HarrixMakeLibrary v.1.7

А. Б. Сергиенко

22 февраля 2015 г.

Аннотация

HarrixMakeLibrary — это программа собирающая библиотеки функций на языке C++ и справку к ним из исходных материалов.

Оглавление

1 Внешний вид программы

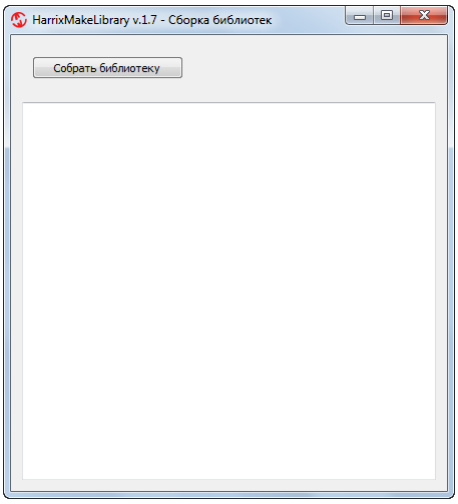

Рисунок 1. Внешний вид программы

При нажатии на кнопку **«Собрать библиотеку»** будет производиться сборка библиотеки вместе с файлами справки. После чего будет открыта папка с сформированными файлами.

В текстовом поле под кнопкой будет отображаться ход работы программы.

Это программа является упрощенной версией программы **MakeHarrixMathLibrary** из библиотеки **HarrixMathLibrary**. Помогает собирать файлы библиотек, если каждая функция прописана в отдельных файлах.

2 Результат работы программы

В папке **source_library** находится исходный материал, который обрабатывается программой HarrixMakeLibrary.exe, в результате чего образуются следующие элементы:

- **Library.cpp** — главный файл библиотеки;
- **Library.h** — заголовочный файл;
- **Library_Help.tex** — файл справки в формате LaTeX.

Обратите внимание, что это не полноценные файлы исходников библиотеки — это только сборка функций в соответствующие файлы. И код этих файлов надо вставлять в ваши основные файлы исходных кодов библиотек. В общем, программа генерирует основу, а всякие дополнительные элементы вроде подключение include и др. это вам самим прописывать.

Все данные файлы собираются в папке **temp_library**.

3 Как собирается библиотека

Исходники библиотеки находятся с папке **source_library**.

Файлы **Library.cpp** и **Library.h** собираются следующим образом.

В папке **source_library** есть директории. Каждая директория — это множество функций какого-то раздела. Перед рассмотрением файлов папки программа добавляет в файл Library.cpp следующий код:

Код 1. Название раздела

```
//*************************************************************
//[Название папки]
//*************************************************************
```

А в файл Library.h добавляется код:

Код 2. Название раздела

```
//[Название папки]
```

После каждой функции в Library.cpp вставляется код:

Код 3. Название раздела

```
//-----------------------------------------------------------------------------
```

Далее программа пробегает по каждой папке, которая представляет собой раздел функций в библиотеке. Каждая функция в разделе предоставляется следующими файлами:

- **<File>.cpp** или **<File>.tpp** — код функции;
- **<File>.h** — заголовочный файл функции;
- **<File>.tex** — справка по функции;
- **<File>.desc** — описание функции;
- **<File>.use** — пример использования функции (из него удаляются пробелы в начале строк, равным числу пробелов вначале первой строки);
- **<File>_<name>.pdf** — множество рисунков, необходимых для справки по функции (необязательные файлы);
- **<File>_<name>.png** — множество рисунков, необходимых для справки по функции (необязательные файлы);

Важно помнить, что каждый *.cpp, *.h, *.tex файл в папках папки **source_library** не является полноценным файлом соответствующего расширения и без сборки в единые файлы библиотеки не может использоваться.

Разница файлов *.cpp и *.tpp в том, что в *.tpp пишется код шаблонов функций, а в *.cpp пишутся обычные функции, и их реализация располагается в Library.cpp файле, тогда как шаблоны располагаются в Library.h файле.

Ниже показан алгоритм (Алгоритм .1) формирования файлов библиотеки.

Итоговое количество функций определяется как количество знаков «;» в h файлах функций, которые располагаются в папках.

Алгоритм .1. Алгоритм собирания файлов библиотеки

Начало алгоритма
Выполнить для всех папок **выполнять**
$Library.cpp+ =$ *Код 1. Название раздела*;
$Library.h+ =$ *Код 2. Название раздела*;
Выполнить для всех файлов папки расширения *.cpp, *.tpp и *.h **выполнять**
Если есть файл *.cpp **тогда**
$Library.cpp+ =< File > .cpp$;
иначе
$ResultTpp+ =< File > .tpp$;
Конец условия
$Library.h+ =< File > .h$;
Конец цикла
Конец цикла
$Library.h+ = ResultTpp$;
Сохранить $Library.cpp$ в папке temp_library;
Сохранить $Library.h$ в папке temp_library;
Конец алгоритма

Стоит отметить, что все разделы функций и сами функции сортируются в алфавитном порядке.

4 Как собирается справка

Исходники файлов справки библиотеки находятся с папке **source_library**.

Файлы **Library.tex** собирается следующим образом, как показано ниже в алгоритме (Алгоритм .2) формирования файлов справки библиотеки.

Некоторые моменты по преобразованию некоторых данных (например, преобразование $< File > .desc$) не рассматривается в алгоритме, но вы можете все посмотреть в исходном коде программы, которая поставляется с данной библиотекой в папке **source**

Также еще собирается файл **FUNCTIONS.md** со списком функций в формате **Markdown**, например, для размещения в GitHub.

5 Исходники HarrixMakeLibrary.exe и справки по ним

HarrixMakeLibrary написан на Qt. Не требует каких-то дополнительных файлов. Исходники программы располагаются в папке **source**.

Исходники справки HarrixMakeLibrary (данного файла, который вы читаете) по располагаются в папке **source\help**. Главный файл исходника справки — это файл HarrixMakeLibrary_Help.tex.

Алгоритм .2. Алгоритм собирания файлов справки библиотеки

Начало алгоритма
$ResultTexList+=$ *Заголовок для списка функций*;
$ResultTexFunctions+=$ *Заголовок для функций*;
Выполнить для всех папок **выполнять**
  $ResultTexList+=$ *Заголовок раздела*;
  $ResultTexFunctions+=$ *Заголовок раздела*;
  $n=0$;
  Выполнить для всех файлов папки расширения *.desc, *.tex, *.h, *.use **выполнять**
    $ResultTexList+=<File>.desc$ в обработке;
    $ResultTexFunctions+=<File>.desc$ в обработке;
    $ResultTexFunctions+=<File>.h$ в обертке;
    $ResultTexFunctions+=<File>.tex$;
    $ResultTexFunctions+=<File>.use$ в обертке;
    $n++$;
  Конец цикла
  Выполнить для всех файлов папки расширения *.pdf и *.png **выполнять**
    Скопировать файл <File>.<png|pdf> в папку **\images**;
  Конец цикла
Конец цикла
$Library\_Help.tex+=ResultTexList$;
$Library\_Help.tex+=ResultTexFunctions$;
Сохранить $Library\_Help.tex$ в папке temp_library;
Конец алгоритма