!Mola documentar código C con doxygen!

Angel Perles

Abril 2011

Table of Contents

1. Objetivo

Documentar el código de nuestros programas es fundamental para recordar de qué va o para que otros puedan aprovecharlo.

Por otra parte es una pesadez ¿no?, y más pesado es si nos toca crear un manual que explica de qué van nuestras funciones. Y, si dejamos pasar unos días, lo más seguro es que no recordemos ni de que van, retrasando el trabajo de desarrollo o de documentación.

A mi me gusta programar y ver los resultados, y después pasar a otra cosa, mariposa. Documentar no me gusta nada, así que he decidido probar suerte con doxygen.

Doxigen permite generar documentación automática a partir de los comentarios que insertamos en nuestro pogramas. Hay otras herramientas que hacen lo mismo, pero he elegido ésta por estar ampliamente aceptada en la comunidad "open" y en muchos productos comerciales.

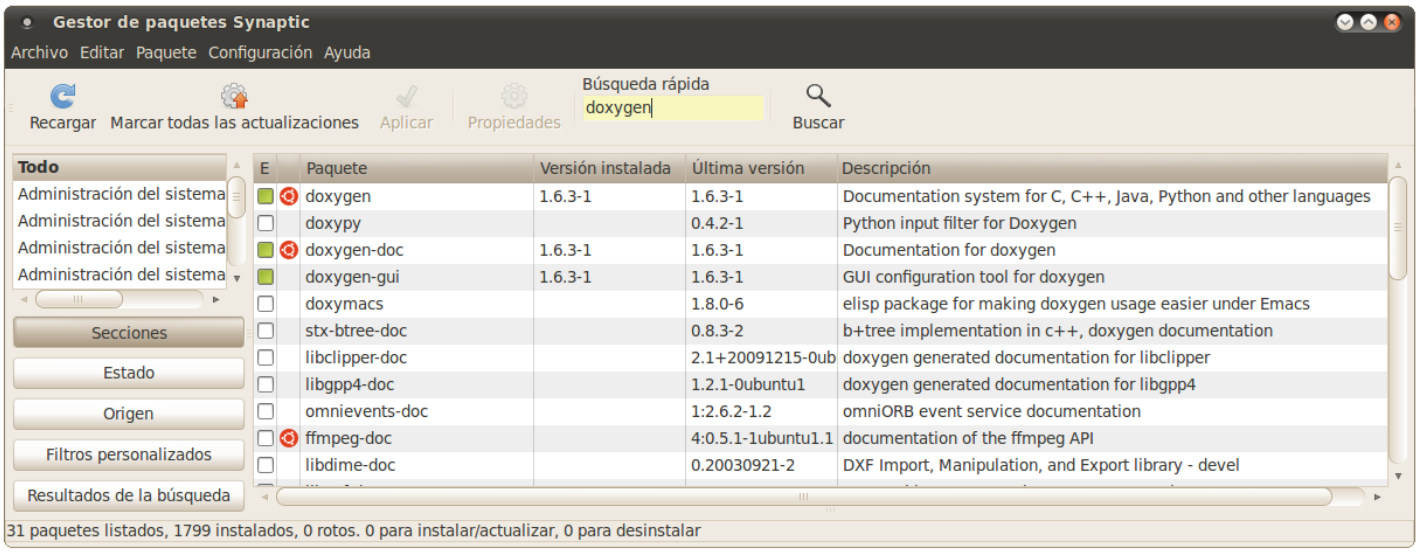
Como a otros les puede resultar útil y yo me olvido fácilmente, he decidido hacer un resumen muy "al grano" de cómo aprovechar esta herramienta.

2. Instalando doxygen en Linux

La herramienta está disponible en los repositorios, así que es inmediato. Para que las cosas sean aún más fáciles, se instalará una herramienta gráfica para generar la configuración de doxygen.

Como ejemplo de instalación, la siguiente figura muestra la selección de paquetes en Synaptic para Ubuntu Linux 10.04 LTS, donde se han seleccionado los paquetes "doxygen", "doxygen-doc" y doxygen-gui". Le damos a "instalar" y listo.

Figure 1. Seleccionando doxygen en Synaptic



3. Instalando doxygen en Windows

Dicen que funciona en Windows y en MAC), pero no lo he probado.

Si alguien está interesado, puede probar, me lo cuenta y lo incorporo.

4. Añadiendo comentarios doxygen al código

4.1. El formato de los comentarios

Si echamos un vistazo a un código C comentado para que se genere automáticamente la documentación verempos que, aparentemente, no son más que comentarios normalitos. El detalle está en introducir esos comentarios de una manera especial para que sea reconocida por la herramienta.

Para hacerse una idea, introducir un comentario doxygen consiste en empezar un comentario en C usando dos "*" despues de la barra "/". Por ejemplo:

```
/**    <--- marca para doxygen

  Vaya, vaya, quien iba a decir que esto era tan fácil.
*/
```

Dentro del comentario podremos introducir símbolos especiales seguidos de palabras clave que permiten introducir dintintos tipos de documentación.

4.2. Comentando un módulo C

Es importante que, cuando se desarrolla un módulo C se inserte en su cabecera información sobre el nombre del archivo que lo contien, la fecha, su funcionalidad, el autor, etc. Para esta tarea se puede utilizar las siguientes palabras clave:

- @file nombre del archivo contenedor del módulo.
- @brief descripción "corta" de la tarea del módulo. Se supone que se añade también una descripción larga.
- @author autor del módulo.
- @date fecha de creación de módulo.

Por ejemplo,

```
/**
  @file cdm_daq.cpp
  @brief Master-side implementation of simple access functions to DAQ services provided by a ChapDAQ slave

  Bla, bla .... demà serà

  @author Angel Perles
  @date 5/2010, 4/2011

*/
```

4.3. Comentando funciones

En una funcioón es interesante dar una descripción de su cometido y, por supuesto, indicar qué parámetros de entrada necesita y qué devuelve como resultado. Para esta tarea se puede usar:

- @brief descripción corta del cometido de la función
- @param para indicar el cometido de un parámetro
- @returns para indicar qué devuelve una función
- @verbatim y @endverbatim para inserttar un trozo de texto "literal", quizá un ejemplo de uso

Por ejemplo, los siguientes comentarios se deben colocar antes de la función a describir:

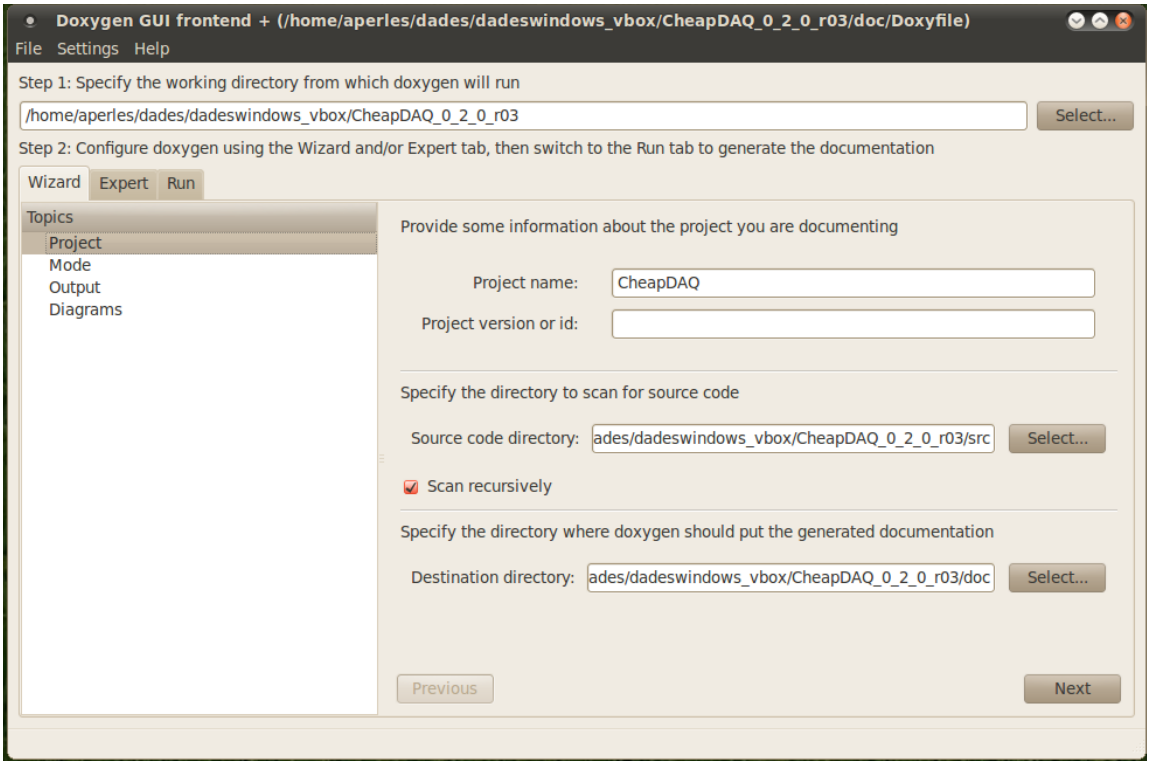
```
/*****************************************************************************************/
/**
  @brief Writes the digital output of a given line in a given subdevice

  Most digital output systems are organized in groups of lines, (8, 16, 32 ...) and the aim of this function is to handle only
  one line a given group

  @param subdevice target subdevice, an integer between 0 and 255
  @param line line to be modifided, in general, a value between 0 and 31, but depends on the target slave
  @param value state to be written to thew line, mus be 0 or 1

  @returns the result of the operation, being cd_Error_NoError if all worked OK

  Example:
  \verbatim
      cd_Error my_error;

      my_error = cdm_DAQWriteDigitalLine(2,31,1);
      if (my_error != CD_Erro_NoError) {
            printf("Ups, this failed with error %s\n", cd_ErrorStr(my_error);
      }
   \endverbatim

  */
cd_Error cdm_DAQWriteDigitalLine(uint8_t subdevice, uint8_t line, uint8_t value) {
 ...

}
```

5. Generando la documentación

Para generar la documentación es necesario crear un archivo llamado `doxyfile` con los parámetros deseados. Con el propósito de facilitar las cosas, se recurrirá a la aplicación de asistencia doxywizard para generar este archivo y producir la documentación.

Una vez ejecutado doxywizard, se procederá a rellenar los parámetros del proyecto de documentació. La siguiente figura muestra la ventana inicial en modo "wizard", que es el que vamos a utilizar.

Figure 2. Datos del proyecto



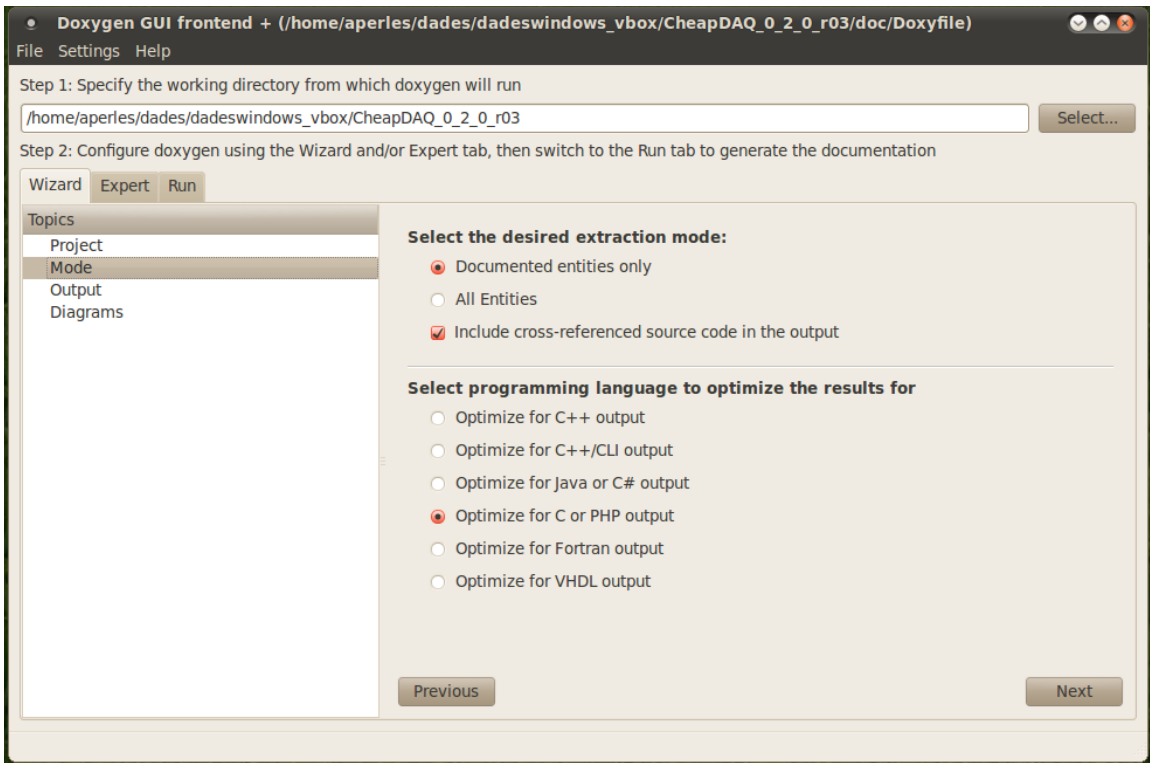
En el paso "1" se deberá indicar el directorio de trabajo raíz del proyecto.

En el paso "2" se deben ir eligiendo los distintos "topics" y ajustarlos a nuestras necesidades.

En el "topic" "Project" se deberá dar un nombre al proyecto de documentación, se deberá indicar dónde está el código fuente a analizar y se deberá indicar dónde depositar la documentación.

Pasando a "topic" "Mode" nos encontramos el diálogo mostrado en la siguiente figura.

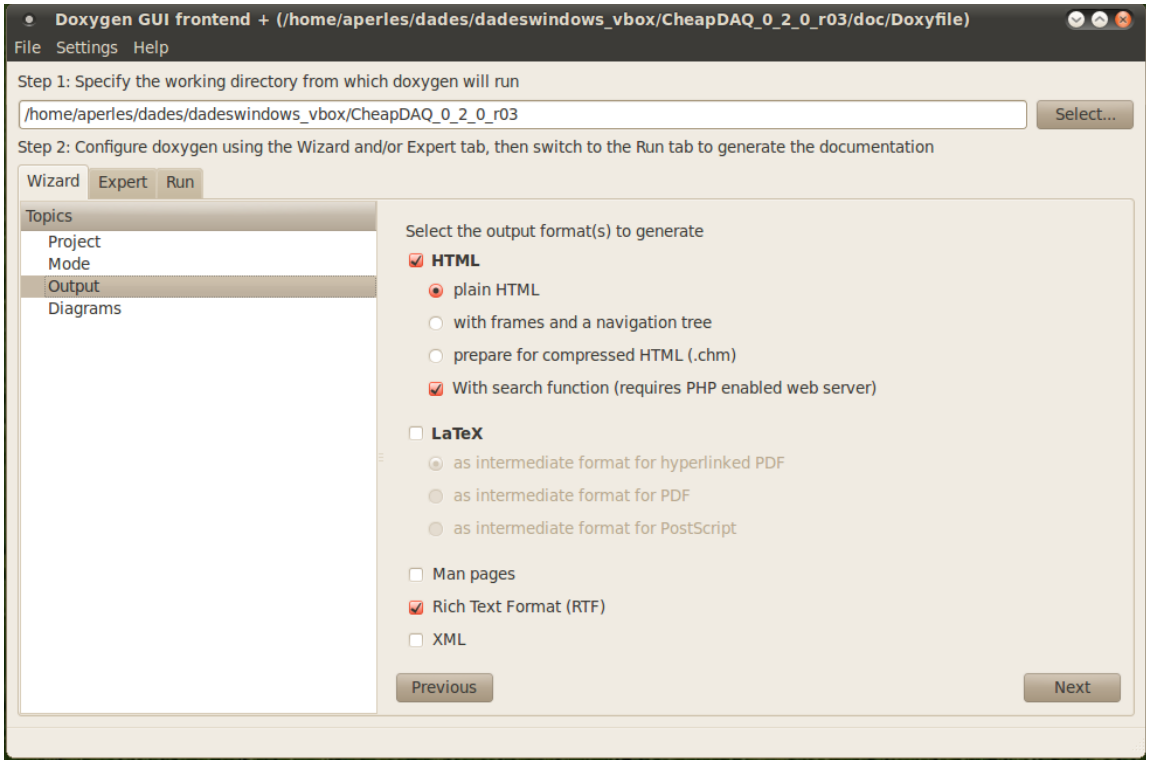
Figure 3. Datos del modo de trabajo



En este paso se indica si se debe documentar todos los archivos de código que se localicen o solo los documentados. Además se deberá indicar el lenguaje de programación empleado.

Pasando al "topic" "Output" nos encontramos el diálogo mostrado en la siguiente figura.

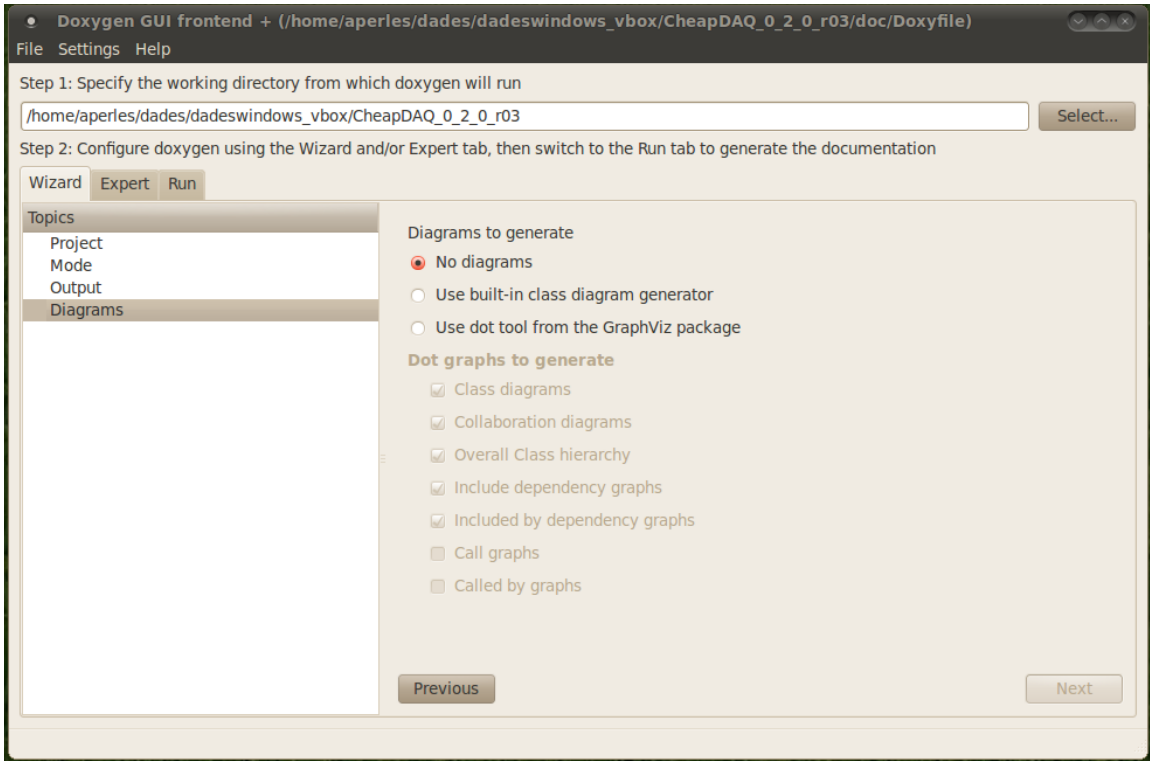
Figure 4. Datos del formato de salida



En este paso se seleccionará el formato de la salida de documentación.

Pasando al "topic" "Diagrams" nos encontramos el diálogo mostrado en la siguiente figura.

Figure 5. Datos para la generación de diagramas



En este paso es posible indicar que se generen diagramas que indiquen la relación entre distintas partes del código. Esto es muy úyil, por ejemplo, para documentar clases en C++.

Una vez configurados todos los parámetros, debemos guradar la configuración por si queremos recuperarla en otro momento. Utilizar File->Save para ello.

Si queremos ajustar aún más la configuración, podemos acudir a la pestaña "Expert".

Por último, seleccionando la pestaña "Run" podemos ejecutar "doxygen" y, tachaaannn, generar la documentación.

6. El resultado

El resultado de la ejecución de doxygen sobre los ejemplos anteriores se puede ver aquí en formato html.