

Extensão

Construindo vivências,
Transformando a realidade

Curso:

**Git e GitHub: trabalhe
colaborativamente e mostre
todo seu potencial como
desenvolvedor**

Professor responsável:

Henrique Poyatos

<henrique.poyatos@ecossistemaanima.com.br>





Henrique Poyatos



- Mestrado em Educação Profissional pelo Centro Paula Souza;
- Especialista em *Blockchain* pela Universidade de Buffalo;
- Especialista em Gerenciamento de Projetos PMI pela FIAP e Tecnólogo em processamento de dados pela FIAP;
- Experiência como coordenador acadêmico e conteudista no FIAP ON (*Digital Learning*).
- Foi gerente de projetos em consultorias para clientes como Caixa Econômica Federal e Nossa Caixa.
- Atualmente, professor de graduação e pós e assessor nacional de Estratégia Acadêmica na Anima Educação.

E-mail: henrique.poyatos@ecossistemaanima.com.br

LinkedIn: <https://www.linkedin.com/in/hpoyatos/>



Vamos criar uma comunidade virtual de aprendizagem?

Não espere até o próximo sábado! Vamos manter contato e aprender durante o período de nosso encontro e também além do período “em sala de aula”?

- **Avisos, Links interessantes, Atividades regulares e extras, troca de ideias!**

Link do servidor no Discord:

<https://discord.gg/ZNPtUnNvqU>

ENTRE NO SERVIDOR, pegue a role de aluno E ME PROCURE NO PRIVADO (Henrique Poyatos#5480) para ganhar a role aluno-curso-python-2022-2 e entrar no canal privativo git-e-github-trabalhe-colaborativamente-e-mostre-todo-seu-potencial-como-desenvolvedor-2022-2

Não temos WhatsApp! ENTRE NESTE SERVIDOR DISCORD <https://discord.gg/ZNPtUnNvqU> E ME PROCURE NO PRIVADO (Henrique Poyatos#5480) para entrar no canal privativo git-e-github-trabalhe-colaborativamente-e-mostre-todo-seu-potencial-como-desenvolvedor-2022-2



**SEJAM
BEM-
VINDOS!**

Cursos de extensão
possuem curta duração

São ministrados por
Professores
responsáveis

Possuem foco na
ampliação do
repertório de
conhecimentos

Promovem interação
entre pessoas, áreas e
instituições



INFORMAÇÕES IMPORTANTES AO ALUNO

Você deve estar matriculado e visualizando seu projeto no Ulife – sala de aula virtual.

As matrículas foram realizadas de acordo com ordem de inscrições e de preferência. Assim, caso não esteja matriculado, não poderá participar dos encontros, pois não receberá certificado e horas equivalentes em seu histórico, conforme estabelecido em Edital.

Não é permitido o compartilhamento de links dos encontros.

Os critérios para aprovação no curso são: envolvimento, participação e presença nos encontros síncronos.

Para acessar os encontros, utilize o link disponibilizado em seu Ulife – sala de aula virtual.

Uma vez aprovado (desempenho lançado acima de 70%), em até 45 dias você receberá as horas em seu histórico escolar e poderá baixar seu certificado por meio de protocolo que é gerado automaticamente no seu Ulife, no caminho: **Menu > Serviços > Solicitações online > Emissão de Certificado de Curso de Extensão.**

ATENÇÃO: o protocolo é gerado automaticamente! Você não precisa abrir protocolo!

Git e GitHub: trabalhe colaborativamente e mostre todo seu potencial como desenvolvedor

Ementa: Uma quantidade cada vez maior de recrutadores exige dos potenciais candidatos à uma vaga de desenvolvimento de sistemas sua conta no GitHub. “SHOW ME THE CODE!”, eles dirão! Todo seu potencial como desenvolvedor pode ser comprovado ali. Chega de fazer provas de conhecimento infrutíferas, todas as provas que a empresa precisa de que você é um excelente profissional podem estar ali!

O Git é o mais utilizado controlador de versões de código-fonte do planeta, sendo o GitHub sua principal plataforma. Não sabe como utilizá-los? Pois esse curso é para você! Veremos os principais comandos do Git, como ele se integra com o GitHub e outras ferramentas do tipo IDE (como o Visual Studio Code) e até mesmo outras integrações e automações bem bacanas. Venha conosco e nunca mais perca um código-fonte importante ou uma vaga profissional que lhe peça o GitHub!

Carga horária: 12 horas

Quando: sábados das 8h00 às 10h30

Datas: 22/10/2022, 29/10/2022, 05/11/2022 e 12/11/2022



Git e GitHub: trabalhe colaborativamente e mostre todo seu potencial como desenvolvedor

Objetivo geral

- Criar um repositório GitHub e realizar o controle de versões de software usando git.

Objetivos específicos

- Controlar versões de código usando git;
- Publicar seu portfolio profissional usando o GitHub;
- Controlar código-fonte com ramos (branches);
- Automação de processos DevOps usando GitHub.



Git e GitHub: trabalhe colaborativamente e mostre todo seu potencial como desenvolvedor

Cronograma (previsto)

Aula 1 – O que é o git e GitHub, como explorar os recursos

Comandos básicos do git, sincronização com o GitHub

Aula 2 – Comandos básicos do git, sincronização com o GitHub

Aula 3 – Controle de versão usando branches

Aula 4 – Automação de processos usando GitHub



Git e GitHub: trabalhe colaborativamente e mostre todo seu potencial como desenvolvedor

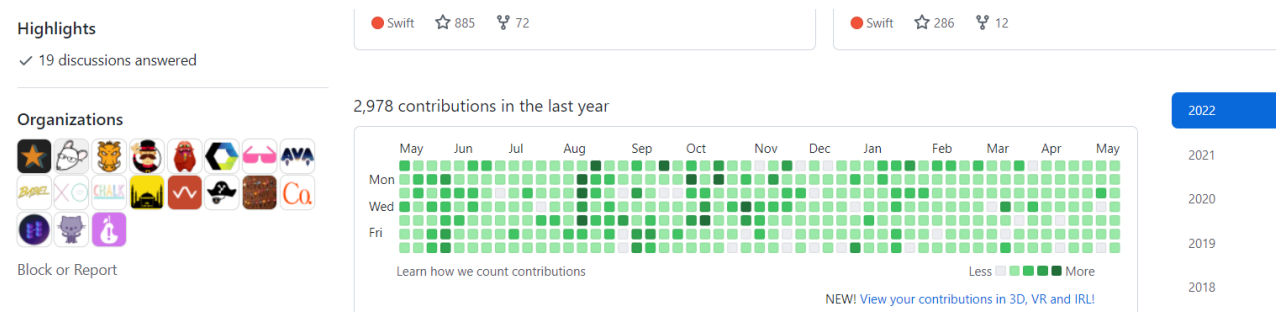
Pré-requisitos:

- **Nenhum!**



Requisitos de Aprovação do Curso

- **Assistir às aulas ao vivo:** Precisas estar presente e participar de pelo menos três das quatro aulas dos dias 22/10/2022, 29/10/2022, 05/11/2022 e 12/11/2022;
- **Criar uma conta no GitHub:** Vamos resolver isso hoje (ou sábado que vem)!
- **Seu GitHub precisa ser movimentado:** nas últimas três semanas de curso, espera-se um movimento mínimo, o gráfico no dashboard precisa ficar verdinho! (avaliarei no dia 12/11 o GitHub que for informado na lista de presença)



O que o curso é (e o que não é)

- **Vamos aprender na prática!:** Oserão fornecidos slides e até mesmo material complementar. Contudo, o curso é prática. Vamos aprender fazendo juntos!
- **O curso é básico!:** O curso não tem pré-requisitos e começa do básico? Já conhece bastante? Ótimo! Mas aqui o ritmo vai ser devagar!
- **Perdeu uma aula? Não se desespere:** é para isso que servem as gravações das aulas disponíveis no Ulife. Só na vale perder a aula, não assistir à gravação e chegar na semana que vem achando que vai entender algo...
- **A aula não é divã ou plantão para problemas:** “Professor, estou com problema ...”. Temos centenas de pessoas nesta aula, ao vivo. Vamos otimizar o tempo de aula. Procure-me de forma privada no discord e vamos resolver fora do horário da aula!
- **Tem lista de presença?** Sempre! Ela será disponibilizada apenas das dez da manhã em diante, por favor, não insista.
- **Paramos apenas dez minutos:** Faremos um *break* rápido das 9h30 às 9h40, peço sua atenção plena no restante do período.



Agenda de hoje

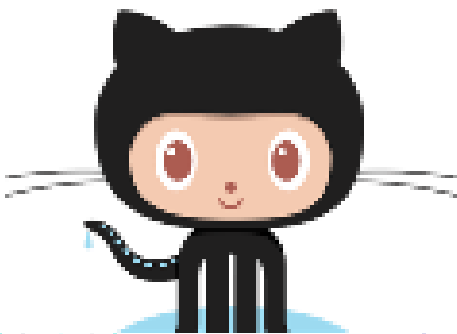
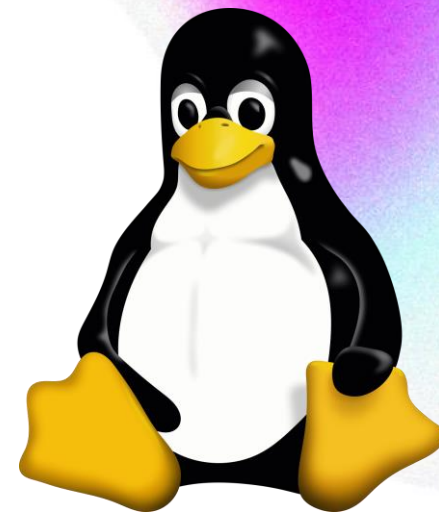
- O que é o git?
- Importância do git, exemplos de uso
- O que é o GitHub?
- Importância do GitHub, exemplos de uso
- *Hands-on!*



git

O que é o git?

- É um SCM – Source Control Management, ou um gerenciador de controle de versões (conhecido também como Git-SCM);
- Diferente da concorrência, seus repositórios são distribuídos!
- Criado por Linus Torvalds em 2005 para melhorar o gerenciamento das versões do Linux.



Gerência de Configuração de Software

- Durante o desenvolvimento de software, precisamos saber estas informações:
- Quem realizou a mudança?
- Por que realizou a mudança?
- O que mudou e quando?
- Podemos reproduzir (ou mesmo reverter) esta mudança?



Gerência de Configuração de Software

- **Identificação**
- **Documentação**
- **Controle**
- **Auditoria**



Artefatos possíveis em um SCM

- **Código-fonte;**
- **Documentação do Software;**
- **Manual de usuário.**



Problema exemplo

Você precisa editar um código que está em um *cloud* pessoal (ex. Google Drive);

1. Baixa o arquivo;
2. Faz as alterações necessárias;
3. Sobe o arquivo no Google Drive novamente.



Problema exemplo

Agora um colega seu também quer editar o mesmo código:

- 1. Vocês baixam o mesmo arquivo ao mesmo tempo;**
- 2. Editam o código-fonte, cada um em sua máquina;**
- 3. Os dois sobem o arquivo, um é sobrescrito pelo outro, que subiu 1º. perde suas alterações.**



Solução: um software que faça um MERGE

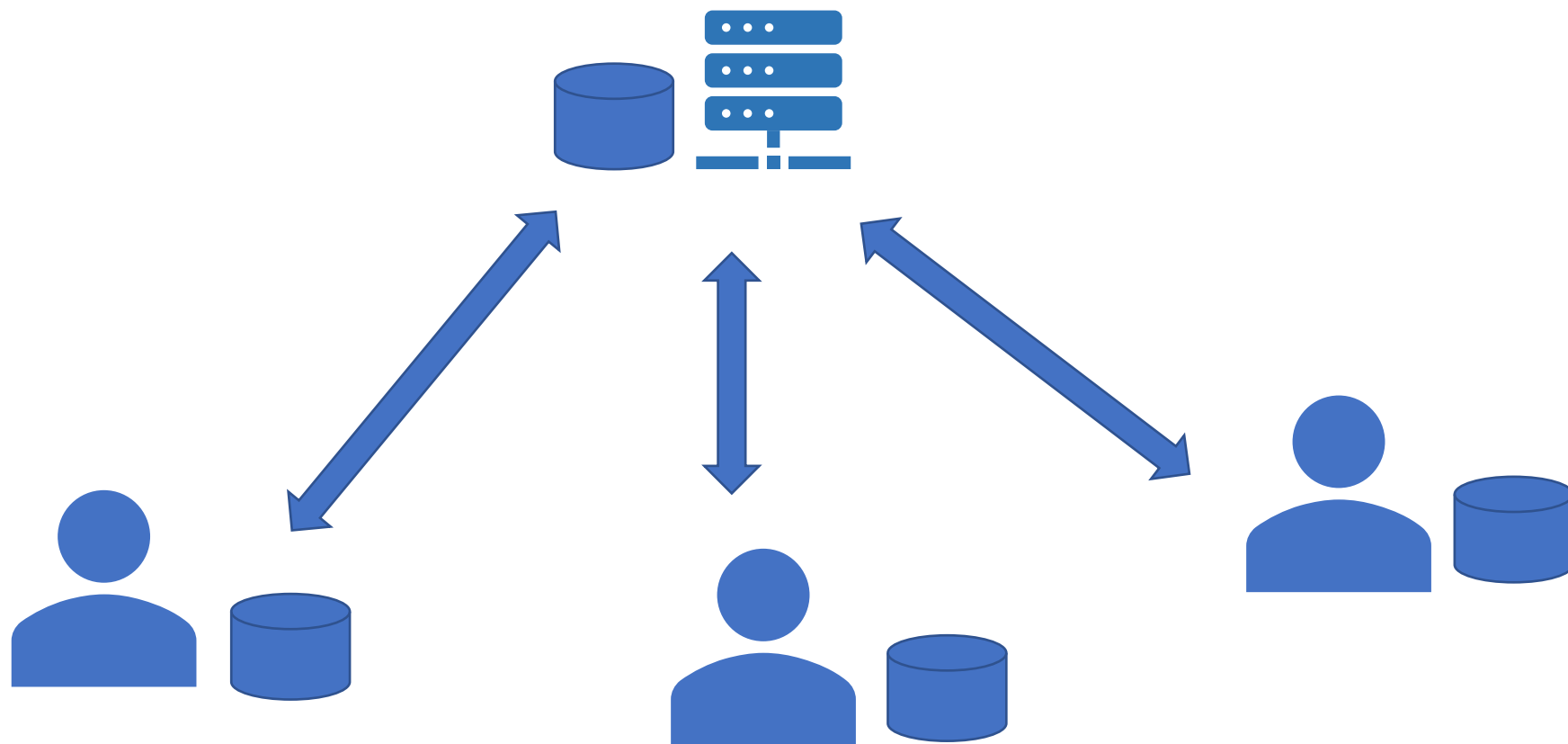


A concorrência

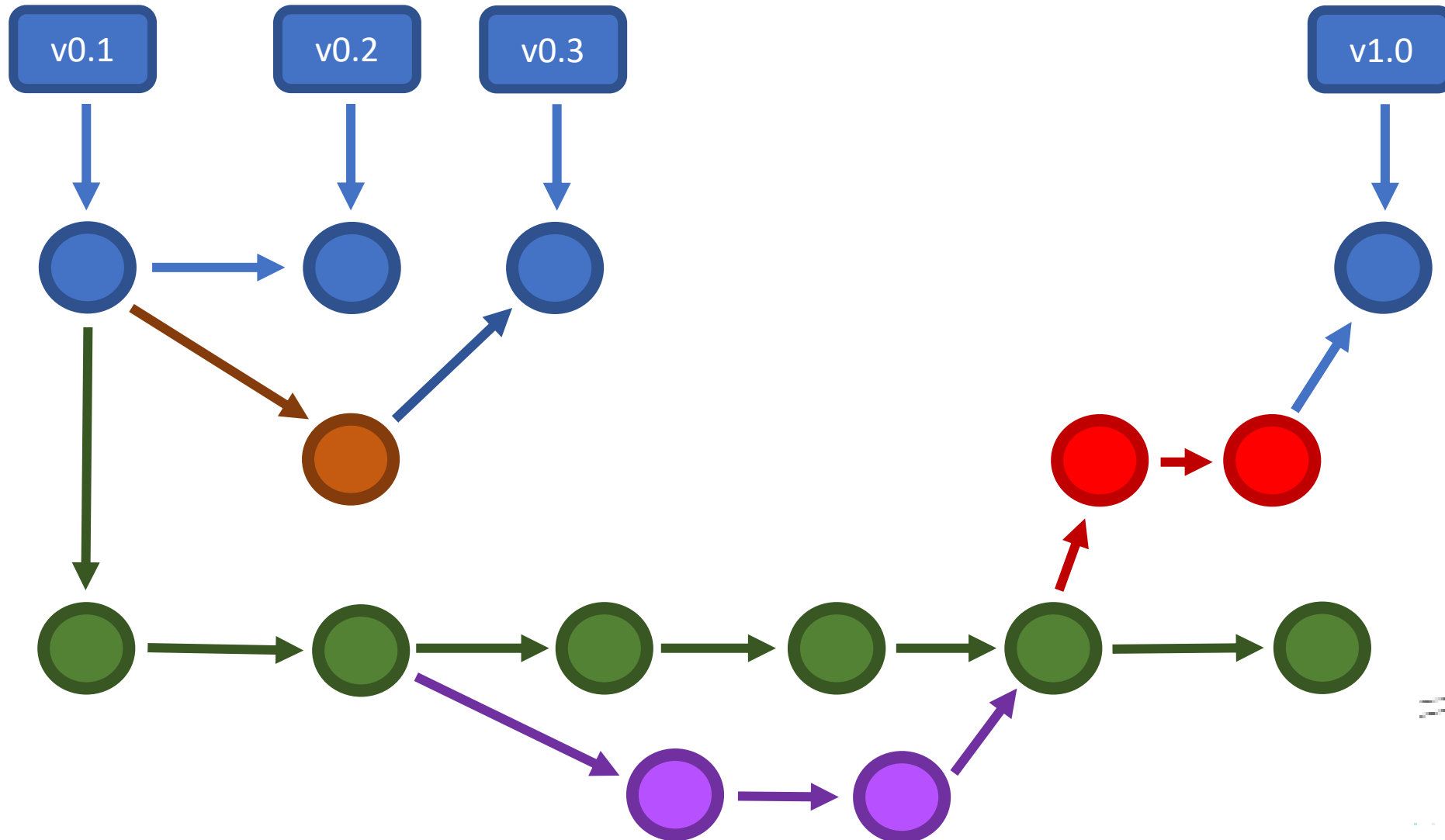
- **Concurrent Versioning System (CVS)**
- **Subversion (SVN);**
- **Mercurial**
- **Bazaar**
- **Microsoft Visual Source Safe (VSS) – descontinuado**
- **Contudo, o Git é mais rápido e eficiente.**



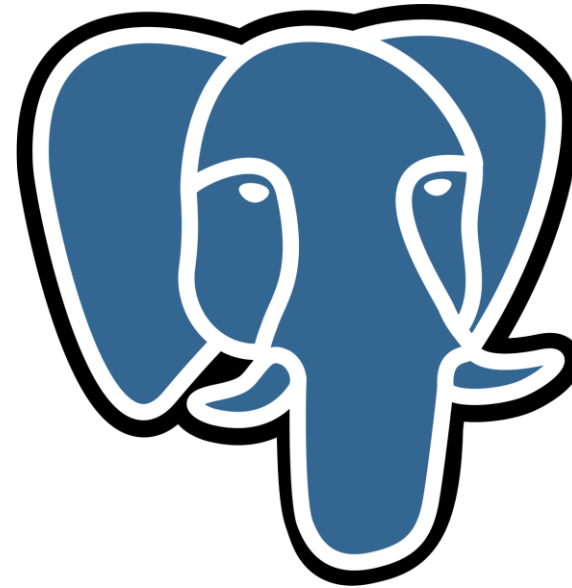
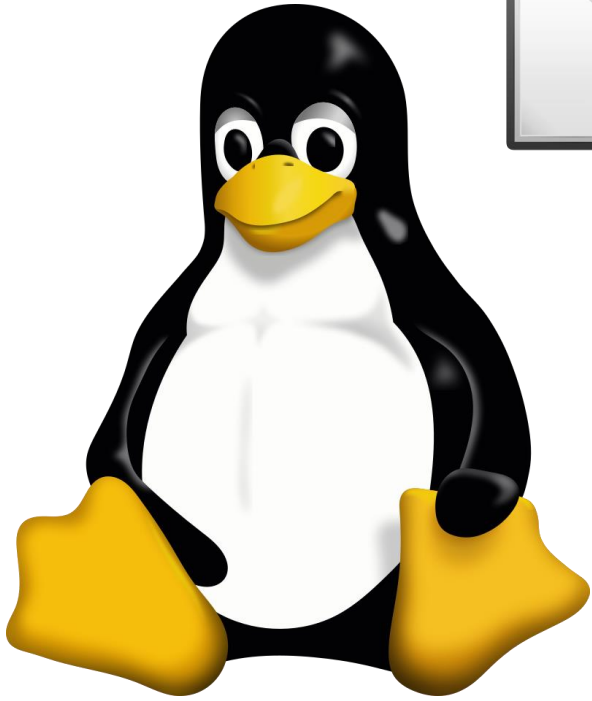
Git é distribuído



Controle de Versões – como funciona?



Quem usa Git?



Microsoft



Git Hub

O que é GitHub?

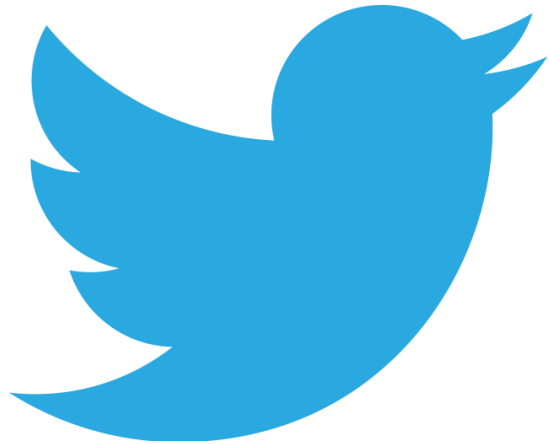
- Principal servidor de repositórios Git
- Surgiu em 2008;
- 73 milhões de desenvolvedores – Novembro/2021;
- 200 milhões de repositórios (sendo pelo menos 28 milhões deles públicos) – Novembro/2021 (<https://en.wikipedia.org/wiki/GitHub>);
- Microsoft começou a usar significativamente em 2012 e, em 2018, adquiriu o GitHub em um negócio milionário de US\$ 7,5 bilhões. (<https://www.theverge.com/2018/10/26/17954714/microsoft-github-deal-acquisition-complete>)



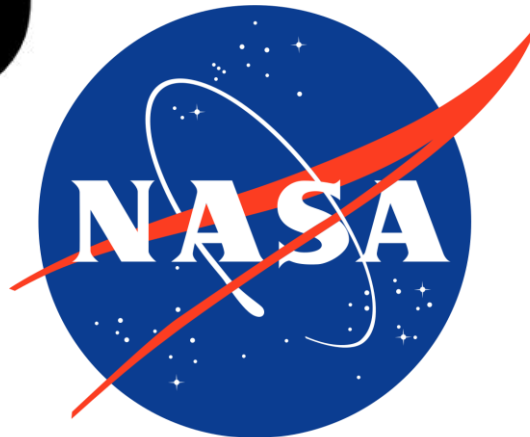
Quem usa GitHub?

Google

facebook®



Spotify®



zoom

LinkedIn



Hands-on Git

Instalação do git

- **Windows:** Baixe o git em <https://git-scm.com/download/win>
- **Mac OS X/Linux:** Baixe a partir de um repositório do sistema operacional
- **Sistemas baseados em Debian:**
`sudo apt-get install git`

Download for Windows

[Click here to download](#) the latest (2.36.1) 64-bit version of Git for Windows. This is the most recent maintained build. It was released 4 days ago, on 2022-05-09.

Other Git for Windows downloads

Standalone Installer

[32-bit Git for Windows Setup.](#)

[64-bit Git for Windows Setup.](#)

Portable ("thumbdrive edition")

[32-bit Git for Windows Portable.](#)

[64-bit Git for Windows Portable.](#)

Using winget tool

Install [winget](#) tool if you don't already have it, then type this command in command prompt or Powershell.

```
winget install --id Git.Git -e --source winget
```



git help

- **Como obter ajuda?**
git help

```
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
        [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
        [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
        [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
        [--super-prefix=<path>] [--config-env=<name>=<envvar>]
        <command> [<args>]
```

These are common Git commands used in various situations:

start a working area (see also: `git help tutorial`)

<code>clone</code>	Clone a repository into a new directory
<code>init</code>	Create an empty Git repository or reinitialize an existing one

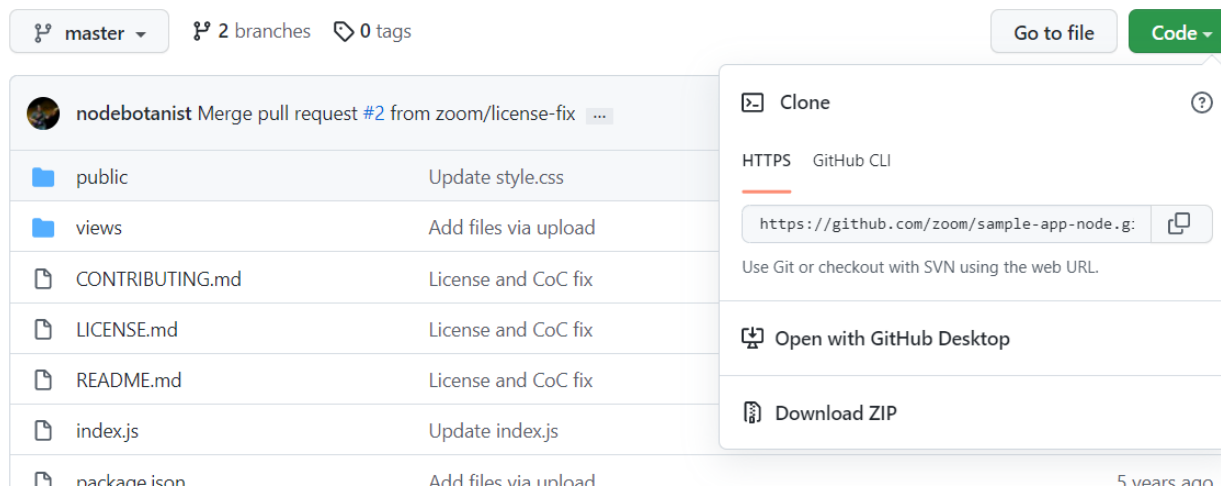
work on the current change (see also: `git help everyday`)

<code>add</code>	Add file contents to the index
<code>mv</code>	Move or rename a file, a directory, or a symlink
<code>restore</code>	Restore working tree files
<code>rm</code>	Remove files from the working tree and from the index



git clone – Clonado repositórios

- Para clonar diretórios, use:
`git clone <url do repositório>`
- Exemplo: acesse <https://github.com/zoom>



- `git clone https://github.com/zoom/sample-app-node.git`



git init – Iniciando um repositório (local)

- Para iniciar um repositório (local), crie um diretório e digite: `git init`
- Criará um diretório `.git` (oculto) que fará todo o controle de versões do repositório.

OS (C:) > Users > hpoya > Projetos > MeuPrimeiroProjeto > .git				
Nome	Data de modificação	Tipo	Tamanho	
hooks	13/05/2022 21:22	Pasta de arquivos		
info	13/05/2022 21:22	Pasta de arquivos		
objects	13/05/2022 21:22	Pasta de arquivos		
refs	13/05/2022 21:22	Pasta de arquivos		
config	13/05/2022 21:22	Arquivo	1 KB	
description	13/05/2022 21:22	Arquivo	1 KB	
HEAD	13/05/2022 21:22	Arquivo	1 KB	

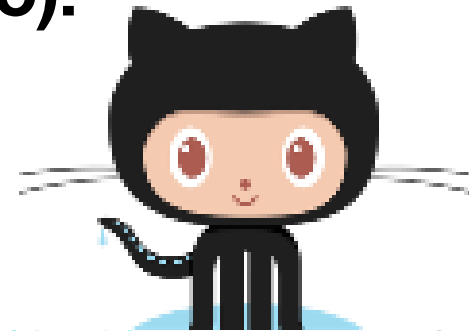


‘Versionando’ no repositório (local)

- **Vamos a um exemplo de arquivo (index.html)**

```
<html>
  <head>
    <title>Meu primeiro projeto</title>
  </head>
  <body>Meu primeiro projeto</body>
</html>
```

- **Verificando a situação do repositório (a qualquer momento):**
git status



‘Versionando’ no repositório (local)

- Para adicionar um arquivo no repositório (e passar a controlar suas versões:
`git add <arquivo>`
- Submetendo/publicando o arquivo:
`git commit [-m "Mensagem"]`
- Verifique as alterações com:
`git show`



‘Versionando’ no repositório (local)

- **Realize uma alteração**

```
<html>
  <head>
    <title>Meu primeiro projeto</title>
  </head>
  <body>Meu primeiro projeto<br>Linha
2</body>
</html>
```

- **Cheque com:**
`git status`
- **Verifique a alteração em detalhes com:**
`git diff <arquivo>`
- **Adicione e versione novamente... git add, commit, show, ... 😊**



Removendo arquivos do repositório (local)

- Crie um arquivo teste.txt
- Cheque com:
`git status`
- Adicione no repositório
- Se arrependeu?

```
git rm <arquivo> --cached
```

E sem `--cached` caso queira remover o arquivo de tudo!

```
git rm <arquivo>
```



O arquivo README.md

Um README, muitas vezes, é o primeiro item que um visitante verá ao visitar seu repositório. Os arquivos README geralmente incluem informações sobre:

- O que o projeto faz
- Por que o projeto é útil
- Como os usuários podem começar a usar o projeto
- Onde os usuários podem obter ajuda com seu projeto
- Quem mantém e contribui com o projeto

If you put your README file in your repository's hidden .github, root, or docs directory, GitHub will recognize and automatically surface your README to repository visitors.

If a repository contains more than one README file, then the file shown is chosen from locations in the following order: the .github directory, then the repository's root directory, and finally the docs directory.

Fontes: <https://docs.github.com/pt/repositories/managing-your-repositorys-settings-and-features/customizing-your-repository/about-readmes>

e <https://www.alura.com.br/artigos/como-criar-um-readme-para-seu-perfil-github>



Identificando-se corretamente

Configure seu nome e e-mail:

```
git config --global user.name "MeuUserName"
```

```
git config --global user.e-mail "meuemail@email.com"
```



Hands-on Git Hub

Repositório padrão main x master

O GitHub (assim com vários projetos) abandonaram a terminologia master/slave e passou a usar o termo “main” para a *branch* principal

Para facilitar e alterar a *branch* principal:

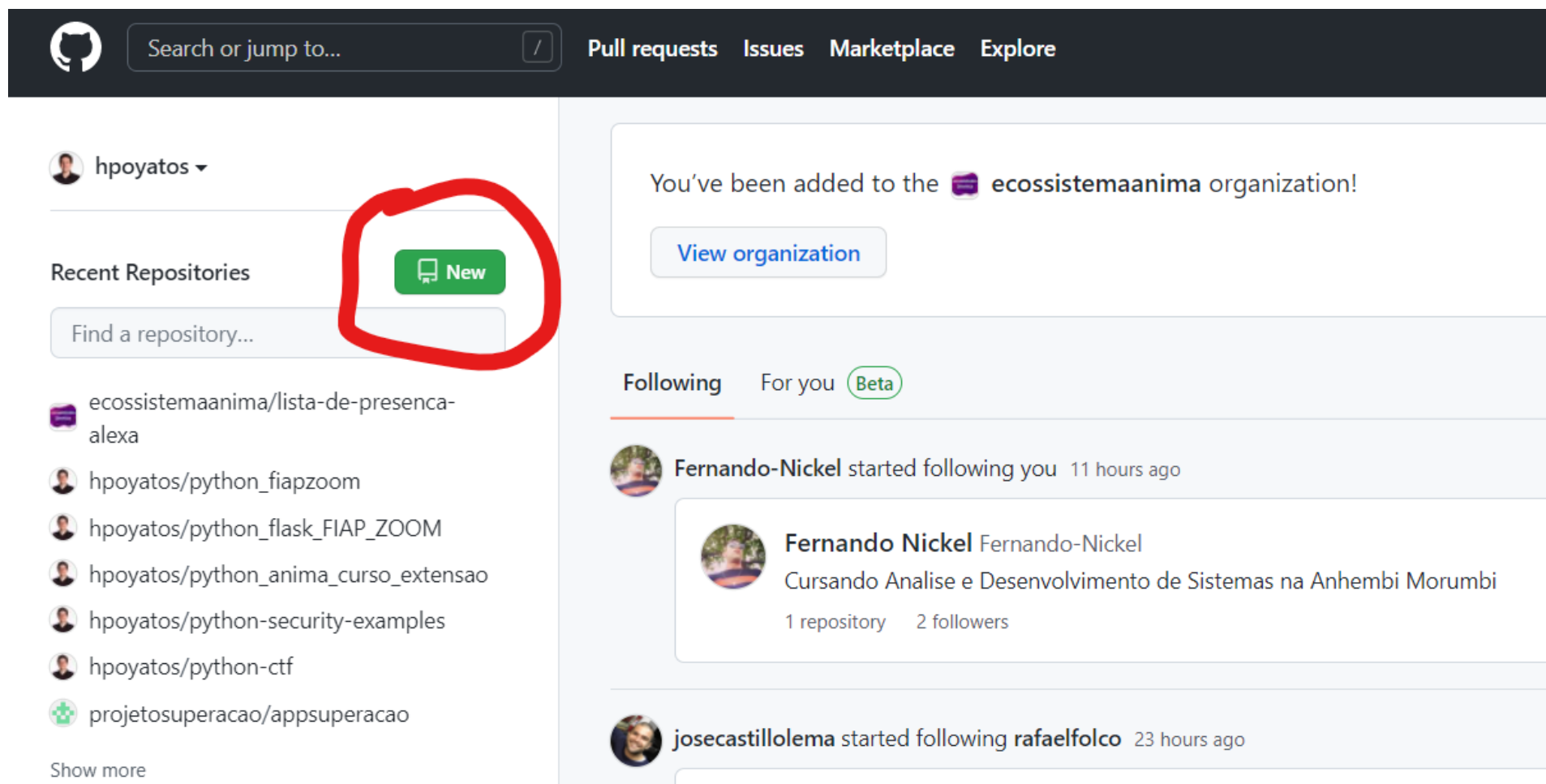
```
git config --global init.defaultBranch main
```


Contudo, dá para criar repositório com o parâmetro “b”

```
git init -b main
```



Git Hub – Hands-on




The screenshot shows the GitHub homepage for user **hpoyatos**. The top navigation bar includes the GitHub logo, a search bar with the text "Search or jump to...", and links for "Pull requests", "Issues", "Marketplace", and "Explore". On the left sidebar, under the user profile, there is a "Recent Repositories" section with a search bar "Find a repository...". A red circle highlights a green "New" button with a computer icon. Below the search bar, a list of repositories is shown, including "ecossistemaanima/lista-de-presenca-alexa" and several repositories by "hpoyatos". The main content area features a notification: "You've been added to the  **ecossistemaanima** organization!" with a "View organization" button. Below this, there are tabs for "Following" and "For you (Beta)". The "Following" tab is active, showing a list of users who started following the user, including "Fernando-Nickel" and "josecastillolema".

hpoyatos ▾

Recent Repositories

Find a repository...

 New

ecossistemaanima/lista-de-presenca-alexa


hpoyatos/python_fiapzoom

hpoyatos/python_flask_FIAP_ZOOM


hpoyatos/python_anima_curso_extensao

hpoyatos/python-security-examples

hpoyatos/python-ctf


 projetosuperacao/appsuperacao


Show more


You've been added to the  **ecossistemaanima** organization!

[View organization](#)

Following For you (Beta)

 **Fernando-Nickel** started following you 11 hours ago

 **Fernando Nickel** Fernando-Nickel
Cursando Analise e Desenvolvimento de Sistemas na Anhembí Morumbi
1 repository 2 followers

 **josecastillolema** started following **rafaelfolco** 23 hours ago



Git Hub – Hands-on

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

[Import a repository.](#)

Owner *



hpoyatos ▾



Repository name *

testeGitHub



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

Description (optional)



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.



Git Hub – Hands-on

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.](#)

.gitignore template: None ▼

.gitignore template X

py

Python

... your code. [Learn more.](#)

① You are creating a public repository.



Git Hub – Hands-on

Add .gitignore

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

.gitignore template: Python ▾

Choose a license

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

License: None ▾

License

×

Filter...

✓ None

Apache License 2.0

GNU General Public License v3.0

MIT License

BSD 2-Clause "Simplified" License

BSD 3-Clause "New" or "Revised" License

Boost Software License 1.0

Creative Commons Zero v1.0 Universal

Eclipse Public License 2.0

GNU Affero General Public License v3.0

GNU General Public License v2.0


GNU Lesser General Public License v2.1

© 2022 GitHub, Inc.

[Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#)






Git Hub – Hands-on

 hpoyatos / testeGitHub Public


[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

[main](#) [1 branch](#) [0 tags](#) [Go to file](#) [Add file](#) [Code](#)

 hpoyatos Initial commit 420a668 now 1 commit

 .gitignore	Initial commit	now
 README.md	Initial commit	now

README.md



testeGitHub



Criar um repositório remoto

Crie um repositório no GitHub;

Configure o repositório remoto;

Crie um arquivo README.md;

Faça o upload das alterações;

```
git remote add <remote> <url>
```

Geralmente o repositório remoto é chamado de “origin”, mas você o batiza como quiser!



Comandos para sincronizar com o repositório remoto

Para baixar do repositório remoto, digite:

```
git pull <remote> <branch>
```

Para submeter ao repositório remoto, digite:

```
git push <remote> <branch>
```

