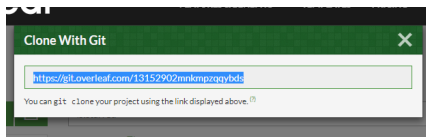
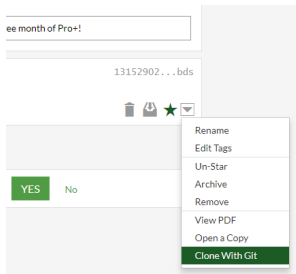


The screenshot shows the 'File History' window in Overleaf. It lists three versions of a document, each with a 'restore file' link in the top right corner. The first version is the current one, with 'Versao 2.0' and 'Versao 3.0' highlighted in green. The second version has 'Versao 2.0' and 'Versao 3.0' highlighted in red. The third version has 'Versao 3.0' highlighted in red.

```
& SERVICES
File History
Allinger Medeiros 12 minutes ago:
--
\begin{document}
(Type your content here.)
Versao 2.0
Versao 3.0
\end{document}
restore file
Allinger Medeiros 12 minutes ago:
--
\begin{document}
(Type your content here.)
Versao 2.0
Versao 3.0
\end{document}
restore file
Allinger Medeiros 12 minutes ago:
--
Versao 3.0
```

# Overleaf - Git

- Também é possível clonar um projeto do Overleaf.



# Overleaf - Git

- Com o projeto aberto, a URL para realização do clone pode ser obtida ao clicar sobre o link SHARE.

The screenshot shows the Overleaf interface with the 'SHARE' button highlighted by a red arrow. The 'Share' dialog box is open, displaying the following information:

- Read & Edit Link:** <https://www.overleaf.com/13152902mnkmpzqybd>
- Read Only Link:** <https://www.overleaf.com/read/wtdywrghftqb>
- Clone With Git:** <https://git.overleaf.com/13152902mnkmpzqybd>

The 'Clone With Git' link is highlighted with a red arrow. Below the 'Clone With Git' link, there is a note: "You can `git clone` your project using the link displayed above."

The 'Share' dialog box also includes a 'Publish' section with the text: "Publish to the Overleaf Gallery to have your project featured on our site." and a 'More Sharing Options' section with a link: "Make this a protected project (requires Pro)".

# Overleaf - Git

- As revisões salvas anteriormente são exibidas como commits no repositório clonado.

```
MINGW64:/c:/Allinger/workspaces/repositorio/overleaf/13152902mnkmpzq...
kmpzqybd (master)
$ git log
commit 214cd6de4d3ef5d1625a2b340f447bc8691e323c (HEAD -> master, origin/master,
origin/HEAD)
Author: Allinger Medeiros <allingermedeiros@gmail.com>
Date: Fri Jan 12 03:24:47 2018 +0000

    Update on Overleaf.

commit 0004bcaa1dd68f0b6bd1e5c947199bb85f15bd8
Author: Allinger Medeiros <allingermedeiros@gmail.com>
Date: Fri Jan 12 02:41:25 2018 +0000

    versao2.0

commit ad1f1ec5b52b4ef8e8cac700329fa330cfa8e4fa
Author: Allinger Medeiros <allingermedeiros@gmail.com>
Date: Fri Jan 12 02:40:22 2018 +0000

    versao1.0

Allinger@PiradoPC MINGW64 /c:/Allinger/workspaces/repositorio/overleaf/13152902mn
kmpzqybd (master)
$ git log
```

# Overleaf - Git

Alguns links que tratam dessa integração entre o Overleaf e o git:

- [https://www.overleaf.com/help/  
233-how-do-i-connect-an-overleaf-project-with-a-repo-c](https://www.overleaf.com/help/233-how-do-i-connect-an-overleaf-project-with-a-repo-c)
- [https://www.overleaf.com/help/  
230-how-do-i-push-a-new-project-to-overleaf-via-git](https://www.overleaf.com/help/230-how-do-i-push-a-new-project-to-overleaf-via-git)
- [https://www.overleaf.com/blog/  
195-new-collaborate-online-and-offline-with-overleaf-a](https://www.overleaf.com/blog/195-new-collaborate-online-and-offline-with-overleaf-a)
- [https://www.overleaf.com/articles/  
git-and-overleaf-integration/qmdncpnqwfxx](https://www.overleaf.com/articles/git-and-overleaf-integration/qmdncpnqwfxx)



# Repositórios git

- Não seguirá um fluxo de trabalho definido;
- Não existirá o papel do integrador;
- Não há restrições quanto o branch master;
- Recomendações:
  - Atualizar o seu repositório local com frequência;
  - Utilizar o bom senso e se comunicar com os demais membros.



# Criar repositório (Administrador do Laboratório)

Para criar o repositório de um artigo (Administrador do Laboratório):

- criar usuário e grupo do projeto no LDAP (apenas se não tiver sido criado anteriormente)
- `ssh <dali>@lsdi.ufma.br`
- `mkdir /gitroot/artigos/<repositório do seu artigo>`
- `cd /gitroot/artigos/<repositório do seu artigo>`
- `sudo chown -R <usuario>:<grupo>`  
`/gitroot/artigos/<repositório do seu artigo>`
- `sudo chmod -R g+ws /gitroot/artigos/<repositório do seu artigo>`



# Criar repositório (Usuário ou Integrador)

Para criar o repositório de um artigo (Usuário ou Integrador):

- `ssh <usuario do projeto>@lsdi.ufma.br`
- `cd /gitroot/artigos/<repositório do seu artigo>`
- `git init --bare --shared=group`





# Fontes de informação adicional

Mais informações sobre o git podem ser consultadas nos seguintes sites:

- <https://git-scm.com/book/pt-br/v2>
- <https://git-scm.com/book/pt-br/v1>

