

Asignatura

Programación



UNIVERSAE
Instituto Superior de FP

Asignatura

Programación

UNIDAD 6


Clases y utilización de objetos



UNIVERSAE
Instituto Superior de FP

Concepto de clase

- Principal elemento de la programación orientada a objeto.
- Representación abstracta de una entidad del mundo real
- La clase es la plantilla o estructura.
- Los objetos son clases que contienen datos dentro de su estructura
- La instanciación de una clase es un objeto en ejecución.
- Puede tener visibilidad pública o defecto



- ❖ Nombre.
- ❖ Edad
- ❖ Sexo

- Saludar
- Comer
- Estudiar



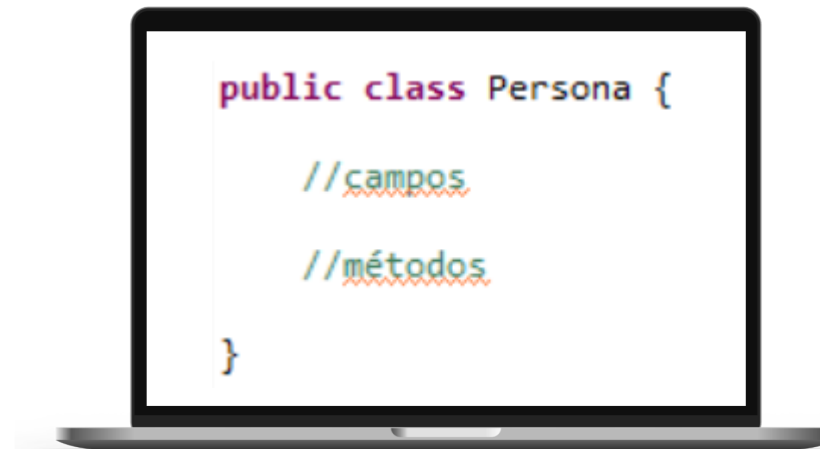
Estructura

- Campos o atributos
Son las propiedades que tiene una entidad
- Métodos
Son las acciones que puede hacer la entidad



Sintaxis

[visibilidad] class Nombre [extends Superclase] [implements Interface1, Interface2, ...] {}



Campos o atributos

- Atributos, variables, fields.
- Son las propiedades de la clase que contienen valores.
- Cada clase tendrá un valor diferente
- Si hay un campo igual para todas las clases se declarara static
- Pueden tener visibilidad pública, privada, protegida o por defecto.
- Es recomendable que tengan visibilidad privada
- Se usen métodos para obtener o modificar sus datos.



- ❖ Nombre
- ❖ Primer apellido
- ❖ Segundo apellido
- ❖ Edad



Sintaxis

[visibilidad] [static] tipo nombreCampo

```
public class Persona {  
  
    //atributos  
    static int numeroBrazos;  
  
    private String nombre;  
    private String apellido1;  
    private String apellido2;  
    private int edad;  
  
    //métodos  
  
}
```

Métodos

- Son las acciones que puede realizar la clase
- Usan los datos que hay en los campos o de los parámetros que reciben.
- Pueden devolver un resultado según un tipo de datos o nada indicando void
- Un método puede ser estático y no requiere que se creen objetos de la clase para usarlo.
- Pueden tener visibilidad pública, privada, protegida o por defecto

Según su funcionalidad se dividen en:

- Constructores
- Observadores (get)
- Modificadores (set)
- Métodos personalizados



Sintaxis

[visibilidad] [static] tipo nombre ([tipo param1][,tipo param.n])



Sobrecarga (Overloading)

- Varios métodos con el mismo nombre
- Tienen un número de parámetros diferente en cada uno de ellos

```
public class Persona {  
  
    //atributos  
    static int numeroBrazos;  
  
    private String nombre;  
    private String apellido1;  
    private String apellido2;  
    private int edad;  
  
    //métodos  
    //observador o getter  
    public String getNombre() {  
        return nombre;  
    }  
    //modificador o setter  
    public void setNombre(String nombre) {  
        this.nombre = nombre;  
    }  
    //método personalizado  
    public String saludar() {  
        return nombre + " " + apellido1 + " " + apellido2 + " te saluda";  
    }  
}
```



Constructores

- Son métodos encargados de inicializar las clases
- Si la clase no contiene ningún constructor, usa uno por defecto, vacío.
- Para instanciar o crear un nuevo objeto se invocará a un constructor junto con la palabra reservada `new`

```
Persona persona =  
    new Persona("Pepe", "Gines", "Gonzalez", 30);  
  
Persona persona2 =  
    new Persona("Carlota", "Miranda");  
  
Persona persona3 =  
    new Persona();  
persona3.setNombre("María");
```

```
public Persona() {  
  
}  
  
public Persona(String nombre, String apellido1) {  
    this.nombre = nombre;  
    this.apellido1 = apellido1;  
}  
  
public Persona(String nombre, String apellido1, String apellido2, int edad) {  
    this.nombre = nombre;  
    this.apellido1 = apellido1;  
    this.apellido2 = apellido2;  
    this.edad = edad;  
}
```



Destructores

- En Java no hay forma explícita de destruir objetos.
- Existe un recolector de basura (garbage collector) que se encarga de liberar memoria de objetos que no se usan o se han quedado sin referencias en memoria.
- Se puede invocar el recolector de basura, mediante `System.gc()`



Encapsulación

Encapsulación y visibilidad son dos aspectos fundamentales para aplicar niveles de protección de cara a acceder a las clases y sus miembros.

- **Public:** Tiene visibilidad desde cualquier parte del programa.
- **Protected:** Permite tener la visibilidad completa, excepto desde otro paquete
- **Package:** (No se pone ningún modificador). Tiene visibilidad solo dentro del mismo paquete.
- **Private:** Tiene visibilidad solo dentro de la misma clase

MODIFICADOR	Public	Protected	Package	Private
Acceso desde la misma clase	SI	SI	SI	SI
Acceso desde otras clases del mismo paquete	SI	SI	SI	NO
Acceso desde una subclase en el mismo paquete	SI	SI	SI	NO
Acceso desde subclases en otros paquetes	SI	SI	NO	NO
Acceso desde otras clases en otros paquetes	SI	NO	NO	NO

```
package es.encap;  
  
class A {  
  
}
```

```
package es.encap.nivell;  
  
public class B {  
  
    private A claseA;  
  
}
```

```
package es.encapsulamiento;  
  
import es.encap.*;  
import es.encap.nivell.*;  
  
public class C {  
  
    A claseA;  
    B claseB;  
  
}
```

Librerías y paquetes

Paquetes

- Son estructuras de carpetas
- Forma jerárquica
- Permiten agrupar clases que comparten misma funcionalidad
- Para definir un paquete se usa la palabra package
- Para importar una clase se usa la palabra import

```
package ejemplo;

import es.concesionario.modelo.Coche;

public class Test {

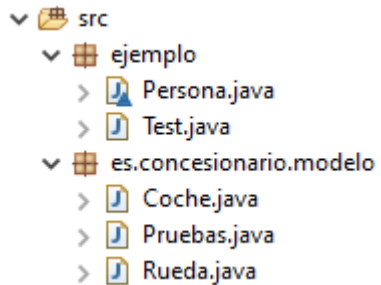
    public static void main(String[] args) {

        Persona persona =
            new Persona("Pepe", "García");

        Coche coche =
            new Coche("Seat", "Ibiza");

    }

}
```



Librerías

- Son agrupaciones de paquetes y clases
- Cumplen una funcionalidad específica
- Permiten su reutilización en otros programas



Java.io

Gestiona la entrada y salida de los programas..



java.util

Modelado de datos, colecciones, internalización, etc.



java.security

Funcionalidad de criptografía, firma digital y funciones de seguridad



Planteamiento de ejercicio

Queremos tener una aplicación destinada a la matriculación de alumnos en un centro educativo. Un centro educativo esta formado por aulas, alumnos, docentes y compuesto por cursos.



Diseña las clases que veas oportunas, teniendo en cuenta sus campos y métodos. Una vez finalizado mira de construir el centro educativo y que tenga su lógica.



Resumen

1. Concepto de clase
2. Campos o atributos
3. Métodos
4. Constructores e instanciación
5. Encapsulación
6. Librerías
7. Planteamiento de ejercicio

The background is a solid blue color with a subtle, large-scale grid pattern. Overlaid on this are numerous small, light blue arrows pointing in various directions, creating a sense of movement and flow. The overall aesthetic is modern and technological.

UNIVERSAE

— CHANGE YOUR WAY —