# Using OpenCV with gcc and CMake

> **Note:** We assume that you have successfully installed OpenCV in your workstation.

- The easiest way of using OpenCV in your code is to use CMake. A few advantages (taken from the Wiki):
  1. No need to change anything when porting between Linux and Windows
  2. Can easily be combined with other tools by CMake( i.e. Qt, ITK and VTK )
- If you are not familiar with CMake, checkout the tutorial on its website.

## Steps

### Create a program using OpenCV

Let's use a simple program such as DisplayImage.cpp shown below.

```
#include <stdio.h>
#include <opencv2/opencv.hpp>

using namespace cv;

int main(int argc, char** argv )
{
    if ( argc != 2 )
    {
        printf("usage: DisplayImage.out <Image_Path>\n");
        return -1;
    }

    Mat image;
    image = imread( argv[1], 1 );

    if ( !image.data )
    {
        printf("No image data \n");
        return -1;
    }
    namedWindow("Display Image", WINDOW_AUTOSIZE );
    imshow("Display Image", image);

    waitKey(0);

    return 0;
}
```

### Create a CMake file

Now you have to create your CMakeLists.txt file. It should look like this:

```
cmake_minimum_required(VERSION 2.8)
project( DisplayImage )
find_package( OpenCV REQUIRED )
add_executable( DisplayImage DisplayImage.cpp )
target_link_libraries( DisplayImage ${OpenCV_LIBS} )
```

### Generate the executable

This part is easy, just proceed as with any other project using CMake:

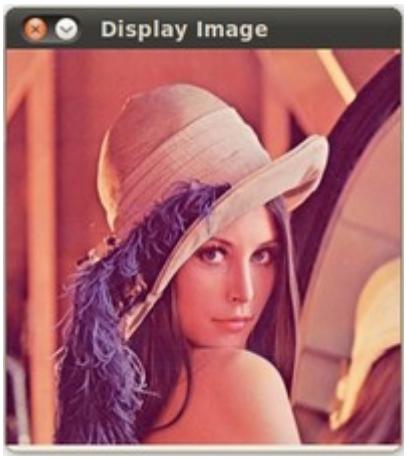
```
cd <DisplayImage_directory>
cmake .
make
```

## Result

By now you should have an executable (called DisplayImage in this case). You just have to run it giving an image location as an argument, i.e.:

```
./DisplayImage lena.jpg
```

You should get a nice window as the one shown below:




## Help and Feedback

You did not find what you were looking for?

Ask a question on the **Q&A forum**.
If you think something is missing or wrong in the documentation, please file a **bug report**.