



High performance. Delivered.

## Application Delivery Fundamentals: Java

Module 4: Classes and Objects

#### Introduction

MIT Faculty Video:

"Classes and Objects"

## **Module Objectives**

- No final deste módulo, os participantes serão capazes de:
  - Definir e aplicar o conceito de Programação Orientada a Objetos
  - Definir e diferenciar classes de instâncias
  - Definir o conceito de coleta de lixo
  - Explicar a estrutura do pacote Java e o caminho da classe
  - Criar, usar e importar classes
  - Definir o conceito de construtor e finalizador
  - ➤ Identifique e use modificadores de acesso
  - Identifique e crie atributos e métodos estáticos



## Overview of Object-Oriented Programming

- A Programação Orientada a Objetos é um padrão de programação que utiliza objetos e suas interações para projetar e implementar aplicativos;
- Objetos são entidades que servem como os blocos de construção básicos de um aplicativo orientado à objetos;
- Um objeto é uma entidade independente com atributos e comportamentos;

# Overview of Object-Oriented Programming

#### **Objeto:**



## Atributos e Comportamento

- Variáveis declaradas como parte da classe representam os dados contidos pelas instâncias dessa classe;
- Os valores armazenados por essas variáveis representam os 'atributos' ou 'estados' atuais do objeto
- Os métodos declarados como parte da classe representam as operações que as instâncias da classe podem executar;
- Diz-se que essas operações são o 'comportamento' do objeto

## Informações dentro de um objeto

#### Cada objeto define três tipos básicos de informações:

**Identity** 

 Um objeto deve descrever os recursos que permitirão a seus usuários diferenciá-lo de outros objetos. Precisa ter uma identidade. Mesmo que dois objetos compartilhem os mesmos recursos, cada objeto tem uma identidade única.

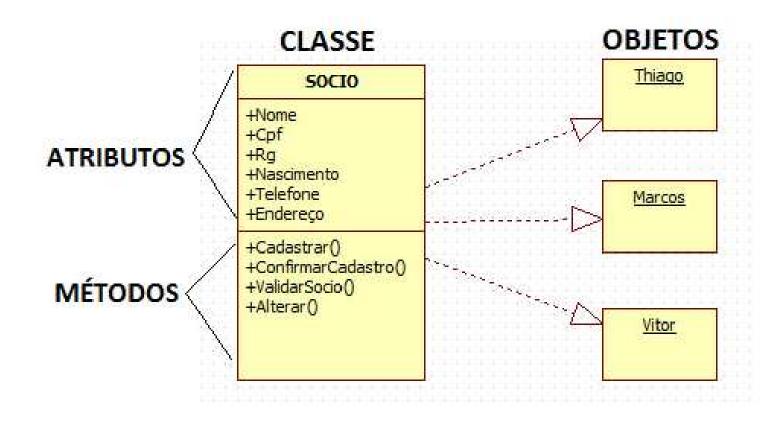
**Attributes** 

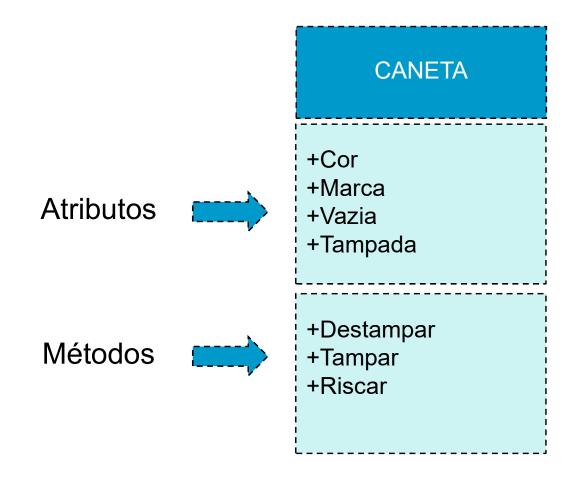
• Um objeto deve ser capaz de se descrever. Esse tipo de informação é armazenado nos atributos de um objeto que forma a estrutura do objeto..

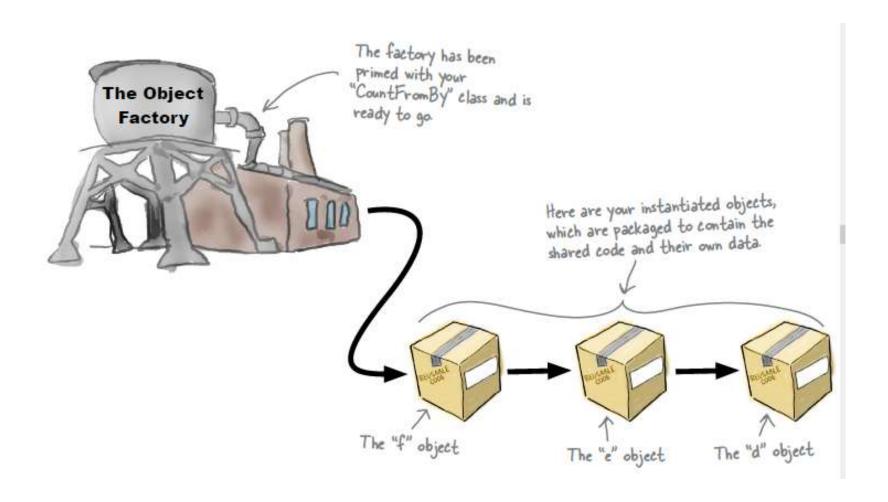
**State** 

 Um objeto deve ser capaz de descrever sua condição atual, chamada estado. Às vezes, o estado do objeto é representado pelos valores de cada um de seus atributos.

- Uma classe é um modelo para um objeto específico.
- Uma classe define os atributos e métodos que todos os objetos pertencentes à classe possuem.
- Os atributos e métodos de uma classe são chamados de "campos" ou "membros".
- Uma instância refere-se a um objeto que é membro de uma classe específica. Todos os objetos que pertencem a uma classe são instâncias dessa classe.







## Overview of Object-Oriented Programming

#### Classe:

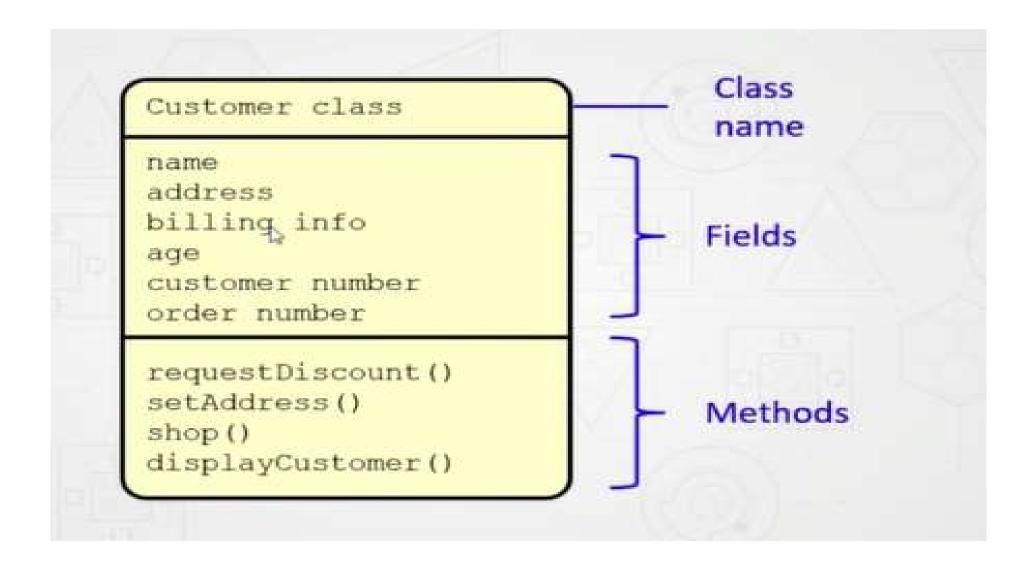
Class Name = Substantivos

Fields = Adjetivos

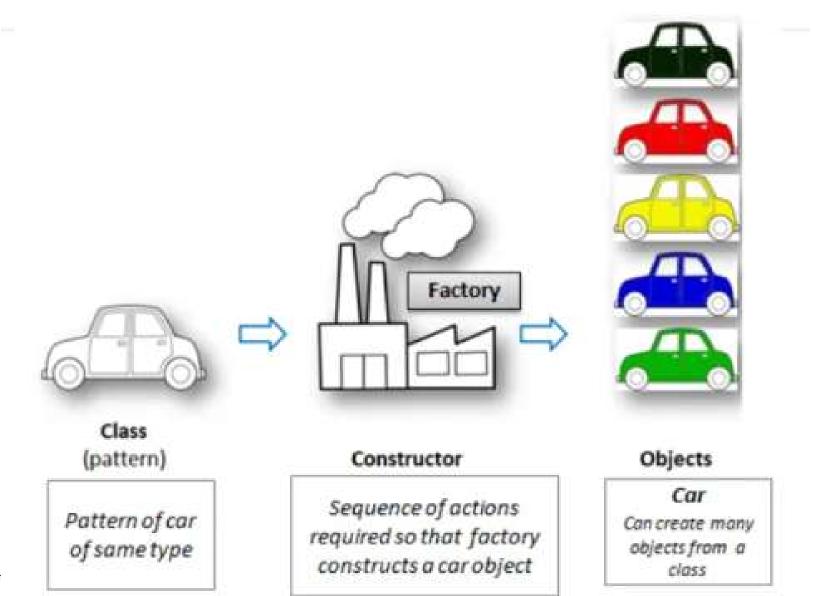
Methods = Verbos

```
Class declaration
   public class Customer {
       public String name = "Junior Duke";
                                                           Fields
       public int custID = 1205;
                                                           (Properties)
       public String address;
                                                           (Attributes)
       public int orderNum;
       public int age;
       public void displayCustomer(){
           System.out.println("Customer: "+name);
                                                           Methods
                                                           (Behaviors)
10
11
```

# Overview of Object-Oriented Programming



## Classes vs. Instances (cont.)



Copyrigh

## Criando e Manipulando Instancias da Classe

- Instâncias de uma classe podem ser criadas usando a palavrachave 'new'. Esse processo é chamado de "instanciação".
- A palavra-chave 'new' cria o objeto com base na classe especificada e retorna uma referência ao objeto recém-criado.
- A referência é então recebida por um identificador apropriado.

```
<Class Type> <Identifier> = new <Class
Type>(<Constructor Parameters>)

Person him = new Person("John Doe", 25, 'M');
```

i

Refer to the ClassInstanceSample.java sample code.

## Criando e Manipulando Instancias da Classe (cont.)

- Os campos das instâncias de objetos podem ser acessados através dos nomes dos identificadores.
- O acesso aos campos e métodos dos membros é feito através do uso da notação 'ponto'.

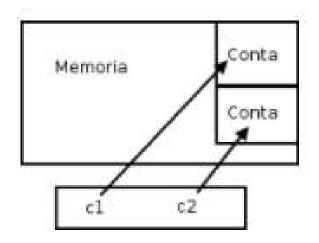
```
<reference>.<attribute name>
<reference>.<member method>(<parameters>);
him.setAge(23)
him.getName();
```

i

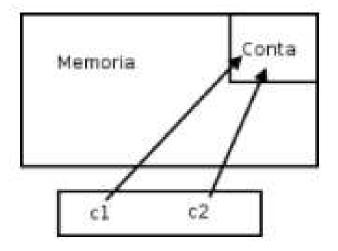
Refer to the ClassInstanceSample.java and Person.java sample code.

#### NEW

```
Conta c1;
c1 = new Conta();
Conta c2;
c2 = new Conta();
```



Conta c1 = **new** Conta(); Conta c2 = c1;



#### **Custom Classes**

- Um aspecto primário de uma linguagem de programação orientada a objetos é a capacidade de definir classes personalizadas;
- Uma declaração de classe é o componente básico de um aplicativo Java;
- Todo o código é criado e anexado a uma declaração de classe.
   Não há código que não pertença a uma classe;

## **Custom Classes (cont.)**

- Uma classe Java denota uma categoria de objetos e atua como um modelo para a criação de tais objetos;
- As classes declaram a que pacote pertencem;
- Classes declaram quais classes importar de outros pacotes;
- Os campos (também conhecidos como variáveis ou atributos) se referem às propriedades da classe;
- Métodos (também conhecidos como operações) se referem a comportamentos que a classe exibe;
- O arquivo de origem que contém a declaração de classe deve ter o mesmo nome que sua classe pública;

## Package Structure and the Class Path

- Packages são coleções de classes.
- Packages são usados para agrupar classes com base na funcionalidade ou relacionamento.
- As classes podem ser atribuídas a um determinado pacote usando o método package keyword:

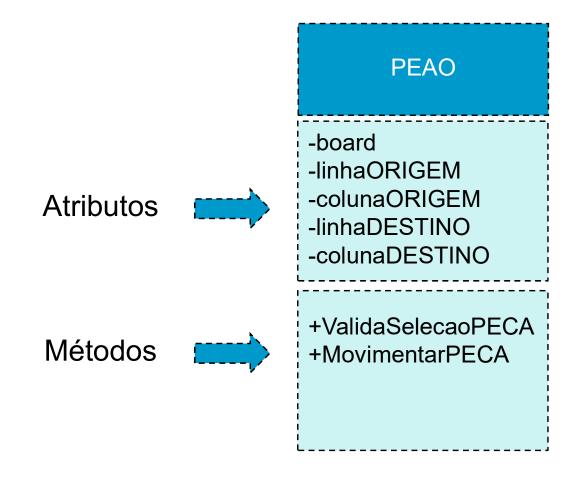
```
package <package_name>;
package sef.module4.samples;
```

 A estrutura da package é representativa da estrutura de diretório em que a definição de classe está localizada.

## **Importing Classes**

- Um programa pode usar recursos externos importando-o;
- Os recursos importados podem ser uma única classe ou um pacote inteiro de classes;
- As classes importadas precisam estar visíveis no caminho da classe para serem acessíveis.

```
import <package_name>; OR import <class-name>;
import java.util.*;
import java.util.Calendar;
```



## **Concept of Constructor**

- Construtores são métodos que definem o estado inicial de um objeto;
- O nome do construtor deve ser o mesmo que o nome da classe;
- O construtor não pode retornar um valor, nem mesmo nulo;
- Construtores n\u00e3o podem ser herdados;
- Se a classe não declarar explicitamente um construtor, o padrão será um construtor no-parameter, do-nothing;

```
public class Cube1
  private int tamanho;

Cube1(int tamanho)
  {
    this.tamanho = tamanho;
  }
}
```

## **Activity** – Create a Java File

- In this activity, you will:
  - Crie um arquivo java Shirt.java dentro do package sef.module4.activity
  - Com os atributos: description, colorCode, price
  - Com os métodos: constructor e display;
  - No método constructor inicialize os atributos;
  - No método display exiba os atributos;



## **Activity** – Create a Java File

- In this activity, you will:
  - Crie um arquivo java DriverShirt.java dentro do package sef.module4.activity com o método main
  - Instancie a classe Shirt;
  - Execute o método Display();



## Activity - Create a Java File

#### Atividade

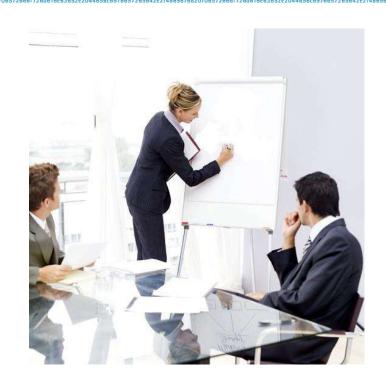
- Crie um Classe java Carro.java dentro do package sef.module4.activity
- Com os atributos: marca, CodigoCor, preço
- Com os métodos: constructor, Buzinar e Exibir;
- No método constructor inicialize os atributos;
- No método exibir exibe as atributos;
- No método Buzinar emitir som



## **Activity** – Create a Java File

#### Atividade

- Crie um classe java CarroPrincipal.java dentro do package sef.module4.activity com o método main
- Instancie a classe Carro;
- Execute o método Exibir();
- Execute o método Buzinar();



## **Perguntas:**

## O que é uma classe?

Um modelo para criação de objetos;

## O que é uma instancia de classe?

Uma instância refere-se a um objeto que é membro de uma classe específica. Todos os objetos que pertencem a uma classe são instâncias dessa classe

## O que é um construtor?

Construtores são métodos que definem o estado inicial de um objeto. Tem o mesmo nome da classe;

## **Garbage Collect**

### "Garbage Collect"

2148696768207065726661726d616e53652e2044656c6976657265642e2148696768207065726561726d616e53652e2044656c6976657265642e2148696768207065726561726d616e53652e2044656c6976657265642e2148696768207065726561726d616e53652e2044656c6976657265642e2148696768207065726561726d616e53652e2044656c6976657265642e2148696768207065726561726d616e53652e2044656c6976657265642e2148696768207065726561726d616e53652e2044656c6976657265642e2148696768207065726561726d616e53652e2044656c6976657265642e2148696768207065726561726d616e53652e2044656c6976657265642e2148696768207065726561726d616e53652e2044656c6976657265642e2148696768207065726561726d616e53652e2044656c6976657265642e2148696768207065726561726d616e53652e2044656c697667726d616e53652e2044656c697667726d616e53652e2044656697667726d616e53652e2044656697667726d616e53652e2044656697667726d616e53652e2044656697667726d616e53652e2044656697667726d616e53652e2044656697667726d616e53652e204465669766772606

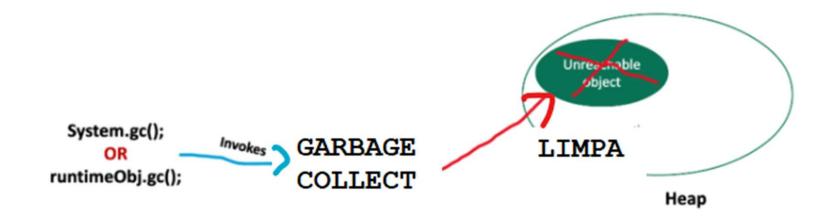


## **Garbage Collect**

- A palavra-chave 'new' aloca explicitamente recursos para um objeto, mas não há uma maneira explícita de dizer ao sistema para 'desalocar' um objeto.
- A Garbage collect é o processo de limpar automaticamente a memória usada por objetos que não serão mais usados pelo programa.
- A chamada do Garbage collect é determinada automaticamente pela Java Virtual Machine, dependendo da necessidade de fazê-lo.
- Um aplicativo pode 'sugerir' que o sistema inicie a coleta de lixo usando System.gc () ou Runtime.getRuntime (). Gc ().



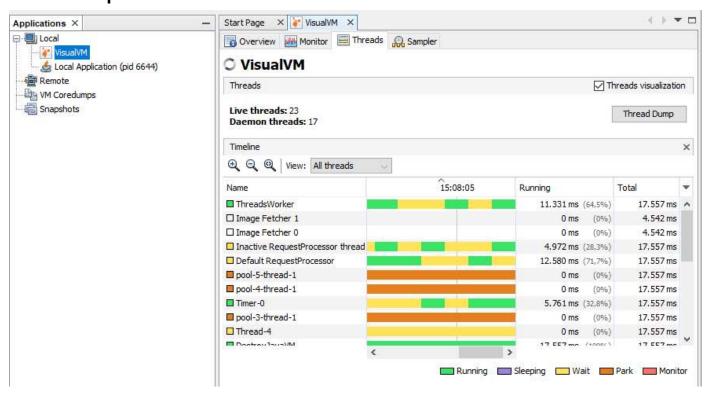
Refer to the garbageCollect.java sample code.



#### JVisualVM

 VisualVM é uma ferramenta que fornece informações detalhadas sobre aplicativos Java enquanto estão sendo executados em uma Java Virtual

Machine;



#### JVisualVM

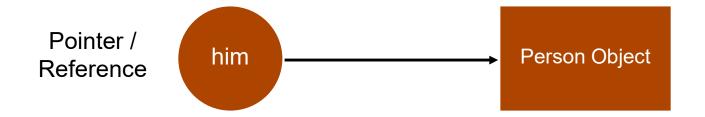
```
Directory of C:\Program Files\Java\jdk1.8.0_181\bin
25/09/2018 09:22 197.464 jvisualvm.exe
1 File(s) 197.464 bytes
```



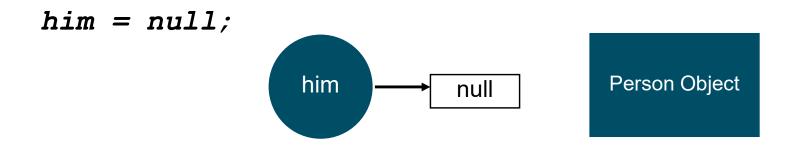
Refer to the jvisualvm.exe sample code.

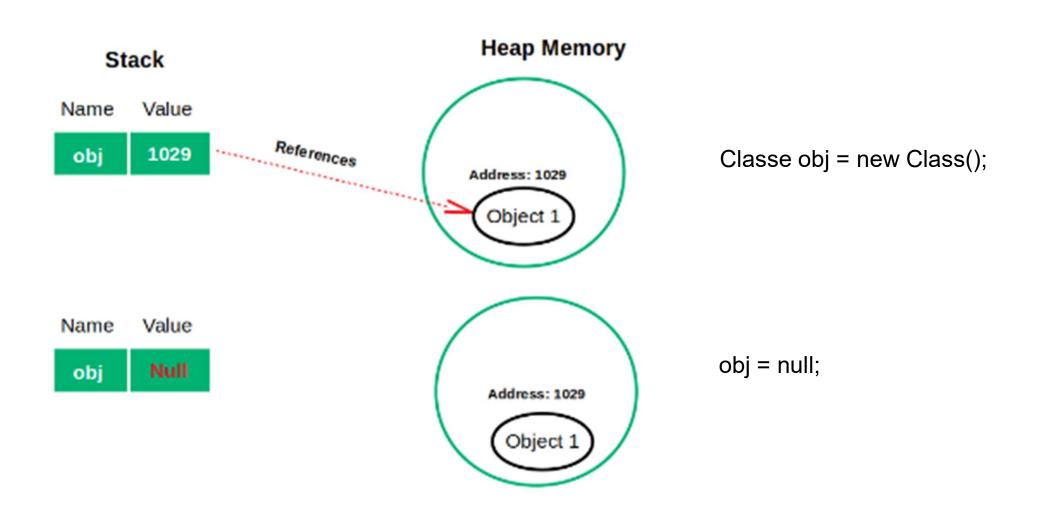
 Objetos estão seguros da garbage collect se houver pelo menos uma variável de referência que se refira ao objeto

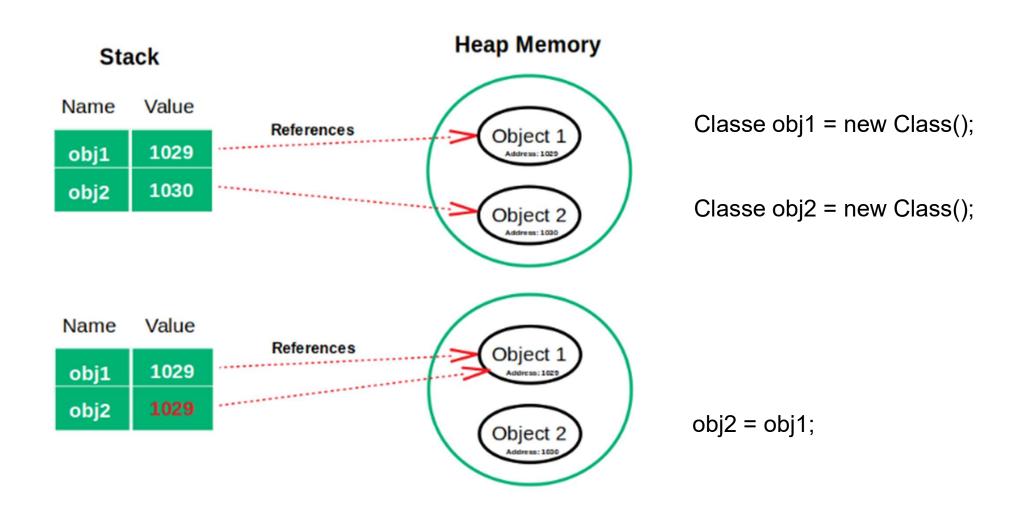
Person him = new Person();

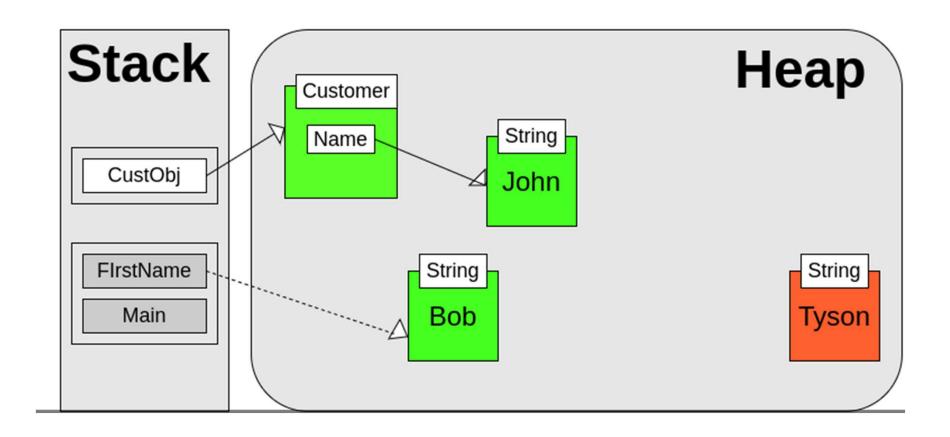


 Objetos são elegíveis para garbage collect quando não houver mais identificadores que se refiram ao objeto









### **Perguntas:**

## O que é um package?

É uma coleção de classes.

## O que é Garbage Collect?

A coleta de lixo é o processo de limpar automaticamente a memória usada por objetos que não serão mais usados pelo programa.

## **Concept of Finalizer**

- Um finalizer() é um método especial que é executado para remover o objeto ou recurso da memória quando não é mais necessário.
- O método finalizer() é usado pelo coletor de lixo para liberar memória de recursos que não são mais necessários.

```
protected void finalize() throws Throwable
  {
   try
        {
        close(); // close open files
        }
   finally
        {
        super.finalize();
        }
}
```

## **Questions and Comments**

 What questions or comments do you have?



