¿Cómo documentar en Java?

¿Por qué documentar?

Documentamos para no complicarle la vida a otros programadores que quieran entender lo que se hizo en un código (saber para qué sirve cada clase y cada método).

Generar Documentación

Podemos generar la documentación de una clase desde nuestro editor o desde la línea de comandos e independientemente de la opción que escojamos, se usará la herramienta `javadoc` de nuestro JDK (Java Development Kit).

La documentación de java se genera en formato HTML de forma local dentro de nuestro proyecto, es importante saber que para generar documentación de alguna clase o método deben usarse etiquetas HTML precedidas del carácter `@`, éstas etiquetas deben estar dentro de un comentario java que inicia con `/**` y termina con `*/`. (los comentarios con etiquetas de documentación pueden ir al inicio de, una clase, un atributo o un método).

Comando JavaDoc

El comando javadoc posee una gran cantidad de parámetros para generar documentación, el primer paso antes de generar cualquier documentación Java es crear un directorio donde residirá la documentación, posteriormente es invocado el comando javadoc con los siguientes parámetros:

- `-d docs` : Indica el directorio donde será depositada la documentación.
- `-version` : Implica que al ser generada la documentación sea desplegado el número de versión, de ser definido en la Clase.
- `-author` : En caso de ser definido el parámetro autor, éste aparecerá en la documentación.
- `-use`: Es el parámetro utilizado para indicar los directorios donde residen las Clases sobre las que será generada la documentación.

Al terminar este proceso será generada la documentación de las distintas Clases dentro del directorio donde está depositada; para observar otros parámetros que puede tomar el comando `javadoc` basta ejecutar : `javadoc -help`.

Etiquetas en JavaDoc

Las etiquetas que comúnmente se generan en javaDoc son las que se muestran en la siguiente tabla.

Etiqueta	Descripción
`@author`	Indica el nombre del sujeto que desarrollo el componente.
`@deprecated`	Indica que algún método o clase u otro componente está obsoleto (es utilizado para indicar que una Clase ha sido repuesta por otra más reciente)
`@param`	Indica un parámetro de un método, se tiene que usar para todos los parámetros del método.
`@return`	Indica el valor de retorno de un método
`@see`	Indica que el componente puede hacer referencia a otro. (referencia a otras clases)
`@throw`	Indica que excepción lanza el método.
`@version`	Indica la versión actual del componente.

Cabrera Sánchez Ana Dhamar