数据通信格式XML和JSON

学习目的

> 学习XML、HTML文件基本构成。
> 理解XML、JSON和HTML的文件结构，对于学习以及工作中生成和解析XML数据和HTML静态网页、解析JSON打下基础。

能够熟练的使用C语言库解析XML和JSON文件。

XML

历史演化

```
GML
```

在20世纪60年代为了促进数据交换和操作，通过IBM公司研究人员的杰出工作，得出了重要的结论：要提高系统的移植性，必须采用一种通用的文档格式，这种文档的格式必须遵守特定的规则。这也就是创建通用标记语言（外语全称：Generalized Markup Language、外语缩写：GML）的指导原则，从人们所产生的将文件结构化为标准的格式的动机出发，IBM创建了GML。

GML是一种IBM格式化文档语言，用于就其组织结构、各部件及其之间的关系进行文档描述。GML将这些描述标记为章节、重要小节和次重要小节（通过标题的级来区分）、段落、列表、表等。GML在文档具体格式方面，为文档员提供了一些方便，他们不必再为IBM得打印机格式化语言SCRIPT要求的字体规范、行距以及页面设计等浪费精力。

GML是标准通用标记语言的先驱和基础，SGML是当今创建结构化文档描述语言规则的战略集合。很多网页是用HTML标记表示出的，就是使用GML概念创建文档的例子。扩展标志语言（XML）也根源于GML。

```
SGML
```

它源于1969年IBM公司开发的文档描述语言GML，GML主要用来解决不同系统中文档格式不同的问题。后经过多年发展，1986年经ISO批准为国际标准ISO8897，并被称为SGML。

制定SGML的基本思想是把文档的内容与样式分开。在SGML中，标记分两种：一种用来描述文档显示的样式，称为程序标记；另一种用来描述文档中语句的用途，称为描述标记。一个SGML文件通常分三个层次：结构、内容和样式。结构为组织文档的元素提供框架，内容是信息本身，样式控制内容的显示。
SGML的平台无关性、结构化、可扩展等特性，使得它使用范围很广，被许多大型公司开始用来创建和发布信息。

```
HTML
```

超文本标记语言(HTML)起源于标准通用标记语言(SGML)，由世界上最大的粒子物理研究实验室欧洲核子研究中心CERN(the European Organization for Nuclear Research)于1991年首先提出，是推动Web迅速发展的原动力。

在互联网发展的早期，为了在各种网络环境之间、不同文件格式之间进行交流，在SGML基础上，CERN提出了超文本标记语言(Hyper Text Markup Language，HTML)的概念。

HTML是一种用来制作超文本文档的简单标记语言，它定义了一组标记符号(tag)，对文件的内容进行标注，指出内容的输出格式，如字体大小、颜色、背景颜色、表格形式、各部分之间逻辑上的组织等，从而实现了文件格式的标准化。简单地说，HTMI。文件包含了文档数据和显示样式两部分，其中文档数据是显示在Web浏览器中的数据内容，显示样式则规定了这些内容在浏览器中以何种格式、样子呈现给用户。通过统一使用支持HTMI。的浏览软件，用户可以在任意异构的网络环境中阅读同一个文件，得到相同的显示结果，并可以对文件进行跳跃式阅读，展现了很强的表现力。

1980年，物理学家蒂姆·伯纳斯-李在欧洲核子研究中心（CERN）在承包工程期间，为使CERN的研究人员使用并共享文档，他提出并创建了原型系统ENQUIRE。1989年，伯纳斯-李在一份备忘录中提出了一个基于互联网的超文本系统[3]。他规定了HTML并在1990年底写出了浏览器和服务器软件。同年，伯纳斯-李与CERN的数据系统工程师罗伯特·卡里奥联合为项目申请资助，但未被CERN正式批准。在他的个人笔记中[4]伯纳斯-李列举了“一些使用超文本的领域”，并把百科全书列为首位[5]。

HTML的首个公开描述出现于一个名为“HTML标签”的文件中，由蒂姆·伯纳斯-李于1991年底提及[6][7]。它描述了18个元素，包括HTML初始的、相对简单的设计。除了超链接标签外，其他设计都深受CERN内部一个以标准通用标记语言（SGML）为基础的文件格式SGMLguid的影响。这些元素在HTML 4中仍有11个存在[8]。

伯纳斯-李认为HTML是SGML的一个应用程序。

https://zh.wikipedia.org/wiki/HTML

XML是由互联网联盟(Wor1d Wide Web Consortium，W3C)的XML工作组定义的。

这个工作组是这样描述该语言的:
“扩展标记语言(XML)是SGML的子集，其目标是允许普通的SGML在Web上以目前HTML的方式被服务、接收和处理。XML被设计成易于实现，且可在SGML和HTML之间互相操作。”

HTML的出现极大地推动了世界范围内的互联网的发展，万维网就是一个主要成果。然而，HTML在某些方面是一种通用编码的倒退。 第一，HTML为了获得精简编码方式的有效性，而抛弃了通用编码的一些基本原则。例如，通用编码要求一个文档类型能用于任何目的，要求用户编码时应重载标签而不是重新定义特殊目的的标签。

第二，HTML中的很多标签是纯粹追求显示效果的，这显然与通用编码的初衷相违背。这种简化的结构很难区分文档的开始部分和结束部分。

如今采用HTML编码的文档非常依赖纯格式化以致不能被用作其他目的。虽然HTML有着诸多的问题，但是我们不能抹杀它在标记语言的发展过程中为Web的发展带来飞跃所起到的重要作用。至少，HTML使得全世界的人对电子文档化和资源链接充满了兴趣。

为了回归到理想的通用编码状态，一些人试着改变SGML以适应Web，或者干脆改变Web以适应SGML，不幸的是，这被证实是非常困难的。SGML是如此的巨大，不可能被塞进小小的Web浏览器中。所以采用一种更小的，却又能保持SGML通用性的语言是唯一可行的途径，由此，令人兴奋不已的可扩展标记语言诞生了。 终于在20世纪90年代中期，由一些公司和组织共同组建的互联网联盟开始致力于创造一种融合SGML灵活性和HTML简单性的标记语言。1998年，W3C就发布了XML1.0规范，使用它来简化Internet的文档信息传输。

更多信息移步

http://wiki.mbalib.com/wiki/XML

XHTML 是以 XML 格式编写的 HTML。XHTML 指的是可扩展超文本标记语言，XHTML 是更严格更纯净的 HTML 版本，XHTML 是以 XML 应用的方式定义的 HTML，XHTML 得到所有主流浏览器的支持.

为什么使用 XHTML?
因特网上的很多页面包含了"糟糕"的 HTML。
譬如

```
<html>
<head>
<title>This is bad HTML</title>
<body>
<h1>Bad HTML
<p>This is a paragraph
</body>

XML 是一种必须正确标记且格式良好的标记语言。
今日的科技界存在一些不同的浏览器技术。其中一些在计算机上运行，而另一些可能在移动电话或其他小型设备上运行。小型设备往往缺乏解释"糟糕"的标记语言的资源和能力。
所以 - 通过结合 XML 和 HTML 的长处，开发出了 XHTML。XHTML 是作为 XML 被重新设计的 HTML。
```

再着重说一说XML特点和作用

```
XML 指可扩展标识语言（eXtensible Markup Language）
XML 的设计宗旨是传输数据，而非显示数据
XML 标签没有被预定义。您需要自行定义标签。
作为一种通用的数据存储和通信格式被广泛应用。
描述的数据作为一棵树型的结构而存在。
```

```
在知乎上关于XML的讨论 请移步 http://www.zhihu.com/question/20311704
```

```
不被察觉的XML
请用压缩软件打开doc文件 exc1文件发现本质上EXCEL和WORD 本质上都是xm1文件加上一些描述文件
```

XML 应用于 web 开发的许多方面，常用于简化数据的存储和共享。

XML 把数据从 HTML 分离。如果你需要在 HTML 文档中显示动态数据，那么每当数据改变时将花费大量的时间来编辑 HTML。通过 XML，数据能够存储在独立的 XML 文件中。这样你就可以专注于使用 HTML 进行布局和显示，并确保修改底层数据不再需要对 HTML 进行任何的改变。

一个负责貌美如花，一个负责糊口养家。

XML 简化数据共享

```
在真实的世界中，计算机系统和数据使用不兼容的格式来存储数据。
XML 数据以纯文本格式进行存储，因此提供了一种独立于软件和硬件的数据存储方法。
这让创建不同应用程序可以共享的数据变得更加容易。
```

XML 简化数据传输

```
通过 XML，可以在不兼容的系统之间轻松地交换数据。
对开发人员来说，其中一项最费时的挑战一直是在因特网上的不兼容系统之间交换数据。
由于可以通过各种不兼容的应用程序来读取数据，以 XML 交换数据降低了这种复杂性。
```

XML 用于创建新的 Internet 语言

```
很多新的 Internet 语言是通过 XML 创建的：
```

其中的例子包括：

```
XHTML - 最新的 HTML 版本
WSDL - 用于描述可用的 web service
WAP 和 WML - 用于手持设备的标记语言
RSS - 用于 RSS feed 的语言
RDF 和 OWL - 用于描述资源和本体
SMIL - 用于描述针针对 web 的多媒体
```

第一个简单的xml文件

```
<?xml version="1.0" encoding="utf-8"?>
<note>
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
</note>
```

x规则说明
它定义 XML 的版本（1.0）和所使用的编码 UTF-8，version不可省略，encoding可以省略；

<note>描述文档的根元素，有且只有一个根元素。

所有 XML 元素都须有关闭标签

```
XML 元素使用 XML 标签进行定义。
在 HTML，经常会看到没有关闭标签的元素：<p>This is a paragraph
在 XML 中，省略关闭标签是非法的。所有元素都必须有关闭标签：必须是<p>This is a paragraph</p>
<br>同html中类型一致自定义标签是成对出现，有开始就需要结束。
<br>接下来4行描述根节点的4个子元素(to,from,heading,body)
<br>最后一行是定义跟元素的结尾</note>
<br>该XML文件描述的就是从Jani发给Tove的一封信。
```

XML标签对大小写敏感。

```
在 XML 中，标签 <Letter> 与标签 <letter> 是不同的。
必须使用相同的大小写来编写打开标签和关闭标签：
```

XML属性值必须加引号。

```
    <note date="11/11/2016">
    </note>
```

这是XML的注释。

```
    <!-- This is a comment -->
```

格式化工具

```
帮助我们把杂乱无章的xml文件以非常优雅，有层次感的形式呈现出来。
工具

    XMLViewr
网站有很多

    http://tool.oschina.net/codeformat/xml/

格式化之后就是这样的

    <?xml version="1.0" encoding="utf-8"?>
    <note>
        <to>Tove</to>
        <from>Jani</from>
        <heading>Reminder</heading>
        <body>Don't forget me this weekend!</body>
    </note>
```

XML元素

XML元素指的是开始标签直到结束标签的部分，可以包含文本、属性、其他元素 中的一种或者多种。 比如下面这个xml文件

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<bookstore>
    <book category="CHILDREN">
        <title>Harry Potter</title>
        <author>J K. Rowling</author>
        <year>2005</year>
        <price>29.99</price>
    </book>
    <book category="WEB">
        <title>Learning XML</title>
        <author>Erik T. Ray</author>
        <year>2003</year>
        <price>39.95</price>
    </book>
</bookstore>
```

在上面的示例中<bookstore> <book>都有元素内容 因为他们包含其他元素。<book>元素也有属性category="WEB"， <title> <author> <year> <price>有文本内容。

```
<?xml version="1.0" encoding="utf-8"?>
<student_list>
    <student name="王非" residence _address="北京">
        <address>北京</address>
    </student>
    <student name="行者" residence_address="深圳">
        <address>北京</address>
    </student>
</student_list>
```

XML元素是可拓展的
如果在上述示例文件中加上等额外信息会导致收到该xml文件的程序在解析的时候崩溃吗?答案是否定的，因为在xml文件的时候是只获取指定元素的数据，对于新增的数据只能不能获取到而已并不会引起数据解析错误。XML 的优势之一，就是可以在不中断应用程序的情况下进行扩展。

CDATA全称character DATA（字符数据） 他的作用是将整个文本内容解释为纯字符数据 CDATA段的一般形式为<[CDATA[你的文本内容]]> 注意CDATA是XML的关键字，必须严格要求大写。这样特殊字符就可以不经过特殊处理了。

环境搭建

- cygwin (like Linux):

  gcc gcc-core gcc-g++ gcc-mingw-core gcc-mingw-g++ make gdb binutils ./configure —enable-threads=no（使其不依赖于pthread）

- Linux

  ./configure —enable-threads=no && make Linux下只需要libmxml.a mxml.h这两个文件即可

- VC/Visual Studio X

  1. 使用VS X打开vcnet文件夹下的mxml.sln(VS2015测试有效)
  2. 点击确定升级工程文件
  3. config.h把第64行代码进行注释//#define HAVE_PTHREAD_H 1 ；如果没有请略过
  4. 选择静态库、设置好release 或者debug模式 点击生成解决方案
  5. 将生成的文件（release模式把mxml1.lib以及原有的mxml.h 和config.h）添加到新的项目文件中；然后在项目->属性->链接器->附加依赖项->里面加上mxml1.lib->确定退出即可开始工作。
  6. 搭建完毕。windows下只要libmxml.a mxml.h vcnet/config.h 这三个文件

C语言解析XML

如何使用C语言解析xml文件呢?需不要要自己造个轮子来解析xml文件呢?

C语言工具中存在了很多优秀的工具，我们站在巨人的肩膀上开发就能事半功倍，坐地日行八万里，巡天遥看一千河。
工具有很多，用会一个其他的都是类似的。 本文选择的XML解析库是minixml。（当然还有其他很多优秀的解析库比如libxml，有兴趣的同学可以自己去研究一下。）在Linux或者Windows的VS中均可使用。

mxml下载地址

```
http://www.msweet.org/downloads.php?L+Z3
```

minixml常用函数接口

均需要包含头文件

```
#include <mxml.h>
```

创建一个新xml文件

```
    函数接口:mxml_node_t    *mxmlNewXML(const char *version);
    参数1:version默认传入"1.0"即可
    返回值:返回新创建的xml文件节点
    提示：虽然xml中可以指定字符编码，但是在mxml中默认的编码就是UTF8编码。
```

其中一个节点新增节点

```
    函数接口:mxml_node_t    *mxmlNewElement(mxml_node_t *parent, const char *name);
    参数1:父节点指针
    参数2:新节点名称
    返回值:返回新创建的子节点
```

设置节点属性名和值

```
    函数接口:void mxmlElementSetAttr(mxml_node_t *node, const char *name,
                          const char *value);
    参数1:被设置的节点的地址
    参数2:设置的属性名
    参数3:设置的属性值
```

创建结点的文本

```
    函数接口：mxml_node_t *mxmlNewText ( mxml_node_t *parent, int whitespace, const char
*string);
    参数1：被设置的结点的地址
    参数2：1 = leading whitespace, 0 = no whitespace
    参数3：文本
```

保存节点到xml文件

```
    函数接口:int mxmlSaveFile(mxml_node_t *node, FILE *fp,mxml_save_cb_t cb);
    参数1:node表示希望被保存的xml节点树的根节点，指向xml结构的节点指针
    参数2:fp为C中使用fopen函数打开文件所返回的FILE类型的指针
    参数3:默认情况下使用MXML_NO_CALLBACK即可

        MXML_INTEGER_CALLBACK-所有的数据节点包含以空格分割的整数。
        MXML_OPAQUE_CALLBACK-所有的数据节点包含"不透明"字符串（CDATA）。
        MXML_REAL_CALLBACK-所有的数据节点包含以空格分割的浮点数。
        MXML_TEXT_CALLBACK-所有的数据节点包含以空格分割的文本字符串。
```

删除节点内存

```
    函数接口:mxmlDelete(mxml_node_t *node);
    参数1:node为指向节点树的指针。
    该函数将释放node节点指针所指向的整棵节点树，而不用我们去一个一个节点释放。如果该节点还有父节点，会从节点树中先使用mxmlRemove()移除该节点。
```

从文件中加载xml

```
    函数接口:mxml_node_t    *mxmlLoadFile(mxml_node_t *top, FILE *fp,mxml_type_t (*cb)(mxml_node_t *));
    参数1:top为加载的节点的父节点  如果是文档节点则填NULL即可
    参数2:fopen函数返回的文件的指针
    参数3:默认情况下使用MXML_NO_CALLBACK即可
```

另外除了从文件中加载xml之外，还可以从缓冲区中使用mxmlLoadString()函数加载XML节点数。

获取节点属性

```
    函数接口:const char *mxmlElementGetAttr(mxml_node_t *node, const char *name);
    参数1:node为指向结点树的指针
    参数2:name为获取的属性名
    返回值:指定属性名的属性值
```

获取指定节点的文本内容。

```
    函数接口：  const char    *mxmlGetText(mxml_node_t *node, int *whitespace);
```

跳转到下一个节点

```
    函数接口:mxml_node_t    *mxmlWalkNext(mxml_node_t *node, mxml_node_t *top,int descend);
    遍历到XML树中的下一个逻辑节点。
    node  当前节点
    top   顶级节点
    descend参数有三个备选项
    MXML_NO_DESCEND
        含义是不查看任何的子节点在XML元素层次中，仅查看同层级的节点或者父节点直到到达根节点或者top节点.
    MXML_DESCEND_FIRST含义是向下搜索到一个节点的第一个匹配子节点，但不再继续向下搜索。你一般使用于遍历一个父节点的直接的子节点。
    MXML_DESCEND含义是可以一直向下搜索.
```

总之,descend 确定了是否向下搜索子节点；通常你将使用 MXML_DESCEND_FIRST作为第一次搜索，然后使用使用 MXML_NO_DESCEND来发现更多的这个节点的直接子节点。

查找节点

```
    mxml_node_t *mxmlFindElement(mxml_node_t *node, mxml_node_t *top,
                         const char *name, const char *attr,
                const char *value, int descend);

    node节点为被查找的结点
    top为顶层结点
    name，attrr，value为NULL时表示任意匹配；否则为精确匹配。
    descend同上个接口使用方式。
```

练习:新建一个指定格式的xml文件

```
<?xml version="1.0"?>
<html>
    <head>
    <title>Document</title>
    </head>
    <body>
        <p style="color:red">helloworld </p>
    </body>
```

```
</html>
```

马上练习

```
<?xml version="1.0" encoding="utf-8"?>
<bookstore>
    <book category="CHILDREN">
        <title>Harry Potter</title>
        <author>J K. Rowling</author>
        <year>2005</year>
        <price>29.99</price>
    </book>
    <book category="WEB">
        <title>Learning XML</title>
        <author>Erik T. Ray</author>
        <year>2003</year>
        <price>39.95</price>
    </book>
</bookstore>
```

特别提示 由于创建XML文件的API接口和解析XML文件的API接口工作原理以及使用方式是类似的，所以这个功能就不详细讲解了。我们重点放在解析xml文件。工作当中如果需要用到此功能，再查询一下函数接口即可。

解析一个xml文件中指定的数据

文件整体结构

```
<?xml version="1.0" encoding="utf-8"?>
<html>
    <head>
    <meta charset="UTF-8">
    <title>Document</title>
    </head>
    <body>
        <!--- 所有的新闻条目都在body节点上并列----->
        <a href="/kx/detail?id=61800" target=_blank>
            <li id="">
                <span class="time">17:35</span>
                <p> 阿富汗警方：阿富汗首都喀布尔发生大爆炸 </p>
            </li>
        </a>
    </body>
</html>
```

每条新闻的结构

```
<a href="/kx/detail?id=61800" target=_blank>
    <li id="">
        <span class="time">
            17:35
        </span>
        <p>
            阿富汗警方：阿富汗首都喀布尔发生大爆炸
        </p>
    </li>
</a>
```

编译命令

```
gcc -Os -g -Wall -D_THREAD_SAFE -D_REENTRANT -c main.c && gcc -Os -g -o main main.o libmxml.a -lpthread
```

解析QQ数据

从0003_QQPicConfig_for_read.xml文件中解析每张图片的md5值以及网络地址 以及图片格式(jpg还是png)

解析天气数据

0004_china_weather_data.xml来源于http://flash.weather.com.cn/wmaps/xml/china.xml。
从0004_china_weather_data.xml文件中获取出 每个省份名称、相应的省会城市名、天气情况、最高、最低气温、以及风力情况。

minixml处理带有空格的文本的解决方案

Mini-XML中的节点类型定义和其他有些解析器有些不同，其中整数、浮点、和文本节点是指在一个XML元素中一系列的使用空格作为分割的值，每个元素可以拥有多个以上节点，并可以选择使用空格分开，如：aa bb cc，Mini-MXML在使用参数：MXML_TEXT_CALLBACK进行载入时，将在abc元素下面生成3个text类型的子节点。而不透明字符串类型(OPAQUE)则不进行字符串分割，在载入时需要使用MXML_OPAQUE_CALLBACK参数，将所有字符串形成一个子节点。

操作如下：

```
使用MXML_OPAQUE_CALLBACK模式：mxmlLoadFile(NULL, fp, MXML_OPAQUE_CALLBACK)
```

引用获取字符串时：

```
mxml_node_t *Tmp = mxmlFindElement(Root, Tree, name,NULL, NULL,MXML_DESCEND_FIRST);
```

在MXML_OPAQUE_CALLBACK模式下导入文件，引用获取的字符串：

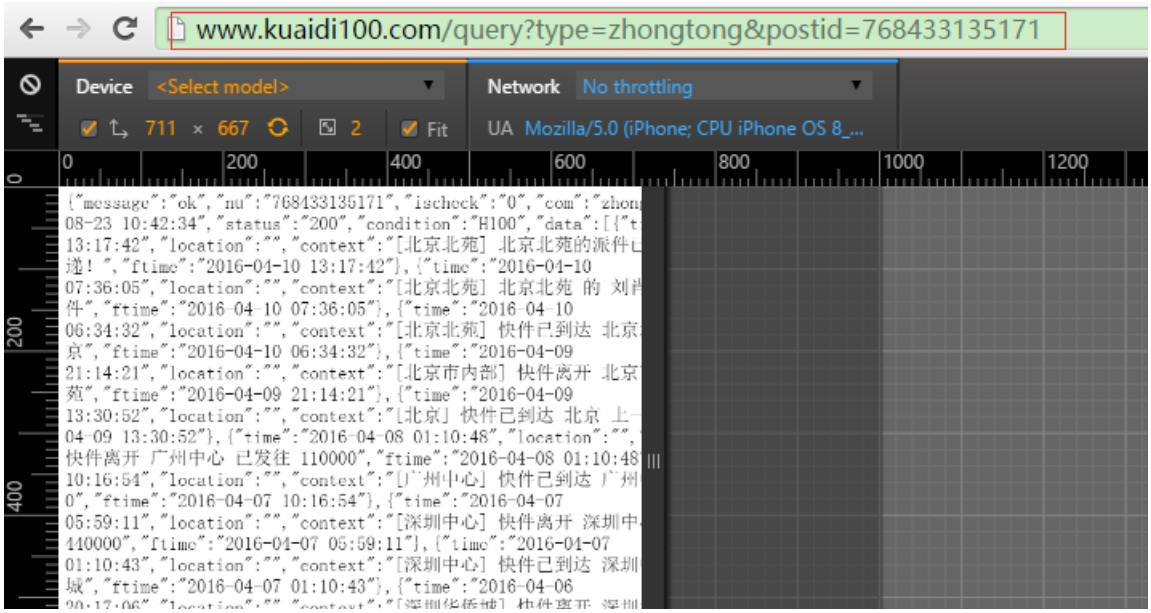
```
Tmp->child->value.opaque
```

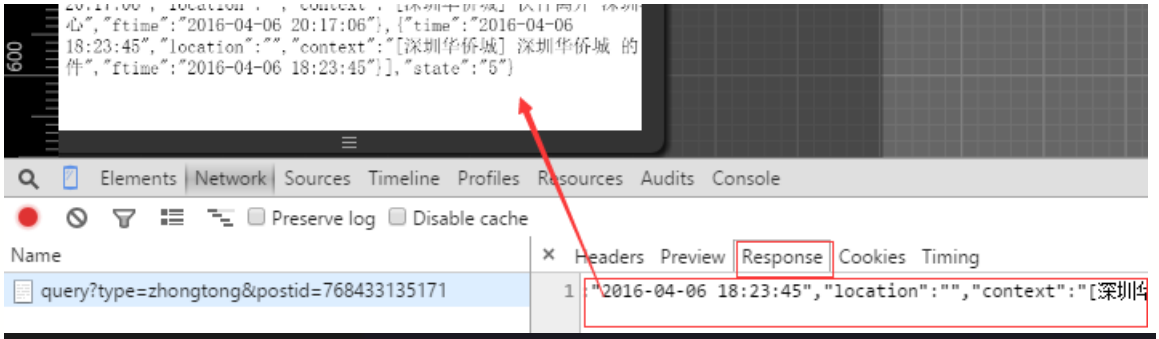
JSON

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的用途





首单立减5元！8月22日周一7点至19点，限南京

输入您的手机号码安装 Uber 应用程序，您就可获得 CN¥5 off next ride。

* 手机号码

+86 157 1555

SyntaxError: JSON Parse error: Unexpected identifier "undefined"

JSON语法说明

```
先来看一个简单的JSON
简单的JSON
{
    "stars": [
        { "name":"Faye" ,"address":"北京" },
        { "name":"andy" ,"address":"香港" },
        { "name":"eddie","address":"台湾" },
    ]
}
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），字典（dictionary），哈希表（hash table），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）。

上述的描述如果不明白，那就换一种方式归纳一下JSON的三种语法：

```
    键/值对 key:value，用半角冒号分割。
```

```
比如  "name":"Faye"
key必须是字符串类型，键/值对的值可以是：
    数字（整数或浮点数）
    字符串（在双引号中）
    逻辑值（true 或 false）
    null(空值  未设置)
    对象（在花括号中）
    数组（在方括号中）
key是字符串类型，标准写法请加双引号，以免有可能报错。
```

```
    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)。 { "stars":[ { "name":"Faye" ,"address":"北京" }, { "name":"andy" ,"address":"香港" }, { "name":"eddie","address":"台湾" }, ] }
```

```
{
    "love": ["乒乓球","高尔夫","斯诺克","羽毛球","LOL","撩妹"]
}
```

总结最终的构成

object {} { members } members pair pair , members

pair string : value

array

```
[]
[ elements ]
```

elements

```
value
value , elements
```

value

```
string
number
object
array
true
false
null
```

格式化工具

本地格式化

JsonView
在线格式化

http://json.cn/
http://www.json.org.cn/

C语言的JSON库

cJSON库下载地址 https://github.com/DaveGamble/cJSON
c语言JSON库比较多，这里就讲其中一种叫cJSON的库，所有的库使用大同小异，有兴趣的同学们请自行研究。
如果学习了C++，其实会发现面向对象的语言在解析JSON的时候那才叫天生丽质。

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

```
/* Supply a block of JSON, and this returns a cJSON object you can interrogate. Call cJSON_Delete when finished. 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*/
extern cJSON *cJSON_Parse(const char *value);
```

将传入的JSON结构转化为字符串

```
/* Render a cJSON entity to text for transfer/storage. Free the char* when finished. */
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

```
/* Delete a cJSON structure. */
void cJSON_Delete(cJSON *c)
```

> 练习1： 将字符串 char *char_json = "{"habit":"1o1"}";解析为json保存为0001.json

这里只说明主要库函数，其他库函数基本类似。请直接查看cJSON.h文件。

主要库函数：

> 创建函数返回创建的json结构

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))
```

> 练习2 使用上述库函数创建一个如下的json文件，名字为02.json

```
{
    "country":"china","stars":
        [
            {"name":"Faye","address":"beijing"},
            {"name":"andy","address":"HK"},
            {"name":"eddie","address":"Taiwan"}
        ]
}
```

C语言库函数解析JSON文件

> **主要库函数：**针对每个函数做一个示例代码演示调用

```
/* Get item "string" from object. Case insensitive. 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;    }


/* Returns the number of items in an array (or object). 返回数组结点array中成员的个数(数组大小) */
extern int    cJSON_GetArraySize(cJSON *array);


/* Retrieve item number "item" from array "array". Returns NULL if unsuccessful.
   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);

/* Macro for iterating over an array  遍历数组*/
#define cJSON_ArrayForEach(pos, head)   for(pos = (head)->child; pos != NULL; pos = pos->next)
```

物流接口/快递接口：

http://www.kuaidi100.com/query?type=快递公司代号&postid=快递单号

ps:快递公司编码:申通="shentong" EMS="ems" 顺丰="shunfeng" 圆通="yuantong" 中通="zhongtong" 韵达="yunda" 天天="tiantian" 汇通="huitongkuaidi" 全峰="quanfengkuaidi" 德邦="debangwuliu" 宅急送="zhaijisong"

解析练习 获取并显示快递编号nu;然后取出data中的每条信息并且予以显示

显示结果如下

```
mo@test-pc JSON/json03
$ ls
01_json.c                           cJSON.c          json01.pro.user
02_json.c                           cJSON.h          main.exe
03_express_information.json         deployment.pri   main.exe.backup
03_json_parse_express_information.c  json01.pro

mo@test-pc JSON/json03
$ gcc -o main 03_json_parse_express_information.c  cJSON.c
mo@test-pc JSON/json03

$ ./main.exe
express info[nu] is 768433135999

time:2016-04-10 13:17:42
info:北京北苑的派件已签收，感谢您使用中通快递！

time:2016-04-10 07:36:05
info:北京北苑 的 刘肖18010456227 正在派件

time:2016-04-10 06:34:32
info:快件已到达 北京北苑 上一站是 北京

time:2016-04-09 21:14:21
info:快件离开 北京市内部 已发往 北京北苑

time:2016-04-09 13:30:52
info:快件已到达 北京 上一站是 0

time:2016-04-08 01:10:48
info:快件离开 广州中心 已发往 110000

time:2016-04-07 10:16:54
info:快件已到达 广州中心 上一站是 0

time:2016-04-07 05:59:11
info:快件离开 深圳中心 已发往 440000

time:2016-04-07 01:10:43
info:快件已到达 深圳中心 上一站是 深圳华侨城

time:2016-04-06 20:17:06
info:快件离开 深圳华侨城 已发往 深圳中心

time:2016-04-06 18:23:45
info:深圳华侨城 的 邹国 已收件
```

课后作业

作业1 调用cJSON相关函数创建以下格式的04.json文件

```
{"love":[
    {"country":"china","stars":[
       {"name":"Faye","address":"北京"},
       {"name":"andy","address":"香港"},
       {"name":"eddie","address":"台湾"}
    ]},
    {"country":"USA","stars":[
      {"name":"Michael Jackson","address":"Indiana"},
      {"name":"Schwarzenegger","address":"California"}
    ]},
    {"country":"Janpe","stars":[
      {"name":"Aoi sola","address":"北京"}
    ]}
]}
```

关于JSON与XML的区别的讨论

```
http://www.zhihu.com/question/25636060
```