

# Criando pipelines com o Jenkins

Samuel Maciel Sampaio  
@samukasmk  
samuel@smk.net.br

# Sobre mim

- Samuel Maciel Sampaio
- 29 anos
- Administrador de sistemas a 6 anos
- Desenvolvedor Backend Python a 2 anos
- Entusiasta da cultura de DevOps
- @samukasmk (telegram, face, gmail)
- <http://bit.ly/SamuelSampaio>

# O que é o Jenkins

- Jenkins é uma ferramenta de integração e entrega contínua.
- Que proporciona mais agilidade e produtividade, na execução de processos rotineiros.
- Tanto de qualidade de software, com a execução automatizada de testes unitários, aceitação e funcional.
- Como em tarefas operacionais de versionamento, deploys e até backups! #OCéuÉOLimite

# O que são pipelines

- Pipelines são fluxos de trabalho (workflows), seguidos por estágios ou passos;
- Agora existe plugins nativos do Jenkins para definição de pipelines
- Trazendo mais praticidade ao se definir fluxos de execução em um único Job

# Histórico

- Em meados de 2013 foi lançado o plugin (Build-Pipeline-Plugin) que interconecta varios jobs do jenkins em um workflow

The screenshot displays the Jenkins 'Build Pipeline' interface. At the top, the browser address bar shows 'localhost:8082/view/Build%20pipeline/'. The Jenkins header includes a search bar and the user 'marcinp | log out'. The main title is 'Build Pipeline: My pipeline'. Below this, there are icons for 'Run', 'History', 'Configure', 'Add Step', 'Delete', and 'Manage'. The interface shows two pipeline versions, 7 and 8, each with a sequence of steps connected by arrows. Version 8 is the current active version.

Pipeline version	Test	Release	Deploy to Test	Generate docs	Deploy to Pre-Prod	Deploy to Prod
8	Jun 26, 2012 5:30:48 PM 8 10 sec marcinp	Jun 26, 2012 5:31:03 PM 8 12 sec	Jun 26, 2012 5:31:46 PM 6 12 sec marcinp	Jun 26, 2012 5:31:20 PM 8 9.1 sec		
7	Jun 26, 2012 5:26:25 PM 7 10 sec marcinp	Jun 26, 2012 5:26:40 PM 7 12 sec	Jun 26, 2012 5:27:13 PM 5 13 sec marcinp	Jun 26, 2012 5:26:57 PM 7 9.1 sec	Jun 26, 2012 5:27:31 PM 4 10 sec	Jun 26, 2012 5:27:46 PM 4 15 sec

Page generated: 26-Jun-2012 17:31:39 Jenkins ver. 1.470

# O presente momento

- Em abril de 2016 foi lançado um novo plugin nativo do Jenkins, chamado apenas de (Pipeline-Plugin), onde é possível definir vários passos do mesmo workflow, no mesmo job, em uma visualização de colunas.

The screenshot shows the Jenkins web interface for a pipeline job named 'example'. The top navigation bar includes the Jenkins logo, a search bar, and links for 'Admin' and 'log out'. Below the navigation bar, the left sidebar contains various actions: 'Back to Dashboard', 'Status', 'Changes', 'Build Now', 'Delete Pipeline', 'Configure', 'Move', 'Full Stage View', and 'Pipeline Syntax'. The main content area is titled 'Pipeline of example' and features a 'Recent Changes' icon and a link to 'add description'. Below this, the 'Stage View' section displays 'Average stage times' for three stages: 'My stage 1' (360ms), 'My stage 2' (99ms), and 'My final stage' (218ms). A table below the stage view shows the same timing data for the current build. The 'Build History' section on the left lists recent builds, with build #4 being the most recent. The 'Permalinks' section at the bottom provides a link to the last build.

Stage	My stage 1	My stage 2	My final stage
Average stage times:	360ms	99ms	218ms
Current Build	360ms	99ms	218ms

Build Number	Timestamp
#4	Dec 2, 2016 12:04 PM
#3	Dec 2, 2016 12:03 PM
#2	Dec 2, 2016 11:19 AM

# Como definir um pipeline customizado

- Os workflows customizados, são definidos por groovy scripts, como uma forma de DSL simples. Onde cada plugin do jenkins disponibiliza os recursos, como funções. Chamados nas documentações de steps
- <https://jenkins.io/doc/pipeline/steps/>
- Esse groovy script pode ser configurado como string, na configuração do job, ou passado através de um projeto como arquivo Jenkinsfile

# Exemplo de um script de pipeline (Jenkinsfile)

```
1 node {  
2     stage 'My stage 1'  
3     sh 'set'  
4  
5     stage 'My stage 2'  
6     echo "Estou executando no job: " + env.JOB_NAME;  
7  
8     stage 'My final stage'  
9     echo "All is OK!"  
10 }
```



# Meu case de exemplo

- Desenvolvi uma aplicação em Flask
- Com 3 endpoint:
  - / (home)
  - /ping (exemplo de uma api rest json)
  - /modal (exemplo de um modal html)
- Com testes unitários
  - Se o /ping responde 200, com um json
  - Utilizado (py.test, pytest-flask e pytest-flakes)
- Com testes funcionais
  - Se a pagina externa responde com um modal
  - Utilizado (selenium, pytest-selenium e phantomjs)

<https://github.com/samukasmk/grupy-flask-jenkins.git>

# Executando o projeto na maquina local

- Clonando o projeto do github:
  - `git clone https://github.com/samukasmk/grupy-flask-jenkins.git`
- Criando o virtual env
  - `cd grupy-flask-jenkins`
  - `virtualenv --python=python3.5 venv`
- Instalando as dependencias:
  - `./venv/bin/pip install -r requirements.txt`
- Executando a aplicação
  - `./venv/bin/python app.py`
- Executando os testes unitários
  - `./venv/bin/py.test -ra -v --flakes tests/unit_tests`
- Executando os testes funcionais
  - `./venv/bin/py.test -ra -v --flakes --driver PhantomJS --driver-path=/opt/phantomjs/bin/phantomjs tests/functional_tests`

# Você percebeu ?

## Nós já temos um pipeline de integração contínua !

0. Adicionar gatilho webhook no repositório git
1. Clonar/Efetuar checkout do projeto
2. Criar o virtual env de testes
3. Instalar as dependencias
4. Executar os testes unitários
5. Executar os testes funcionais
6. Disponibilizar para deploy

# O tal Jenkinsfile...

```
node {  
    stage 'Git workspace'  
    git 'https://github.com/samukasmk/grupy-flask-jenkins.git'  
  
    stage 'Build venv'  
    sh 'virtualenv --python=python3.5 venv'  
  
    stage 'Install deps'  
    sh './venv/bin/pip install -r requirements.txt'  
  
    stage 'Unit tests'  
    sh './venv/bin/py.test -ra -v --flakes tests/unit_tests'  
  
    stage 'Functional tests'  
    sh './venv/bin/py.test -ra -v --flakes --driver PhantomJS ' \  
        '--driver-path=/opt/phantomjs/bin/phantomjs tests/functional_tests'  
  
    stage 'Docker push'  
    echo 'Se a app estivesse em um container, poderia ser um docker push'  
}
```

Hand's on

Vamos um Job com pipelines no  
Jenkins

OBRIGADO!

Duvidas?

@SamukaSMk