

TAREA DEL TEMA 4

ETS

15/05/2023

Autor: Derimán Tejera Fumero

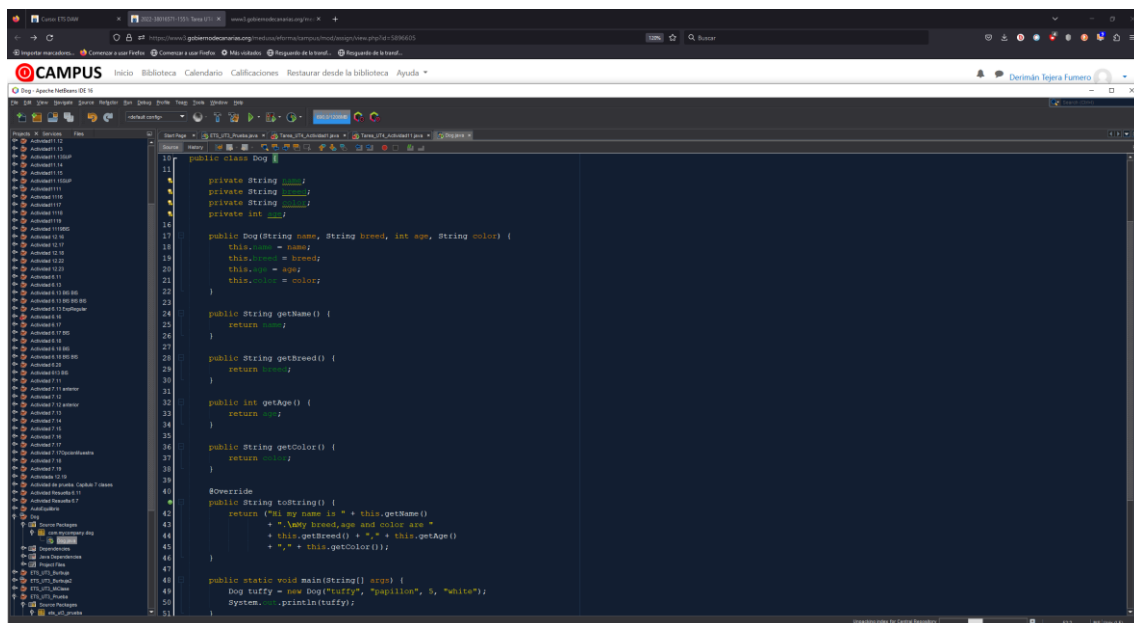
ÍNDICE

Actividad 1.....	2
Actividad 2.....	5

Actividad 1: Revisa el código del archivo Dog.java y encuentra todas las malas prácticas de programación que están presentes. Deberás realizar las correcciones, ya sea de forma manual o utilizando las herramientas de refactorización propias de tu IDE. También puedes utilizar un analizador de código si lo crees conveniente.

Debes añadir al informe, el código totalmente corregido, la explicación de los cambios que has realizado y en base a qué lo has hecho. Recuerda personalizar el entorno y mostrar claramente todo lo solicitado.

El código corregido:

A screenshot of an IDE (IntelliJ IDEA) showing the corrected code for Dog.java. The code is as follows:

```
1 public class Dog {
2     private String name;
3     private String breed;
4     private String color;
5     private int age;
6
7     public Dog(String name, String breed, int age, String color) {
8         this.name = name;
9         this.breed = breed;
10        this.age = age;
11        this.color = color;
12    }
13
14    public String getName() {
15        return name;
16    }
17
18    public String getBreed() {
19        return breed;
20    }
21
22    public int getAge() {
23        return age;
24    }
25
26    public String getColor() {
27        return color;
28    }
29
30    @Override
31    public String toString() {
32        return "Mi my name is " + this.getName()
33            + " " + this.breed + "and color: " +
34            this.getColor() + " " + this.getAge()
35            + " " + this.getColor();
36    }
37
38    public static void main(String[] args) {
39        Dog tuffy = new Dog("tuffy", "pugilón", 5, "white");
40        System.out.println(tuffy);
41    }
42 }
```

La versión en texto sería:

```
public class Dog {

    private String name;
    private String breed;
    private String color;
    private int age;

    public Dog(String name, String breed, int age, String color) {
        this.name = name;
        this.breed = breed;
        this.age = age;
        this.color = color;
    }

    public String getName() {
        return name;
    }

    public String getBreed() {
        return breed;
    }

    public int getAge() {
        return age;
    }

    public String getColor() {
        return color;
    }

}
```

```

    }

    @Override
    public String toString() {
        return ("Hi my name is " + this.getName()
            + ".\nMy breed,age and color are "
            + this.getBreed() + "," + this.getAge()
            + "," + this.getColor());
    }

    public static void main(String[] args) {
        Dog tuffy = new Dog("tuffy", "papillon", 5, "white");
        System.out.println(tuffy);
    }
}

```

Las modificaciones realizadas son:

1. Las variables de instancia de han hecho private para mantener el encapsulamiento, también se han separado para que sea mas legible, dónde antes era:

```

String name, breed, color;
int age;

```

Ahora será:

```

private String name;
private String breed;
private String color;
private int age;

```

2. Los nombres de los métodos han sido modificados ya que deben empezar con minúscula, mas concretamente utilizando “camel case”. Dónde antes era:

```

// method 1
public String GetName()
{
    return name;
}

// method 2
public String GetBreed()
{
    return breed;
}

// method 3
public int GetAge()
{
    return age;
}

// method 4
public String GetColor()
{
    return color;
}

```

Ahora será:

```

public String getName() {
    return name;
}

public String getBreed() {
    return breed;
}

public int getAge() {
    return age;
}

public String getColor() {
    return color;
}

```

3. No es necesaria una llamada a los getters, ya que podemos acceder directamente al ser "public String toString()" un método de la misma clase.

```

public String toString() {
    return ("Hi my name is " + this.GetName()
        + ".\nMy breed,age and color are "
        + this.GetBreed() + "," + this.GetAge()
        + "," + this.GetColor());
}

```

Ahora será:

```

public String toString() {
    return("Hi my name is "+ this.name +
        ".\nMy breed, age and color are " +
        this.breed + "," + this.age +
        "," + this.color);
}

```

4. No es necesario usar toString cuando mostremos por pantalla:

Donde antes era:

```

public static void main(String[] args) {
    Dog tuffy = new Dog("tuffy", "papillon", 5, "white");
    System.out.println(tuffy.toString());
}

```

Ahora será:

```

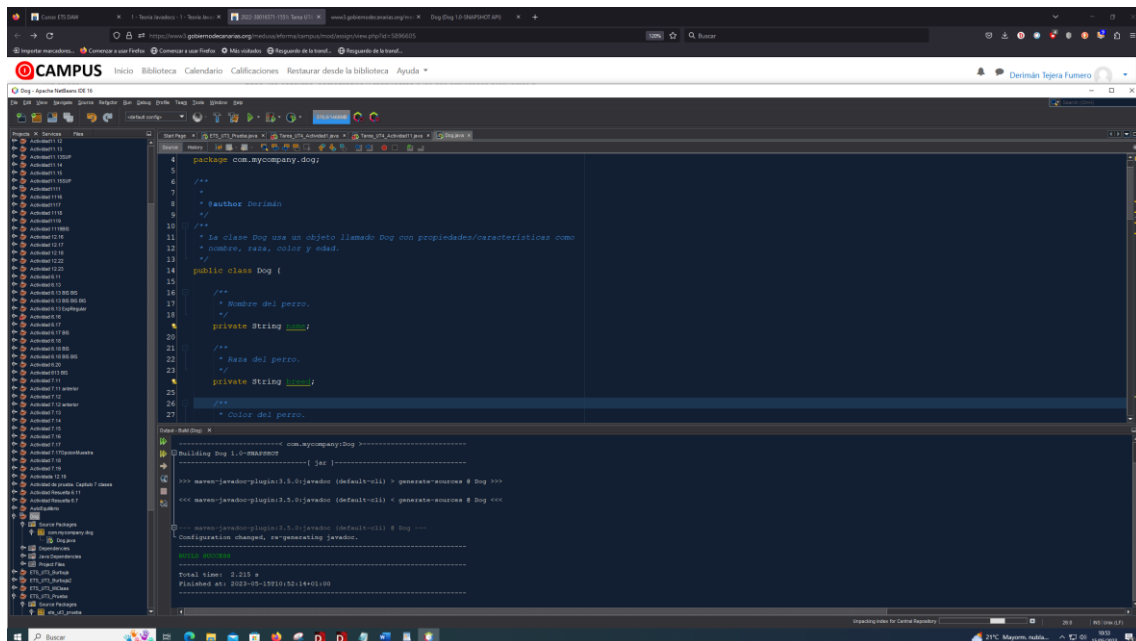
public static void main(String[] args) {
    Dog tuffy = new Dog("tuffy","papillon", 5, "white");
    System.out.println(tuffy);
}

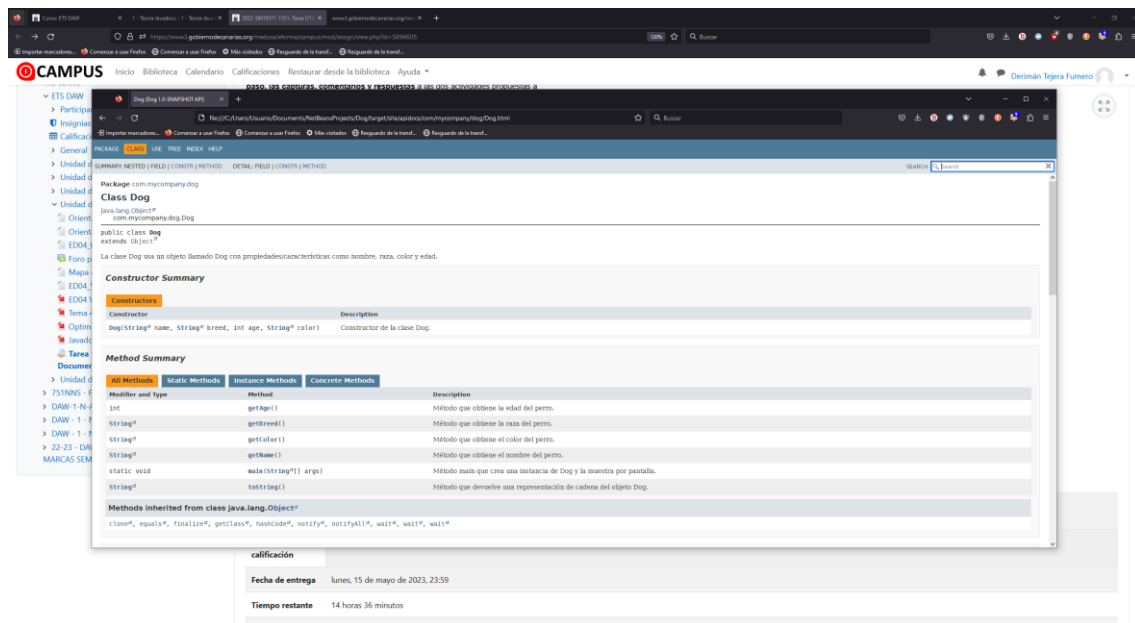
```

Actividad 2: Añadirás al código ya corregido etiquetas Javadocs para que puedes generar la documentación correspondiente. Recuerda usar los comentarios. Deberás añadir al informe el código fuente con las etiquetas y comentarios añadidos. Adjunta también, en una carpeta comprimida, toda la documentación Javadocs generada.

Guarda el documento en formato PDF y súbelo a esta actividad. Sube también la carpeta comprimida con la documentación Javadocs.

Se ha generado la documentación javadoc del código:





La ubicación real de los documentos generados es:
 \Documents\NetBeansProjects\Dog\target\site\apidocs

FIN