

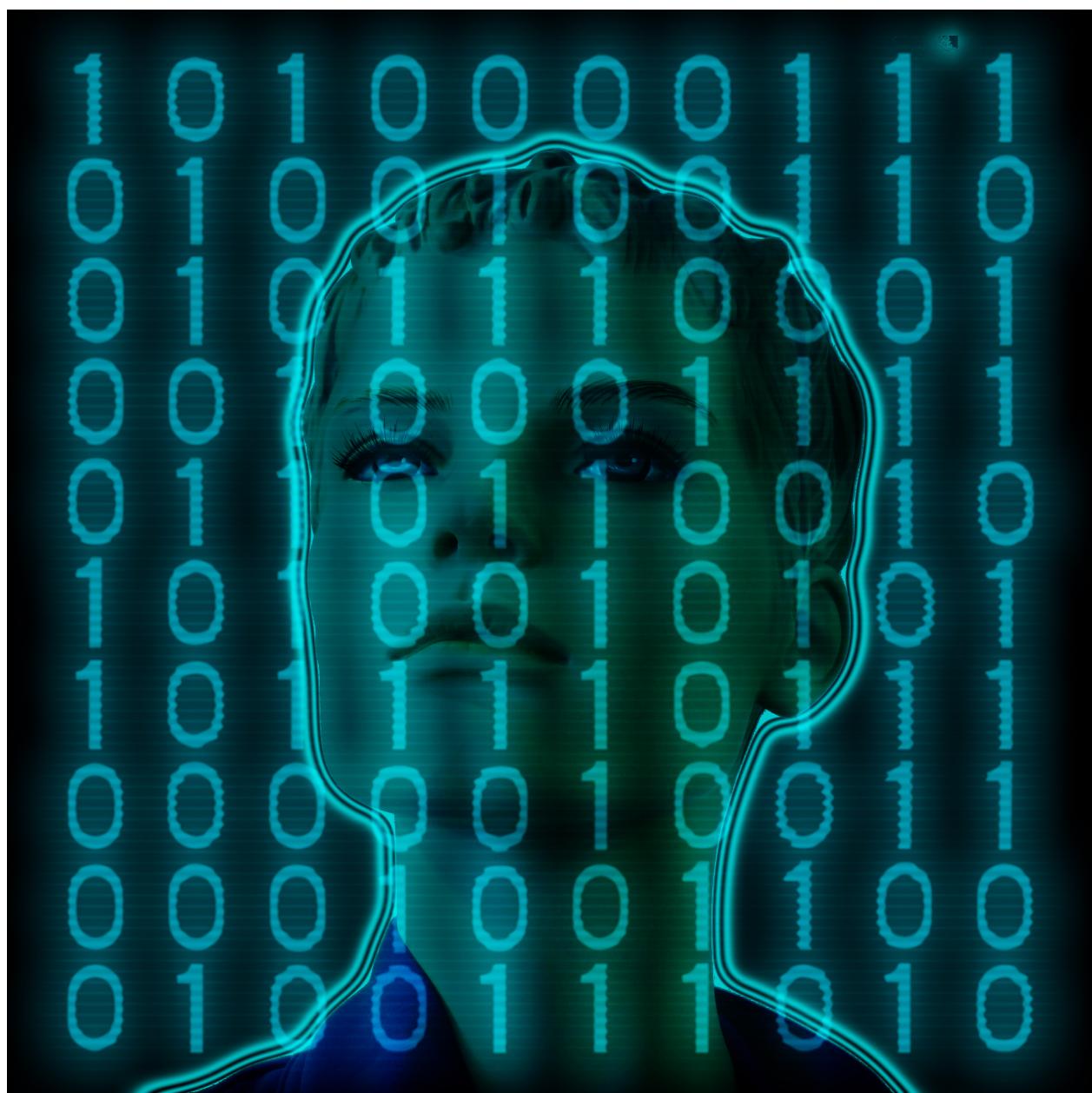
# Vamos desenvolver um Sistema?

Beeeem-vindos alunos queridos!! Bom dia, boa tarde e boa noite! Hoje é um dia especial. Em alguns cliques abaixo você conhecerá conceitos importantes de Programação Orientada a Objetos (POO) partindo de uma linguagem de programação famosa: o Java! Não tenha pressa. Anote em seu caderno o que achar interessante aprofundar, amplie os conceitos e códigos aqui em um projeto próprio, enfim, a casa é sua!

\* Indica uma pergunta obrigatória

---

A escolha da linguagem...



1. Como sabemos, a linguagem é um conjunto de códigos que utilizamos na nossa \* comunicação. É assim entre uma pessoa e outra e também entre pessoa e computador, que no caso só "entende" a linguagem de máquina, ou seja, uma sequência de 0 e 1, como se vê na figura acima. Qual a sua opinião?

*Marcar apenas uma oval.*

- Realmente, seria muito difícil escrever um programa usando somente código binário.
- Linguagens como o Java aproximam o computador da nossa linguagem humana.
- Todas as alternativas estão corretas!

O que vamos precisar?

#### Separe as ferramentas

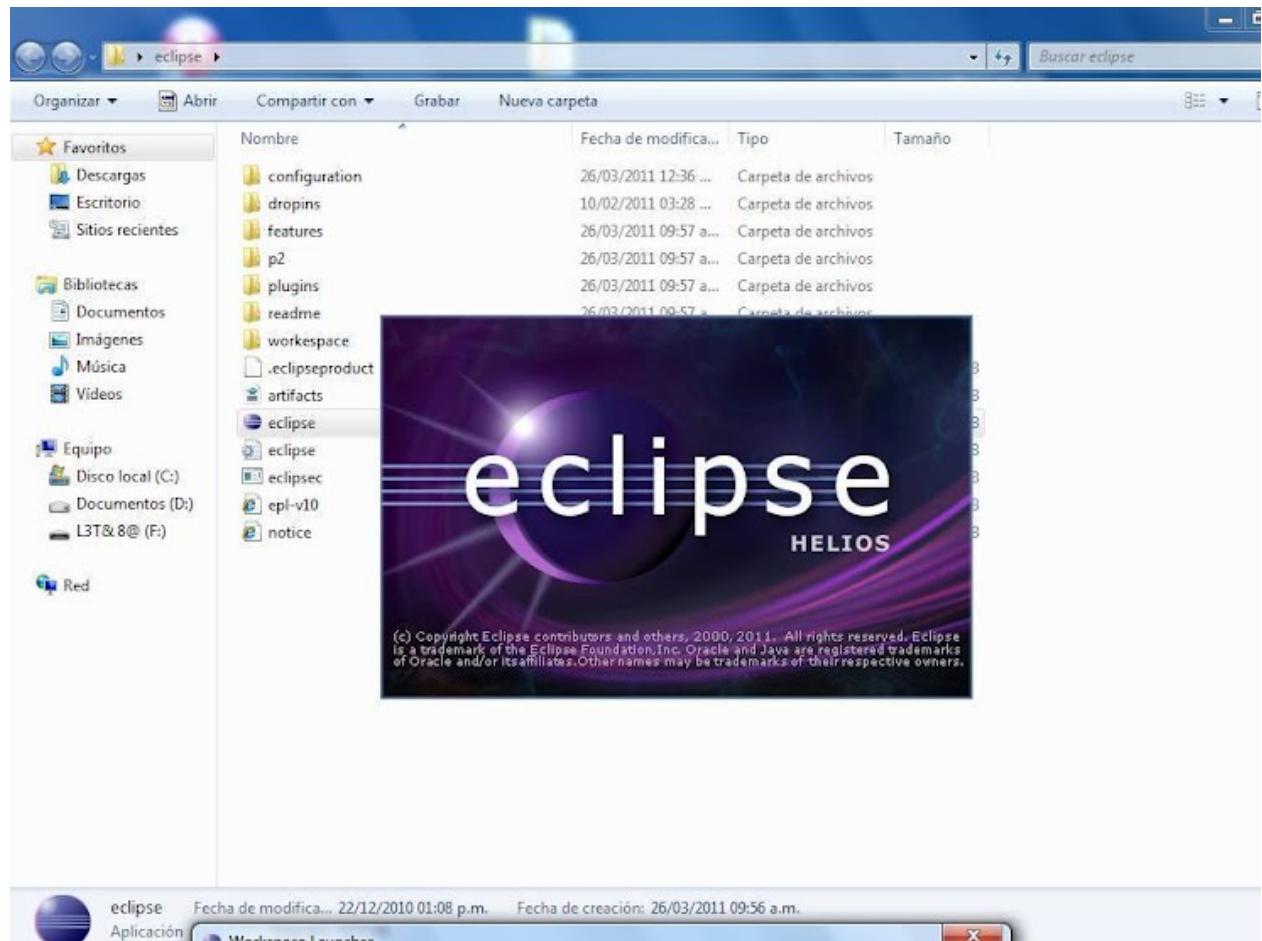
1 - Instalar o JAVA - Precisaremos de um computador com uma JVM, ou seja, uma Java Virtual Machine, a máquina java que permitirá que nosso programa rode em qualquer sistema operacional, tanto Windows quanto Linux ou Mac, enfim. <busque por "JDK" e instale a Java SE Development Kit mais atual para o seu computador>

2 - Conhecer os vocabulários, as palavras, ou os comandos criados pelo JAVA para passar as instruções ao computador que, então, vai nos retornar aquilo que precisamos.

3 - Apoio de uma ferramenta de desenvolvimento, também chamada de "IDE" (Integrated Development Environment). É nela que os códigos ficam mais organizados e com menor chances de erro. <<http://www.eclipse.org>>

4 - Sentar e curtir o resultado!

O Eclipse é uma ferramenta importante de apoio ao desenvolvimento JAVA. É gratuita e de código aberto para qualquer pessoa aproveitar!

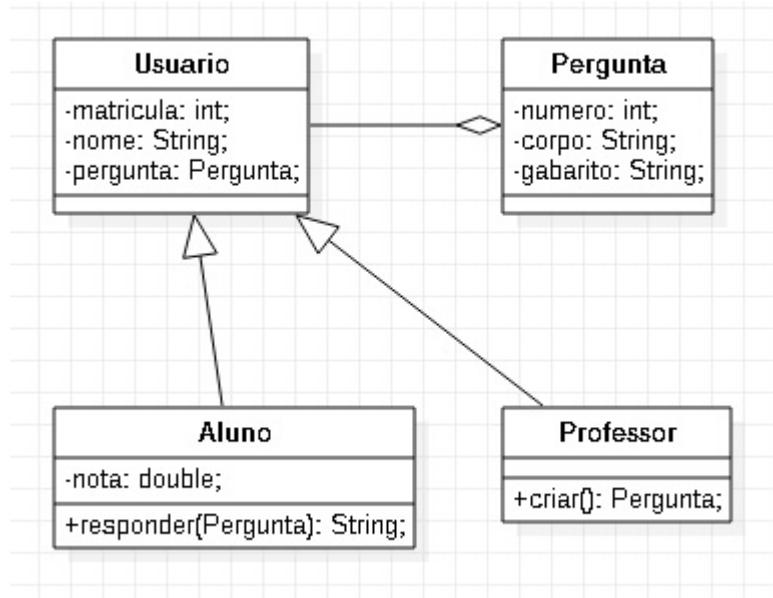


## O projeto

O nosso primeiro código será para desenvolver um Quiz, ou seja, um jogo de perguntas e respostas. As regras do nosso programa são importantes agora, pois evitárá que tenhamos que refazer depois que estiver pronto (retrabalho). É por isso que chamamos de Projeto.

Você pode pensar como se fosse um engenheiro a projetar uma casa. Cada janela, corredor etc, é previsto antes do construtor contratar os pedreiros e isso evita que no futuro seja preciso derrubar alguma parede, não é mesmo?

Observe na imagem que o nosso sistema terá quatro classes. Na parte mais alta temos Usuario que terá apenas número de matrícula e nome. As classes Aluno e Professor são "filhas" de Usuario e, portanto, herdam os seus atributos. Isso economiza código. Não é verdade que o aluno reponde pergunta e o professor cria pergunta? Justamente estes foram os métodos criados!



### E as perguntas?

Observe que há também uma classe Pergunta ligada por agregação à classe Usuario e que, portanto, tanto Aluno quanto Professor "herdam". Na prática, no sistema, o aluno recebe esta pergunta e responde a alternativa "a", "b" ou "c" e o sistema receberá uma String para comparar com o gabarito. Logo vamos ver isso no Eclipse e talvez você entenda melhor.

### O que é int, String e double?

Igual às outras linguagens de programação, em Java temos as variáveis de tipo primitivo e as mais usadas são justamente int, ou seja, a reserva de um espaço na memória do computador para guardar um número inteiro, como 1, 2, ou 3, e double, que guarda um número fracionado ou com vírgula (em java usamos ponto, ok?) tipo 1.1 ou 5.1 etc.

O incrível da Orientação a Objetos em Java é que temos uma classe chamada String e a palavra "string" pode ser traduzida como "texto", ou seja, um tipo Texto. Note que para classes iniciamos sempre com a letra maiúscula, diferenciando dos tipos primitivos. Neste caso, o Java já oferece a classe pronta para facilitar a nossa vida, com funcionalidades diversas para comparar, medir o tamanho, entre outros.

Por que os atributos têm um sinal de - e os métodos estão com + ?

Outra coisa bacana em Orientação a Objetos é a visibilidade dos atributos e métodos. Assim, quando o aluno está com o atributo "- nota" significa que este é PRIVADO, ou seja, o programa principal ou outra classe não poderá alterar a nota diretamente na classe, mas, sim, por um método especialmente criado para tal.

A vantagem disso é que podemos criar CONDIÇÕES para, por exemplo, que a nota lançada não seja um número negativo ou que esteja entre 0 e 10, entre outros. Deste modo, o atributo aluno está ENCAPSULADO, quer dizer, fechado em uma classe e com acesso permitido apenas por um intermediário, que será um método público.

Falando nisso, agora você sabe o que é o sinal "+" antes dos métodos +responder() e +criar(). Eles são públicos! Isso significa que o programa principal, por exemplo, terá acesso direto ao método, sem intermediários. Aliás, os métodos se diferenciam dos atributos e são como as funções que você aprendeu em Lógica de Programação, por isso, no diagrama de classes, possuem os parênteses para receber parâmetros e são seguidos por dois pontos e um tipo de dado para retorno.

O método Responder(), por exemplo, recebe uma Pergunta e devolve ou retorna uma String, que, obviamente, é a resposta do aluno. Mas o método também poderia ser privado se o projetista entendesse a necessidade de algum tipo de proteção daquele método, como, por exemplo, um sistema de um banco que não vai permitir acesso direto a um método como Saque() sem antes confirmar uma senha, não é mesmo?

Bem, estamos quase lá! A seguir já, vamos ver um pouco de código.

### Nosso código

Agora que temos o diagrama de classes, basta transformarmos isso em código com o apoio do Eclipse. Vamos criar um projeto e chamá-lo de Quiz. Em seguida, criamos um pacote e as classes com os mesmos nomes que apareceram no desenho do nosso diagrama de classes.

## Visão do Eclipse com as classes criadas

```

1 package br.com.senai;
2
3 public class Usuario {
4     private int matricula;
5     private String nome;
6
7     public int getMatricula() {
8         return matricula;
9     }
10
11    public void setMatricula(int matricula) {
12        this.matricula = matricula;
13    }
14
15    public String getName() {
16        return nome;
17    }
18
19    public void setName(String nome) {
20        this.nome = nome;
21    }
22 }

```

Observe aqui as classes criadas conforme o nosso esquema, ou seja, com iniciais maiúsculas. Elas estão dentro de um Pacote Java que está dentro de um Projeto Java "Quiz"

Precisamos falar sobre este código!

Quando você abre o Eclipse, ele pede para criar inicialmente o nome do projeto. Fizemos isso. Depois criamos o pacote para receber as classes e o nome escolhido foi "br.com.senai". É o que você também vai fazer quanto estiver trabalhando em uma empresa, sendo que, na organização do Java, "br", "com" e "senai", separados com ponto, são na verdade pastas que você pode encontrar navegando entre arquivos de seu sistema operacional. Trata-se apenas de organização.

Note o lado direito da imagem. Nela você encontra abas do Eclipse com as classes Usuario, Pergunta, Aluno e Professor e a primeira aba selecionada é Usuario. É a classe que vai oferecer o número de matrícula e o nome para Aluno e Professor herdarem. É apenas um início. Precisaremos incrementar, ou seja, anotar as regras do nosso negócio nos nossos métodos.

Faremos isso em breve, mas antes vejamos como estão as classes Aluno e Professor.

## Tela dividida no Eclipse

```

1 package br.com.senai;
2
3 public class Aluno extends Usuario {
4     private double nota;
5
6     public double getNota() {
7         return nota;
8     }
9
10    public void setNota(double nota) {
11        this.nota = nota;
12    }
13    public String responder(Pergunta pergunta) {
14        return null;
15    }
16}
17

1 package br.com.senai;
2
3 public class Professor extends Usuario {
4     public Pergunta criar() {
5         return null;
6     }
7 }
8

```

### Os herdeiros aparecem!

Sim, as classes Aluno e Professor herdaram de Usuário as suas características ou atributos. É por isso que o código aparece como "public class Aluno extends Usuario", ou seja, a classe aluno ou filha "estendeu" a classe mãe. Note que tanto o método responder() quanto o método criar() estão apenas com um "return null" ou seja, não retornam nada e não fazem nada ainda e precisam ser implementados. Foram apenas criados e estão ali esperando as regras do negócio.

Todos os métodos acima, aliás, são públicos mas se diferenciam pelo retorno. No caso os métodos getNota() e setNota() são aqueles tipos de métodos especiais que falamos para ter acesso ao atributo privado "nota". O getNota() retorna um valor double e também não foi implementado ainda. O método setNota () permite alterar a nota. É por isso que recebe um double, mas não retorna nada, ou seja, ele é vazio ou "void". Percebe?

Por hoje é isso pessoal!

2. Qual o seu nome? \*

---

3. Aproveite este espaço para as suas críticas ou sugestões! Ficamos muito felizes \*  
(de verdade!). O espaço é seu.

---



---



---



---



---

Este conteúdo não foi criado nem aprovado pelo Google.

## Google Formulários

