



APRENDERAPROGRAMAR.COM

DOCUMENTAR PROYECTOS
JAVA CON JAVADOC.
COMENTARIOS, SÍMBOLOS,
TAGS (DEPRECATED,
PARAM, RETURN, ETC.)
(CU00680B)

Sección: Cursos

Categoría: Curso “Aprender programación Java desde cero”

Fecha revisión: 2029

Resumen: Entrega nº80 curso Aprender programación Java desde cero.

Autor: Alex Rodríguez

DOCUMENTAR PROYECTOS JAVA CON JAVADOC

Documentar un proyecto es algo fundamental de cara a su futuro mantenimiento. Cuando programamos una clase, debemos generar documentación lo suficientemente detallada sobre ella como para que otros programadores sean capaces de usarla sólo con su interfaz. No debe existir necesidad de leer o estudiar su implementación, lo mismo que nosotros para usar una clase del API Java no leemos ni estudiamos su código fuente.



Javadoc es una utilidad de Oracle para la generación de documentación de APIs en formato HTML a partir de código fuente Java. **Javadoc es el estándar para documentar clases de Java.** La mayoría de los IDEs utilizan javadoc para generar de forma automática documentación de clases. BlueJ también utiliza javadoc y permite la generación automática de documentación, y visionarla bien de forma completa para todas las clases de un proyecto, o bien de forma particular para una de las clases de un proyecto.

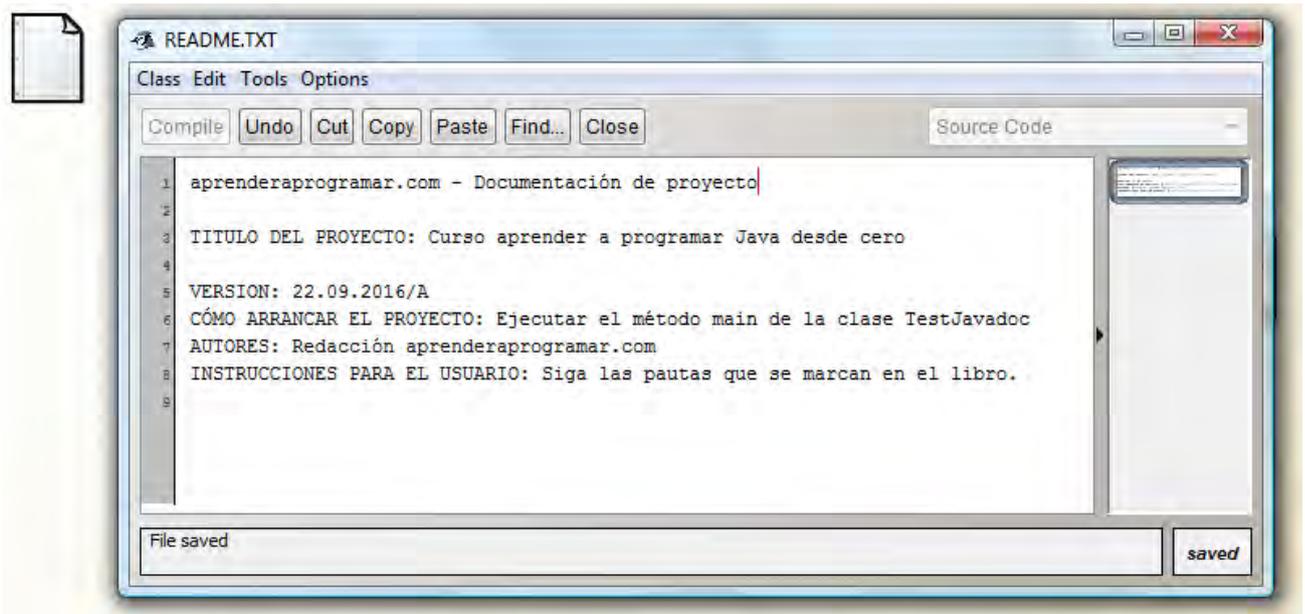
Veamos en primer lugar qué se debe incluir al documentar una clase:

- a) **Nombre de la clase, descripción general, número de versión, nombre de autores.**
- b) **Documentación de cada constructor o método** (especialmente los públicos) incluyendo: nombre del constructor o método, tipo de retorno, nombres y tipos de parámetros si los hay, descripción general, descripción de parámetros (si los hay), descripción del valor que devuelve.

Las variables de instancia o de clase no se suelen documentar a nivel de javadoc.

Para generar la documentación de un proyecto automáticamente hemos de seguir unas normas a la hora de realizar los comentarios dentro del mismo. Si las hemos seguido, en BlueJ disponemos de la opción del menú Tools --> Project Documentation, que nos abre la documentación del proyecto en un navegador web. Para ver la documentación de una clase específica en BlueJ, debemos abrir la ventana de código, y en la parte superior derecha cambiar la pestaña que pone "Source code" por la opción "Documentation".

Además de documentar las clases, **todo proyecto debería tener un archivo Leeme o Readme.** En BlueJ podemos acceder al readme.txt de proyecto haciendo doble click en el icono que representa una hoja en la parte superior izquierda del diagrama de clases. En el readme.txt sería adecuado incluir al menos: título del proyecto, descripción, versión, cómo arrancar el proyecto, autores e instrucciones para los usuarios. Haz doble click en el icono e introduce una descripción para tu proyecto.



Una vez cierres la ventana, la información incluida se guarda automáticamente. Para que javadoc sea capaz de generar documentación automáticamente han de seguirse estas reglas:

- a) La documentación para javadoc ha de incluirse entre símbolos de comentario que han de empezar con una barra y doble asterisco, y terminar con un asterisco y barra simple.

```
/**
 * Esto es un comentario para javadoc ejemplo aprenderaprogramar.com
 */
```

- b) La ubicación le define a javadoc qué representa el comentario: si está incluido justo antes de la declaración de clase se considerará un comentario de clase, y si está incluido justo antes de la signatura de un constructor o método se considerará un comentario de ese constructor o método.
- c) Para alimentar javadoc se usan ciertas palabras reservadas (tags) precedidas por el carácter "@", dentro de los símbolos de comentario javadoc. Si no existe al menos una línea que comience con @ no se reconocerá el comentario para la documentación de la clase.

En la tabla siguiente mostramos algunas de las palabras reservadas (tags), aunque hay más de las que aquí incluimos. Si necesitas una lista completa de las tags con su correspondiente uso realiza una búsqueda de la cadena "javadoc tags" en bing, google o yahoo.

TAG	DESCRIPCIÓN	COMPRENDE
@author	Nombre del desarrollador.	Nombre autor o autores
@deprecated	Indica que el método o clase es obsoleto (propio de versiones anteriores) y que no se recomienda su uso.	Descripción
@param	Definición de un parámetro de un método, es requerido para todos los parámetros del método.	Nombre de parámetro y descripción
@return	Informa de lo que devuelve el método, no se aplica en constructores o métodos "void".	Descripción del valor de retorno
@see	Asocia con otro método o clase.	Referencia cruzada referencia (#método(); clase#método()); paquete.clase; paquete.clase#método()).
@version	Versión del método o clase.	Versión

Las etiquetas `@author` y `@version` se usan para documentar clases e interfaces. Por tanto no son válidas en cabecera de constructores ni métodos. La etiqueta `@param` se usa para documentar constructores y métodos. La etiqueta `@return` se usa solo en métodos de tipo función.

Dentro de los comentarios se admiten etiquetas HTML, por ejemplo con `@see` se puede referenciar una página web como link para recomendar su visita de cara a ampliar información.

A continuación escribe la documentación javadoc para una clase que hayas desarrollado previamente, tratando de utilizar los tags que hemos visto. Por ejemplo:

```
import java.util.ArrayList;
import java.util.Random;

/**
 * Esta clase define objetos que contienen tantos enteros aleatorios entre 0 y 1000 como se le definen al crear un objeto
 * @author: Mario R. Rancel
 * @version: 22/09/2016/A
 * @see <a href = "http://www.aprenderaprogramar.com" /> aprenderaprogramar.com – Didáctica en programación </a>
 */

public class SerieDeAleatoriosD {

    //Campos de la clase
    private ArrayList<Integer> serieAleatoria;
    /**
     * Constructor para la serie de números aleatorios
     * @param numeroltems El parámetro numeroltems define el número de elementos que va a tener la serie aleatoria
     */
    public SerieDeAleatoriosD (int numeroltems) {
        serieAleatoria = new ArrayList<Integer> ();
        for (int i=0; i<numeroltems; i++) { serieAleatoria.add(0); }
        System.out.println ("Serie inicializada. El número de elementos en la serie es: " + getNumeroltems() );
    } //Cierre del constructor
}
```

```

/**
 * Método que devuelve el número de ítems (números aleatorios) existentes en la serie
 * @return El número de ítems (números aleatorios) de que consta la serie
 */
public int getNumeroItems() { return serieAleatoria.size(); }
/**
 * Método que genera la serie de números aleatorios
 */
public void generarSerieDeAleatorios () {
    Random numAleatorio;
    numAleatorio = new Random ();
    for (int i=0; i < serieAleatoria.size(); i++) { serieAleatoria.set(i, numAleatorio.nextInt(1000) ); }
    System.out.print ("Serie generada! ");
} //Cierre del método
} //Cierre de la clase y del ejemplo aprenderaprogramar.com
    
```

Ahora, desde la ventana del editor de código de BlueJ (parte superior derecha) elige la opción “Documentation” en vez de “Source Code”. Se te mostrará una vista similar a esta:

Class SerieDeAleatoriosD

[java.lang.Object](#)

└ **SerieDeAleatoriosD**

public class **SerieDeAleatoriosD** extends java.lang.Object

Esta clase define objetos que contienen tantos enteros aleatorios entre 0 y 1000 como se le definen al crear un objeto de esta clase

See Also: [aprenderaprogramar.com – Didáctica en programación](#)

Constructor Summary	
	<p>SerieDeAleatoriosD(int numeroItems) Constructor para la serie de números aleatorios</p>

Method Summary	
void	<p>generarSerieDeAleatorios() Método que genera la serie de números aleatorios</p>
int	<p>getNumeroItems() Método que devuelve el número de ítems (números aleatorios) existentes en la serie</p>

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

SerieDeAleatoriosD

```
public SerieDeAleatoriosD(int numeroItems)
    Constructor para la serie de números aleatorios
```

Parameters:

`numeroItems` - El parámetro `numeroItems` define el número de elementos que va a tener la serie aleatoria

Method Detail

generarSerieDeAleatorios

```
public void generarSerieDeAleatorios()
    Método que genera la serie de números aleatorios
```

getNumeroItems

```
public int getNumeroItems()
    Método que devuelve el número de ítems (números aleatorios) existentes en la serie
```

Returns:

El número de ítems (números aleatorios) de que consta la serie

Aunque parte de la información que nos aparece todavía no somos capaces de interpretarla, hemos visto cómo los comentarios introducidos en el código siguiendo unas reglas son transformados en documentación por la herramienta javadoc. La documentación del API de Java está generada de esta manera, a gran escala y con un número de tags mayor que el que hemos empleado nosotros. El API de Java nos resulta útil porque está bien documentado. De cara al mantenimiento, ampliación y conexión de nuestros programas **es fundamental que documentemos el código correctamente** y de forma amplia. La documentación javadoc es imprescindible en un proyecto profesional Java. Esta documentación es ideal que se complemente con tantos comentarios libres adicionales como sean necesarios para interpretar bien el código. Los comentarios que no se crean siguiendo las normas de javadoc no aparecerán en la documentación, pero serán útiles para otros programadores o para nosotros mismos cuando tengamos que consultar el código pasado un tiempo después de haberlo generado.

En este curso no mantenemos la estandarización javadoc por motivos de espacio, pero debes tenerla presente y aplicarla para el desarrollo de cualquier proyecto de índole profesional o académica.

EJERCICIO

Crea una clase denominada `miniCalculadoraEjemplo` que tenga dos métodos (basados en el uso de métodos de la clase `Math`): un método `valorAbsoluto` que recibe un número de tipo `double` y devuelva su valor absoluto, y otro método `raizCuadrada` que reciba un número de tipo `double` y devuelva su raíz cuadrada. Documenta esta clase conforme a los estándares `JavaDoc` y comprueba cómo se visualiza la documentación. Para comprobar si tu solución es correcta puedes consultar en los foros aprenderaprogramar.com.

Próxima entrega: CU00681B

Acceso al curso completo en aprenderaprogramar.com -- > Cursos, o en la dirección siguiente:

http://www.aprenderaprogramar.com/index.php?option=com_content&view=category&id=68&Itemid=188