<https://blog.csdn.net/u010727765/article/details/54341419>

libcurl中使用curl\_easy\_setopt()方法来设置libcurl工作过程中的选项，具体选项的设置和意义如下：

函数原型

**[cpp]** [view plain](https://blog.csdn.net/u010727765/article/details/54341419) [copy](https://blog.csdn.net/u010727765/article/details/54341419)

1. CURLcode curl\_easy\_setopt(CURL \*curl, CURLoption option,  parameter);

其中curl是在使用前初始化libcurl模块得到的可操作的句柄，option 就是要设置的选项， parameter就是选项所需要的参数

一下主要对option参数的相关情况进行介绍，仅仅是本人用到的几个选项，更多的选项在以后逐渐补充。

### 1、Behavior Options

* CURLOPT\_VERBOSE, CURLcode curl\_easy\_setopt(CURL \*curl, CURLOPT\_VERBOSE, long onoff);

如果将onoff设置为1，那么libcurl将打印出来很多冗长在curl上操作的信息，这个主要是为了调试和理解libcurl的工作方式，通常在产品中是关闭的。

### 2、Network Options

* CURLOPT\_NETWORK,  CURLcode curl\_easy\_setopt(CURL \*curl, CURLOPT\_URL, char\* url);

该函数将传递libcurl工作是需要的url，url必须是char\* 类型的字符串，并且以空字符结尾，同时还需要满足格式如：scheme://host:port/path。

但是libcurl在发送请求出现问题之前是不会使用这个url也不会对这个url进行检查，也就说，在调用以上函数的时候，即使你传入的url是一个非常怪异格式的url，  
函数的返回值仍然会是CURL\_OK。

如果传入的url中没有“http://”或者“ftp://”等等这样的头，那么libcurl会以当前的主机作为url的头。如果协议最深层次的子域名匹配了DICT,FTP,IMAP,LDAP  
,POP3或者是SMTP等等，那么这些协议就会被使用，否则的话libcurl只会使用http协议，但是在libcurl7.45.0之后，会考虑到是否设置一个默认的协议，这个以后再  
细说。

libcurl中使用的协议，要么是有url的头来指定，要么是从主机来推理，如果有libcurl不支持的协议，当之后在调用curl\_easy\_perform或者curl\_multi\_perform的时候  
会返回CURL\_UNSUPPORTED\_PROTOCAL ，如果要了解libcurl支持的协议，可以使用构建libcurl库生成的curl\_version\_info来查看。

CURLOPT\_URL是在libcurl传输之前唯一一个必须被设置的选项。

URL的主机部分要包含要连接的服务器的地址。这个地址可能是一个完成的服务器的域名，机器本地网络名称，或者是服务器主机的IP地址，还可以代表主机的IPV4或者  
IPV6的地址等等。

### 3、Callback Options

* CRULOPT\_HEADERFUNCTION, CURLcode curl\_easy\_setopt(CURL\* curl, CURLOPT\_HEADERFUNCTION, header\_callback);

header\_callback原型：size\_t header\_callback(char\* buffer, size\_t size, size\_t nitems, void\* userdata);

传递一个客制回调函数的指针，该回调函数必须和header\_callback函数原型一致。

在libcurl一接收到头数据的时候就会调用这个回调函数，这个回调函数会为每一个头数据都调用一次并且仅仅是一个完整的头数据传递给回调函数的时候，使用这个方法解

析头数据很简单。buffer指向的数据的长度就是size乘以nitems，不要认为头数据是一空字符结尾的。名为userdata的指针就是通过CURLOPT\_HEADERDATA这个选项来

设置的那个。

这个回调函数一定会返回实际接收到的字节数，如果返回的数量和传递给函数的数量不一致，那么便会为libcurl产生一个错误信号，这会导致传递失败，同时libcurl中的方

法会返回CURL\_WRITE\_ERROR。

被传到这个函数的一个完整的Http头可以达到CURL\_MAX\_HTTP\_HEADER（100K）字节。

如果这个选项没有被设置，或者是该回调函数设置为NULL，但是CURLOPT\_HEADERDATA已经被设置，而且不是设置为NULL，那么这个回到函数的执行就会被替代，

也就是说，执行函数将是在CURLOPT\_HEADERFUNCTION，如果这个函数没有指定或者设置为NULL，那么默认的，将会执行stream-writing 函数。

很重要的一点，这个回调函数会为初始化一个请求后的所有接收到的回复调用，而不仅仅是最有一个回复。包括了发生在认证授权中的所有回复。如果你仅仅需要对最有

一个回复进行操作，你需要自行在回调函数中收集所有的头并且使用http状态行，例如在将回复划界。

当服务器发送了一大块加密的传送文件，其中也许会包含一个尾部。这个尾部和http的头部一模一样并且这样的一个尾部也将会通过这个回调函数传递给应用程序。这里有

几种方法来发现这个尾部不是一个普通的头部：1).在回复消息主体之后; 2).来自最后一个头部行之后; 3).一个尾部，在所有普通回复头中会找到期望的一个尾部。

对于非http协议的协议例如：FTP、POP3、IMAP 和SMTP，这个方法会在服务器接收到libcurl的命令之后。

版权声明：本文为博主原创文章，未经博主允许不得转载。 https://blog.csdn.net/u010727765/article/details/54341419