



# Engenharia de Software para Nuvem - Aula 1

Jemerson Fernando Maia - [jfnandopr@gmail.com](mailto:jfnandopr@gmail.com)

Pós-graduação em Engenharia de Software para Modernização de Sistemas

BIOPARK EDUCAÇÃO

---

# Agenda

- Introdução a Cloud Computing, Containers, CI/CD, IaC, DEVOPS
- Git e GitHub
- Containers



---

# Computação em Nuvem (Cloud Computing)

# Infraestrutura de TI

- Refere-se aos componentes necessários para executar e gerenciar ambientes de TI empresarial.
  - hardware
  - **software**
  - rede
  - sistema operacional
  - armazenamento de dados





## Infraestrutura

A infraestrutura de TI pode ser implantada nas próprias instalações da organização ou em cloud computing (nuvem).

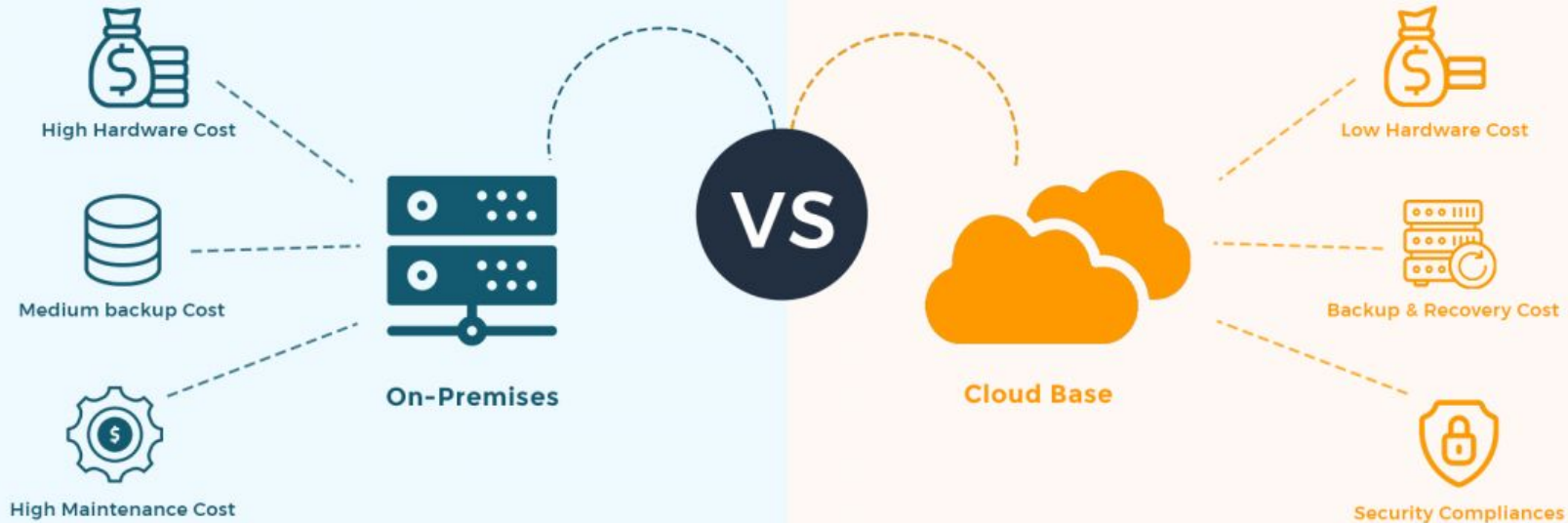


## Cloud Computing

É a entrega de recursos de TI sob demanda por meio da Internet com definição de preço de pagamento conforme o uso.

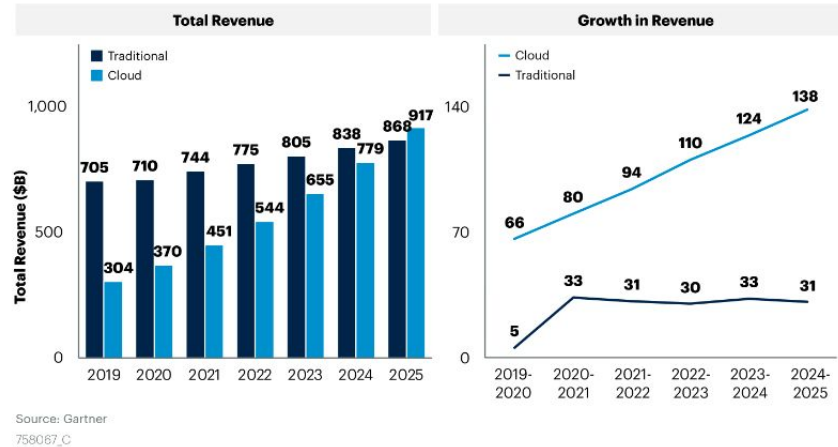


## On-Premises VS Cloud Base



# Cloud Computing

Figure 1: Sizing Cloud Shift, Worldwide, 2019 – 2025



Gartner

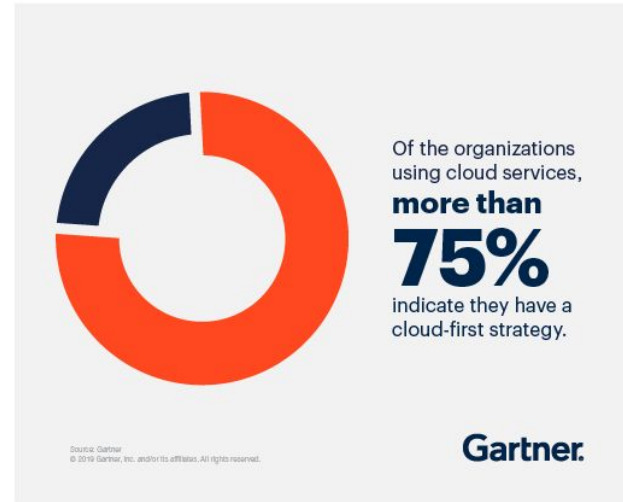
Source: Gartner, February 2022



# Cloud Computing

## Cloud computing is the new norm

Cloud computing continues to evolve from a market disruptor to the expected approach for IT. Although cloud computing has become a foundation of digital business, many organizations still struggle to optimize this powerful tool. Gartner estimates that less than one-third of enterprises have a documented cloud strategy.

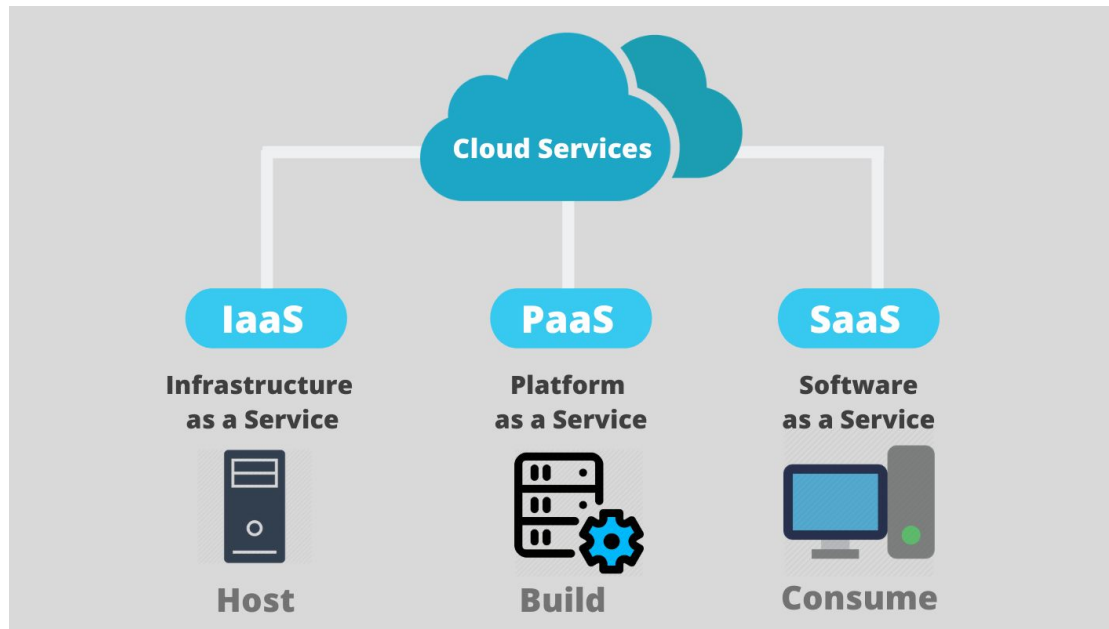




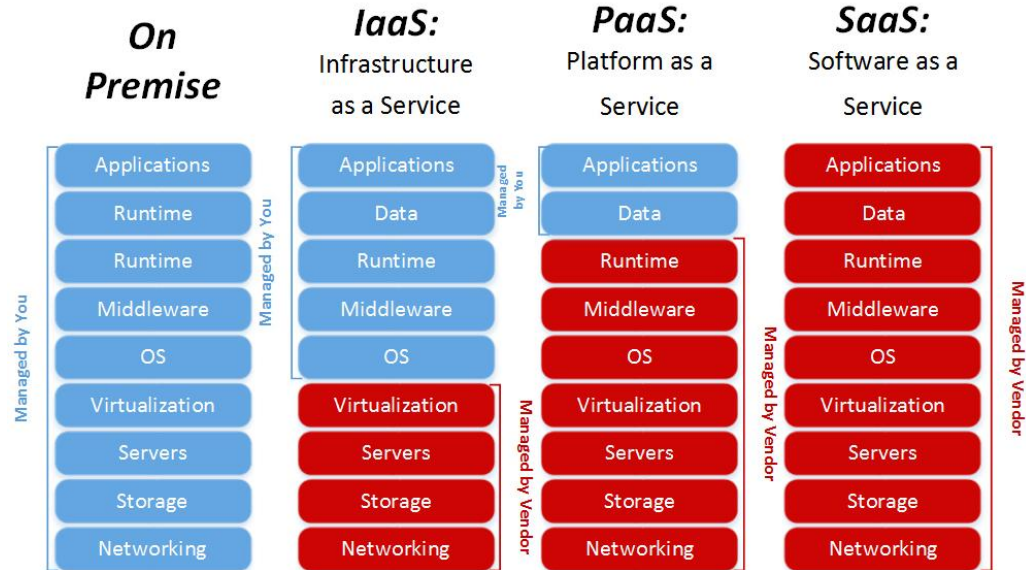
## Causas do crescimento

- oferece maior escalabilidade e agilidade, permitindo que as empresas respondam rapidamente às mudanças no mercado e nas necessidades dos clientes
- adoção de tecnologias de inteligência artificial, analytics e internet das coisas
- arquitetura de softwares robustas e flexíveis
- o aumento do uso de dispositivos e a necessidade de armazenar e processar grandes dados em tempo real
- pandemia (a necessidade de colaboradores remotos acessem serviços de alta performance)

# Modelos de Serviços em Nuvem



# Modelos de Serviços em Nuvem





# Tipos de Implantação



## Nuvem Privada

Utilizada por uma única organização, com dados particulares do negócio.

Seus serviços e infraestrutura se encontram em um centro de dados privado.



## Nuvem Híbrida

Combinação entre nuvem privada e nuvem pública, com serviços geralmente integrados.



## Nuvem Pública

Compartilhável entre várias organizações com um provedor que gerencia o software

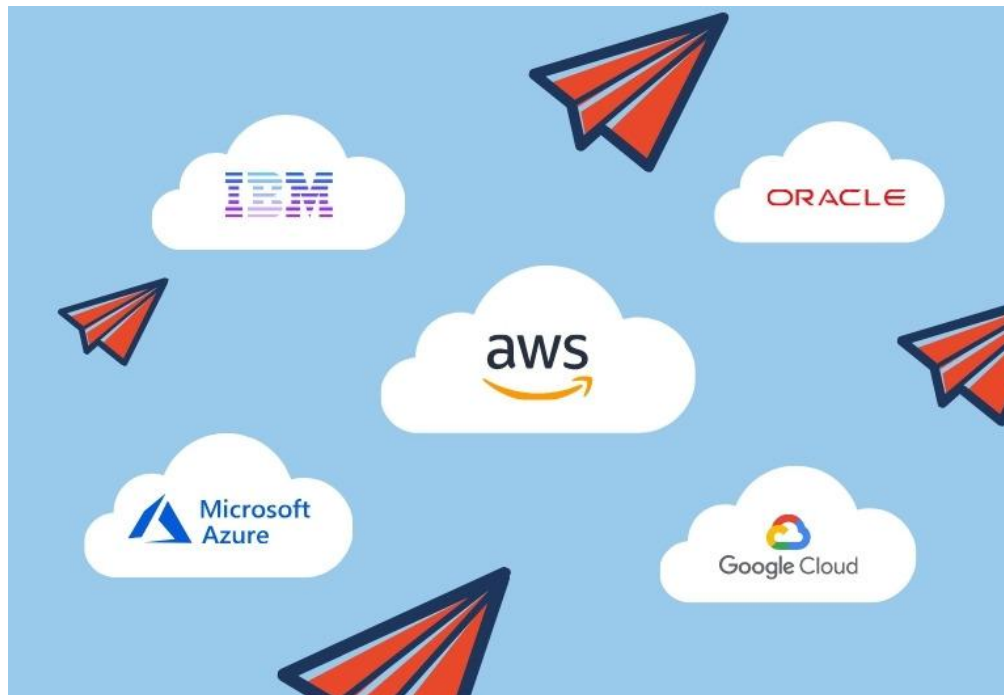


## Vantagens de Utilização

- Uma facilidade muito maior de compartilhar os dados
- Diminuição da sobrecarga de infraestrutura
- Qualidade de acesso aos usuários
- Poder acessá-lo de qualquer lugar
- Otimização do tempo
- Não há inatividade
- Estabilidade no serviço
- O pagamento somente por aquilo que você utiliza
- Tomada de decisão mais ágil
- Serviço com maior durabilidade e menor custo

---

## Alguns Fornecedores





**Como executar minha aplicação em nuvem?**

**Existem várias formas, mas algo que veio para ajudar foram os ...**





# Containers



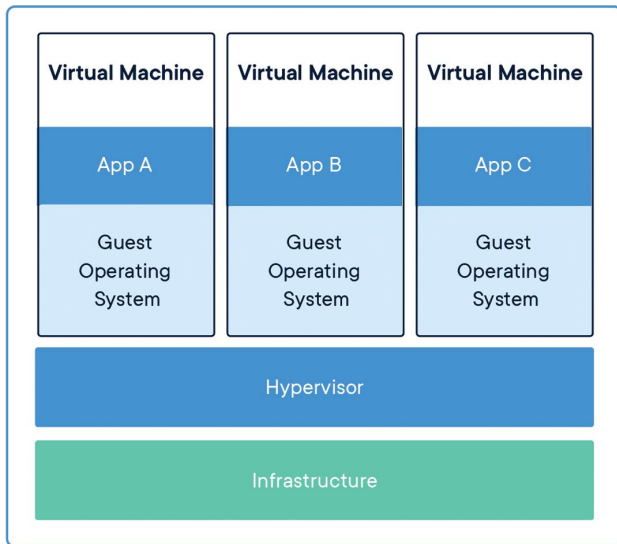
## O que é?

Containers são como se fossem máquinas virtuais modulares e extremamente leves.

É como um ambiente isolado, disposto em um servidor, que divide um único host de controle.

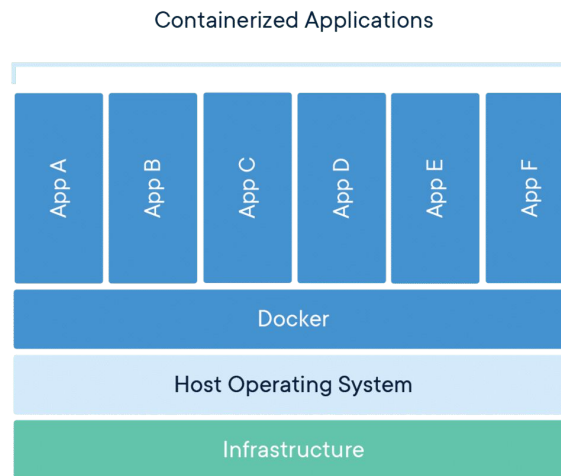
Funcionam um pouco como VMs, mas de uma maneira muito mais específica e granular.

# Virtualização



X

# Containerização





## Containers

A containerização é um conceito antigo no mundo da computação (1979) que ganhou destaque quando o Docker introduziu ferramentas simplificadas para criar e gerenciar contêineres.

Docker é uma tecnologia de containerização que permite a criação e o uso de containers Linux

A close-up shot of a young boy with brown hair, looking up at an adult whose face is partially visible in the upper left corner. The boy has a curious or hopeful expression.

**IT WORKS ON MY MACHINE**

A close-up shot of Johnny Depp as Captain James Norrington, looking down with a serious and somewhat weary expression. The back of a child's head is visible in the foreground on the right.

**THEN WE'LL SHIP YOUR MACHINE**

A wide shot of a man in a dark suit and a young boy sitting on a park bench. The man is leaning forward, resting his head on the boy's shoulder in a moment of emotional connection. The background shows a park with trees and a path.

**AND THAT IS HOW DOCKER WAS BORN**



## Container (Problema)

- O aplicativo que não é executado corretamente quando migrado de um ambiente para outro
  - Normalmente, esses problemas surgem por causa de diferenças na configuração de biblioteca e outras dependências.
  - Os contêineres resolvem esse problema ao fornecer uma infraestrutura leve e imutável para o empacotamento e a implantação de aplicativos.



## Cloud Container

Cloud Container trata-se de um ambiente onde há todas as configurações e compartimentos necessários para executar aplicações.

Quando opta-se por usar o Cloud Container, deixa-se de ter a necessidade de administrar um sistema operacional.



## Benefícios

- Agilidade
- Portabilidade
- Escalabilidade
- Economia de recursos
- Disponibilidade

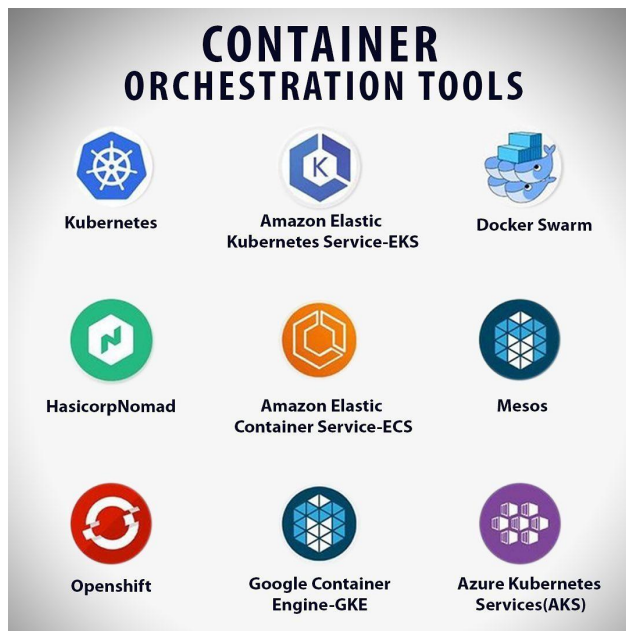




## Orquestração de Containers

- A orquestração automatiza a implantação, o gerenciamento, a escala e a rede dos containers.
  - Cuida do ciclo de vida dos containers de forma autônoma, subindo e distribuindo, conforme nossas especificações ou demandas

# Ferramentas de Orquestração de Containers





**Como implantar minha aplicação em nuvem?**

---

CI/CD



## O que é

Continuous Integration (CI)

Continuous Delivery (CD) ou

Continuous Deployment

CI/CD é um método para entregar aplicações com frequência aos clientes.

Prevê a implementação da automação nas etapas de desenvolvimento de aplicações.

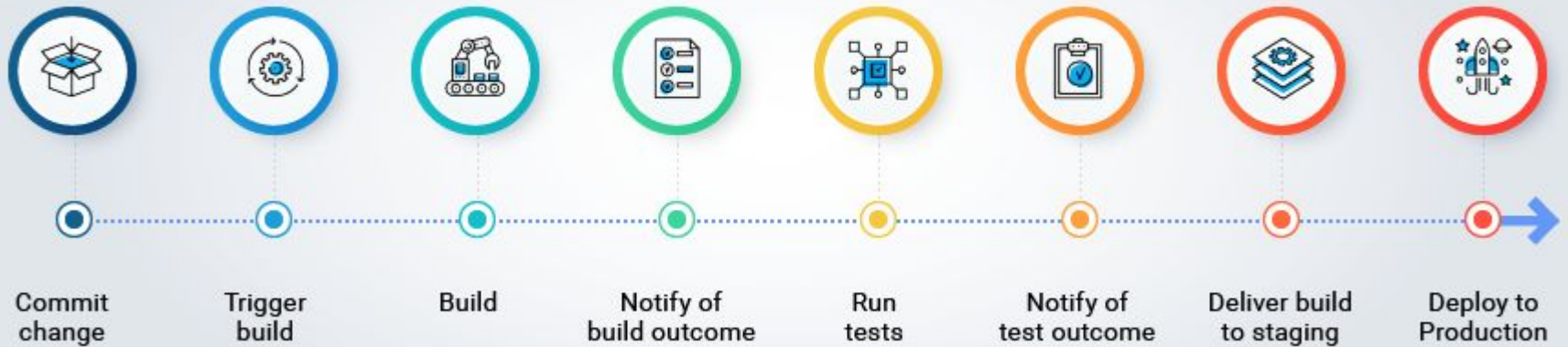


## O que é

- CI
  - O objetivo técnico é estabelecer uma forma consistente e automatizada de criar, empacotar e testar aplicativos.
- CD
  - Automatiza a entrega de aplicativos para ambientes locais selecionados (repositório ou infraestrutura).



## CI/CD PIPELINE





**E aplicações que não rodam em containers, como  
criar e manter os servidores em nuvem?**



---

# Provisionamento & Gerenciamento de Configuração Automatizados



# Provisionamento

Provisionamento é o processo de criação e configuração da infraestrutura de TI.

O provisionamento é um dos primeiros estágios da implantação de servidores, aplicações, componentes de rede, armazenamento, entre outros.

Necessário ser realizado tanto em uma infraestrutura on-premise quanto em nuvem.



## Provisionamento Automatizado

Atualmente, a maioria das tarefas de provisionamento já é executada com facilidade pela automação, usando **infraestrutura como código** (IaC).

A IaC é uma forma de gerenciamento de configuração que codifica os recursos de infraestrutura em arquivos de dados legíveis por máquina e humanos, como o YAML.

Esses arquivos de dados são então lidos e a infraestrutura é provisionada.



# Gerenciamento de Configuração

Gerenciamento de configuração é o processo usado para manter sistemas computacionais, servidores e softwares em um estado desejado, consistentemente.

É uma forma de se certificar de que o sistema funciona como o esperado, ainda que alterações sejam feitas com o passar do tempo.

No desenvolvimento de software, o gerenciamento de configuração é muitas vezes usado junto com o controle de versão e a infraestrutura de CI/CD.



# Gerenciamento de Configuração Automatizadas

Tradicionalmente, isso é feito de forma manual ou por meio de scripts personalizados criados por administradores de sistemas.

A automação do gerenciamento de configurações tem como objetivo reduzir os custos, os erros e a complexidade.



## Benefícios do Provisionamento e Gerenciamento de Configuração Automatizados

- Gerenciamento de várias VM ou nós de execução de software de maneira fácil e organizada
- Rastreabilidade (o que, quando e quem mudou)
- Evitar erros humanos
- Mais segurança
- Rapidez
- Escalabilidade



## Provisionamento e Gerenciamento de Configuração de CI/CD

O processo de CI/CD utiliza fluxos de trabalho de revisão de código baseados em solicitações pull para automatizar a implementação de alterações de código em um sistema de software ativo. Este mesmo fluxo pode ser aplicado às alterações de configuração.

- A CI/CD pode ser configurada para que as solicitações de alteração de configuração aprovadas possam ser imediatamente implementadas em um sistema em execução, bem como o provisionamento e configuração de ambientes de QA e Produção. (**GitOps**)

## Ferramentas para Provisionamento e Gerenciamento de Configuração de Infraestrutura



**'SALTSTACK**







DevOps

## O conflito



o problema não  
são as máquinas,  
é o código

o problema não é  
o código, são as  
máquinas



## O que é?

A combinação de filosofias culturais, práticas e ferramentas que aumentam a capacidade de uma empresa de distribuir aplicativos e serviços em alta velocidade.

É uma mudança cultural em que as equipes adotam uma cultura de engenharia de software, fluxo de trabalho e conjunto de ferramentas que **elevam os requisitos operacionais ao mesmo nível de importância que a arquitetura, design e desenvolvimento.**

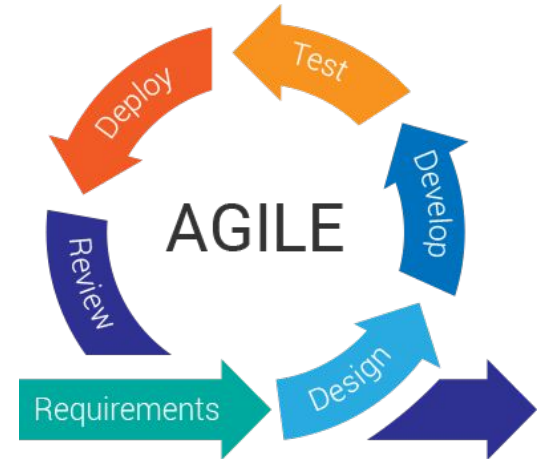


## E o DevSecOps?

O DevSecOps incorpora automaticamente a segurança em todas as fases do ciclo de vida de desenvolvimento de software, permitindo o desenvolvimento de software seguro na velocidade do Agile e do DevOps.

## Como tudo começou - Agile

- Indivíduos e interações, mais que processos e ferramentas
- Software em funcionamento, mais que documentação abrangente
- Responder a mudanças, mais que seguir um plano





## Como tudo começou - Agile

- Apesar do surgimento da metodologia ágil, as equipes de desenvolvimento e operações permaneceram separadas.
- O movimento do DevOps começou a tomar forma entre 2007 e 2008
- Equipes de desenvolvimento e operações se uniram para solucionar disfunções no setor
- O conceito surgiu após a disseminação de concepções como entrega contínua, desenvolvimento ágil e deploy contínuo (presentes no Manifesto Ágil, de 2001)



## Como tudo começou - Agile

- Agile Conference - 2008
  - Infraestrutura Ágil de Andrew Schafer para Patrick Debois
- Velocity Conference da O'Reilly - 2009
  - John Allspaw e Paul Hammond apresentaram a palestra chamada “10+ Deploys per Day: Dev and Ops Cooperation at Flickr”
- DevOpsDays - 2009
  - Patrick Debois decidiu criar sua própria conferência na Bélgica
- Gartner -2011
  - Publicou um relatório no qual afirmava que, até o final de 2015, DevOps se tornaria a principal estratégia em 20% das organizações mundiais.

# Como tudo começou - Agile

 devopsdays EVENTS BLOG SPONSOR SPEAKING ORGANIZING ABOUT

ALL EVENTS

2023

JUNE

Jun 21 - 23: Amsterdam

AUGUST

Aug 1 - 2: Seattle

Aug 8: Global Organizer Summit

Aug 9 - 10: Chicago

Aug 11 - 12: Beijing

Aug 19: Rio De Janeiro

Aug 29 - 30: Dallas

SEPTEMBER

Sep 2: Fortaleza

Sep 6 - 7: Viníus

Sep 7 - 8: Des Moines

Sep 8: Almaty

Sep 12: Boise

Sep 13 - 14: Washington, D.C.

Sep 14 - 15: Ukraine

Sep 18 - 19: Atlanta

Sep 21: Tampa Bay

Sep 21 - 22: London

Sep 25 - 26: Taipei

Sep 27 - 28: Buffalo

Sep 27: Cairo

OCTOBER

Oct 5 - 6: Indianapolis

Oct 6: Madrid

Oct 11 - 12: Eindhoven

Oct 11 - 12: Oslo

Oct 16 - 17: Boston

Nov 16 - 16: Dusseldorf



JUN 21 - 23, 2023

**Amsterdam**



AUG 1 - 2, 2023

**Seattle**



AUG 8, 2023

**Global Organizer Summit**



AUG 9 - 10, 2023

**Chicago**



AUG 11 - 12, 2023

**Beijing**



AUG 19, 2023

**Rio De Janeiro**



AUG 29 - 30, 2023

**Dallas**



SEP 2, 2023

**Fortaleza**

<https://devopsdays.org/>





## C.A.M.S. ( Culture, Automation, Measure, Sharing)

- Culture
  - É preciso colaborar, compartilhar e entender a importância de manter uma relação saudável entre todas as áreas.
- Automation
  - Identificar os processos que sejam repetitivos ou que levam bastante tempo e buscar resolver o quanto antes.
- Measure
  - Deve-se medir tudo que é possível: performance, processos, interações e até mesmo pessoas. O processo de melhoria contínua é o coração do DevOps!
- Sharing
  - Ter uma boa comunicação entre as equipes, incentivar as pessoas a se comunicarem e compartilharem ideias e problemas (Blameless).



## Princípios Básicos

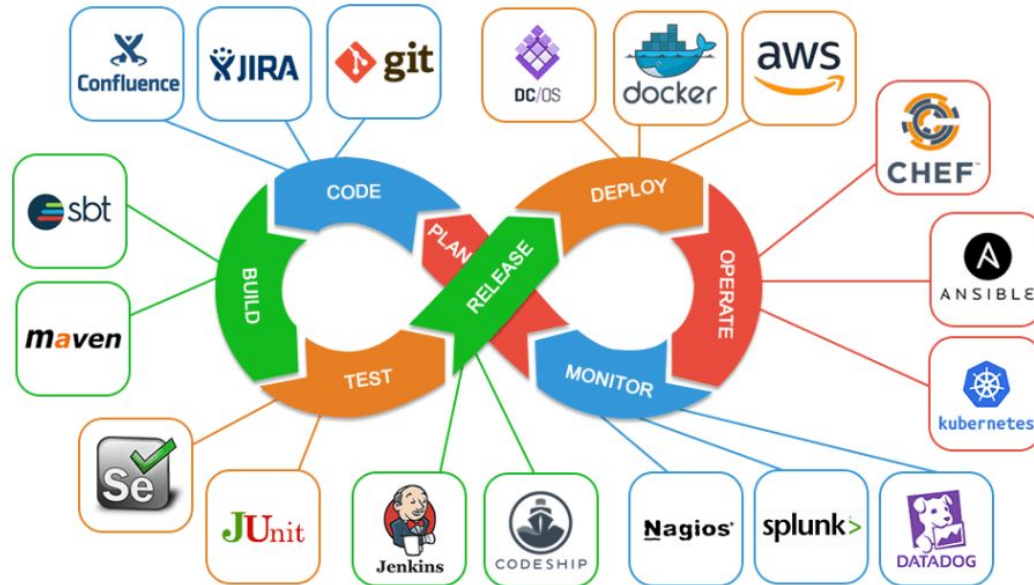
- Ação centrada ao cliente
- Foco no resultado final
- Responsabilidade de ponta a ponta
- Equipes autônomas e multifuncionais
- Melhoria Contínua
- Automatize tudo o que puder



## Habilidades principais do DevOps Engineer

- CI/CD (continuous integration CI e continuous delivery CD)
- Infraestrutura como código (IaC)
- Gerenciamento de configuração
- Contêiner e orquestração
- Gerenciamento de serviços em nuvem
- Scripts
- Monitoramento

# Pipeline DevOps



# Atividade

---



## Atividade em Grupo

Imagine que você foi contratado para implantar a cultura DevOps em uma empresa. Pense e descreva ações e ferramentas para ajudar a melhorar:

- Comunicação
- Automação de processos
- Medir (processos ou sistema) e atuar sobre

Cada grupo apresentará suas soluções e discutiremos sobre



## Controle de Fontes

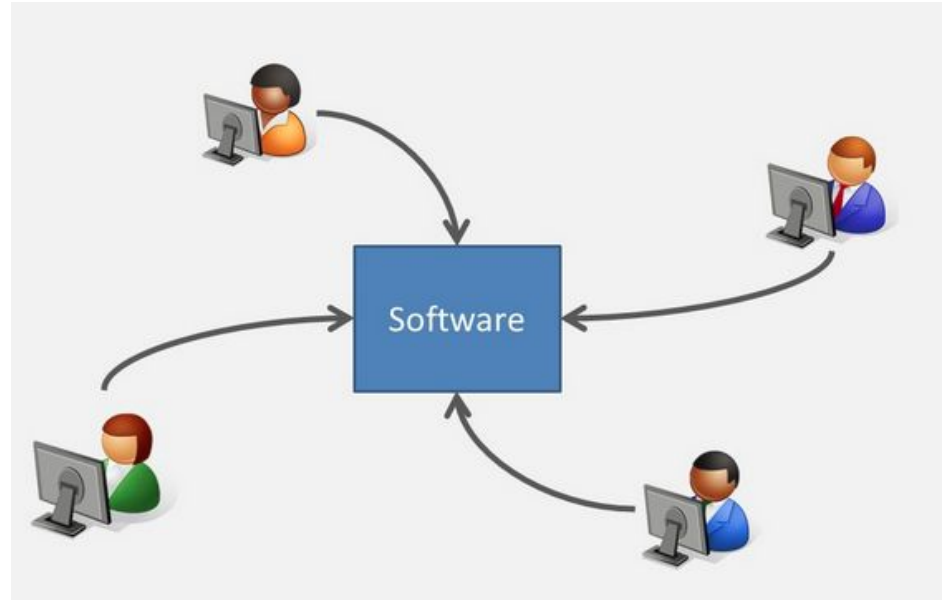
Permite acompanhar as mudanças realizadas no código-fonte, garantindo a integridade do projeto e facilitando a colaboração entre os membros da equipe.





# Controle de Fontes

- Benefícios
  - Histórico de alterações completo
  - Ramificação e mesclagem
  - Rastreabilidade

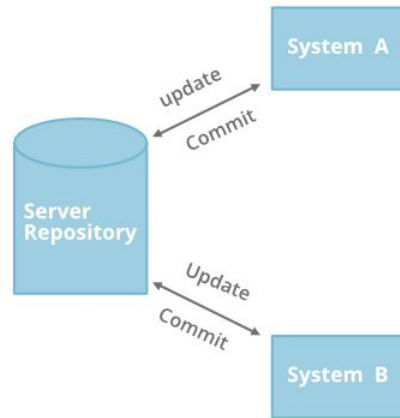


## Controle de Fontes

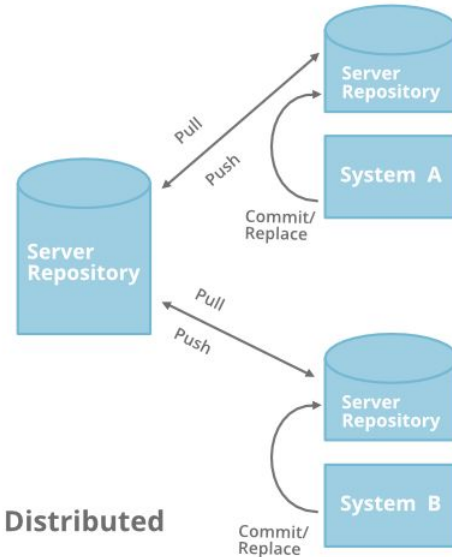


TOP VERSION CONTROL SYSTEMS

# Versionamento Centralizado X Versionamento Distribuído



Centralized



Distributed



# Git

"Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency."

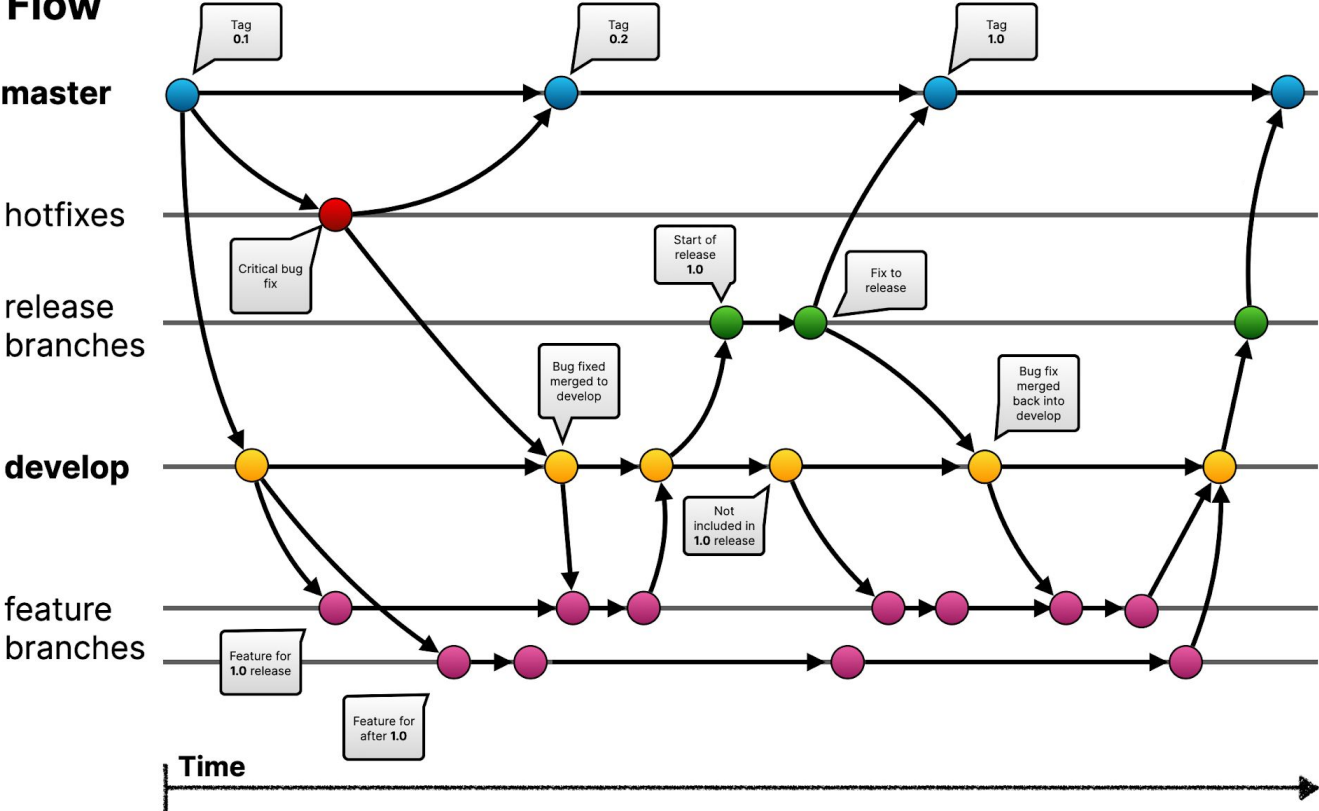
<https://git-scm.com/>



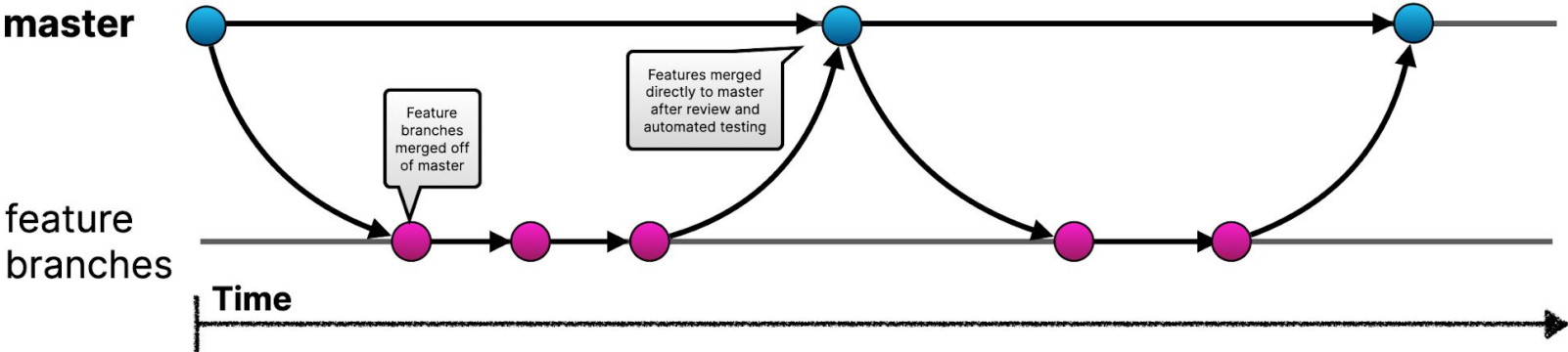
# Branches

Branch (Ramificação) significa que você ramifica a linha principal de desenvolvimento e continua a trabalhar sem alterar essa linha principal.

# Git Flow

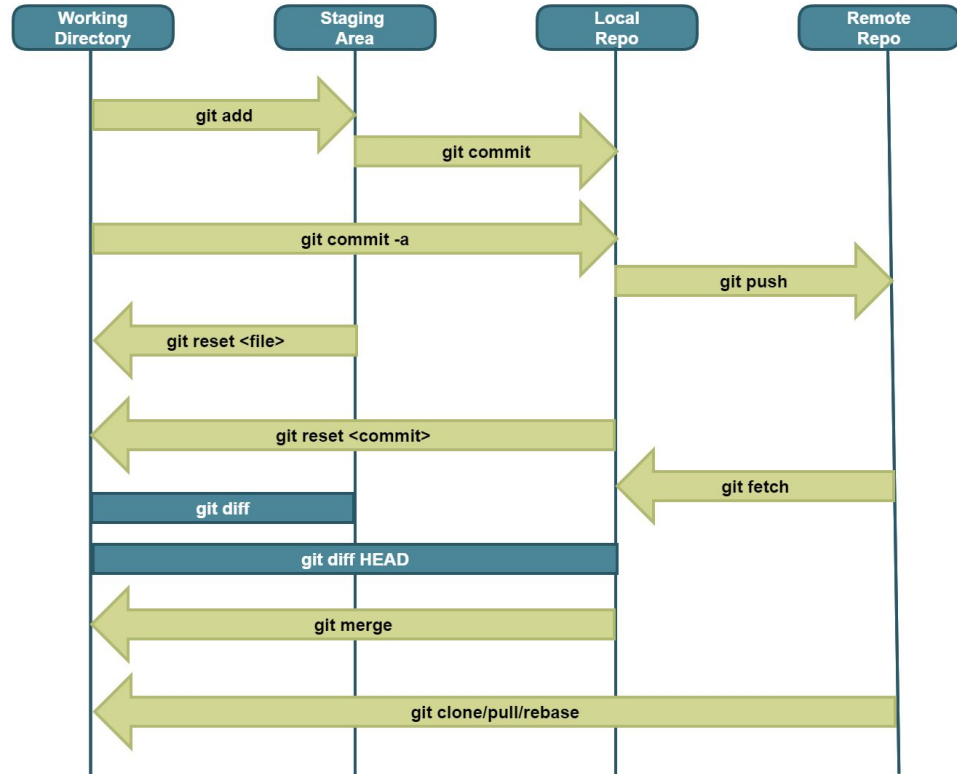


# GitHub Flow



# Comandos


## GIT MERGE





# Instalação

<https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>

 **git** --everything-is-local

Search entire site...

**About**  
**Documentation**  
Reference  
**Book**  
Videos  
External Links

**Downloads**  
**Community**

This book is available in [English](#).  
Full translation available in  
[azərbaycan dili](#),

**Chapters** ▾ 2nd Edition

## 1.5 Getting Started - Installing Git

### Installing Git

Before you start using Git, you have to make it available on your computer. Even if it's already installed, it's probably a good idea to update to the latest version. You can either install it as a package or via another installer, or download the source code and compile it yourself.

**Note**

This book was written using Git version 2. Since Git is quite excellent at preserving backwards compatibility, any recent version should work just fine. Though most of the commands we use should work even in ancient versions of Git, some of them might not or might act slightly differently.



**GitHub**



# GitHub

GitHub é uma plataforma de hospedagem de código-fonte e arquivos com controle de versão usando o Git.

Ele permite que programadores, utilitários ou qualquer usuário cadastrado na plataforma contribuam em projetos privados e/ou Open Source de qualquer lugar do mundo.

## Plataformas Git



Bitbucket



GitHub



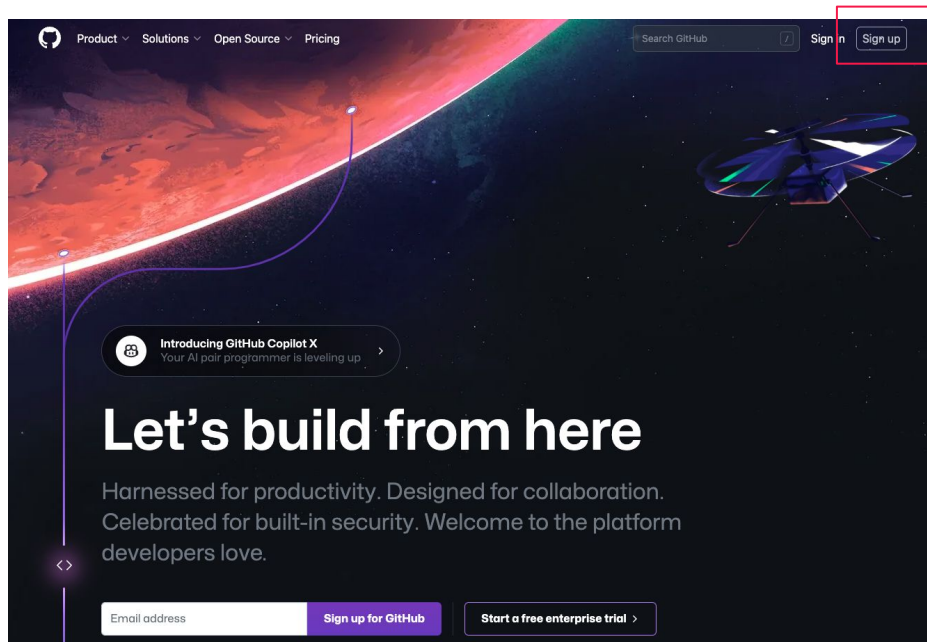
Gitlab



# Trabalhando em equipe

- Usuário/Conta
- Organizações e Repositórios
- Fluxo de desenvolvimento estabelecido
- Ramificação ou ramificações principais protegidas
- Criação de *Pull-Request*
  - Execução de actions
  - Revisão de código (*Code Review*)
- Registro de problemas (Issues) \*

# Usuário/Conta



<https://github.com/>

# Organizações



MaiaDevelopment

Organization account [Switch to another account](#)

[Go to your organization profile](#)

## Compare plans

### Free

All the basics

\$0 per user/month

Current plan

### Team

Advanced collaboration & support

\$4 per user/month

Upgrade to Team

### Enterprise

Security & flexible deployment

\$21 per user/month

Try risk-free for 30 days

Upgrade to Enterprise



## Code management

> Public repositories

Unlimited

Unlimited

Unlimited

> Private repositories

Unlimited

Unlimited

Unlimited



## Code workflow

> GitHub Codespaces

Up to 32 cores

Up to 32 cores

Up to 32 cores

> GitHub Actions

2,000 minutes/month  
Free for public repositories

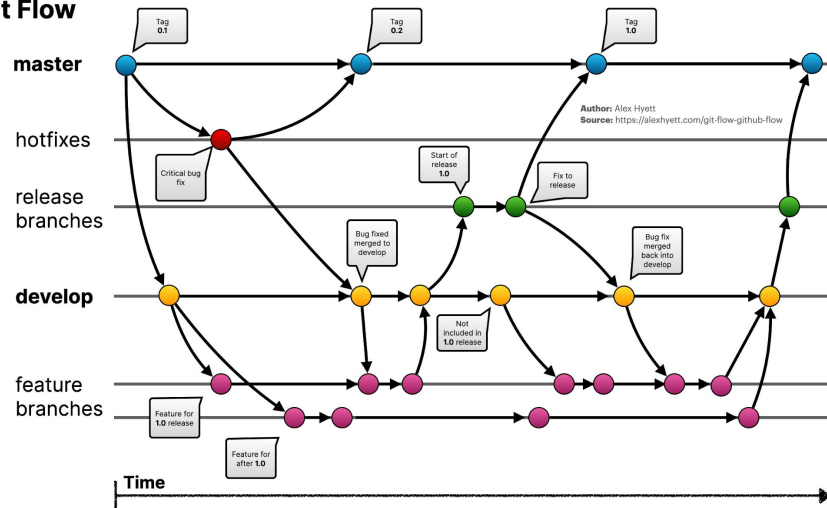
3,000 minutes/month  
Free for public repositories

50,000 minutes/month  
Free for public repositories

<https://docs.github.com/pt/get-started/learning-about-github/githubs-products>

# Fluxo de desenvolvimento

## Git Flow



<> Code Issues Pull requests Actions Projects Security Insights Settings

Search branches...

Overview

Yours

Active

Stale

All branches

New branch

### Default branch

main Updated 2 months ago by jfmandopr

Default



Your main branch isn't protected

Protect this branch from force pushing or deletion, or require status checks before merging. [Learn more](#)

Dismiss

Protect this branch



# Ramificação(ões) Protegida(s)

<> Code Issues Pull requests Actions Projects Security Insights **Settings**

General

Access

Collaborators and teams

Code and automation

**Branches**

Tags

Rules

Beta

Actions

Webhooks

Pages

## Branch protection rule



### Protect your most important branches

[Branch protection rules](#) define whether collaborators can delete or force push to the branch and set requirements for any pushes to the branch, such as passing status checks or a linear commit history.



Your rules won't be enforced on this private repository until you [upgrade this organization account to GitHub Team or Enterprise](#).

Branch name pattern \*

main

# Criação de Pull-Request

<> Code Issues **Pull requests** Actions Projects Security Insights Settings

test had recent pushes less than a minute ago

Compare & pull request

Filters is:pr is:open

Labels 9

Milestones 0

New pull request

<> Code Issues **Pull requests 1** Actions Projects Security Insights Settings

Filters is:pr is:open

Labels 9

Milestones 0

New pull request

☐ 1 Open 0 Closed

Author

Label

Projects

Milestones

Reviews

Assignee

Sort

☐ Create README.md

#1 opened now by jfnandopr

# Revisão de Código

## Create README.md #1

[Edit](#)[<> Code](#)

jfnandopr wants to merge 1 commit into [main](#) from [test](#)

Conversation 0

Commits 1

Checks 0

Files changed 1

+1 -0

Changes from all commits ▾ File filter ▾ Conversations ▾ Jump to ▾ ⚙ ▾

0 / 1 files viewed

[Review in codespace](#)

[Finish your review 1 ▾](#)

1 README.md



☐ Viewed



@@ -0,0 +1 @@

1 + # poc-terraform-aws



jfnandopr Pending

Author



Não precisa adicionar mais informações?



Reply...

# Registro de problemas (Issues)

< > Code Issues 1 Pull requests 1 Actions Projects Security Insights Settings

Filters


Labels 9

Milestones 0

New issue

☐ 1 Open ✓ 0 Closed

Author ▼ Label ▼ Projects ▼ Milestones ▼ Assignee ▼ Sort ▼

☐  **Faltando archivo README**  
#2 opened now by jfnandopr

# Hands on

---



# Hands On

- Criar conta GitHub
- Criar novo Codespace vazio
  - Configurar chave ssh
    - `ssh-keygen -t ed25519 -C "email.da@conta"`
    - `cat ~/.ssh/id_ed25519.pub`
    - Configurar em Profile -> Settings -> SSH and GPG Keys
- Clonar o repositório **biopark-nodejs-api** (`git@github.com:jfnandopr/biopark-nodejs-api.git`)
- Criar nova branch
- Realizar alterações e commits
- Criar PR
- Mesclar (merge) a PR na branch principal
- Explorar Organizações e Settings



Docker



## O que é?

É uma tecnologia de containerização que permite a criação e o uso de containers Linux

O Docker possibilita o empacotamento de uma aplicação ou ambiente inteiro dentro de um container, e a partir desse momento o ambiente inteiro torna-se portátil para qualquer outro Host que contenha o Docker instalado

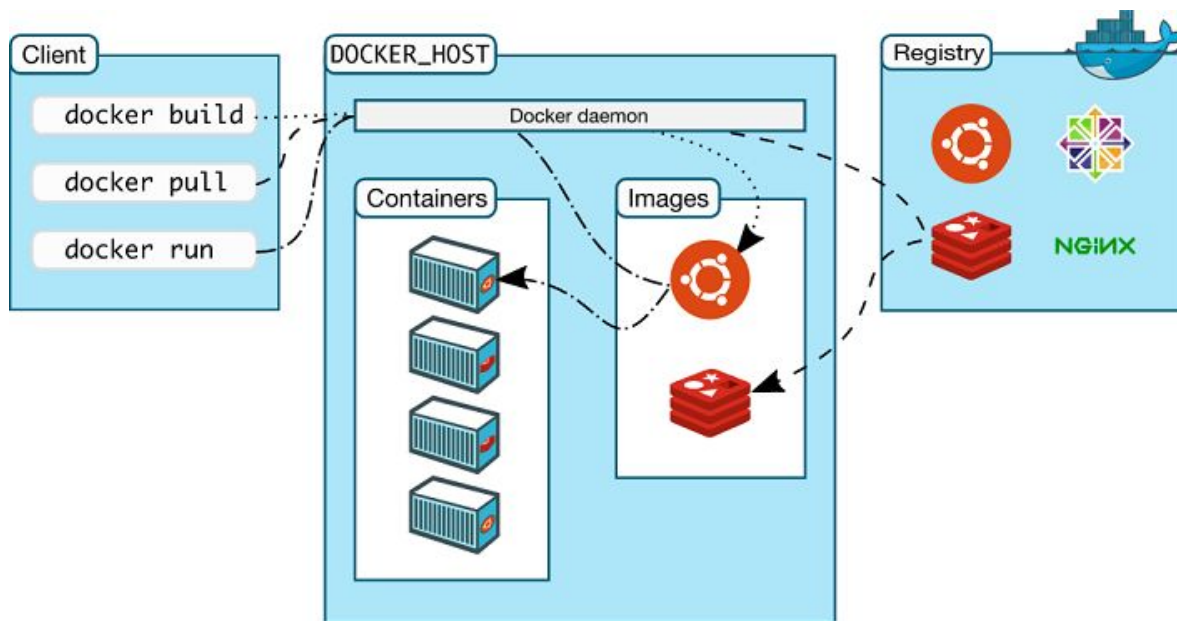


**BUT IT WORKS...**

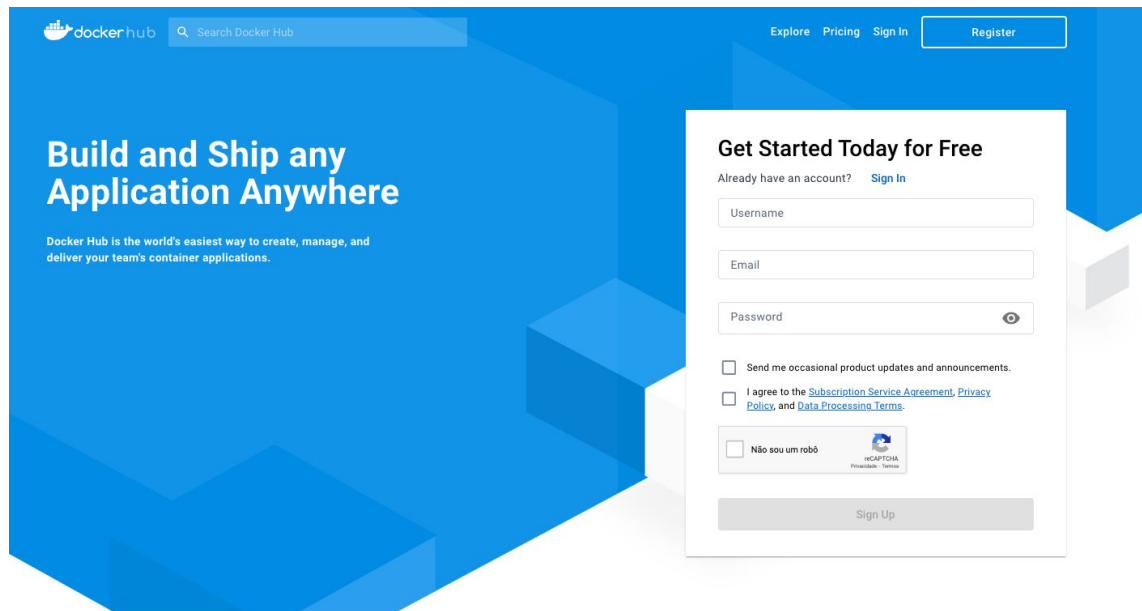


**ON MY MACHINE**

# Arquitetura



# Docker Hub

The image shows the Docker Hub website's registration page. The background is a vibrant blue with a geometric, low-poly design. On the left, the text 'Build and Ship any Application Anywhere' is prominently displayed in white. Below it, a smaller line of text states: 'Docker Hub is the world's easiest way to create, manage, and deliver your team's container applications.' The top navigation bar includes the Docker Hub logo, a search bar, and links for 'Explore', 'Pricing', 'Sign In', and a 'Register' button. A white registration form is overlaid on the right side of the page. The form has a title 'Get Started Today for Free' and a link for existing users to 'Sign In'. It contains input fields for 'Username', 'Email', and 'Password' (with a toggle for visibility). Below these are three checkboxes: one for receiving product updates, one for agreeing to the terms and privacy policy (with links), and a CAPTCHA checkbox labeled 'Não sou um robô' (I am not a robot) with a reCAPTCHA logo. A 'Sign Up' button is at the bottom of the form.

Build and Ship any Application Anywhere

Docker Hub is the world's easiest way to create, manage, and deliver your team's container applications.

Get Started Today for Free

Already have an account? [Sign In](#)


Username

Email

Password

☐ Send me occasional product updates and announcements.


☐ I agree to the [Subscription Service Agreement](#), [Privacy Policy](#), and [Data Processing Terms](#).


☐ Não sou um robô 

Sign Up

<https://hub.docker.com/>

# Instalação

 docker docs

 Search the docs

Home

Guides

Manuals

Reference

Samples

Contribute

[Home](#) / [Manuals](#) / [Docker Engine](#) / [Install](#) / [Overview](#)

Docker Desktop

Docker Extensions

Docker Engine

Overview

Install

Overview

CentOS

Debian

Fedora

RHEL

SLES

Ubuntu

Raspbian

Binaries

Post-installation steps

 **Docker Desktop for Linux**

<https://docs.docker.com/engine/install/>



## Comandos

- **docker images:** lista imagens baixadas.
- **docker search:** procura e lista imagens do docker hub.
- **docker pull:** download da imagem do docker hub.
- **docker ps:** lista containers que estão rodando.
- **docker rm:** remove um container.
- **docker rmi:** remove uma imagem.
- **docker run:** cria e inicia um container.
- **docker start/stop/restart:** inicia, para ou reinicia um container.



## Exemplo

```
$ docker run [OPTIONS] IMAGE [COMMAND] [ARG...]
```

```
$ docker run -it --name server1 ubuntu
```

```
$ docker ps -a
```

```
$ docker start server1
```

```
$ docker exec -it server1 bash
```

```
$ docker restart server1
```

```
$ docker stop server1
```

```
$ docker rm server1
```



# Network

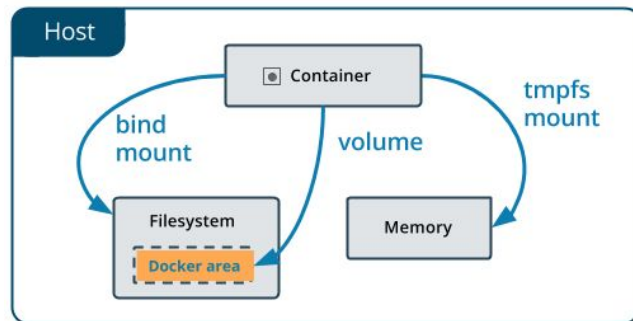
```
$ docker network ls
```

Usada principalmente para estabelecer comunicação entre os contêineres do Dock

# Volume

```
$ docker volume ls
```

Usado para manter dados persistentes, compartilhar arquivos/configurações





# Hands on

---



## Hands On

- Acessar Codespace
- Executar um container
  - `docker run -itd --name meu-so ubuntu`
- Executar os comandos
  - `docker images`
  - `docker ps -a`
  - `docker logs meu-so`
  - `docker exec -it meu-so bash`

---

# Dockerfile

## O que é?

Serve como a receita para construir um container, permitindo definir um ambiente personalizado e próprio para seu projeto pessoal ou empresarial.





# Exemplo

```
FROM ubuntu

RUN apt-get update -y; \
    apt-get install -y nginx curl; \
    rm -rf /var/lib/apt/lists/*

VOLUME ["/var/www/html", "/var/log/nginx"]

EXPOSE 80/tcp

WORKDIR /var/www/

COPY entrypoint.sh /entrypoint.sh
RUN chmod +x /entrypoint.sh

ADD index.tar.gz /var/www/html/

ENTRYPOINT ["/entrypoint.sh"]
CMD ["/usr/sbin/nginx", "-g", "daemon off;"]
```



## Criando a Imagem e Executando o Container

```
$ docker build -t biopark/nginx ". "
```

```
$ docker run --name proxy -d -p 80:80  
biopark/nginx
```



## Disponibilizando a imagem no docker hub

```
$ echo "suasenha" > ~/dh-pass.txt
```

```
$ cat ~/dh-pass.txt | docker login -u username --password-stdin
```

```
Authenticating with existing credentials...  
Login Succeeded
```

```
Logging in with your password grants your terminal complete access to your account.  
For better security, log in with a limited-privilege personal access token. Learn more at  
https://docs.docker.com/go/access-tokens/
```

```
$ docker tag biopark/nginx account/image-name
```

```
$ docker push account/image-name
```

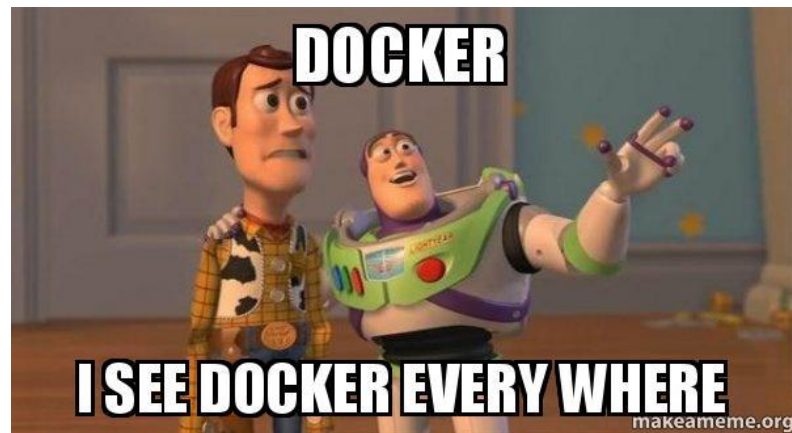


## Hands On

- Acessar Codespace
- Clonar o repositório *biopark-docker*
- Acessar a pasta dockerfile
- Alterar o arquivo html adicionando o nome
- Construir a imagem e enviar para Docker Hub

<https://docs.docker.com/language/nodejs/build-images/>



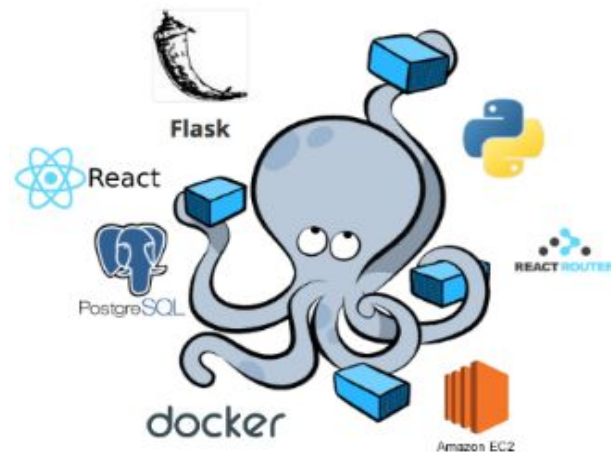


---

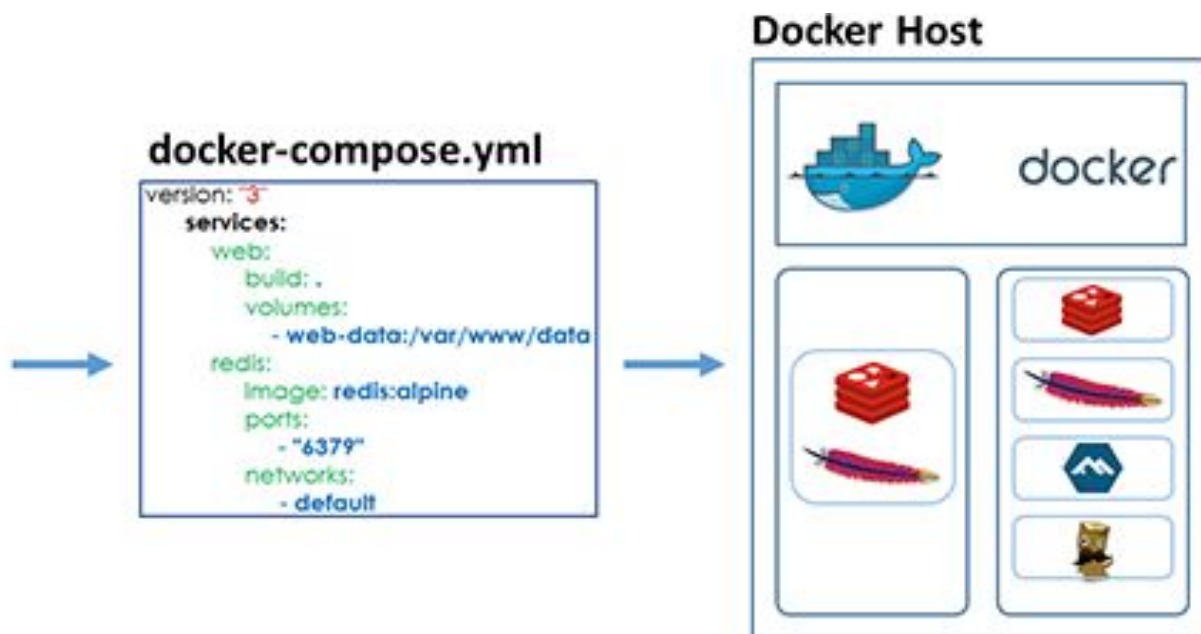
# Docker Compose

# O que é?

Docker Compose é o orquestrador de containers da Docker.



# Estrutura





# Estrutura

```
services:
  web:
    image: wordpress
    restart: always
    depends_on:
      - db
    ports:
      - 80:80
    networks:
      - frontend
      - backend
    environment:
      WORDPRESS_DB_HOST: db
      WORDPRESS_DB_USER: root
      WORDPRESS_DB_PASSWORD: 123456
      WORDPRESS_DB_NAME: wordpress
    volumes:
      - wordpress:/var/www/html

  db:
    image: mysql:5.7
    restart: always
    environment:
      MYSQL_DATABASE: wordpress
      MYSQL_ROOT_PASSWORD: '123456'
    volumes:
      - db:/var/lib/mysql
    networks:
      - backend
```

```
volumes:
  wordpress:
  db:

networks:
  frontend:
  backend:
```



## Commandos

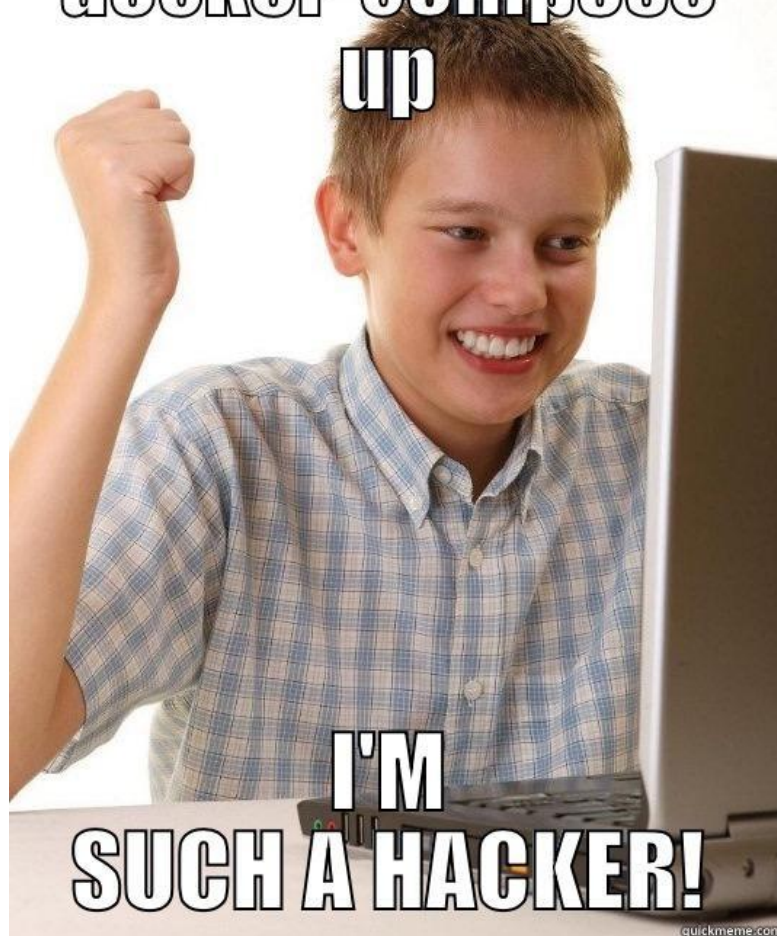
```
$ docker-compose up -d
$ docker-compose start SERVICE_NAME
$ docker-compose stop SERVICE_NAME
$ docker-compose restart SERVICE_NAME
$ docker-compose down
$ docker-compose ps
$ docker-compose build
$ docker-compose up -d -build
```



## Hands On

- Acessar Codespace
- Clonar o repositório *biopark-docker*
- Acessar a pasta docker-compose
- Iniciar serviços

**docker-compose  
up**



**I'M  
SUCH A HACKER!**



**Até a próxima**

