Controle de versão Git

O que é um repositório Git?

- Um repositório do Git é um armazenamento virtual para projetos. Ele permite que você salve versões do código, que você pode acessar quando precisar.

git init

- Este comando pressupõe que você já tem uma pasta de projeto existente dentro da qual gostaria de criar o repositório. Você primeiro fará **cd** à pasta raiz do projeto e depois executará o comando git init.

```
$ git init
```

git clone

- Se um projeto já foi configurado em um repositório central, o comando clonar é a forma mais comum para os usuários obterem um clone de desenvolvimento local. A url pode ser com o protoclo SSH ou HTTPS.

$ git clone https://github.com/totemarcal/CRUDNode.git

git add

- O git addcomando adiciona uma mudança no diretório de trabalho à área de preparação. Ele diz ao Git que você deseja incluir atualizações para um arquivo específico no próximo commit.

  $ git add .

  $ git add arquivo.txt

git commit

- O comando captura um instante das mudanças atualmente testadas do projeto. O **git add** comando é usado para preparar mudanças no projeto que serão armazenadas em um commit.

$ git commit -m “comentário do commit”

git remote

- Se você utilizou **git init** para fazer um repositório atualizado, você não terá um repositório remoto para colocar mudanças. Você precisará adicionar uma url de repositório remoto à sua git config local

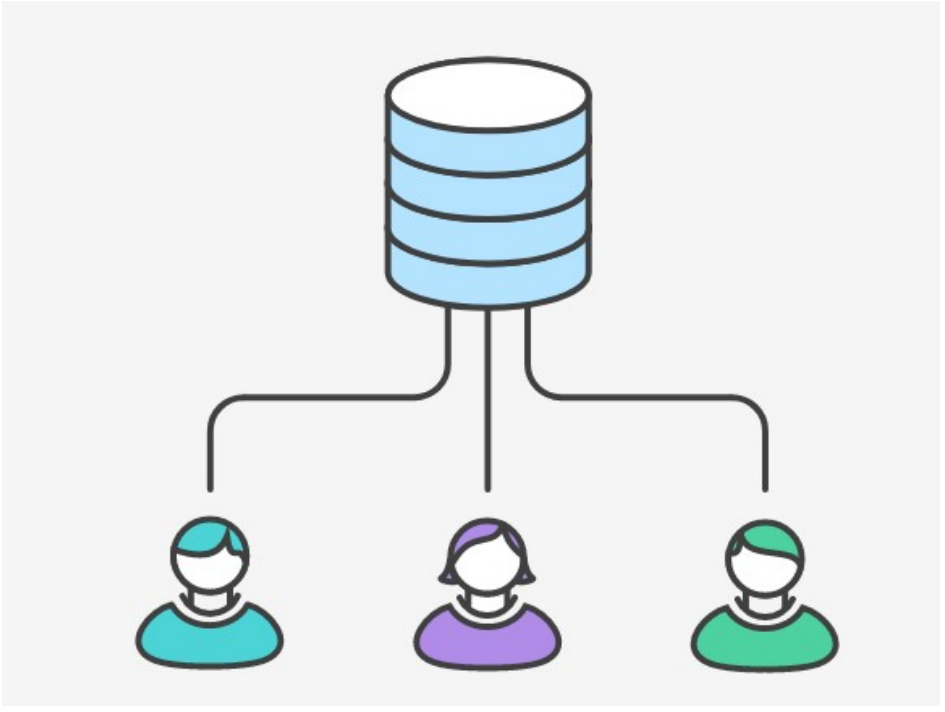
- $ git remote add origin https://github.com/totemarcal/CRUDNode.git

git push

- Este comando mapeará o repositório remoto em em uma referência em seu repositório local. Depois de ter mapeado o repositório remoto, você pode colocar ramificações locais nele.

- $ git push -u origin master

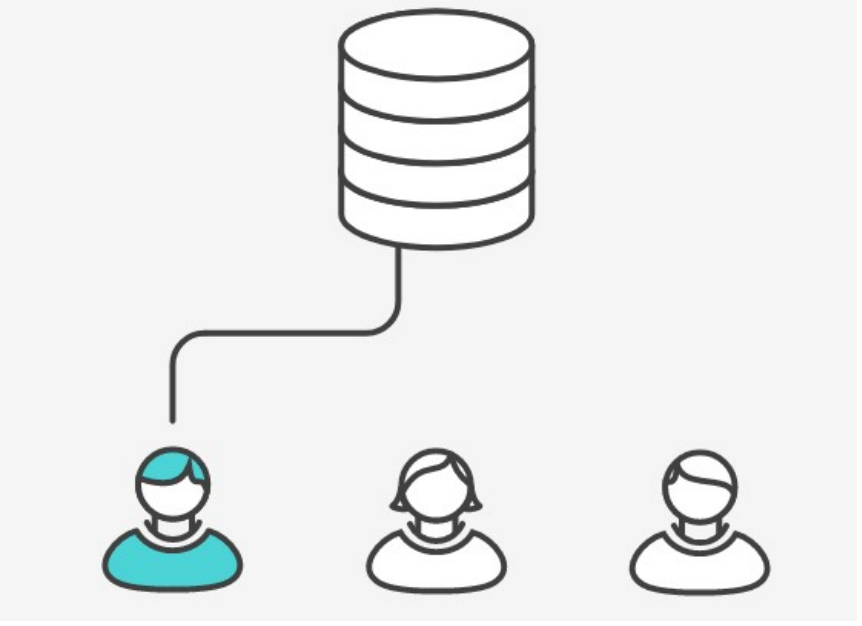
Fluxo de trabalho centralizado



John trabalha em seu recurso

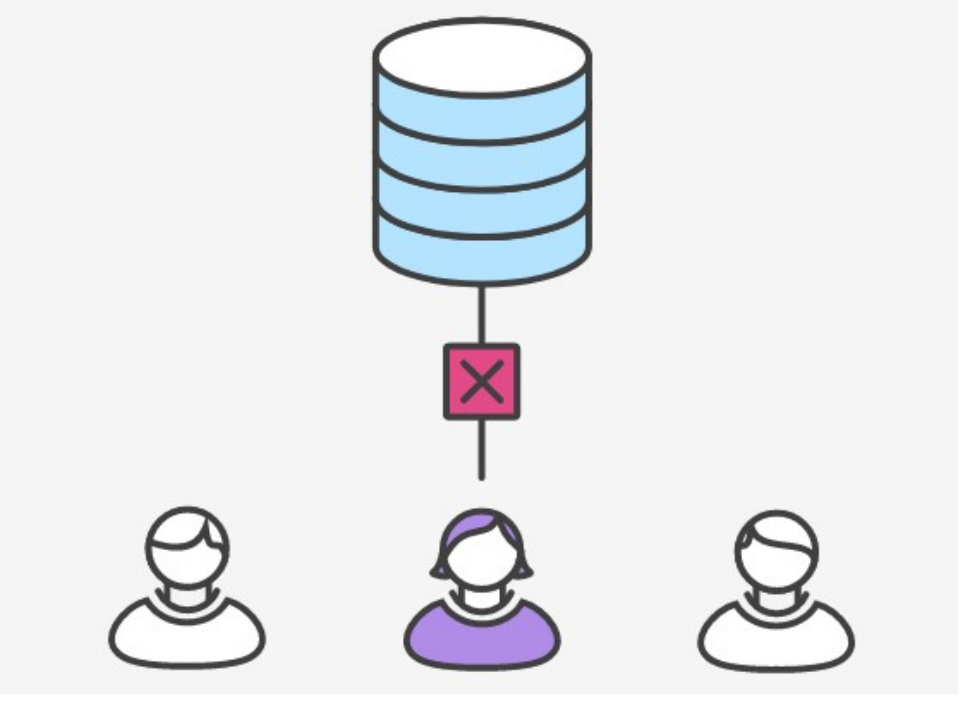
Mary trabalha em seu recurso

John publica seu recurso

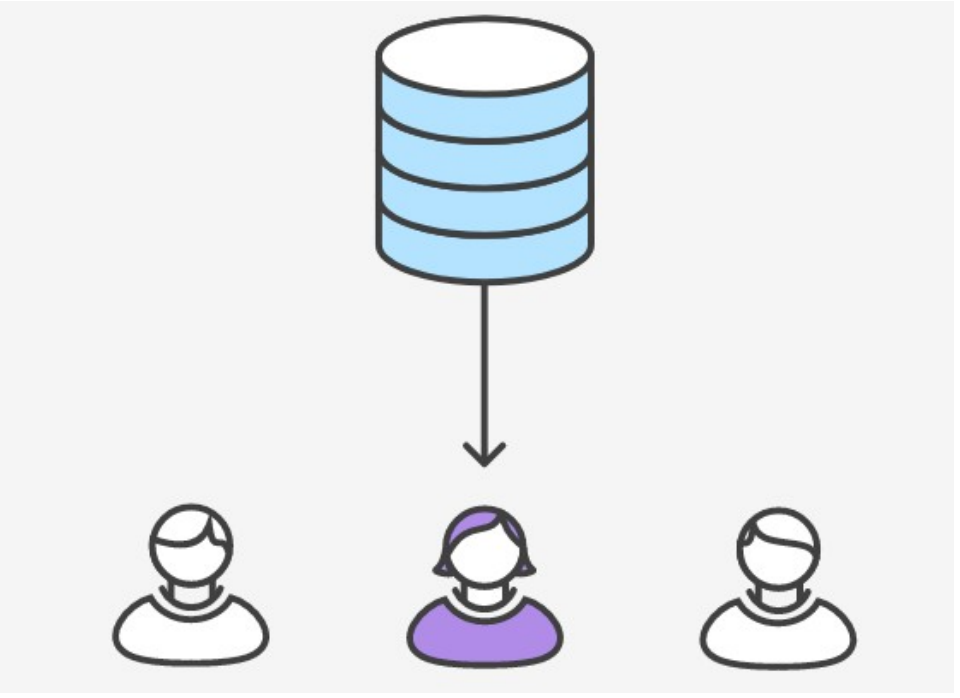


git push origin master

Mary tenta publicar seu recurso



Mary faz o rebase

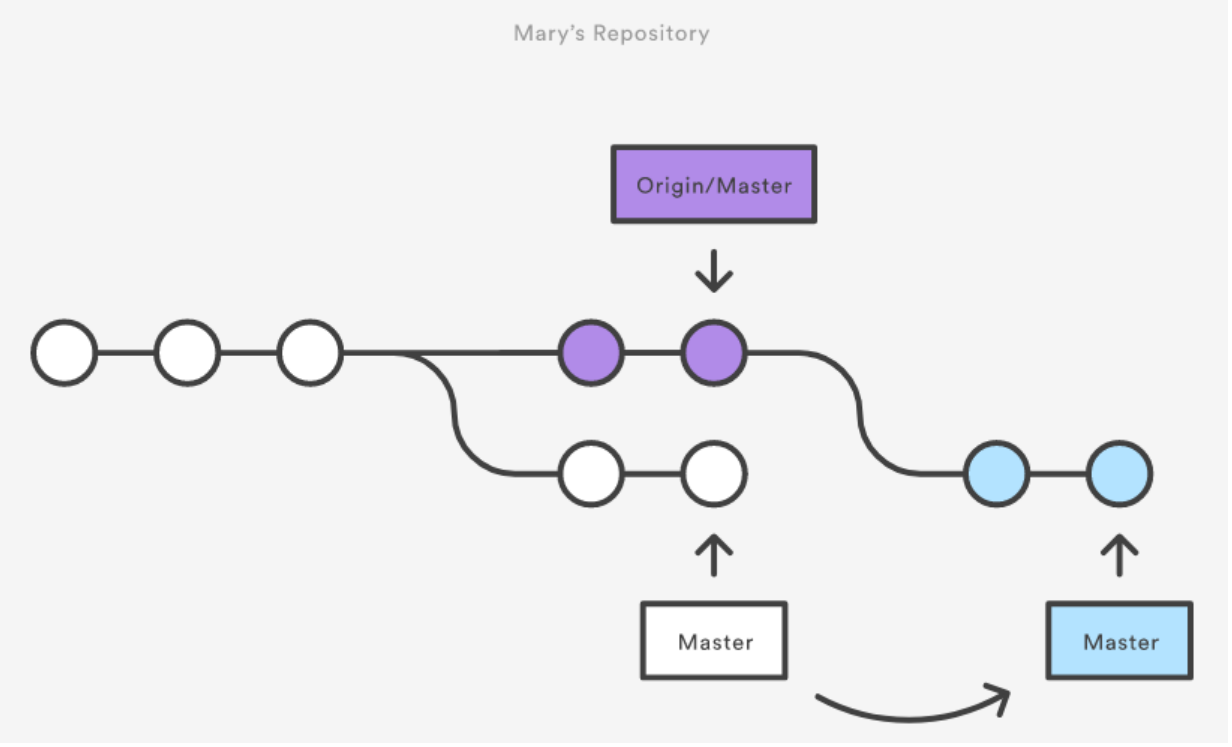


```
git pull --rebase origin master
```

git pull

- O git pull executa primeiro o **git fetch** que baixa o conteúdo do repositório remoto especificado. Em seguida, um **git merge** é executado para mesclar as referências de conteúdo remoto e direciona para um novo commit de mesclagem local.
- A opção **--rebase** diz ao Git para mover todas as confirmações para a ponta da ramificação master após sincronizar com as mudanças do repositório central

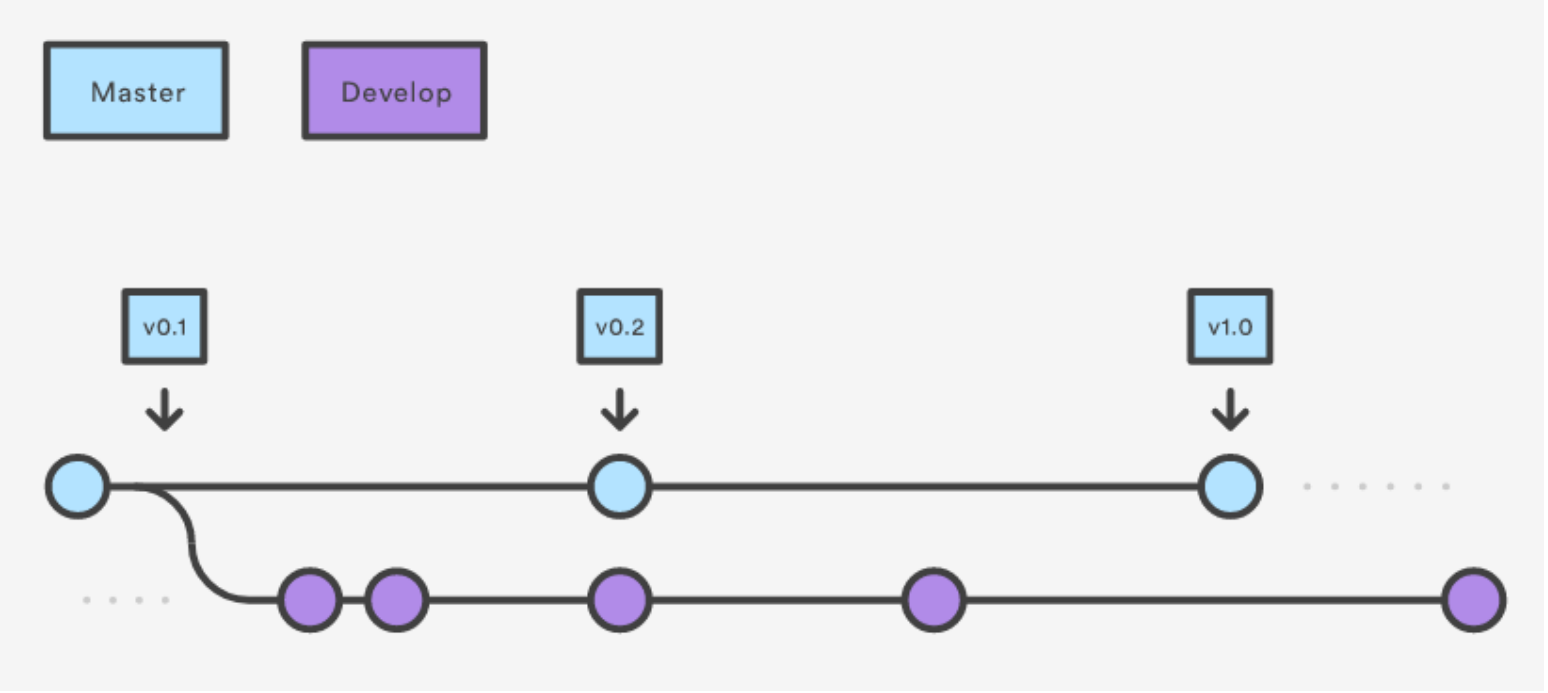
git pull



Fluxo de trabalho de Gitflow

- O Gitflow Workflow define um modelo de ramificação rigoroso projetado com base no lançamento do projeto. Isto oferece uma estrutura robusta para gerenciar projetos maiores.

Como funciona



git flow init

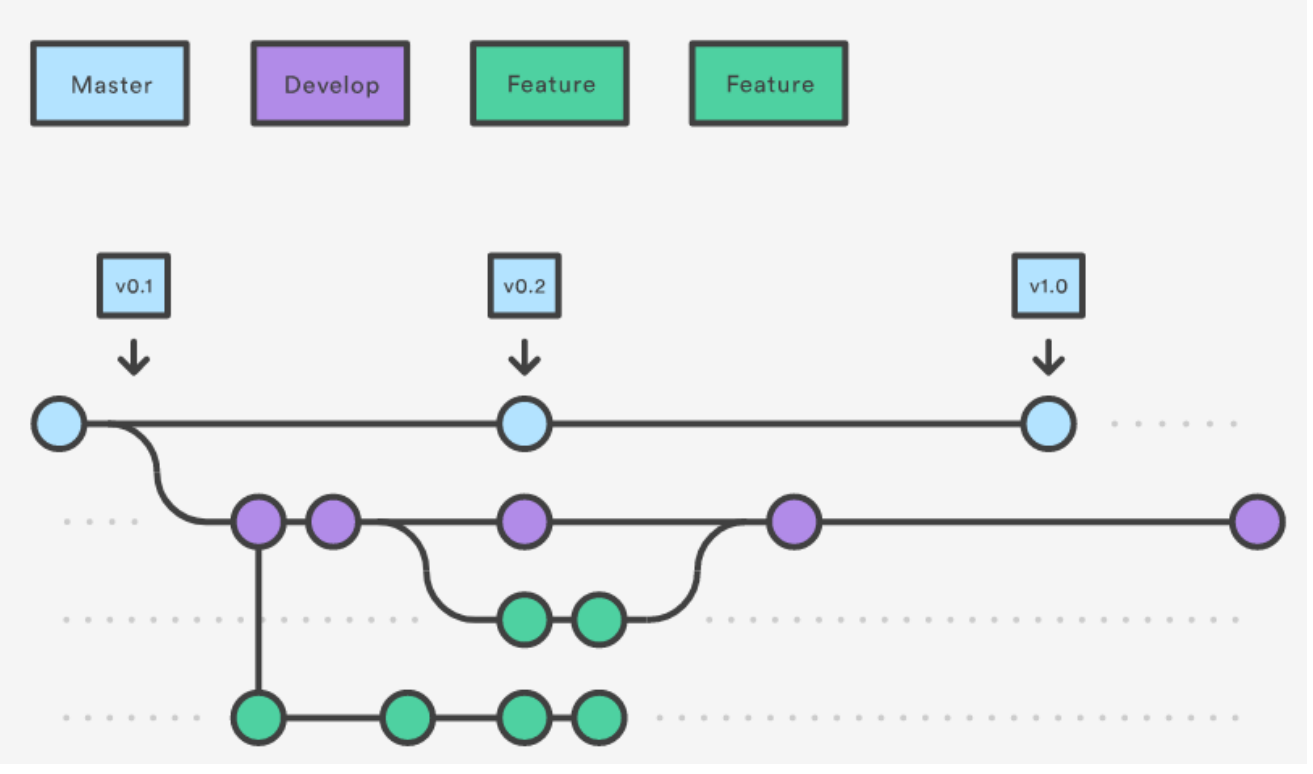
```
$ git flow init


Initialized empty Git repository in ~/project/.git/
No branches exist yet. Base branches must be created now.
Branch name for production releases: [master]
Branch name for "next release" development: [develop]


How to name your supporting branch prefixes?
Feature branches? [feature/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []


$ git branch
* develop
  master
```

Ramificações de feature



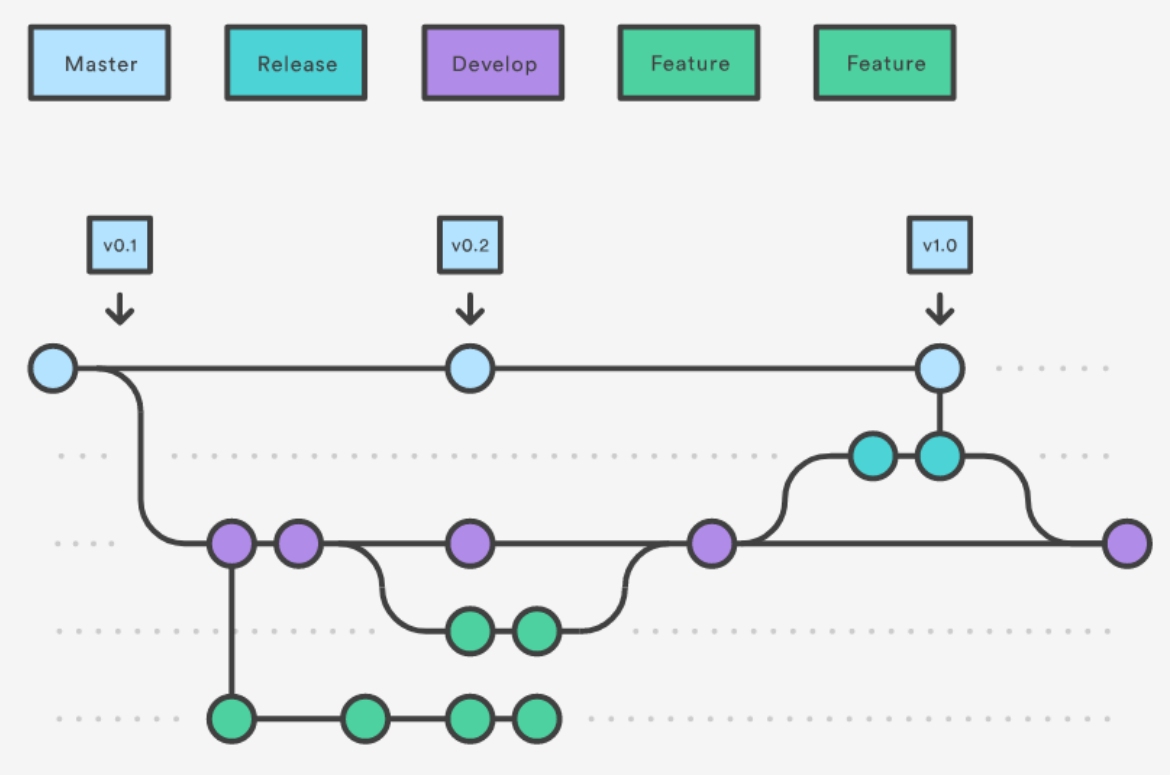
Ramificação de feature

- Criação da ramificação da featura

  $ git flow feature start feature_branch
- Finalização da ramificação de recurso

  $ git flow feature finish feature_branch

Ramificações de release



Ramificações de release

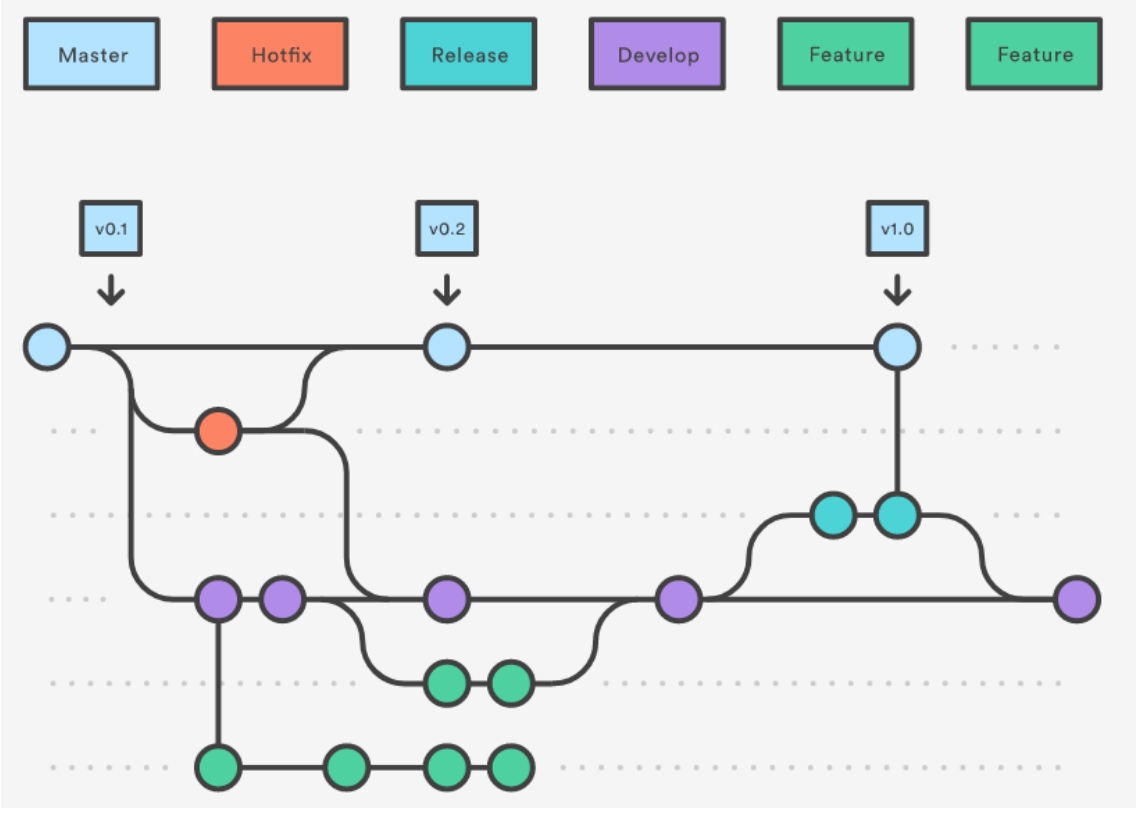
- Criação de ramificação de release

  $ git flow release start 0.1.0
- Merge da release com branch master e develop

  $ git merge release/0.1.0
- Finalização da ramificação de release

  $ git flow release finish '0.1.0'

Ramificações de hotfix



Ramificações de hotfix

- Criação de uma ramificação de hotfix

  $ git flow hotfix start hotfix_branch
- Finalização de uma ramificação de hotfix

  $ git flow hotfix finish hotfix_branch

Fluxo geral do Gitflow

1. Uma ramificação de desenvolvimento é criada a partir da branch principal
2. Uma ramificação de lançamento é criada a partir da ramificação de desenvolvimento
3. Ramificações de recurso são criadas a partir da ramificação de desenvolvimento
4. Quando um recurso é concluído, ele é mesclado na ramificação de desenvolvimento

Fluxo geral do Gitflow

5. Quando a ramificação de lançamento é concluída, ela é mesclada nas ramificações de desenvolvimento e principal
6. Caso um problema seja detectado na branch principal,uma ramificação de hotfix é criada a partir da principal
7. Após a conclusão da ramificação de hotfix, ela é mesclada para as ramificações de desenvolvimento e principal

Prática

1. Crie uma conta no Github
2. Crie um repositorio com o nome de de testeGit
3. Crie uma pasta chamada teste
4. Crie um arquivo chamada teste.txt
5. Inicialize o repositório na pasta teste
6. Associe o repositório local ao repositório do Github

Prática

7. Saia da pasta teste e crie a pasta teste2
8. Dentro da pasta teste2 clone o repositório do Github
9. Crie uma feature e commit uma alteração
10. Crie uma release e commit
11. Crie um hotfix e commit uma alteração