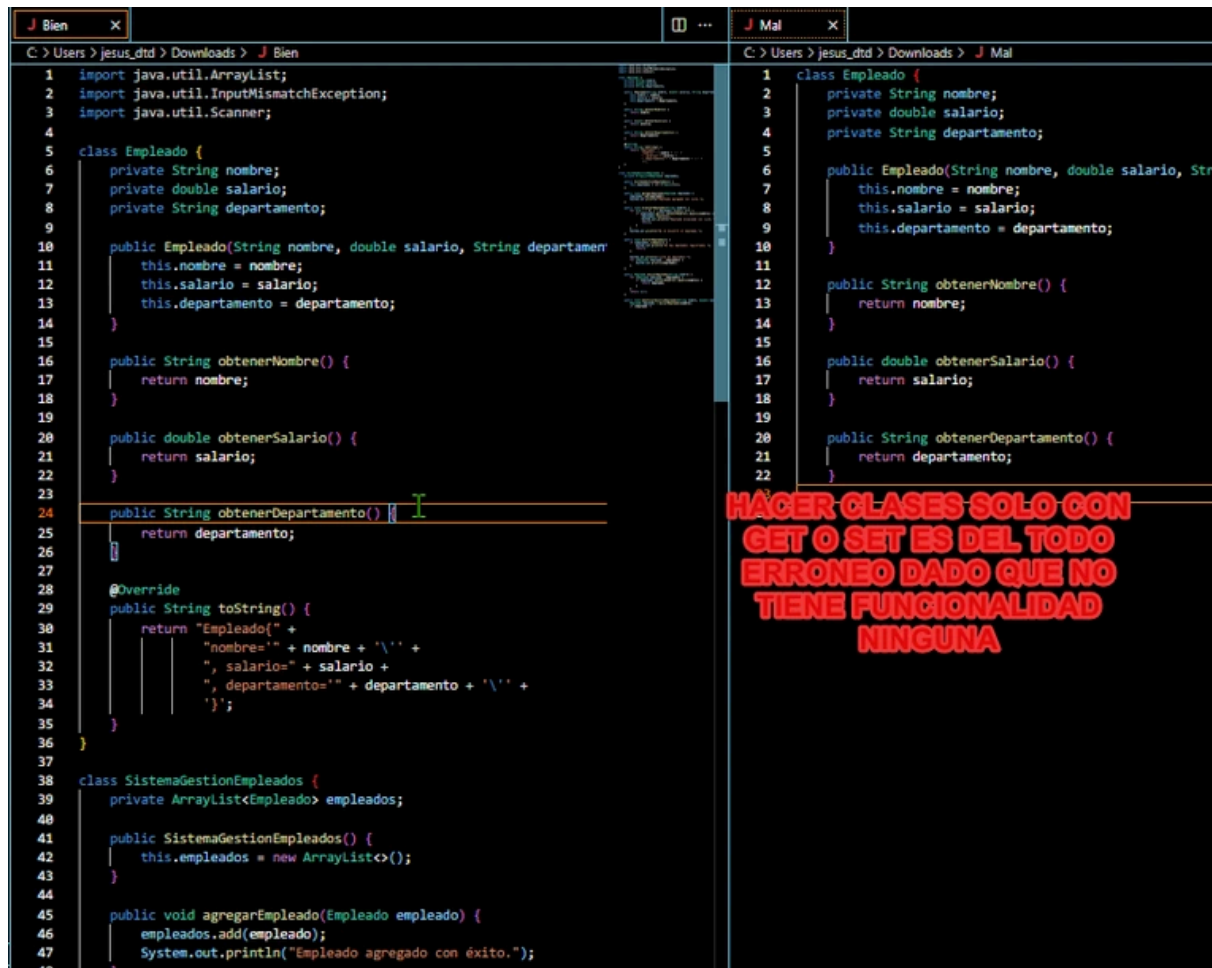


MALOS OLORES

CLASES TRIVIALES



```
1 import java.util.ArrayList;
2 import java.util.InputMismatchException;
3 import java.util.Scanner;
4
5 class Empleado {
6     private String nombre;
7     private double salario;
8     private String departamento;
9
10    public Empleado(String nombre, double salario, String departamen
11        this.nombre = nombre;
12        this.salario = salario;
13        this.departamento = departamento;
14    }
15
16    public String obtenerNombre() {
17        return nombre;
18    }
19
20    public double obtenerSalario() {
21        return salario;
22    }
23
24    public String obtenerDepartamento() {
25        return departamento;
26    }
27
28    @Override
29    public String toString() {
30        return "Empleado{" +
31            "nombre='" + nombre + '\'' +
32            ", salario=" + salario +
33            ", departamento='" + departamento + '\'' +
34            '}';
35    }
36 }
37
38 class SistemaGestionEmpleados {
39     private ArrayList<Empleado> empleados;
40
41     public SistemaGestionEmpleados() {
42         this.empleados = new ArrayList<>();
43     }
44
45     public void agregarEmpleado(Empleado empleado) {
46         empleados.add(empleado);
47         System.out.println("Empleado agregado con éxito.");
48     }
49 }
```

HACER CLASES SOLO CON GET O SET ES DEL TODO ERRONEO DADO QUE NO TIENE FUNCIONALIDAD NINGUNA

COMENTARIOS EXCESIVAMENTE LARGOS

J Bien

C:\Users\jesus_dhd\Downloads> J Bien

```
1 import java.util.ArrayList;
2 import java.util.InputMismatchException;
3 import java.util.Scanner;
4
5 class Empleado {
6     private String nombre;
7     private double salario;
8     private String departamento;
9
10    public Empleado(String nombre, double salario, String departamen
11        this.nombre = nombre;
12        this.salario = salario;
13        this.departamento = departamento;
14    }
15
16    public String obtenerNombre() {
17        return nombre;
18    }
19
20    public double obtenerSalario() {
21        return salario;
22    }
23
24    public String obtenerDepartamento() {
25        return departamento;
26    }
27
28    @Override
29    public String toString() {
30        return "Empleado{" +
31            "nombre=" + nombre + '\'' +
32            ", salario=" + salario +
33            ", departamento=" + departamento + '\'' +
34            '}';
35    }
36
37    class SistemaGestionEmpleados {
38        private ArrayList<Empleado> empleados;
39    }
```

J Mal

C:\Users\jesus_dhd\Downloads> J Mal

```
1 import java.util.ArrayList;
2 import java.util.InputMismatchException;
3 import java.util.Scanner;
4
5 /*La clase SistemaGestionEmpleados implementa funcionalidades básicas
6 para la gestión de empleados. Se podrían agregar más funcionalidades,
7 como la edición de información de empleados, la generación de informes,
8 la exportación de datos, etc.
9 La clase utiliza una lista de objetos Empleado para almacenar la información
10 de los empleados. Esta es una buena opción para manejar una cantidad moderada de datos.
11 Si se espera manejar una gran cantidad de datos, se podría considerar utilizar una base
12 de datos.
13 El código está bien estructurado y utiliza comentarios para explicar su propósito.
14 Esto facilita la comprensión y el mantenimiento del código.
15 El sistema de gestión de empleados presentado en esta clase Java es una herramienta útil
16 para organizar y administrar información sobre los empleados de una empresa.
17 El código está bien escrito y documentado, y se puede ampliar fácilmente para agregar
18 más funcionalidades.
19 Este comentario es excesivamente largo y detallado. En un contexto real, es probable
20 que un comentario de este tipo no sea necesario. Sin embargo, este comentario puede
21 ser útil como referencia para comprender mejor la estructura y funcionalidad del código
22 */
23 class Empleado {
24     private String nombre;
25     private double salario;
26     private String departamento;
27
28     public Empleado(String nombre, double salario, String departamento) {
29         this.nombre = nombre;
30         this.salario = salario;
31         this.departamento = departamento;
32     }
33
34     public String obtenerNombre() {
35         return nombre;
36     }
37
38     public double obtenerSalario() {
39         return salario;
40     }
```

No es bueno poner comentarios
excesivamente largos dado que suelen
ser confusos y mala imagen para el
programa

CLASES CON POCO CONTENIDO O CON Poca RAZÓN DE EXISTIR

J Bien

C:\Users\jesus_dhd\Downloads> J Bien

```
1 import java.util.ArrayList;
2 import java.util.InputMismatchException;
3 import java.util.Scanner;
4
5 class Empleado {
6     private String nombre;
7     private double salario;
8     private String departamento;
9
10    public Empleado(String nombre, double salario, String departamen
11        this.nombre = nombre;
12        this.salario = salario;
13        this.departamento = departamento;
14    }
15
16    public String obtenerNombre() {
17        return nombre;
18    }
19
20    public double obtenerSalario() {
21        return salario;
22    }
23
24    public String obtenerDepartamento() {
25        return departamento;
26    }
27
28    @Override
29    public String toString() {
30        return "Empleado{" +
31            "nombre=" + nombre + '\'' +
32            ", salario=" + salario +
33            ", departamento=" + departamento + '\'' +
34            '}';
35    }
36
37 }
```

J Mal

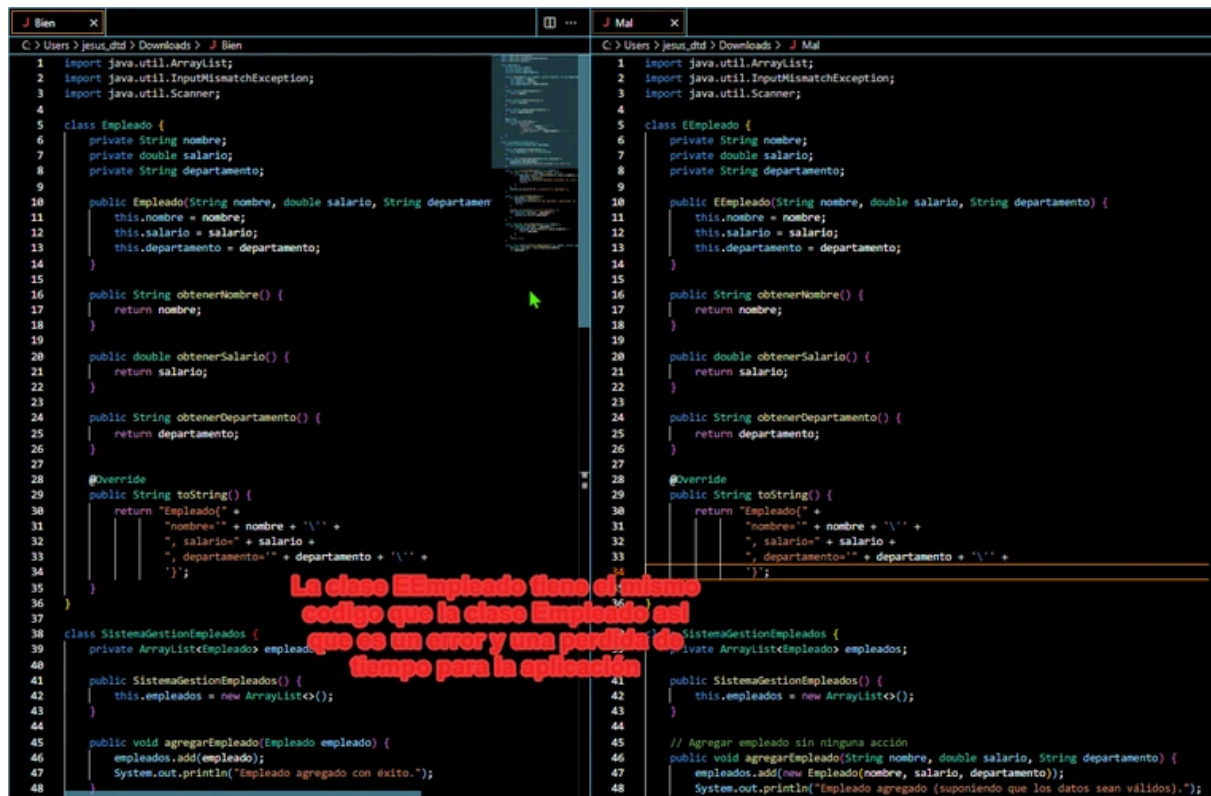
C:\Users\jesus_dhd\Downloads> J Mal

```
1 import java.util.ArrayList;
2 import java.util.InputMismatchException;
3 import java.util.Scanner;
4
5 class Empleado {
6     private String nombre;
7     private double salario;
8     private String departamento;
9
10    public Empleado(String nombre, double salario, String departamento) {
11        this.nombre = nombre;
12        this.salario = salario;
13        this.departamento = departamento;
14    }
15
16    public String obtenerNombre() {
17        return nombre;
18    }
19
20    public double obtenerSalario() {
21        return salario;
22    }
23
24    public String obtenerDepartamento() {
25        return departamento;
26    }
27
28    @Override
29    public String toString() {
30        return "Empleado{" +
31            "nombre=" + nombre + '\'' +
32            ", salario=" + salario +
33            ", departamento=" + departamento + '\'' +
34            '}';
35    }
36
37 }
```

La clase
SistemaGestionEmpleados no tiene
cometido dado que no hace nada
util ni que merezca la pena ni para
el usuario ni el programador

```
38 class SistemaGestionEmpleados {
39     private ArrayList<Empleado> empleados;
40
41     public SistemaGestionEmpleados() {
42         this.empleados = new ArrayList<>();
43     }
44
45     // Agregar empleado sin ninguna acción
46     public void agregarEmpleado(String nombre, double salario, String departamento) {
47         empleados.add(new Empleado(nombre, salario, departamento));
48         System.out.println("Empleado agregado (suponiendo que los datos sean válidos).");
49     }
```

EL MISMO CÓDIGO REPETIDO EN CLASES DISTINTAS



```
1 import java.util.ArrayList;
2 import java.util.InputMismatchException;
3 import java.util.Scanner;
4
5 class Empleado {
6     private String nombre;
7     private double salario;
8     private String departamento;
9
10    public Empleado(String nombre, double salario, String departamento) {
11        this.nombre = nombre;
12        this.salario = salario;
13        this.departamento = departamento;
14    }
15
16    public String obtenerNombre() {
17        return nombre;
18    }
19
20    public double obtenerSalario() {
21        return salario;
22    }
23
24    public String obtenerDepartamento() {
25        return departamento;
26    }
27
28    @Override
29    public String toString() {
30        return "Empleado[" +
31            "nombre=" + nombre + '\'' +
32            ", salario=" + salario +
33            ", departamento=" + departamento + '\'' +
34            ']';
35    }
36 }
37
38 class SistemaGestionEmpleados {
39     private ArrayList<Empleado> empleados;
40
41     public SistemaGestionEmpleados() {
42         this.empleados = new ArrayList<>();
43     }
44
45     public void agregarEmpleado(Empleado empleado) {
46         empleados.add(empleado);
47         System.out.println("Empleado agregado con éxito.");
48     }
49 }
```

La clase **Empleado** tiene el mismo código que la clase **Empleado** así que es un error y una pérdida de tiempo para la aplicación

```
1 import java.util.ArrayList;
2 import java.util.InputMismatchException;
3 import java.util.Scanner;
4
5 class Empleado {
6     private String nombre;
7     private double salario;
8     private String departamento;
9
10    public Empleado(String nombre, double salario, String departamento) {
11        this.nombre = nombre;
12        this.salario = salario;
13        this.departamento = departamento;
14    }
15
16    public String obtenerNombre() {
17        return nombre;
18    }
19
20    public double obtenerSalario() {
21        return salario;
22    }
23
24    public String obtenerDepartamento() {
25        return departamento;
26    }
27
28    @Override
29    public String toString() {
30        return "Empleado[" +
31            "nombre=" + nombre + '\'' +
32            ", salario=" + salario +
33            ", departamento=" + departamento + '\'' +
34            ']';
35    }
36 }
37
38 class SistemaGestionEmpleados {
39     private ArrayList<Empleado> empleados;
40
41     public SistemaGestionEmpleados() {
42         this.empleados = new ArrayList<>();
43     }
44
45     // Agregar empleado sin ninguna acción
46     public void agregarEmpleado(String nombre, double salario, String departamento) {
47         empleados.add(new Empleado(nombre, salario, departamento));
48         System.out.println("Empleado agregado (suponiendo que los datos sean válidos).");
49     }
50 }
```

ATRIBUTOS DE USO ESPORÁDICO

```
1 import java.util.ArrayList;
2 import java.util.InputMismatchException;
3 import java.util.Scanner;
4
5 class Empleado {
6     private String nombre;
7     private double salario;
8     private String departamento;
9
10    public Empleado(String nombre, double salario, String departamen
11    {
12        this.nombre = nombre;
13        this.salario = salario;
14        this.departamento = departamento;
15    }
16
17    public String obtenerNombre() {
18        return nombre;
19    }
20
21    public double obtenerSalario() {
22        return salario;
23    }
24
25    public String obtenerDepartamento() {
26        return departamento;
27    }
28
29    @Override
30    public String toString() {
31        return "Empleado{" +
32            "nombre=" + nombre + '\'' +
33            ", salario=" + salario +
34            ", departamento=" + departamento + '\'' +
35            '}';
36    }
37
38    class SistemaGestionEmpleados {
39        private ArrayList<Empleado> empleados;
40
41        public SistemaGestionEmpleados() {
42            this.empleados = new ArrayList<>();
43        }
44
45        public void agregarEmpleado(Empleado empleado) {
46            empleados.add(empleado);
47            System.out.println("Empleado agregado con éxito.");
48        }
49    }
50
51    class Empleado {
52        private String nombre;
53        private double salario;
54        private String departamento;
55        private String fechaNacimiento; // Atributo de uso esporádico
56        private String telefonoPersonal; // Atributo de uso esporádico
57
58        public Empleado(String nombre, double salario, String departamento, String fechaNaci
59        {
60            this.nombre = nombre;
61            this.salario = salario;
62            this.departamento = departamento;
63            this.fechaNacimiento = fechaNacimiento;
64            this.telefonoPersonal = telefonoPersonal;
65        }
66
67        @Override
68        public String toString() {
69            return "Empleado{" +
70                "nombre=" + nombre + '\'' +
71                ", salario=" + salario +
72                ", departamento=" + departamento + '\'' +
73                ", fechaNacimiento=" + fechaNacimiento + '\'' +
74                ", telefonoPersonal=" + telefonoPersonal + '\'' +
75                '}';
76        }
77    }
78
79    class SistemaGestionEmpleados {
80        private ArrayList<Empleado> empleados;
81
82        public void mostrarEmpleadosConFechaNacimiento() {
83            if (empleados.isEmpty()) {
84                System.out.println("No hay empleados registrados.");
85                return;
86            }
87
88            System.out.println("Lista de empleados con fecha de nacimiento:");
89            for (Empleado empleado : empleados) {
90                System.out.println(empleado.obtenerNombre() + " - Fecha de nacimiento: " + emp
91            }
92        }
93    }
```

COMANDOS GIT

CONFIGURACIÓN

```
jesus_dtd@DESKTOP-4J1KT02 MINGW64 ~
$ git config --global user.name "Terino"

jesus_dtd@DESKTOP-4J1KT02 MINGW64 ~
$ git config --global user.email "jesusteri14@gmail.com"
```

CREACIÓN DE PROYECTO

```
jesus_dtd@DESKTOP-4J1KT02 MINGW64 ~
$ git init
Initialized empty Git repository in C:/Users/jesus_dtd/.git/
```

Se inicia un proyecto en la carpeta .git

TRABAJO CON ARCHIVOS

```
jesus_dtd@DESKTOP-4J1KT02 MINGW64 ~ (master)
$ git add Downloads/Bien

jesus_dtd@DESKTOP-4J1KT02 MINGW64 ~ (master)
$ git status
warning: could not open directory 'Configuración local/': Permission denied
warning: could not open directory 'Cookies/': Permission denied
warning: could not open directory 'Datos de programa/': Permission denied
warning: could not open directory 'Entorno de red/': Permission denied
warning: could not open directory 'Impresoras/': Permission denied
warning: could not open directory 'Menú Inicio/': Permission denied
warning: could not open directory 'Mis documentos/': Permission denied
warning: could not open directory 'Plantillas/': Permission denied
warning: could not open directory 'Reciente/': Permission denied
warning: could not open directory 'SendTo/': Permission denied
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   Downloads/Bien
```

CONFIRMACIÓN DE CAMBIOS

```
jesus_dtd@DESKTOP-4J1KT02 MINGW64 ~ (master)
$ git commit -m "Añadido el archivo Bien"
[master (root-commit) 54a1a91] Añadido el archivo Bien
 1 file changed, 84 insertions(+)
 create mode 100644 Downloads/Bien
```

RESTABLECIMIENTO DE CAMBIOS

```
jesus_dtd@DESKTOP-4J1KT02 MINGW64 ~ (master)
$ git checkout -- Downloads/Bien
```

RAMAS

```

jesus_dtd@DESKTOP-4J1KT02 MINGW64 ~ (master)
$ git branch
* master

jesus_dtd@DESKTOP-4J1KT02 MINGW64 ~ (master)
$ git branch newbranch

jesus_dtd@DESKTOP-4J1KT02 MINGW64 ~ (master)
$ git branch
* master
  newbranch

```

PROGRAMA JAVADOC

```

import java.util.Random;
import java.util.Scanner;

/**
 * Clase principal del juego Adivina el Número.
 *
 * @author [Jesús Terino Rodríguez]
 * @version 3.0
 * @since 2024-05-11
 */
public class AdivinaElNumero {

    /**
     * Punto de entrada del programa.
     *
     * @param args Argumentos de línea de comandos.
     */
    public static void main(String[] args) {
        Random random = new Random();
        Scanner sc = new Scanner(System.in);

        /**
         * Número aleatorio que el usuario debe adivinar.
         *
         * @see java.util.Random#nextInt(int)
         */
        int numeroSecreto = random.nextInt(100) + 1; // Número aleatorio entre 1 y 100

        /**
         * Número de intentos restantes para adivinar el número secreto.
         */
        int intentosRestantes = 10;

        System.out.println("¡Bienvenido al juego Adivina el Número!");
        System.out.println("Tienes " + intentosRestantes + " intentos para adivinar un número entre 1 y 100.");
        int intento;
        do {
            System.out.print("Ingresa tu suposición: ");
            intento = sc.nextInt();

            if (intento == numeroSecreto) {
                System.out.println("¡Felicidades! Adivinaste el número en " + (10 - intentosRestantes) + " intentos.");
                break;
            } else if (intento < numeroSecreto) {
                System.out.println("Tu suposición es demasiado baja. Intenta de nuevo.");
            } else {
                System.out.println("Tu suposición es demasiado alta. Intenta de nuevo.");
            }
        }
    }
}

```

Estas son las etiquetas incluidas con Javadoc

```

        if (intento == numeroSecreto) {
            System.out.println("¡Felicidades! Adivinaste el número en " + (10 - intentosRestantes) + " intentos.");
            break;
        } else if (intento < numeroSecreto) {
            System.out.println("Tu suposición es demasiado baja. Intenta de nuevo.");
        } else {
            System.out.println("Tu suposición es demasiado alta. Intenta de nuevo.");
        }

        intentosRestantes--;
    } while (intentosRestantes > 0 && intento != numeroSecreto);

    if (intentosRestantes == 0) {
        System.out.println("¡Has perdido! El número secreto era: " + numeroSecreto);
    }

    sc.close();
}

```

Esta son las otras etiquetas que he puesto de javadoc

```

/**
 * Método que verifica si la entrada del usuario es un número válido.
 *
 * @param entrada Cadena que contiene la entrada del usuario.
 * @return 'true' si la entrada es un número válido, 'false' en caso contrario.
 * @throws NumberFormatException Si la entrada no es un número.
 */
private static boolean esNumeroValido(String entrada) {
    try {
        Integer.parseInt(entrada);
        return true;
    } catch (NumberFormatException e) {
        return false;
    }
}
}

```