<https://www.cnblogs.com/liunianshiwei/p/6087596.html>

JSON概述

　　JSON： JavaScript 对象表示法（ JavaScript Object Notation） 。是一种轻量级的数据交换格式。 它基于ECMAScript的一个子集。 JSON采用完全独立于语言的文本格式， 但是也使用了类似于C语言家族的习惯（ 包括C、 C++、 C#、 Java、 JavaScript、 Perl、 Python等） 。这些特性使JSON成为理想的数据交换语言。 易于人阅读和编写， 同时也易于机器解析和生成(一般用于提升网络传输速率)。  
　　JSON 解析器和 JSON 库支持许多不同的编程语言。 JSON 文本格式在语法上与创建 JavaScript 对象的代码相同。 由于这种相似性， 无需解析器， JavaScript 程序能够使用内建的 eval() 函数， 用 JSON 数据来生成原生的 JavaScript 对象。  
　　JSON 是存储和交换文本信息的语法。 类似 XML。 JSON 比 XML 更小、 更快， 更易解析。  
　　JSON 具有自我描述性， 语法简洁， 易于理解。

JSON语法说明

　　先来看一个简单的JSON

[复制代码](javascript:void(0);)

1 {

2 "stars": [

3 {

4 "name": "Faye",

5 "address": "北京"

6 },

7 {

8 "name": "andy",

9 "address": "香港"

10 },

11 {

12 "name": "eddie",

13 "address": "台湾"

14 },

15

16 ]

17 }

[复制代码](javascript:void(0);)

　　　　JSON 语法是 JavaScript 对象表示法语法的子集。数据在键/值对中；数据由逗号分隔；花括号保存对象， 也称一个文档对象；方括号保存数组， 每个数组成员用逗号隔开， 并且每个数组成员可以是文档对象或者数组或者键值对 。

JSON基于两种结构：  
　　　　“名称/值”对的集合（A collection of name/value pairs）。不同的编程语言中，它被理解为对象（object），纪录（record），结构（struct），字（dictionary），哈希表（hashtable），有键列表（keyed list），或者关联数组 （associative array）。  
　　　　值的有序列表（An ordered list of values）。在大部分语言中，它被实现为数组（array），矢量（vector），列表（list），序列（sequence）。

JSON的三种语法：

　　　　键/值对 key:value，用半角冒号分割。 比如 "name":"Faye"   
　　　　文档对象 JSON对象写在花括号中，可以包含多个键/值对。比如{ "name":"Faye" ,"address":"北京" }。   
　　　　数组 JSON 数组在方括号中书写： 数组成员可以是对象，值，也可以是数组(只要有意义)。 {"love": ["乒乓球","高尔夫","斯诺克","羽毛球","LOL","撩妹"]}

附cJSON库下载地址 https://github.com/DaveGamble/cJSON

cJSON库在使用的时候只需要如下两步:将cJSON.c(或者库文件) 和 cJSON.h添加到项目中即可;如果在命令行中进行链接 还需要加上-lm 表示链接math库 .

C语言函数库写JSON文件 ：

　　　　从缓冲区中解析出JSON结构：extern cJSON \*cJSON\_Parse(const char \*value);   
　　　　　　　　解析一块JSON数据返回cJSON结构， 在使用完之后调用cJSON\_Delete函数释放json对象结构。

　　　　将传入的JSON结构转化为字符串 ：extern char \*cJSON\_Print(cJSON \*item);   
　　　　　　　　可用于输出到输出设备， 使用完之后free(char \*) 。

　　　　将JSON结构所占用的数据空间释放 ：void cJSON\_Delete(cJSON \*c)

　　　　创建一个值类型的数据 ：extern cJSON \*cJSON\_CreateNumber(double num);  
　　　　　　　　　　　　　　　 extern cJSON \*cJSON\_CreateString(const char \*string);  
　　　　　　　　　　　　　　　 extern cJSON \*cJSON\_CreateArray(void);

　　　　创建一个对象（文档） ：extern cJSON \*cJSON\_CreateObject(void);

　　　　数组创建以及添加 ：cJSON \*cJSON\_CreateIntArray(const int \*numbers,int count);  
　　　　　　　　　　　　　 void cJSON\_AddItemToArray(cJSON \*array, cJSON \*item);

　　　　JSON嵌套 ：

　　　　　　　　　　【 向对象中增加键值对】 cJSON\_AddItemToObject(root, "rows", 值类型数据相关函数());  
　　　　　　　　　　【 向对象中增加数组】 cJSON\_AddItemToObject(root, "rows", cJSON\_CreateArray());  
　　　　　　　　　　【 向数组中增加对象】 cJSON\_AddItemToArray(rows, cJSON\_CreateObject());

　　　　几个能提高操作效率的宏函数 ：#define cJSON\_AddNumberToObject(object,name,n) \  
　　　　　　　　　　　　　　　　　　　　　　  cJSON\_AddItemToObject(object, name,cJSON\_CreateNumber(n))  
　　　　　　　　　　　　　　　　　　 #define cJSON\_AddStringToObject(object,name,s)\  
　　　　　　　　　　　　　　　　　　　　　　  cJSON\_AddItemToObject(object, name, cJSON\_CreateString(s))

C语言库函数解析JSON文件 ：

　　　　根据键找json结点 ：extern cJSON \*cJSON\_GetObjectItem(cJSON \*object,const char \*string)

　　　　判断是否有key是string的项 如果有返回1 否则返回0 ：extern int cJSON\_HasObjectItem(cJSON \*object,const char \*string)  
　　　　　　　　　　　　　　　　　　　　　　{ return cJSON\_GetObjectItem(object,string)?1:0; }

　　　　返回数组结点array中成员的个数 ：extern int cJSON\_GetArraySize(cJSON \*array);

　　　　根据数组下标index取array数组结点的第index个成员 返回该成员节点 ：extern cJSON \*cJSON\_GetArrayItem(cJSON \*array,int index);

　　　　遍历数组 ：#define cJSON\_ArrayForEach(pos, head)        for(pos = (head)->child; pos != NULL; pos = pos->next)

我们先来小试牛刀，编写一个输出到屏幕的简单json结构，代码1如下：

[复制代码](javascript:void(0);)

1 #include<stdio.h>

2 #include<stdlib.h>

3 #include"cJSON.h"

4

5 int main(void)

6 {

7 char \*data = "{\"love\":[\"LOL\",\"Go shopping\"]}";

8 //从缓冲区中解析出JSON结构

9 cJSON \* json= cJSON\_Parse(data);

10

11 //将传入的JSON结构转化为字符串 并打印

12 char \*json\_data = NULL;

13 printf("data:%s\n",json\_data = cJSON\_Print(json));

14

15 free(json\_data);

16 //将JSON结构所占用的数据空间释放

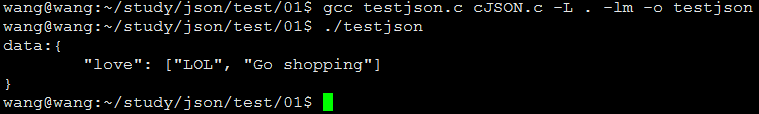
17 cJSON\_Delete(json);

18 return 0;

19 }

[复制代码](javascript:void(0);)

如下所示编译程序，执行程序，可以看到屏幕输出一个简单的json结构



下面我们在改进一下程序，输出到屏幕的同时，生成对应的json文件，代码2如下：

[复制代码](javascript:void(0);)

1 #include<stdio.h>

2 #include<stdlib.h>

3 #include<string.h>

4 #include"cJSON.h"

5

6 int main(void)

7 {

8 char \*char\_json = "{\"habit\":\"lol\"}";

9 //从缓冲区中解析出JSON结构

10 cJSON \*json = cJSON\_Parse(char\_json);

11 //将传入的JSON结构转化为字符串 并打印

12 char \*buf = NULL;

13 printf("data:%s\n",buf = cJSON\_Print(json));

14 //打开一个exec.json文件，并写入json内容

15 FILE \*fp = fopen("exec.json","w");

16 fwrite(buf,strlen(buf),1,fp);

17

18 fclose(fp);

19 free(buf);

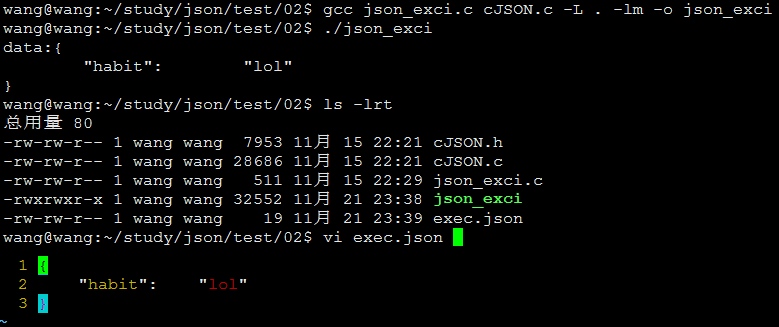
20 cJSON\_Delete(json);

21 return 0;

22 }

[复制代码](javascript:void(0);)

如下图所示进行编译，执行程序，查看生成的对应的json文件



下面我们生成一个生成一个json结构稍微复杂一点的json文件，代码3如下：

[复制代码](javascript:void(0);)

1 #include<stdio.h>

2 #include<stdlib.h>

3 #include<string.h>

4 #include"cJSON.h"

5

6 int main(void)

7 {

8 //创建一个空的文档（对象）

9 cJSON \*json = cJSON\_CreateObject();

10

11 //向文档中增加一个键值对{"name":"王大锤"}

12 cJSON\_AddItemToObject(json,"name",cJSON\_CreateString("王大锤"));

13 //向文档中添加一个键值对

14 //cJSON\_AddItemToObject(json,"age",cJSON\_CreateNumber(29));

15 cJSON\_AddNumberToObject(json,"age",29);

16

17 cJSON \*array = NULL;

18 cJSON\_AddItemToObject(json,"love",array=cJSON\_CreateArray());

19 cJSON\_AddItemToArray(array,cJSON\_CreateString("LOL"));

20 cJSON\_AddItemToArray(array,cJSON\_CreateString("NBA"));

21 cJSON\_AddItemToArray(array,cJSON\_CreateString("Go shopping"));

22

23 cJSON\_AddNumberToObject(json,"score",59);

24 cJSON\_AddStringToObject(json,"address","beijing");

25

26 //将json结构格式化到缓冲区

27 char \*buf = cJSON\_Print(json);

28 //打开文件写入json内容

29 FILE \*fp = fopen("create.json","w");

30 fwrite(buf,strlen(buf),1,fp);

31 free(buf);

32 fclose(fp);

33 //释放json结构所占用的内存

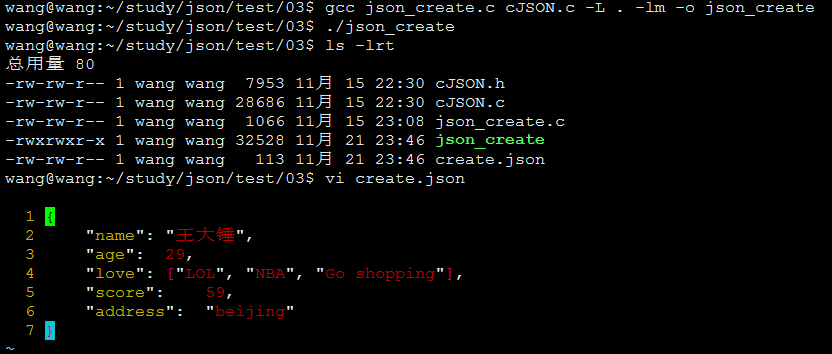
34 cJSON\_Delete(json);

35 return 0;

36 }

[复制代码](javascript:void(0);)

编译程序，执行程序，查看生成的json文件



下面再来编写一个，代码4如下：

[复制代码](javascript:void(0);)

1 #include<stdio.h>

2 #include<string.h>

3 #include<stdlib.h>

4 #include"cJSON.h"

5

6 int main(void)

7 {

8 //先创建空对象

9 cJSON \*json = cJSON\_CreateObject();

10 //在对象上添加键值对

11 cJSON\_AddStringToObject(json,"country","china");

12 //添加数组

13 cJSON \*array = NULL;

14 cJSON\_AddItemToObject(json,"stars",array=cJSON\_CreateArray());

15 //在数组上添加对象

16 cJSON \*obj = NULL;

17 cJSON\_AddItemToArray(array,obj=cJSON\_CreateObject());

18 cJSON\_AddItemToObject(obj,"name",cJSON\_CreateString("Faye"));

19 cJSON\_AddStringToObject(obj,"address","beijing");

20 //在对象上添加键值对

21 cJSON\_AddItemToArray(array,obj=cJSON\_CreateObject());

22 cJSON\_AddItemToObject(obj,"name",cJSON\_CreateString("andy"));

23 cJSON\_AddItemToObject(obj,"address",cJSON\_CreateString("HK"));

24

25 cJSON\_AddItemToArray(array,obj=cJSON\_CreateObject());

26 cJSON\_AddStringToObject(obj,"name","eddie");

27 cJSON\_AddStringToObject(obj,"address","TaiWan");

28

29 //清理工作

30 FILE \*fp = fopen("create.json","w");

31 char \*buf = cJSON\_Print(json);

32 fwrite(buf,strlen(buf),1,fp);

33 fclose(fp);

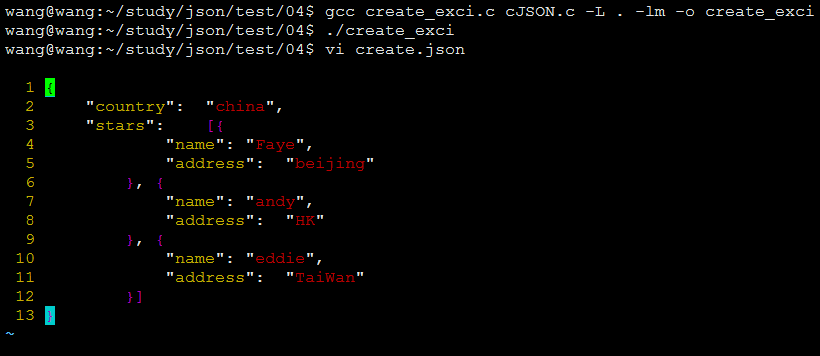
34 cJSON\_Delete(json);

35 return 0;

36 }

[复制代码](javascript:void(0);)

编译执行程序，效果如下图所示：



写了这么多生成xml的例子，下面我们来一个解析的例子，代码5如下：

[复制代码](javascript:void(0);)

1 #include<stdio.h>

2 #include<string.h>

3 #include<stdlib.h>

4 #include"cJSON.h"

5

6 int main(void)

7 {

8 char \*string = "{\"family\":[\"father\",\"mother\",\"brother\",\"sister\",\"somebody\"]}";

9 //从缓冲区中解析出JSON结构

10 cJSON \*json = cJSON\_Parse(string);

11 cJSON \*node = NULL;

12 //cJOSN\_GetObjectItem 根据key来查找json节点 若果有返回非空

13 node = cJSON\_GetObjectItem(json,"family");

14 if(node == NULL)

15 {

16 printf("family node == NULL\n");

17 }

18 else

19 {

20 printf("found family node\n");

21 }

22 node = cJSON\_GetObjectItem(json,"famil");

23 if(node == NULL)

24 {

25 printf("famil node == NULL\n");

26 }

27 else

28 {

29 printf("found famil node\n");

30 }

31 //判断是否有key是string的项 如果有返回1 否则返回0

32 if(1 == cJSON\_HasObjectItem(json,"family"))

33 {

34 printf("found family node\n");

35 }

36 else

37 {

38 printf("not found family node\n");

39 }

40 if(1 == cJSON\_HasObjectItem(json,"famil"))

41 {

42 printf("found famil node\n");

43 }

44 else

45 {

46 printf("not found famil node\n");

47 }

48

49 node = cJSON\_GetObjectItem(json,"family");

50 if(node->type == cJSON\_Array)

51 {

52 printf("array size is %d",cJSON\_GetArraySize(node));

53 }

54 //非array类型的node 被当做array获取size的大小是未定义的行为 不要使用

55 cJSON \*tnode = NULL;

56 int size = cJSON\_GetArraySize(node);

57 int i;

58 for(i=0;i<size;i++)

59 {

60 tnode = cJSON\_GetArrayItem(node,i);

61 if(tnode->type == cJSON\_String)

62 {

63 printf("value[%d]:%s\n",i,tnode->valuestring);

64 }

65 else

66 {

67 printf("node' type is not string\n");

68 }

69 }

70

71 cJSON\_ArrayForEach(tnode,node)

72 {

73 if(tnode->type == cJSON\_String)

74 {

75 printf("int forEach: vale:%s\n",tnode->valuestring);

76 }

77 else

78 {

79 printf("node's type is not string\n");

80 }

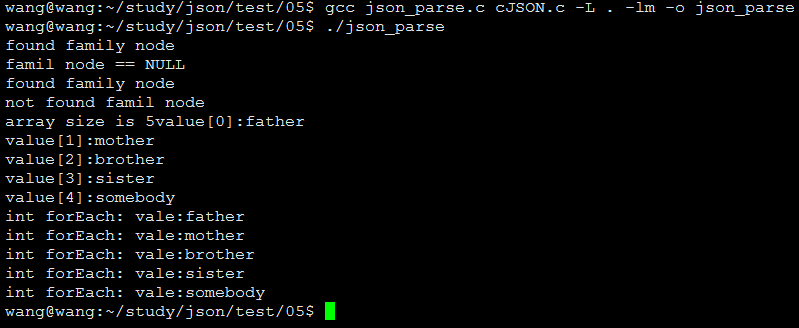
81 }

82 return 0;

83 }

[复制代码](javascript:void(0);)

编译程序，执行程序，解析出json信息如下图所示，



关于更多的json学习，可以自己根据提供的cJSON.h cJSON.c进行深入的学习，学无止境，我这里就介绍到这里了。

努力不一定有结果，有结果不一定是努力