xmobots

Wecomp

V Semana da Computação | IFSP

Dominando o Fluxo:
Git e Git Flow Descomplicados

Marco Pereira | 03/09/2024

xmobots

Marco Pereira

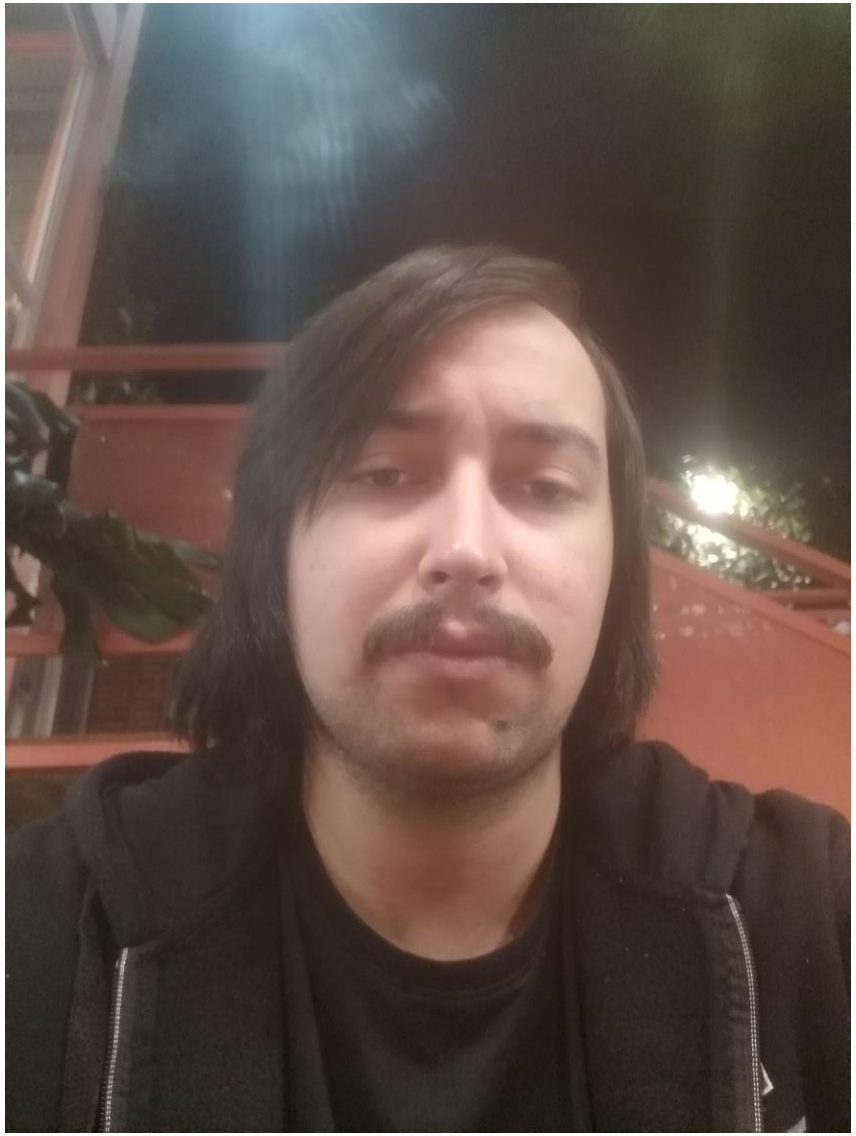
Desenvolvedor P&D na XMobots desde 2021

Universidade Paulista
Bacharelado · Computer Science
2019 - 2022

Senai São Paulo
Técnico em Mecatrônica
2019 - 2020

Senai São Paulo
Técnico de Eeletroeletrônica
2019 - 2019

Senai São Paulo
CAI Eeletricista de Manutenção Eeletroeletrônica
2017 - 2018



xmobots

Agenda

Parte I

- Revisão dos conceitos gerais do Git:
  - O que é versionamento;
  - O que é o Git;
  - Diferença entre Git e GitHub/GitLab;
  - O que é um repositório;
    - Tipos de repositório.
  - Branches;
  - Add e stage;
  - Commit;
  - Push e pull;
  - Clone;
  - Merge;
  - Tags.

xmobots

Agenda

Parte II

- Git Flow:
  - O que é o Git Flow?
  - Por que ele foi criado?
  - Quais são os benefícios no seu uso?
  - Comparação entre o Git Flow e outros fluxos de trabalho;
    - GitHub Flow;
    - GitLab Flow.
  - Estrutura do Git Flow (branches);
  - Comandos do Git Flow;
  - Integração CI/CD, pipelines e hooks;
  - Atividade prática;
    - Criar uma feature, release e hotfix e integrá-las a um repositório remoto.

Git

Parte I

xmobots

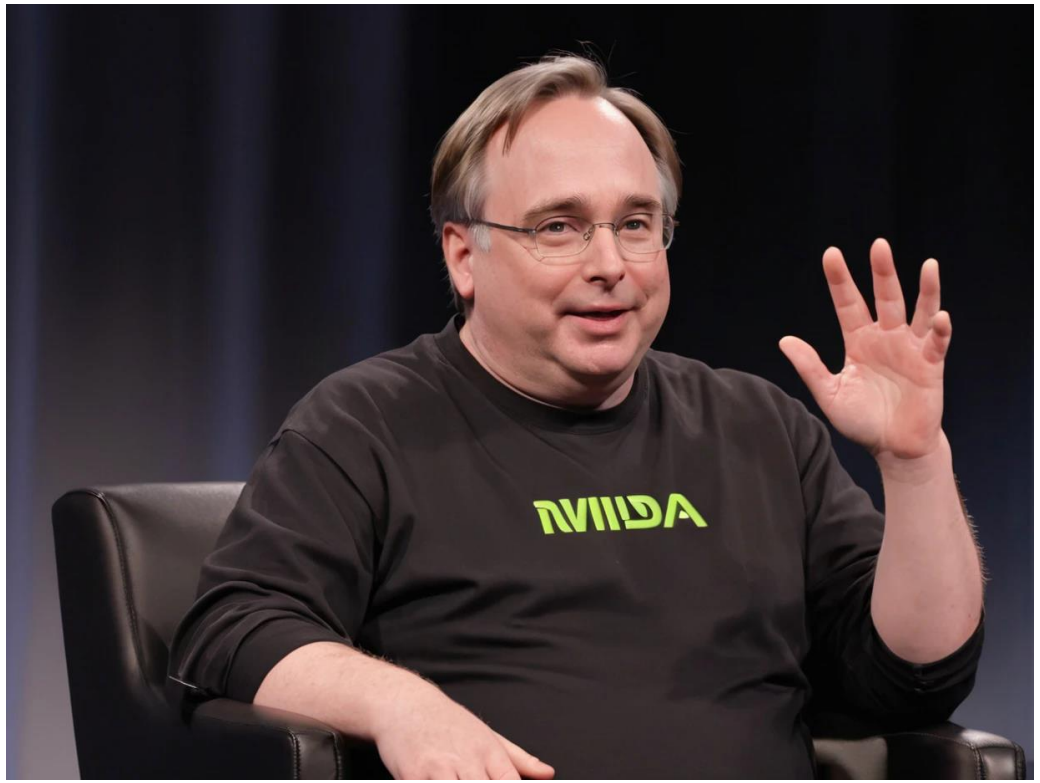
Versionamento

O que é?

- Gerenciamento de diferentes versões de um software (código-fonte), sistema ou modelo.

Git

O que é?



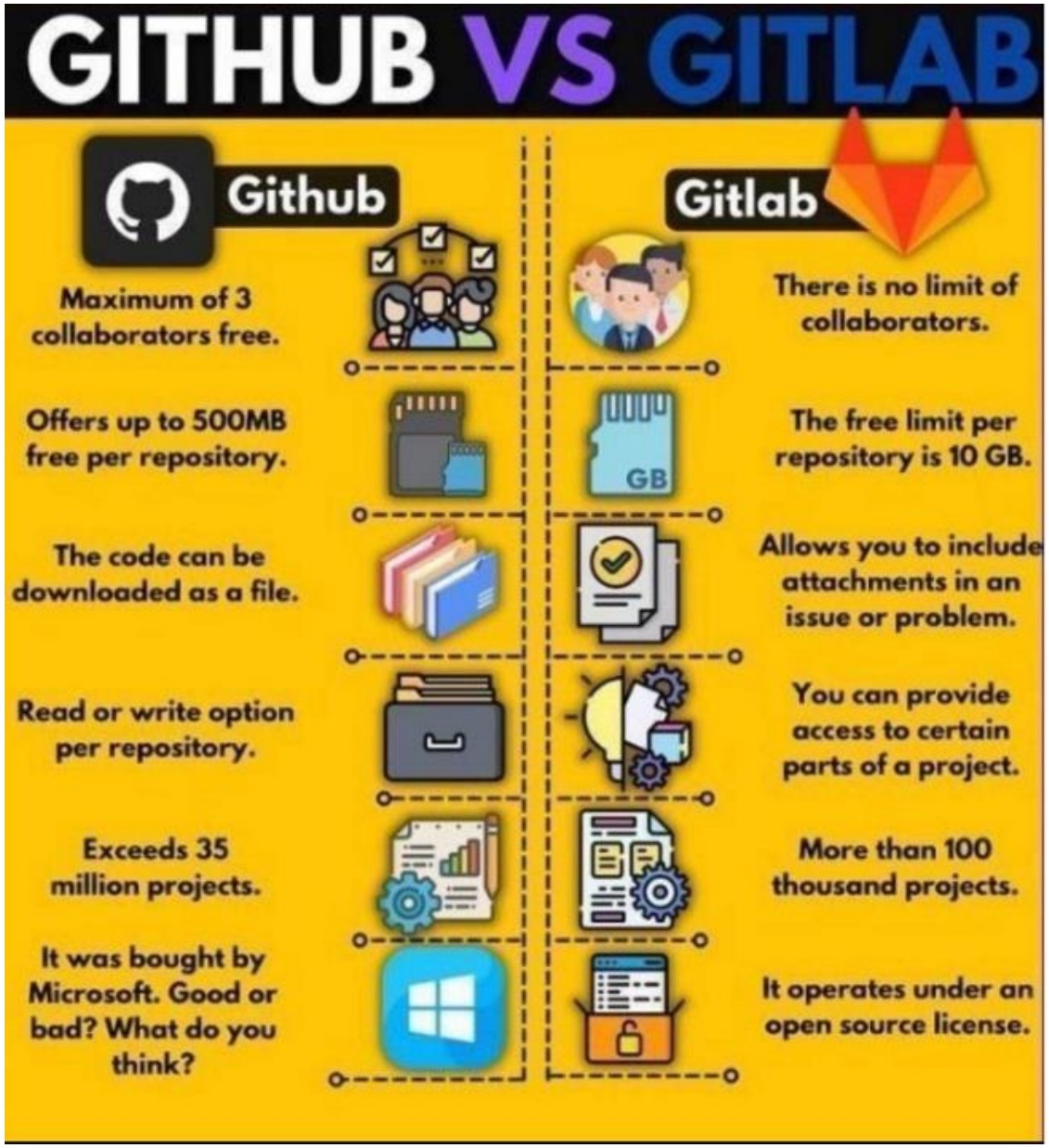
- Software utilizado para gerenciamento das versões de um ou mais arquivos;
- Permite manter um histórico de todas as alterações feitas em um projeto;
- Permite que vários desenvolvedores trabalhem em um mesmo projeto simultaneamente;
- Sistema distribuído;
- Criado em 2005 por Linus Torvalds para ser usado no desenvolvimento do kernel Linux.

Plataformas

GitHub e GitLab

- São servidores que utilizam o Git como sistema de controle de versão;

- São plataformas para hospedagem do código-fonte.



xmobots

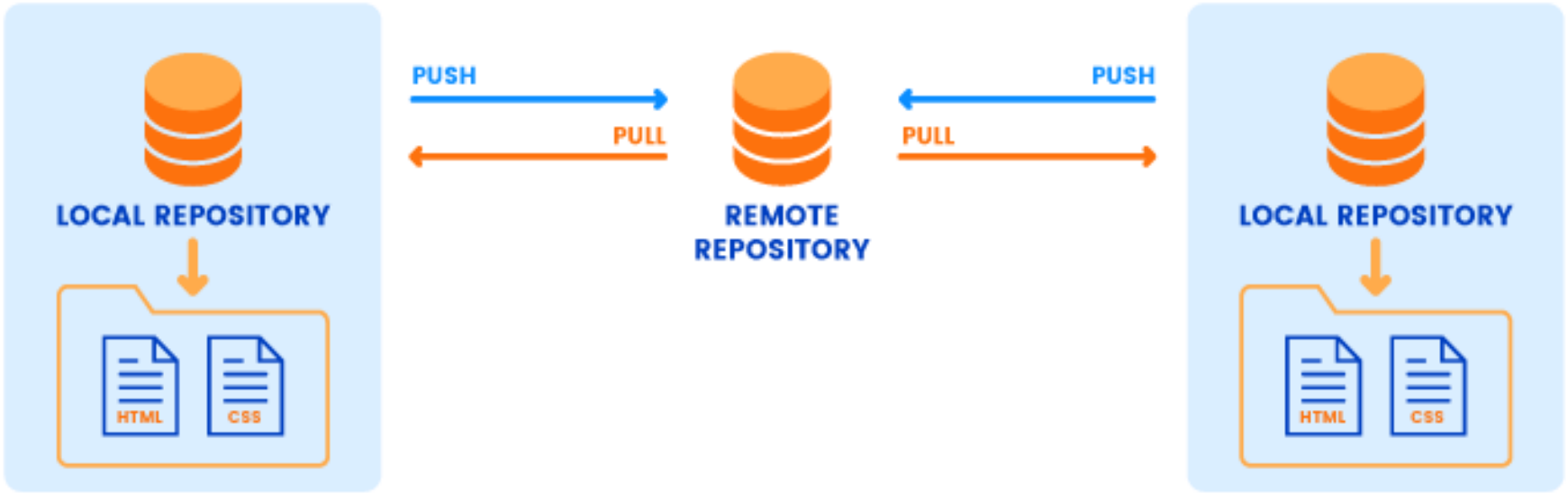
Repositórios

O que são e quais os tipos

- Diretório ou espaço de armazenamento onde o projeto é mantido:
  - Arquivos do projeto;
  - Histórico de revisões;
  - Configurações.

- Repositório local:
  - É uma cópia off-line do projeto.

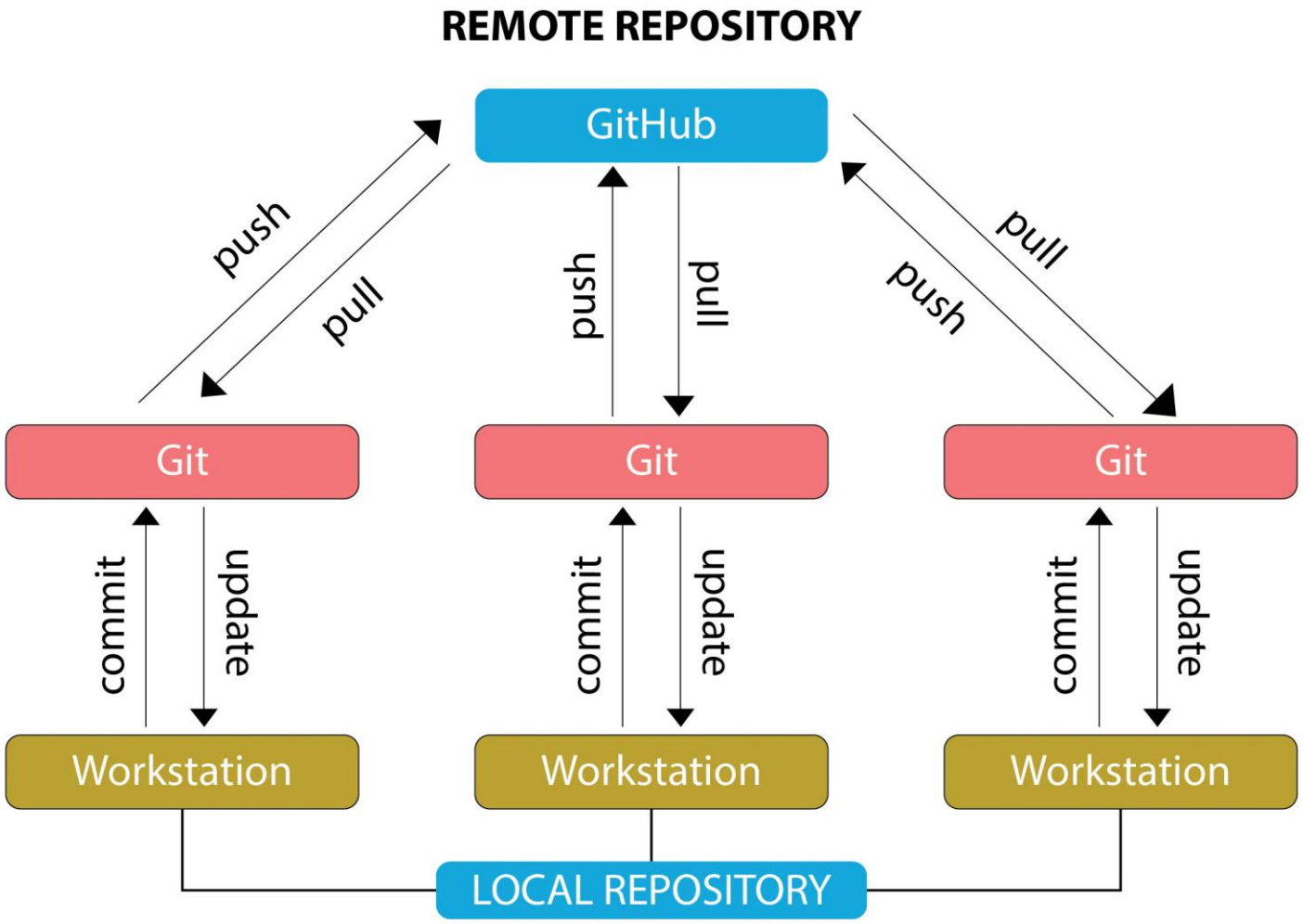
- Repositório remoto:
  - É uma versão do projeto armazenada em um servidor.



xmobots

Repositórios

O que são e quais os tipos

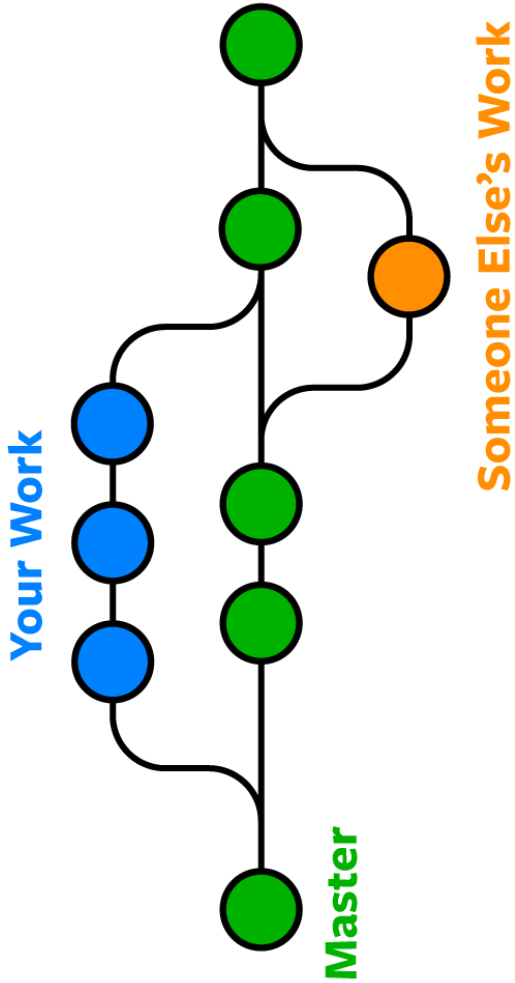


xmobots

Branches

O que são?

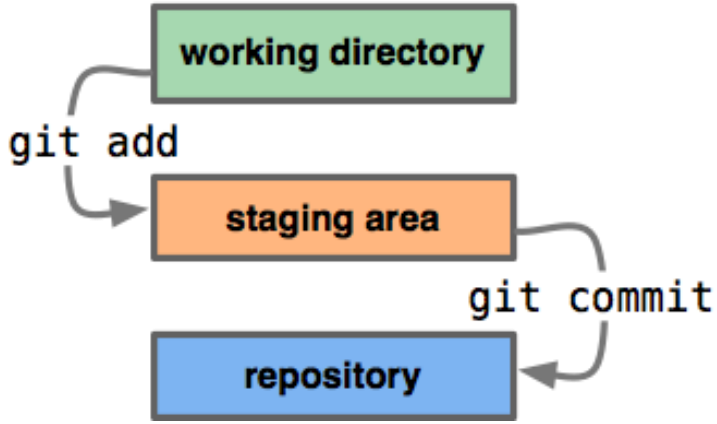
- Versões isoladas do projeto (ramificações);
- No Git Flow, elas são utilizadas para organizar o desenvolvimento;
- Master/main: onde o código mais estável e pronto para produção é mantido;
- Develop: onde uma versão em desenvolvimento é mantida;
- Feature: funcionalidade específica;
- Hotfix: correção de problemas críticos em produção.



xmobots

Staging

O que é?



- Após realizar alterações no repositório, podemos salvá-las;
- Staging é o processo de salvar arquivos ou partes de arquivos;
- Ao fazer stage de um arquivo, você está dizendo ao Git que deseja incluir as mudanças feitas naquele arquivo no próximo snapshot do repositório;
- O comando "add" é usado para adicionar mudanças ao "staging area" (área de preparação).

Commit

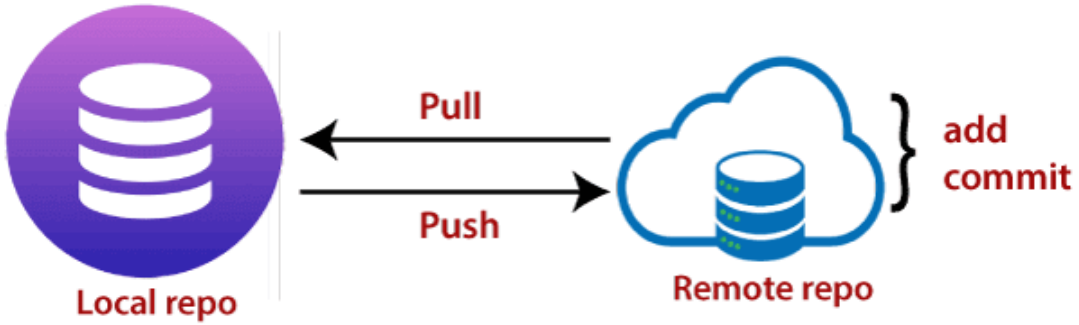
O que é?

- Após realizar alterações no repositório, podemos salvá-las;
- Salva efetivamente as mudanças que estão na área de preparação;
- Um commit no Git é como um "snapshot" ou uma foto do estado atual do projeto;
- É composto por:
  - Identificador único (hash);
  - Mensagem;
  - Metadados;
  - Referências.

Push/pull

O que são?

- São usados para sincronizar o repositório local com o remoto e vice-versa;
- Push: envia os commits do repositório local para o remoto;
- Pull: puxa os commits do repositório remoto para o local.

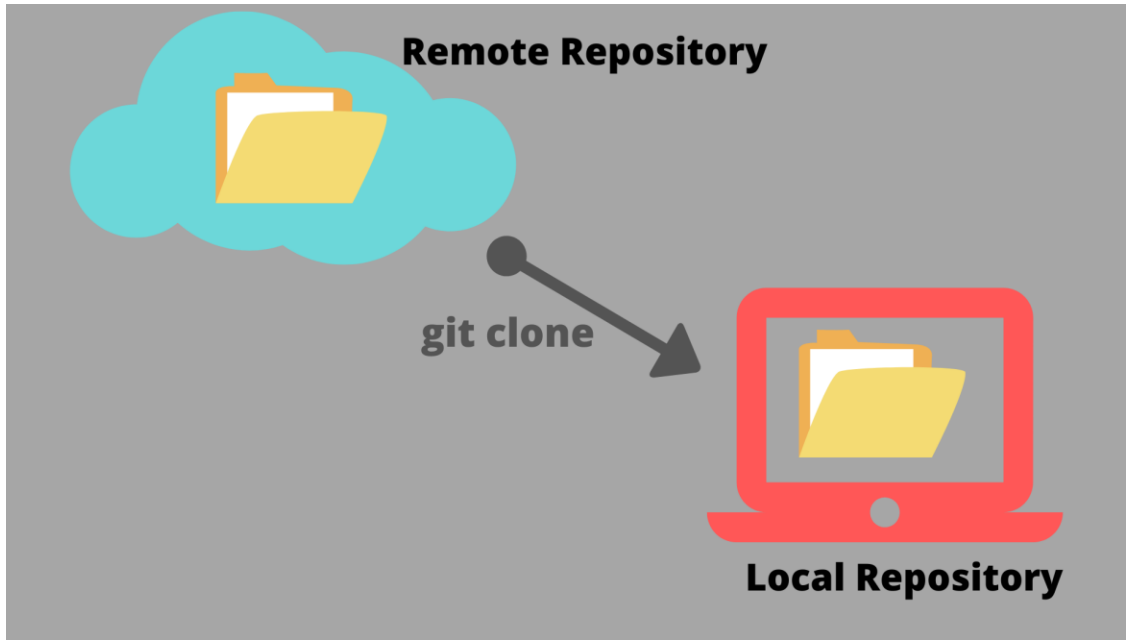


xmobots

Clone

O que é?

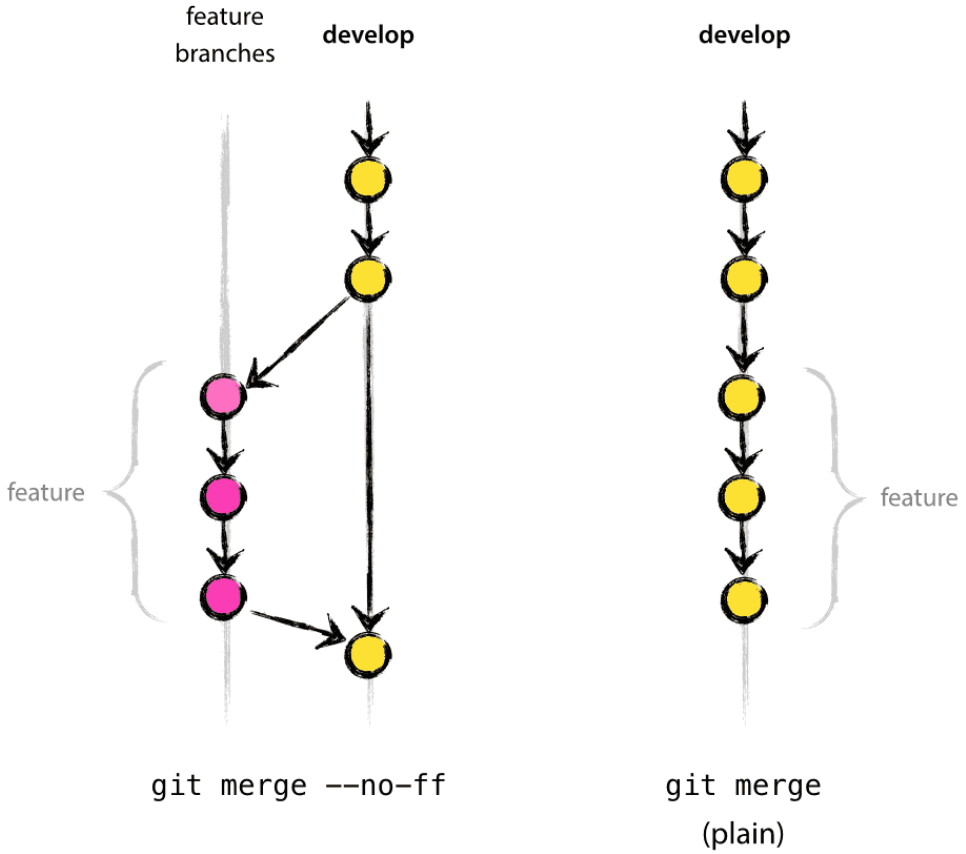
- Cria uma cópia de um repositório remoto na máquina;
- O repositório criado é local;
- A cópia inclui:
  - Todos os arquivos do repositório original;
  - Histórico de commits;
  - Branches;
  - Tags.



Merge

O que é?

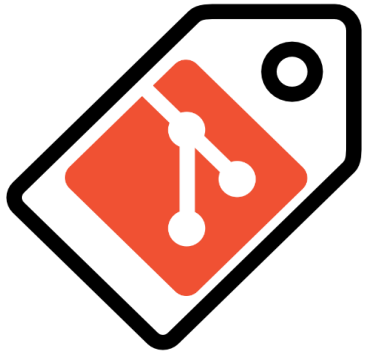
- Combina (mescla) mudanças de diferentes branches em uma única branch;
- Fast-forward: não cria um commit, apenas avança o ponteiro;
- Conflito: ocorre ao alterar as mesmas linhas de um arquivo:
  - Git não consegue mesclar automaticamente as mudanças.



xmobots

Tags

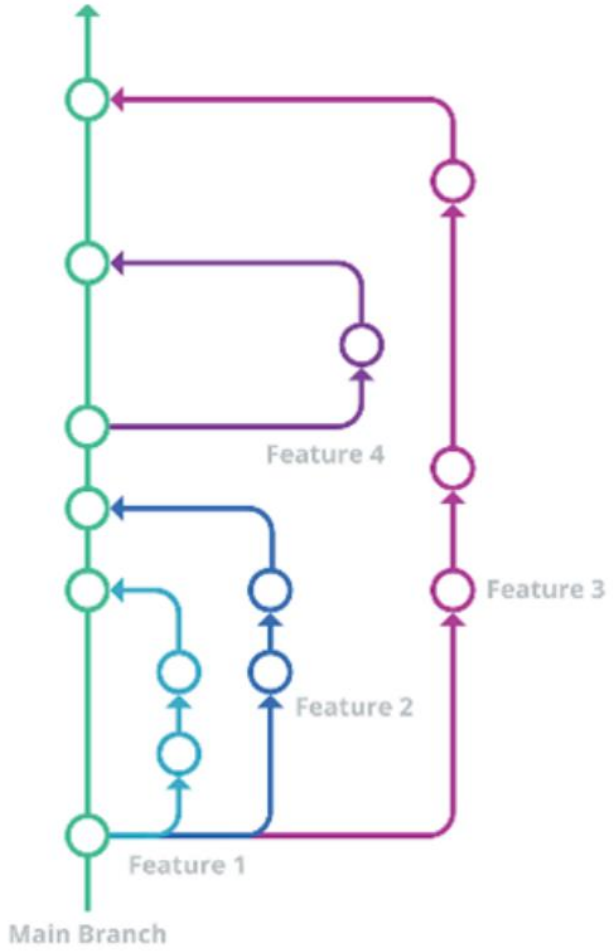
O que são?



- São referências que apontam para commits específicos;
- São usadas para marcar pontos importantes no histórico de um projeto;
- São marcadores ou etiquetas que permitem identificar de forma rápida e permanente uma versão específica do código;
- Geralmente, marcam as releases (versões) de um projeto.

Git Flow

Parte II



xmobots

Git Flow

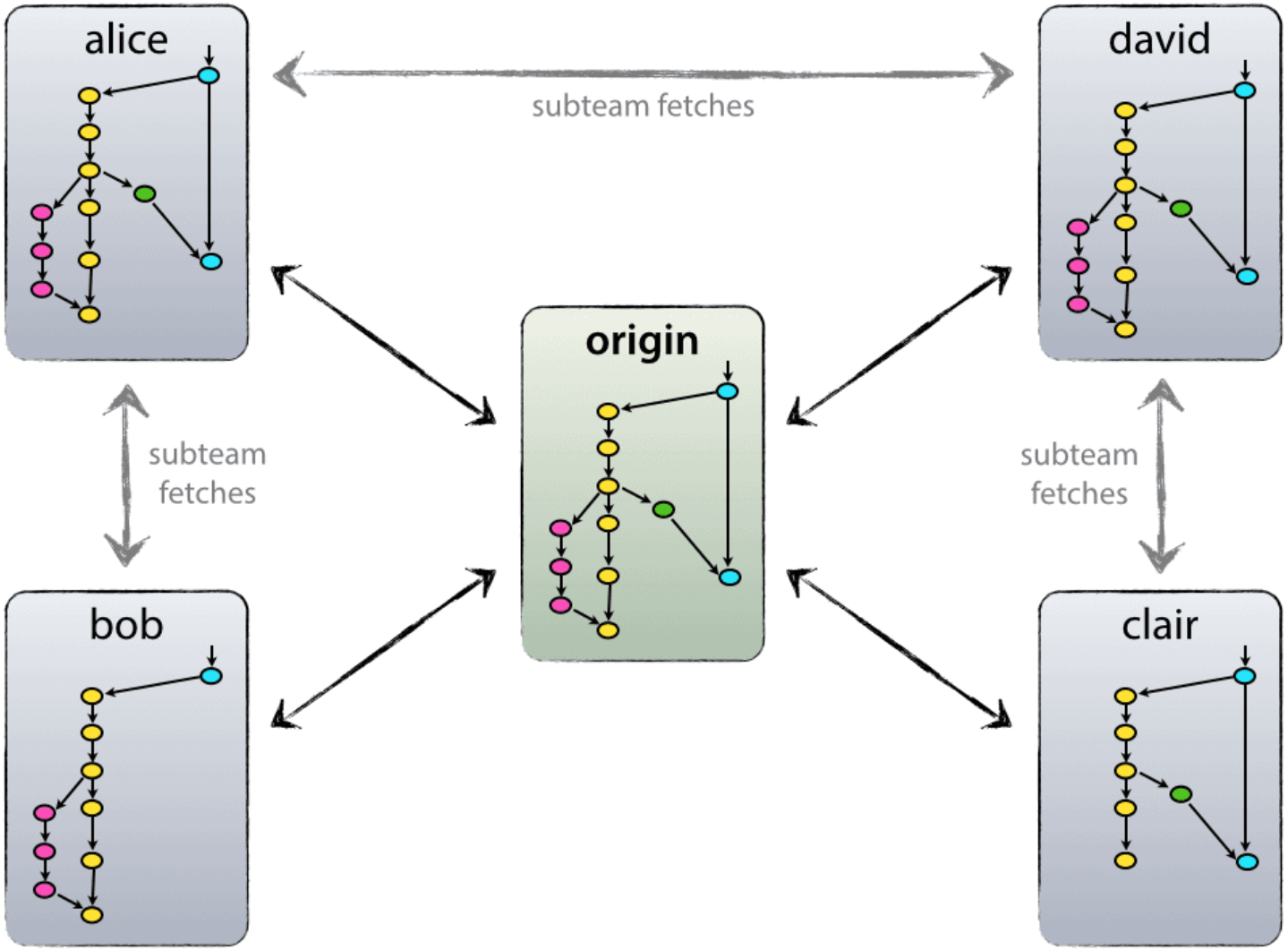
O que é?

- Como controlar e coordenar o trabalho paralelo de diversas pessoas no mesmo projeto?
- Fluxo de trabalho padronizado para o Git;
- Define como criar e gerenciar branches para facilitar o desenvolvimento, testes e lançamentos de versões;
- Conjunto de procedimentos que cada membro da equipe precisa seguir para chegar a um processo de desenvolvimento de software gerenciado;
- Foi criado em 2010 pelo engenheiro de software holandês, Vincent Driessen.

xmobots

Git Flow

O que é?



xmobots

Git Flow

Por que ele foi criado?

- Foi criado para fornecer uma estrutura organizada e eficiente para o desenvolvimento de software;
- A ideia é melhorar a fluidez do desenvolvimento e organizar as branches;
- Sem ele, problemas de coordenação, integração e gerenciamento de versões podem ocorrer;
- Fluxo de trabalho mais sistemático e eficiente para desenvolvimento de software.

Git Flow

Benefícios

- Proporciona uma estrutura bem definida para branches;
- Permite que múltiplas funcionalidades e correções sejam desenvolvidas simultaneamente sem interferir umas nas outras;
- Proporciona um processo claro para preparar e lançar novas versões do software;
- Proporciona um modelo de trabalho compartilhado que facilita a colaboração entre membros da equipe.

Git Flow

Github Flow e GitLab Flow

GitHub FLow

- Abordagem simples e minimalista;
- Ciclo de desenvolvimento baseado em CI/CD;
- Adequado para projetos com ciclos de desenvolvimento rápidos;
- Apenas uma branch principal e features.

GitLab FLow

- GitHub Flow mais flexível;
- Permite criar branches adicionais para ambientes específicos;
- Adequado para projetos de diversos tamanhos e complexidades.

Git Flow

Estrutura

- Branches principais:
  - **Master**: código de produção;
    - Criada junta com o repositório;
    - Por definição, cada commit na master:
      - É uma release;
      - Deve ter uma tag de versão.
    - Convenção de nome: `master`.
  - **Develop**: integração de novas funcionalidades.
    - Deriva da master.
    - Convenção de nome: `develop`.

Git Flow

Estrutura

- Branches auxiliares:
  - **Feature**: novas funcionalidades de uma versão;
    - Deriva da develop;
    - Merge na develop;
      - Excluída após isso.
    - Convenção de nome: qualquer coisa exceto `master`, `develop`, `release-*` ou `hotfix-*`.

Git Flow

Estrutura

- Branches auxiliares:
  - **Release**: ajustes finais antes do lançamento;
    - Deriva da develop;
    - Merge na develop e master;
      - Excluída após isso.
    - É como um ambiente de homologação;
    - Define o número da próxima versão;
    - Correção de pequenos bugs;
    - Convenção de nome: `release-x.y.z`.

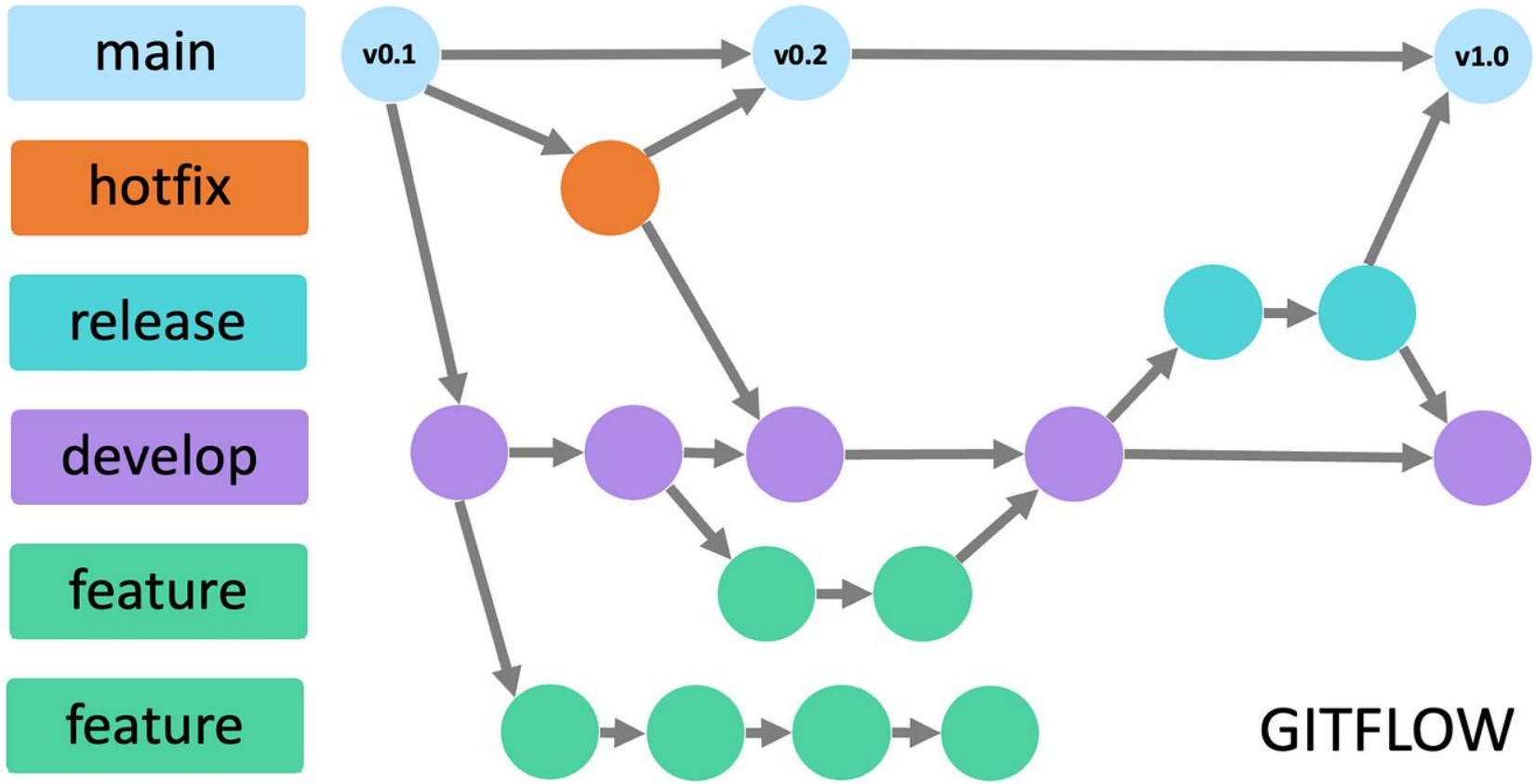
Git Flow

Estrutura

- Branches auxiliares:
  - **Hotfix**: correção de bugs críticos em produção.
    - Deriva da master;
    - Merge na master e develop (talvez na release também);
      - Excluída após isso.
    - Semelhante à branch de release;
    - Convenção de nome: `hotfix-x.y.z`.

Git Flow

Estrutura



xmobots

Git Flow

Comandos

- `git flow`: facilita o uso de uma estratégia de gerenciamento do fluxo de trabalho.
  - `init`: inicializa o repositório com a estrutura de branches do Git Flow;
  - `feature`: cria/finaliza uma branch de desenvolvimento;
  - `release`: cria/finaliza uma branch de lançamento;
  - `hotfix`: cria/finaliza uma branch de correção rápida.

Git Flow

CI/CD, pipelines e hooks

- CI (Continuous Integration ou Integração Contínua): alterações de código são integradas e testadas continuamente após um push/merge;

- CD (Continuous Delivery ou Entrega Contínua): deploy automático no ambiente de homologação ou produção após um push/merge;

- Pipeline: sequência de etapas automáticas que são executadas para realizar tarefas como build, teste e deploy do código;

- Hooks: scripts que o Git executa automaticamente em determinados pontos do fluxo de trabalho do Git.

Git Flow

CI/CD, pipelines e hooks

- Por conta das branches eternas, o Git Flow não se adapta muito bem ao CI/CD;
- Git Flow é adequeado para lançamentos mais demorados;
- O comando rebase não faz sentido com o Git Flow;
- CI/CD exige simplicidade e rapidez, com uma integração frequente de código e lançamentos rápidos:
  - Dificultado pelas múltiplas branches e fases separadas de desenvolvimento e lançamento do Git Flow.

Git e Git Flow

Referências

- GIT. About - Git. Disponível em: <https://git-scm.com/about>;

- DRIESSEN, V. A successful Git branching model. Disponível em: <https://nvie.com/posts/a-successful-git-branching-model/>;

- ATLASSIAN. Gitflow Workflow | Atlassian Git Tutorial. Disponível em: <https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>.

Git e Git Flow

Links

- PDF dos comandos do Git: https://drive.google.com/file/d/1k7RWrxWnlABsVskSLh4LaZQyRyS5D8g3/view?usp=sharing

xmobots | robotics for everyone everywhere

Muito obrigado!