



Controle de versão, branches e merges

Disciplina: Ambiente de Desenvolvimento e Operações
Aula 02

Prof. Vanderson Bossi

vanderson.bossi@faculdadeimpacta.com.br

Referência desta aula

- GitHub. GitHub Guides.
- Disponível em <https://guides.github.com>

Controle de versão

- Gerenciamento do histórico de todas as modificações realizadas nos artefatos de um projeto.

Controle de versão

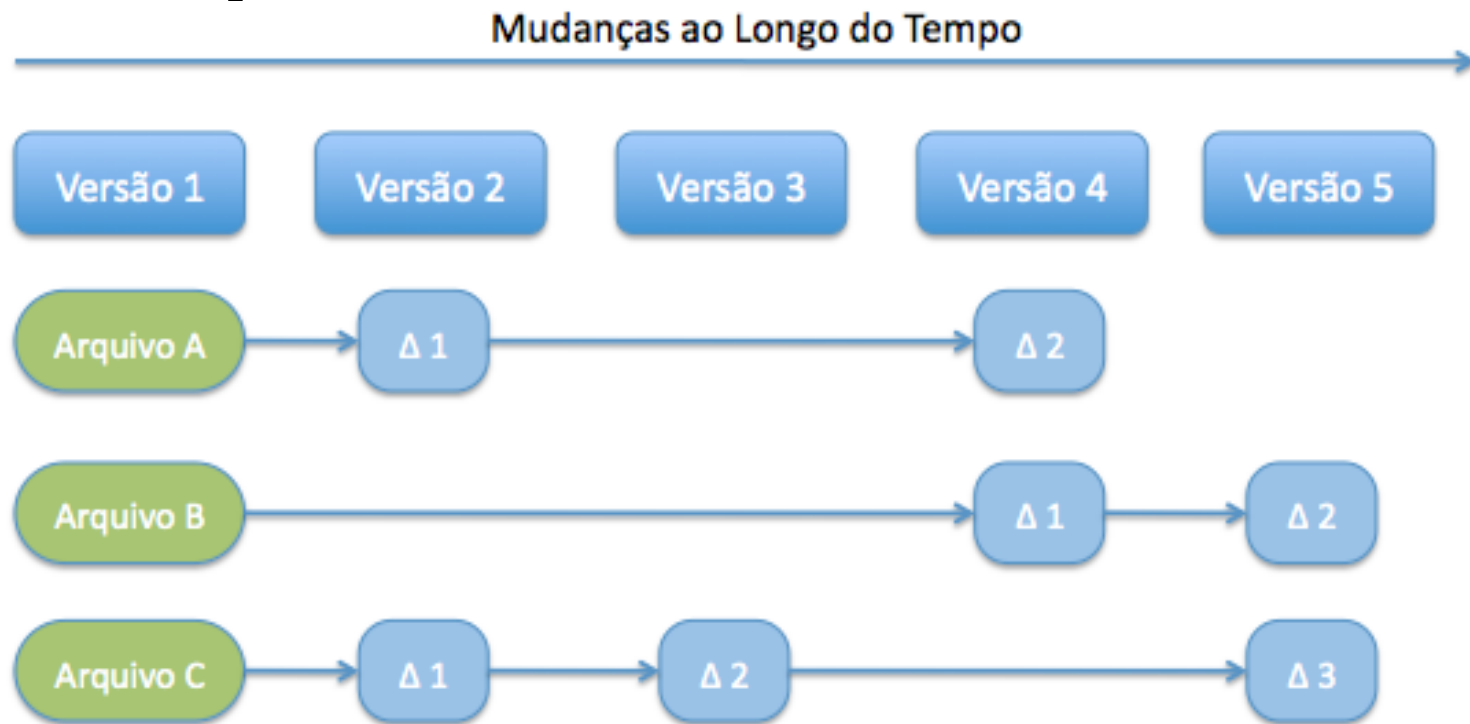
- Isto inclui:
 - Documentação
 - Script de testes
 - Casos de testes automatizados
 - Scripts de configuração e rede
 - Scripts de implantação
 - Criação do banco de dados
 - Atualizações e inicialização
 - Configuração de plataforma tecnologia
 - Bibliotecas
 - Ferramental
 - Documentação técnica

Controle de versão

- Significa que o conjunto deve ter algum identificador único, como o número da compilação ou o número do item de mudança no controle de versão.

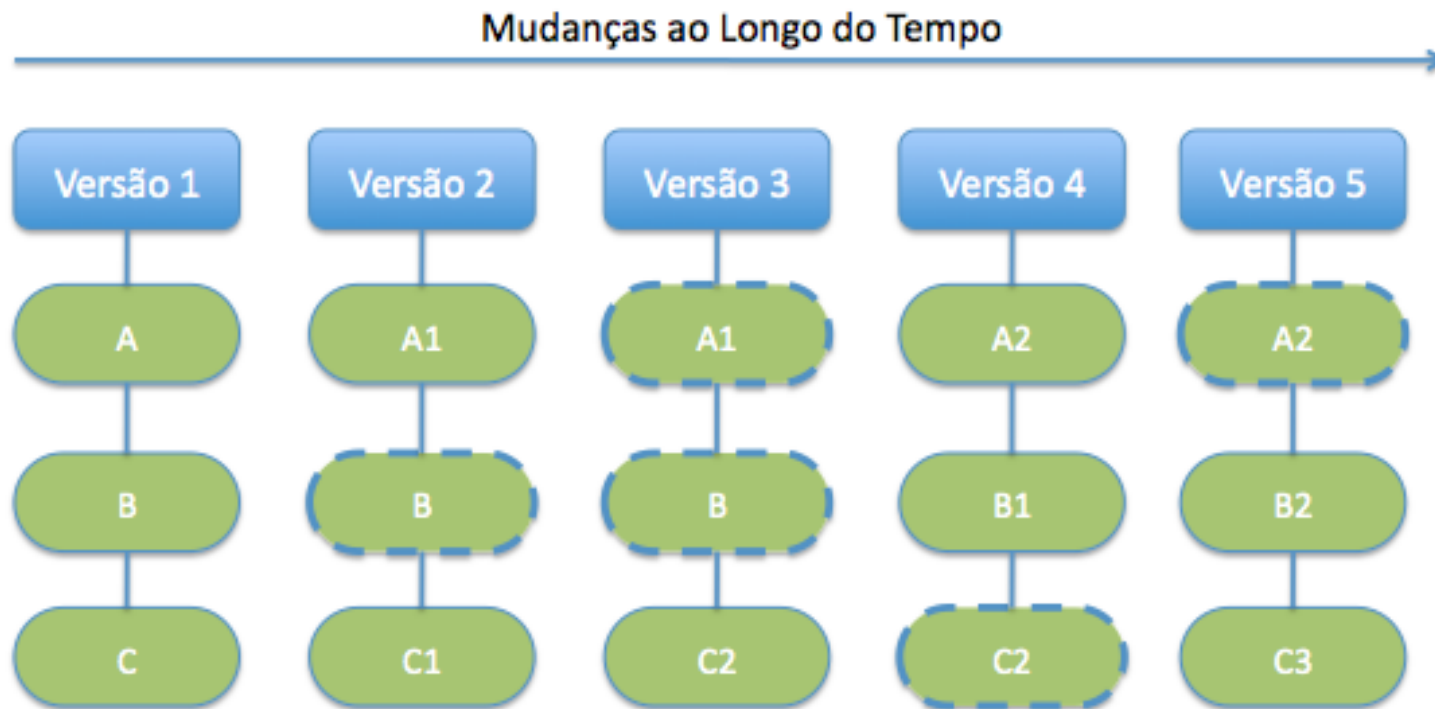
Controle de versão

- Sistema de versionamento que armazena as diferenças de versão



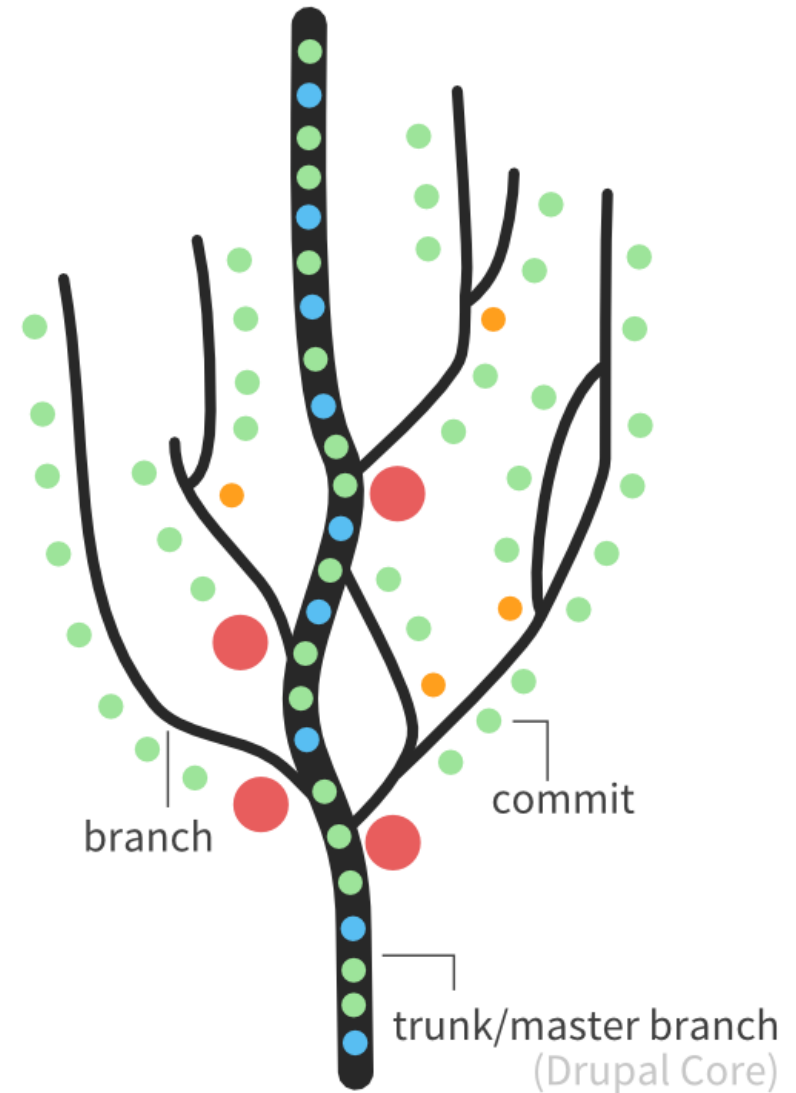
Controle de versão

- Controle de versão no Git



Controle de versão

- **Árvore de versões**

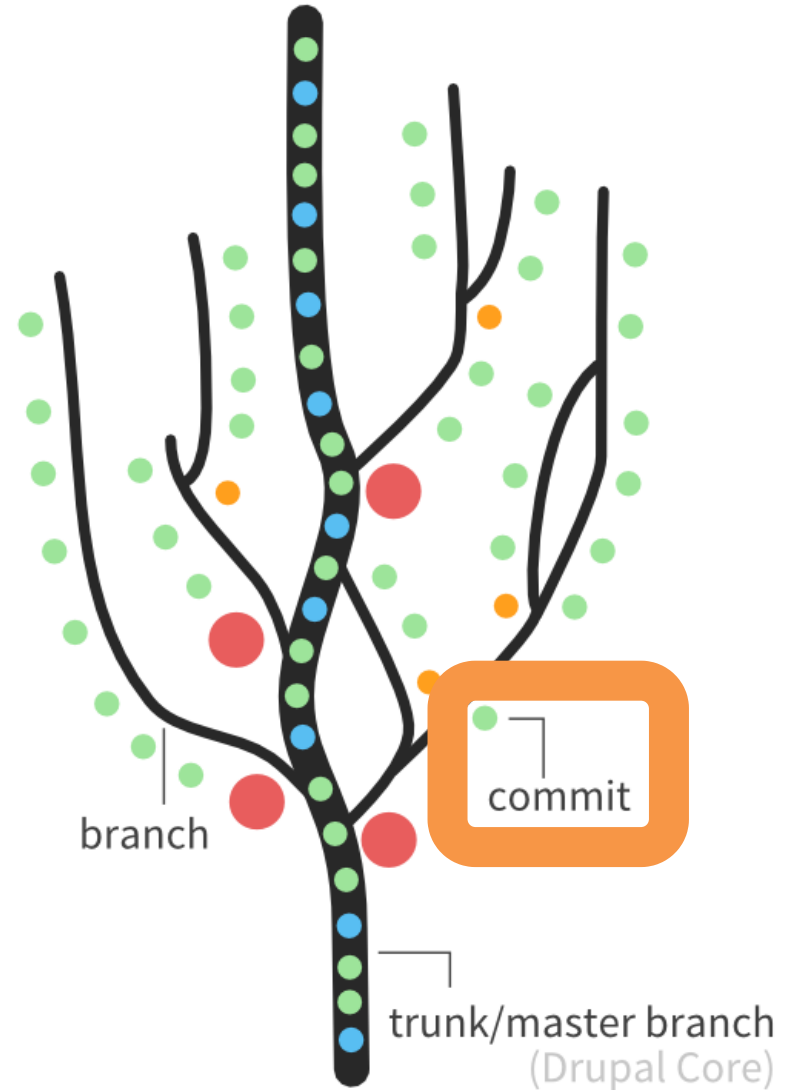


- Fonte: <http://www.drupal.org/node/991716>

Controle de versão

- **Commit**

Registro de modificações no histórico (entregar ou enviar uma modificação).



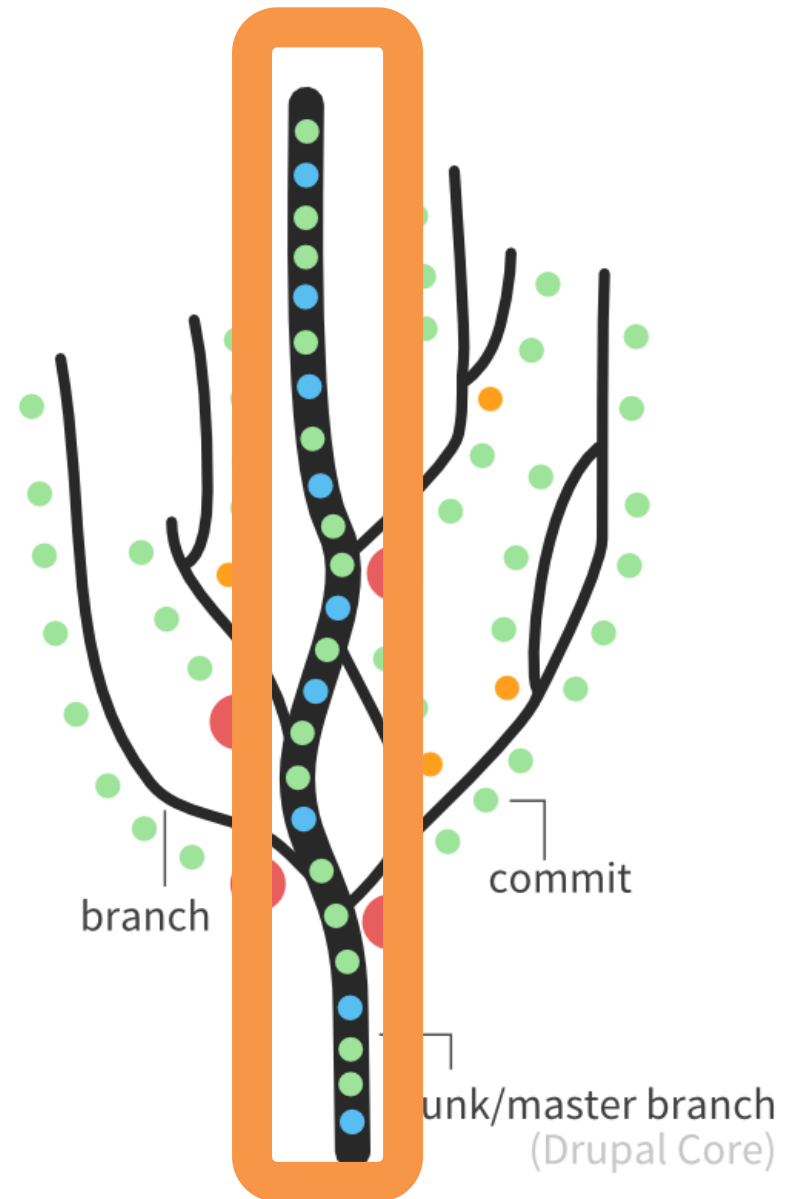
- Fonte: <http://www.drupal.org/node/991716>

Controle de versão

- Tronco (trunk) ou ramo master (branch master)

Histórico principal de modificações.

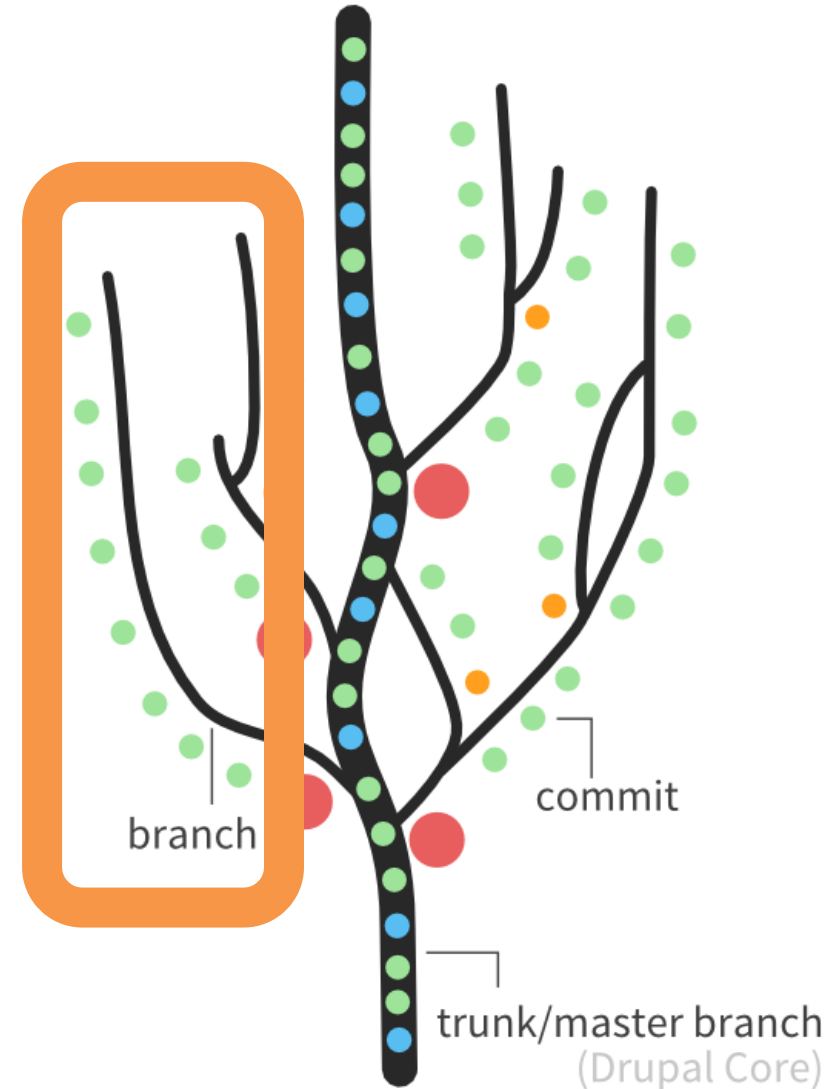
- Fonte: <http://www.drupal.org/node/991716>



Controle de versão

- **Ramo (branch)**

Histórico de modificações que se originam de outro histórico.

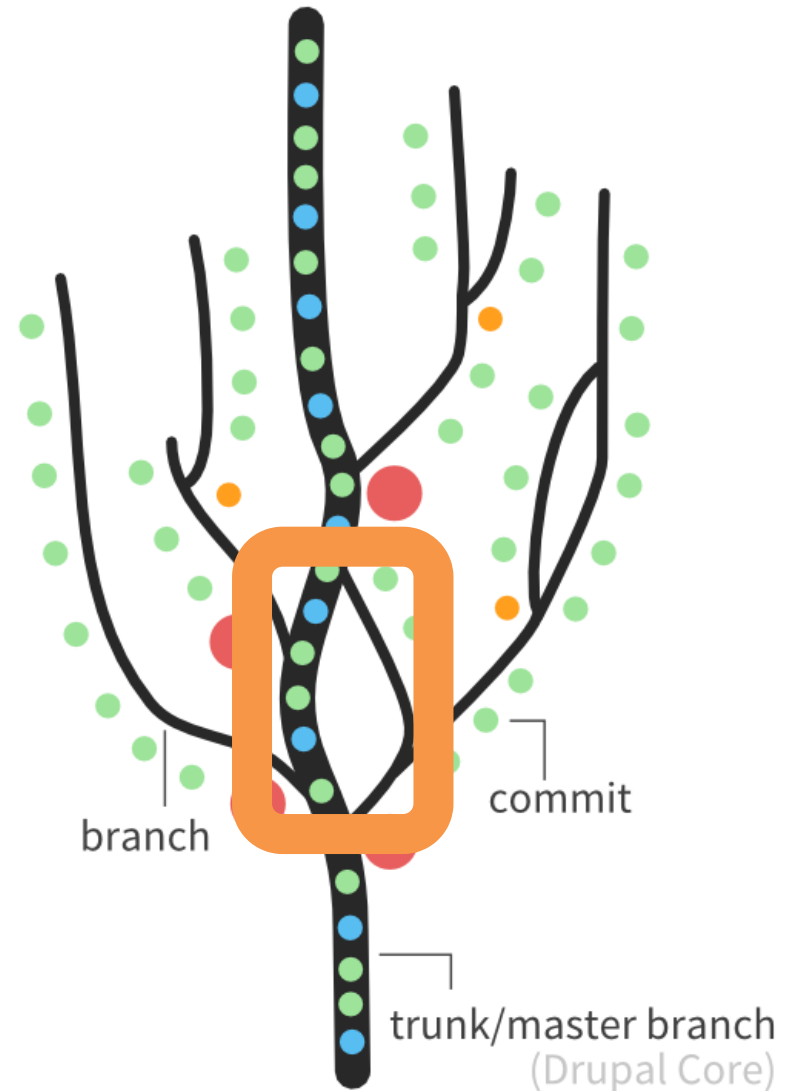


- Fonte: <http://www.drupal.org/node/991716>

Controle de versão

- Merge (fundir, misturar)

Incorporação das modificações de um branch em outro.



- Fonte: <http://www.drupal.org/node/991716>

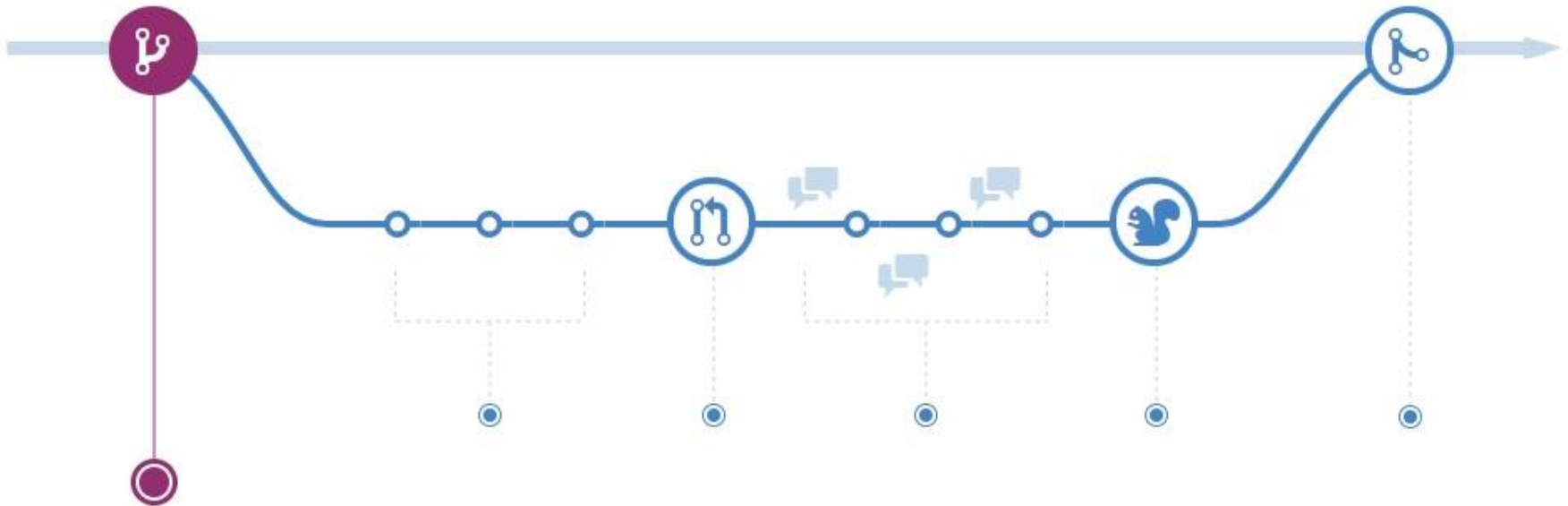
Workflow recomendado pelo Github

- 1. Criar um branch.**
- 2. Realizar os commits desejados no branch criado.**
- 3. Criar um pull request (solicitação de atualização).**
- 4. Discutir e revisar suas alterações com outros colaboradores.**
- 5. Deploy (implantar)**
- 6. Realizar o merge no branch master.**

Fonte: <https://guides.github.com/introduction/flow>

Workflow recomendado pelo Github

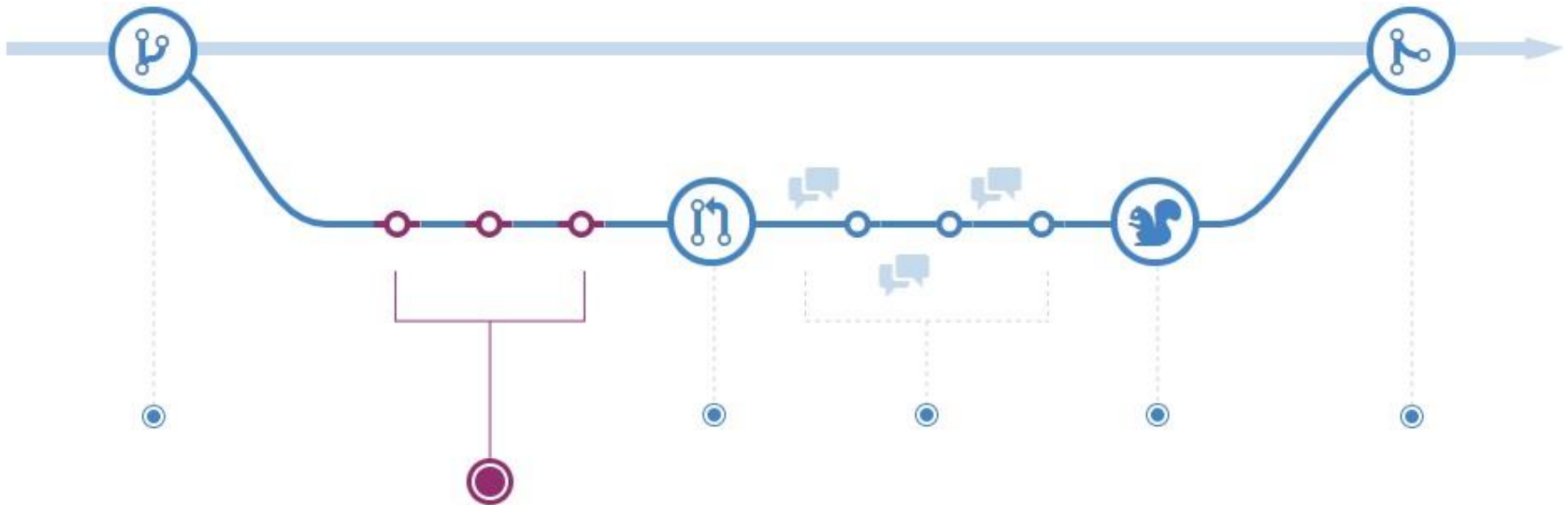
1. Criar um branch.



Fonte: <https://guides.github.com/introduction/flow>

Workflow recomendado pelo Github

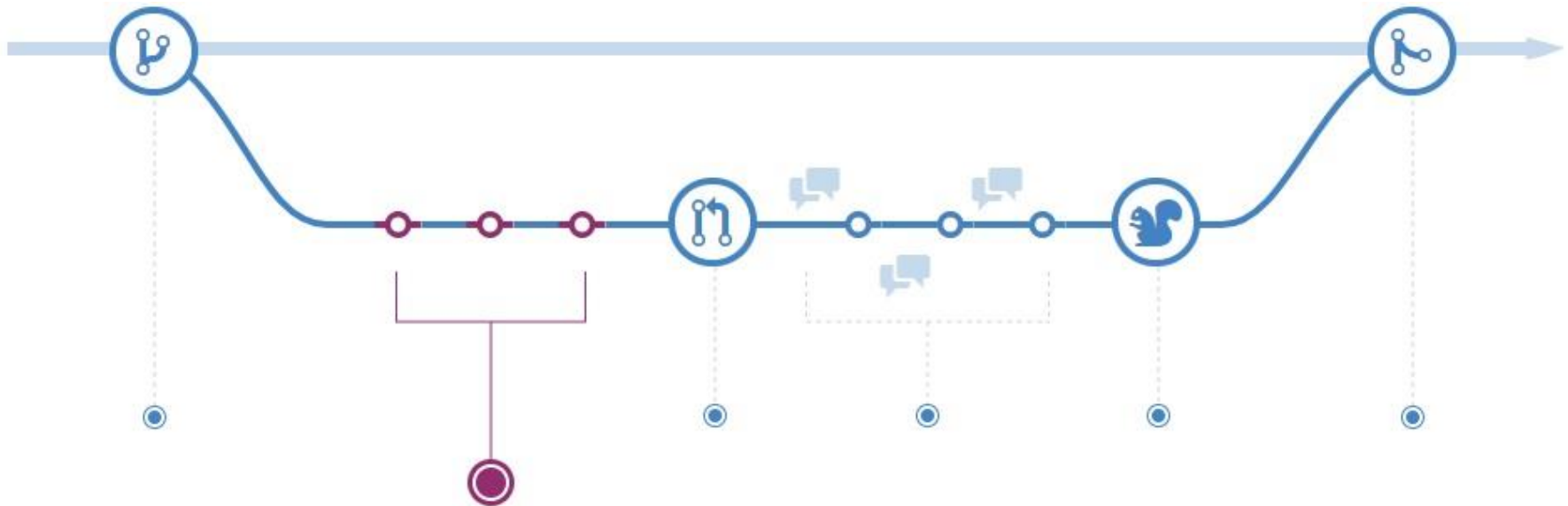
2. Realizar os commits desejados no branch criado.



Fonte: <https://guides.github.com/introduction/flow>

Workflow recomendado pelo Github

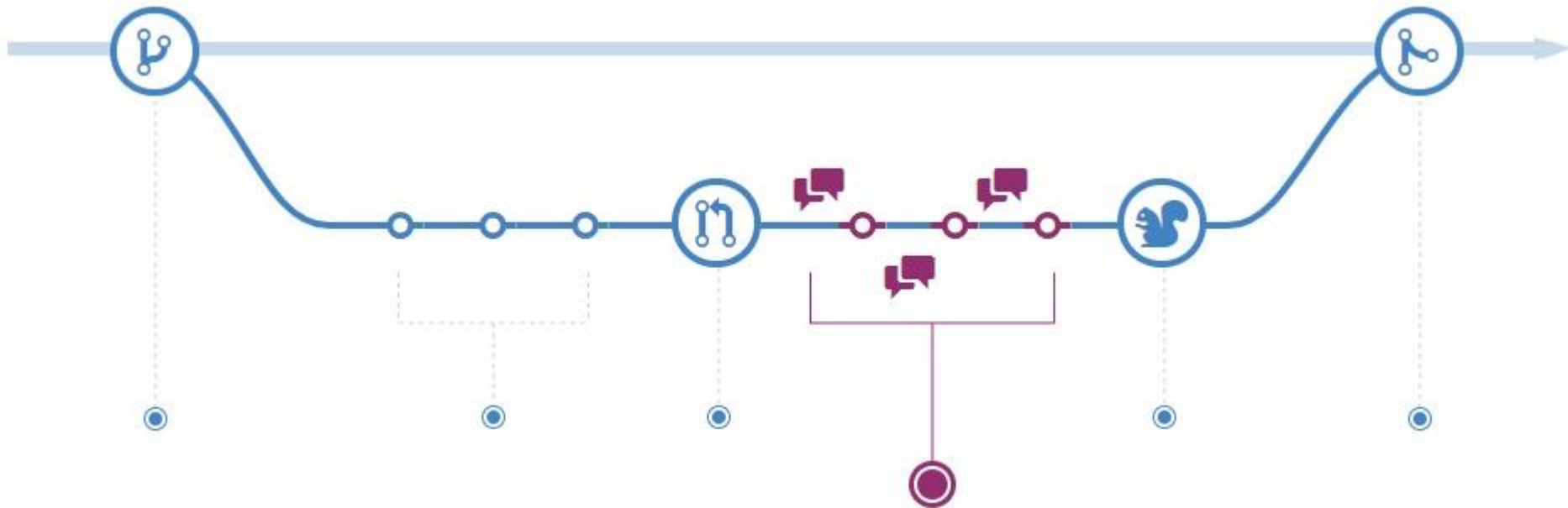
3. Criar um pull request (solicitação de atualização).



Fonte: <https://guides.github.com/introduction/flow>

Workflow recomendado pelo Github

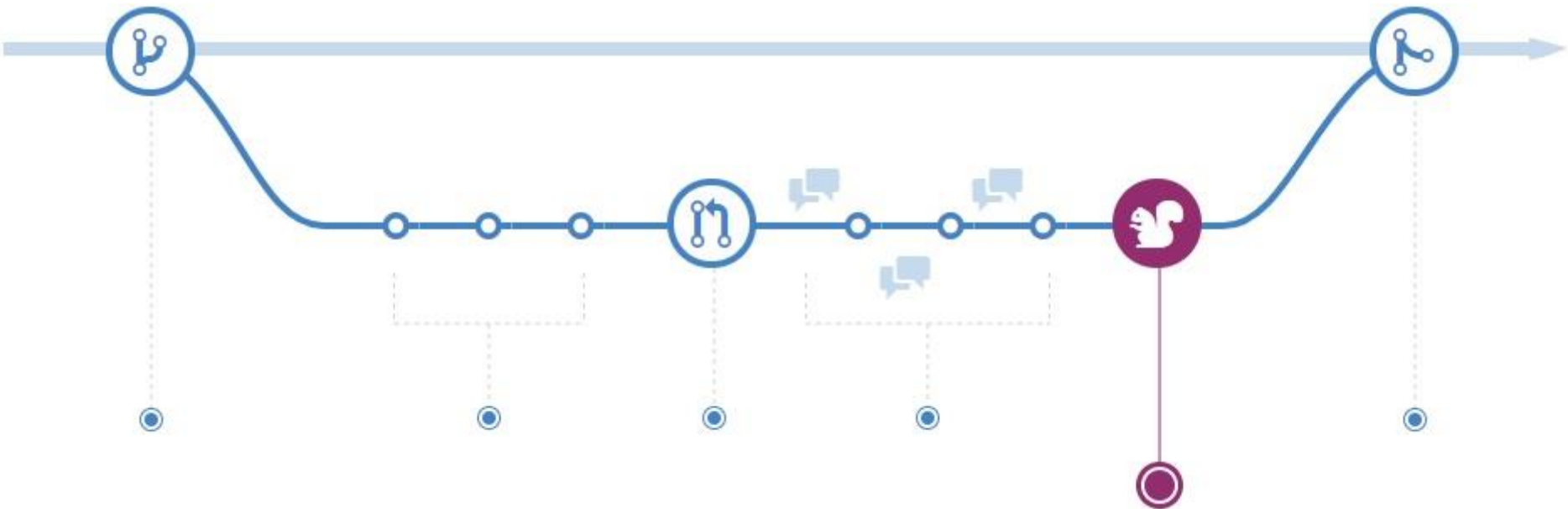
4. Discutir e revisar suas alterações com outros colaboradores.



Fonte: <https://guides.github.com/introduction/flow>

Workflow recomendado pelo Github

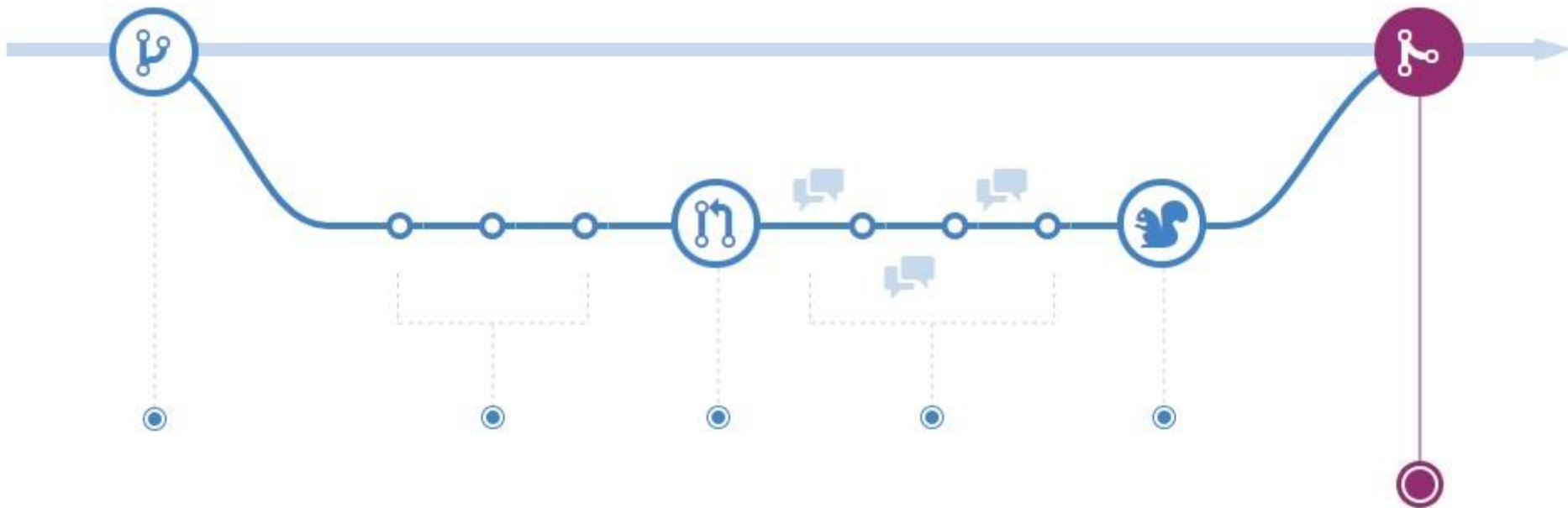
5. Deploy (implantar)



Fonte: <https://guides.github.com/introduction/flow>

Workflow recomendado pelo Github

6. Realizar o merge no branch master.



Fonte: <https://guides.github.com/introduction/flow>

GIT – Instalação

- Primeiro devemos testar se o computador a ser utilizado possui o GIT instalado. Para isso abra o **prompt** (CMD) no Windows e digite:

```
C:\Users\vanderson>git --version
```

- Se o resultado for parecido com:

```
git version 2.9.2.windows.1
```

Nada mais precisa ser feito.

- Caso algo parecido com o seguinte apareça:
'git' não é reconhecido como um comando interno ou externo, um programa operável ou um arquivo em lotes.
Entre em <https://git-scm.com/download/win> e baixe a versão compatível com o seu Windows.

GIT – Instalação

- Na instalação, basta aceitar todos os valores padrões do instalador até o fim. Quando acabar a instalação, teste novamente usando o CMD (não esqueça de abrir um novo).

```
C:\Users\vanderson>git -version
git version 2.13.3.windows.1
```

- Pronto, agora você tem o Git instalado na sua máquina. Se você não gosta de usar a linha de comando, existem vários clientes de Git para o Windows.
- Em sala vamos usar o cliente integrado no VSCode, mas alguns outros notáveis que possam interessar são: Github Desktop, SourceTree, TortoiseGIT.

GIT – Conta no Github

- O GitHub é o maior hospedador de repositórios GIT do mundo.
- Podemos usá-lo para colocar quantos repositórios quisermos, desde que estes sejam abertos.
- A maioria dos grandes softwares de código aberto estão no Github (<https://github.com/facebook/react>
<https://github.com/mozilla/gecko-projects>
<https://github.com/angular/angular.js>)
- Ótimo portfólio para desenvolvedores.
- Gratuito!! (Para ter repositórios privados é necessário pagar).

GIT – Conta no Github

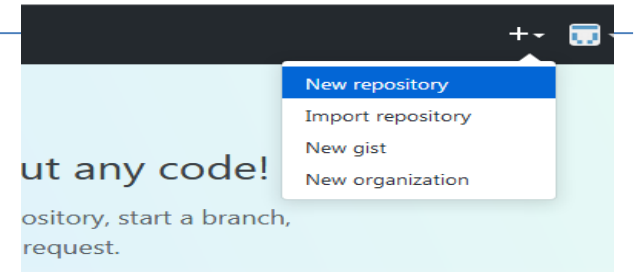
- Criar uma conta no Github é simples, basta criar um usuário e uma senha e possuir um e-mail válido:
 - Entre em <https://github.com>, clique no menu **Sign Up** e siga as instruções dos formulários.
 - Confirme o seu e-mail e pronto, a sua conta do Github já está pronta.
 - Vamos cobrir apenas as ações básicas do Git, para um tutorial interativo mais adequado, tente: <https://try.github.io>

Git – Primeiro Repositório

- O Git é um software de versionamento de código (VCS), utilizado para rastrear todas as mudanças feitas nos nossos softwares, de maneira elegante, mostrando todas as mudanças, linha a linha.
- A unidade principal do Git (e dos outros VCS) é o repositório (**repository**). Ele é uma pasta comum, com arquivos dentro, onde o Git rastreará todas as modificações feitas.
- Podemos criar um repositório no nosso PC e depois subir para o Github. Para essa aula, vamos criar primeiro no Github e depois passar (clonar) para o computador.

Git – Primeiro Repositório

- Conectado ao Github, clique no ícone + e depois no item *New Repository*
- No formulário, digite o nome do repositório (deve ser único entre os seus) e uma descrição.
- O resto é opcional, mas sempre é bom inicializar o projeto com o README (arquivo com instruções do projeto), um Gitignore File (veremos adiante) e uma licença.



Create a new repository

A repository contains all the files for your project, including the revision history.

Owner

Repository name

Great repository names are short and memorable. Need inspiration? How about...

Description (optional)

☒ Public

Anyone can see this repository. You choose who can commit.

☐ Private

You choose who can see and commit to this repository.

☒ Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you already have the repository on your computer.

Add .gitignore: Python

Add a license: Apache License 2.0



Create repository

Git – Primeiro Repositório

- Ao clicar no botão de criação (CREATE REPOSITORY), você será redirecionado para a página dele e pronto, seu primeiro repositório está criado.

The screenshot shows a GitHub repository page for 'TecWeb'. At the top, there are buttons for 'Watch', 'Star', and 'Fork', each with a count of 0. Below these are navigation tabs: 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Settings', and 'Insights'. The repository name 'TecWeb' is displayed, followed by a description: 'Repositório para entregas da disciplina TecWeb'. Below the description, there are statistics: '1 commit', '1 branch', '0 releases', and '1 contributor'. A row of buttons includes 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find file', and 'Clone or download'. The file list shows three files: '.gitignore', 'LICENSE', and 'README.md', all from the 'Initial commit' made 'a minute ago'. The 'README.md' file is selected, showing its content which includes the title 'TecWeb' and the same description as the repository page.

Repositório para entregas da disciplina TecWeb

1 commit 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

File	Commit	Time
.gitignore	Initial commit	a minute ago
LICENSE	Initial commit	a minute ago
README.md	Initial commit	a minute ago

README.md

TecWeb

Repositório para entregas da disciplina TecWeb

Git – Criando o seu primeiro site no Github

- Criado o seu repositório, está na hora de usá-lo para termos o nosso primeiro site simples.
- Vamos primeiro fazer uma cópia local do repositório, processo chamado de clonagem (**clone**).
- Na página do seu repositório, clique no botão **Clone or download** e copie o endereço que aparece com o protocolo https.
- Na linha de comando, vá até a pasta onde quer deixar a cópia e coloque o seguinte comando:

```
git clone https://github.com/USUARIO/REPO.git
```

- O processo deve pedir o seu usuário e senha.

Git – Criando o seu primeiro site no Github

The screenshot shows a GitHub repository interface. At the top, there's a navigation bar with links for 'This repository', 'Search', 'Pull requests', 'Issues', 'Marketplace', and 'Gist'. Below this, the repository name is redacted with a blue box. To the right, there are buttons for 'Watch' (0), 'Star' (0), and 'Fork' (0). A secondary navigation bar includes 'Code', 'Issues' (0), 'Pull requests' (0), 'Projects' (0), 'Wiki', 'Settings', and 'Insights'. The repository description is 'Repositório para entregas da disciplina TecWeb', with an 'Add topics' link and an 'Edit' button. A summary bar shows '1 commit', '1 branch', '0 releases', '1 contributor', and 'Apache-2.0' license. Below this, there are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find file', and a green 'Clone or download' button. The 'Clone or download' dropdown menu is open, showing 'Clone with HTTPS' (selected), 'Use SSH', and a text input field containing a redacted URL. Below the input field are buttons for 'Open in Desktop' and 'Download ZIP'. The repository file list shows '.gitignore', 'LICENSE', and 'README.md', all marked as 'Initial commit'.

Git – Criando o seu primeiro site no Github

- Estando clonado repare que todos os arquivos que eram listados na página da web estão disponíveis na pasta clonada.
- Existe uma outra pasta escondida (**.git**). Ela é utilizada pelo próprio git, não há necessidade de mexer.
- Com o repositório clonado, crie na raiz dele um arquivo chamado **index.html** com o seguinte texto (use o notepad++):

```
<h1> Olá Mundo TecWeb </h1>
```





Git – Criando o seu primeiro site no Github

- Agora vamos subir essa modificação. Toda vez que houver algo a ser mandado para o repositório original do GIT, são feitos três passos:
 - Se houverem arquivos **novos**, você deve adicioná-los no controle do git (**git add**).
 - Com tudo certo, você deve dizer para o git aceitar as suas mudanças e calcular todas as diferenças (**git commit**)
 - Com tudo calculado, você deve enviar as mudanças até a fonte original (**git push**)
- Usando essa sequência podemos ver que o arquivo agora está dentro do repositório.

Git – Criando o seu primeiro site no Github

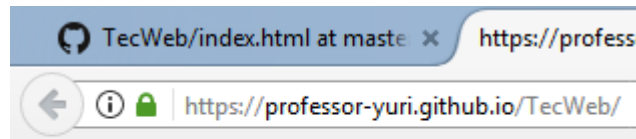
```
$ git add index.html
$ git commit -m "Primeiro Commit"
[master 40c5f2b] Primeiro Commit
 1 file changed, 1 insertion(+)
 create mode 100644 index.html
$ git push
Username for 'https://github.com': vanderson.bossi@impacta.edu.br
Counting objects: 3, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 296 bytes | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
```

Adiciona
mensagem

 .gitignore	Initial commit	7 days ago
 LICENSE	Initial commit	7 days ago
 README.md	Initial commit	7 days ago
 index.html	Primeiro Commit	9 minutes ago

Git – Criando o seu primeiro site no Github

- Para configurar o Github para mostrar o seu site, clique em **settings**, desça a tela que aparecer até o item **GitHub Pages**.
- No subitem **Source** escolha a opção *master branch* e depois em **Save**.
- Pronto, o seu site está navegável no endereço:
<https://USUARIO.github.io/REPOSITORIO/>



Olá mundo, TecWeb!

Exercício 1

- a) Acesse sua conta no GitHub.
- b) Crie um repositório com o nome **devops-aula02**. O repositório deve ser **público** e inicializado com um arquivo **README.md**.
- c) Escreva neste arquivo os nomes de seus 3 atores/atrizes preferidos e efetue o **commit**.
- d) Clique no botão que está com o rótulo **Branch: master**. Escreva no campo texto o nome **ramo-musicas** e clique na opção **Create branch: ramo-musicas**.
- e) Edite o arquivo **README.md** (clique no arquivo e depois clique no botão com a figura de um lápis/caneta). Altere o nome do 2º item na sua lista e adicione um item a mais no final da lista.

Exercício 1 - Continuação

- f) Efetue o **commit** após as alterações.
- g) Ainda no branch **ramo-musicas**, clique no botão **Create a new file** e crie um arquivo chamado **musicas.md**. Escreva o nome de 3 músicas de que você gosta e efetue o **commit**.
- h) Selecione o **branch master** e verifique que o conteúdo está diferente do **branch ramo-musicas**.
- i) Crie um **branch** chamado **ramo-games** a partir do **master**.
- j) Crie neste novo **branch** um novo arquivo chamado **games.md** e escreva nele os nomes de 3 jogos de que você gosta.
- k) Verifique que agora seu repositório possui 3 **branches**, cada um deles com um conjunto de arquivos diferentes.

Exercício 2 – 1/3

- a) Acesse sua conta no GitHub.
- b) Acesse o repositório **devops-aula02**.
- c) Selecione a aba **Pull requests** e pressione o botão **New pull request**.
- d) Na página **Compare changes**, selecione no primeiro campo o **branch master**, e no segundo campo o **branch: ramo-games**.
- e) Pressione o botão **Create pull request**.
- f) Preencha o campo com uma mensagem descrevendo a modificação (por exemplo, "inclusão da lista de jogos") e pressione **Create pull request**.

