**1.准备工具**

**一 .OpenCV**： [官网地址](https://link.juejin.cn?target=https%3A%2F%2Fopencv.org%2F%23) [下载地址](https://link.juejin.cn?target=https%3A%2F%2Fsourceforge.net%2Fprojects%2Fopencvlibrary%2F)

**二 .CMake**： [官网地址](https://link.juejin.cn?target=https%3A%2F%2Fcmake.org%2F) [下载地址](https://link.juejin.cn?target=https%3A%2F%2Fcmake.org%2Fdownload%2F)

下载建议 可以3.25.x 好像Clion目前仅支持到3.25.x，我这里使用cmake3.19因为我的clion使用这个版本

下载地址：<https://cmake.org/files/v3.19/>

**三：TDM-GCC** ： [下载地址](https://link.juejin.cn?target=https%3A%2F%2Fjmeubank.github.io%2Ftdm-gcc%2Fdownload%2F)

这里很多网站的帖子都建议下载minGW ，*但是不少Min-GW编译器不支持C++版本11或更高版本。OpenCV中的线程需要这个版本*。所以我建议使用TDM—GCC，exe直接安装就行了，还有一个优点就是他会自己你配置环境变量。很方便

# 2.安装tdm-gcc

## 1、直接运行下载好的exe文件，为了方便我把它安装到clion的文件夹里面，他会自动配置环境变量，安装好后如图

|  |
| --- |
|  |

## 2.测试

### 打开一个cmd窗口，输入gcc -v，如图，说明安装成功

|  |
| --- |
|  |

# 3.配置clion使用tdm-gcc作为编译器

## 1、打开clion，点击customize，点击右边的all settings，如图

|  |
| --- |
|  |

## 2.在弹出的窗口点击build，execution,deployment下面的toolchain，如图

|  |
| --- |
|  |

## 3.这里的是我们之前的配置，这个minGW版本不支持多线程，我们改为TDM-GCC，如图，他会自动搜索相关工具，搜索完成后如下图

|  |
| --- |
|  |

## 4.点击apply然后点击ok，设置完成如图

|  |
| --- |
|  |

## 5，新建一个简单下面测试一下，一切正常

|  |
| --- |
|  |

# 4.编译tdm-gcc版本的opencv库

## 1.在opencv源码外面创建一个cv\_tdm\_gcc\_build文件夹，用来存放编译后的库文件

|  |
| --- |
|  |

## 2.打开D:\programs\cmake-3.19.0-win64-x64\bin\cmake-gui.exe，然后填写opencv3.4.1源码路径和输出路径，如图

|  |
| --- |
|  |

## 3.点击config按钮，在弹出的窗口中选择**MinGW Makefiles，如图**

|  |
| --- |
|  |

## 4.点击finish按钮，然后就开始配置项目了，需要等待一段时间

|  |
| --- |
|  |

## 5.配置完成后如图，注意此时只是opencv核心配置完成

|  |
| --- |
|  |

## 6. 拖动滑块到OPENCV\_EXTRA\_MODULES\_PATH并且点击...按钮选择contrib模块所在路径，这里是：F:\tools and softwares\opencv3.4.1+contrib3.4.1\opencv\_contrib-3.4.1\modules

|  |
| --- |
|  |

## 注意：不要把路径粘贴到这里，需要点击按钮，然后在弹出的对话框里面查找，否则报错

## 7.勾选OPENCV\_ENABLE\_NONFREE

|  |
| --- |
|  |

## 8.勾选Build\_opencv\_world

|  |
| --- |
|  |

## 9需要再点击一次config，等待红色消失

|  |
| --- |
|  |

## 10.点击generate按钮， 等待生成完成

|  |
| --- |
|  |

### 完成后界面如下

|  |
| --- |
|  |

## 11.现在可以关闭cmake了

# 5.clion编译opencv

## 1.打开cmd窗口，进入F:\tools and softwares\opencv3.4.1+contrib3.4.1\cv\_tdm\_gcc\_build目录

|  |
| --- |
|  |

## 2.输入命令：mingw32-make -j4回车

|  |
| --- |
|  |

### 然后就开始编译了

|  |
| --- |
|  |

### 编译过程比较漫长，等待大概2个小时，编译完成如图

|  |
| --- |
|  |

## 3.编译成功后，执行mingw32-make install安装，如图

|  |
| --- |
|  |

### 安装完成如图

|  |
| --- |
|  |

## 4. mingw32-make install之后，mingw\_build目录下会生成一个install目录，这里就是编译好的opencv库。

|  |
| --- |
|  |

## 5. 为了方便使用我把opencv库拷贝到clion里面，改名：built-opencv

|  |
| --- |
|  |

## 6.为了避免opencv程序运行是缺少动态链接库，我们把built-opencv\x64\mingw\bin目录也加入环境变量的Path中。

|  |
| --- |
|  |

# 6.clion opencv测试，

## 1.新建一个clioncvdemo1项目，配置CMakeLists.txt如下

|  |
| --- |
| cmake\_minimum\_required(**VERSION 3.19**) project(**clioncvdemo1**)  set(**CMAKE\_CXX\_STANDARD 14**) set(**OpenCV\_DIR D:/programs/CLion\_2021.1.2\_Protable/built-opencv**) set(**OpenCV\_LIBS D:/programs/CLion\_2021.1.2\_Protable/built-opencv/x64/mingw/lib**) find\_package(**OpenCV 3 REQUIRED**) include\_directories(**D:/programs/CLion\_2021.1.2\_Protable/built-opencv/include**) add\_executable(**clioncvdemo1 main.cpp**) target\_link\_libraries(**clioncvdemo1 ${OpenCV\_LIBS}**) *# D:\programs\CLion\_2021.1.2\_Protable\built-opencv\x64\mingw\lib* |

这里OpenCV\_LIBS设置的环境变量为：**D:/programs/CLion\_2021.1.2\_Protable/built-opencv/x64/mingw/lib**

 CMakeList.txt主要添加包含目录，就是程序中引入的opencv相关头文件，不会报找不到问题，设置库目录，就是把opencv相关的lib依赖加入项目中，避免编译报错

## 2.修改main.cpp的代码

|  |
| --- |
| #include **<iostream>** #include **<opencv2/opencv.hpp>** #include **<opencv2/dnn/dnn.hpp>** #include **<string>  using namespace** std; **using namespace** cv;  **int** main() {  cout << **CV\_VERSION** << endl;  Mat src;  src = imread(**"../girl-nice.png"**);  imshow(**"girl"**,src);  waitKey();  destroyAllWindows();  **return** 0; } |

注意：

1，修改了CMakeLists.txt后，如果不会自动刷新，需要在他的上面点击右键-》Reload Cmake Project来重新载入设置

2.用clion生成的程序可执行文件在源文件的下一级目录，所以图片必须用“../图片名称”来获取，（好像又方法设置）

3.生成的exe文件后不会自动调用需要你进入exe文件的目录里面调用

# 扩展：改进所以图片必须用“../图片名称”来获取，使用cmake的文件复制方法

## 修改CmakeList.txt,添加把图片复制到输出目录的代码

|  |
| --- |
| cmake\_minimum\_required(**VERSION 3.19**) project(**clioncvdemo1**)  set(**CMAKE\_CXX\_STANDARD 14**) set(**OpenCV\_DIR D:/programs/CLion\_2021.1.2\_Protable/built-opencv**) set(**OpenCV\_LIBS D:/programs/CLion\_2021.1.2\_Protable/built-opencv/x64/mingw/lib**) find\_package(**OpenCV 3 REQUIRED**) include\_directories(**D:/programs/CLion\_2021.1.2\_Protable/built-opencv/include**) add\_executable(**clioncvdemo1 main.cpp**)  #复制图片到输出目录 file(**COPY ${CMAKE\_SOURCE\_DIR}/girl-nice.png DESTINATION ${EXECUTABLE\_OUTPUT\_PATH}/**) target\_link\_libraries(**clioncvdemo1 ${OpenCV\_LIBS}**) |

## 然后可以把main.cpp文件改为下面的代码

|  |
| --- |
| #include **<iostream>** #include **<opencv2/opencv.hpp>  using namespace** std; **using namespace** cv;  **int** main() {  cout << **CV\_VERSION** << endl;  Mat src;  src = imread(**"./girl-nice.png"**);  imshow(**"girl"**,src);  waitKey();  destroyAllWindows();  **return** 0; } |

## 然后编译运行程序，效果如下

|  |
| --- |
|  |

# 到此，用Clion来配置opencv开发环境就成功了