ENCAPSULAMENTO

DISCIPLINA: PARADIGMAS ORIENTADA A OBJETOS

ENCAPSULAMENTO

- Mecanismo que define na OO que possibilita restringir o acesso a variáveis e métodos da classe.
- Os detalhes de implementação ficam ocultos ao usuário da classe
- Não é preciso conhecer detalhes sobre os atributos de um objeto, só é preciso ter meios (métodos ou comportamentos) para manipular essas informações

```
.
.
Cliente c = new
Cliente( ).
c.nome=” Ana” ;
.
.
```

Essa manipulação não é recomendada.

VANTAGENS DO ENCAPSULAMENTO

- Ocultar detalhes de implementação
- Tornar o código mais legível
- Minimizar os erros de programação
- Restringir o conteúdo das variáveis
- Facilitar a ampliação do código em função de novas atualizações.

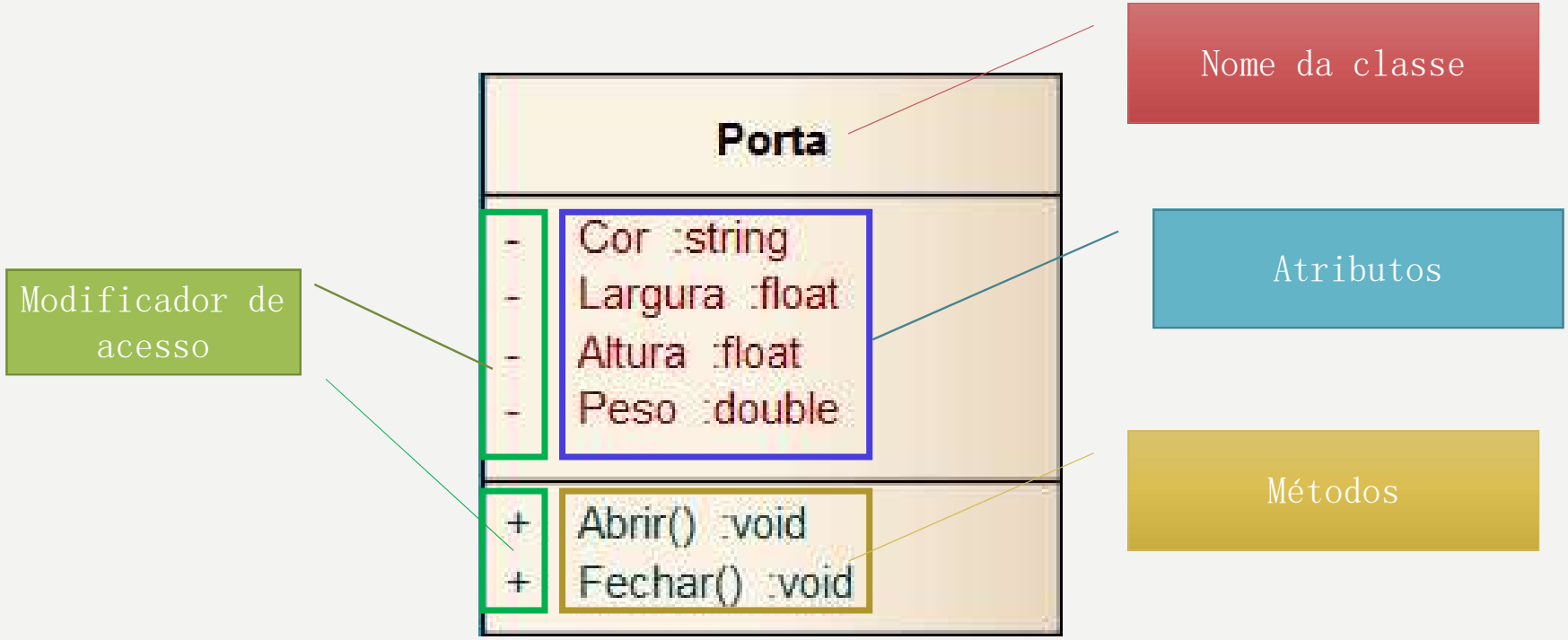
IMPLEMENTANDO O ENCAPSULAMENTO

- Utilizamos o qualificador PRIVATE nos atributos de uma classe para torna-los privado, então só o método do objeto poderá acessar os atributos.
- Já os métodos de uma classe utilizam os qualificador PUBLIC, que o torna público, e isso significa que ele pode ser chamado de qualquer outra classe, desde que seja precedido da referência de seu objeto.

```
public class Cliente{
        private String nome, endereco;
        private int cpf;

        public Cliente( ){
                nome="";
                endereço="";
                cpf=0;
        }
        public void setNome(String snome){
                nome=snome;
        }
```

ENTENDENDO A REPRESENTAÇÃO DA CLASSE PELA NOTAÇÃO DA UML



Fonte: https://www.ateomomento.com.br/uml-diagrama-de-classes/

PALAVRA RESERVADA THIS

- Faz referência ao objeto corrente, isto é, ao objeto que chamou o método.
- This é um apelido para qualquer objeto que chamar esse método

```
public class UsaCliente{
    public static void main(String args[])
{
        Cliente c=new Cliente( );
        c.alteraNome("Ana");
        c.alteraEndereco("Rua A");
        c.alteraCpf(1235);
        System.out.println(c.forneceNom
    e( ));
    }
}
```

```
public class UsaCliente{
    public static void main(String args[])
{
        Cliente c=new Cliente( );
        this.alteraNome("Ana");
        this.alteraEndereco("Rua A");
        this.alteraCpf(1235);
        System.out.println(c.forneceNom
    e( ));
    }
}
```