

IOS – Instituto de
Oportunidade Social

GIT 01 - Versionamento de código

Parte 01



- Controle de Versão
- O que é GitHub ?
- Repositório GitHub
- O que é Git ?
- Funcionamento do Git

IOS – Instituto de
Oportunidade Social

Controle de Versão



O controle de versão de um projeto é importante para ajudar pessoas a **gerenciarem alterações realizadas** em algum arquivo, através do **registro de todas as modificações** realizadas. Para a engenharia de software, o controle de versões é uma classe de sistemas responsáveis por gerenciar mudanças em programas de computadores, documentos, páginas web ou outras coleções de informação.

Um sistema de controle de versões (**VCS, Version Control System**) é um software computacional encarregado de gerenciar diferentes versões no desenvolvimento de um documento qualquer. Esse tipo de sistema é bastante utilizado em empresas e instituições de tecnologia e no desenvolvimento de software para o controle de versões de projetos. Além disso, ele é bastante comum no desenvolvimento de softwares livres. Os VCS mais comuns são: **CVS, Mercurial, Git e SVN** (Subversion); e as comerciais: **SourceSafe, TFS, PVCS** (Serena), **ClearCase** e **Azure Devops**.

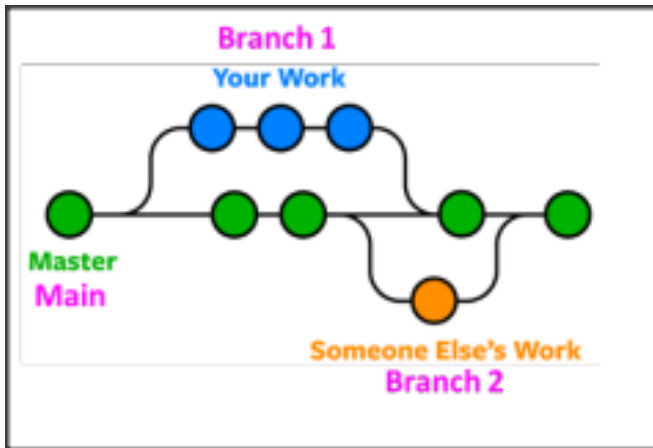
Vamos utilizar o **Git**, pois se integra com a ferramenta **GitHub**.

Vantagens do sistema de versionamento:

- **Controle do histórico:** esse tipo de sistema mantém um **histórico de versões** do documento/projeto, permitindo **resgatar versões** mais antigas e estáveis.
- **Trabalho em equipe:** o documento/projeto pode ser **compartilhado** com diversas pessoas e permite que elas trabalhem e realizem **modificações** nesse conjunto de informações **ao mesmo tempo**. Também, é possível controlar o **acesso** de cada usuário ou grupo de usuários.

- **Marcação** e resgate de **versões estáveis**: os desenvolvedores que utilizam um VCS podem marcar versões estáveis do projeto em desenvolvimento para ser possível resgatar essas versões se necessário.
- **Ramificação do projeto**: os sistemas de controle de versões permitem criar linhas diversas de desenvolvimento para que os **programadores trabalhem paralelamente** sem que uma linha interfira na outra ou interferir na linha principal do projeto.

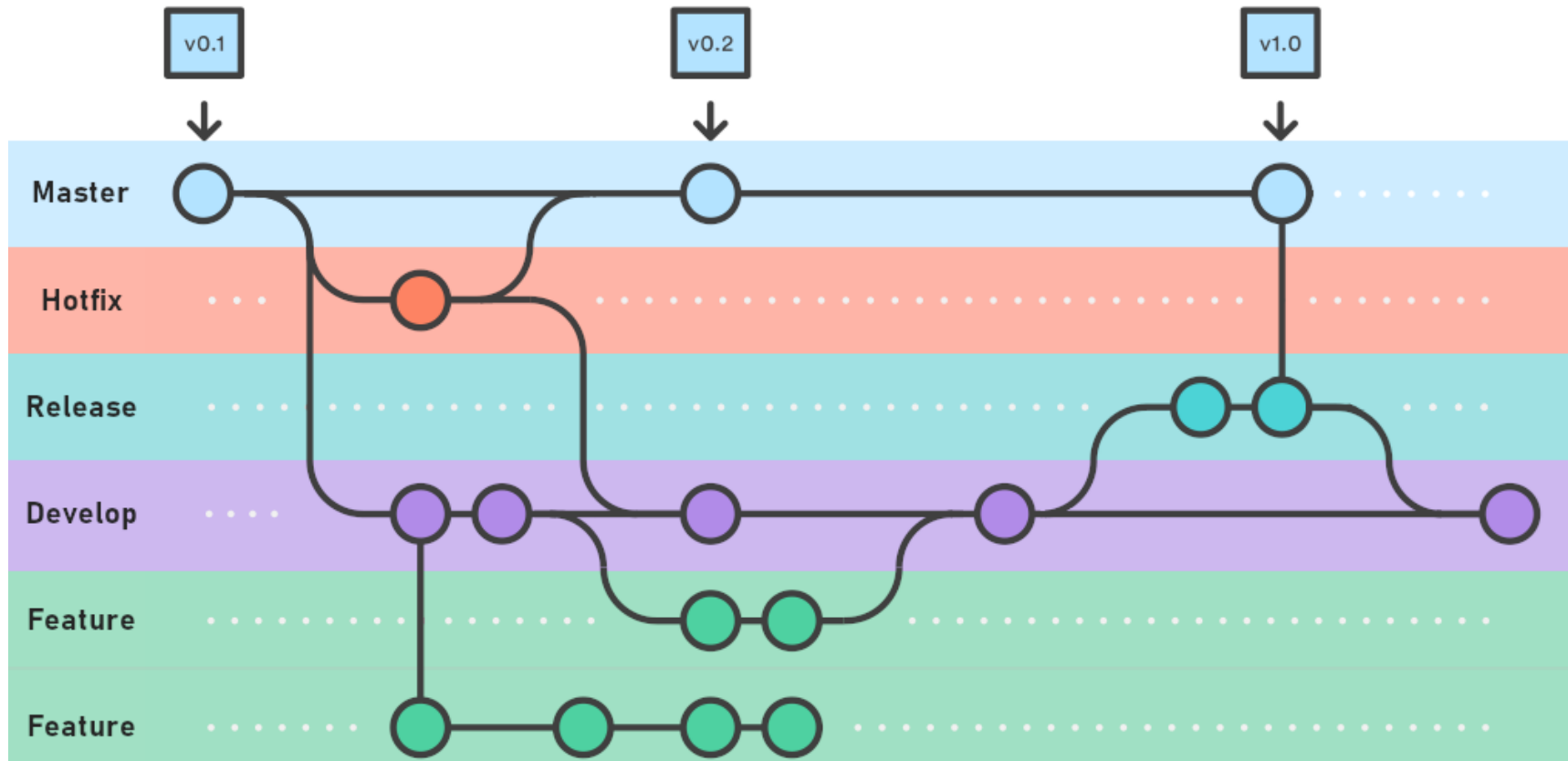
Branch é a duplicação de um objeto sob controle de versão, tal como um código fonte ou uma árvore de diretório, são ramificações ou bifurcações dentro de um projeto git.



Comando para criação de Branch:

git branch Nome_do_Branch

Conceito Git Flow



IOS – Instituto de
Oportunidade Social

O que é Github ?



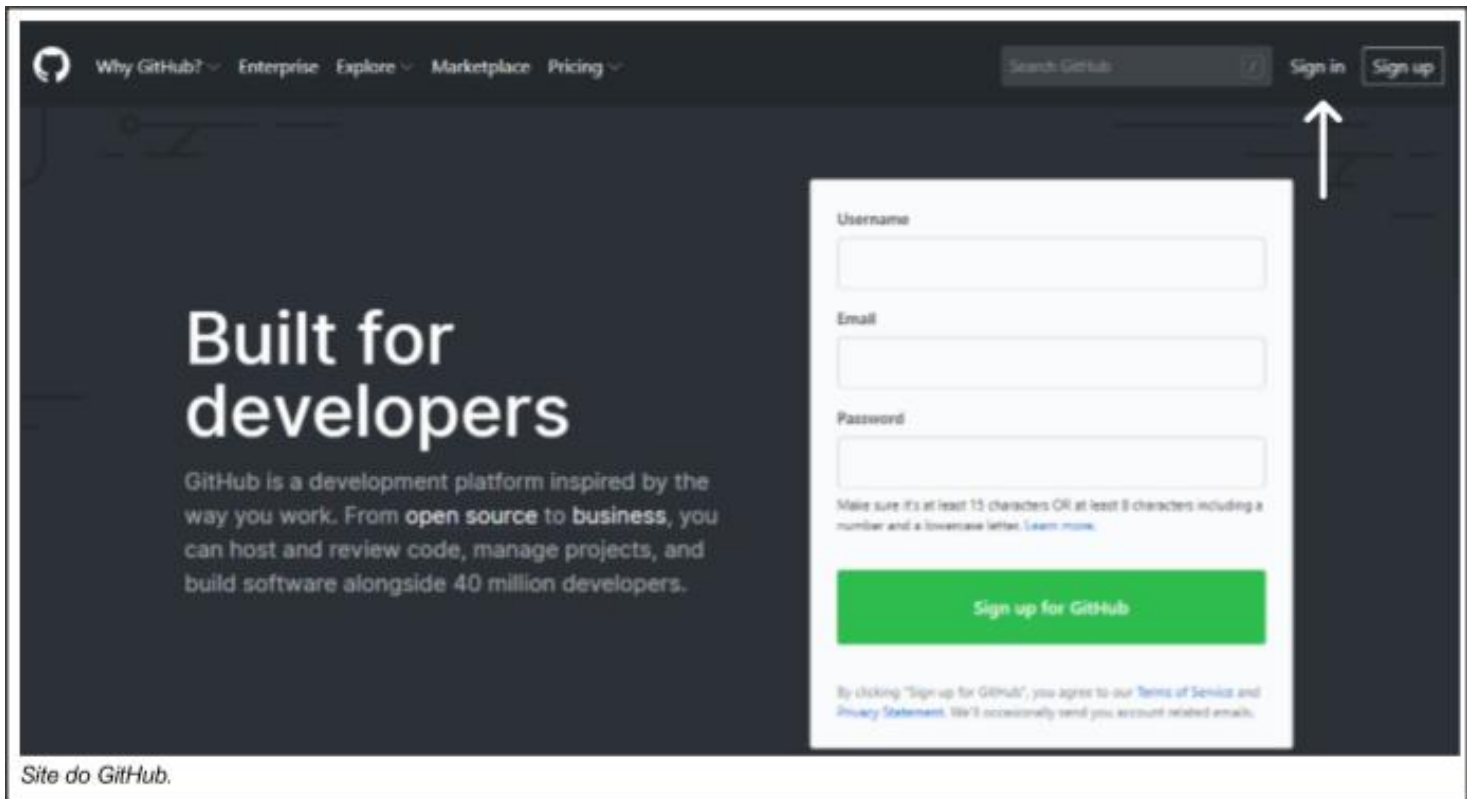
O que é Github ?

O GitHub é um **repositório que fornece a hospedagem** para o desenvolvimento de software e **controle de versão usando Git**. Ele oferece funcionalidades do Git como: controle distribuído de versões e um gerenciamento de código-fonte (SCM, Source Code Management) e, também, funcionalidades próprias como: rastreamento de erros, controle de acesso de colaboração, requisição de recursos, gerenciamento de tarefas, wikis, etc.



O que é Github ?

O site para você criar uma conta no GitHub é: <https://github.com/>, para criar sua conta você deve clicar em **Sign up**.



O que é Github ?



Planos do Github:

- **Free:** repositórios públicos e privados ilimitados, até três colaboradores, fórum de problemas (issues), monitoramento de bugs e ferramentas de gerenciamento de projetos.
- **Pro:** acesso ilimitado a todos os repositórios, número ilimitado de colaboradores, fórum de problemas (issues), monitoramento de bugs e ferramentas avançadas.

O que é Github ?

- **Team:** todas as funcionalidades mencionadas anteriormente, controle de **acesso de equipe** e gerenciamento de usuários.
- **Enterprise:** todas as funcionalidades do plano Team, **hospedagem própria** ou em nuvem, prioridade na prestação de suporte técnico, login único (single sign-on) e mais.



IOS – Instituto de
Oportunidade Social

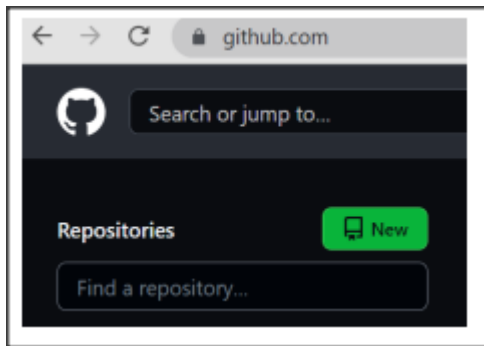
Repositório GitHub



Criando um repositório.

O **repositório** é onde vamos **organizar e armazenar** o nosso **projeto**. Entenda cada repositório como uma gaveta, e nela vamos armazenar os arquivos relacionados com o projeto. Algo muito parecido quando organizamos a gaveta de meias, camisas e calças.

Repositório Remoto



É possível escolher entre:

- Nome do Repositório que será criado
- Owner do projeto
- Compartilhamento Public e Private
- Adição de arquivo README informativo
- Adição de arquivo .gitignore com arquivos ou diretórios a serem ignorados no Commit/Push do projeto


Criando Repositório Remoto

Create a new repository

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

Owner *

Repository name *

 Esquiro ▾


/

Teste_IOS ✓


Great repository names are short and memorable. Need inspiration? How about [congenial-invention?](#)

Description (optional)

Exemplo usado para o tutorial

 **Public**

Anyone on the internet can see this repository. You choose who can commit.

 **Private**

You choose who can see and commit to this repository.

Initialize this repository with:

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

☐ **Add a README file**

This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**

Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**

A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

Vinculando Branch Remota



Comando para associar o repositório local ao repositório remoto:
git remote add origin <https://github.com/CaminhoDoProjeto.git>
OBS: Informar o link do GitHub do projeto

IOS – Instituto de
Oportunidade Social

O que é o Git ?



O que é o Git ?

O **Git** é um sistema **open-source** de controle de versões gratuito e distribuído projetado para trabalhar com diversos tipos de projetos. Além disso, ele fornece ao desenvolvedor uma maneira muito mais inteligente e eficaz de **organizar seu projeto**. O Git foi inicialmente projetado e desenvolvido por **Linus Torvalds**, em **2005**, para o desenvolvimento do **kernel Linux**, mas foi adotado por muitos outros projetos. Alguns dos objetivos do Git são: velocidade, design simples, forte suporte para o **desenvolvimento não linear** (permitir muitas ramificações em paralelo), totalmente distribuído, capaz de lidar com grandes projetos eficientemente como, por exemplo, o kernel do Linux.



O que é o Git ?

Como foi dito, o Git é um **sistema de controle de versão distribuído** e ele pode ser usado como um servidor pronto para uso. Os **servidores dedicados** de Git incluem funcionalidades extras, tais como:

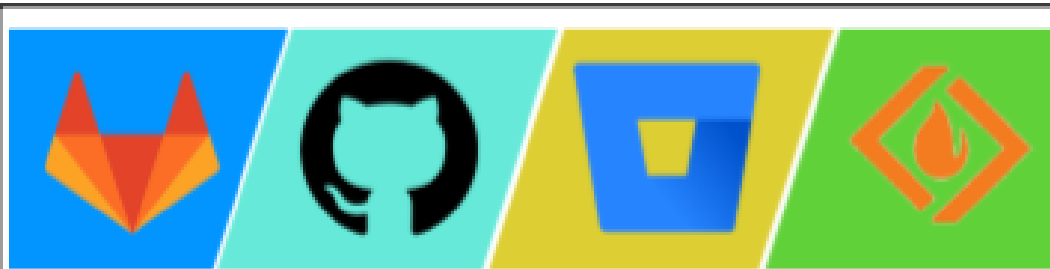
- Controle de acesso;
- Revisão por pares;
- Ferramentas de visualização;
- Repositório, etc.



Repositório é o lugar onde os códigos ficam localizados e eles podem ser públicos ou privados. Entenda nesse momento: um **sistema repositório é diferente de um sistema versionador**.

O que é o Git ?

Os **serviços de Git** são **plataformas de hospedagem de código-fonte** e permitem que os desenvolvedores contribuam em projetos **privados** ou **abertos** (mais conhecidos como projetos **open source**). Cada um pode ter funcionalidades ou recursos específicos e diversos serviços de Git possuem repositórios fechados, em que seus colaboradores ou funcionários de uma determinada empresa desenvolvam seus projetos com ferramentas de gerenciamento organizacional. Os serviços de Git mais populares são **GitHub**, **GitLab**, **Gitea**, **Bitbucket** e **SourceForce**.



Símbolos dos serviços de Git, na ordem a esquerda para a direita: GitLab, GitHub, Bitbucket e SourceForce.

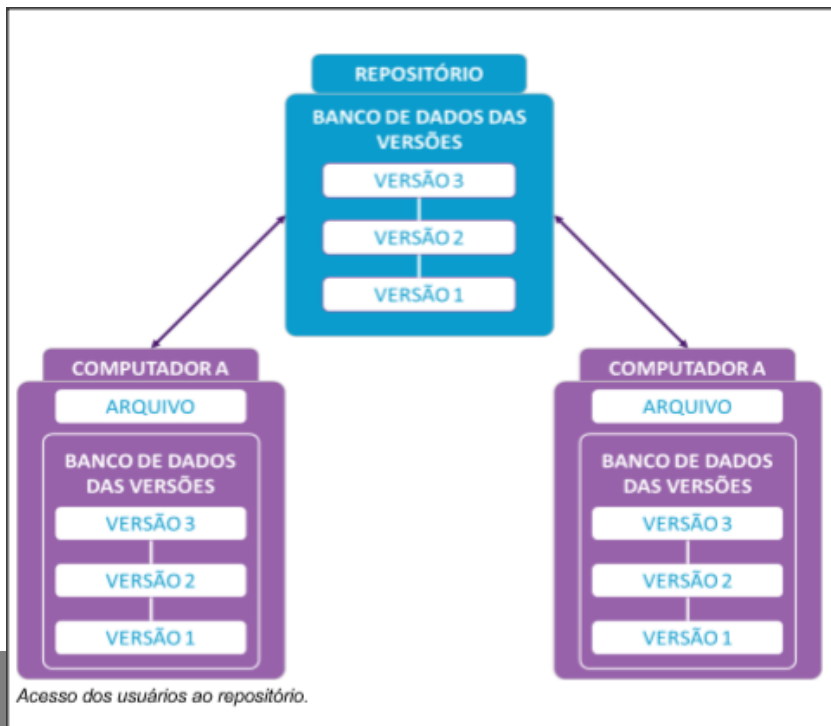
IOS – Instituto de
Oportunidade Social

Funcionamento do Git



Funcionamento do Git

Os servidores, como o **GitHub**, são **repositórios remotos**, que podem ser acessados por diversos usuários. Esses usuários podem fazer o **download** de uma **cópia do repositório** em seu computador de trabalho, realizar alterações no código localmente e depois submeter a atualização para o repositório.



Assim o **arquivo** no **computador local**, que está sendo atualizado, pode assumir três estados: modificado, staged e committed.

O estado **Modificado** (apenas alterou) significa que você alterou o arquivo, mas não deu o commit para o seu banco de dados local.

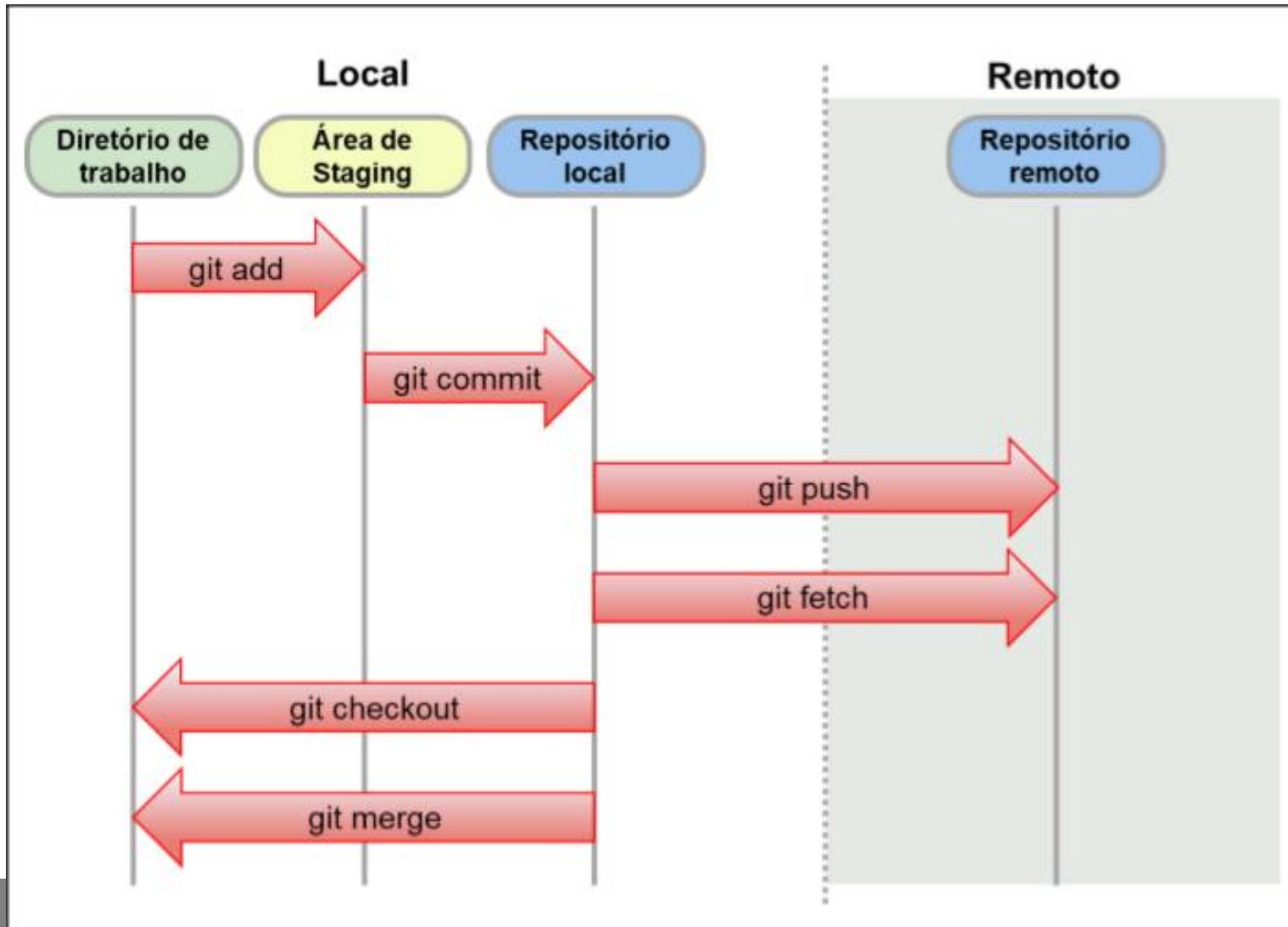
O estado **Stage** (git add) significa que você escolheu quais alterações do arquivo modificado na sua versão atual deseja enviar ao **commit**.

O estado **Committed** (git commit) significa, que o arquivo modificado está salvo no repositório local.

Então, para garantir o desenvolvimento do projeto de uma maneira correta e segura, o Git tem uma estrutura basicamente dividida em dois ambientes:

- **Remoto:** que é composto pelo Repositório remoto.
- **Local:** que é composto pelo Repositório local, pela Área de **Staging** e pelo **Diretório de trabalho**.

Funcionamento do Git



IOS – Instituto de
Oportunidade Social

Exercícios



- Efetuar o cadastro no Portal GitHub
- Criar Repositório Público no GitHub para armazenar os exercícios realizados nas aulas