IFRN

PROGRAMAÇÃO ORIENTADA A OBJETOS EM C#

Encapsulamento

Prof. Gilbert Azevedo

Objetivos

- Entender o conceito de Encapsulamento
- Utilizar modificadores de acesso
- Refinar o modelo do triângulo no C#
- Compreender os conceitos de Estado, Comportamento e Identidade
- Utilizar o Diagrama de Classes para representar graficamente uma Classe

Modelo de um Sistema Simples

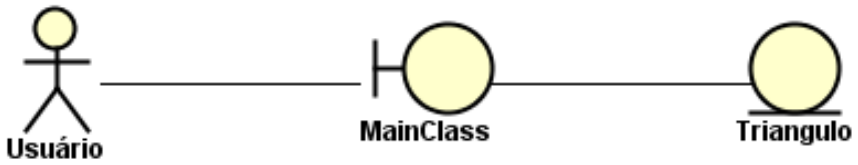
- O modelo de um sistema simples é composto por uma classe de interface com o usuário e uma classe de entidade
- A classe que executa o método “Main”, onde fica a entrada e saída de dados, é a interface com o usuário (UI – *User Interface*)
- A classe da entidade representa algo do mundo real que foi modelado, sendo usada para solucionar algum problema
- É uma boa prática de programação não conectar o usuário com a entidade diretamente

Modelo de um Sistema Simples

- No exemplo, MainClass é a classe de UI, Triangulo é a entidade

```
class MainClass {
  public static void Main () {
    Triangulo x = new Triangulo();
    x.b = 10;
    x.h = 20;
    double area = x.CalcArea();
    Console.WriteLine(area);
  }
}
```

```
class Triangulo {
  public double b, h;
  public double CalcArea() {
    return b * h / 2;
  }
}
```



Encapsulamento

- O encapsulamento é um conceito chave da POO
- O objetivo é evitar que um objeto armazene valores indevidos, como no exemplo: as dimensões de um triângulo não podem ser negativas
- Quando isso ocorre, o objeto está com um “Estado Inválido”

```
class MainClass {
  public static void Main () {
    Triangulo x = new Triangulo();
    x.b = -10;
    x.h = -20;
    ...
  }
}
```

```
class Triangulo {
  public double b, h;
  public double CalcArea() {
    return b * h / 2;
  }
}
```

Encapsulamento

- Para evitar o “estado inválido”, os atributos do objeto não devem ser visíveis para a interface com o usuário; eles são encapsulados.
- O encapsulamento é feito mudando a visibilidade dos atributos para privado (*private*), conforme abaixo.
- Com isso, a MainClass perde o acesso aos atributos.

```
x.b = -10;
x.h = -20;
```

- Apenas na classe Triangulo, os atributos b e h podem ser usados.

```
class Triangulo {
  private double b, h;
  public double CalcArea() {
    return b * h / 2;
  }
}
```

Métodos de Acesso

- Métodos de acesso (gets e sets) são usados então para recuperar e atribuir valores aos atributos
- Métodos “sets” são usados para definir os valores dos atributos, devendo validar esses valores
- Métodos “gets” são usados para retornar os valores armazenados
- O estado do objeto, no exemplo, é garantido porque não é possível armazenar um valor negativo

```
class Triangulo {
  private double b, h;
  public void SetBase(double v) {
    if (v > 0) b = v; }
  public void SetAltura(double v) {
    if (v > 0) h = v; }
  public double GetBase() {
    return b; }
  public double GetAltura() {
    return h; }
  public double CalcArea() {
    return b * h / 2; }
}
```

Métodos de Acesso

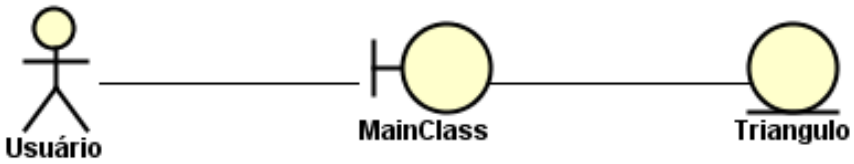
- Os métodos de acesso podem ser chamados a partir da UI
- Se um valor negativo for informado, ele não é armazenado

```
class MainClass {
  public static void Main () {
    Triangulo x = new Triangulo();
    x.SetBase(10);
    x.SetAltura(20);
    ...
  }
}
```

```
class Triangulo {
  private double b, h;
  public void SetBase(double v) {
    if (v > 0) b = v; }
  public void SetAltura(double v) {
    if (v > 0) h = v; }
  public double GetBase() {
    return b; }
  public double GetAltura() {
    return h; }
  public double CalcArea() {
    return b * h / 2; }
}
```

Modificadores de Acesso

- Em resumo: modificadores de acesso são usados para controlar a visibilidade de membros de uma classe
  - private: indica que o membro é visível apenas dentro da própria classe
  - public: indica que o membro é visível por todas as classe do sistema
- Geralmente, os atributos de entidades são privados e os métodos são públicos
  - No exemplo, a MainClass não pode acessar os atributos da classe Triangulo, apenas os métodos



Modelo Final do Triângulo

- O modelo OO do triângulo define então dois atributos privados (encapsulados)
- Quatro métodos de acesso públicos, usados para acessar os atributos
- E um método público para realizar a operação de cálculo da área

```
class Triangulo {
  private double b, h;
  public void SetBase(double v) {
    if (v > 0) b = v; }
  public void SetAltura(double v) {
    if (v > 0) h = v; }
  public double GetBase() {
    return b; }
  public double GetAltura() {
    return h; }
  public double CalcArea() {
    return b * h / 2; }
}
```

Versão Final da UI

- A classe de UI instancia um objeto Triangulo
- Acessa os métodos para definir suas dimensões
- E chama a operação que calcula sua área
- Somente as classe de UI devem utilizar a classe Console para realizar a comunicação com o usuário

```
class MainClass {
  public static void Main () {
    Triangulo x = new Triangulo();
    x.SetBase(10);
    x.SetAltura(20);
    double area = x.CalcArea();
    Console.WriteLine(area);
  }
}
```

Estado, Comportamento e Identidade

- Quando uma classe é instanciada, um objeto é criado na memória e passa a existir. O objeto criado ganha uma Identidade.

```
class MainClass {
  public static void Main () {
    Triangulo x = new Triangulo();
    Triangulo y = new Triangulo();
  }
}
```

- Dois objetos foram criados no código acima. Eles ficam em locais distintos da memória sendo referenciados por *x* e *y*. Cada um deles tem uma identidade diferente.

Estado, Comportamento e Identidade

- Os dados armazenados por um objeto determina o seu Estado.
- Na classe Triangulo, o estado é o valor dos atributos b e h.
- Objetos diferentes podem ter o mesmo estado, embora tenham diferentes identidades.
- As operações realizadas por um objeto define seu comportamento.
- Na classe Triangulo, os métodos set/get e CalcArea definem o comportamento dos objetos da classe.
- Todos os objetos da classe possuem o mesmo comportamento.

Resumo: Classe

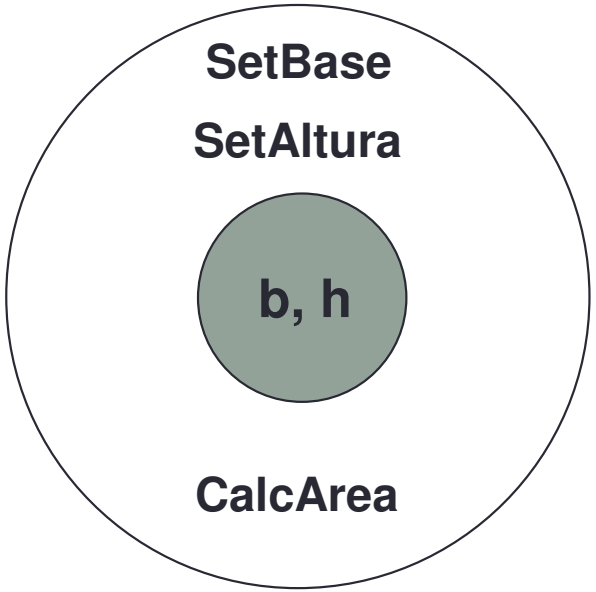
- Classe
  - É a unidade de software da POO que descreve uma entidade
  - Lista as características da entidade (Atributos)
  - Lista o comportamento da entidade (Métodos)
  - Define um novo tipo de variável

Resumo: Objeto

- Objeto
  - É uma instância (ou objeto) da classe
  - Armazena valores em cada atributo – Estado
  - Executa as funções definidas nos métodos – Comportamento
  - Ocupa um local de memória que o identifica unicamente em relação aos outros objetos – Identidade
  - É sempre acessado por uma referência

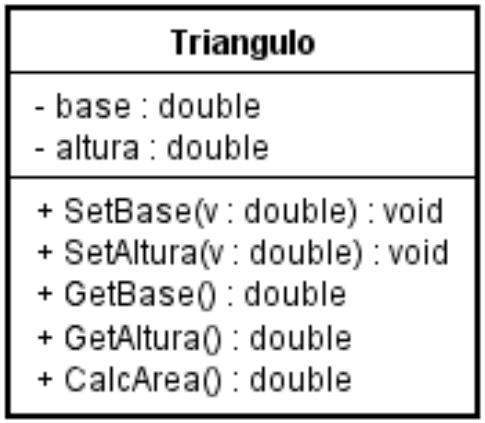
Representação Gráfica

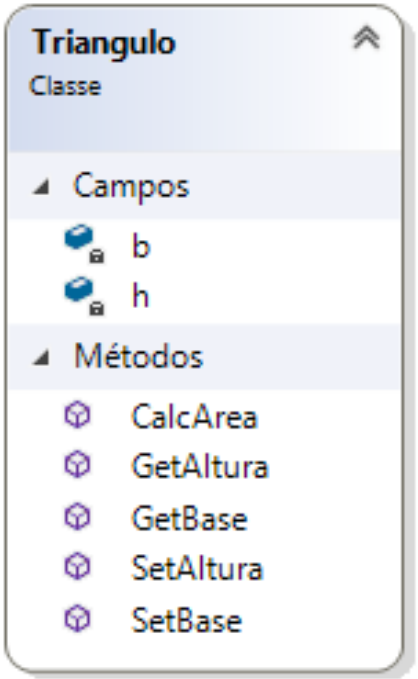
- A classe pode ser representada graficamente por uma rosquinha (“donut”) com os membros encapsulados no centro e os públicos na sua periferia
- Donut da classe Triângulo



Representação em UML

- A UML (Linguagem de Modelagem Unificada) é largamente utilizada na modelagem de sistemas orientados a objetos
- Diagrama de Classe: Apresenta as classes de um sistema de informação





Referências

- Microsoft Visual C# 2010 – Passo a passo, John Sharp, Bookman, 2010
- Classes (Guia de Programação em C#)
  - https://docs.microsoft.com/pt-br/dotnet/csharp/programming-guide/classes-and-structs/classes