2 LibCurl 编程

2.1 LibCurl 编程流程

在基于 LibCurl 的程序里，主要采用 callback function（回调函数）的形式完成传输任务，用户在启动传输前设置好各类参数和回调函数，当满足条件时 libcurl 将调用用户的回调函数实现特定功能。下面是利用 libcurl 完成传输任务的流程：

1. 调用 curl_global_init()初始化 libcurl
2. 调用 curl_easy_init()函数得到 easy interface 型指针
3. 调用 curl_easy_setopt 设置传输选项
4. 根据 curl_easy_setopt 设置的传输选项，实现回调函数以完成用户特定任务
5. 调用 curl_easy_perform（）函数完成传输任务
6. 调用 curl_easy_cleanup（）释放内存

在整过过程中设置 curl_easy_setopt()参数是最关键的，几乎所有的 libcurl 程序都要使用它。

2.2 重要函数

1.CURLcode curl_global_init(long flags);

描述：

这个函数只能用一次。(其实在调用 curl_global_cleanup 函数后仍然可再用)

如果这个函数在 curl_easy_init 函数调用时还没调用，它讲由 libcurl 库自动完成。

参数：flags

CURL_GLOBAL_ALL　　//初始化所有的可能的调用。

CURL_GLOBAL_SSL　　//初始化支持 安全套接字层。

CURL_GLOBAL_WIN32　　//初始化 win32套接字库。

CURL_GLOBAL_NOTHING　　//没有额外的初始化。

2 void curl_global_cleanup(void);

描述：在结束 libcurl 使用的时候，用来对 curl_global_init 做的工作清理。类似于 close 的函数。

3 char *curl_version( );

描述:打印当前 libcurl 库的版本。

4 CURL *curl_easy_init( );

描述:

curl_easy_init 用来初始化一个 CURL 的指针(有些像返回 FILE 类型的指针一样).相应的在调用结束时要用 curl_easy_cleanup 函数清理.

一般 curl_easy_init 意味着一个会话的开始.它的返回值一般都用在 easy 系列的函数中.

5 void curl_easy_cleanup(CURL *handle);

描述:

这个调用用来结束一个会话.与 curl_easy_init 配合着用.

参数:

CURL 类型的指针.

6 CURLcode curl_easy_setopt(CURL *handle, CURLoption option, parameter);

描述:这个函数最重要了.几乎所有的 curl 程序都要频繁的使用它.它告诉 curl 库.程序将有如何的行为.比如要查看一个网页的 html 代码等.(这个函数有些像 ioctl 函数)参数:

1 CURL 类型的指针

2各种 CURLoption 类型的选项.(都在 curl.h 库里有定义,man 也可以查看到)

3 parameter 这个参数 既可以是个函数的指针,也可以是某个对象的指针,也可以是个 long 型的变量.它用什么这取决于第二个参数.

CURLoption 这个参数的取值很多.具体的可以查看 man 手册.

7 CURLcode curl_easy_perform(CURL *handle);描述:这个函数在初始化 CURL 类型的指针 以及 curl_easy_setopt 完成后调用.就像字面的意思所说 perform 就像是个舞台.让我们设置的

option 运作起来.参数:
CURL 类型的指针.

3.3 curl_easy_setopt 函数介绍

本节主要介绍 curl_easy_setopt 中跟 http 相关的参数。注意本节的阐述都是以 libcurl 作为主体，其它为客体来阐述的。

1. CURLOPT_URL
   设置访问 URL
2. CURLOPT_WRITEFUNCTION，CURLOPT_WRITEDATA
   回调函数原型为：**size_t function(void *ptr, size_t size, size_t nmemb, void *stream);**函数将在 libcurl 接收到数据后被调用，因此函数多做数据保存的功能，如处理下载文件。CURLOPT_WRITEDATA 用于表明 CURLOPT_WRITEFUNCTION 函数中的 stream 指针的来源。
3. CURLOPT_HEADERFUNCTION，CURLOPT_HEADERDATA
   回调函数原型为 size_t function(void *ptr, size_t size,size_t nmemb, void *stream); libcurl 一旦接收到 http 头部数据后将调用该函数。CURLOPT_WRITEDATA 传递指针给 libcurl，该指针表明 CURLOPT_HEADERFUNCTION 函数的 stream 指针的来源。
4. CURLOPT_READFUNCTION CURLOPT_READDATA
   libCurl 需要读取数据传递给远程主机时将调用 CURLOPT_READFUNCTION 指定的函数，函数原型是：size_t function(void *ptr, size_t size, size_t nmemb,void *stream). CURLOPT_READDATA 表明 CURLOPT_READFUNCTION 函数原型中的 stream 指针来源。
5. CURLOPT_NOPROGRESS，CURLOPT_PROGRESSFUNCTION，CURLOPT_PROGRESSDATA
   跟数据传输进度相关的参数。CURLOPT_PROGRESSFUNCTION 指定的函数正常情况下每秒被 libcurl 调用一次，为了使 CURLOPT_PROGRESSFUNCTION 被调用，CURLOPT_NOPROGRESS 必须被设置为 false，CURLOPT_PROGRESSDATA 指定的参数将作为 CURLOPT_PROGRESSFUNCTION 指定函数的第一个参数

6. CURLOPT_TIMEOUT，CURLOPT_CONNECTIONTIMEOUT:

CURLOPT_TIMEOUT 由于设置传输时间，

CURLOPT_CONNECTIONTIMEOUT 设置连接等待时间

7. CURLOPT_FOLLOWLOCATION

设置重定位 URL

CURLOPT_RANGE: CURLOPT_RESUME_FROM:

断点续传相关设置。CURLOPT_RANGE 指定 char *参数传递给 libcurl，用于指明 http 域的 RANGE 头域，例如：

表示头500个字节：bytes=0-499

表示第二个500字节：bytes=500-999

表示最后500个字节：bytes=-500

表示500字节以后的范围：bytes=500-

第一个和最后一个字节：bytes=0-0,-1

同时指定几个范围：bytes=500-600,601-999

CURLOPT_RESUME_FROM 传递一个 long 参数给 libcurl，指定你希望开始传递的

偏移量。

3.4 curl_easy_perform 函数说明（error 状态码）

该函数完成 curl_easy_setopt 指定的所有选项，本节重点介绍 curl_easy_perform 的返回值。返回0意味一切 ok，非0代表错误发生。主要错误码说明：

1. CURLE_OK

任务完成一切都好

2 CURLE_UNSUPPORTED_PROTOCOL

不支持的协议，由 URL 的头部指定

3 CURLE_COULDNT_CONNECT

不能连接到 remote 主机或者代理

4 CURLE_REMOTE_ACCESS_DENIED

访问被拒绝

5 CURLE_HTTP_RETURNED_ERROR

Http 返回错误

6 CURLE_READ_ERROR

读本地文件错误