Creación y uso de Javadoc con Netbeans

Javadoc es la utilidad de Java para extraer y generar documentación directamente del código en formato HTML.

Para que la documentación sea en verdad útil debemos escribir los comentarios del código de acuerdo a las recomendaciones de Javadoc que indica:

- Los comentarios de documentación de Javadoc se colocan entre los delimitadores
  /**
  */
- Dentro de los delimitadores se agrupan varias líneas, y cada línea precedida por *
- Los comentarios de Javadoc están formados por DESCRIPCIÓN + ETIQUETAS (TAGS)
- Los comentarios DEBEN colocarse ANTES de la declaración de una clase, campo, método o constructor
- Dentro de los delimitadores se podrá escribir etiquetas HTML

Su uso es recomendable para todo proyecto, incluido los proyectos pequeños porque nos ayuda a tener siempre un código bien documentado. Es una buena regla de programación.

TAGS

@author - El nombre del autor del proyecto por ejemplo pepito :)
@version - La versión del proyecto
@see - Añade una referencia a una clase, método o enlace web
@param - Nombre de parámetro utilizado en un método incluido su significado
@return - El resultado de un método incluido su descripción
@exception - Nombre de la excepción más una descripción
@throws - Nombre de la excepción más una descripción
@deprecated - Añade una alerta al usuario de que el método que sigue a continuación ya no debe usarse y que será eliminado en versiones posteriores.
@since - Indica el nº de versión desde la que existe el método.

Al escribir los comentarios también podemos hacer uso de algunas etiquetas HTML para decorar un poco más nuestra documentación.

Para indicarle a Javadoc que queremos incluir documentación, debemos comenzar los comentarios de la siguiente manera:

```
/**
* Esto para Javadoc
*/
```

Veamos su uso con un ejemplo.

Creamos una clase llamada **"Clase_java"** y en el encabezado colocamos el autor de la clase, la instrucción @see con un enlace a una página web, también podemos agregar la versión de la clase:

```
/**
 * @author Mouse
 * @see <a href="http://www.jc-mouse.net">http://www.jc-mouse.net</a>
 * @version 1.2 07 de Mayo de 2013
 */
public class Clase_Java {
...
}
```

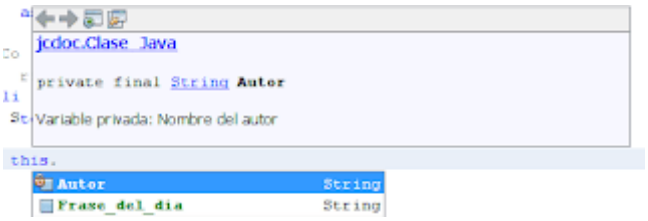
También podemos documentar las variables que hacen parte de nuestro código

```
/**
 * Variable privada: Nombre del autor
```

```
 */
private final String Autor = "jc Mouse";

/**
 * Variable publica: Una frase para reflexionar
 */
public String Frase_del_dia = "Carpe diem";
```

Cuando se haga uso de estas variables comentadas, en el editor de Netbeans podremos ver algo como esto:



No debemos olvidar tambien comentar el constructor de clase

```
/**
 * Constructor de clase
 */
public Clase_Java(){
...
}
```

Cuando creamos un método es recomendable documentar los parámetros de entrada, si es que los tuviera, y si este método retorna algún resultado, utilizar la marca **@return**, si por el contrario el método es de la forma **VOID**, no se usa nada. Así también para las excepciones que puedan ocurrir se usa **@exception**. En la descripción del método, se puede incluir un ejemplo de uso encerrado en las etiquetas PRE, por ejemplo:

```
    /**
     * Método que convierte una cadena de la forma "yyyy-MM-dd HH:mm"
a Date
     * <PRE> Clase_Java cj = new Clase_Java();
     * Date date = cj.StringToDate("2012-10-01 12:00")</PRE>
     * @param value String que debe tener la forma "yyyy-MM-dd HH:mm"
     * @return Date Un objeto Date con la fecha parseada
     * @exception ParseException Error de parseo, ocurre cuando no se
puede convertir un String a Date
     * @since incluido desde la version 1.0
     */
    public Date StringToDate( String value )
    {
        Date date = new Date();
        SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd
HH:mm");
         try {
            date = (java.util.Date) formatter.parse( value );
        } catch (ParseException ex) {
            System.err.println( ex.getMessage() );
        }
        return date;
    }
```

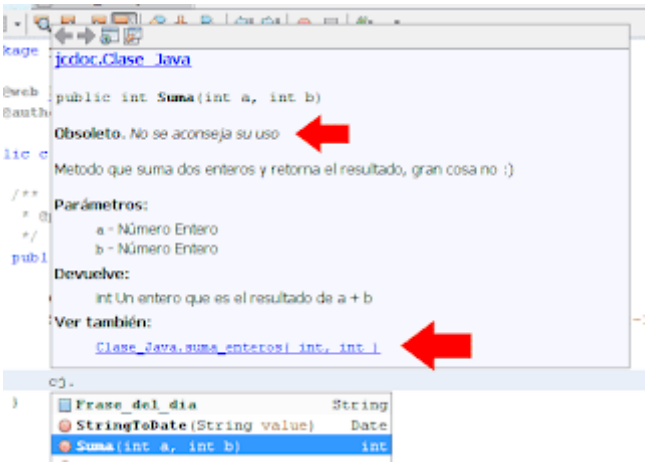
Cuando utilicemos este método, podremos ver que Netbeans nos despliega toda la información en cuanto se hace referencia al nombre StringToDate.

Cuando tenemos varias versiones de un proyecto y vamos actualizando los métodos, antes de eliminar los métodos que ya no se harán uso, es preferible, primero alertar al usuario que ciertas funciones dejaran de existir en versiones posteriores, para esto, en la descripción se añade la marca **@deprecated**, y la marca **@see**, que hace referencia a la función de reemplazo, es decir, supongamos que en una primera versión tenemos un método llamado **Suma** pero que en una nueva versión se decide crear una nueva versión **suma_enteros**, entonces lo que debemos hacer es añadir en la documentación de Suma, la alerta de que esta función ya no se utilizara más y que es mejor usar la nueva función suma_enteros, es decir:

```
/**
     * Metodo que suma dos enteros y retorna el resultado, gran cosa
no :)
     * @param a Número Entero
     * @param b Número Entero
     * @return int Un entero que es el resultado de a + b
     * @deprecated No se aconseja su uso
     * @see suma_enteros( int, int )
     */
    public int Suma( int a , int b)
    {
        return a + b;
    }

    /**
     * Metodo que suma dos enteros positivos y retorna el resultado
     * @param a Número Entero
     * @param b Número Entero
     * @return int Un entero que es el resultado de a + b, si los
numeros son negativos, retorna cero.
     */
    public int suma_enteros( int a, int b)
    {
        int resultado = 0;
        if( a>0 && b>0)
            resultado = a + b;
        return resultado;
    }
```

Ahora, cuando se haga uso del método Suma, este nos aparecerá con un subrayado y con la alerta correspondiente:



El código completo de la clase es:

```
package jcdoc;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
/**
 * @author Mouse
 * @see <a href="http://www.jc-mouse.net">http://www.jc-mouse.net</a>
 * @version 1.2 07 de Mayo de 2013
 */
public class Clase_Java {

    /**
     * Variable privada: Nombre del autor
     */
    private final String Autor = "jc Mouse";
    /**
     * Variable publica: Una frase para reflexionar
     */
    public String Frase_del_dia = "Carpe diem";

    /**
     * Constructor de clase
     */
    public Clase_Java(){
        System.out.println( this.Autor + " te aconseja '" +
this.Frase_del_dia + "'" );
    }

    /**
     * Método que convierte una cadena de la forma "yyyy-MM-dd HH:mm"
a Date
     * <PRE> Clase_Java cj = new Clase_Java();
     * Date date = cj.StringToDate("2012-10-01 12:00")</PRE>
     * @param value String que debe tener la forma "yyyy-MM-dd HH:mm"
     * @return Date Un objeto Date con la fecha parseada
     * @exception ParseException Error de parseo, ocurre cuando no se
puede convertir un String a Date
     * @since incluido desde la version 1.0
     */
    public Date StringToDate( String value )
    {
        Date date = new Date();
        SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd
HH:mm");
         try {
            date = (java.util.Date) formatter.parse( value );
        } catch (ParseException ex) {
            System.err.println( ex.getMessage() );
        }
        return date;
    }
```

```
    /**
     * Metodo que suma dos enteros y retorna el resultado, gran cosa
no :)
     * @param a Número Entero
     * @param b Número Entero
     * @return int Un entero que es el resultado de a + b
     * @deprecated No se aconseja su uso
     * @see suma_enteros( int, int )
     */
    public int Suma( int a , int b)
    {
        return a + b;
    }

    /**
     * Metodo que suma dos enteros positivos y retorna el resultado
     * @param a Número Entero
     * @param b Número Entero
     * @return int Un entero que es el resultado de a + b, si los
numeros son negativos, retorna cero.
     */
    public int suma_enteros( int a, int b)
    {
        int resultado = 0;
        if( a>0 && b>0)
            resultado = a + b;
        return resultado;
    }

}
```

Para generar los HTML, vamos a Ejecutar -> Generar JavaDoc. y esperamos unos segundos a que nos genere todos los archivos.