tolimit

格物 致知 诚意 正心 修身 齐家 治国 平天下

lwfbibi

博客访问： 115972
博文数量： 31
博客积分： 10
博客等级： 民兵
技术积分： 1561
用 户 组： 普通用户
注册时间： 2012-03-09 00:51
认证徽章： 认证会员

加关注 短消息
论坛 加好友

个人简介

努力 奋斗 专注于linux内核 微信：tolimit 扣扣：348958453

文章分类

文章存档

我的朋友


书虫逍遥


CU官方博


伊Eva


Leesuna

linux - 使用curl实现新浪天气API应用

2015-01-26 09:37:18

分类： LINUX

新浪天气API使用方法

API地址:http://php.weather.sina.com.cn/xml.php?city=%B1%B1%BE%A9&password=DJOYnieT8234jlsK&day=0

红色标记为城市代码(也就是城市的中文转为GB2312的十六进制代码,比如北京对应的GB2312十六进制代码为B1B1BEA9),实际上需要查哪个城市就把红色标记改为对应城市代码即可.而实际上打开此url后对应的是一个xml文件,里面包括了此城市的天气信息.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- published at 2014-10-31 17:08:03 -->
<Profiles>
<Weather>
<city>北京</city>
<status1>小雨</status1>
<status2>多云</status2>
<figure1>xiaoyu</figure1>
<figure2>duoyun</figure2>
<direction1>无持续风向</direction1>
<direction2>无持续风向</direction2>
<power1>≤3</power1>
<power2>≤3</power2>
<temperature1>14</temperature1>
<temperature2>9</temperature2>
<ssd>5</ssd>
<tgd1>16</tgd1>
<tgd2>16</tgd2>
<zwx>2</zwx>
<ktk>6</ktk>
<pollution>3</pollution>
<xcz></xcz>
<zho></zho>
<diy></diy>
<fas></fas>
<chy>4</chy>
<zho_shuoming>暂无</zho_shuoming>
<diy_shuoming>暂无</diy_shuoming>
<fas_shuoming>暂无</fas_shuoming>
<chy_shuoming>套装、夹衣、风衣、夹克衫、西服套装、马甲衬衫配长裤</chy_shuoming>
<pollution_l>轻度</pollution_l>
<zwx_l>弱</zwx_l>
<ssd_l>略凉</ssd_l>
<fas_l>暂无</fas_l>
<zho_l>暂无</zho_l>
<chy_l>夹衣类</chy_l>
<ktk_l>适宜开启(制热)</ktk_l>
<xcz_l>暂无</xcz_l>
<diy_l>暂无</diy_l>
<pollution_s>对空气污染物扩散无明显影响</pollution_s>
<zwx_s>紫外线弱</zwx_s>
<ssd_s>感觉有些凉，但是凉意微薄，不影响户外活动的开展。</ssd_s>
<ktk_s>适宜开启空调</ktk_s>
<xcz_s>暂无</xcz_s>
<gm>2</gm>
<gm_l>易发期</gm_l>
<gm_s>天气很凉，季节转换的气候，慎重增加衣服；较易引起感冒；</gm_s>
<yd>5</yd>
<yd_l>不适宜</yd_l>
<yd_s>虽然晴空万里，但是天气较凉，多数人不适宜户外运动；</yd_s>
<savedate_weather>2014-10-31</savedate_weather>
<savedate_life>2014-10-31</savedate_life>
<savedate_zhishu>2014-10-31</savedate_zhishu>
</Weather>
</Profiles>
```

现在我们的工作就是使用curl打开这个xml文件并且将其转换为我们需要的数据结构.

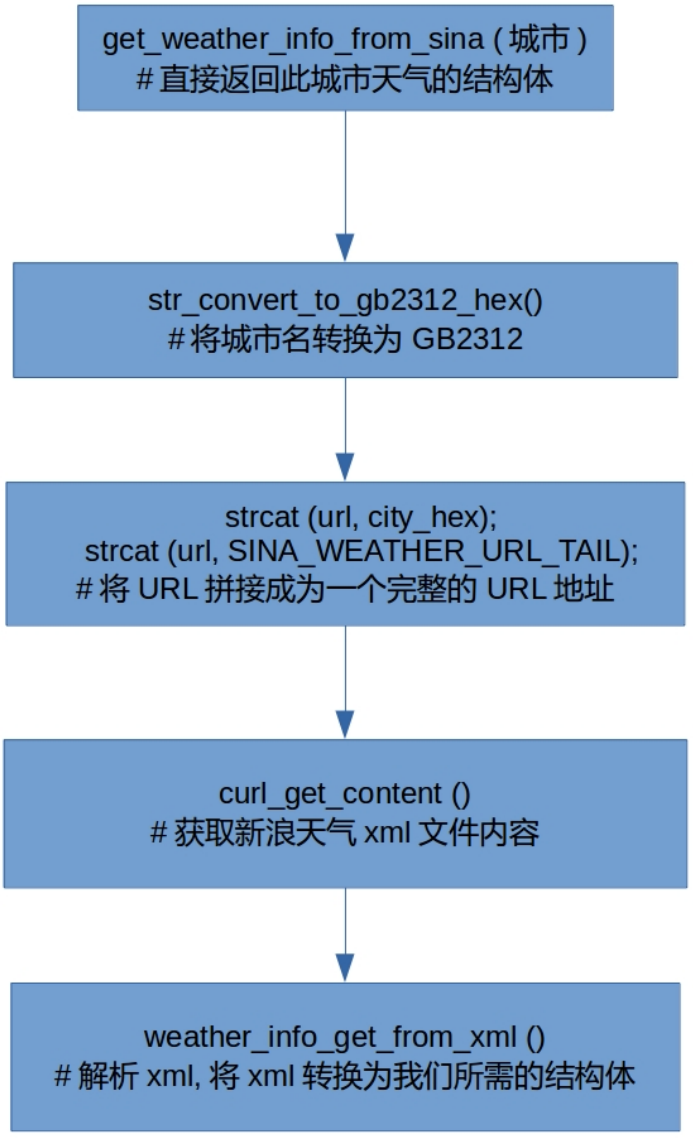
先看看我们自定义的数据结构,用于存放天气信息(感觉还有更好的定义方法):

```
#ifndef __SINA_WEATHER__
#define __SINA_WEATHER__

#include <stdio.h>

#define FALSE (0)
#define TRUE (!0)

#define SINA_WEATHER_URL_HEAD "http://php.weather.sina.com.cn/xml.php?city="
#define SINA_WEATHER_URL_TAIL "&password=DJOYnieT8234jlsK&day=0"

struct date {
    short year;
    short month;
    short day;
};

struct weather_info {
    char * city;

    //天气情况(中文)
    char * status1;
    char * status2;

    //天气情况(拼音)
    char * figure1;
    char * figure2;

    //风向
    char * direction1;
    char * direction2;

    //风级
    char * power1;
    char * power2;

    //温度
    int temperature1;
    int temperature2;

    //体感指数数值
    int ssd;
    //体感度指数
    char * ssd_l;
    //体感度指数说明
    char * ssd_s;

    //体感温度
    int tgd1;
    int tgd2;

    //紫外线指数数值
    int zwx;
    //紫外线指数
    char * zwx_l;
    //紫外线指数说明
    char * zwx_s;

    //空调指数数值
    int ktk;
    //空调指数
    char * ktk_l;
    //空调指数说明
    char * ktk_s;

    //污染指数数值
    int pollution;
    //污染扩散条件
    char * pollution_l;
    //污染指数说明
    char * pollution_s;

    //穿衣指数数值
    int chy;
    //穿衣指数
    char * chy_l;
    //穿衣说明
    char * chy_shuoming;

    //洗车指数数值
    int xcz;
    //洗车指数
    char * xcz_l;
    //洗车指数说明
    char * xcz_s;
```

```
86.
87.
88.       //感冒指数数值
89.       int gm;
90.       //感冒指数
91.       char * gm_l;
92.       //感冒指数说明
93.       char * gm_s;
94.
95.       //运动指数数值
96.       int yd;
97.       //运动指数
98.       char * yd_l;
99.       //运动指数说明
100.      char * yd_s;
101.
102.      struct date weather_date;
103.      struct date life_date;
104.      struct date zhishu_date;
105.  };
106.
107.  struct weather_info * get_weather_info_from_sina (const char * city_name);
108.
109.  #endif
```

先说一下流程,首先设计想法是就定义一个接口,调用时只需要输入一个城市名称,则自动返回此城市的天气信息结构体,而在内部,分别执行了UTF-8转GB2312格式, curl初始化, 获取天气XML信息, 填充struct weather_info结构体.

流程图:



代码(sina_weather.c):

```
#include <stdio.h>
#include <curl/curl.h>
#include <iconv.h>

#include "sina_weather.h"

//根据key获取xml中对应的值
char * get_xml_key_value (const char * xml, char * key)
{
    char * head;
    char * tail;
    char stamp[1024];
    char * value;

    sprintf(stamp, "<%s>", key);
    if ((head = strstr (xml, stamp)) == NULL)
        return NULL;

    head = head + strlen (stamp);
    while (isspace (head[0]))
        head ++;

    sprintf(stamp, "", key);

    if ((tail = strstr (head, stamp)) == NULL)
        return NULL;
```

```
27.
28.     while (isspace(tail[-1]) && (tail > head))
29.         tail --;
30.
31.     if (tail > head){
32.         value = calloc (tail - head + 1, 1);
33.         memcpy (value, head, tail - head);
34.         value[tail - head] = 0x00;
35.     }
36.
37.     return value;
38. }
39.
40.
41. //UTF-8转GB2312
42. int utf8_to_gb2312 (const char * src, char * dest_buf)
43. {
44.     iconv_t con;
45.     int src_length = strlen (src);
46.
47.     if( (con = iconv_open ("gb2312", "utf-8")) ==0 )
48.           return FALSE;
49.
50.     memset (dest_buf, 0, src_length);
51.
52.     if(-1 == iconv (con, &src, &src_length, &dest_buf, &src_length))
53.          return FALSE;
54.     iconv_close (con);
55.
56.     return TRUE;
57. }
58.
59. //将一个GB2312字符串转换为它的十六进制字符串(每个字符用%隔开)
60. char * str_convert_to_gb2312_hex (const char * src)
61. {
62.     if (src == NULL)
63.         return NULL;
64.
65.     int length = strlen (src) - 2;
66.     int index = 0;
67.     int num = 0;
68.     char * gb_str = calloc (((length) + 1), 1);
69.     char * hex_str = calloc ((length * 3) + 1, 1);
70.
71.     utf8_to_gb2312 (src, gb_str);
72.     while ((index / 3) < length) {
73.         hex_str[index] = '%';
74.         sprintf (&hex_str[index + 1], "%02X", gb_str[index / 3] & 0xff);
75.         index = index + 3;
76.     }
77.     hex_str[length * 3] = '\0';
78.     return hex_str;
79. }
80.
81. void convert_to_gb2312_clean (char * str)
82. {
83.     free (str);
84. }
85.
86. //curl下载时的回调函数,当下载到数据时会调用此函数进行操作
87. size_t down_data_callback (void * ptr, size_t size, size_t nmemb, void * user_buf)
88. {
89.     strcat (user_buf, ptr);
90.     return size * nmemb;
91. }
92.
93. uint64_t curl_get_content (const char * url, char * content)
94. {
95.     if ((url == NULL) || (content == NULL))
96.         return;
97.
98.     CURL * curl = curl_easy_init ();
99.     curl_easy_setopt (curl, CURLOPT_URL, url); //设置curl的目标url地址
100.     curl_easy_setopt (curl, CURLOPT_TIMEOUT, 15); //下载超时时间
101.     curl_easy_setopt (curl, CURLOPT_NOSIGNAL, 1); //屏蔽其他信号
102.     curl_easy_setopt (curl, CURLOPT_WRITEFUNCTION, down_data_callback); //设置下载回调函数
103.     curl_easy_setopt (curl, CURLOPT_WRITEDATA, content); //设置下载数据保存缓冲区(此参数会传到down_data
    _callback的user_buf)
104.
105.     CURLcode retval = curl_easy_perform (curl);
106.     if (retval == CURLE_OK) {
107.         double down_length = 0;
108.         curl_easy_getinfo (curl, CURLINFO_CONTENT_LENGTH_DOWNLOAD, &down_length);
109.         curl_easy_cleanup (curl);
110.         return (uint64_t)down_length;
111.     }
112.     curl_easy_cleanup (curl);
113.     return 0;
114. }
115.
116. struct weather_info * weather_info_get_from_xml (const char * xml, struct weather_info * w_info)
117. {
118.     if ((xml == NULL) || (w_info == NULL))
119.         return w_info;
120.
121.     char * str_data = NULL;
122.
```

```
123.      w_info->city = get_xml_key_value (xml, "city");
124.      w_info->status1 = get_xml_key_value (xml, "status1");
125.      w_info->status1 = get_xml_key_value (xml, "status1");
126.      str_data = get_xml_key_value (xml, "temperature1");
127.      w_info->temperature1 = (int) atoi (str_data);
128.      free (str_data);
129.      str_data = get_xml_key_value (xml, "temperature2");
130.      w_info->temperature2 = (int) atoi (str_data);
131.      free (str_data);
132.      //此处省略N多,基本都同上操作.
133.  }
134.
135.  struct weather_info * get_weather_info_from_sina (const char * city_name)
136.  {
137.      if (city_name == NULL)
138.          return NULL;
139.
140.      CURL * curl = NULL;
141.      CURLcode retval = 0;
142.      char xml_data[4096];
143.      char url[1024] = SINA_WEATHER_URL_HEAD;
144.      char * city_hex = str_convert_to_gb2312_hex (city_name);
145.      struct weather_info * sina_w_info = calloc (sizeof (struct weather_info), 1);
146.
147.      strcat (url, city_hex);
148.      strcat (url, SINA_WEATHER_URL_TAIL);
149.
150.      convert_to_gb2312_clean (city_hex);
151.
152.      uint64_t down_size = curl_get_content (url, xml_data);
153.
154.      if (down_size == 0) {
155.          free (sina_w_info);
156.          return NULL;
157.      }
158.
159.      weather_info_get_from_xml (xml_data, sina_w_info);
160.      return sina_w_info;
161.  }
```

可以看出所有的代码封装成了一个`struct weather_info * get_weather_info_from_sina (const char * city_name)`，当需要使用使用时直接调用此函数即返回城市天气结构体。

CURL:

或许有些同学不太清楚怎么在c中使用curl库,其实使用的方法很简单,就简单说明一下,在ubuntu下安装curl库命令如下

```
# apt-get install libcurl-nss-dev
```

在程序编译时需要加入curl库的选项:

```
-lcurl
```

使用curl下载时常规的代码顺序为:

```
1.  CURL * curl = curl_easy_init (); //初始化
2.
3.      curl_easy_setopt (curl, CURLOPT_URL, url); //具体设置代表的意思在http://curl.haxx.se/libcurl/c/curl_easy_setopt.html
4.      curl_easy_setopt (curl, CURLOPT_TIMEOUT, 15);
5.      curl_easy_setopt (curl, CURLOPT_NOSIGNAL, 1);
6.      curl_easy_setopt (curl, CURLOPT_WRITEFUNCTION, down_data_callback);
7.      curl_easy_setopt (curl, CURLOPT_WRITEDATA, content);
8.
9.      CURLcode retval = curl_easy_perform (curl); //执行操作
10.     if (retval == CURLE_OK) {
11.        //添加所需要处理的代码
12.        ........
13.     }
14.     curl_easy_cleanup (curl); //清空
```

在使用curl下载时,会设置CURLOPT_WRITEFUNCTION,此设置主要是为了在下载有数据过来时,调用down_data_callback这个回调函数,并将数据保存到content这个缓冲区中.

```
1.  curl_easy_setopt (curl, CURLOPT_WRITEFUNCTION, down_data_callback);
2.  curl_easy_setopt (curl, CURLOPT_WRITEDATA, content);
3.
4.  size_t down_data_callback (void * ptr, size_t size, size_t nmemb, void * user_buf)
5.  {
```

```
6.      strcat (user_buf, ptr);
7.      return size * nmemb;
8.  }
```

小结

curl作为一个C的下载库还是非常方便使用的，其用法很简单，还能够支持断点续传，程序需要用到下载功能的同学可以试试curl库，而现在网上许多公共API接口形式都是与新浪天气API类似，比如google地球的API,百度地图API等都与它类似，这里只做一个入门，深入的话可以以此作为一个入口进行学习。

附录**(1**代表白天**,2**代表晚上**):**

	天气情况中文
	天气情况拼音
	风向
	风级
	温度
	体感指数数值
	体感度指数
	体感度指数说明
	体感温度
	紫外线指数数值
	紫外线指数
	紫外线指数说明
	空调指数数值
	空调指数
	空调指数说明
	污染指数数值
	污染物扩散条件
	污染指数说明
	洗车指数数值
	洗车指数
	洗车指数说明
	穿衣指数数值
	穿衣指数
	穿衣说明
	感冒指数数值
	感冒指数
	感冒指数说明
	运动指数数值
	运动指数
	运动指数说明

	天气预报日期 生活日期 指数日期

上一篇：关于for,while,dowhile效率测试
下一篇：关于linux系统如何实现fork的研究(一)

相关热门文章

linux 常见服务端口	linux dhcp peizhi roc
xmanager 2.0 for linux配置	关于Unix文件的软链接
【ROOTFS搭建】busybox的httpd...	求教这个命令什么意思，我是新...
openwrt中luci学习笔记	sed -e "/grep/d" 是什么意思...
什么是shell	谁能够帮我解决LINUX 2.6 10...

给主人留下些什么吧！~~

评论热议

请登录后评论。
登录 注册