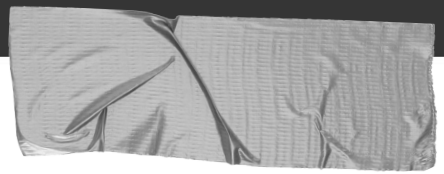
Python para processamento de Vídeo

Meta

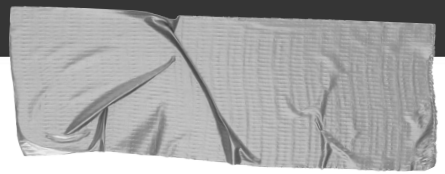
Lays Rodrigues e Yuri Vasquez



Lays Rodrigues

Desenvolvedora Python | Meta

- **Cursando Sistemas de Informação**
  Universidade Federal Fluminense
- **KDE**
  Porque Software Livre *matters*.
- **Mindfulness**
  “Every day is a chance to train your mind for a happier, healthier life.”
- lays.silva@meta.com.br
- lays147.org



Yuri Vasquez

Especialista em Tecnologia | Rede Globo

- **Entusiasta devops**
  Cultura devops!!
- **Tech Lead ocasional**
  Colaboração S2
- **Fã de boardgames**
  Partiu dominion?
- yuri.vasquez@tvglobo.com.br

Copa do Mundo de 2018?

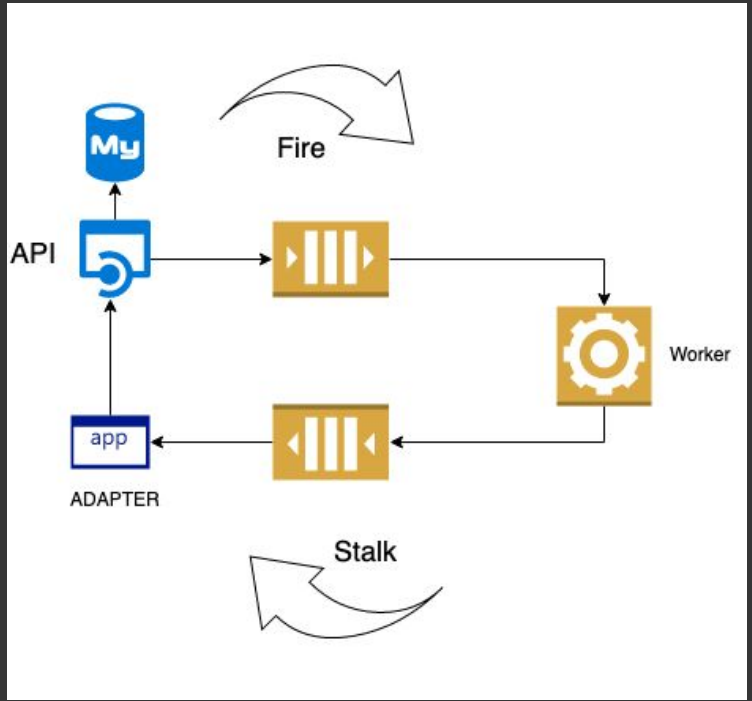
Foram mais de **165**

Horas, de conteúdo exclusivo FIFA processado e importado pelo fluxo em um processo 100% transparente

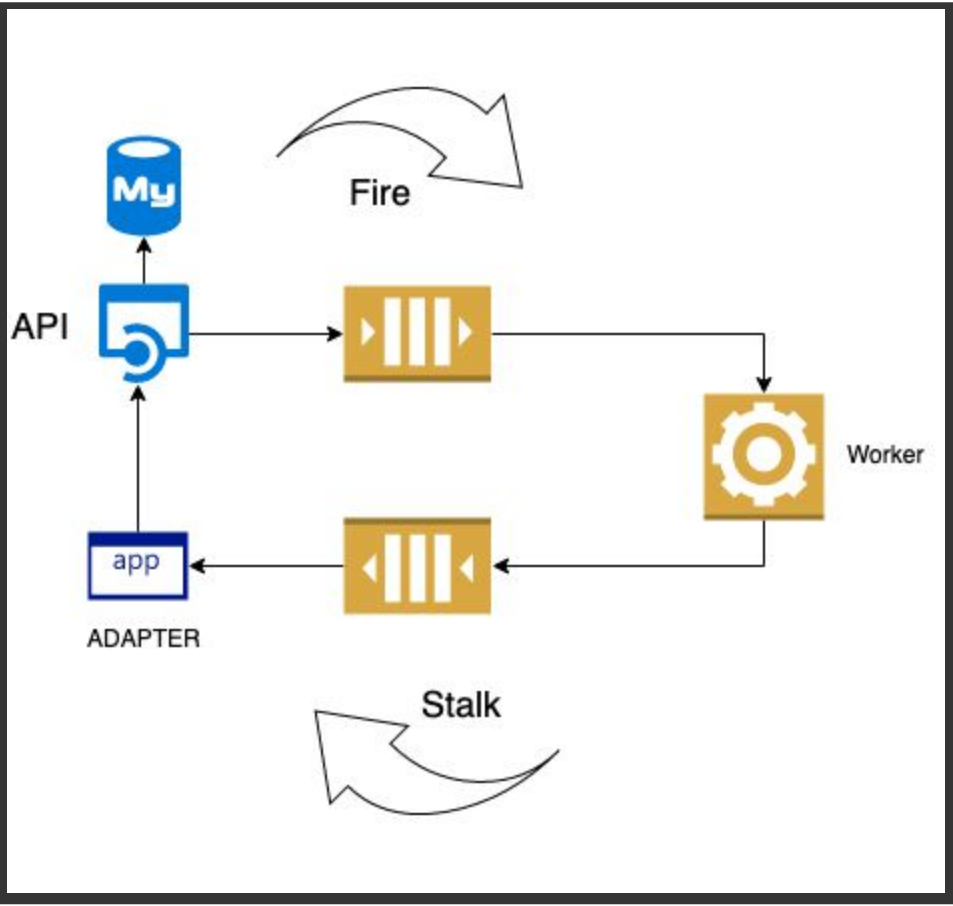
Como isso aconteceu?

~~SWF = Simple Workflow~~

Workflow *simples* e *escalável* para processamento de vídeo para uso interno da Rede Globo



Zoom In



SWF - API

- API REST
- Flask + SQLAlchemy + kombu ...
- Gerencia todo o ciclo de vida do Job

SWF - Adapter

- Kombu + gevent + requests
- Consome das filas de resposta
- Envia requests para API

SWF - Dashboard

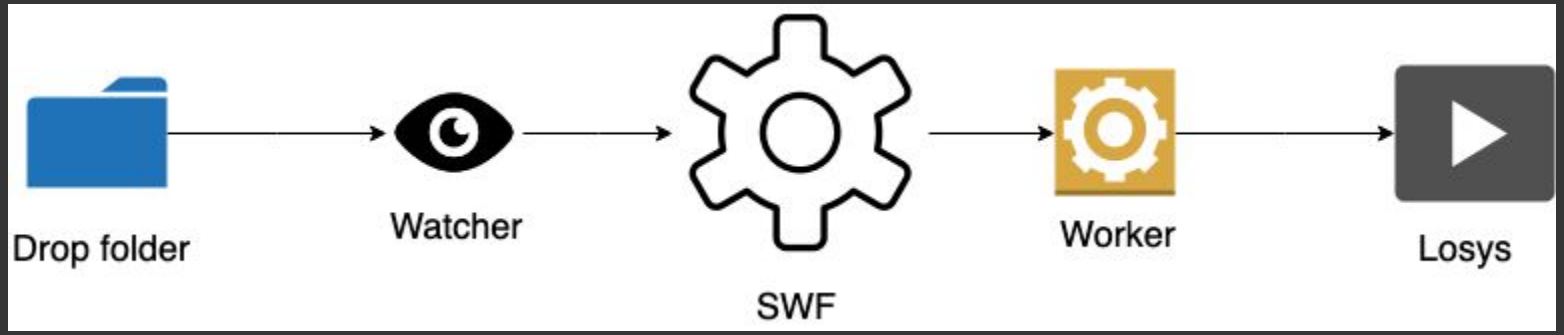
- Acompanhamento de jobs
- VueJS + Vuex + axios

IMPORTAPI
ANALYTICS

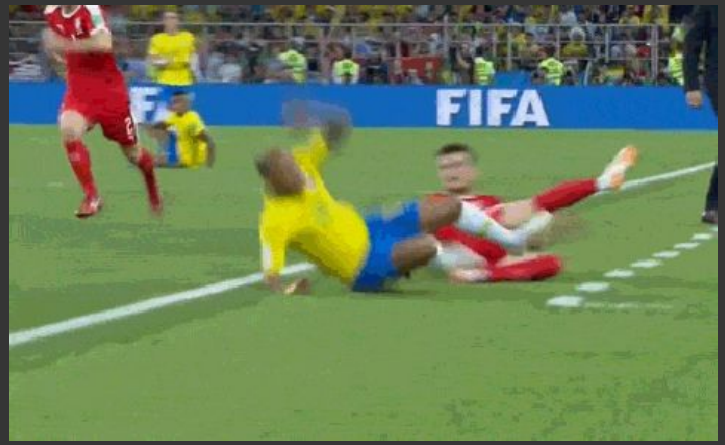
Pesquisar...

ID	NOME	DATA DE CRIAÇÃO	WORKFLOW	STATUS	FILA	STEP	PROGRESSO
7	hello floripa 4	26/04/2019 22:54:43	Audio Extract	Processing	audio	**EXTRACT AUDIO (2/2)**	0 %
6	hello floripa 3	26/04/2019 22:54:18	Audio Extract	Processing	audio	**ANALYSIS (1/2)**	2 %
5	hello floripa 2	26/04/2019 22:52:27	Audio Extract	Processing	audio	**EXTRACT AUDIO (2/2)**	0 %
4	hello floripa	26/04/2019 22:41:08	Audio Extract	Processing	audio	**ANALYSIS (1/2)**	0 %
3	hello tdc	26/04/2019 22:37:04	Audio Extract	Processing	audio	**ANALYSIS (1/2)**	0 %
2	test	26/04/2019 19:46:00	Audio Extract	Processing	audio		100 %
1	Monty Python - Bridge of Death- 0D7hFHfLEyk	26/04/2019 17:43:02	Audio Extract	Processing	audio		100 %

Zoom Out



neymar_rolando.mp4



POST /api/ingests

```
{
    metadata:{
        name: "Neymar Rolando",
        media_path: 20190427_a9c7e.mp4
    }
}
```

Job

1. Analisa o vídeo
2. Gera a alta resolução (Formato TV)
3. Gera a baixa resolução (browser / network friendly)
4. Copia para os storages
5. Cria asset no Losys
6. Envia alta para parceiros
7. Apaga os temporários

A taça do mundo é nossa!

Replicando

Como replicar um sistema de processamento de vídeo para atender outras necessidades sem ter que reescrever outro sistema do 0?

Fluxos Criados

Ao final desta seção, o público deverá ser capaz de visualizar:

- **Migração de Acervo Esporte**
  Duração: **< 7 meses**
- **Migração de Acervo Jornalismo**
  Estimativa: **< 1 mês**
- **Fluxos de Exportação**

  Fluxo contínuo:

  - RJ: **7065 items**
  - SP: **4746 items**

Bibliotecas!

Fudd

- SDK para desenvolvimento de workers
- Abstrai transporte (AMQP)
- Separa o job em passos
- Facilita a reutilização de passos
- worker == sequência de passos

STEP

(a.k.a. passo)

```python
from pathlib import Path
from fudd import JobHandler, Result, Ok
from ..util import progress_copy

class CopyFileStep:
    def __init__(self, workdir: str, targetdir: str):
        self.workdir = Path(workdir)
        self.targetdir = Path(targetdir)

    def run(self, job_handler: JobHandler) -> Result:

        src_path = workdir / job_handler.data['files']['media_path']
        target_path = workdir / self.target_path(job_handler)
        job_handler.worklog.info('Starting copy from %s to %s',
                                 src_path, target_path)

        for progress in progress_copy(src_path, target_path):
            job_handler.update_progress(progress)

        return Ok()
```

```python
from fudd import JobHandler, Result, Ok

class NoopStep:
    __display_name__ = 'Useless Step'

    def run(self, job_handler: JobHandler) -> Result:

        return Ok()
```

```python
from pathlib import Path
from fudd import JobHandler, Result, Ok
from ..util import progress_copy

class CopyFileStep:
    def __init__(self, workdir: str, targetdir: str):
        self.workdir = Path(workdir)
        self.targetdir = Path(targetdir)

    def run(self, job_handler: JobHandler) -> Result:

        src_path = workdir / job_handler.data['files']['media_path']
        target_path = workdir / self.target_path(job_handler)
        job_handler.worklog.info('Starting copy from %s to %s',
                                 src_path, target_path)

        for progress in progress_copy(src_path, target_path):
            job_handler.update_progress(progress)

        return Ok()
```

pathlib magic


```
src_path = workdir / job_handler.data['files']['media_path']
```


Argumentos do job

fudd magic

```
job_handler.worklog.info(
        Extracting audio from %s',
        src_path,
)
```

Pesquisar...

Worklog

WORKER
Audio Worker 1

DATA
26/04/2019
22:54:43

STATUS
INFO

Analysing data

WORKER
Audio Worker 1

DATA
26/04/2019
22:55:56

STATUS
INFO

Extracting audio from /mnt/workdir/6cf6b464688f11e9ba958c85903bbe6d.mp4

WORKER
Audio Worker 1

DATA

Falha durante execução de worker

__ fudd magic

```
job_handler.update_progress(progress)
```

Total Jobs: 7

ID	NOME	DATA DE CRIAÇÃO	WORKFLOW	STATUS	FILA	STEP	PROGRESSO
7	hello floripa 4	26/04/2019 22:54:43	Audio Extract	Processing	audio	**ANALYSIS (1/2)**	37 %
6	hello floripa 3	26/04/2019 22:54:18	Audio Extract	Processing	audio	**ANALYSIS (1/2)**	2 %

Patience

- Biblioteca de file watch
- Usa o watchdog
- Suporta estabilização de arquivo

Sauron

- SDK para desenvolvimento de watchers
- Integrado com SWF
- Definido por conf + 'parser' de metadados

PARSER

(a.k.a. extrator de metadados)

```python
class MyParser:
    def parse(self, file_path: str) -> swf_client.Ingest:

        name = splitext(basename(file_path))[0].replace('_', ' ')
        return swf_client.Ingest(
            name=name,
            workflow_id=conf['workflow_id'],
            metadata={'source_name': 'TDC'},
            file_metadata={'media_path': file_path},
        )
```

WATCHER

(a.k.a. app class)

```
watcher = WatcherHelper(
        parser=MyParser(),
        watchdir=conf['watchdir'],
        workdir=conf['workdir'],
        import_url=conf['import_url'],
        ext=['*.mp4'],
        remove_meta=False,
        timeout=conf['growth_threshold'],
    )
watcher.run()
```

Maiores Desafios

FFMPEG

- Encoding de vídeo
- IRC

SFTP

- Tuning
  - paramiko vs ssh2

O que mais tem na caixinha?

- Operação 100% automatizada com Ansible + Gitlab-Ci
- Todo o projeto é um pacote python
- devpi

Quem contribuiu

- Ciro Chang
- Cristina Silva
- Gabriel Paladino
- Lays Rodrigues
- Marcos Lima
- Raphael Marra
- Yuri Vasquez


Obrigado!
Perguntas?
Meta
lays.silva@meta.com.br
yuri.vasquez@tvglobo.com.br