\	ISIBILIDADE DOS OBJETOS
	O que é a <b>UML</b> ?
0 0	ue é o <b>Diagrama de Classes</b> ?
Ilustre a <b>repre</b>	e <b>sentação de uma Classe</b> dentro de um Diagrama de Classes:
llustre u	m <b>Diagrama de Classes completo</b> :
- res	ervada para a questão acima -
O que é o co	nceito de <b>visibilidade</b> dentro da POO?
- res	ervada para a questão acima -
	podemos <b>usar as visibilidades</b> numa entro de um Diagrama de Classes:
- res	ervada para a questão acima -
	bilidades <b>são identificadas na maioria</b>

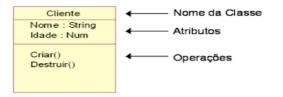
das linguagens de programação?

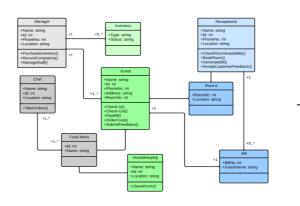
## **VISIBILIDADE DOS OBJETOS**

UML: é a Unified Modelling Language (Linguagem de Modelagem Unificada), ela é uma espécie de padrão de modelagem para o desenvolvimento de softwares. A UML traz uma série de conceitos, ferramentas e diagramas que podem ser implementados para ajudar na visualização dos elementos de um sistema. Para a área de POO especificamente, o diagrama mais importante é o Diagrama de Classes.

Diagrama de Classes: é um diagrama específico para a representação das classes durante a modelagem de um projeto de software. Essa diagrama torna possível a boa visualização das

- classes, seus atributos e métodos. Ele segue o seguinte padrão:
   Classes: Representada pelo Retângulo, seu nome fica posicionado num container no topo do
- retângulo, por convenção as classes devem ser CamelCase; **Atributos:** Ficam posicionados no container abaixo do nome da Classe, por convenção os
- atributos são escritos em letra minúscula; - Métodos: Ficam posicionados no container mais baixo do retângulo, por convenção são escritos
- em letra minúscula e ao final sempre teremos os parênteses representando a entrada de argumentos dos métodos;





O conceito de Visibilidade em POO consiste no quanto os elementos de uma Classe - atributos e métodos - são acessíveis do lado de fora da Classe. Podemos dividir os níveis de acessibilidade em 3, que são:

- Públicos: Elementos que PODEM SER acessados de fora da Classe, eles são representados em UML pelo símbolo de "+";
- Privados: Elementos que NÃO PODEM SER acessados de fora da Classe, eles são representados em UML pelo símbolo de "-";
- Protegidos: Elementos que SÓ PODEM SER ACESSADOS PELAS SUBCLASSES DE UMA CLASSE, em UML eles são representados pelo símbolo de "#";



Na grande maioria das linguagens de programação, nós atribuímos as visibilidades antes de colocar o nome dos elementos, para identificar as visibilidades usamos as seguintes palavras reservadas:

- public: para representar elementos públicos;
- private: para representar elementos privados;
- protected: para acessar elementos que pertencem a Super Classe;

Lembrando que, isso não é uma regra, afinal, temos muitas linguagens que tratam o uso de classes e visibilidades de formas diferentes;

VISIBILIDADE DOS OBJETOS	VISIBILIDADE DOS OBJETOS
Ilustre o exemplo das visibilidades da Classe UML acima agora levado para o contexto das visibilidades de uma Classe dentro da Linguagem de Programação:	Classe: Caneta # aqui vai o nome da Classe  public string modelo # pode ser acessado á partir de uma instância public string cor # pode ser acessado á partir de uma instância private float ponta # só pode ser acessado dentro da Classe protected int carga # pode ser acessado só pelas Classes Filhas protected bool tampada # pode ser acessado só pelas Classes Filhas
- reservada para a questão acima -	public Rabiscar() # o método pode ser acessado á partir se tampada == True então # de uma instância Escrever Erro se não Escrever "Rabisco" Fim do Método
- reservada para a questão acima -	public Tampar() # o método pode ser acessado á partir de tampada = True # uma instância Fim do Método
Como as Instâncias podem acessar os elementos públicos das Classes?	Elas podem acessar usando notação ponto seguida pelo nome do elemento, da seguinte forma:  nome_instancia.nomeAtributo = "valor"no caso de atributos; nome_instancia.nomeMetodo()no caso de métodos;  Lembrando mais uma vez, que isso não é uma regra, pode variar de
	Linguagem para Linguagem.