

Curso .NET

Unidade Curricular: Laboratório Web

Docente: David Jardim

FICHA DE TRABALHO 11

Exercícios:

1. Neste exercício, o objetivo é aplicar o conceito de Dependency Injection (DI) aplicado a classes. Crie um projeto do tipo (Console App .NET Framework)
 - a. Implemente uma classe abstrata para representar veículos. Um veículo é caracterizado pelos seguintes atributos: travel (tipo enumerado: LAND, WATER, AIR), color, weight, brand, model e engine. A classe Vehicle deve definir um método abstrato Start para ligar o veículo
 - b. Implemente uma classe para representar o motor dos veículos. A classe Engine deve ter como atributos, o binário (torque), a cilindrada (displacement) e a potência em cavalos (horsepower)
 - c. Implemente as classes Car e Motorcycle por herança da classe Vehicle
 - i. A classe Car deve ter informação sobre o número de portas e os lugares;
 - ii. A classe Motorcycle deve ter informação sobre o tipo (enumerado: SPORT, CRUISER, ADVENTURE) e a velocidade máxima;
 - iii. Ambas as classes devem sobrepor o método toString;
 - iv. Ambas as classes devem implementar os métodos abstratos;
 - d. Crie uma classe VehicleTest que tem como único atributo um objeto do tipo Car
 - i. Implemente o construtor por parâmetros
 - ii. Adicione uma propriedade que expõe o atributo da classe que representa o objeto Car
 - iii. Crie uma nova instância desta classe na função Main da classe Program usando o construtor por parâmetros
 - iv. No Main da classe program invoque o método Start da propriedade Car da classe VehicleTest
 - v. Que alterações são necessárias para alterar o objeto do tipo Car para Motorcycle?
 - e. Crie uma interface IVehicle que define um método abstrato Drive
 - i. As classes Car e Motorcycle devem implementar a interface IVehicle
 - f. Efetue as alterações necessárias na classe VehicleTest para alterar o tipo do atributo de Vehicle para IVehicle

2. Neste exercício, o objetivo é aplicar o conceito de Dependency Injection (DI) aplicado a Controllers ASP.NET Core ao projeto realizado na ficha de trabalho anterior (Ficha 10)
- a. Crie as interfaces necessárias para aplicar o conceito de DI aos controllers implementados
 - b. Implemente as interfaces nos respetivos modelos
 - i. Efetue as alterações necessárias para que a propriedade que representa a lista dos recursos seja agora definida na interface
 - c. Modifique as funções da classe JsonLoader para devolverem uma lista de recursos em vez de uma referência para a classe
 - d. No construtor por omissão dos modelos, inicialize a lista de recursos utilizando as funções implementadas na classe JsonLoader
 - e. Adicione a dependência da classe Employee como um serviço com o seguinte tempo de vida:
 - i. Transient
 - ii. Scoped
 - iii. Singleton
 - f. Qual a diferença entre cada um dos tempos de vida em execução? Pode utilizar breakpoints e fazer debug para verificar.