



Git

www.unifei.edu.br - Instituto de Ciências Tecnológicas

ECO* – Conteúdo comum

Prof. Eduardo R. Felipe

Versionador de arquivos

Git

O que é Git?

- É um controle de versão de arquivos distribuído, que permite:
 - Salvar versões de arquivos de um projeto (snapshots)
 - Armazenar estas versões de forma transparente ao usuário
 - Permite que equipes trabalhem com cópias (*branches*) do mesmo código para posterior sincronismo (*merge*) em uma versão final
- Conceitos importantes:
 - Repositório – Local onde os recursos serão versionados
 - Branch – Estrutura que cria um “marco” de versionamento e que pode ser combinada com outras *branches*

Instalação

- Antes de realizar a instalação, confira se o Git já se encontra disponível no seu sistema operacional.
 - Pelo prompt de comando, pode-se digitar: `> git --version`
- *Caso não tenha o git*, pode-se baixar e instalar o software pela página:
 - <https://www.git-scm.com/downloads>
- O git é uma ferramenta (software) que é usado comumente pela linha de comando do prompt, mas há versões gráficas que facilitam seu uso:
 - <https://www.git-scm.com/downloads/guis>
- Obs: Nestes slides, o prompt de comando é representado pelo sinal `>`

Inicialização de repositório

- Ao iniciar um projeto (ou com um projeto já em andamento), pode-se digitar na pasta (raiz) do projeto:

```
> git init
```

- Na medida que os arquivos forem criados e alterados, deve-se adicioná-los ao versionador com o comando:

```
> git add <nomeArq>
```

- Ou, para múltiplos arquivos (todos):

```
> git add .
```

Status e confirmação

- Para verificar o status dos arquivos monitorados pelo Git, digita-se (na pasta do projeto):

```
> git status
```

- Para confirmar as alterações nos arquivos e criar um marco:

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

- Para verificar os commits criados:

```
> git log
```

Verificações

- Para trocar o nome da branch atual para outro nome

```
> git branch -m <new-branch-name>
```

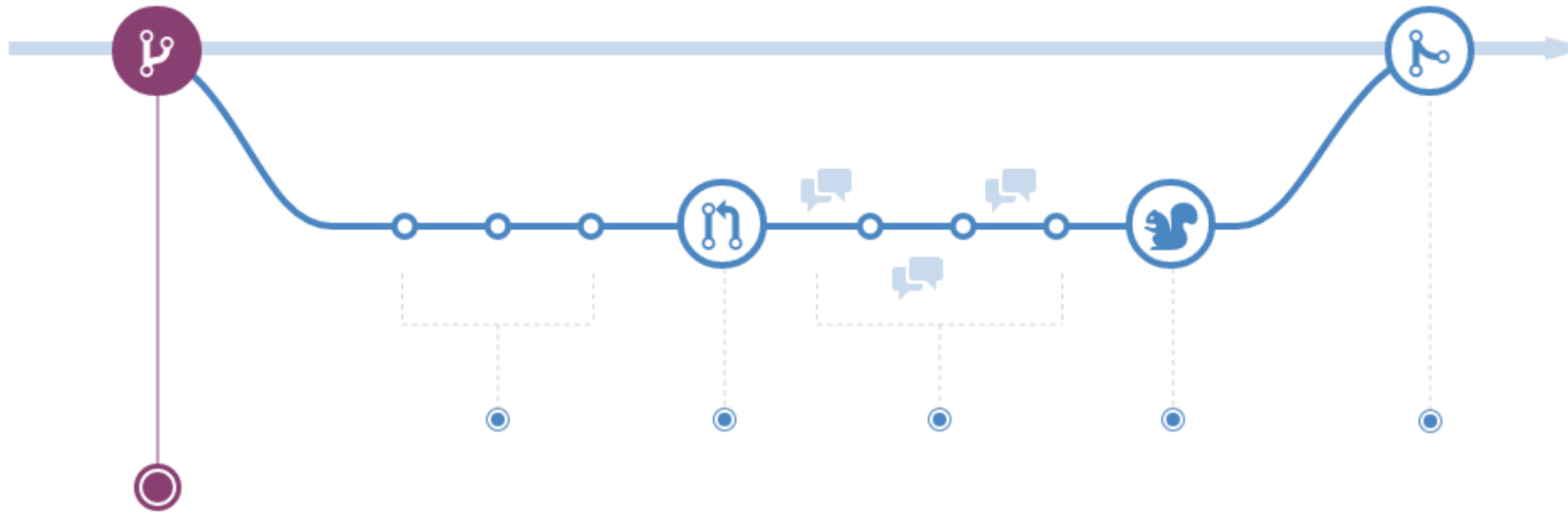
- Para verificar as alterações do último commit

```
> git show
```

Identidade (em computadores não pessoais)

- Antes do comando *push* (empurrar), para que os arquivos “subam” para o servidor git (*origin*), em seu primeiro processo ou em computadores não pessoais, é necessário registrar sua identidade no repositório local.
- Se você já tentou enviar os arquivos (push), o help do git deve ter exibido o seguinte comando:

```
> git config --global user.name "<seu nome de usuário git>"  
> git config --global user.email "<seu email de usuário git>"
```
- De modo que **NÃO** use a diretiva **--global** em computadores públicos! Ou seja, use esta diretiva somente em sua máquina pessoal para que os demais repositórios reconheçam seu perfil.



Ramificações

Branches

Ramificações (branches)

- Para criar uma nova ramificação do projeto (branch) a fim de preservar o código existente, usa-se o comando:

```
> git branch <nomeBranch>
```

- Para verificar as branches disponíveis localmente:

```
> git branch
```

- Para mudar de uma branch para outra:

```
> git checkout <nomeBranchDestino>
```

Para combinar ramificações

- Para combinar ramificações, entre na *branch* destino (caso não esteja):

```
> git checkout <nomeBranch>
```

- E use o comando:

```
> git merge <branchAserCombinada>
```

Excluir ramificações

- Para listar as branches disponíveis no projeto:

```
> git branch
```

- Para deletar alguma *branch* que não seja mais útil ao projeto, use o comando:

```
> git branch -D <nomeBranch>
```



GitHub

Publicando o projeto

Github, Bitbucket

GitHub

- Criar uma conta no site: <https://github.com>
- Clicar em New
- Preencher os dados para um novo repositório e nomeá-lo
 - Obs: **Não** crie o readme e .gitignore e a licença neste momento (poderá fazê-lo depois)
- Pode-se escolher um tipo de licença e confirmar
- As seguintes orientações devem ser exibidas:
 - Quick setup — if you've done this kind of thing before
 - **...or create a new repository on the command line**
 - **...or push an existing repository from the command line**
- Copie a URL gerada pelo GitHub, ela será usada para associar nosso projeto pessoal com o projeto na nuvem do GitHub

Associando o projeto local ao GitHub

- **Importante**

- Por padrão, em minha máquina pessoal, ao criar um projeto git, o inicializador cria uma branch chamada **master**. Já no GitHub, ao criar um projeto, a branch principal é nomeada como **main**.
- Para facilitar o entendimento e trânsito de branches, é recomendável renomear uma das branches para padronizar a nomenclatura. *Caso você tenha a mesma situação.*
- Para renomear a branch local de **master** para **main**, use o comando:

```
> git branch -m <novoNome>
```
- Após conferido se as branches (local e remota) possuem o mesmo nome (a menos que deseje subir outra branch para o repositório remoto), pode-se associar os repositórios com o comando:

```
> git remote add origin <urlGeradaNoGitHub>
```
- Para conferir a associação entre repositório local e remoto, use:

```
> git remote -v
```

Enviar o arquivo para o GitHub

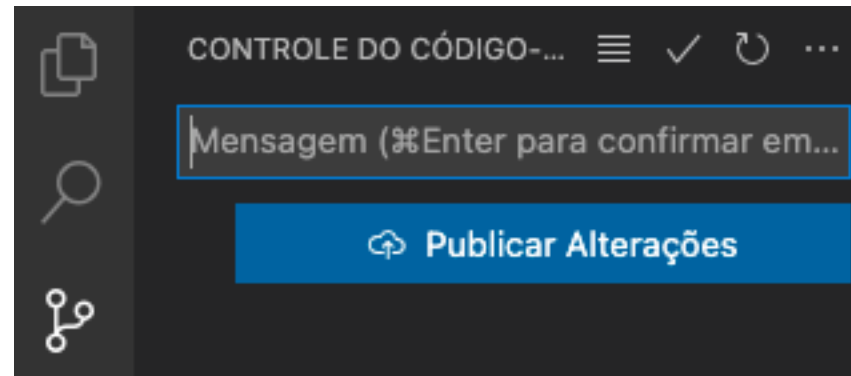
- Após a associação de repositório local com o repositório remoto, pode-se “empurrar” os arquivos locais para o repositório na nuvem:
 > `git push -u origin main`
- Na interface do GitHub, entrar no projeto e verificar a guia `<> Code`
- Para futuras alterações não é necessário associar o projeto novamente, basta usar a sequência (resumo):
 > `git add <arq>`
 > `git commit -m <descricao>`
 > `git push -u origin main`

Dica – Login para push

- Dica: A interface pode pedir seus dados de login no GitHub para proteger o repositório remoto. Para evitar esse preenchimento todas as vezes, pode-se configurar o SSH para o github “reconhecer sua máquina” como proprietária da conta. No seu ícone de perfil, entre em Settings >> [SSH](#).
- Ou ainda usar para este repositório local, o comando:
 > `git config credential.helper store`
- Dica: Para sempre inicializar o repositório local com o nome da *branch* como `main`, use:
 > `git config --global init.defaultBranch <name>`

Visual Studio Code

- Alguns editores e IDEs fornecem ferramentas nativas para este procedimento, a exemplo do VS Code
- Neste caso estes programas oferecem uma abordagem de alto nível (gráfica) para realizar estes comandos



Miscelânea

.gitignore, readme.md

Evitar arquivos no repositório remoto

- Para definir que determinados arquivos e/ou pastas não sejam enviados ao repositório remoto, crie um arquivo denominado:
`.gitignore`
- Obs: O arquivo começa com um ponto final “.”
- Abra-o em um editor de texto e adicione as referências, exemplo:
`*.log`
`build/`
`temp-*`

Arquivo de descrição do projeto

- Todo projeto deve ter um arquivo de descrição. Para repositórios no GitHub, o padrão é criar um arquivo com o nome:
`readme.md`
- Para formatar este arquivo com estilo, usa-se o formato markdown:
 - <https://github.com/tchapi/markdown-cheatsheet/blob/master/README.md>

Clonando um repositório
existente

Cópia (clone)

- Para baixar um repositório existente no GitHub para seu computador local, entre no repositório desejado no GitHub e pelo botão Code, copie a URL
- Entre em um diretório (pasta) de sua escolha, e use:
 - `> git clone <url>`
- Será criada uma sub-pasta com o nome do projeto

Resumo

Comandos

- > `git init` - inicio do controle de versionamento
- > `git add <nomeArq>` - adiciona o arquivo especificado ao monitoramento
- > `git add .` - adiciona todos os arquivos (e sub pastas) ao monitoramento
- > `git status` - exibe o status dos arquivos no monitoramento
- > `git commit -m "<Descricao>"` - grava os arquivos em um snapshot (momento)
- > `git log` - exibe todos os commits realizados
- > `git show` - exibe detalhes do último commit
- > `git branch <nomeBranch>` - cria uma nova branch
- > `git branch` - exibe as branches existentes

Comandos

- > `git checkout <nomeBranchDestino>` - alterna para a branch destino
- > `git merge <branchAserCombinada>` - combina duas branches (sendo destino a branch atual)
- > `git branch -D <nomeBranch>` - deleta a branch
- > `git branch -m <novoNome>` - renomeia a branch
- > `git remote add origin <urlGeradaNoGitHub>` - associa um repositório remoto ao repositório local
- > `git remote -v` - exibe o repositório remoto associado
- > `git push -u origin main` - envia os arquivos da branch atual (main) para o repositório remoto (origin)



- [https://training.github.com/downloads/pt BR/github-git-cheat-sheet.pdf](https://training.github.com/downloads/pt_BR/github-git-cheat-sheet.pdf)
- https://githubtraining.github.io/training-manual/#/01_getting_ready_for_class
- <https://www.toptal.com/developers/gitignore>
- <https://github.com>
- <https://bitbucket.org>
- Uma pessoa que realmente indico para vocês acompanharem, é o Fábio Akita:
 - <https://youtu.be/6Czd1Yetaac>
 - <https://youtu.be/6OokP-NE49k>



eduardo.felipe@unifei.edu.br