

UNIDADE 2 - Tema 1 - Mecanismos de Visibilidade/Acessibilidade



Microfundamento: Algoritmos e abstração de dados

Eixo (<https://pucminas.instructure.com/courses/49042>) > Microfundamento

(<https://pucminas.instructure.com/courses/49011/pages/microfundamento-algoritmos-e-abstracao-de-dados>) > UNIDADE 2 - Tema 1 - Mecanismos de Visibilidade/Acessibilidade

MECANISMOS DE VISIBILIDADE / ACESSIBILIDADE

Nesta sessão vamos explorar melhor os mecanismos de visibilidade de membros de uma classe. É uma sessão mais curta que as outras já desenvolvidas, mas vamos fazer um exercício completo e analisar isso de maneira mais detalhada.

NESTA SESSÃO

Mecanismos de Visibilidade/Acessibilidade

Os Mecanismos de Visibilidade, também chamados na literatura de Modificadores de Acesso, são padrões que controlam o acesso às classes, aos seus atributos e a seus métodos.

Vamos rever uma classe que desenvolvemos na sessão de estudos anterior:

```
class Círculo
{
    private double _Raio;

    public double Raio
    {
        get { return _Raio; }
        set { _Raio = value; }
    }
}
```

```
}  
  
public double CalculaÁrea()  
{  
    return Math.PI * Math.Pow(_Raio, 2);  
}  
}
```

Quando utilizamos **private** o atributo, ou o método, só será visível dentro da própria classe em que ele foi declarado. Você não consegue “enxergar” a variável `_Raio` fora dessa classe, por exemplo, em program e dentro de main. Na classe `Círculo` sim, ela é visível e nós a utilizamos no **get**, no **set**, no `CalculaÁrea()`.

A propriedade **Raio** (agora sem o “_”) já é visível dentro e fora da classe `Círculo`. Ela é **public**. O mesmo acontece com o método `CalculaÁrea()`. Um objeto definido em *main*, por exemplo, consegue ler e colocar valor na propriedade **Raio** e também consegue executar o método `CalculaÁrea()`.

Observe que não colocamos **public** na classe. Quando não colocamos ela é **public** como padrão. E se é **public** é possível criarmos instâncias dessa classe, criarmos objetos dessa classe.

A coisa é realmente interessante. O encapsulamento oculta o código no objeto e somente por meio de métodos específicos é que pode ser acessada, daí o uso de **get** e **set**. Para colocarmos valor no campo da classe temos que invocar a propriedade, que é pública. Temos que passar pelo crivo lógico dela. Não deixamos o acesso livre ao campo. Já falamos sobre isso anteriormente quando citamos o termo Ocultação de Informação.

Existe um fator externo de qualidade de *software* que se chama Robustez. É na realidade a propriedade que um *software* funcionar mesmo em condições anormais, ou adversas, daquelas que foram especificadas. Robustez garante o funcionamento do *software* em condições imprevistas. Outro fator de qualidade de *software* se chama Correção, que se refere à propriedade de um produto de *software* executar uma função exatamente como foi previsto e definido pelo usuário.

Muito bem, existem inúmeros outros Modificadores de Acesso. Nesse microfundamento vamos nos deter no `public` e no `private`. Mas com o tempo você vai descobrir o que é o `protected`, o `internal`, o `protected internal`, e outros.

Assista ao vídeo abaixo:

A interface mostra uma barra de progresso ou status com o texto "Carregando mídia incorporada..." em um fundo branco com uma borda azul.

Agora, vamos praticar o nosso conteúdo.

REFLEXÃO

Você viu dois fatores externos de qualidade de software: Robustez e Correção. Reveja os conceitos por trás desses dois termos e faça uma reflexão sobre o uso de modificadores de acesso na construção de classes.

Você percebe o quanto isso é importante?




APÓS CONCLUIR ESTA SESSÃO DE ESTUDO, RESPONDA:

Um Número Complexo possui a forma “ $a+bi$ ”, sendo “ a ” a Parte Real do número e “ b ” a parte imaginária. O caractere “ i ” se refere à raiz quadrada de -1 ($\sqrt{-1}$).

Pesquise a forma como são feitas as operações de Soma, Subtração, Multiplicação e Divisão de números complexos.

Crie uma classe de nome “Complex” e implemente a solução para esse problema. Crie os atributos PReal e PImag (as duas partes do número complexo) e os métodos solicitados. Preste bastante atenção à visibilidade de atributos e métodos.

Clique [aqui](https://pucminas.instructure.com/courses/49011/files/2899733?wrap=1) (<https://pucminas.instructure.com/courses/49011/files/2899733?wrap=1>)  (https://pucminas.instructure.com/courses/49011/files/2899733/download?download_frd=1) para conferir o gabarito.

[↑ Voltar ao topo](#)



. UNIDADE 2 .

TIPOS ABSTRATOS DE DADOS - CLASSES - IMPLEMENTAÇÃO

Tema 1: Tipos Abstratos de Dados



Definição de um TAD - Classes e Objetos

(<https://pucminas.instructure.com/courses/49011/pages/unidade-2-tema-1-definicao-de-um-tad-classes-e-objetos>)



Atributos, Propriedades e Métodos de Classe

(<https://pucminas.instructure.com/courses/49011/pages/unidade-2-tema-1-atributos-propriedades-e-metodos-de-classe>)



Mecanismos de Visibilidade/Acessibilidade



Construtores de Classe

(<https://pucminas.instructure.com/courses/49011/pages/unidade-2-tema-1-construtores-de-classe>)



Síntese e referências

(<https://pucminas.instructure.com/courses/49011/pages/unidade-2-sintese-e-referencias>)



Atividade objetiva 2

(<https://pucminas.instructure.com/courses/49011/quizzes/171709>)