# Doxygen使用简介 = =

目录

[Doxygen使用简介 = = 1](#_Toc243466113)

[更新记录 1](#_Toc243466114)

[**在代码中加入注释** 1](#_Toc243466115)

[文件 1](#_Toc243466116)

[命名空间 2](#_Toc243466117)

[类 2](#_Toc243466118)

[函数 3](#_Toc243466119)

[结构体 4](#_Toc243466120)

[枚举 4](#_Toc243466121)

[变量 5](#_Toc243466122)

[宏 5](#_Toc243466123)

[使用DoxyWizard生成CHM文档 6](#_Toc243466124)

[**使用Visual Assist Snippets快速创建注释** 10](#_Toc243466125)

## 更新记录

2009-10-14 创建文档

2009-10-15 更新：[**使用Visual Assist Snippets快速创建注释**](#_使用Visual_Assist_Snippets快速创建注释)

2009-10-16 更新：[**输出图形设置**](#Dot)目录

## 在代码中加入注释

Doxygen是根据代码中的注释自动生成的，它的起始标识是///。

下面是一些注释样例：

### 文件

|  |
| --- |
| **代码** |
| |  | | --- | | //////////////////////////////////////////////////////////////////////////  /// @file IDBOp.h  /// @brief DBOp接口  /// @version 0.1  ////////////////////////////////////////////////////////////////////////// | | |
| **效果** |
|  |
| **VA Snippets**[可以直接使用cpl文件，详细请看[使用Visual Assist Snippets快速创建注释](#_使用Visual_Assist_Snippets快速创建代码)] |
| ////////////////////////////////////////////////////////////////////////////  // Copyright(c) 1999-$YEAR$, TQ Digital Entertainment, All Rights Reserved  // Author: Your Name  // Created: $YEAR$/$MONTH\_02$/$DAY\_02$  //  /// @file $FILE\_BASE$.$FILE\_EXT$  /// @brief $end$  /// @version 0.1  //////////////////////////////////////////////////////////////////////////// |

### 命名空间

|  |
| --- |
| **代码** |
| |  | | --- | | //////////////////////////////////////////////////////////////////////////  /// @namespace dbop  /// @brief 数据库操作命名空间 | | |
| **效果** |
|  |
| **VA Snippets**[可以直接使用cpl文件，详细请看[使用Visual Assist Snippets快速创建注释](#_使用Visual_Assist_Snippets快速创建代码)] |
| ////////////////////////////////////////////////////////////  /// @namespace $end$  /// @brief |

### 类

|  |
| --- |
| **代码** |
| |  | | --- | | //////////////////////////////////////////////////////////////////////////  /// @class IDBOp  /// @brief 数据库操作类  ///  /// \n通过本类可以对数据库进行连接/断开、设置字符集、查询、更新等基本操作。 | | |
| **效果** |
|  |
| **VA Snippets**[可以直接使用cpl文件，详细请看[使用Visual Assist Snippets快速创建注释](#_使用Visual_Assist_Snippets快速创建代码)] |
| //////////////////////////////////////////////////////////////////////////  /// @class $end$  /// @brief |

### 函数

|  |
| --- |
| **代码** |
| |  | | --- | | /////////////////////////////////////  /// @brief 执行SQL语句  /// @param [in] pszSQL 要执行的SQL语句  /// @param [in] bAsyn 是否异步操作（默认为false，即同步）  /// @return bool 是否执行成功（如果是同步，则执行成功; 如果是异步，则SQL语句加入队列中）  /// @note 执行中可能发生的错误信息通过GetQueryError() 获得。  /// @see GetQueryError()  virtual bool Execute(const char\* const pszSQL, bool bAsyn = false) = 0; | | |
| **效果** |
|  |
| **VA Snippets**[可以直接使用cpl文件，详细请看[使用Visual Assist Snippets快速创建注释](#_使用Visual_Assist_Snippets快速创建代码)] |
| /////////////////////////////////////  /// @brief  /// @param [in]  /// @param [out]  /// @return |

### 结构体

|  |
| --- |
| **代码** |
| |  | | --- | | //////////////////////////////////////////////////////////////////////////  /// @struct DB\_INFO  /// @brief 数据库信息结构体  ////////////////////////////////////////////////////////////////////////// | | |
| **效果** |
|  |
| **VA Snippets**[可以直接使用cpl文件，详细请看[使用Visual Assist Snippets快速创建注释](#_使用Visual_Assist_Snippets快速创建代码)] |
| //////////////////////////////////////////////////////////////////////////  /// @struct $end$  /// @brief |

### 枚举

|  |
| --- |
| **代码** |
| |  | | --- | | //////////////////////////////////////////////////////////////////////////  /// @enum DB\_TYPE  /// @brief 数据库类型  //////////////////////////////////////////////////////////////////////////  enum DB\_TYPE  {  DBTYPE\_MYSQL, ///< MySQL数据库类型  DBTYPE\_ORACLE, ///< Oracle数据库类型  DBTYPE\_MSSQL ///< Microsoft SQL Server数据库类型  }; | | |
| **效果** |
|  |
| **VA Snippets**[可以直接使用cpl文件，详细请看[使用Visual Assist Snippets快速创建注释](#_使用Visual_Assist_Snippets快速创建代码)] |
| //////////////////////////////////////////////////////////////////////////  /// @enum $end$  /// @brief |

### 变量

|  |
| --- |
| **代码** |
| |  | | --- | | char m\_szDBName[LEN\_DB\_INFO\_STR]; ///< 数据库名 | | |
| **效果** |
|  |
| **VA Snippets** |
| ///< $end$ |

### 宏

|  |
| --- |
| **代码** |
| |  | | --- | | /// @brief 导出符号定义 | | |
| **效果** |
|  |
| **VA Snippets** |
| /// @brief $end$ |

其他还有许多使用方式，

可以参考doxygen的文档：<http://www.stack.nl/~dimitri/doxygen/commands.html#cmdcode>。

附生成主页面的指令：

/// @mainpage 主页标题

/// @section section\_id 段落名

/// 段落

其他指令也可以带入，如

/// @code

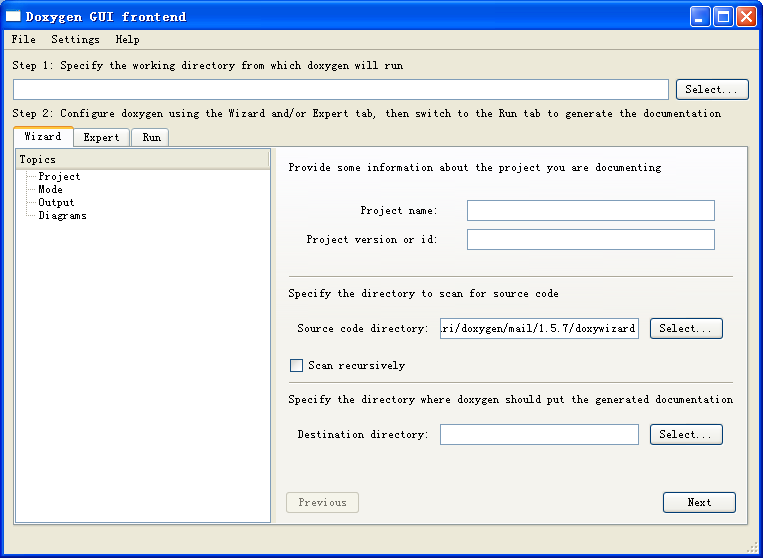
/// 代码段

/// @endcode

/// @see 参照

## 使用DoxyWizard生成CHM文档

安装好后，开始菜单会多出doxygen菜单，打开里面的DoxyWizard。界面如下图。

Step1是Doxygen的工作目录，请指定一个已存在的非中文的文件夹。

Step2做具体配置工作。

首先是**Wizard**选项卡：

* **Project**

Project name: 项目名称

Project version or id: 项目版本号

Source code directory: 项目源码目录

Destination directory: 文档输出目录

* **Mode**

保持默认选项（**Document Entity Only**与**Optimize for C++ output**）即可。

* **Output**

要生成CHM文档请选择HTML项中的**prepare for compressed HTML (.chm)**。

同时将**With search function (requires PHP enabled web server)**的钩去掉。

LaTeX，如果不需要在文档中生成LaTeX公式的话可以不选。

* **Diagrams**

选择第二项**Use Build-In class diagram generator**，将使用Doxygen内置的生成功能生成每个类的类图（如果它只有一个类的时候是不会生成的 = =）。

如果需要使用更强大的功能比如类继承体系图，请选择第三项**Use dot tool from the GraphViz package**，此功能需要安装***GraphViz***软件。

其次是**Export**选项卡，配置项比Wizard内容多出许多，这里只做简单介绍。

* **Project**

OUTPUT\_LANGUAGE，选择Chinese。

TAB\_SIZE 是Tab的长度，默认为8，大家根据自己喜好……

* **Build**

默认是会生成public方法，但是貌似有时会莫名奇妙地少掉一些方法的详细信息。

这里也选上EXTRACT\_ALL，它保证输出所有public方法及protected方法，static方法不在此范围内。

若要包含static方法的注释，请选中EXTRACT\_STATIC。

同理EXTRACT\_PRIVATE。

我们生成文档的目的是为了方便各位调用类与函数，因此生成ALL、STATIC、LOCAL\_CLASSES就好了吧 = =。

* **Messages**

生成时的提示信息，默认即可。

* **Input**

Input为输入目录，支持多个目录，我们可以放入项目目录和Include目录。

下面的Exclude是忽略目录与文件。

* **Source Browser**

源码浏览器，默认即可。

* **Index**

钩选ALPHABETICAL\_INDEX，类中将有一个组合类型索引项。如下图所示：



* **HTML**

如果你之前选择了**prepare for compressed HTML (.chm)**，

这里的GENERATE\_HTMLHELP项会是钩选状态。

它下面的CHM\_FILE填写你的CHM文档名字。

HHC\_LOCATION则选择你的HTML Help WorkShop安装目录下的hhc程序，

一般会在C:/Program Files/HTML Help Workshop/hhc.exe。

Doxygen生成的默认是UTF-8，因此若不指定CHM\_INDEX\_ENCODING为GBK的话，CHM会有部分乱码。

钩选TOC\_EXPAND，doxygen会为你生成左边树目录。

* **Dot**

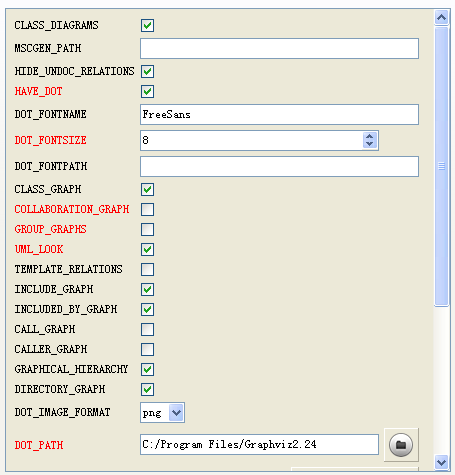
如果你选用内置的生成功能（即选择**Use Build-In class diagram generator**），此时CLASS\_DIAGRAMS会是钩选状态，而HAVE\_DOT则是未选状态，默认即可;

如果你选用***GraphViz***的dot工具生成（即选择**Use dot tool from the GraphViz package**）情况则相反，请你钩选上CLASS\_DIAGRAMS。此时你需要设置下面的DOT\_PATH为***GraphViz***的安装目录，否则将无法生成。

另外以下选项钩选则生成对应的图，不选则不生成：

* + CLASS\_GRAPHS 类图
  + COLLABORATION\_GRAPH 协作图
  + GROUP\_GRAPHS 组图
  + UML\_LOOK 是否UML外观
  + INCLUDE\_GRAPH include
  + INCLUDED\_BY\_GRAPH 被include
  + CALL\_GRAPH 调用
  + CALLER\_GRAPH 被调用
  + DIRECTORY\_GRAPH 目录图
  + GRAPHICAL\_HIERARCHY 继承体系图

建议钩选以上下划线的几项。效果应如下所示：



DOT\_IMAGE\_FORMAT是生成的图片类型，有PNG/JPG/GIF三种格式可选。

其他项没有用过，请大家自己研究 = =。

配置好后即可运行，进入Run选项卡，单击**Run doxygen**即开始生成。

对话框会显示调试信息，生成好后点击下面的**Show HTML Output**可以打开生成的HTML首页。

chm文件则在你指定的生成目录中自己找。

关闭前不要忘了保存配置文件，下次可以继续使用。

它会自动提示你是否需要保存，你也可以选择File菜单的Save项自己保存。

完毕！^^

## 使用Visual Assist Snippets快速创建注释

请下载与本文同在tools目录下的<Cpp.tpl>，

覆盖C:\Documents and Settings\Administrator\Application Data\VisualAssist\Autotext下的同名文件，即可在VA启用的情况下使用。

具体方法请查看[Cpp.tpl 使用说明.docx](Cpp.tpl%20使用说明.docx)