## Modificadores de Escopo em C#

Os modificadores de escopo definem a visibilidade e acessibilidade de classes, métodos, propriedades, etc., no código C#. Eles são essenciais para o encapsulamento e a organização do código, garantindo que os detalhes de implementação fiquem ocultos e que apenas as interfaces necessárias sejam expostas.

### Principais Modificadores de Escopo

1. **public**: O membro é acessível de qualquer lugar.
2. **private**: O membro é acessível apenas dentro da classe onde é declarado.
3. **protected**: O membro é acessível dentro da classe onde é declarado e nas classes derivadas.
4. **internal**: O membro é acessível apenas dentro do mesmo assembly.
5. **protected internal**: O membro é acessível dentro do mesmo assembly e nas classes derivadas.

## Exemplo Completo de Modificadores de Escopo

```
using System;

namespace Empresa
{
    public abstract class Funcionario
    {
        // Propriedade pública
        public string Nome { get; set; }

        // Propriedade interna (acessível dentro do mesmo assembly)
        internal int Id { get; set; }

        // Propriedade protegida
        protected decimal Salario { get; set; }

        // Propriedade protegida interna (acessível dentro do mesmo assembly e classes derivadas)
        protected internal string Departamento { get; set; }

        public Funcionario(string nome, int id, decimal salario, string departamento)
        {
            Nome = nome;
            Id = id;
            Salario = salario;
            Departamento = departamento;
        }

        // Método público abstrato
        public abstract void Trabalhar();

        // Método protegido
        protected void ExibirSalario()
        {
            Console.WriteLine($"Salário: {Salario}");
        }
```

```
    }

    public class Desenvolvedor : Funcionario
    {
        // Propriedade pública
        public string LinguagemDeProgramacao { get; set; }

        public Desenvolvedor(string nome, int id, decimal salario, string
departamento, string linguagemDeProgramacao)
            : base(nome, id, salario, departamento)
        {
            LinguagemDeProgramacao = linguagemDeProgramacao;
        }

        public override void Trabalhar()
        {
            Console.WriteLine($"{Nome} está codificando em {LinguagemDeProgramacao} no
departamento {Departamento}.");
            ExibirSalario();
        }
    }

    public class Gerente : Funcionario
    {
        // Propriedade privada
        private int NumeroDeSubordinados { get; set; }

        internal Gerente(string nome, int id, decimal salario, string departamento,
int numeroDeSubordinados)
            : base(nome, id, salario, departamento)
        {
            NumeroDeSubordinados = numeroDeSubordinados;
        }

        public override void Trabalhar()
        {
            Console.WriteLine($"{Nome} está gerenciando {NumeroDeSubordinados}
subordinados no departamento {Departamento}.");
            ExibirSalario();
        }

        // Método internal (acessível dentro do mesmo assembly)
        internal void DefinirNumeroDeSubordinados(int numero)
        {
            NumeroDeSubordinados = numero;
        }
    }

    public class TesteAcesso
    {
        public static void Main(string[] args)
        {
```

```
            Desenvolvedor dev = new Desenvolvedor("Ana", 101, 6000m, "TI", "C#");
            Gerente gerente = new Gerente("Carlos", 102, 8000m, "RH", 10);

            dev.Trabalhar();
            gerente.Trabalhar();

            // Acesso ao método internal dentro do mesmo assembly
            gerente.DefinirNumeroDeSubordinados(12);
            gerente.Trabalhar();

            // Testes de acesso às propriedades
            Console.WriteLine(dev.Nome); // Acessível
            Console.WriteLine(dev.LinguagemDeProgramacao); // Acessível

            // Acesso permitido a Id porque estamos no mesmo assembly
            Console.WriteLine(dev.Id); // Acessível

            // Departamento é acessível devido ao modificador protected internal
            Console.WriteLine(dev.Departamento); // Acessível

            // O acesso ao Salario e ExibirSalario é restrito, demonstrando o uso de
modificadores
            // Console.WriteLine(dev.Salario); // Erro: Salario é protected
            // gerente.ExibirSalario(); // Erro: ExibirSalario é protected

            // O acesso a NumeroDeSubordinados é restrito
            // Console.WriteLine(gerente.NumeroDeSubordinados); // Erro:
NumeroDeSubordinados é private
        }
    }
}
```

### Explicação Expandida

- **public**: Os métodos `Trabalhar` e as propriedades `Nome` e `LinguagemDeProgramacao` são públicos e podem ser acessados de qualquer lugar.
- **protected**: O método `ExibirSalario` e a propriedade `Salario` são protegidos, permitindo o acesso apenas dentro da classe `Funcionario` e suas classes derivadas.
- **private**: A propriedade `NumeroDeSubordinados` da classe `Gerente` é privada, acessível apenas dentro da classe `Gerente`.
- **internal**: O construtor da classe `Gerente` e o método `DefinirNumeroDeSubordinados` são internos, restringindo o acesso apenas ao mesmo assembly.
- **protected internal**: A propriedade `Departamento` é protegida interna, acessível dentro do mesmo assembly e nas classes derivadas.

Essa implementação cobre todos os modificadores de escopo, demonstrando como eles podem ser utilizados para controlar a visibilidade e acessibilidade dos membros da classe em C#.