

USER

creame un documento con una guia de los terminados con la documentacion de Javadoc, para que sea sencillo de aprender

ASSISTANT

Guía de Aprendizaje de Javadoc: Terminología y Documentación

Bienvenido a esta guía introductoria sobre **Javadoc**. Aquí aprenderás cómo utilizar los términos y etiquetas más comunes para documentar tu código Java de manera efectiva. Esta guía está diseñada para ser sencilla y práctica, ideal tanto para principiantes como para desarrolladores intermedios.

¿Qué es Javadoc?

Javadoc es una herramienta incluida en el JDK (Java Development Kit) que permite generar documentación automática en formato HTML a partir de comentarios especiales colocados en el código fuente. Estos comentarios no solo mejoran la legibilidad del código, sino que también facilitan su mantenimiento y colaboración en equipos.

Estructura Básica de un Comentario Javadoc

Un comentario Javadoc siempre comienza con `/**` y termina con `*/`. Dentro de este bloque, puedes incluir descripciones y etiquetas específicas para estructurar la información.

Ejemplo básico:

```
/**
 * Esta clase representa un ejemplo básico de documentación con
 Javadoc.
 */
public class Ejemplo {
    // Código de la clase...
}
```

Etiquetas Comunes de Javadoc

A continuación, se presentan las etiquetas más utilizadas en Javadoc, junto con ejemplos prácticos.

1. @author

Se utiliza para indicar el autor de la clase o método.

```
/**
 * Clase principal del programa.
 *
 * @author Nombre del Autor
 */
public class MiClase {
    // Código de la clase...
}
```

2. @version

Indica la versión actual del archivo o componente.

```
/**
 * Clase principal del programa.
 *
 * @author Nombre del Autor
 * @version 1.0
 */
public class MiClase {
    // Código de la clase...
}
```

3. @param

Describe los parámetros de un método.

```
/**
 * Suma dos números enteros.
 *
 * @param a Primer número entero.
 * @param b Segundo número entero.
 * @return La suma de los dos números.
 */
public int sumar(int a, int b) {
    return a + b;
}
```

4. @return

Explica el valor que retorna un método.

```
/**
 * Obtiene el nombre del usuario.
 *
 * @return El nombre del usuario como cadena de texto.
 */
public String getNombre() {
    return "Usuario";
}
```

5. @throws o @exception

Indica las excepciones que puede lanzar un método.

```
/**
 * Divide dos números.
```

```

*
* @param numerador El número que será dividido.
* @param denominador El número por el cual se divide.
* @return El resultado de la división.
* @throws ArithmeticException Si el denominador es cero.
*/
public double dividir(int numerador, int denominador) throws
ArithmeticException {
    if (denominador == 0) {
        throw new ArithmeticException("División por cero no
permitida.");
    }
    return (double) numerador / denominador;
}

```

6. @see

Referencia otras clases, métodos o recursos relacionados.

```

/**
* Clase que maneja operaciones matemáticas básicas.
*
* @see MathUtil
*/
public class Calculadora {
    // Código de la clase...
}

```

7. @since

Indica desde qué versión del software o biblioteca se introdujo un elemento.

```

/**
* Método para calcular el cuadrado de un número.
*
* @param numero El número a elevar al cuadrado.
* @return El cuadrado del número.
* @since 1.1

```

```
*/  
public int calcularCuadrado(int numero) {  
    return numero * numero;  
}
```

8. @deprecated

Marca un método o clase como obsoleto, sugiriendo no usarlo en versiones futuras.

```
/**  
 * Este método ha sido reemplazado por {@link  
 #calcularCuadrado(int)}.  
 *  
 * @param numero El número a elevar al cuadrado.  
 * @return El cuadrado del número.  
 * @deprecated Usar {@link #calcularCuadrado(int)} en lugar de este  
 método.  
 */  
@Deprecated  
public int viejoCalcularCuadrado(int numero) {  
    return numero * numero;  
}
```

9. @serial

Se usa para documentar campos de serialización en clases que implementan Serializable.

```
/**  
 * Clase que implementa Serializable.  
 *  
 * @serial Esta clase contiene datos importantes que deben ser  
 serializados.
```

```

*/
public class Datos implements java.io.Serializable {
    private static final long serialVersionUID = 1L;

    /**
     * Campo de datos.
     *
     * @serial El campo de datos debe ser serializado.
     */
    private String datos;

    // Getters y setters...
}

```

Ejemplo Completo de Documentación con Javadoc

Aquí tienes un ejemplo completo que combina varias etiquetas:

```

/**
 * Clase que representa una calculadora básica.
 *
 * @author Juan Pérez
 * @version 1.0
 * @since 1.0
 */

```

```

public class Calculadora {

    /**
     * Suma dos números enteros.
     *
     * @param a Primer número entero.
     * @param b Segundo número entero.
     * @return La suma de los dos números.
     * @throws IllegalArgumentException Si alguno de los números es
negativo.
     * @see #restar(int, int)
     */
    public int sumar(int a, int b) throws IllegalArgumentException {
        if (a < 0 || b < 0) {
            throw new IllegalArgumentException("Los números deben
ser positivos.");
        }
        return a + b;
    }

    /**
     * Resta dos números enteros.
     *
     * @param a Minuendo.
     * @param b Sustraendo.
     * @return La diferencia entre los dos números.
     */
    public int restar(int a, int b) {
        return a - b;
    }
}

```

Generando Documentación con Javadoc

Para generar la documentación HTML a partir de tus comentarios Javadoc, sigue estos pasos:

1. Abre una terminal o línea de comandos.
2. Navega hasta el directorio donde se encuentra tu archivo `.java`.
3. Ejecuta el siguiente comando:

```
javadoc -d docs MiClase.java
```

Esto generará la documentación en el directorio docs.

4. Abre el archivo `index.html` dentro del directorio docs para ver la documentación en tu navegador.

Conclusión

Documentar tu código con Javadoc no solo mejora su calidad, sino que también facilita la comprensión y mantenimiento del mismo. Las etiquetas vistas en esta guía te proporcionan las herramientas necesarias para crear documentación profesional y completa.

Si tienes alguna duda o necesitas más ejemplos, no dudes en explorar la [documentación oficial de Javadoc](#).

¡Feliz codificación!