



PROGRAMAÇÃO O.O. (C#)



Detalhes sobre variáveis
Professor: João Luiz Lagôas

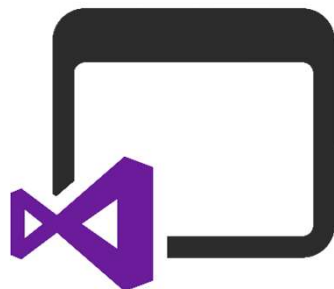


Jogo Vingadores

(motivação)



- O programa é composto por três classes



Solução/Projeto



Programa.cs (onde está o método Main())



Heroi.cs



Batalha.cs

Exemplo Prático

Classe



```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano = False;
        private int Vida = 6;
        private int Ataque = 1;
        private int Defesa = 1;
        private int Velocidade = 1;
        public static int Quantidade = 0;

        public Heroi(string n, bool eH, int at, int def, int vel)
        {
            Nome = n;
            EhHumano = eH;
            SetAtaque(at); SetDefesa(def); SetVelocidade(vel);
            Quantidade++;
        }

        // Métodos
        public string DescreverDados()
        {
            string descricao = "Nome: " + Nome + "\nEhHumano: " + EhHumano +
                               "\nVida: " + Vida + " \nAtaque: " + Ataque +
                               "\nDefesa: " + Defesa + "\nVelocidade: " + Velocidade;

            return descricao;
        }
        // outros métodos.....
    }
}
```

Exemplo Prático

Classe



```
namespace Aula05
{
    class Batalha
    {
        private Heroi Combatente1;
        private Heroi Combatente2;

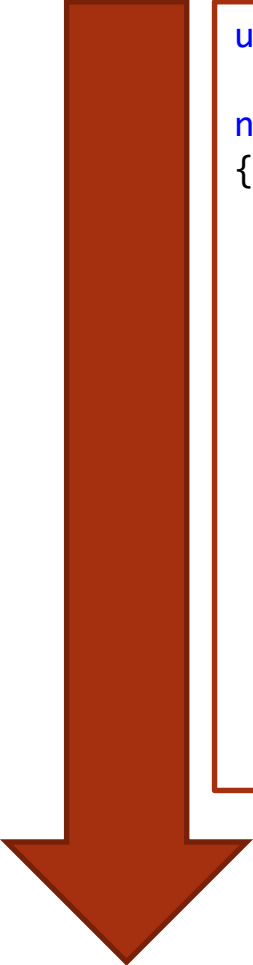
        public Batalha(Heroi h1, Heroi h2)
        {
            Combatente1 = h1;
            Combatente2 = h2;
        }

        public void Combater(string decisao1, string decisao2)
        {
            :
        }

        public bool CombatentesEstaoVivos()
        {
            if (Combatente1.GetVida() <= 0 || Combatente2.GetVida() <= 0)
                return false;
            else
                return true;
        }
    }
}
```

Exemplo Prático

Prático (Classe Programa)



```
using System;

namespace Aula05
{
    class Program
    {
        static void Main(string[] args)
        {
            Heroi sam = new Heroi("Falcão",
                                true, 2, 2, 5);
            Heroi bucky = new Heroi("Soldado Invernal",
                                   true, 4, 3, 3);

            Batalha arena = new Batalha(sam, bucky);

            •
            •
            •
        }
    }
}
```

Exemplo Prático

Prático (Classe Programa)



```
using System;

namespace Aula05
{
    class Program
    {
        static void Main(string[] args)
        {
            Heroi sam = new Heroi("Falcão",
                                true, 2, 2, 5);
            Heroi bucky = new Heroi("Soldado Invernal",
                                   true, 4, 3, 3);

            Batalha arena = new Batalha(sam, bucky);

            •
            •
            •
        }
    }
}
```

Exemplo Prático

Prático (Classe Programa)



sam



bucky

```
using System;

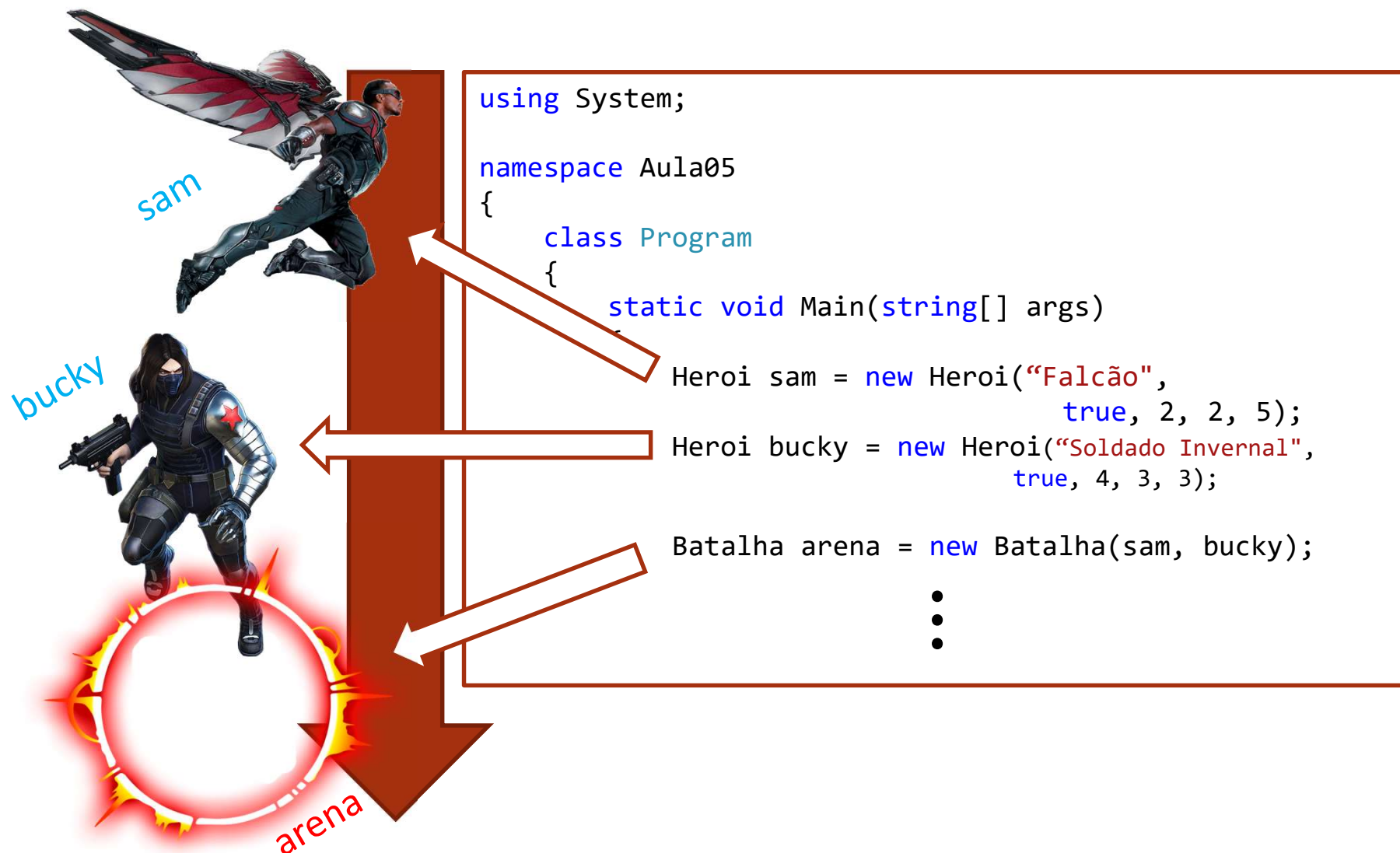
namespace Aula05
{
    class Program
    {
        static void Main(string[] args)
        {
            Heroi sam = new Heroi("Falcão",
                                true, 2, 2, 5);
            Heroi bucky = new Heroi("Soldado Invernal",
                                   true, 4, 3, 3);

            Batalha arena = new Batalha(sam, bucky);

            •
            •
            •
        }
    }
}
```

Exemplo Prático

Prático (Classe Programa)



Exemplo Prático

Prático (Classe Programa)



```
using System;

namespace Aula05
{
    class Program
    {
        static void Main(string[] args)
        {
            Heroi sam = new Heroi("Falcão",
                                true, 2, 2, 5);
            Heroi bucky = new Heroi("Soldado Invernal",
                                   true, 4, 3, 3);

            Batalha arena = new Batalha(sam, bucky);
            ...
        }
    }
}
```

Exemplo Prático

Execução do código...



```
Batalha arena = new Batalha(sam, bucky);

while(arena.CombatentesEstaoVivos())
{
    Console.WriteLine("Novo Combate: ");

    Console.WriteLine("1. " + sam.DescreverDados() + "\n");
    Console.WriteLine("2. " + bucky.DescreverDados() + "\n");
    Console.WriteLine("Combatente 1, faça seu movimento...");
    string decisao1 = Console.ReadLine();
    Console.Clear();

    Console.WriteLine("1. " + sam.DescreverDados() + "\n");
    Console.WriteLine("2. " + bucky.DescreverDados() + "\n");
    Console.WriteLine("Combatente 2, faça seu movimento...");
    string decisao2 = Console.ReadLine();
    Console.Clear();

    arena.Combater(decisao1, decisao2);
}

Console.WriteLine("Batalha encerrada!");

Console.ReadLine();
}
}
```

Observando os atributos, o que há de diferente?

```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano = False;
        private int Vida = 6;
        private int Ataque = 1;
        private int Defesa = 1;
        private int Velocidade = 1;
        public static int Quantidade = 0;

        public Heroi(string n, bool eH, int at, int def, int vel)
        {
            Nome = n;
            EhHumano = eH;
            SetAtaque(at); SetDefesa(def); SetVelocidade(vel);
            Quantidade++;
        }

        // Métodos
        public string DescreverDados()
        {
            string descricao = "Nome: " + Nome + "\nEhHumano: " + EhHumano +
                               "\nVida: " + Vida + " \nAtaque: " + Ataque +
                               "\nDefesa: " + Defesa + "\nVelocidade: " + Velocidade;

            return descricao;
        }

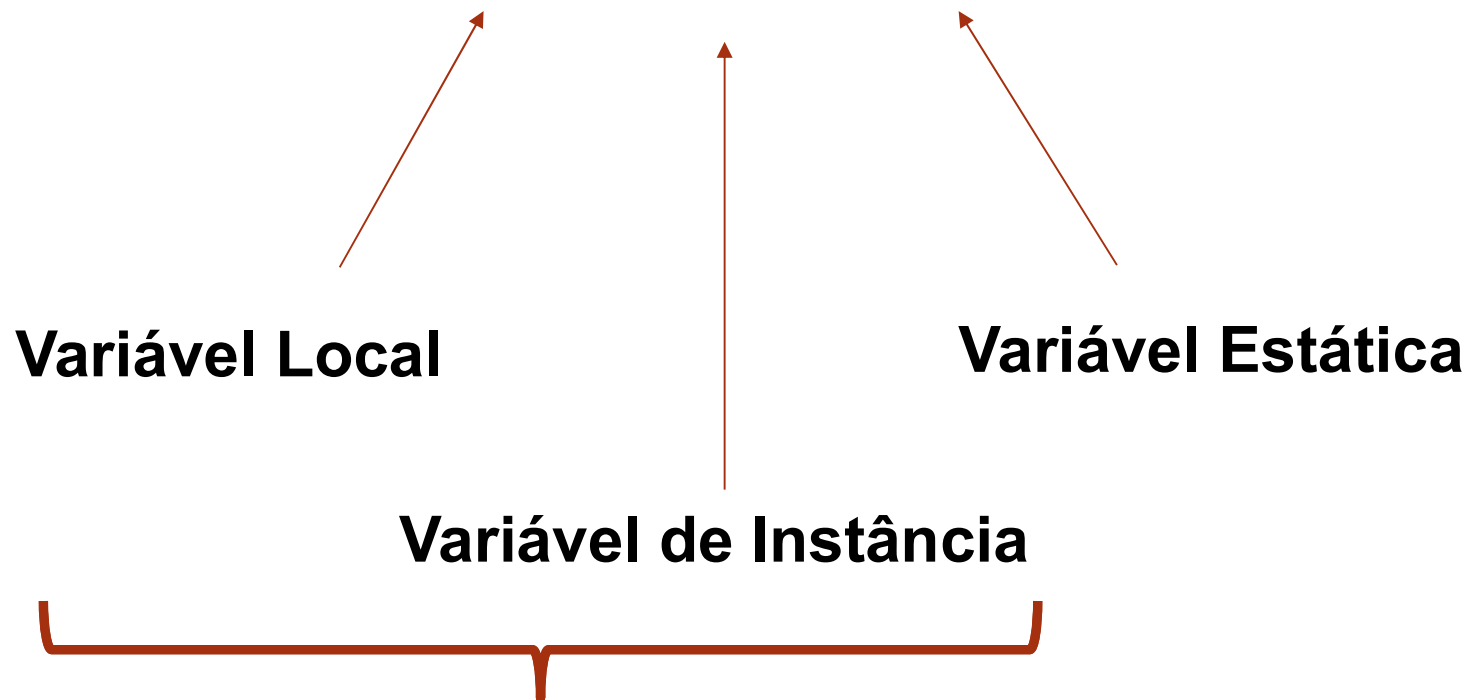
        // outros métodos.....
    }
}
```



Classificação de variáveis



Variáveis



Classificação de variáveis



- Variáveis declaradas como atributos de uma classe são denominadas **variáveis de instância**. Sabemos que cada objeto (instância) de uma classe tem um conjunto de atributos (um conjunto de variáveis de instância).
- Variáveis declaradas no corpo de um método são denominadas **variáveis locais**. Elas só existem naquele local/escopo onde foram declaradas.

Exemplo Prático

Variável de Instância



```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano = False;
        private int Vida = 6;
        private int Ataque = 1;
        private int Defesa = 1;
        private int Velocidade = 1;
        public static int Quantidade = 0;

        public Heroi(string n, bool eH, int at, int def, int vel)
        {
            Nome = n;
            EhHumano = eH;
            SetAtaque(at); SetDefesa(def); SetVelocidade(vel);
            Quantidade++;
        }

        // Métodos
        public string DescreverDados()
        {
            string descricao = "Nome: " + Nome + "\nEhHumano: " + EhHumano +
                               "\nVida: " + Vida + " \nAtaque: " + Ataque +
                               "\nDefesa: " + Defesa + "\nVelocidade: " + Velocidade;

            return descricao;
        }

        // outros métodos.....
    }
}
```

Exemplo Prático

Variável de Instância



Variáveis
de Instância

```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano = False;
        private int Vida = 6;
        private int Ataque = 1;
        private int Defesa = 1;
        private int Velocidade = 1;
        public static int Quantidade = 0;

        public Heroi(string n, bool eH, int at, int def, int vel)
        {
            Nome = n;
            EhHumano = eH;
            SetAtaque(at); SetDefesa(def); SetVelocidade(vel);
            Quantidade++;
        }

        // Métodos
        public string DescreverDados()
        {
            string descricao = "Nome: " + Nome + "\nEhHumano: " + EhHumano +
                               "\nVida: " + Vida + " \nAtaque: " + Ataque +
                               "\nDefesa: " + Defesa + "\nVelocidade: " + Velocidade;

            return descricao;
        }

        // outros métodos.....
    }
}
```

Exemplo Prático

Variável Local



```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano = False;
        private int Vida = 6;
        private int Ataque = 1;
        private int Defesa = 1;
        private int Velocidade = 1;
        public static int Quantidade = 0;

        public Heroi(string n, bool eH, int at, int def, int vel)
        {
            Nome = n;
            EhHumano = eH;
            SetAtaque(at); SetDefesa(def); SetVelocidade(vel);
            Quantidade++;
        }

        // Métodos
        public string DescreverDados()
        {
            string descricao = "Nome: " + Nome + "\nEhHumano: " + EhHumano +
                               "\nVida: " + Vida + "\nAtaque: " + Ataque +
                               "\nDefesa: " + Defesa + "\nVelocidade: " + Velocidade;

            return descricao;
        }

        // outros métodos.....
    }
}
```


Exemplo Prático

Variável Local



```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano = False;
        private int Vida = 6;
        private int Ataque = 1;
        private int Defesa = 1;
        private int Velocidade = 1;
        public static int Quantidade = 0;

        public Heroi(string n, bool eH, int at, int def, int vel)
        {
            Nome = n;
            EhHumano = eH;
            SetAtaque(at); SetDefesa(def); SetVelocidade(vel);
            Quantidade++;
        }

        // Métodos
        public string DescreverDados()
        {
            string descricao = "Nome: " + Nome + "\nEhHumano: " + EhHumano +
                               "\nVida: " + Vida + "\nAtaque: " + Ataque +
                               "\nDefesa: " + Defesa + "\nVelocidade: " + Velocidade;

            return descricao;
        }

        // outros métodos.....
    }
}
```

Variáveis Locais

Variável Local

Exemplo Prático

Valores default



```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano = False;
        private int Vida = 6;
        private int Ataque = 1;
        private int Defesa = 1;
        private int Velocidade = 1;
        public static int Quantidade = 0;

        public Heroi(string n, bool eH, int at, int def, int vel)
        {
            Nome = n;
            EhHumano = eH;
            SetAtaque(at); SetDefesa(at); SetVelocidade(vel);
            Quantidade++;
        }

        // Métodos
        public string DescreverDados()
        {
            string descricao = "Nome: " + Nome + "\nEhHumano: " + EhHumano +
                               "\nVida: " + Vida + "\nAtaque: " + Ataque +
                               "\nDefesa: " + Defesa + "\nVelocidade: " + Velocidade;

            return descricao;
        }

        // outros métodos.....
    }
}
```

E esses valores predefinidos?

Valores default

Variáveis de Instância



Normalmente, variáveis de instância são inicializadas com valores default

1. **0** para tipos numéricos (int, float, double, decimal, etc),
2. **false** para o tipo booleano (bool)
3. **null** para tipos de referência.

Exemplo Prático

Variável de Instância



```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano;
        private int Vida;
        private int Ataque;
        private int Defesa;
        private int Velocidade;
        public static int Quantidade;

        // Construtor que não faz nada
        public Heroi()
        {
        }

        // Métodos...
    }
}
```

Quais serão os valores os atributos de uma instância de Heroi?

Exemplo Prático

Variável de Instância



```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano;
        private int Vida;
        private int Ataque;
        private int Defesa;
        private int Velocidade;
        public static int Quantidade;

        // Construtor que não faz nada
        public Heroi()
        {
        }

        // Métodos...
    }
}
```

```
using System;

namespace Aula05
{
    class Program
    {
        static void Main(string[] args)
        {
            Heroi tony = new Heroi();

            •
            •
            •
        }
    }
}
```

Exemplo Prático

Variável de Instância



```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano;
        private int Vida;
        private int Ataque;
        private int Defesa;
        private int Velocidade;
        public static int Quantidade;

        // Construtor que não faz nada
        public Heroi()
        {
        }
    }
}
```

Nome do objeto:

tony

Valor dos atributos:

Nome = null

EhHumano = false

Vida = 0

Ataque = 0

Defesa = 0

Velocidade = 0

```
using System;

namespace Aula05
{
    class Program
    {
        static void Main(string[] args)
        {
            Heroi sam = new Heroi();

            •
            •
            •
        }
    }
}
```

Valores default

Variáveis de Instância

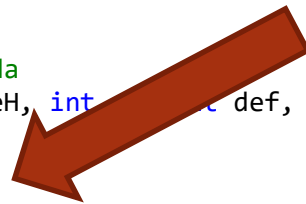


Sabendo disso, é comum definirmos no construtor quais serão os valores padrão das minhas variáveis de instância ou se eu irei inicializa-las com um valor passado como argumento.

```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano;
        private int Vida;
        private int Ataque;
        private int Defesa;
        private int Velocidade;
        public static int Quantidade;

        // Construtor que não faz nada
        public Heroi(string n, bool eH, int at, int def, int vel)
        {
            Nome = n;
            EhHumano = eH;
            SetAtaque(at); SetDefesa(def); SetVelocidade(vel);
            Quantidade++;
        }
        // Métodos...
    }
}
```

As variáveis de instância
serão inicializadas com os
valores passados pelo
construtor neste caso.



Valores default

Variáveis de Instância



Mas se quisermos determinar de forma fácil valores default, basta realizar a atribuição na declaração dos atributos.

```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano;
        private int Vida = 6;
        private int Ataque = 1;
        private int Defesa = 1;
        private int Velocidade = 1;
        public static int Quantidade = 0;

        // Construtor que não faz nada
        public Heroi(string n, bool eH, int at, int def, int vel)
        {
            Nome = n;
            EhHumano = eH;
            SetAtaque(at); SetDefesa(def); SetVelocidade(vel);
            Quantidade++;
        }
        // Métodos...
    }
}
```


Valores default

Variáveis Locais



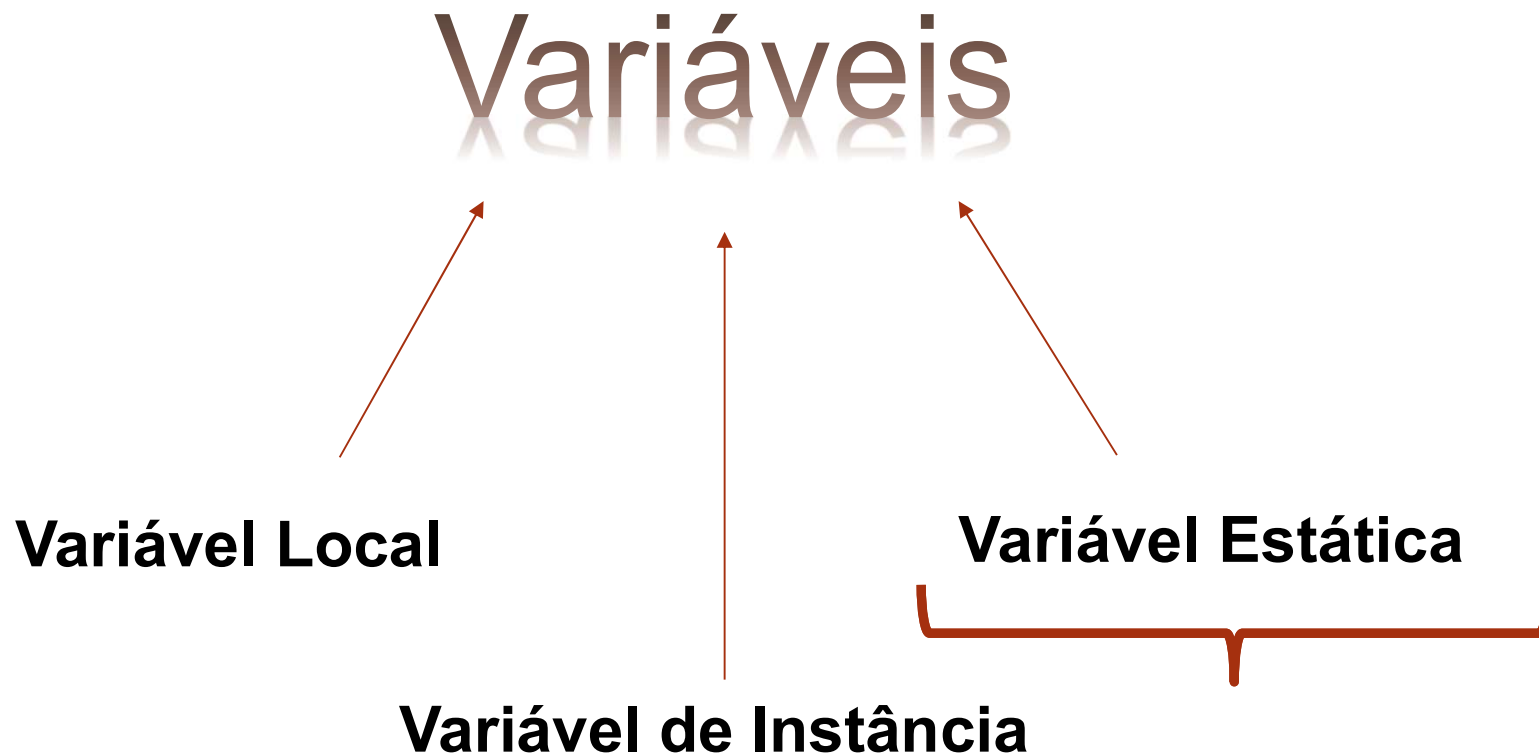
Já as variáveis locais não são inicializadas com valores default! O Visual Studio irá indicar um erro caso você tente usar uma variável local que nunca recebeu um valor!

```
class Program
{
    static void Main(string[] args)
    {
        int x;

        Console.WriteLine(x);
    }
}
```

Erro: uso de variável não atribuída!

Classificação de variáveis



Classificação de variáveis



- Existe um outro tipo de classificação para variáveis denominado **variáveis estáticas** (*static*), onde todos os objetos de uma mesma classe compartilham uma variável.
- Dizemos que uma variável estática é uma **variável de classe**.

Variáveis Estáticas



- Todo objeto tem sua própria cópia das variáveis de instância da sua classe. Em alguns casos, **apenas uma cópia de uma variável** em particular deve ser **compartilhada por todos os objetos da classe**.
- Uma variável **static** é usada nesses casos. Uma variável estática representa uma informação da classe como um todo (e NÃO de apenas um objeto). **Todos** os objetos dessa classe compartilham a mesma variável estática.

Exemplo Prático

Variável Estática



```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano = False;
        private int Vida = 6;
        private int Ataque = 1;
        private int Defesa = 1;
        private int Velocidade = 1;
        public static int Quantidade = 0;

        public Heroi(string n, bool eH, int at, int def, int vel)
        {
            Nome = n;
            EhHumano = eH;
            SetAtaque(at); SetDefesa(at); SetVelocidade(vel);
            Quantidade++;
        }

        // Métodos
        public string DescreverDados()
        {
            string descricao = "Nome: " + Nome + "\nEhHumano: " + EhHumano +
                               "\nVida: " + Vida + "\nAtaque: " + Ataque +
                               "\nDefesa: " + Defesa + "\nVelocidade: " + Velocidade;

            return descricao;
        }

        // outros métodos.....
    }
}
```

Exemplo Prático

Variável Estática



Variável Estática

```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano = False;
        private int Vida = 6;
        private int Ataque = 1;
        private int Defesa = 1;
        private int Velocidade = 1;
        public static int Quantidade = 0;

        public Heroi(string n, bool eH, int at, int def, int vel)
        {
            Nome = n;
            EhHumano = eH;
            SetAtaque(at); SetDefesa(at); SetVelocidade(vel);
            Quantidade++;
        }

        // Métodos
        public string DescreverDados()
        {
            string descricao = "Nome: " + Nome + "\nEhHumano: " + EhHumano +
                               "\nVida: " + Vida + "\nAtaque: " + Ataque +
                               "\nDefesa: " + Defesa + "\nVelocidade: " + Velocidade;

            return descricao;
        }

        // outros métodos.....
    }
}
```

Exemplo Prático

Variáveis Estáticas



```
namespace Aula05
{
    class Heroi
    {
        public string Nome;
        public bool EhHumano = False;
        private int Vida = 6;
        private int Ataque = 1;
        private int Defesa = 1;
        private int Velocidade = 1;
        public static int Quantidade = 0;

        public Heroi(string n, bool eH, int at, int def, int vel)
        {
            Nome = n;
            EhHumano = eH;
            SetAtaque(at); SetDefesa(def); SetVelocidade(vel);
            Quantidade++;

        }

        // Métodos
        public string DescreverDados()
        {
            string descricao = "Nome: " + Nome + "\nEhHumano: " + EhHumano +
                               "\nVida: " + Vida + " \nAtaque: " + Ataque +
                               "\nDefesa: " + Defesa + "\nVelocidade: " + Velocidade;

            return descricao;
        }
        // outros métodos.....
    }
}
```

Valor default é 0

Sempre que um construtor é chamado ele incrementa a variável Quantidade.

Exemplo Prático

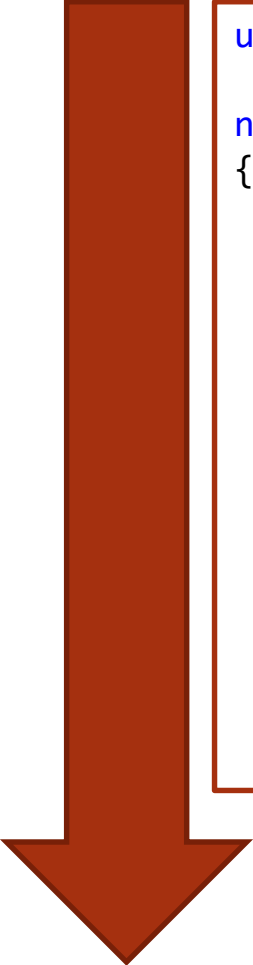
Variáveis Estáticas



Variáveis Estáticas

Quantidade = 0

Pertence à classe: Heroi



```
using System;

namespace Aula05
{
    class Program
    {
        static void Main(string[] args)
        {
            Heroi sam = new Heroi("Falcão",
                                true, 2, 2, 5);
            Heroi bucky = new Heroi("Soldado Invernal",
                                   true, 4, 3, 3);

            Console.WriteLine(Heroi.Quantidade);

            •
            •
            •
        }
    }
}
```


Exemplo Prático

Variáveis Estáticas



Nome do objeto:

sam

Valor dos atributos:

Nome = "Falcão"

EhHumano = true

Vida = 6

Ataque = 2

Defesa = 2

Velocidade = 5



Variáveis Estáticas

Quantidade = 1

Pertence à classe: Heroi

Aconteceu o incremento
da variável estática!

Aula05

Program

```
static void Main(string[] args)
{
    Heroi sam = new Heroi("Falcão",
                           true, 2, 2, 5);
    Heroi bucky = new Heroi("Soldado Invernal",
                           true, 4, 3, 3);

    Console.WriteLine(Heroi.Quantidade);

    ...
}
```

Exemplo Prático

Variáveis Estáticas



Nome do objeto:

sam

Valor dos atributos:

Nome = "Falcão"

EhHumano = true

Vida = 6

Ataque =

Defesa =

Velocidade =

Aconteceu o incremento
da variável estática outra
vez!

Nome do objeto:

bucky

Valor dos atributos:

Nome = "Soldado Invernal"

EhHumano = true

Vida = 6

Ataque = 4

Defesa = 3

Velocidade = 3

Variáveis Estáticas

Quantidade = 2

Pertence à classe: Heroi

Aula05

Program

```
static void Main(string[] args)
```

```
Heroi sam = new Heroi("Falcão",  
                        true, 2, 2, 5);
```

```
Heroi bucky = new Heroi("Soldado Invernal",  
                        true, 4, 3, 3);
```

```
Console.WriteLine(Heroi.Quantidade);
```

•
•
•



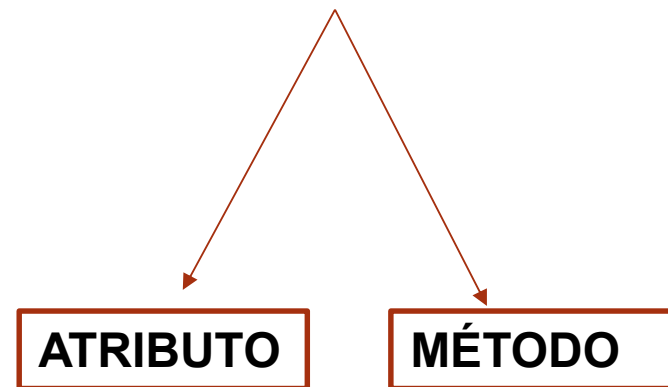
Métodos Estáticos



Assim como atributos/variáveis estáticos existem independentemente de um objeto, métodos estáticos também podem existir (podem ser usados) sem precisar de um objeto!

- Apesar da maioria dos métodos serem executados através de objetos em resposta a chamadas de métodos, isso não é necessariamente sempre o caso.
- Tais métodos são aplicáveis à classe onde é declarado como um todo e são chamados de métodos estáticos.
- Não é incomum para uma classe conter um grupo de métodos estáticos para realizar um trabalho.

static



Não precisam de uma instância para serem usados!

Exemplo

Métodos Estáticos



- Até aqui já utilizamos vários métodos estáticos! Observe o exemplo:

Nome da
classe

Math.Pow(2, 4)

Chamada
ao método

- Você pode chamar qualquer método estático especificando o nome da classe onde o método foi declarado, seguido pelo operador de acesso a membro (.) e o nome do método como.

Console.ReadLine()

Convert.ToInt32("78")

Console.WriteLine("LP2 é irado")

Convert.ToDouble("2")

Exemplo

Métodos Estáticos



- Nós usamos vários métodos estáticos das classes **Math**, **Console** e **Convert**.
- A classe **Math**, por exemplo, provê uma coleção de métodos que te habilitam a realizar cálculos comuns matemáticos.

Mas como o nosso programa entende que esses comandos já programados `Math.Pow()`, `Math.Sqrt()`, `Console.WriteLine()`, etc, podem ser usados??

The .NET Framework Class Library



- Muitas classes predefinidas estão agrupadas em categorias de classes relacionadas chamadas de **namespaces**. Juntos, esses namespaces são referidos como .NET Framework Class Library (**FCL**).
- Através de diretivas precedidas pela palavra reservada **using**, especificamos qual namespace queremos que esteja presente em nosso projeto.

```
using System;
```

→ Diretiva para importar todas as classes que estão no namespace **System**

No site: [https://msdn.microsoft.com/pt-br/library/system\(v=vs.110\).aspx](https://msdn.microsoft.com/pt-br/library/system(v=vs.110).aspx)
há uma referência completa de todas as classes presentes no namespace System

namespace System



- Até agora já utilizamos algumas das classes presentes no namespace System, são elas:

	Console	Representa os fluxos de entrada, saída e erro padrão para aplicativos de console.
	Convert	Converte um tipo de dados base em outro tipo de dados base.
	Math	Fornece constantes e métodos estáticos para trigonométricas, logarítmicas e outras funções matemáticas comuns.

namespace
System