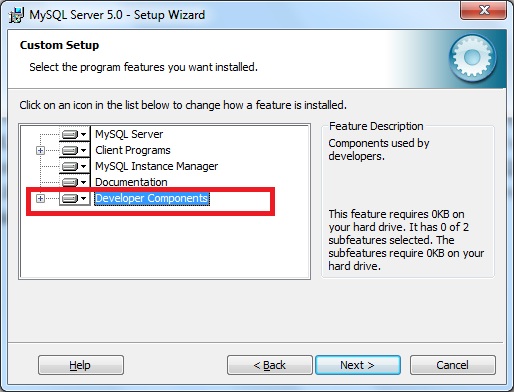
# Các cấu hình cần thiết

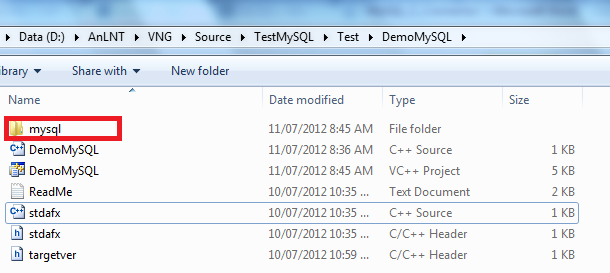
* Cài đặt components

Khi cài đặt MySQL ta có thể thêm các components chứa cái hàm API cung cho ngôn ngữ C cho phép truy cập hệ quản trị CSDL MySQL. Mặc định thì các components này không được chọn để cài đặt. Để biết trong máy chúng ta có các components này hay không thì ta vào thư mục cài đặt MySQL tìm 2 folder là include chứa cái file.h có dạng my\_???.h, mysql\_???.h và folder lib chứa libmysql.dll và libmysql.lib. Nếu không có thì chúng ta phải install lại MySQL và chọn Modify và chọn thêm Developer Components

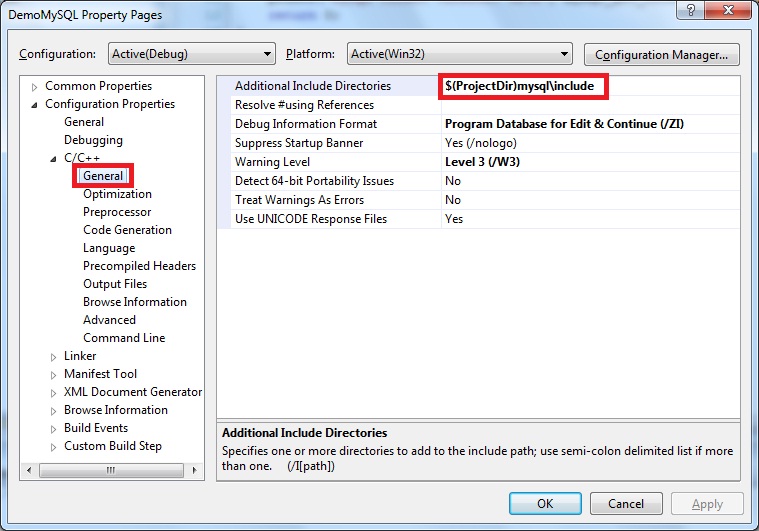


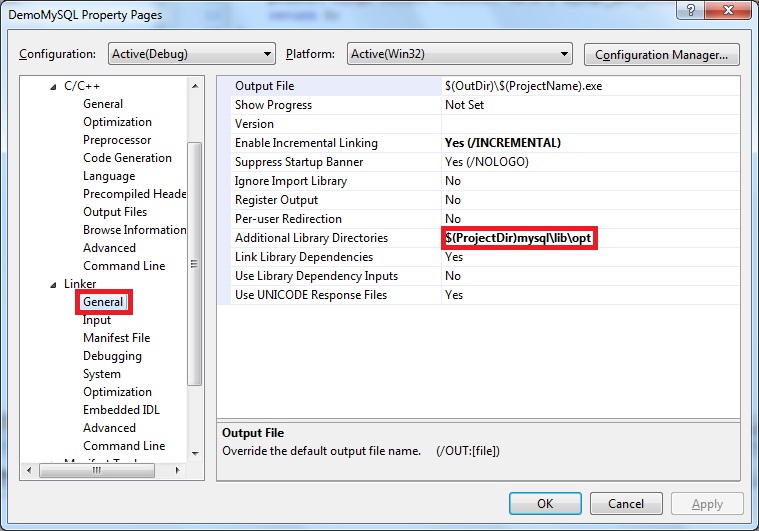
Sau khi thêm vào các components thì trong thư mục cài đặt MySQL sẽ có 2 thư mục là include và lib

* Trong thư mục root của project tạo một folder mysql, sao đó copy 2 folder include và lib ở trên vào folder mysql này



* Cấu hình cho project link đến 2 thư mục include và lib





* Thêm #pragma comment(lib, "libmysql") vào trước hàm main chương trình
* Vào folder **$(ProjectDir)**/lib/opt/ lấy libmysql.dll bỏ vào chung folder với file exe của project.
* Test thử chương trình sau

#include <my\_global.h>

#include <mysql.h>

#pragma comment(lib, "libmysql")

int \_tmain(int argc, \_TCHAR\* argv[])

{

printf("MySQL Client version: %s\n", mysql\_get\_client\_info());

return 0;

}

# C API Data Structures

MYSQL

Cấu trúc này như là một handle của một connection đến database. Hầu hết mọi hàm muốn thao tác với database đều phải thông qua cấu trúc này. Không copy cấu trúc này vào một biến khác để dùng. Không có sự đảm bảo biến copy lại có thể sử dụng được.

# C API Function

# mysql\_init

**MYSQL \*mysql\_init(MYSQL \*mysql)**

*Hàm này khởi tạo một đối tượng kiểu cấu trúc MYSQL dùng để kết nối đến database*

**Cách dùng**

* Cách 1 : truyền tham số là NULL và nhận kết quả là con trỏ đến đối tượng này.

MYSQL \*conn = mysql\_init(NULL);

* Cách 2 : Khai báo sẵn một biến kiểu cấu trúc MYSQL và truyền địa chỉ của biến đó cho tham số lúc gọi hàm

MYSQL mysql;

mysql\_init(&mysql);

**Giá trị trả về :**

* Thành công : con trỏ đến đối tượng MYSQL được tạo.
* Thất bại : NULL do không đủ bộ nhớ

**Lưu ý** : tất cả những đối tượng MYSQL nào được tạo từ hàm mysql\_init thì sau khi dùng xong phải gọi hàm mysql\_close để giải phóng tài nguyên.

# mysql\_close

**void mysql\_close(MYSQL \*mysql)**

*Hàm này giải phỏng tài nguyên được cấp cho đối tượng MYSQL*

**Cách dùng**

Truyền vào địa chỉ của đối tượng MYSQL muốn giải phóng tài nguyên

# Mysql\_real\_connect

**MYSQL \*mysql\_real\_connect(MYSQL \*mysql,**

**const char \*host,**

**const char \*user,**

**const char \*passwd,**

**const char \*db,**

**unsigned int port,**

**const char \*unix\_socket,**

**unsigned long client\_flag)**

*Hàm này thiết lập một kết nối dựa trên một cấu trúc MYSQL. Hàm này được gọi sau hàm mysql\_init và gọi trước tất cả các cấu query đến database*

**Tham số**

* Mysql : địa chỉ của biến cấu trúc MYSQL
* Host : tên máy hoặc địa chỉ IP của server. Nếu tham số host này bằng NULL hoặc “localhost” thì lần kết nối này sẽ connect tới máy tính cục bộ.
* User : user name
* Passwd : password
* Db : tên của database muốn kết nối tượng
* Port : nếu port = 0 thì sẽ chọn port mặt định củ MySQL. Nếu port khác 0 thì sẽ dùng port đó để kết nối.
* Unix\_socket : để NULL
* Client\_flag : các tùy chọn mở rộng. Ví dụ : giao thức nén dữ liệu, server cho phép gửi một lúc nhiều câu lệnh, …

**Giá trị trả về**

* Thành công : chính là tham số đầu tiên truyền vào hàm
* Thất bại : null

**Cách dùng**

MYSQL \*mySqlConnection = mysql\_init(NULL);

char strUserName[] = "root";

char strPassword[] = "hai";

char strServerAddress[] = "localhost";

char strDatabase[] = "quanlynhanvien";

if (NULL == mysql\_real\_connect(mySqlConnection, strServerAddress, strUserName,

strPassword, strDatabase, 0, NULL, 0)) {

printf("ket noi that bai");

return 1;

}