使用 doxygen 生成 chm 文档

一、 注释规范

1. 文件头的标注

```
/**
* @file      文件名
* @brief     对文件的简述  可用\n（换行）
* @Details.
* @author    作者名
* @version   1.0.0.1(版本号) \n
*-----------------------------------------------------\n
* Change History :\n
* <Date>     | <Version> | <Author>    |   <Change>    \n
*-----------------------------------------------------\n
* 2014/01/24 | 1.0.0.1   |   作者名    |  Create file   \n
*/
```

2. 命名空间

```
/**
 * @brief 命名空间的简单概述 \n(换行)
 * 命名空间的详细概述
 */
 namespace str{ }
```

3. 类、结构、枚举标注

```
/**
* @brief 类的简单概述 \n(换行)
* 类的详细概述
*/
class Example{ };
```

枚举类型定义、结构体类型定义注释风格类似

```
/**
* @brief 简要说明文字
*/
typedef struct 结构体名字
{
   成员 1, /*!< 简要说明文字 */ or ///<说明,  /**<说明 */ 如果不加<，则会认为是成员 2 的注释
}结构体别名;
```

4. 函数注释原则

```
/**
* @brief 函数简要说明-测试函数
* @param index        参数1
* @param t            参数2 @see CTest
*
* @return 返回说明
*        -<em>false</em> fail
*        -<em>true</em> succeed
*/
bool Test(int index, const CTest& t);
```

note：指定函数注意项事或重要的注解指令操作符
note 格式如下：

```
@note 简要说明
```

retval：指定函数返回值说明指令操作符。(注:更前面的 return 有点不同.这里是返回值说明)
retval 格式如下：

```
@retval 返回值 简要说明
```

pre：指定函数前置条件指令操作符
pre 格式如下：

```
@pre 简要说明
```

par：指定扩展性说明指令操作符讲。（它一般跟 code、endcode 一起使用 ）
par 格式如下：

```
@par 扩展名字
```

code、endcode：指定
code、endcode 格式如下：

```
@code
    简要说明(内容)
@endcode
```

see：指定参考信息。
see 格式如下：

```
@see 简要参考内容
```

deprecated：指定函数过时指令操作符。
deprecated 格式如下：

```
@deprecated 简要说明
```

调试 Bug 说明
    解决的 bug 说明，@bug
警告说明 (warning)
    定义一些关于这个函数必须知道的事情，@warning
备注说明 (remarks)
    定义一些关于这个函数的备注信息，@remarks
将要完成的工作 (todo)
    说明哪些事情将在不久以后完成，@todo
使用例子说明 (example)
    例子说明，@example example.cpp

```
/**
    * @brief 打开文件 \n
    * 文件打开成功后，必须使用::CloseFile 函数关闭
    * @param[in] fileName    文件名
    * @param[in] fileMode    文件模式，可以由以下几个模块组合而成：
    *     -r 读取
    *     -w 可写
    *     -a 添加
    *     -t 文本模式(不能与 b 联用)
    *     -b 二进制模式(不能与 t 联用)
    * @return 返回文件编号
    *  --1 表示打开文件失败(生成时:.-1)
    * @note 文件打开成功后，必须使用::CloseFile 函数关闭
    * @par 示例:
    * @code
    *        //用文本只读方式打开文件
    *        int ret = OpenFile("test.txt", "a");
    * @endcode
    * @see 函数::ReadFile::CloseFile (“::”是指定有连接功能,可以看文档里的
CloseFile 变成绿,点击它可以跳转到 CloseFile.)
    * @deprecated 由于特殊的原因，这个函数可能会在将来的版本中取消
    */
    int OpenFile(const char* fileName, const char* fileMode);

    /**
    * @brief 关闭文件
    * @param [in] file    文件
    *
    * @retval 0     成功
    * @retval -1    失败
    * @pre file 必须使用 OpenFile 的返回值
    */
    int CloseFile(int file);
```

-：生成一个黑心圆.

-#：指定按顺序标记。

::：指定连接函数功能。（注：空格和“:”有连接功能,但建议还是使用”::”。只对函数有用。）

它们格式如下: (-和::例子前面有了,就介绍-#例子。)

- 简要说明

-# 简要说明

::函数名

例：

```
 /**
  * @param [in] person 只能输入以下参数：
  * -# a:代表张三          // 生成 1. a:代表张三
  * -# b:代表李四          // 生成 2. b:代表李四
  * -# c:代表王二          // 生成 3. c:代表王二
 */
 void GetPerson(int p);
```

5. 变量注释

```
/// 简述
/** 详细描述. */
```

或者

```
//! 简述
//! 详细描述
//! 从这里开始
int m_variable_1; ///< 成员变量 m_variable_1 说明
int m_variable_2; ///< 成员变量 m_variable_1 说明

/**
* @brief 成员变量 m_c 简要说明
*
* 成员变量 m_variable_3 的详细说明，这里可以对变量进行
* 详细的说明和描述，具体方法和函数的标注是一样的
*/
bool m_variable_3;
```

如果变量需要详细说明的可已按照 m_varibale_3 的写法写,注意,m_variable_2 和 m_variable_3 之间一定需要空行，否则会导致 m_variable_2 的简述消失

6. 模块标注

模块定义格式:

```
/**
* @defgroup 模块名   页的标题名 (模块名只能英文,这个可以随便取. 在一个源文件里不能相同)
* @{ (跟 c 语言{一样起作用域功能)
```

```
*/
… 定义的内容 …
/** @} */

例：
/**
* @defgroup HenryWen Example.cpp
* @{
*/
  … 定义的内容 …
/** @} */
```

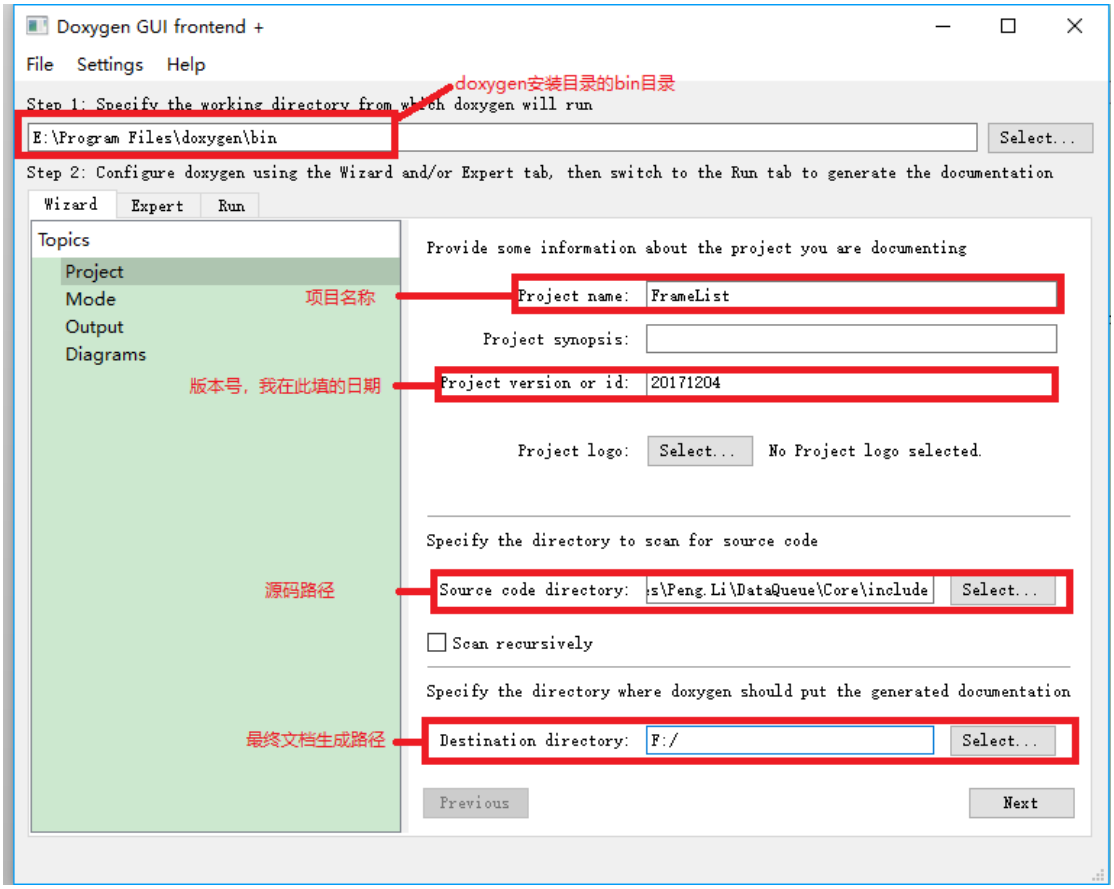
7. 分组标注

分组定义格式：

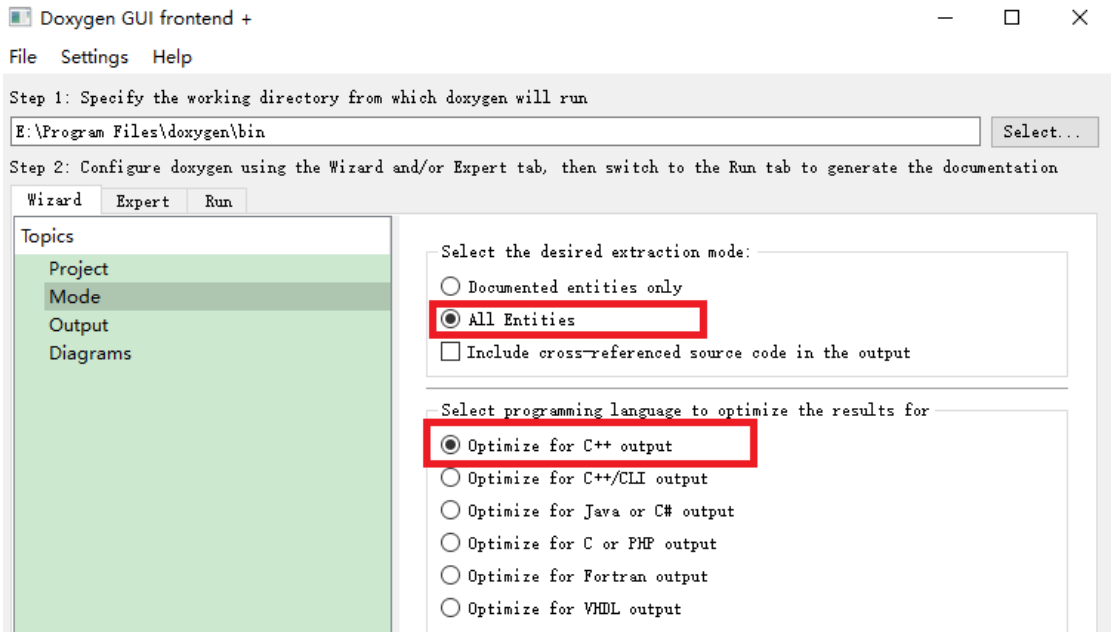
```
/**
* @name 分组说明文字
* @{
*/
… 定义的内容 …
/** @} */

例：
/**
* @name PI 常量
* @{
*/
#define PI 3.1415926737
/** @} */

/**
* @name 数组固定长度常量
* @{
*/
const int g_ARRAY_MAX = 1024;
/** @} */
```

二、 生成 chm 文档

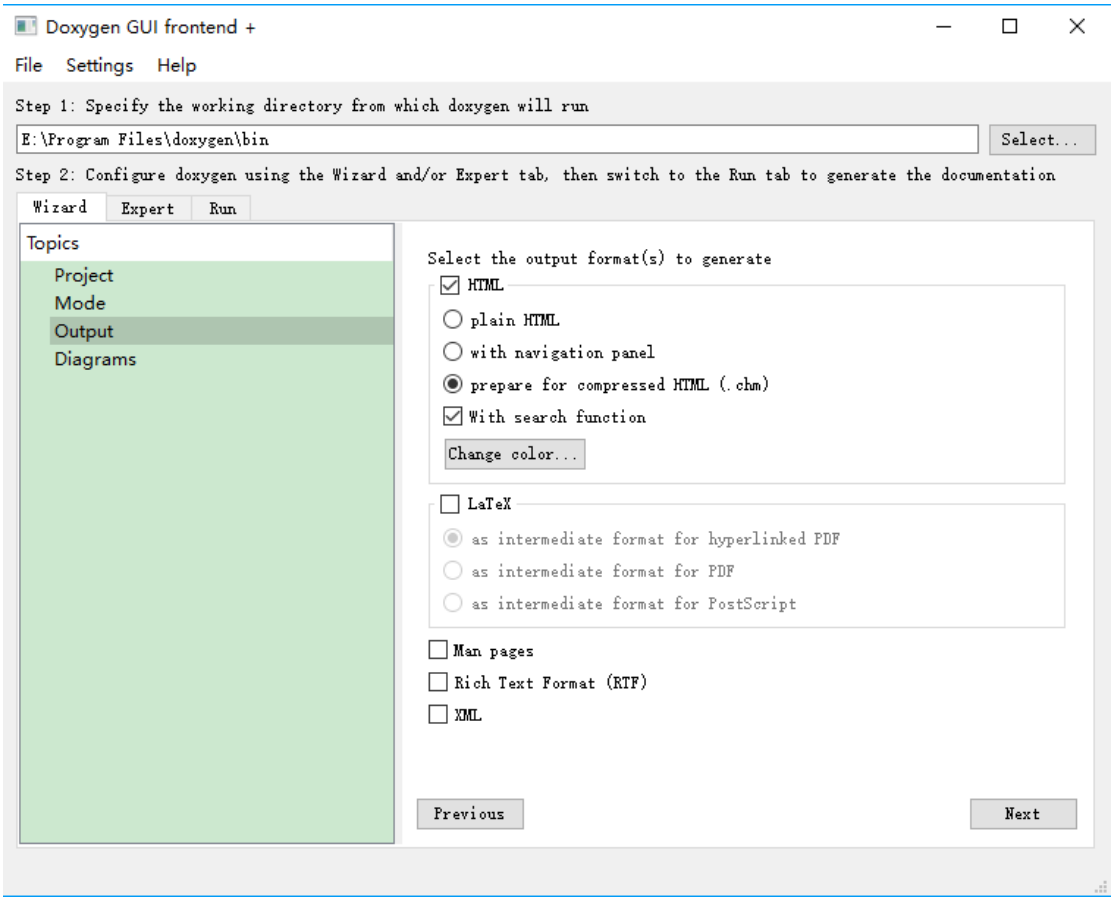
1. 分别安装 doxygen、graphviz、htmlhelp，然后在系统环境变量 path 下加入 doxygen 安装目录下的 bin 目录
2. 运行 doxygen 安装目录下 bin 目录中的 doxywizard.exe
3. Wizard->Project 页按照下图进行设置调整参数。



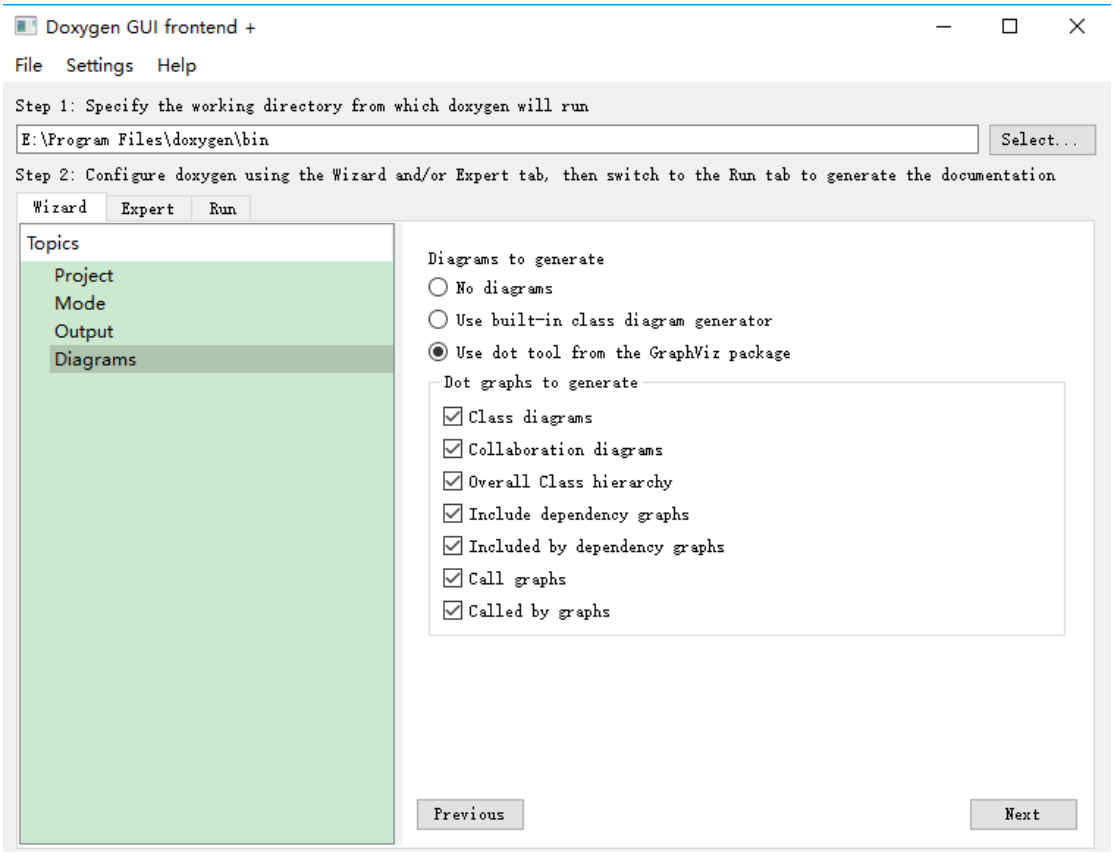
4. Wizard->Mode 页按照下图进行设置调整参数。



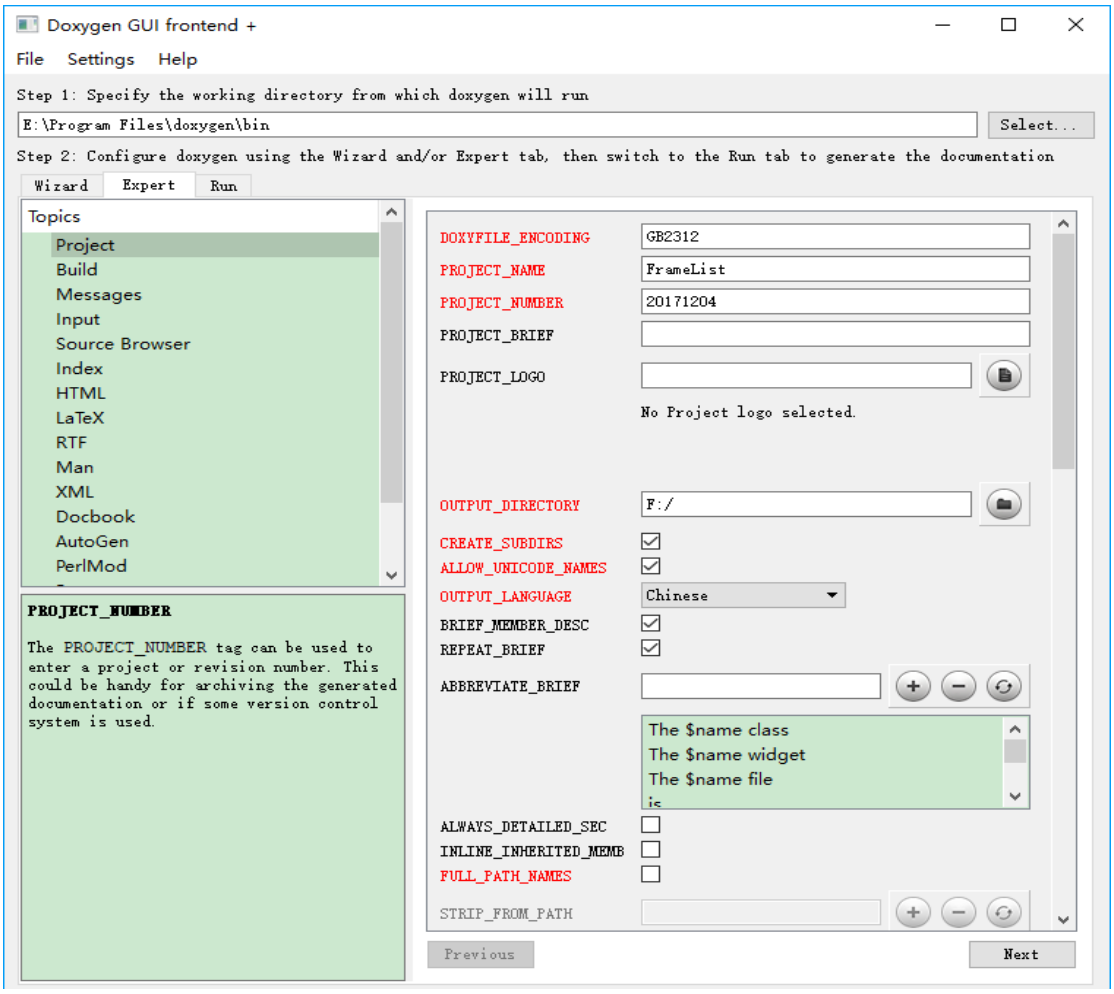
5. Wizard->OutPut 页按照下图进行设置调整参数。



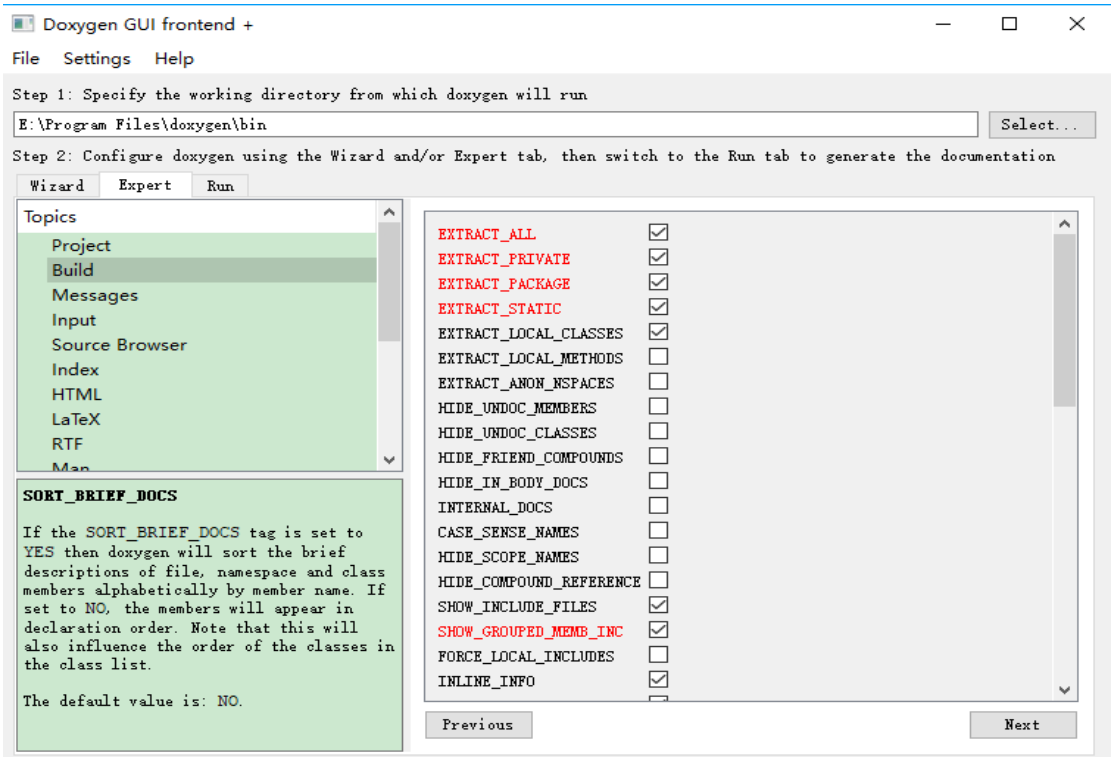
6. Wizard->Diagrams 页按照下图进行设置调整参数。



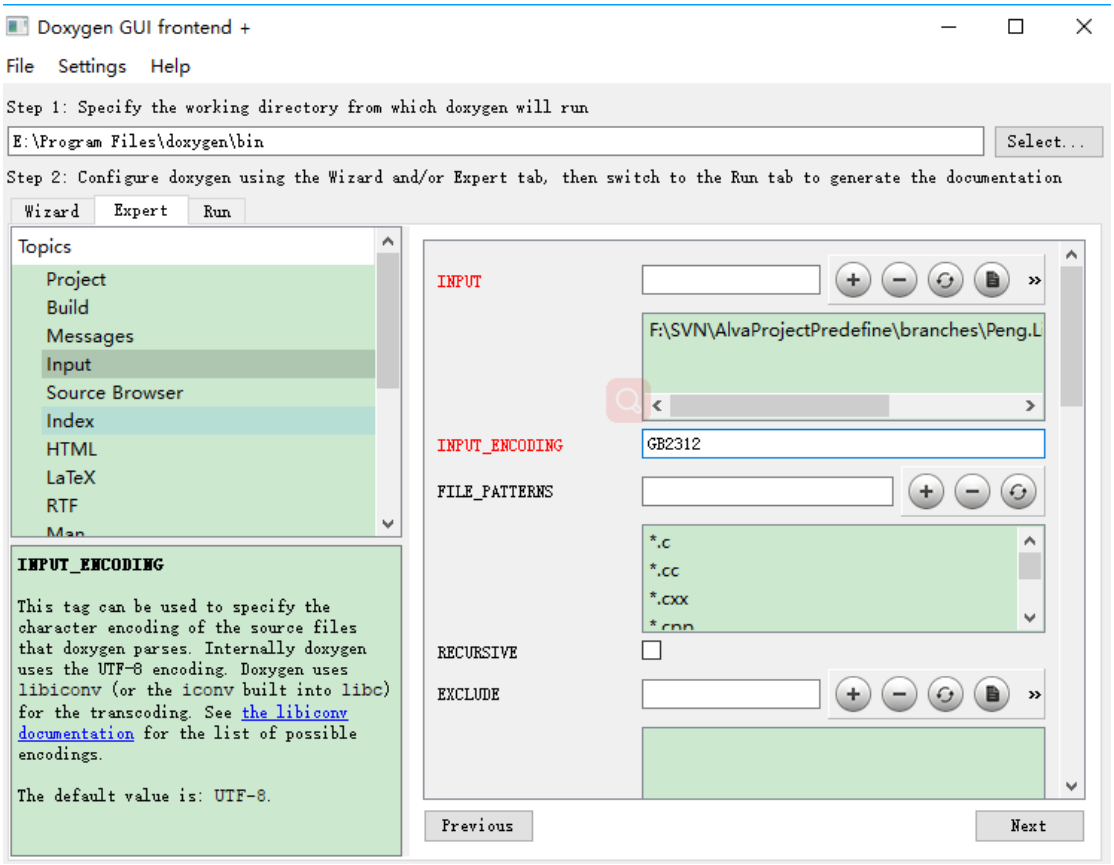
7. Expert>Project 页按照下图进行设置调整参数。



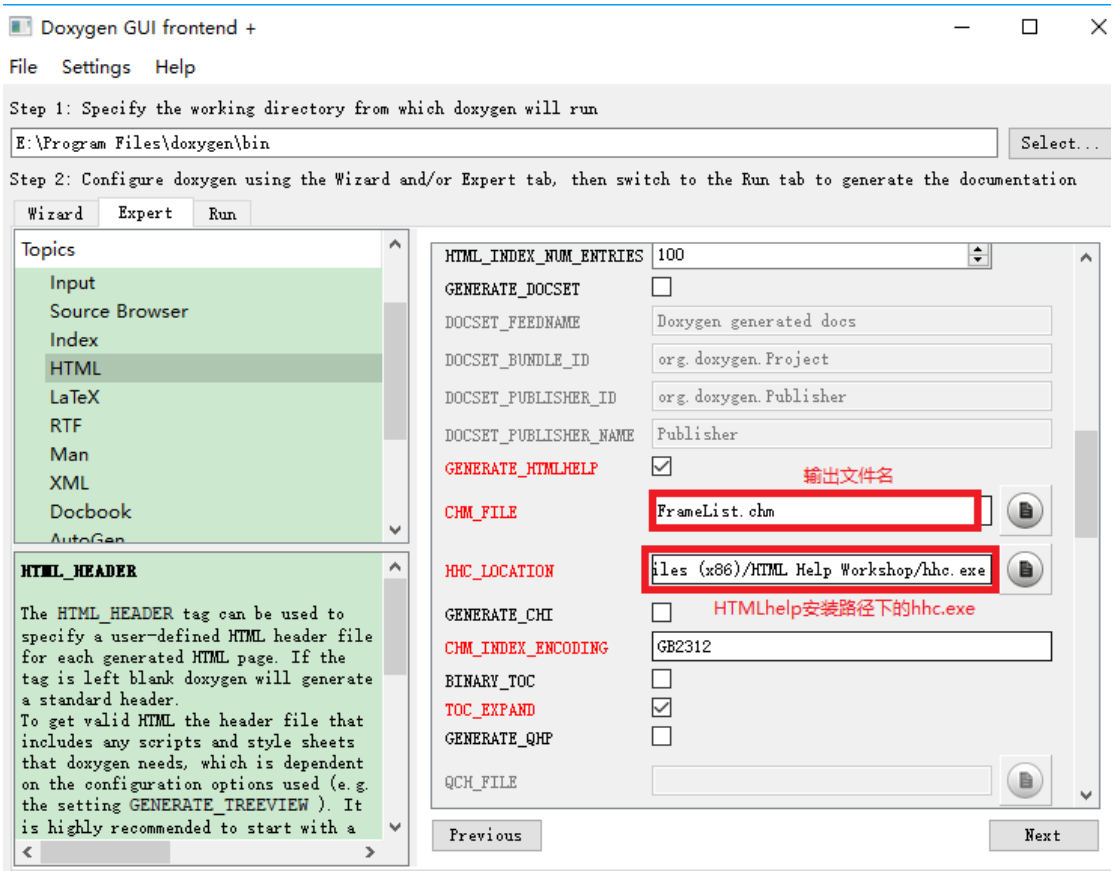
8. Expert>Build 页按照下图进行设置调整参数。



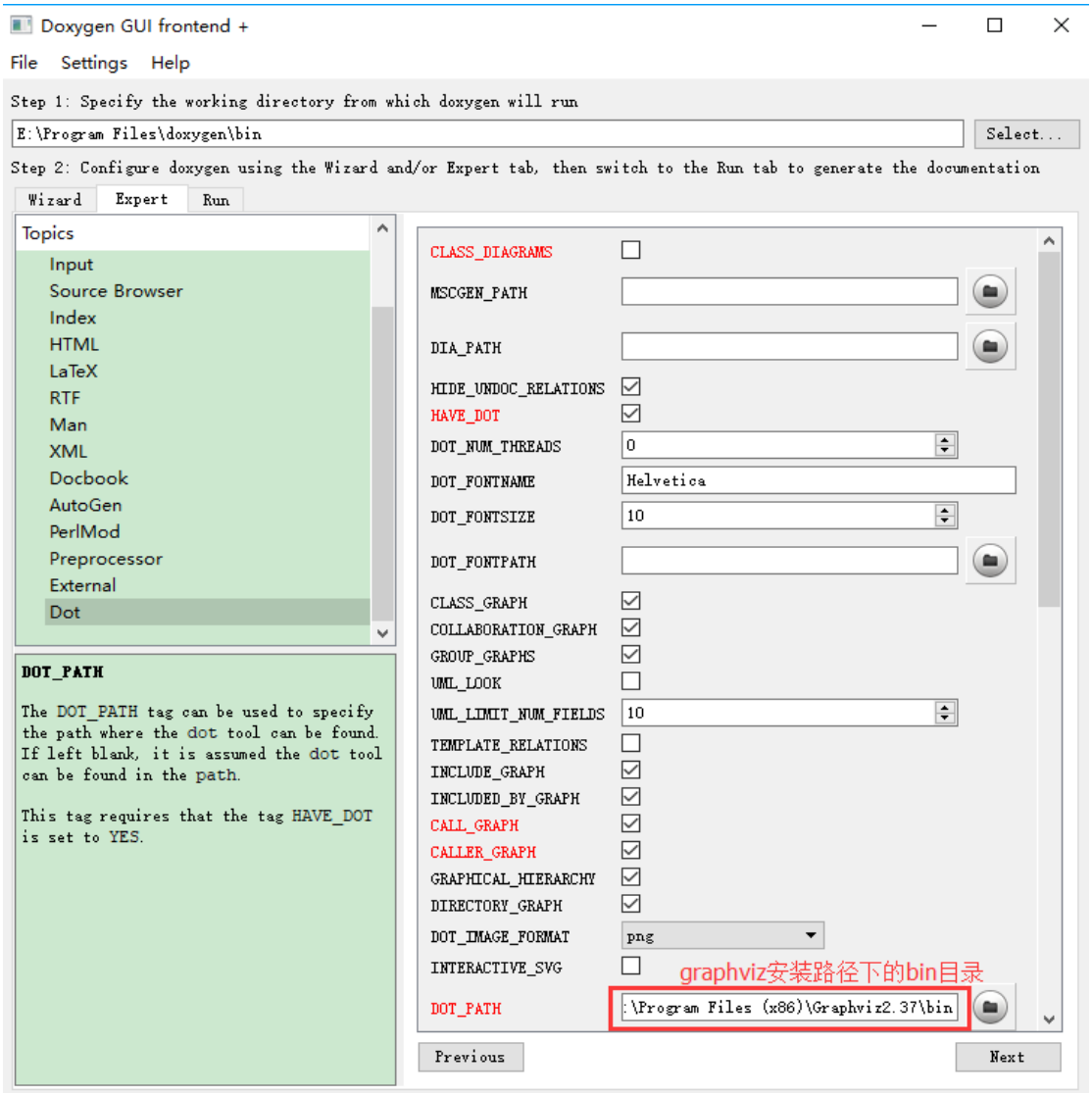
9. Expert>Input 页按照下图进行设置调整参数。



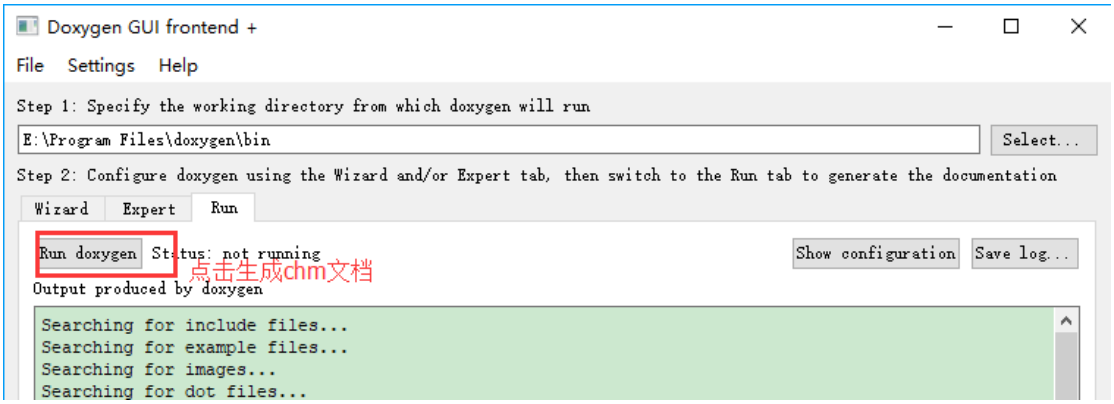
10. Expert>Html 页按照下图进行设置调整参数。



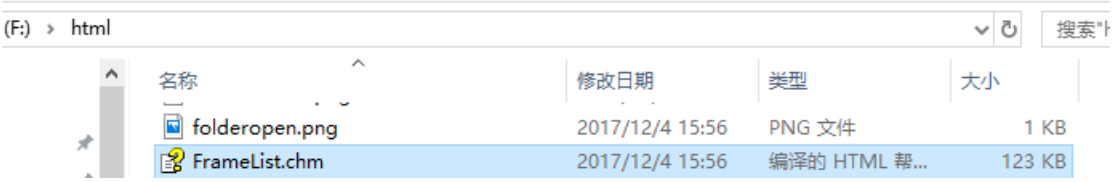
11. Expert>Dot 页按照下图进行设置调整参数。



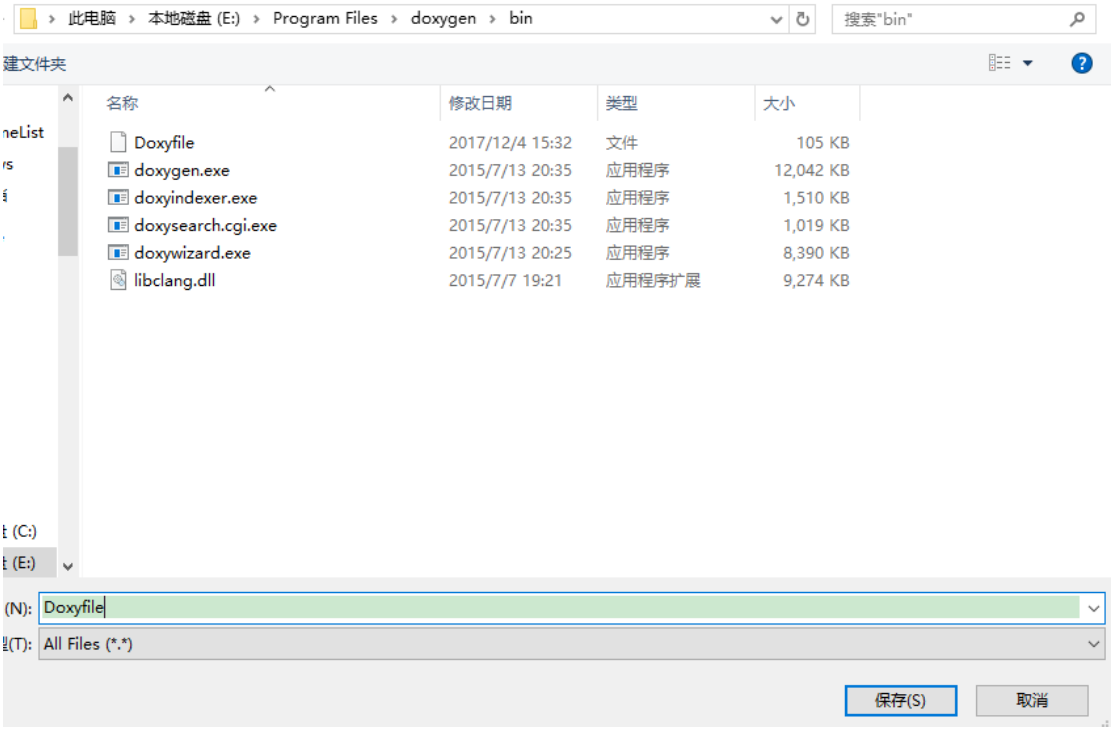
12. Run 页按照下图进行点击生成 chm 文件。



13. chm 文件生成路径。



14. 到此 chm 文件生产完毕，为了下次代码调整后再次生产 chm 文件此时我们可以将目前所做的工作进行保存，点击 File->Save as，保存到 doxygen/bin 目录下，文件名为 Doxyfile。



15. 后期使用可以通过打开中的 file 下的 open 菜单打开 Doxyfile 文件重新调整设置生产新的 chm 文件。
16. 如果要生成多个 include 目录下的.h 文件的说明文档，则用 notepad++打开你保存的 Doxyfile 文件，然后找见 INPUT，添加目录即可，此时生成的文档说明在一个 chm 文件中。