# O CASAMENTO SAGRADO ENTRE O PIPFILE, PIP E VIRTUALENV

# @allythy

- Gestão da Tecnologia  da Informaçã
- Colaborador e autor do SempreUpdat
- Coordenador do PotiLivre
- Fundador e Colaborador do PHP-RN
- Coordenador do GruPy-RN

# O QUE É O PIP ?

O pip é uma ferramenta de gestão de pacotes usado para instalar e gerenciar pacotes Python.

# INSTALAÇÃO DO PIP

## No Debian e derivados (Python3):

```
sudo apt install python3-pip
```

## Usando o script (Python3):

```
curl https://bootstrap.pypa.io/get-pip.py
```

```
python3 get-pip.py
```

# O QUE É VIRTUALENV?

O virtualenv é uma ferramenta que permite que criemos ambientes virtuais isolados para projetos Python.

# INSTALANDO E USANDO O VIRTUALENV

Instalando:

```
sudo pip install virtualenv
```

Criando o ambiente virtual:

```
virtualenv NomeDoAmbiente
```

Ativando o ambiente virtual:

```
source ./NomeDoAmbiente/bin/activate
```

# O QUE É VIRTUALENVWRAPPER?

Junto com o virtualenv existe também um wrapper que facilita a forma como trabalhamos com estes ambientes virtuais. Esse pacote é chamado vritualenvwrapper. O virtualenvwrapper oferece uma série de atalhos para utilizar o virtualenv, além de organizar todos os ambientes virtuais em um único lugar.

# INSTALANDO E USANDO O VIRTUALENVWREPPER

## Instalando:

```
sudo pip install virtualenvwrapper
mkdir ~/.virtualenvs
```

## Colocar no final do seu arquivo `.bashrc`:

```
export WORKON_HOME=~/.virtualenvs
source /usr/local/bin/virtualenvwrapper.sh
```

## Criando virtual env:

```
mkvirtualenv NomeDoVirtualEnv
```

## Ativando:

```
workon  NomeDoVirtualEnv
```

# GERANDO O REQUIREMENTS.TXT

```
pip freeze > requirements.txt
```

```
pip install -r requirements.txt
```

# PIPENV

- Você não precisa mais usar pip e virtualenv separadamente. Eles trabalham juntos.

- Cria automaticamente o ambiente virtual.

- Não precisa do requirements.txt. Temos Pipfile e o Pipfile.lock

# CRIADOR:

Kenneth Reitz

# INSTALAÇÃO

```
pip3 install pipenv
```


LIVE CODE

# Inicializando o projeto:

```
pipenv install request
```

```
pipenv install
```

## Ativando o ambiente:

```
pipenv shell
```

# Escolhendo a versão do Python:

```
pipenv --two
```

```
pipenv --python 2.0.1
```

Executando comandos na virtualenv, mas sem entrar nela:

```
pipenv run [comando]
```

## Instalando um pacote:

```
pipenv install requests
```

## Especificando a versão do Pacote:

```
pipenv install requests==2.13.0
```

## Instalando mais de um pacote ao mesmo tempo:

```
pipenv install requests django pytest
```

Desinstalando um pacote:

```
pipenv uninstall requests
```

Desinstalando todos pacotes, mas ainda deixa no Pipfile:

```
pipenv uninstall --all
```

Desinstala todos pacote de desenvolvimento e também remove os mesmo do Pipfile:

```
pipenv uninstall --all-dev
```

## Atualizando todos pacotes do Pipfile:

```
pipenv update
```

## Atualizando apenas um pacote:

```
pipenv update [nome-do-pacote]
```

Mostra um gráfico das dependências do do seu projeto:

```
pipenv graph
```

Verifica se há vulnerabilidades de segurança e afirma que os requisitos do PEP 508 estão sendo atendidos pelo ambiente atual:

```
pipenv check
```

MAS COMO TODO CASAMENTO,
NÃO É SÓ FLORES

# PARA PESQUISAR

- Poetry
- Hatch


OBRIGADO

# CONTATOS

Telegram: https://t.me/allythy

GitHub: https://github.com/allythy

Site: https://allythy.github.io/

E-mail: allythy@protonmail.com