# Documentación de programas con *doxygen*

## *Documentación interna*

### *Documentación de cada fichero*

Cada uno de los ficheros que formen parte de nuestro proyecto deben estar auto-documentados, en ellos se debe incluir información sobre:

Autor del documento

Descripción del contenido del fichero

Fecha de última modificación

### *Documentación de cada clase, método, módulo, etc.*

A la hora de implementar cada una de las clases, métodos o módulos

Nombre de la clase, método, módulo, etc.

Descripción de las Precondiciones

Descripción de las Postcondiciones

Descripción de parámetros de entrada, salida y entrada salida

Descripción del tipo de dato de retorno

### *Documentación con doxygen*

Doxygen basa la generación de la documentación de programas en la interpretación de los comentarios incluidos en los mismos. Para ello es necesario que estos comentarios sigan unas reglas, y marquen mediante etiquetas el contenido de los mismos.

Los comentarios interpretados por *doxygen* se escriben con doble asterisco al principio.

```
/**
*  Este módulo devuelve el cuadrado de un número pasado por parámetros
*/
int cuadrado(int num);
```

Los comentarios *doxygen* no pueden aparecer en cualquier lugar, sino que deben estar asociados a entidades a documentar: módulos, archivos, variables, funciones, clases, etc.

Para documentar una cierta entidad (módulo, variable, etc.) el comentario debe estar ubicado inmediatamente antes de su definición.

A continuación se revisan algunas de las marcas *doxygen* que nos permitirán formatear los comentarios para que sean entendidos adecuadamente por *doxygen*. Todas las marcas van precedidas del carácter @ (formato Javadoc) o \ (formato Qt):

- \brief ó @brief {descripción breve}
    - Comienza un párrafo que sirve como una descripción breve del cometido del elemento documentado. Se puede omitir en un bloque de documentación si la descripción breve es la primera línea del bloque y ocupa sólo una línea.
- \file ó @file [<nombre-fichero>]
    - Indica que el bloque comentado contiene documentación de un fichero con el nombre nombre-fichero.
- \param ó @param <nombre-parametro> {descripción del parámetro}
    - Comienza la descripción de un parámetro llamado nombre-parámetro que pertenece a un función. **Comprueba la existencia del mismo nombre**.
- \return ó @return {descripción del valor de retorno}
    - Se usa para describir el valor de retorno de una función de forma genérica.
- \retval ó @retval <valor de retorno> {descripción}
    - Se describe un valor de retorno concreto de una función. Por ejemplo:
        - @retval true si el número es primo
        - @retval false si el número no es primo
- \struct ó @struct <nombre>
    - Indica que un bloque de comentario contiene documentación sobre una estructura llamada nombre
- \var ó @var (declaración de variable)
    - Indica que un bloque de comentario contiene documentación sobre una constante o variable.
- \author ó @author{lista de autores}
    - Comienza un párrafo donde se introducen los nombres de uno o más autores.
- \date ó @date {descripción de la fecha}
    - Comienza un párrafo donde se puede introducir una o más fechas. El texto que se introduce no tiene una estructura especial.
- \par ó @par [título] {párrafo}
    - Comienza un párrafo nuevo titulado título. Se considera título hasta el final de la línea. El párrafo tras esta orden se sangra convenientemente. Si se omite el título sólo se comenzará un nuevo párrafo. Funciona dentro de otras órdenes que crean párrafos como \param o \pre
- \pre {descripción de la precondición}
    - Indica que un bloque de comentario contiene documentación sobre una precondición
- \post {descripción de la postcondicion}
    - Indica que un bloque de comentario contiene documentación sobre una postcondición.
- \par ó @par *Complejidad* {párrafo}
    - Comienza un párrafo nuevo titulado Complejidad. Se considera *Complejidad* hasta el final de la línea. El párrafo tras esta orden se sangra convenientemente. Si se omite el título sólo se comenzará un nuevo párrafo. Funciona dentro de otras órdenes que crean párrafos como \param o \pre

## Formato de documentación de módulos

/**

@brief Método que comprueba si dos personas tienen la misma edad x

@param p1: Primera persona x

@param p2: Segunda persona

@return bool indicando que sus edades son iguales x

**Ó**

@retval true si las dos edades son iguales

@retval false si las edades son diferentes

@par descripción:

En este apartado se puede incluir una descripción más detallada.

@pre Las dos personas deben estar creadas y con valores de edad validos

@post Si la edad de la persona 1 es igual a la edad de la persona 2 devuelve true, en caso contrario devuelve false.

@par Complejidad

Aquí indicar la complejidad del módulo

@see Ver la función devolver_edades() de la clase persona.

*/

*bool mismaEdad(Persona p1, Persona p1);*

## Formato de documentación de ficheros

/**

@file Nombre del fichero x

@brief Breve resumen de la unidad (1 línea ) x

@author x

@date x

@par Descripción: x

En este apartado se puede incluir una descripción más detallada.

@par Utilización:

Para su correcto uso se debe llamar primera a la función .....

A continuación generar una instancia y realizar una llamada al método .....

@par Plataforma

En que plataformas funciona

*/