Aula Prática - Desafio 04 - DIO - Bootcamp Engenharia de dados com Python

Descomplicando a criação de pacotes em Python

Karina Kato - Machine Learning Engineer/Machine Learning Tech Lead - Take

Proposta: Fazer um pacote usando a estrutura simples de um módulo para testar os conhecimentos adquiridos.

Objetivos do Projeto

1 - Entender conceitos relacionados aos pacotes

2 - Atualizar o projeto e gerar as distribuições

3 - Publicar o pacote

Parte 1: Introdução e conceitos

Módulo vs Pacote

Módulo: objeto que serve como unidade organizacional do código que é carregado pelo comando de import.

Pacote: coleção de módulos com hierarquia

Modularização

Vantagens da modularização:

- Legibilidade

- Manutenção

- Reaproveitamento de código

Pacote em Python

Vantagens de criar um pacote:

- Facilidade de compartilhamento

- Facilidade de instalação

Conceitos

- Pypi: repositório público oficial de pacotes

- Wheel e Sdist: dois tipos de distribuições

- Setuptools: pacote usado em setup.py para gerar as distribuições

- Twine: pacote usado para subir as distribuições no repositório Pypi

Parte 2: Criar o projeto e gerar as distribuições

Exemplos de estruturas

Estrutura de pacote simples

Estrutura de pacote com vários módulos

Repositórios disponíveis

Passos para criar o projeto

1 - Fork do template

2 - Adição do conteúdo dos módulos do projeto

3 - Edição do arquivo setup.py

4 - Edição do requirements.txt

5 - Edição do README.md

Exemplo de pacote com vários módulos

Arquivos do projeto image-processing

Arquivo setup.py

Usado para especificar como o pacote deve ser construído.

Documentação:

https://setuptools.readthedocs.io/en/latest/setuptools.html

Arquivo requirements.txt

Usado para passar as dependências que devem ser instaladas com o seu pacote. Opcionalmente, podem ser especificadas as versões

Arquivo README.md

Será exibido como documentação na página do Pypi do seu pacote. Foi usado markdown.

Acessar a raiz do projeto

Distribuições

Para subir o pacote, criar uma distribuição binária ou distribuição de código fonte.

As versões mais recentes do pip instalam primeiramente a binária e usam a distribuição de código fonte, apenas se necessário.

De qualquer forma, iremos criar ambas distribuições.

Passos para gerar as distribuições

1 - Comandos de instalação

2 - Comando para criar a distribuição

3 - Criar conta no Test Pypi

Comandos de instalação

python -m pip install --upgrade pip

python -m pip install --user twine

python -m pip install --user setuptools

Comandos para criar distribuições

python setup.py sdist bdist_wheel

Parte 3: Publicando o pacote

Passos para subir o pacote

1 - Criar conta no Test Pypi

2 - Publicar no Test Pypi

3 - Instalar pacote usando Test Pypi

4 - Testar pacote

5 - Criar conta no Pypi

6 - Publicar no Pypi

7 - Instalar pacote usando Pypi

Criando contas no test.Pypi e no Pypi

https://pypi.org/account/register/

https://test.pypi.org/account/register/

Comando para publicar no Test Pypi

python -m twine upload --repository-url https://test.pypi.org/legacy/ dist/*

Comando para instalar o pacote de teste

pip install –-index-url https://test.pypi.org/simple/ image-processing

Comando para publicar no Pypi

python -m twine upload --repository-url https://upload.pypi.org/legacy/ dist/*

Comando para instalar o pacote

python -m pip install package_name

Resumo Geral do processo

1 - Fork do template

2 - Adição do conteúdo dos módulos do projeto

3 - Edição do arquivo setup.py

4 - Edição do requirements.txt

5 - Edição do README.md

1 - Acessar a raiz do projeto

2 - Comandos de instalação

3 - Comando para criar a distribuição

1 - Criar conta no Test Pypi

2 - Publicar no Test Pypi

3 - Instalar pacote usando Test Pypi

4 - Testar pacote

5 - Criar conta no Pypi

6 - Publicar no Pypi

7 - Instalar pacote usando Pypi