Visiblidade de um Objeto:

+ publico: a classe atual e todas as outras classes.

- privado: somente a classe atual.

# protegido: a classe atual e todas as suas sub-classes.

Metodos especiais:

Getter(get) - > Pega uma informação

Setters(set) -> entrega uma informação

Construct -> deixa configurações pre definidas para uma classe, ou seja constrói um novo objeto( OBS: quando criar este método nomear com o mesmo da classe)

Apertar ctrl + i para fazer sozinho;

**Encapsulamento:**

Três pilares EHP (Encapsulamento, herança e polimorfismo)

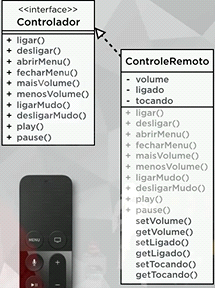
Definição: Ocultar partes independentes da implementação, permitindo construir partes invisíveis ao mundo exterior.

Vantagens:

* Tornar mudanças invisíveis
* Facilitar reutilização do código
* Reduzir efeitos colaterais (ou seja, prejudicar um código)

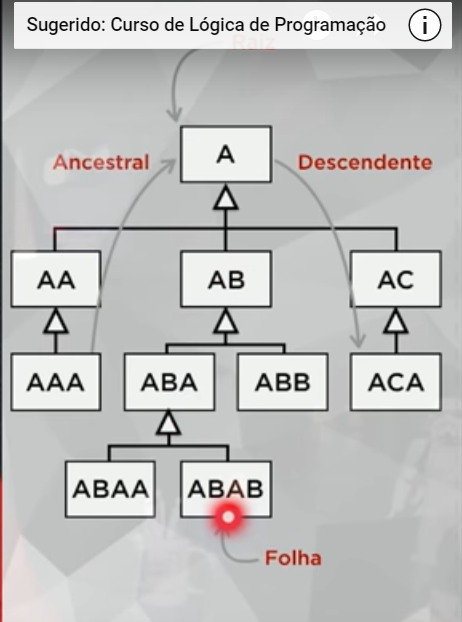
Uso: consta de utilizar uma interface que só contém métodos públicos

Utilizar a palavra **implements** na classe que implementará a interface



**Herança**

utilizar **extends** para herdar outra classe

/

1. herança para implementação -> herança pobre

2. herança para diferença-> Adiciona novos atributos a classe que herda

**Abstrato e Final**

**Classe abstrata->** Não pode ser instanciada, só pode servir como progenitora.Não gera objetos.

**Método abstrato->** Declarado mais nao implementado na progenitora

**Classe final->** Não pode ser herdada por outra classe, ela so obrigatoriamente sera folha. Não pode ter filhos.(estéril)

**Método final->** Não pode ser sobrescrito pelas sua subs classe. Obrigatoriamente herdado

O acesso protegido então passa a dar acesso a sua propia classe e a sua superclasse. Utilizar a palavra final na declaração

**Polimorfismo**