1. cJSON \*cJSON\_CreateObject(void); //创建JSON对象
2. cJSON \*cJSON\_CreateArray(void); //创建JSON数组
3. void cJSON\_AddItemToObject(cJSON \*object,const char \*string,cJSON \*item);

//向对象中添加

1. void cJSON\_AddItemToArray(cJSON \*array, cJSON \*item);

//向数组中添加

1. cJSON\_AddItemToObject(root, "rows", rows = cJSON\_CreateArray());

//向对象中增加数组

1. cJSON\_AddItemToArray(rows, row = cJSON\_CreateObject());

//向数组中增加对象

1. cJSON\_AddItemToObject(root, "value", cJSON\_CreateNumber(value));

//向对象中增加数字

cJSON\_AddItemToObject(root, "string", cJSON\_CreateString(string));

//向对象中增加文件

1. char\* out = cJSON\_Print(root); //转为字符串

cJSON\_Delete(root); //释放

1. 简单说明：
2. char\* name = “123456”;

cJSON\_AddItemToObject(root, "name", cJSON\_CreateString(name));

cJSON\_AddStringToObject(root, "name",name);

以上两个效果完全相同

1. f

通讯录

1. Cjson\* root = cJSON\_Parse(char\* out);

负责解析JSON数据包，并按照cJSON结构体的结构序列化整个数据包。使用该函数会通过malloc函数在内存中开辟一个空间，使用完成需要手动释放。

1. cJSON\_GetArraySize(cJSON \*array);

//返回json数组大小。

1. cJSON \*cJSON\_GetObjectItem(cJSON \*object,const char \*string);

//返回指定字符串对应的json对象

1. cJSON \*cJSON\_GetArrayItem(cJSON \*array,int item);

//返回json数组中指定位置对象。

1. 简单说明：
2. **int** main (**int** argc, **const** **char** \* argv[])
3. {
4. // 创建JSON Object
5. cJSON \*root = cJSON\_CreateObject();
6. // 加入节点（键值对），节点名称为value，节点值为123.4
7. cJSON\_AddNumberToObject(root,"value",123.4);
8. // 打印JSON数据包
9. **char** \*out = cJSON\_Print(root);
10. printf("%s\n",out);
11. // 释放内存
12. cJSON\_Delete(root);
13. free(out);   //使用free函数释放被out占用的内存空间
14. **return** 0;
15. }