

Programação Orientada a Objetos (POO)

Hugo Pereira Monteiro

Nº 27993 – Regime Pós-laboral

Orientação

Ernesto Casanova

Ano letivo

2024-2025

Licenciatura em Engenharia de Sistemas Informáticos

Escola Superior de Tecnologia

Instituto Politécnico do Cávado e do Ave

Conteúdo

Índice de figuras	3
Introdução	4
Fase 1	5
Objetivos.....	5
Estrutura do Código	6
Organização do Código.....	6
Funcionalidades	7
Estrutura de classes	7
Implementação das classes.....	8
Construtores.....	8
Propriedades	8
Herança	9
Objetivos futuros	10
Conclusão	10

Índice de figuras

<i>Figura 1 Atributos classe Cliente</i>	<i>7</i>
<i>Figura 2 Construtores da classe Cliente.....</i>	<i>8</i>
<i>Figura 3 Propriedade da classe Cliente</i>	<i>8</i>
<i>Figura 4 Herança da classe registo para a classe Check_in</i>	<i>9</i>

Introdução

No âmbito da unidade curricular de **Programa Orientada a Objetos**, lecionada pelo docente Ernesto Casanova, foi proposto desenvolver um projeto em C# onde se pretendia soluções para problemas reais de complexidade moderada.

Optou-se pelo tema sugerido pelo docente, alojamentos turísticos, onde será implementado um sistema que permite a gestão de alojamentos turísticos.

Fase 1

A Fase 1, abrange três etapas principais no desenvolvimento do projeto. Em primeiro lugar, é feita a identificação da estrutura de classes, onde são definidas as classes essenciais do sistema, estabelecendo uma base sólida para a organização e funcionamento do código.

Seguidamente, procede-se à implementação essencial dessas classes, assegurando que as funcionalidades básicas sejam cumpridas. Finalmente, são escolhidas e organizadas as estruturas de dados a utilizar, garantindo que o armazenamento e a manipulação da informação estejam alinhados com as necessidades do projeto.

Objetivos

- Consolidar conceitos basilares do Paradigma Orientado a Objetos;
- Analisar problemas reais;
- Desenvolver capacidades de programação em C#;
- Potenciar a experiência no desenvolvimento de software;
- Assimilar o conteúdo da Unidade Curricular.

Estrutura do Código

Organização do Código

A estrutura do código está organizada em dois módulos, sendo que cada módulo possui uma função específica dentro do sistema.

O módulo principal, **‘Main’**, contém a função responsável pela inicialização e execução do programa. Este módulo coordena a execução, chamando as funções e métodos definidos no outro módulos, de forma a garantir o funcionamento do software.

O segundo módulo, **‘ObjetosNegocio’**, agrupa classes fundamentais para a lógica de negócio do sistema. Estas incluem classes como **‘Alojamento’**, **‘Check_in’**, **‘Check_out’**, **‘Cliente’**, **‘Consulta’**, **‘Registo’** e **‘Reserva’**. Cada uma destas classes guarda dados e funcionalidades específicas, permitindo a manipulação eficiente das operações relacionadas a cada entidade do sistema.

Funcionalidades

Estrutura de classes

Uma **estrutura de classes** é a organização de diferentes classes num sistema, onde cada classe representa uma entidade e esta contém atributos e métodos que definem o seu comportamento.

As **classes** servem para a criação de objetos, guardando os dados e funcionalidades relacionadas a uma entidade específica.

Na **figura 1**, uma das classes utilizados no sistema, a estrutura de classes inclui a classe **Cliente**, que possui atributos como **nome**, **cc**, **email**, **telemóvel**, **nif** e **id** (linha 27 a 32), estes representam as informações necessárias na criação do objeto cliente.

Esta classe contém também uma lista estática '**static List<Cliente> clientes**' (linha 34), que serve para guardar todos os clientes registados, posteriormente facilita na gestão e consulta de vários clientes.

```
24  public class Cliente
25  {
26      #region Attributes
27      private string nome;
28      private string cc;
29      private string email;
30      private string telemovel;
31      private string nif;
32      private int id;
33      private static int totalId;
34      static List<Cliente> clientes;
35      #endregion
```

Figura 1 Atributos classe Cliente

Implementação das classes

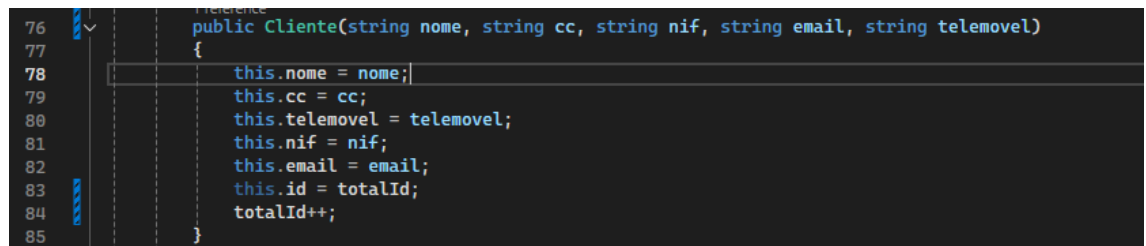
A **implementação das classes** é a criação da estrutura básica e funcionalidades principais das classes que compõem o sistema como os **construtores** e as **propriedades**.

Na **implementação das classes** define-se os métodos fundamentais para que a classe consiga fazer o seu trabalho no software.

Construtores

Os **Construtores** são utilizados para inicializar os objetos com valores iniciais.

Como podemos ver na **figura 2**, o objeto **Cliente** vai ser criado com os valores fornecidos, atribuindo os valores aos atributos correspondentes da classe.



```
76 public Cliente(string nome, string cc, string nif, string email, string telemovel)
77 {
78     this.nome = nome;
79     this.cc = cc;
80     this.telemovel = telemovel;
81     this.nif = nif;
82     this.email = email;
83     this.id = totalId;
84     totalId++;
85 }
```

Figura 2 Construtores da classe Cliente

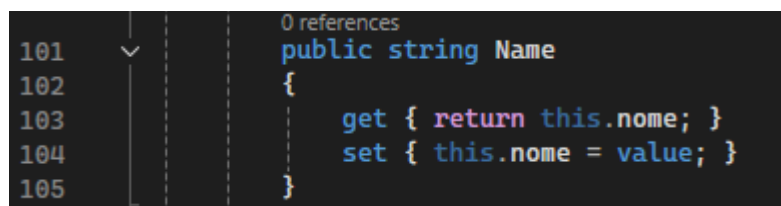
Propriedades

As **propriedades** têm a função de aceder e modificar valores de atributos privados de uma classe a partir dos métodos **get** e **set**.

- **Get** - Retorna o valor do atributo sendo este privado
- **Set** – Modifica o valor privado

Na **figura 3**, **'Name'** (linha 101) é a propriedade que controla o acesso ao campo privado **nome**.

Quando a propriedade é lida, o valor atual do **nome** vai ser retornado pelo método **get**, em seguida o método **set** vai modificar e guardar o novo valor no campo **nome**.



```
101 public string Name
102 {
103     get { return this.nome; }
104     set { this.nome = value; }
105 }
```

Figura 3 Propriedade da classe Cliente

Herança

A **herança** é um conceito fundamental na programação orientada a objetos que permite criar uma nova classe baseada em uma classe existente, aproveitando as propriedades e comportamentos da classe original. A classe recém-criada é chamada de classe derivada ou subclasse. A herança promove a reutilização de código, evitando a duplicação de implementações e facilitando a manutenção. Além disso, ela suporta a criação de hierarquias de classes, onde as classes mais específicas herdam características mais gerais.

Como se pode observar na figura 4, a classe recém-criada '**Check_in**' vai herdar os atributos, construtores, propriedades e métodos da classe **Registo**.

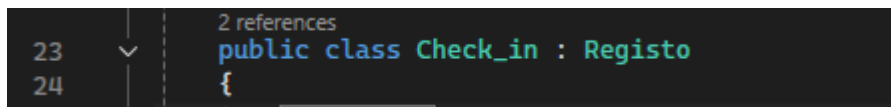
A screenshot of a code editor with a dark background. On the left, line numbers 23 and 24 are visible. A vertical dashed line separates the line numbers from the code. To the right of the line, the text '2 references' is displayed in a light blue font. Below it, the code 'public class Check_in : Registo' is written in a light blue font, followed by an opening curly brace '{' on the next line. A small downward-pointing arrow is visible to the left of the code line.

Figura 4 Herança da classe registo para a classe Check_in

Objetivos futuros

- Melhorar os conhecimentos na linguagem de programação C#.

Conclusão

Em suma, a busca pela aprendizagem e aprimoramento dos conhecimentos na linguagem C#, revelou-se uma jornada enriquecedora. Um programa desta magnitude na linguagem C# proporcionou uma compreensão sólida dos fundamentos da programação.