Aula 3

Linguagem de Programação Aplicada

Prof. Renan Portela Jorge

1

Conversa Inicial

2

O que estudaremos nessa aula?

- **Nessa aula veremos**
  - **Módulos, diretórios e pacotes**
  - **Classes abstratas e herança múltipla**
  - **Introdução a padrões de *design* de *software***

3

Módulos, diretórios e pacotes

4

Módulos

- **Módulo em Python**
  - **Arquivo com definições e declarações de classes, funções, variáveis**
  - **Cada arquivo Python com extensão ".py" pode ser tratado como um módulo**
  - **Os módulos fornecem uma forma conveniente de organizar o código**

5

Diretórios e Pacotes

- **Diretórios em Python: são pastas que podem conter um ou mais módulos ou até mesmo outros diretórios**
- **Pacote em Python: são diretórios que contêm um arquivo especial chamado "init.py". O conteúdo desse arquivo pode ser deixado em branco, mas sua presença indica que o diretório é um pacote Python válido**

6

Classe ABC (Abstract Base Class) e herança múltipla

7

Classe ABC

- Fornecida pelo módulo ABC, permite definir classes abstratas, que
  - Não podem ser instanciadas diretamente
  - Servem como um esqueleto ou modelo para outras classes que a estendem
  - Definem métodos abstratos que devem ser implementados pelas subclasses

8

Herança múltipla

- Fornecida pelo módulo ABC, permite definir classes abstratas, que
  - Não podem ser instanciadas diretamente
  - Servem como um esqueleto ou modelo para outras classes que a estendem
  - Definem métodos abstratos que devem ser implementados pelas subclasses

9

Introdução a padrões de *design* de *software*

10

O que são padrões de *design*? (*design pattern*)

- São padrões de projeto
- São soluções reutilizáveis para problemas recorrentes no desenvolvimento de *software*
- Representam abordagens testadas e comprovadas para projetar e estruturar o código de forma eficiente
- Há vários tipos de *design patterns*, e cada um foca num aspecto do dev. de *software*

11

- Padrões estruturais
  - Lidam com a organização de classes e objetos
  - Fornecem maneiras de compor estruturas maiores a partir de partes menores
    - Adapter
    - Proxy

12

- Padrões comportamentais
  - Lidam com a interação entre objetos e a definição de responsabilidades
  - Gerenciam comunicação e comportamento entre diferentes objetos
    - **Mediator**
    - **Observer**

13

- Padrões de criação
  - Lidam com a criação de objetos
  - Oferecem mecanismos flexíveis para a instanciação de classe
    - **Builder**
    - **Factory**

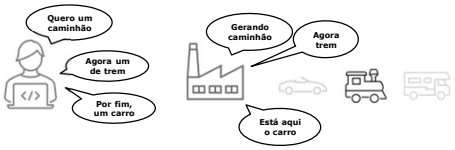
14

Padrão de *design* de criacional: método fábrica

15

Design pattern factory

- Cria objetos sem expor detalhes ao cliente
- O cliente solicita um objeto à *factory*, e ela se encarrega de retornar a instância desejada



16

Padrão de *design* de criacional: *builder*

17

Design pattern: builder

- Usado para criar objetos complexos passo a passo
- Separa a construção do objeto de sua representação
- Permite que o mesmo processo de construção crie diferentes representações

18