

Conceptos fundamentales de Java 4-2: Clases de objeto y piloto Actividades prácticas

# Objetivos de la lección:

* Describir la forma general de un programa Java
* Describir la diferencia entre una clase de objeto y una clase de controlador
* Acceder a un mínimo de dos API de clase Java
* Explicar y dar ejemplos de palabras clave de Java
* Crear clases de objeto
* Crear clases de controlador

# Vocabulario

Identifique el término correspondiente a cada una de las definiciones a continuación.

|  |  |
| --- | --- |
| Importación | Un grupo de clases Java relacionadas. |
| Bloques de código | Secciones de código que se escriben entre llaves. {} |
| CamelCase en mayúscula | Primera letra mayúscula y mayúscula inicial en la primera letra de cada palabra interna. Ejemplo: CuentaCorriente |
| Constante | Valor con nombre que no cambia. |
| CamelCase en minúscula | Primera letra minúscula y mayúscula inicial en la primera letra de cada palabra interna. Ejemplo: nombreEstudiante |
| Clases de controladores | Clase que contiene un método principal. |
| Sentencia de importación | Sentencia de código en un archivo de clase Java que incluye código Java de otro paquete o clase. |
| Clase objeto creado por el programador | Clase que define las instancias de objetos que hay que usar en otra clase. |
| Componentes de Java | Código precedido por //. Los comentarios se utilizan para aclarar la lógica de programación. El compilador ignora los comentarios. |
| Palabras clave de Java | Palabra que tiene una función especial en el lenguaje Java y no puede utilizarse como nombre de clase, método o variable. |
| API de Java | Biblioteca de clases Java disponible para importar en una clase creada por el programador. |
| Clase objeto | Estructura de un objeto, incluidas las variables de clase, los constructores y los métodos. |
| Constructor | Tipo especial de método que es una plantilla para un objeto. |
| Parámetros | Valores que se envían a un método o constructor para que se utilicen en un cálculo o para sustituirlos por valores de la clase. |
| Variables | Valores como números, caracteres o booleanos. Referencias a objetos como un objeto CuentaBancaria. |
| Modificadores de acceso | Palabras clave para especificar la accesibilidad de una clase (o tipo) y sus miembros. Por ejemplo: public, private, protected, default |
| Métodos | Bloque de código dentro de una clase que se usa para cambiar la información de la clase o acceder a ella. |

# Inténtelo/resuélvalo:

1. **Enumere los componentes que forman un archivo .java. Enumere los componentes en el orden en que esperaría verlos en un programa Java.**

R/ La sintaxis de un lenguaje define los elementos de dicho lenguaje y como se combinan para formar un programa.

1. Identificadores
2. Tipos de datos
3. Palabras reservadas
4. Sentencias
5. Bloques de código
6. Comentarios
7. Expresiones
8. Operadores
9. **Describa la diferencia entre el formato CamelCase en mayúsculas y en minúsculas y proporcione un ejemplo de cuándo los usaría.**

R/ El nombre de la clase es un sustantivo etiquetado con mayúscula.

Las anteriores encadenan palabras con la primera letra en mayúscula, pero la palabra principal lleva mayúscula

Los nombres de constantes se declaran en letras mayúsculas con el modificador final.

Ejemplo: ThisIsUpperCamelCase

El nombre del paquete opcional se define antes de una sentencia de importación en minúscula.

Estás encadenan palabras con la primera letra en mayúscula, pero la palabra principal no lleva mayúscula.

Los nombres de variables son breves pero significativos en minúsculas.

Ejemplo: thisIsLowerCamelCase

1. **¿Qué sintaxis se utiliza para importar todo el paquete de utilidades de Java? Y si importase todo el paquete, ¿también necesitaría importar las clases adicionales en el mismo paquete por separado?**

R/ package <package\_name>;

import <other\_package>;

Public class class name

{ <variables (also know as fields)>;

<constructor método (s)>;

<other methods>;

Importar define otras claves o grupos de clases que usted usa en su clase.

La sentencia de importación brinda información del compilador que identifica las clases externas usadas con la clase actual.

● Se usa para identificar paquetes o clases de objetos que desea usar en su clase

● Puede importar una sola clase en todo el paquete

● Puede incluir múltiples sentencias de importación

● Las sentencias de importación siguen la declaración del paquete y precede la declaración de clase.

import.java.util.scanner;

1. **Escriba la sintaxis de una clase de objeto Java simple con el nombre Student y el siguiente formato: Student Name: Lisa Palombo**

**Student ID: 123456789 Student Status: Active**

**La información del estudiante se almacenará en las siguientes variables:**

**fName, lName, stuId, stuStatus.**

R/

1. **Escriba el código para una clase de controlador que cree un objeto Student y muestre la información sobre el objeto en la pantalla.**

**R/**

import java.util.Scanner;

public class Persona {

public Persona(String name, int edad) {

}

public static void main(String[] args) {

Scanner sc = new Scanner ( System.in);

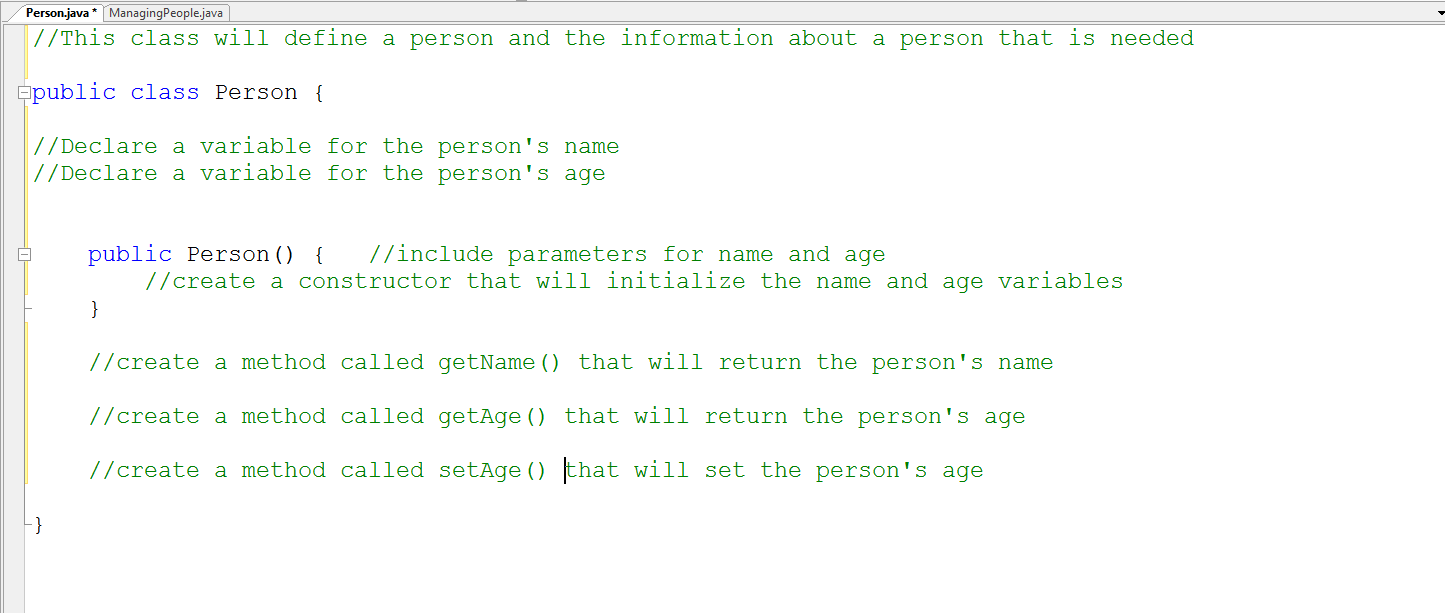
System.out.println ("ingrese el nombre de la persona");

String name = sc.next();

}

}

1. **Enumere 10 palabras clave de Java de esta lección.**
2. ● Paquete
3. ● Importar
4. ● Clase
5. ● Variable o campo
6. ● Constructores
7. ● Métodos
8. ● Modificadores de acceso
9. ● Bloques de código
10. ● Constantes
11. ● Api de Java
12. **Complete la clase de objeto creada por el programador que aparece a continuación. Tenga en cuenta las instrucciones de los comentarios.**



**R/**

import java.util.Scanner;

public class Person {

public Person(String name, int edad) {

// TODO Auto-generated constructor stub

}

public static void main(String[] args) {

Scanner sc = new Scanner ( System.in);

System.out.println ("ingrese el nombre de la persona");

String name = sc.next();

System.out.println ("ingrese la edad de la persona");

int edad = sc.nextInt();

Person per = new Person (name, edad);

per.setAge(30);

per.setAge("christian");

System.out.println("la edad de la persona es = " + per.toString());

Person p1 = new Person (name, edad);

Person p2 = new Person (name, edad);

if (p1.getName()==p2.getName())

{

System.out.println(p1.getName()+"Tiene la misma edad que = "+p2.getName());

}

else

{

System.out.println(p1.getName()+"no tiene la misma edad que = "+p2.getName());

}

}

private void setAge(String string) {

// TODO Auto-generated method stub

}

private void setAge(int i) {

// TODO Auto-generated method stub

}

private String getName() {

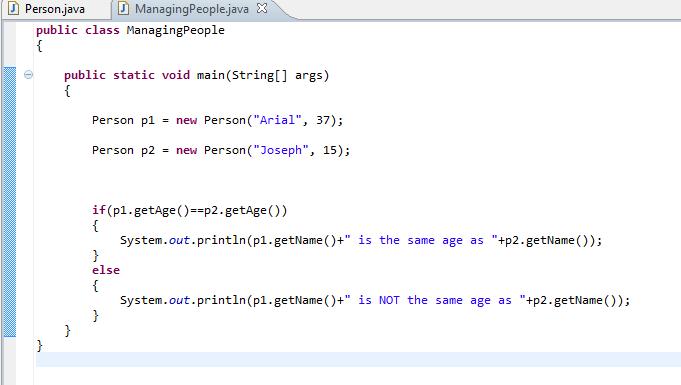
// TODO Auto-generated method stub

return null;

}

}

1. **Complete la siguiente clase de controlador. Tenga en cuenta las instrucciones de los comentarios.**



R/

public class EjercicioUnoperson {

//todos los atributos de la clase deben ser de tipo privados

// zona de atributos o caracteristicas.

private String name;

private int age;

//el constructor siempre se llemara igual que la clase

public void Person (String name, int age) {

this.name = name;

this.age = age;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

@Override

public String toString() {

return "EjercicioUnoperson [name=" + name + ", age=" + age + "]";

}

}