Entornos de desarrollo

Act.01_UT05. Refactorización, optimización y documentación. Javadoc

Francisco José García Cutillas | 1FPGS_DAM

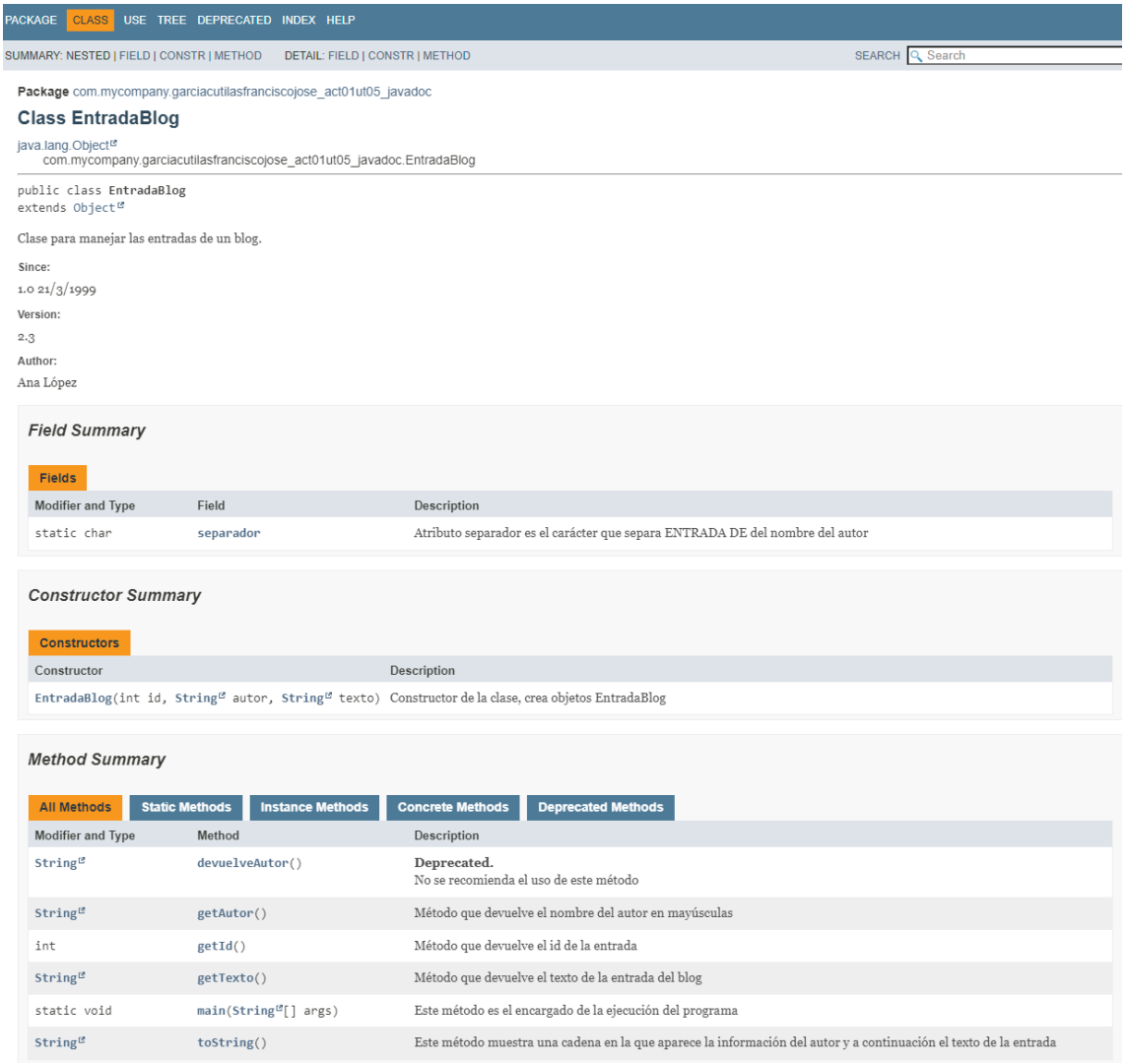
Índice

Parte 1

Ejercicio 1

```java
package com.mycompany.garciacutilasfranciscojose_act01ut05_javadoc;

/**
 * Clase para manejar las entradas de un blog.
 *
 * @author Ana López
 * @since 1.0 21/3/1999
 * @version 2.3
 */
public class EntradaBlog {

    /**
     * Atributo separador es el carácter que separa ENTRADA DE del nombre del autor
     */
    public static char separador = ':';
    private int id;
    private String texto;
    private String autor;

    /**
     * Constructor de la clase, crea objetos EntradaBlog
     * @param id Número de entrada al blog
     * @param autor Nombre del autor de la entrada
     * @param texto Texto que contiene la entrada
     * @throws IllegalArgumentException Este objeto lanza la excepción si el número de
     * entrada es negativo
     */
    public EntradaBlog(int id, String autor, String texto) throws IllegalArgumentException {
        if (id <= 0) {
            throw new IllegalArgumentException( s: "El id no puede ser negativo");
        }
        this.id = id;
        this.autor = autor;
        this.texto = texto;
    }

    /**
     * Este método muestra una cadena en la que aparece la información del autor y
     * a continuación el texto de la entrada
     * @return devuelve la información del autor con su entrada correspondiente
     */
    @Override
    public String toString() {
        String cad = "";
        cad += "\nENTRADA DE" + separador + autor;
        cad += "\n " + texto;
        return cad;
    }

}
```

```
    /**
     * Método que devuelve el id de la entrada
     * @return id de la entrada del blog
     */
    public int getId() {
        return this.id;
    }

    /**
     * Método que devuelve el texto de la entrada del blog
     * @return devuelve el texto contenido en la entrada del blog
     */
    public String getTexto() {
        return this.texto;
    }

    /**
     * Método que devuelve el nombre del autor en mayúsculas
     * @return devuelve el nombre del autor en mayúsculas
     */
    public String getAutor() {
        return this.autor.toUpperCase();
    }

    /**
     * Método que devuelve el nombre del autor
     * @deprecated No se recomienda el uso de este método
     * @see #getAutor() se recomienda el uso del método getAutor()
     * @return devuelve el nombre del autor
     */
     @Deprecated
    public String devuelveAutor() {
        return this.autor;
    }

    /**
     * Este método es el encargado de la ejecución del programa
     * @param args No tiene por qué recibir argumentos
     */
    public static void main(String[] args) {
        EntradaBlog el = new EntradaBlog( id: -3,  autor: "ana",  texto: "Últimas noticias, sale a la venta Windows 9");
        System.out.println( x: el);
    }

}
```



PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD SEARCH Search

Package com.mycompany.garciacutilasfranciscojose_act01ut05_javadoc

Class EntradaBlog

java.lang.Object
com.mycompany.garciacutilasfranciscojose_act01ut05_javadoc.EntradaBlog

```
public class EntradaBlog
extends Object
```

Clase para manejar las entradas de un blog.

Since:

1.0 21/3/1999

Version:

2.3

Author:

Ana López

Field Summary

Fields

Modifier and Type	Field	Description
static char	separador	Atributo separador es el carácter que separa ENTRADA DE del nombre del autor

Constructor Summary

Constructors

Constructor	Description
EntradaBlog(int id, String autor, String texto)	Constructor de la clase, crea objetos EntradaBlog

Method Summary

All Methods | Static Methods | Instance Methods | Concrete Methods | Deprecated Methods

Modifier and Type	Method	Description
String	devuelveAutor()	**Deprecated.** No se recomienda el uso de este método
String	getAutor()	Método que devuelve el nombre del autor en mayúsculas
int	getId()	Método que devuelve el id de la entrada
String	getTexto()	Método que devuelve el texto de la entrada del blog
static void	main(String[] args)	Este método es el encargado de la ejecución del programa
String	toString()	Este método muestra una cadena en la que aparece la información del autor y a continuación el texto de la entrada

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Field Details

separador

```
public static char separador
```

Atributo separador es el carácter que separa ENTRADA DE del nombre del autor

Constructor Details

EntradaBlog

```
public EntradaBlog(int id,
                   String autor,
                   String texto)
            throws IllegalArgumentException
```

Constructor de la clase, crea objetos EntradaBlog

Parameters:

`id` - Número de entrada al blog

`autor` - Nombre del autor de la entrada

`texto` - Texto que contiene la entrada

Throws:

`IllegalArgumentException` - Este objeto lanza la excepción si el número de entrada es negativo

Method Details

toString

```
public String toString()
```

Este método muestra una cadena en la que aparece la información del autor y a continuación el texto de la entrada

Overrides:

`toString` in class `Object`

Returns:

devuelve la información del autor con su entrada correspondiente

getId

`public int getId()`

Método que devuelve el id de la entrada

Returns:
id de la entrada del blog

getTexto

`public String getTexto()`

Método que devuelve el texto de la entrada del blog

Returns:
devuelve el texto contenido en la entrada del blog

getAutor

`public String getAutor()`

Método que devuelve el nombre del autor en mayúsculas

Returns:
devuelve el nombre del autor en mayúsculas

devuelveAutor

`@Deprecated`
`public String devuelveAutor()`

Deprecated.
No se recomienda el uso de este método

Método que devuelve el nombre del autor

Returns:
devuelve el nombre del autor

See Also:
`se recomienda el uso del método getAutor()`

main

`public static void main(String[] args)`

Este método es el encargado de la ejecución del programa

Parameters:
`args` - No tiene por qué recibir argumentos

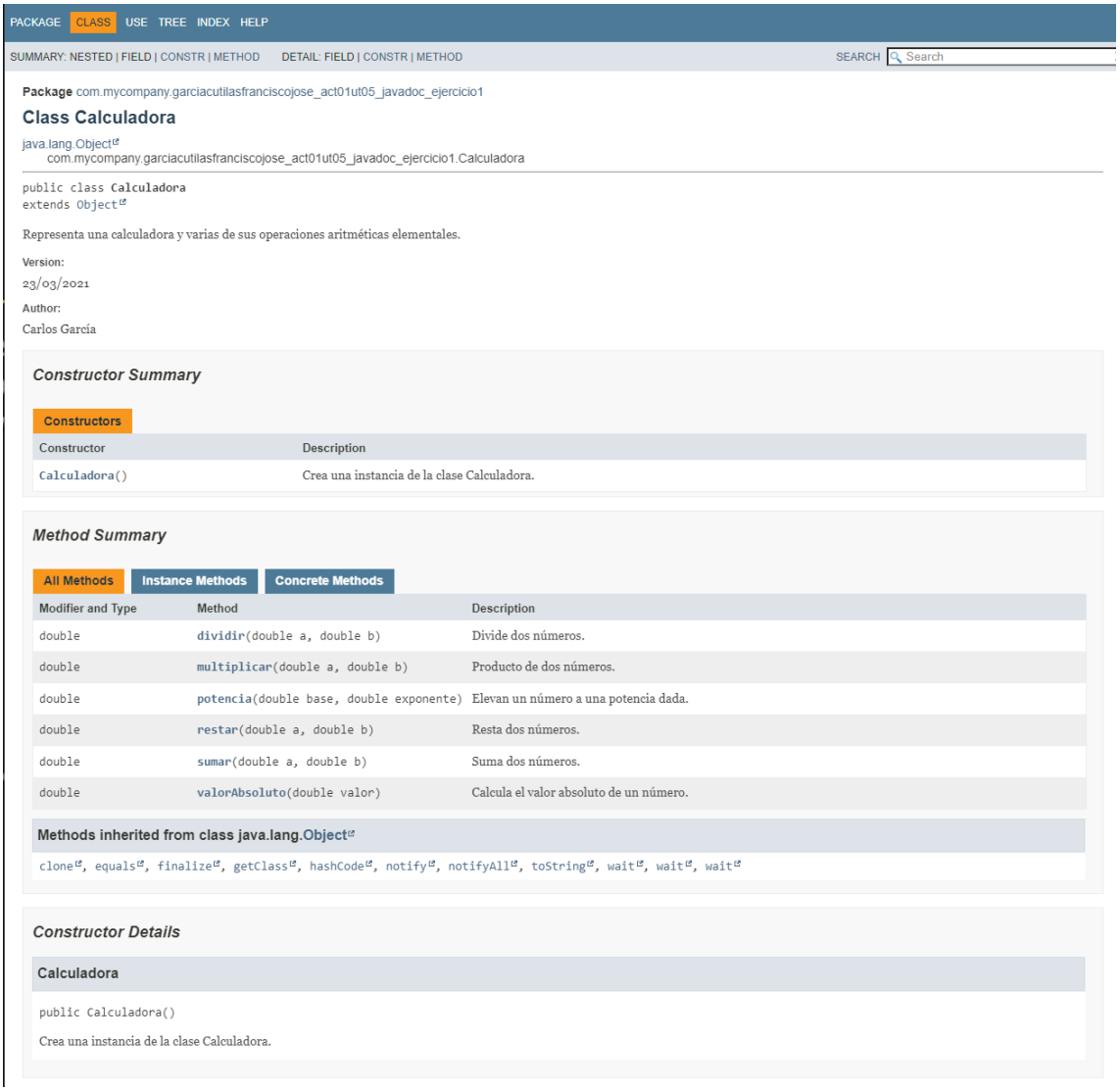
Parte 2

Ejercicio 1

Haz al menos 2 capturas de pantalla de la documentación generada automáticamente en Netbeans, a partir del siguiente código, que ya tiene los comentarios JavaDocs. Mete en el documento tanto el código como las capturas de pantalla.

```java
package com.mycompany.garciacutilasfranciscojose_act01ut05_javadoc_ejercicio1;

/**
 * Representa una calculadora y varias de sus operaciones aritméticas
 * elementales.
 *
 * @author Carlos García
 * @version 23/03/2021
 */
public class Calculadora {

    /**
     * Representa la constante matemática de pi.
     */
    private static final double PI = 3.14159265358979;
    /**
     * Representa la constante matemática de Euler.
     */
    private static final double E = 2.718281828459;

    /**
     * Crea una instancia de la clase Calculadora.
     */
    public Calculadora() {
    }

    /**
     * Suma dos números.
     *
     * @param a Primer operando de la suma.
     * @param b Segundo operando de la suma.
     * @return Suma de los dos números pasados como argumentos.
     */
    public double sumar(double a, double b) {
        return a + b;
    }

    /**
     * Resta dos números.
     *
     * @param a Primer operando de la resta.
     * @param b Segundo operando de la resta.
     * @return Resta los dos números pasados como argumentos.
     */
    public double restar(double a, double b) {
        return a - b;
    }

}
```

```java

    /**
     * Producto de dos números.
     *
     * @param a Primer operando del producto.
     * @param b Segundo operando del producto.
     * @return Producto de los dos números pasados como argumentos.
     */
    public double multiplicar(double a, double b) {
        return a * b;
    }

    /**
     * Divide dos números.
     *
     * @param a Dividendo.
     * @param b Divisor.
     * @return División de los dos números pasados como argumentos.
     * @throws ArithmeticException si intenta dividir entre cero
     */
    public double dividir(double a, double b) throws ArithmeticException {
        if (b != 0) {
            return a / b;
        } else {
            throw new ArithmeticException( s: "Intento de división entre cero.");
        }
    }

    /**
     * Elevan un número a una potencia dada.
     *
     * @param base Base a elevar.
     * @param exponente Exponente de la potencia.
     * @return Potencia de un número.
     */
    public double potencia(double base, double exponente) {
        return Math.pow( a: base,  b: exponente);
    }

    /**
     * Calcula el valor absoluto de un número.
     *
     * @param valor Número a calcular su absoluto.
     * @return Valor absoluto.
     */
    public double valorAbsoluto(double valor) {
        return Math.abs( a: valor);
    }
}

```



PACKAGE CLASS USE TREE INDEX HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD SEARCH Search

Package com.mycompany.garciacutilasfranciscojose_act01ut05_javadoc_ejercicio1

Class Calculadora

java.lang.Object
com.mycompany.garciacutilasfranciscojose_act01ut05_javadoc_ejercicio1.Calculadora

```
public class Calculadora
extends Object
```

Representa una calculadora y varias de sus operaciones aritméticas elementales.

Version:
23/03/2021

Author:
Carlos García

Constructor Summary

Constructors

Constructor	Description
`Calculadora()`	Crea una instancia de la clase Calculadora.

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method	Description
double	`dividir(double a, double b)`	Divide dos números.
double	`multiplicar(double a, double b)`	Producto de dos números.
double	`potencia(double base, double exponente)`	Elevan un número a una potencia dada.
double	`restar(double a, double b)`	Resta dos números.
double	`sumar(double a, double b)`	Suma dos números.
double	`valorAbsoluto(double valor)`	Calcula el valor absoluto de un número.

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Details

Calculadora

```
public Calculadora()
```

Crea una instancia de la clase Calculadora.

Method Details

sumar

```
public double sumar(double a,
                    double b)
```

Suma dos números.

Parameters:

`a` - Primer operando de la suma.

`b` - Segundo operando de la suma.

Returns:

Suma de los dos números pasados como argumentos.

restar

```
public double restar(double a,
                     double b)
```

Resta dos números.

Parameters:

`a` - Primer operando de la resta.

`b` - Segundo operando de la resta.

Returns:

Resta los dos números pasados como argumentos.

multiplicar

```
public double multiplicar(double a,
                          double b)
```

Producto de dos números.

Parameters:

`a` - Primer operando del producto.

`b` - Segundo operando del producto.

Returns:

Producto de los dos números pasados como argumentos.

dividir

```
public double dividir(double a,
                      double b)
               throws ArithmeticException
```

Divide dos números.

Parameters:

`a` - Dividendo.

`b` - Divisor.

Returns:

División de los dos números pasados como argumentos.

Throws:

`ArithmeticException` - si intenta dividir entre cero

potencia

```
public double potencia(double base,
                       double exponente)
```

Elevan un número a una potencia dada.

Parameters:

`base` - Base a elevar.

`exponente` - Exponente de la potencia.

Returns:

Potencia de un número.

valorAbsoluto

```
public double valorAbsoluto(double valor)
```

Calcula el valor absoluto de un número.

Parameters:

`valor` - Número a calcular su absoluto.

Returns:

Valor absoluto.

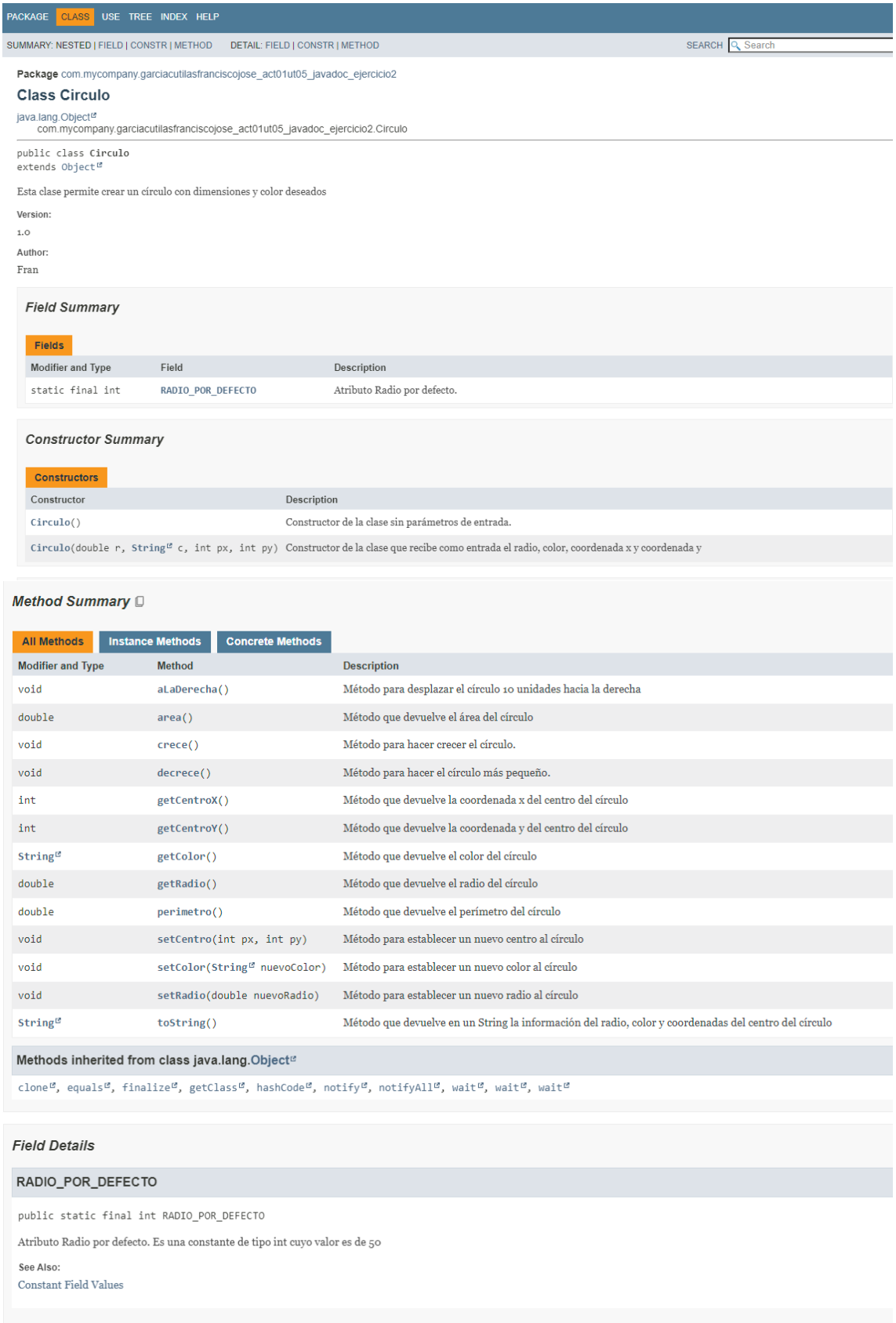
Ejercicio 2

Dado el siguiente código JAVA, utilizalo en un proyecto en Netbeans, comenta en formato JavaDocs la clase, propiedades de la clase y métodos, y genera la documentación automáticamente. Haz varias capturas de pantalla de la documentación generada de manera automática. Mete en el documento tanto el código comentado como las capturas de pantalla de la documentación generada.

```java
package com.mycompany.garciacutilasfranciscojose_act01ut05_javadoc_ejercicio2;

/**
 * Esta clase permite crear un círculo con dimensiones y color deseados
 *
 * @author Fran
 * @version 1.0
 */
public class Circulo {

    private double radio;
    private String color;
    private int centroX, centroY;

    /**
     * Atributo Radio por defecto. Es una constante de tipo int cuyo valor es de 50
     */
    public final static int RADIO_POR_DEFECTO = 50;

    /**
     * Constructor de la clase sin parámetros de entrada.
     * Crea un círculo con el radio por defecto, color negro y centro en coordenada x=100 e y=100
     */
    public Circulo() {
        this.radio = RADIO_POR_DEFECTO;
        color = "negro";
        centroX = 100;
        centroY = 100;
    }

    /**
     * Constructor de la clase que recibe como entrada el radio, color, coordenada x y coordenada y
     *
     * @param r Radio del círculo
     * @param c Color del círculo
     * @param px Coordenada x del centro del círculo
     * @param py Coordenada y del centro del círculo
     */
    public Circulo(double r, String c, int px, int py) {
        radio = r;
        color = c;
        centroX = px;
        centroY = py;
    }
```

```java
    /**
     * Método que devuelve el radio del círculo
     * @return Radio del círculo
     */
    public double getRadio() {
        return radio;
    }

    /**
     * Método que devuelve el color del círculo
     * @return Color del círculo
     */
    public String getColor() {
        return color;
    }

    /**
     * Método que devuelve la coordenada x del centro del círculo
     * @return Coordenada x del centro del círculo
     */
    public int getCentroX() {
        return centroX;
    }

    /**
     * Método que devuelve la coordenada y del centro del círculo
     * @return Coordenada y del centro del círculo
     */
    public int getCentroY() {
        return centroY;
    }

    /**
     * Método para establecer un nuevo radio al círculo
     * @param nuevoRadio Nuevo radio de tipo double
     */
    public void setRadio(double nuevoRadio) {
        this.radio = nuevoRadio;
    }

    /**
     * Método para establecer un nuevo color al círculo
     * @param nuevoColor Nuevo color de tipo String
     */
    public void setColor(String nuevoColor) {
        color = nuevoColor;
    }

    /**
     * Método para establecer un nuevo centro al círculo
     * @param px Coordenada x de tipo int
     * @param py Coordenada y de tipo int
     */
    public void setCentro(int px, int py) {
        centroX = px;
        centroY = py;
    }

    /**
     * Método para desplazar el círculo 10 unidades hacia la derecha
     */
    public void aLaDerecha() {
        centroX += 10;
    }

    /**
     * Método para hacer crecer el círculo. Con él se consigue que el círculo crezca
     * su radio actual por 1.3
     */
    public void crece() {
        radio = radio * 1.3;
    }

    /**
     * Método para hacer el círculo más pequeño. Con él se consigue que el círculo
     * decrezca su radio entre 1.3
     */
    public void decrece() {
        radio = radio / 1.3;
    }
```

```
126
127     /**
128      * Método que devuelve el área del círculo
129      * @return área del círculo
130      */
131     public double area() {
132         return Math.PI * radio * radio;
133     }
134
135     /**
136      * Método que devuelve el perímetro del círculo
137      * @return perímetro del círculo
138      */
139     public double perimetro() {
140         return 2 * Math.PI * radio;
141     }
142
143     /**
144      * Método que devuelve en un String la información del radio, color y coordenadas del
145      * centro del círculo
146      * @return Información del círculo en forma de String
147      */
149     public String toString() {
150         String res = "Circulo de radio " + radio;
151         res += ", color " + color + " y centro (" + centroX + "," + centroY + ")";
152         return res;
153     }
154 }
```



PACKAGE CLASS USE TREE INDEX HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD SEARCH Search

Package com.mycompany.garciacutilasfranciscojose_act01ut05_javadoc_ejercicio2

Class Circulo

java.lang.Object
com.mycompany.garciacutilasfranciscojose_act01ut05_javadoc_ejercicio2.Circulo

```
public class Circulo
extends Object
```

Esta clase permite crear un círculo con dimensiones y color deseados

Version:

1.0

Author:

Fran

Field Summary

Fields

Modifier and Type	Field	Description
static final int	RADIO_POR_DEFECTO	Atributo Radio por defecto.

Constructor Summary

Constructors

Constructor	Description
Circulo()	Constructor de la clase sin parámetros de entrada.
Circulo(double r, String c, int px, int py)	Constructor de la clase que recibe como entrada el radio, color, coordenada x y coordenada y

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method	Description
void	aLaDerecha()	Método para desplazar el círculo 10 unidades hacia la derecha
double	area()	Método que devuelve el área del círculo
void	crece()	Método para hacer crecer el círculo.
void	decrece()	Método para hacer el círculo más pequeño.
int	getCentroX()	Método que devuelve la coordenada x del centro del círculo
int	getCentroY()	Método que devuelve la coordenada y del centro del círculo
String	getColor()	Método que devuelve el color del círculo
double	getRadio()	Método que devuelve el radio del círculo
double	perimetro()	Método que devuelve el perímetro del círculo
void	setCentro(int px, int py)	Método para establecer un nuevo centro al círculo
void	setColor(String nuevoColor)	Método para establecer un nuevo color al círculo
void	setRadio(double nuevoRadio)	Método para establecer un nuevo radio al círculo
String	toString()	Método que devuelve en un String la información del radio, color y coordenadas del centro del círculo

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Field Details

RADIO_POR_DEFECTO

```
public static final int RADIO_POR_DEFECTO
```

Atributo Radio por defecto. Es una constante de tipo int cuyo valor es de 50

See Also:

Constant Field Values

Constructor Details

Circulo

```
public Circulo()
```

Constructor de la clase sin parámetros de entrada. Crea un círculo con el radio por defecto, color negro y centro en coordenada x=100 e y=100

Circulo

```
public Circulo(double r,
               String c,
               int px,
               int py)
```

Constructor de la clase que recibe como entrada el radio, color, coordenada x y coordenada y

Parameters:

`r` - Radio del círculo

`c` - Color del círculo

`px` - Coordenada x del centro del círculo

`py` - Coordenada y del centro del círculo

Method Details

getRadio

```
public double getRadio()
```

Método que devuelve el radio del círculo

Returns:

Radio del círculo

getColor

```
public String getColor()
```

Método que devuelve el color del círculo

Returns:

Color del círculo

getCentroX

```
public int getCentroX()
```

Método que devuelve la coordenada x del centro del círculo

Returns:

Coordenada x del centro del círculo

getCentroY

```
public int getCentroY()
```

Método que devuelve la coordenada y del centro del círculo

Returns:
Coordenada y del centro del círculo

setRadio

```
public void setRadio(double nuevoRadio)
```

Método para establecer un nuevo radio al círculo

Parameters:
`nuevoRadio` - Nuevo radio de tipo double

setColor

```
public void setColor(String nuevoColor)
```

Método para establecer un nuevo color al círculo

Parameters:
`nuevoColor` - Nuevo color de tipo String

setCentro

```
public void setCentro(int px,
                      int py)
```

Método para establecer un nuevo centro al círculo

Parameters:
`px` - Coordenada x de tipo int
`py` - Coordenada y de tipo int

aLaDerecha

```
public void aLaDerecha()
```

Método para desplazar el círculo 10 unidades hacia la derecha

crece

```
public void crece()
```

Método para hacer crecer el círculo. Con él se consigue que el círculo crezca su radio actual por 1.3

decrece

```
public void decrece()
```

Método para hacer el círculo más pequeño. Con él se consigue que el círculo decrezca su radio entre 1.3

area

```
public double area()
```

Método que devuelve el área del círculo

Returns:
área del círculo

perimetro

```
public double perimetro()
```

Método que devuelve el perímetro del círculo

Returns:
perímetro del círculo

toString

```
public String toString()
```

Método que devuelve en un String la información del radio, color y coordenadas del centro del círculo

Overrides:
`toString` in class `Object`

Returns:
Información del círculo en forma de String