

COMO AUTOMATIZAR SEU PRIMEIRO DEPLOY COM DOCKER



Hugo Iuri


Connection Day
2019



O processo de deploy dos softwares, quando executado de forma manual, além de não trazer segurança ao processo, geralmente ocupa muito tempo da equipe de operações.

Nesta palestra vamos abordar uma forma simples de operacionalizar entrega contínua de uma aplicação feita em Node.js em um ambiente Docker.

Oi meu nome é Hugo

Sou um desenvolvedor apaixonado
por novas tecnologias.

Trabalho com Docker e Node.js
desde 2014



@hugoiuri



hugoiuri.dev

contato@hugoiuri.dev



Processo de entrega de softwares

O processo de deploy tradicional

É gerado manualmente um artefato implantável;

É disponibilizado um roteiro de implantação;

É criada uma janela de manutenção no sistema;

Alguém segue o roteiro e implanta manualmente o sistema;

Caso algum problema aconteça a pessoa implantadora executa o rollback;

Caso dê tudo certo o processo é finalizado.

Entrega Contínua

O artefato de implantação é gerado automaticamente através de um commit;

Uma aprovação manual pode ser requerida com um clique;

A implantação é feita automaticamente utilizando um script;

Testes pós implantação são executados automaticamente;

Em caso de falha o rollback é feito automaticamente e o time é notificado;

Em caso de sucesso o time é notificado automaticamente;

12 horas

Economizadas por implantação

≈ 240 horas

Economizadas por mês com implantação automatizada

O Docker

- É uma plataforma que possibilita a criação de “ambientes” isolados;
- Fornece o isolamento da árvore de processos, memória, disco e rede;
- Possibilita o empacotamento e distribuição da aplicação;
- Reduz o desperdício de recursos utilizando o mesmo kernel do host;
- Facilita a manutenção e operação dos sistemas em produção;
- É uma excelente ferramenta para o desenvolvimento de sistemas.

O Docker Swarm

É um orquestrador de containers criado pela Docker;

É 100% integrado ao Docker;

Possui uma curva de aprendizado muito baixa para usuários do Docker;

É simples de ser implantado e mantido no ambiente de produção;

É uma excelente opção para quem está começando no mundo dos containers.

Docker e Kubernetes

O Kubernetes é um orquestrador de containers desenvolvido pela Google;

Atualmente é o orquestrador mais popular e usado pela comunidade;

É extremamente poderoso, porém muito complexo para iniciantes;

Pode ser utilizado com Docker ou outros sistemas de containers

No final de 2019 a Docker vai lançar uma versão do Kubernetes customizada para usuários Docker com integração ao Docker Compose e a sintaxe familiar aos usuários do Docker.

A estrutura do nosso projeto

Desenvolvemos uma plataforma composta por micro serviços;

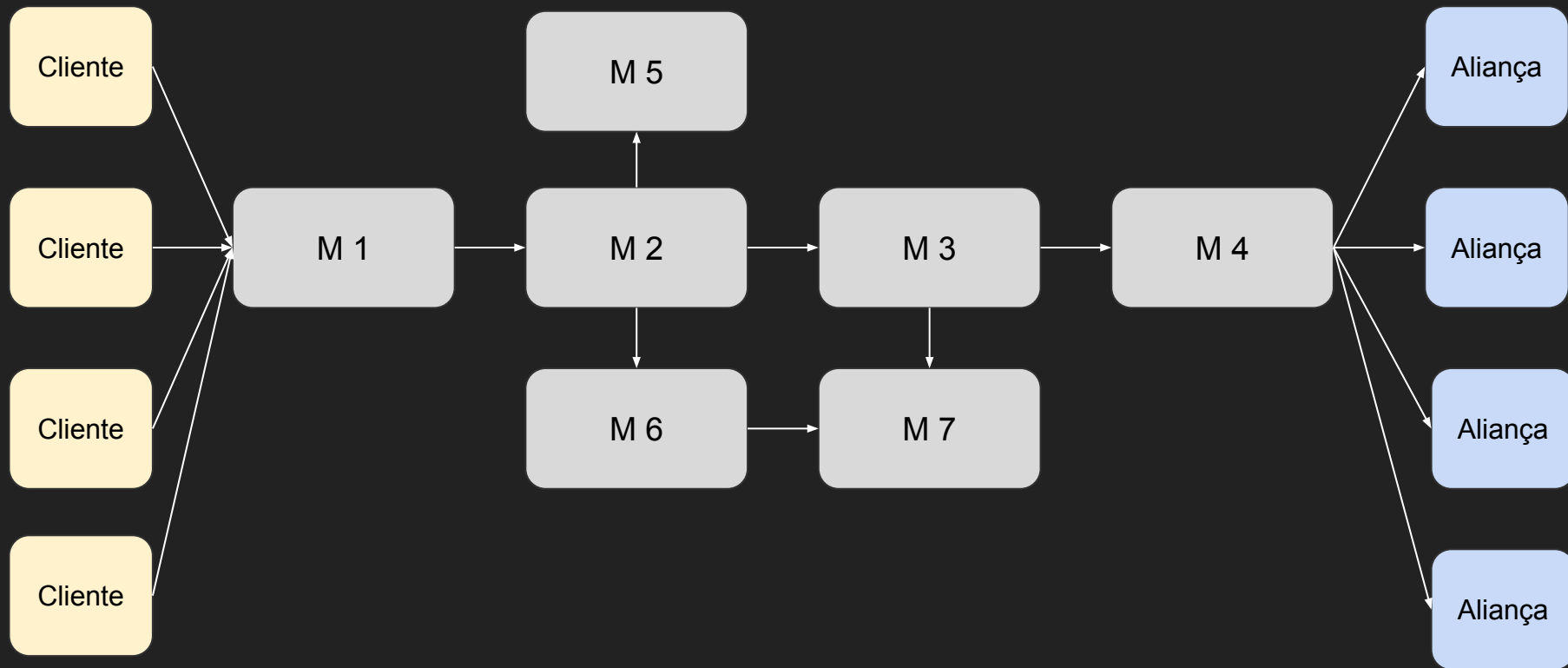
Utilizamos o Javascript e Node.js como principal plataforma de desenvolvimento;

Utilizamos o Docker Swarm como orquestrador;

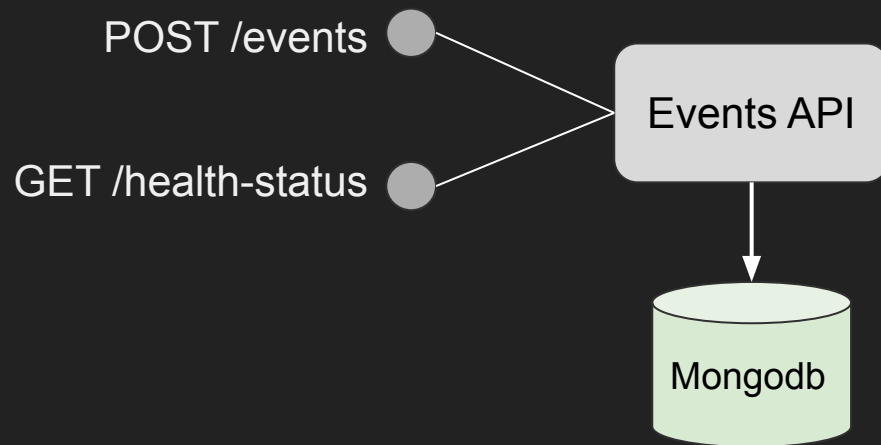
O Pipeline de integração contínua feito no CircleCi;

Todo o processo de deploy é automatizado.

Como funciona a nossa solução



Nossa API de demonstração



Construindo a nossa imagem Docker - Dockerfile

```
1  FROM node:10.15.3-alpine as builder
2  WORKDIR /opt/events-api
3  COPY . /opt/events-api
4  RUN npm i --production
5
6  FROM node:10.15.3-alpine
7  RUN apk --no-cache add ca-certificates
8  WORKDIR /opt/events-api
9  COPY --from=builder /opt/events-api .
10 ENTRYPOINT ["node", "./src/index.js"]
```

Construindo a nossa imagem Docker - build

```
$ docker login -u $DOCKER_USER -p $DOCKER_PASSWORD
```

```
$ docker build -t NOME_DA_IMAGEM:TAG ex: events-api:1.0.0
```

```
$ docker push NOME_DA_IMAGEM:TAG ex: events-api:1.0.0
```


Construindo a nossa imagem Docker - build

 minutrade / events-api

This repository does not have a description 

 Last pushed: 8 hours ago

Docker commands

To push a new tag to this repository,

```
docker push minutrade/events-api:tagname
```

Tags

This repository contains 2 tag(s).

1.0.1



 8 hours ago

1.0.0



 9 hours ago

[See all](#)

Fazendo o deploy da imagem - Docker Compose

```
1  version: '3.3'
2  services:
3    events-api:
4      # {PACKAGE_VERSION} é substituído automaticamente durante o
5      # deploy pela versão do package.json
6      image: minutrade/events-api:{PACKAGE_VERSION}
7      ports:
8        - "8339:8339"
9      environment:
10        - "PORT=8339"
11        - "DATABASE_URI=mongodb://localhost:27017/events"
12        - "DATABASE_NAME=events"
13      deploy:
14        mode: replicated
15        replicas: 1
16        update_config:
17          parallelism: 1
18          delay: 5s
19        placement:
20          constraints: [node.labels.engine == true]
```

Fazendo o deploy da imagem - Docker Swarm

```
$ docker login -u $DOCKER_USER -p $DOCKER_PASSWORD
```

```
$ docker stack deploy --with-registry-auth -c events-api-stack.yml events-api
```

```
$ docker service ps events-api_events-api
```

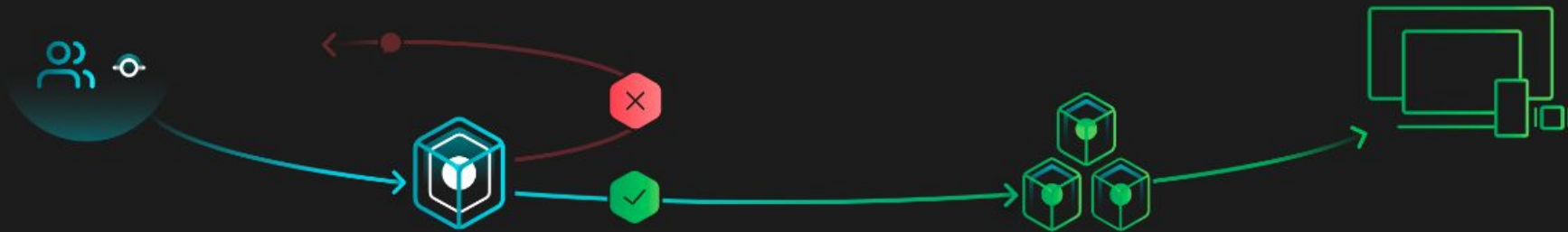
```
$ docker service logs events-api_events-api
```

Circleci


É um sistema de automação de pipelines



É um software como serviço SAAS

Funciona online e completamente gerenciado







Configuração da Pipeline - CircleCI

 SUCCEEDED

 Rerun 

master / automated-deploy-process

 ajuste no circle.yml

 8 hrs ago	 02:43
	 a2e9ff6

2 jobs in this workflow



Configuração da Pipeline - Workflow

```
48   workflows:
49     version: 2
50     automated-deploy-process:
51       jobs:
52         - build:
53           context: org-global
54         - deploy:
55           context: org-global
56           filters:
57             branches:
58               only: master
59           requires:
60             - build
```

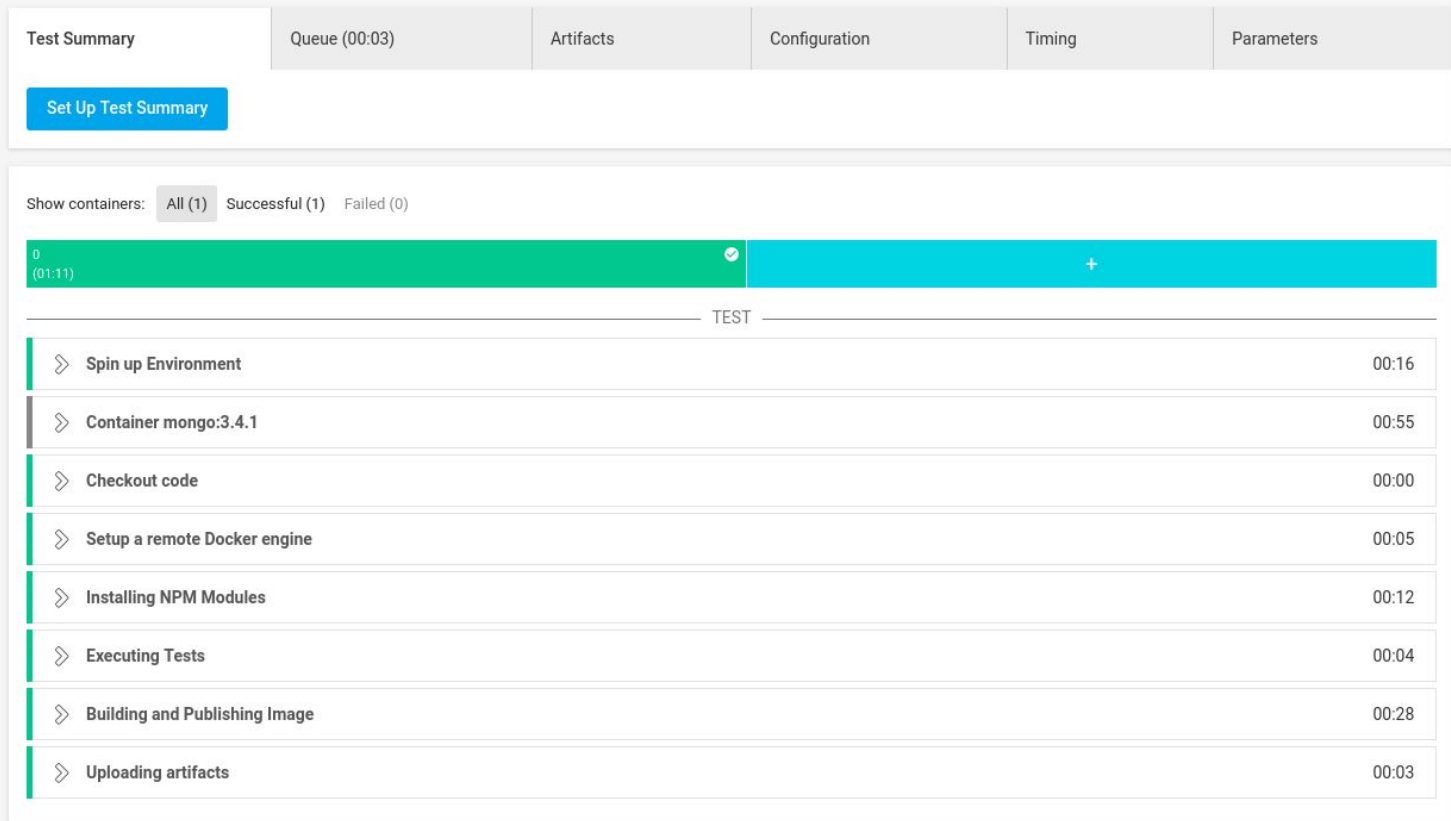
Configuração da Pipeline - Build

```
3   build:
4     working_directory: ~/build
5     docker:
6       - image: minutrade/circleci-build-node:10.15.3
7         auth:
8           username: $DOCKER_USER
9           password: $DOCKER_PASSWORD
10      - image: mongo:3.4.1
11    steps:
12      - checkout
13      - setup_remote_docker:
14        version: 18.05.0-ce
15      - run:
16        name: Installing NPM Modules
17        command: npm install
18      - run:
19        name: Executing Tests
20        command: npm test
21      - run:
22        name: Building and Publishing Image
23        command: bash ./scripts/publish-docker-image.sh
24      - store_artifacts:
25        path: coverage
```

Configuração da Pipeline - Build

```
12 | | | - checkout
13 | | | - setup_remote_docker:
14 | | |   version: 18.05.0-ce
15 | | | - run:
16 | | |   name: Installing NPM Modules
17 | | |   command: npm install
18 | | | - run:
19 | | |   name: Executing Tests
20 | | |   command: npm test
21 | | | - run:
22 | | |   name: Building and Publishing Image
23 | | |   command: bash ./scripts/publish-docker-image.sh
```


Execução do job build



Execução do job build

 Checkout code Setup a remote Docker engine Installing NPM Modules Executing Tests Building and Publishing Image

Configuração da Pipeline - Deploy

```
27   deploy:
28     working_directory: ~/build
29     docker:
30       - image: minutrade/circleci-build-node:10.15.3
31         auth:
32           username: $DOCKER_USER
33           password: $DOCKER_PASSWORD
34     steps:
35       - checkout
36       - setup_remote_docker:
37         version: 18.05.0-ce
38       - run:
39         name: Deploying Release Image in Dev
40         command: |
41           export SWARM_DEPLOY_CA=$SWARM_DEV_CA
42           export SWARM_DEPLOY_CERT=$SWARM_DEV_CERT
43           export SWARM_DEPLOY_KEY=$SWARM_DEV_KEY
44           export SWARM_DEPLOY_HOST=$SWARM_DEV
45
46           bash ./scripts/swarm-deploy.sh
```

Configuração da Pipeline - Deploy

```
35 - checkout
36 - setup_remote_docker:
37   |   version: 18.05.0-ce
38 - run:
39   |   name: Deploying Release Image in Dev
40   |   command: |
41     |     export SWARM_DEPLOY_CA=$SWARM_DEV_CA
42     |     export SWARM_DEPLOY_CERT=$SWARM_DEV_CERT
43     |     export SWARM_DEPLOY_KEY=$SWARM_DEV_KEY
44     |     export SWARM_DEPLOY_HOST=$SWARM_DEV
45
46     |     bash ./scripts/swarm-deploy.sh
```

Configuração TLS Docker



TLS - Transport Layer Security

É um protocolo de criptografia projetado para a internet

Permite uma comunicação segura entre o cliente e o servidor

- <https://docs.docker.com/v17.12/datacenter/ucp/2.2/guides/admin/configure/use-your-own-tls-certificates/>
- <https://docs.docker.com/engine/security/https/>

Execução do job Deploy

Test Summary	Queue (00:02)	Artifacts	Configuration	Timing	Parameters
Set Up Test Summary					
Show containers: All (1) Successful (1) Failed (0)					
<div>0 (01:24)  </div>					
TEST					
<div><div> Spin up Environment</div><div>00:20</div></div>					
<div><div> Checkout code</div><div>00:03</div></div>					
<div><div> Setup a remote Docker engine</div><div>00:06</div></div>					
<div><div> Deploying Release Image in Dev</div><div>00:52</div></div>					

Execução do job Deploy



Checkout code



Setup a remote Docker engine



Deploying Release Image in Dev

Execução do job Deploy

```
$ #!/bin/bash -eo pipefail
export SWARM_DEPLOY_CA=$SWARM_DEV_CA
export SWARM_DEPLOY_CERT=$SWARM_DEV_CERT
export SWARM_DEPLOY_KEY=$SWARM_DEV_KEY
export SWARM_DEPLOY_HOST=$SWARM_DEV
```

```
bash ./scripts/swarm-deploy.sh
```

```
Updating project events-api to version 1.0.1 on docker swarm-
Docker Swarm Host: ci-dev.bonuz.com:9443
Project Name: events-api
Stack Name: events-api
Image: minutrade/events-api
Version: 1.0.1
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
```

```
Login Succeeded
Current image: minutrade/events-api:1.0.0
Updating service events-api_events-api (id: 91gfq5tlu5n7jfbcgi0ygb4zx)
Number of replicas: 1/1
Deploy verified!
The image minutrade/events-api:1.0.1 was deployed in swarm-
```

Exit code: 0



Execução do job Deploy

```
Updating project events-api to version 1.0.1 on docker swarm-
Docker Swarm Host: ci-dev.bonuz.com:9443
Project Name: events-api
Stack Name: events-api
Image: minutrade/events-api
Version: 1.0.1
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
Current image: minutrade/events-api:1.0.0
Updating service events-api_events-api (id: 91gfq5tlu5n7jfbcgi0ygb4zx)
Number of replicas: 1/1
Deploy verified!
The image minutrade/events-api:1.0.1 was deployed in swarm-
```

Links

Código fonte disponível em:

<http://bit.ly/ConnectionDay19>

Apresentação disponível em:

<http://bit.ly/ConnectionDay19Presentation>

Thank You!

Aitäh

Je vous remercie

Tak

Ačiū

Дзякуй

Dziękuję Ci

Děkuji

Takk skal du ha

ارکش

謝謝

Спасибо

고맙습니다

Obrigado

Grazie

Paldies

Kiitos

Mulțumesc

Hvala

Gratias tibi

Хвала вам

Gracias

Tack

Danke

Дякую

Dank je

Ευχαριστώ