**Empacotamento de dados em Python - Issue Sprint 01**

**Duo:** Cecília e Larissa

Empacotamento dentro de funções, pode passar um número indeterminado de argumentos dentro de uma função com o comando (\*args)

<https://www.youtube.com/watch?v=1TbOk_r3eYc>

**Empacotamento de aplicações:**

Um número crescente de sistemas operacionais – incluindo Linux, Mac OS e Windows – pode ser configurado para executar aplicações empacotadas como imagens leves, usando um arranjo relativamente moderno, frequentemente referido como [virtualização em nível de sistema operacional](https://en.wikipedia.org/wiki/Operating-system-level_virtualization) ou *conteinerização*.

Essas técnicas são principalmente Python agnósticas, porque empacotam sistemas de arquivos inteiros de sistemas operacionais, não apenas Python ou pacotes Python.

A adoção é mais extensa entre os servidores Linux, onde a tecnologia se originou e onde as tecnologias abaixo funcionam melhor:

* [AppImage](https://appimage.org/)
* [Docker](https://www.fullstackpython.com/docker.html)
* [Flatpak](https://flatpak.org/)
* [Snapcraft](https://snapcraft.io/)

Tutorial para criar um pacote em Python:

<https://packaging.python.org/pt_BR/latest/tutorials/packaging-projects/>

# 

# 

# 

# **O Que é o Empacotamento de Python?**

O Empacotamento de Python compreende o processo de **criar pacotes ou bibliotecas reutilizáveis em Python**. Um pacote é um conjunto de **módulos** que podem ser facilmente importados e usados em diferentes projetos. O empacotamento é feito em uma estrutura hierárquica de diretórios.

Ao empacotar um código em Python, além da presença da **modularidade**, que permite **separar funcionalidades específicas em componentes individuais**, os pacotes criados podem ser **disponibilizados em repositórios públicos**, como o **PyPI (Python Package Index)**, e instalados usando ferramentas como o **pip**. Isso facilita a manutenção e disseminação do código, permitindo que outros desenvolvedores o utilizem com apenas alguns comandos.

**Criando e Empacotando um Pacote em Python:**

**1. Estrutura do Pacote:**

A primeira etapa é organizar a estrutura de diretórios do pacote. Geralmente, os pacotes em Python seguem uma estrutura padrão, com um diretório principal que leva o nome do pacote e contém os módulos e subpacotes.

**2. Definição de módulos:**

Em seguida, definimos os módulos. Cada módulo é um arquivo Python que contém o código correspondente a uma funcionalidade específica.

**3. Arquivo de configuração:**

Um arquivo de configuração, como o setup.py, é necessário para definir metadados sobre o pacote, como nome, versão e dependências e também para especificar as instruções de instalação do pacote.

**4. Empacotamento e distribuição:**

Por fim, é possível empacotar e distribuir o pacote. Isso pode ser feito gerando um arquivo de distribuição, como um arquivo .tar.gz ou .whl, que pode ser instalado em outros projetos.

**Fonte:** <https://awari.com.br/empacotamento-de-python-tudo-o-que-voce-precisa-saber-sobre-a-criacao-de-pacotes-em-python/?utm_source=blog&utm_campaign=projeto+blog&utm_medium=Empacotamento%20De%20Python:%20Tudo%20O%20Que%20Voc%C3%AA%20Precisa%20Saber%20Sobre%20A%20Cria%C3%A7%C3%A3o%20De%20Pacotes%20Em%20Python>

**Tutoriais (vídeos):**

<https://www.youtube.com/watch?v=obFYFaqTOAE>

<https://www.youtube.com/watch?v=qOYU46tBGIg>

Tutorial muito bom: <https://youtu.be/_14MS9oFZZI?si=iujfw2Pktuyk3twY>

Material de apoio deste tutorial: <https://doc.clickup.com/36904728/d/h/1367rr-543/246df89507b2ab3>