

# CURSO DE VERÃO

---

## Sistema de Controle de Versão usando GIT com GitHub

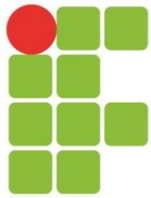
- Prof. Dra. Paloma Oliveira
- Email: [paloma.oliveira@ifmg.edu.br](mailto:paloma.oliveira@ifmg.edu.br)



**GitHub**



- O Github é um serviço de hospedagem distribuído desenvolvido em Ruby on Rails para projetos que utilizam o controle de versão Git.
- É utilizado como repositório online de códigos fonte para projetos de código aberto.
- Site oficial: <http://github.com>



INSTITUTO  
FEDERAL  
MINAS GERAIS  
Campus  
Formiga

# Criar um usuário



The image is a mockup of the GitHub sign-up page. On the left, the GitHub Octocat mascot is visible. The main heading reads 'How people build software'. Below it, a paragraph states: 'Millions of developers use GitHub to build personal projects, support their businesses, and work together on open source technologies.' On the right side, there is a sign-up form with three input fields. The first field contains 'palomafga', the second contains 'palomafga@gmail.com', and the third contains seven dots representing a password. Below the password field, a small text note says: 'Use at least one letter, one numeral, and seven characters.' A green button labeled 'Sign up for GitHub' is positioned below the form. At the bottom right, a disclaimer states: 'By clicking "Sign up for GitHub", you agree to our **terms of service** and **privacy policy**. We'll occasionally send you account related emails.'

## How people build software

Millions of developers use GitHub to build personal projects, support their businesses, and work together on open source technologies.

palomafga

palomafga@gmail.com

.....

Use at least one letter, one numeral, and seven characters.

**Sign up for GitHub**

By clicking "Sign up for GitHub", you agree to our **terms of service** and **privacy policy**. We'll occasionally send you account related emails.

# Criar um usuário

## Welcome to GitHub

You've taken your first step into a larger world, @palomafga.



Completed  
Set up a personal account



Step 2:  
Choose your plan



Step 3:  
Tailor your experience

### Choose your personal plan

- ☒ Unlimited public repositories for free.
- ☐ Unlimited private repositories for \$7/month. ([view in BRL](#))

Don't worry, you can cancel or upgrade at any time.

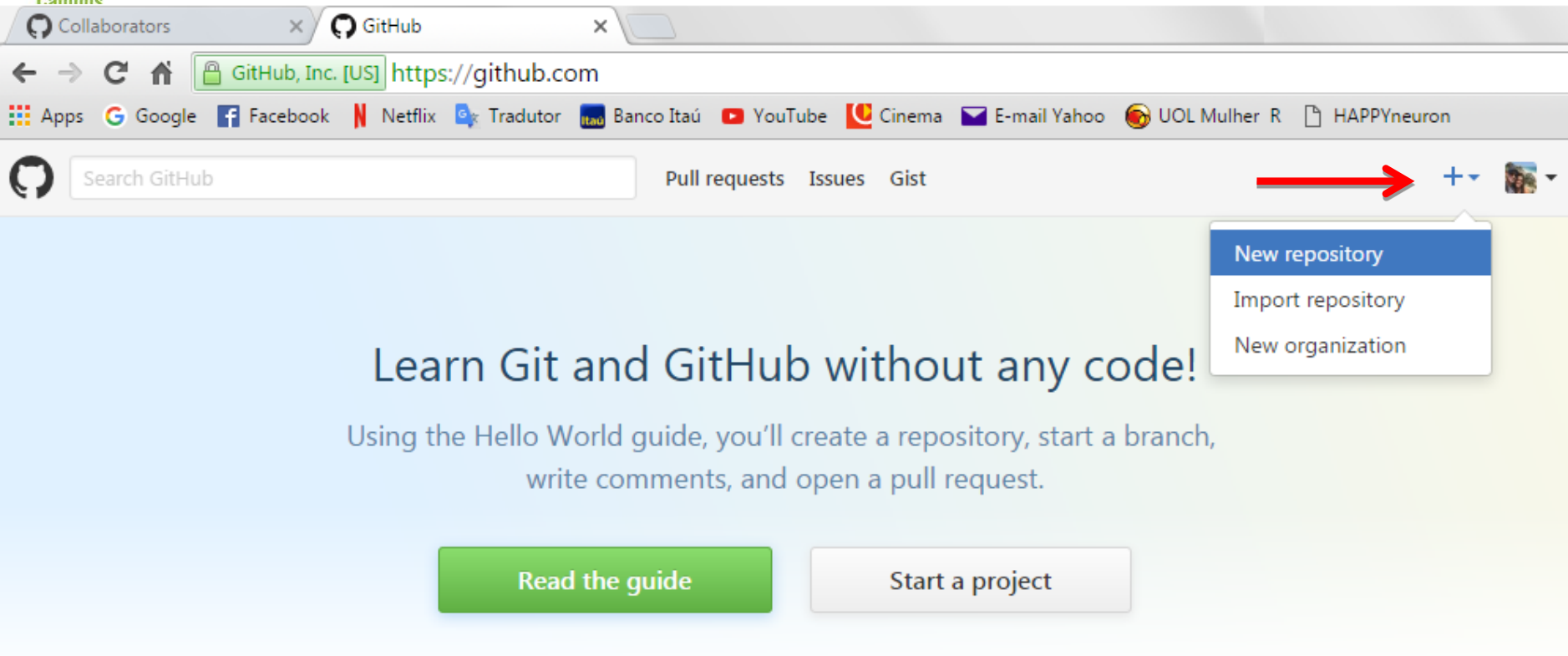
- ☐ **Help me set up an organization next**  
Organizations are separate from personal accounts and are best suited for businesses who need to manage permissions for many employees.  
[Learn more about organizations.](#)

### Both plans include:

- ✓ Collaborative code review
- ✓ Issue tracking
- ✓ Open source community
- ✓ Unlimited public repositories
- ✓ Join any organization

Continue

# Criando Repositório



The screenshot shows the GitHub homepage in a web browser. The browser's address bar displays "https://github.com". The page features a navigation bar with links for "Collaborators", "GitHub", "Pull requests", "Issues", and "Gist". A search bar is located on the left, and a user profile icon is on the right. A red arrow points to the "+" button next to the profile icon. A dropdown menu is open, showing three options: "New repository", "Import repository", and "New organization". The main content area has the heading "Learn Git and GitHub without any code!" and a subheading "Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request." Below this are two buttons: "Read the guide" (green) and "Start a project" (grey).

# Confirmar email

[Personal](#)[Open source](#)[Business](#)[Explore](#)[Pricing](#)[Blog](#)[Support](#)[Your dashboard](#)

## Please verify your email address

Before you can contribute on GitHub, we need you to verify your email address.  
An email containing verification instructions was sent to **palomafga@gmail.com**.

Didn't get the email? [Resend verification email](#) or [change your email settings](#).

# Create a new repository

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

---

Owner

Repository name



palomaoliveira ▾



HelloWorld

Great repository names are short and memorable. Need inspiration? How about **animated-couscous**.

Description (optional)

Meu primeiro repositório no GITHUB

---



**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**

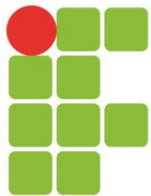
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▾

Add a license: **None** ▾



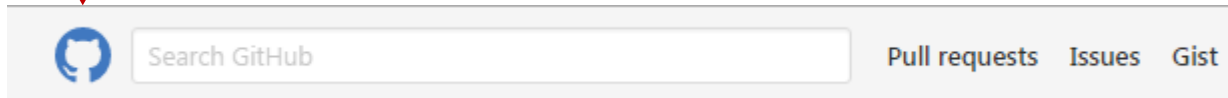
Create repository



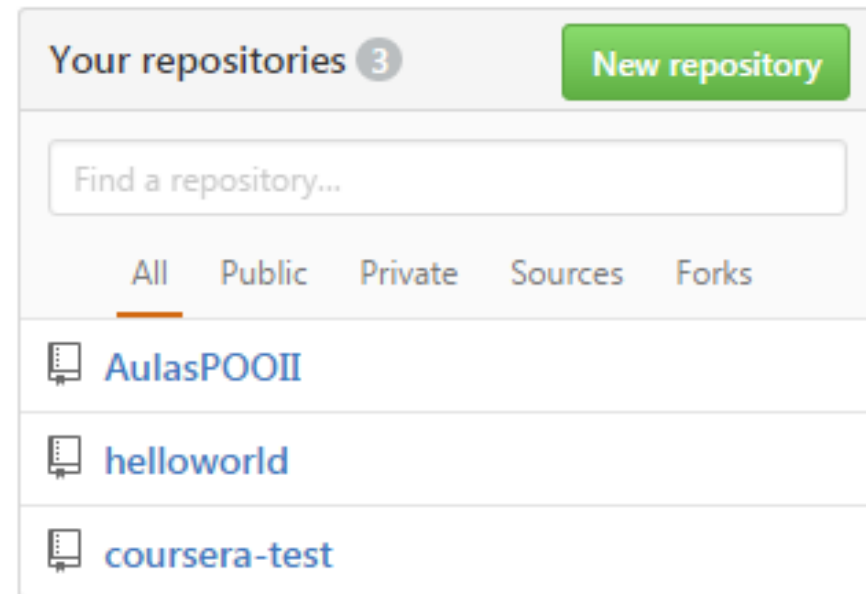
INSTITUTO  
FEDERAL  
MINAS GERAIS  
Campus  
Formiga

# Criando Repositório

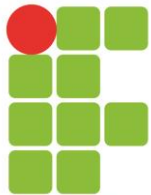
Visualizar sua página inicial



Lista de repositórios



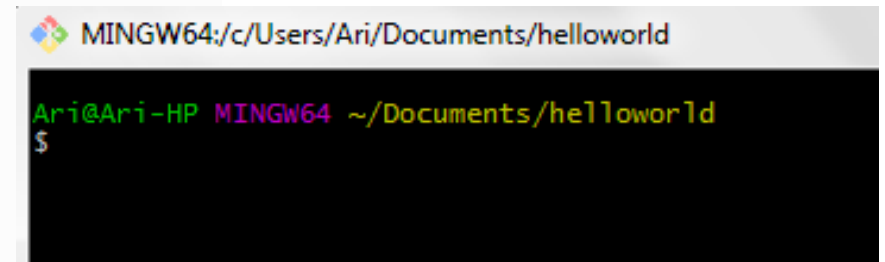
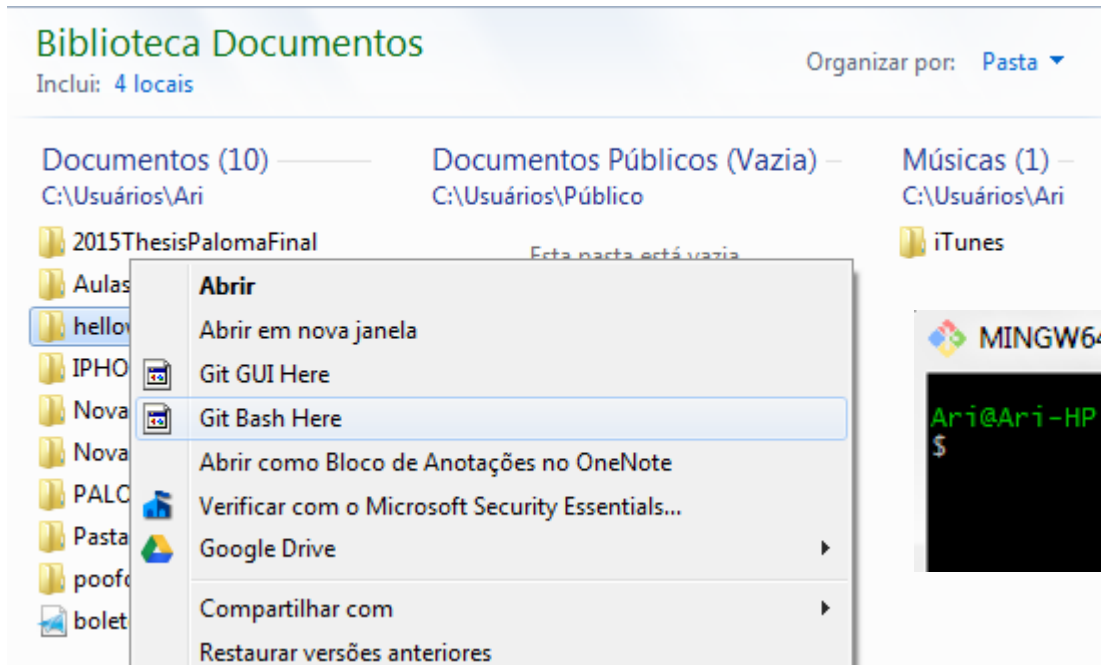




INSTITUTO  
FEDERAL  
MINAS GERAIS  
Campus  
Formiga

# Repo local

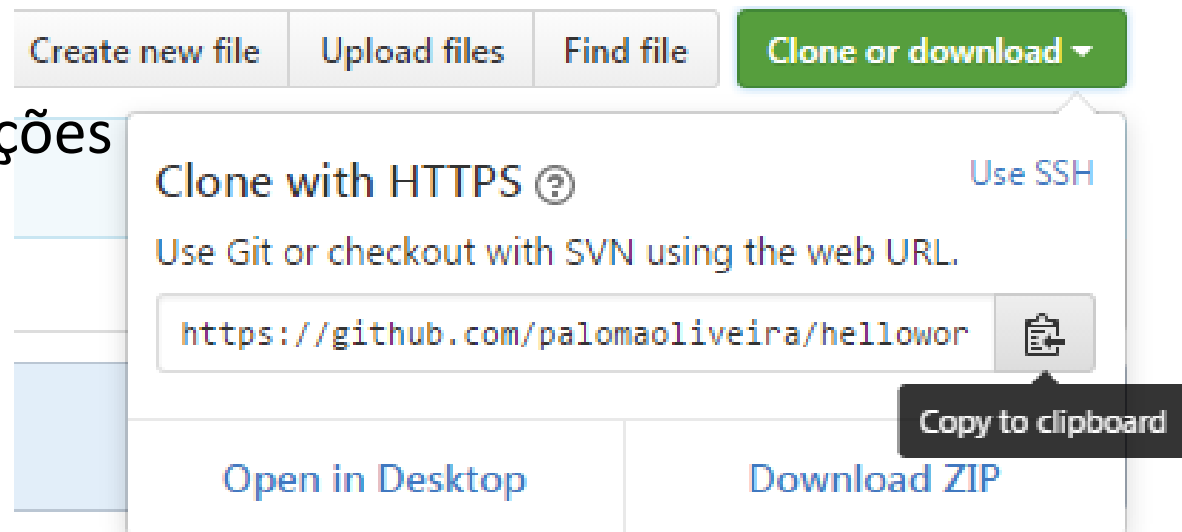
1. Criar um diretório com o nome do projeto
2. Iniciar o Git bash



# Repo local - configurando

3. Iniciar o controle de versão **git init**
4. Adicionar o repositório remoto
  - Git add
  - **git remote add origin <repositorio git>**
  - <repositorio git> endereço do repositório GitHub

- Para ver as configurações
- **\$ vim .git/config**



# git pull

---

## 5. Realiza o primeiro backup do repositório – pull

- Comando para buscar novas atualizações do servidor

• **git pull origin master**

- Em repositórios compartilhados é sempre bom atualizar seu repositório para depois enviar as modificações para o servidor

# Exercício

---

- Utilize o repositório local criado na aula anterior, iremos enviar esses arquivos para o GitHub
- Va no local do diretório e com o botão direito do mouse Git Bash here

# git push

---

- Comando git push
  - O comando **git push** empurra as suas modificações para o servidor, incluindo-as no histórico do projeto.
  - Quando os outros integrantes da equipe fizerem um **git pull**, essas modificações serão baixadas e incluídas no repositório local da pessoa

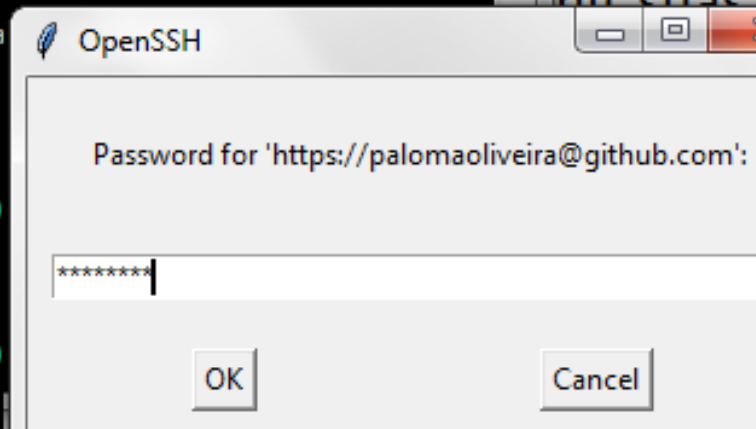
# git push

- **git push origin master**
- Será solicitado seu username e senha do GitHub

```
remote: Counting objects: 13, done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 13 (delta 1), reused 9 (delta 0), pa
Unpacking objects: 100% (13/13), done.
From https://github.com/palomaoliveira/helloworld
 * branch          master    -> FETCH_HEAD
 * [new branch]     master    -> origin/master

Ari@Ari-HP MINGW64 ~/Documents/helloworld (master)
$ git status
On branch master
nothing to commit, working tree clean

Ari@Ari-HP MINGW64 ~/Documents/helloworld (master)
$ git push origin master
Username for 'https://github.com': palomaoliveira
```



- Verifique seu repositório no GitHub

# Exercício - push

---

- Apague o que você digitou no arquivo .txt e crie 3 novas linhas.
- Salve o arquivo
- **git add .**
- **git status**
- Commit o arquivo: **git commit -m "commit repo remote"**
- **git push origin master**
- Verifique no GitHub as mudanças...veja como fica fácil controlar as alterações.
- Crie um novo arquivo e disponibilize ele no repositório;

# Arquivos no repositório

---

- Perceba que apenas o branch master vai para o GitHub
- Para enviar branches para o repositório:
  - Entre no branch local: **git checkout funcionalidade1**
- Enviar o branch para o repositório remoto
  - **git push origin funcionalidade1**
- Cria um novo branch no repositório remoto
- Veja a mudança no GitHub



# Clonando o repositório

---

- **git clone <url do seu repositório> repo2** – repo2 é o nome do seu diretório no seu repositório local
- Acesse o diretório, veja que todos os seus arquivos estão lá conforme no servidor remoto
- Perceba que o clone é apenas do branch master
- **git branch** – mostra branches locais (no seu comp)
- **git branch -a** – mostra branches locais e remoto

# Clonando branch

---

- **git branch -b <nome do novo branch> <nome do branch no GitHub>**
- **Ls** – para ver os arquivos do seu branch
- **git pull** – sincroniza todos os arquivos, ou seja, busca por novas atualizações

# Exercício push e pull

---

- Coloque seus dois repositórios em um novo diretório chamado gitcode
- Insira uma nova linha no arquivo teste2.txt
- Adicione o arquivo no seu repositório e comit a modificação
- Em seguida envia para o servidor
- **git push origin master**

# Exercício push e pull

---

- Abra seu arquivo teste2 no repositório RepoGit01
- Ele foi modificado? Não, porque?
- Precisamos puxar as alterações do servidor
- **git pull**
- E agora?

# Colaboração

---

- **Shared repository:**

- todos os autores tem permissão para alterar (add, push, commit, pull, criar *branches*, ...) o repositório principal.
- Mais utilizado para projetos com poucos colaboradores.

# Colaboração

---

- Fork & Pull:

- todos os colaboradores trabalham em *forks* do projeto original.
- Para incorporar o trabalho é necessário enviar um *pull request* para o mantenedor do projeto
  - mantenedor decide se junta (merge) ou não as modificações.
- Mais utilizado quando há muitos colaboradores.

# Evitando conflitos

---

# Referências

---

- <http://culturadigital.br/teceduca/files/2015/05/tutorial-github.pdf>
- <http://www.defensoria.pi.gov.br/gestor/public/uploader/documentos/4/84a0bcaa34768dfdbc090b3caab1b660.pdf>
- <http://www.devmedia.com.br/usando-o-github-e-o-netbeans/24603>