UNIVERSIDAD POLITÉCNICA DE SAN LUIS POTOSÍ

Programación III
POO JAVA

Modificadores de Métodos y Variables

MTI. Guadalupe Ledesma Ramos guadalupe.ledesma@upslp.edu.mx

Modificadores de variables

- **public**: Pública, puede acceder todo el mundo a esta variable.
- **Ninguno**: Es “amistosa”, puede ser accedida por cualquier miembro del package, pero no por otras clases que pertenezcan a otro package distinto.
- **protected**: Protegida, sólo pueden acceder a ella las clases hijas de la clase que posee la variable y las que estén en el mismo package.
- **private**: Privada, nadie salvo la clase misma puede acceder a estas variables. Pueden acceder a ella todas las instancias de la clase.
- **static**: Estática, esta variable es la misma para todas las instancias de una clase, todas comparten ese dato. Si una instancia lo modifica todas ven dicha modificación. Es una variable que pertenece a la clase, NO al objeto.
- **final**: Final, se emplea para definir constantes, un dato tipo final no puede variar nunca su valor. No tiene porque inicializarse en el momento de definirse, pero cuando se inicializa ya no puede cambiar su valor. Las constantes finales suelen ser también static.

Modificadores de un método

- **public**: Pública, puede acceder todo el mundo a este método.
- **Ninguno**: Es “amistoso”, puede ser accedida por cualquier miembro del package, pero no por otras clases que pertenecen a otro package.
- **protected**: Protegido, sólo pueden acceder a ella las clases hijas de la clase que posea el método y las que estén en el mismo package.
- **private**: Privada, nadie salvo la clase misma puede acceder a estos métodos.
- **static**: Estática, es un método al cual se puede invocar sin crear ningún objeto de dicha clase. Desde un método estático sólo podemos invocar otros métodos que también sean estáticos. Si se invoca desde la clase en la que se encuentra definido, basta con escribir su nombre. Si se le invoca desde una clase distinta, debe anteponerse a su nombre, el de la clase en la que se encuentra seguido del operador punto (.) <NombreClase>.metodoEstatico
- **final**: Final, se trata de un método que no podrá ser cambiado por ninguna clase que herede de la clase donde se definió. Es un método que no se puede “sobrescribir”.

Modificadores de acceso

	La misma clase	Otra clase del mismo paquete	Subclase de otro paquete	Otra clase de otro paquete
public	X	X	X	X
protected	X	X	X	
default	X	X		
private	X			

Variables estáticas o de clase

- Son propias únicamente de la clase y no de los objetos que pueden crearse de la misma, por lo tanto, sus valores son compartidos por todos los objetos de la clase. Van precedidas del modificador static.

- Si se invoca desde la clase en la que se encuentra definido, basta con escribir su nombre.

- Si se le invoca desde una clase distinta, debe anteponerse a su nombre, el de la clase en la que se encuentra seguido del operador punto (.) <NombreClase>.variableEstatica

- Suelen emplearse para definir constantes comunes a todos los objetos de la clase.

Ejemplo variables y método final

```
iables | Output - Validacion (run) ×
run:
Este método no se sobreescribe
Mi cadena Java Al declararse se inicializa
BUILD SUCCESSFUL (total time: 1 second)
```

```java
package validacion;
public class VariableFinalStatic {
    private final String CADENA1;
    private static String cadena2;
    private final static String CADENA3="Al declararse se inicializa";

    public VariableFinalStatic(){
        CADENA1="Mi cadena";
        cadena2 = "Java";
        //CADENA3="No se puede es final y static";
    }
    public final static void inicializa(){
        System.out.println("Este método no se sobreescribe");
    }
    void metodo(){
        cadena2="Puede acceder a todas las variables";
        inicializa();
    }
    final void imprime(){
        System.out.println(CADENA1 + " "+ cadena2 + " " + CADENA3);
    }
    public static void main(String args[]){
        inicializa();
        // metodo(); // No se puede ya que no es static
        VariableFinalStatic v1 = new VariableFinalStatic();
        v1.imprime();
    }
}
```