JOGOS DIGITAIS : PPI2

Orientação a Objetos: Encapsulamento

Professor Sergio Souza Novak

Introdução

- Encapsulamento vem de encapsular

- Em POO significa separar o programa em partes, o mais isoladas possível.

Introdução

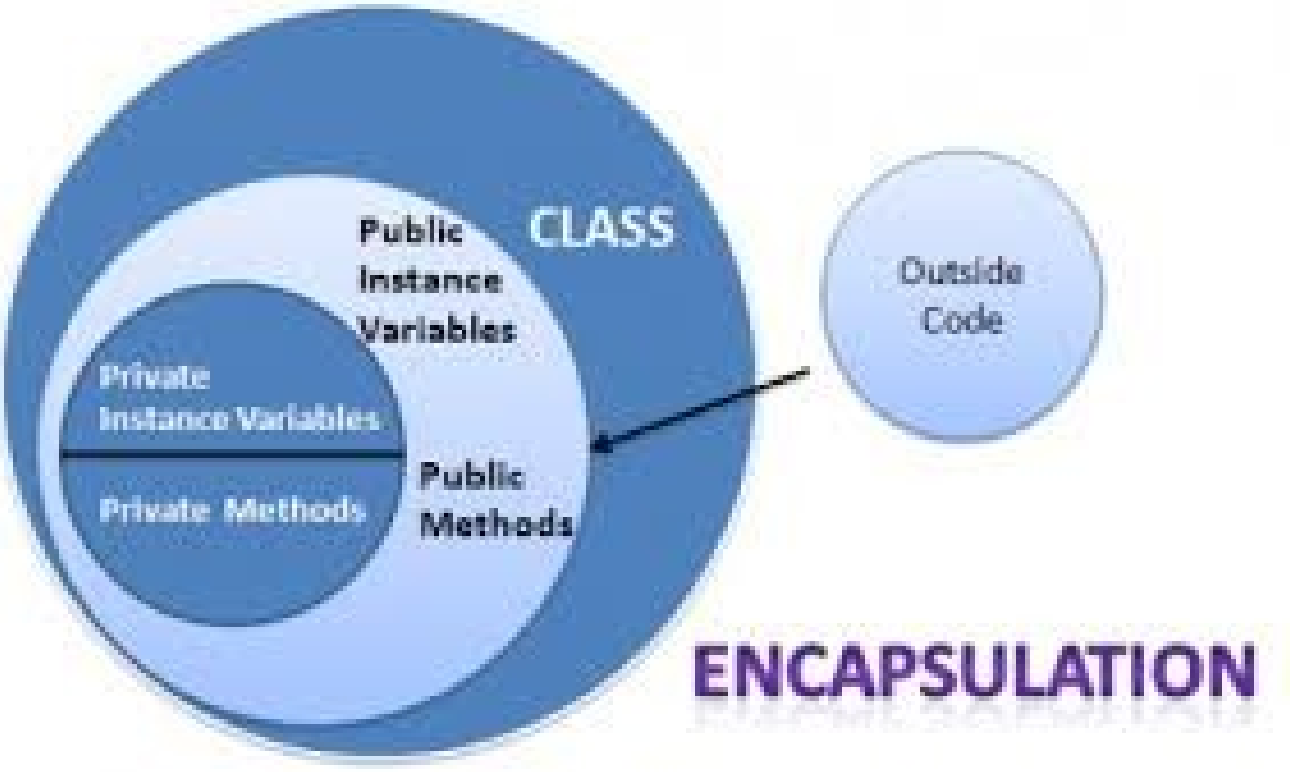
- A ideia é tornar o software mais **flexível**, **fácil de modificar** e de **criar novas implementações**.

Introdução

- Para exemplificar: podemos pensar em uma dona de casa (usuário) utilizando um liquidificador (sistema)
- O usuário **não necessita conhecer detalhes** do funcionamento interno do sistema para poder utilizá-lo

Introdução

- Precisa apenas conhecer a interface, no caso, os botões que controlam o liquidificador.


CLASS
Public
Instance
Variables
Private
Instance Variables
Private Methods
Public
Methods
Outside
Code
ENCAPSULATION

Vantagens

- toda parte encapsulada pode ser modificada sem que os usuários da classe em questão sejam afetados.

Vantagens

- No exemplo do liquidificador:
- um técnico poderia substituir o motor do equipamento por um outro totalmente diferente.
- sem que a dona de casa seja afetada

Vantagens

- No exemplo do liquidificador:
- um técnico poderia substituir o motor do equipamento por um outro totalmente diferente.
- sem que a dona de casa seja afetada

Como o encapsulamento é implementado?

```
class Pessoa
  {
      string nome;
      int olhos, bracos, pernas;
      string cor_olhos;
      string cor_cabelos;
      public void andar(int velocidade)
      {
          // Ande um pouco
      }
      public void falar()
      {
          // Converse mais
      }
      public void comer(string comida)
      {
          // alimente-se mais
      }
  }
```

- Crie um objeto do tipo de pessoa e cheque se o Visual Studio reconhece os métodos e propriedades a partir do **IntelliSense**.

- Repare que os métodos são reconhecidos, no entanto as propriedades (atributos) ainda continuam invisíveis

- identificamos que isso acontece por causa dos modificadores de acesso.

Encapsulamento

- O nível de proteção está sendo restringido, isso na POO é definido como **encapsulamento**.

Encapsulamento

- Para resolvermos os níveis de acesso dos atributos seria muito mais fácil declararmos todos como “public”
- a visibilidade se estenderia a todos a classes da aplicação.
- **no entanto não é uma boa prática fazer isto**
- **pois desta forma toda classe pode alterar os atributos**

Getters and Setters

- Uma forma elegante de tratar isto é manter os atributos como privados e utilizar os **“Getters and Setters”.**

Getters and Setters

- O método “**Get()**” será responsável por retornar alguma informação do objeto atual
- o “**Set(param)**” realizará a alteração do objeto.

```
class Pessoa
   {
       string nome;
   //..

      public string getNome()
      {
          return nome; //retorna o nome
      }
       public void setNome(string nome)
      {
          this.nome = nome; //altera o
nome
      }

   }
```

- no método **getNome**(), nenhum parâmetro é especificado

- Já em **setNome(string nome)** declaramos um parâmetro que é justamente o valor a ser alterado e não possui nenhum tipo de retorno, por isso utilizamos a palavra “void”.

...Demonstrar o uso de Encapsulamento criando uma classe Conta...