

# Manual de uso de

The logo for Doxygen, featuring the word "Doxygen" in a stylized, bold, blue font with a white outline, set against a dark blue rectangular background.

*Documentación automática de código fuente C#  
mediante Doxygen*

<http://www.doxygen.org/>

**Índice**

1Introducción.....3

2Instalación.....3

3Obtener la documentación de un proyecto.....4

4Documentar un proyecto.....7

# 1 Introducción

**Doxygen** es un documentador automático, capaz de extraer la documentación de los propios fuentes del programa. Es capaz de extraer documentación especialmente de *Java*, *C*, *C++* y *C#*, si bien otros lenguajes como *Python* también están soportados. En este documento se describe de manera sencilla las operaciones más comunes con esta herramienta: el formato de los comentarios en el código fuente (el pequeño precio que hay que pagar para poder disfrutar de estas ventajas), la configuración de *Doxygen* y su ejecución para la obtención de la documentación.

*Doxygen* se basa principalmente en *JavaDoc*, una herramienta que desde finales de los años 90 se distribuye con el lenguaje de programación *Java*, y que funciona exactamente de la misma forma (si bien *Doxygen* cubre muchos más lenguajes y aporta mucha más funcionalidad).

## 2 Instalación

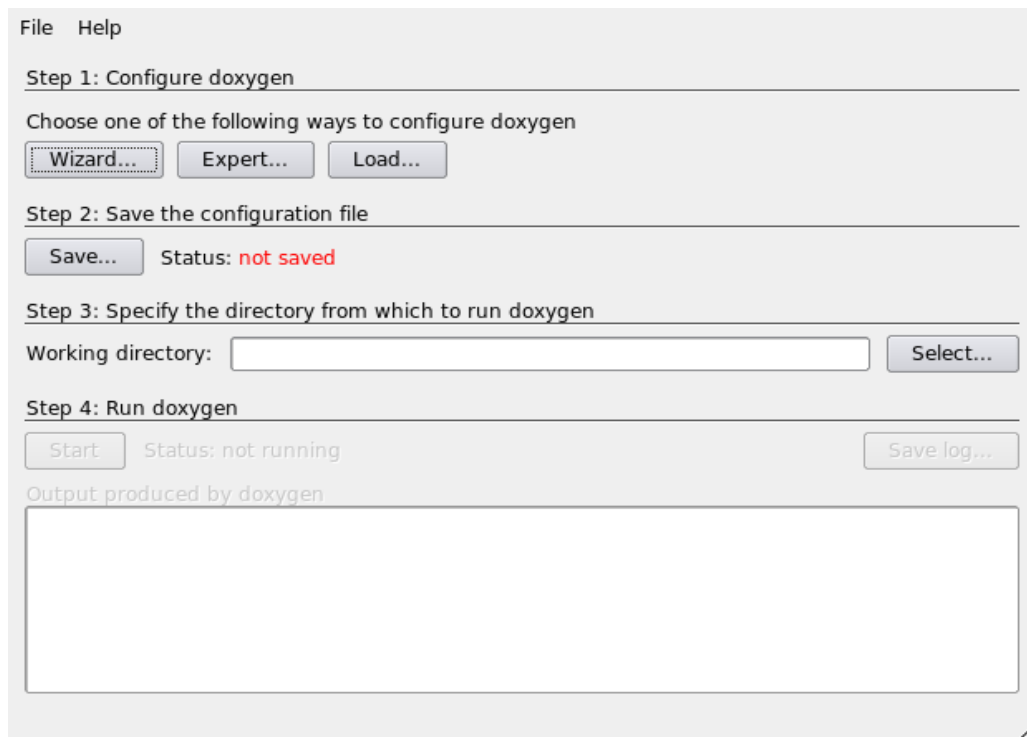


Figura 1: **Doxywizard**, la interfaz gráfica de **Doxygen**

Si bien *Doxygen*<sup>1</sup> es una herramienta de línea de comando, al que hay que pasarle un archivo de configuración preparado para el proyecto previamente, el trabajo se alivia bastante empleando *Doxywizard*, una interfaz gráfica para *Doxygen* que se instala automáticamente con la versión para **Windows**, pero que hay que instalar aparte (*doxygen-gui*) en entornos **Linux** (tal y como se muestra más abajo). Su aspecto en todos los sistemas (incluyendo también **Mac**), es el que se muestra en la figura 1.

A continuación, se muestra la línea de comando necesaria para instalar el software de los repositorios, para la distribución **Ubuntu Linux**, y derivadas (**Lubuntu**, **Kubuntu**...). En otras distribuciones, es probable que se utilicen otros gestores de software, y por tanto, el comando no sea válido para ellas.

```
$ sudo apt-get install doxygen doxygen-gui graphviz
```

1 <http://www.doxygen.org>

Para **Fedora Linux**:

```
$ sudo urpmi doxygen doxygen-doxywizard graphviz
```

Para su correcto funcionamiento, precisa del paquete de diagramación *GraphViz*<sup>2</sup>, que en entornos **Linux** se instala de manera muy sencilla según la distribución usada (como se ha visto más arriba), pero que en entornos **Windows** es necesario retocar para su correcta disponibilidad. Para que *Doxygen* encuentre a *GraphViz*, es necesario que éste último se encuentre en las rutas del sistema, almacenadas en la variable `PATH`. Así, una vez instalados ambos paquetes (supongamos en la unidad C:), será necesario ejecutar las siguientes órdenes para trabajar correctamente.

```
$ set PATH=c:\archivos de programa\graphviz\bin\;%PATH%  
$ doxywizard
```

En caso de que no se instale el paquete *GraphViz*, se pierden los gráficos de llamadas a funciones, lo cuál tampoco es vital si no se precisa.

### 3 Obtener la documentación de un proyecto

La manera más sencilla para trabajar con *Doxygen* es utilizar el *Wizard* (asistente). Durante esta sección se asumirá que se desea documentar un proyecto de gestión temperaturas que reside en un directorio llamado *GestionTemperaturas*. Si se pulsa en el botón *Wizard* (figura 2), aparece el asistente que nos permitirá obtener la documentación del proyecto.

En primer lugar, se pregunta el título de la documentación, en *project name*. Lo más importante aquí, en cualquier caso, es indicar el lugar donde residen los fuentes (*source code directory*), así como el lugar donde se va a generar la documentación (*desination directory*). Una buena idea es crear un subdirectorio dentro del directorio del proyecto (como *doc*, en el ejemplo), de manera que siga estando todo localizado en un mismo lugar.

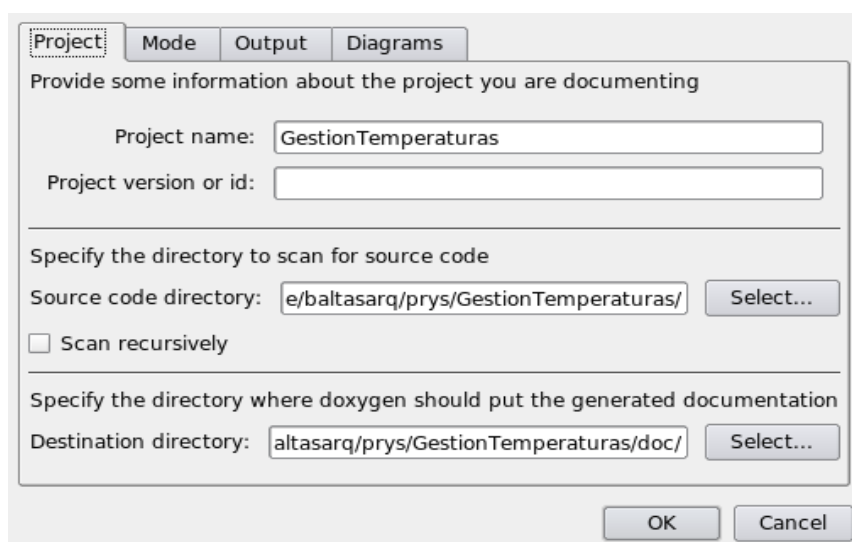
The image shows a screenshot of the 'Doxygen Wizard' dialog box, specifically the 'Project' tab. The dialog has four tabs: 'Project', 'Mode', 'Output', and 'Diagrams'. The 'Project' tab is active. The text 'Provide some information about the project you are documenting' is at the top. Below this, there are two input fields: 'Project name:' with the text 'GestionTemperaturas' and 'Project version or id:' which is empty. A horizontal line separates this section from the next. The next section is titled 'Specify the directory to scan for source code'. It contains an input field for 'Source code directory:' with the text 'e/baltasarq/prys/GestionTemperaturas/' and a 'Select...' button to its right. Below this is a checkbox labeled 'Scan recursively' which is currently unchecked. Another horizontal line follows. The final section is titled 'Specify the directory where doxygen should put the generated documentation'. It contains an input field for 'Destination directory:' with the text 'altasarq/prys/GestionTemperaturas/doc/' and a 'Select...' button to its right. At the bottom right of the dialog are 'OK' and 'Cancel' buttons.

Figura 2: Utilizando el asistente para configurar Doxygen.

A continuación, se selecciona el lenguaje de programación a emplear para el proyecto (figura 3). En realidad, *Doxygen* genera la documentación en cualquiera de las formas, pero seleccionando el lenguaje de programación la salida está mucho más pulida.

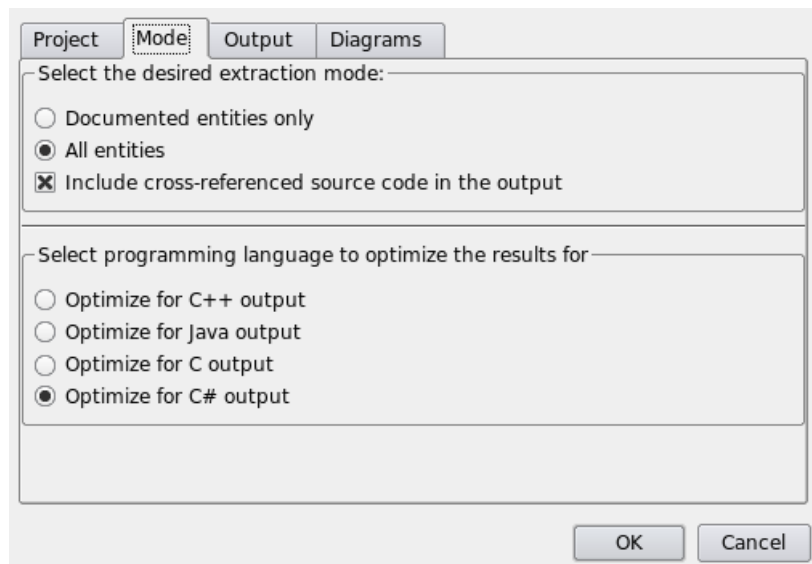


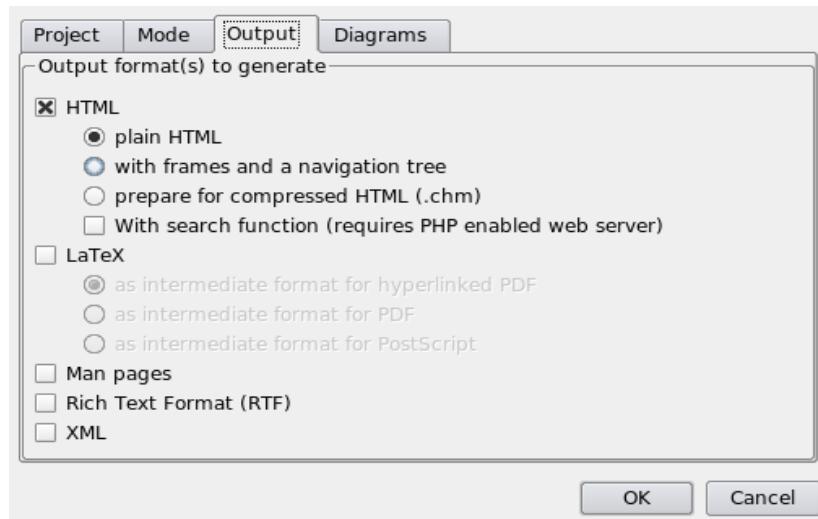
Figura 3: Selección entre lenguajes de programación disponibles.

Las dos primeras opciones son mucho más interesantes: *Documented entities only* / *All entities*. La primera se refiere a incluir en la documentación sólo aquellas entidades (funciones, constantes,...), para la que se ha escrito información específica. La segunda documenta todo en el programa. Probablemente la primera sea interesante para usuarios de una librería o aplicación, mientras la segunda es más interesante para los desarrolladores de la misma.

Finalmente, *include cross-reference source code in the output*, incluye el código fuente, de manera que se pueda navegar por él, desde la propia documentación, de función en función.

El tipo de documentación a generar es lo siguiente a elegir, tal y como se ve en la figura 4. Probablemente, la salida más útil sea HTML, aunque el formato RTF puede ser mucho más interesante para cuando se desea obtener un manual impreso.

Para terminar, los diagramas a generar, como se aprecia en la figura 5.



*Figura 4: Título del proyecto.*

Sin duda, lo mejor es utilizar *dot tool* de *GraphViz*, puesto que el diagramador interno de *Doxygen* no es demasiado completo (no genera los grafos de llamadas entre funciones, por ejemplo). Por defecto aparece desmarcada la opción *call graphs*, que precisamente son los grafos que se acaban de mencionar.

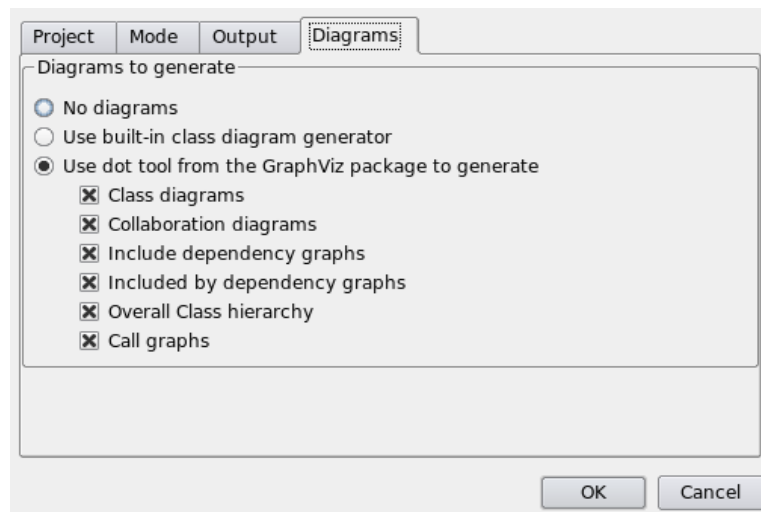


Figura 5: Perfiles y selección del compilador.

Sólo queda un pequeño detalle: *Doxygen* genera toda la documentación en inglés, por lo que si se desea cambiarlo a español, se debe acceder a la configuración avanzada: se puede hacer fácilmente, pulsando en *expert* (figura 6). En *output language*, el menú desplegable permite elegir español (*Spanish*) como lengua de salida.

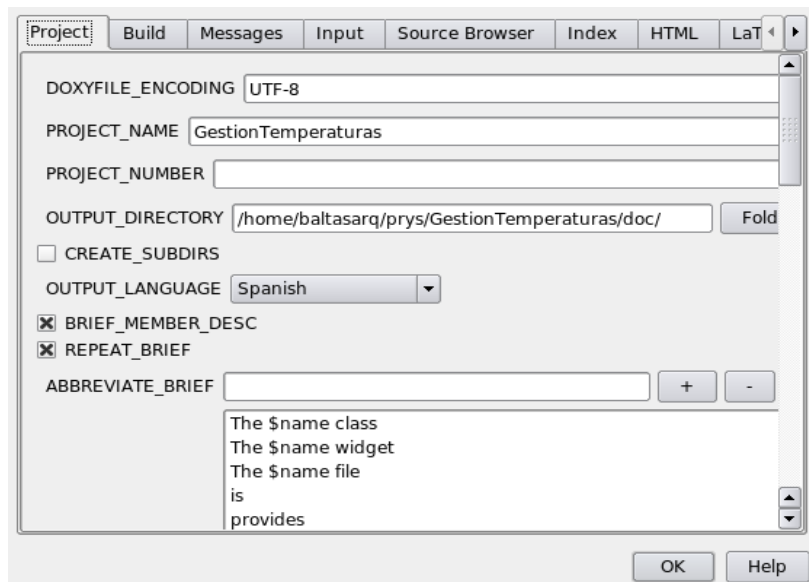


Figura 6: Configuración experta de *doxygen*, necesaria para cambiar la lengua de salida.

Una vez hecho todo ésto, sólo es necesario guardar el archivo configurador para *Doxygen* (pulsando *save*) en alguna parte. El mejor lugar es el directorio *doc* creado dentro del directorio del proyecto. También se debe elegir el directorio de trabajo (*working directory*), que de nuevo debería ser este directorio *doc*.

Una vez pulsado *Start*, la documentación habrá sido creada dentro del ya mencionado directorio *doc*, típicamente será *doc/html*. Sólo es necesario ya ejecutar el navegador sobre el archivo *index.html* para poder navegar por toda la documentación generada.

## 4 Documentar un proyecto

Aunque *Doxygen* realiza muchas tareas por el programador, obviamente no puede saber lo que significa una constante, o para qué sirve una función. Precisamente esa información debemos hacérsela saber mediante unos comentarios especiales.

Un comentario de una sola línea en C# empieza por `//`. Si el comentario empezara por `///`, *Doxygen* tomará este comentario como que contiene información importante para él, y lo procesará.

Dentro de uno de estos comentarios especiales, puede incluirse distinta información, para lo cuál se utilizan una suerte de nodos XML:

Nodo	Significado
<code>param name=""</code>	Información sobre un parámetro de una función.
<code>returns</code>	Información sobre lo que devuelve una función.
<code>summary</code>	Explicación breve sobre lo que hace una función, de manera que no sea necesario verse toda la documentación en detalle. Aparece en el índice general.
<code>see cref=""</code>	Permite establecer referencias cruzadas con otras entidades.

Un ejemplo sería el siguiente:

```
public static class Estadistica {  
  
    /// <summary>  
    /// Calcula la media del vector pasado por parámetro  
    /// </summary>  
    /// <param name="v">El vector de reales</param>  
    /// <returns>La media, como número real</returns>  
    public double calculaMedia(double[] v)  
    {  
    }  
  
    /// <summary>  
    /// Calcula la desviación típica del vector pasado por parámetro  
    /// </summary>  
    /// <param name="v">El vector de reales</param>  
    /// <returns>La desviación típica, como número real</returns>  
    public double calculaDesvTipica(double[] v)  
    {  
    }  
}
```

*Doxygen* tiene muchas más posibilidades y ventajas, pero su manejo básico no supone conocer más que estos detalles. El precio a pagar para que *Doxygen* genere una buena documentación, es, simplemente, utilizar un formato especial para los comentarios que se incluirían para describir las funciones, el cuál es perfectamente asumible.



Archivo Editar Ver Marcadores Widgets Herramientas Ayuda

Página de inicio GestionTemperaturas: Referenci...

file://localhost/home/baltasa Google

### Funciones

double [calculaMedia](#) (double v[], int numEltos)

double [calculaDesvTipica](#) (double v[], int numEltos)

### Documentación de las funciones

**double calculaDesvTipica ( double v[],  
int numEltos  
)**

**Devuelve:**  
La desviación típica de los elementos del vector

**Ver también:**  
[calculaMedia](#)

Definición en la línea 20 del archivo [etadistica.cpp](#).

Hace referencia a [calculaMedia\(\)](#).

Referenciado por [main\(\)](#).

Gráfico de llamadas para esta función:

calculaDesvTipica

calculaMedia

**double calculaMedia ( double v[],  
int numEltos  
)**

La función calcula la media del vector pasado por parámetro

**Parámetros:**  
v El vector de reales  
numEltos El número de elementos a valorar en el vector

**Devuelve:**  
La media como número real

Definición en la línea 6 del archivo [etadistica.cpp](#).

Referenciado por [calculaDesvTipica\(\)](#), y [main\(\)](#).

100%