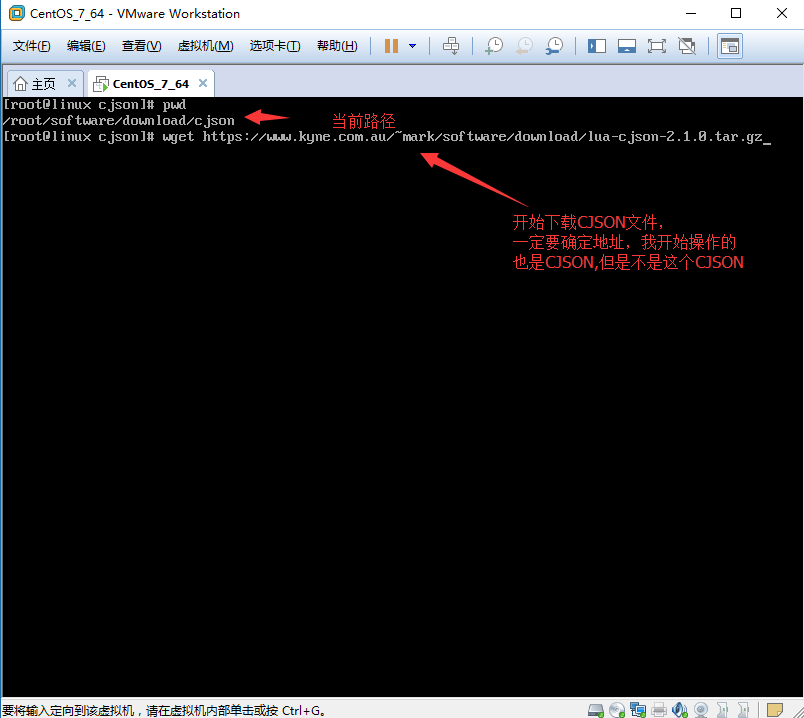
**一、引言**  
  
         学习Redis也有一段时间了，感触还是颇多的，但是自己很清楚，路还很长，还要继续。上一篇文章简要的介绍了如何在Linux环境下安装Lua，并介绍了在Linux环境下如何编写Lua脚本和运行。由于我们这个系列是以Redis为主，所以也介绍了Redis和Lua如何进行整合、运行。  
  
        在Lua脚本中有一个很重要的数据类型，那就是table类型，大家可以把Lua的table类型暂时理解为数组，只是Lua的table类型的下标可以是数字，可以是字符，除了（nil）类型，其他类型都可以做为table类型的下标。我们在使用Redis和Lua的过程中，比较多的会用到这个table类型，今天的主要任务就是介绍table类型的使用、CJSON的解析和如何通过C#语言来对Redis和Lua的进行整合操作。后续我还会推出针对Lua脚本语法的文章来扩充这个系列。  
  
**二、Lua简介**  
      
         Lua 是一个小巧的脚本语言。其设计目的是为了嵌入应用程序中，从而为应用程序提供灵活的扩展和定制功能。Lua由标准C编写而成，几乎在所有操作系统和平台上都可以编译，运行。Lua并没有提供强大的库，这是由它的定位决定的。所以Lua不适合作为开发独立应用程序的语言。  
  
        在Redis中使用Lua有很多好处，它可以减少网络开销，把多个操作一次性打包执行。Lua脚本天生支持原子性的操作，避免开启第三方事务，提高了性能。代码重用也是一个重要的好处，写好的代码会被加载到Redis内存中，可以供其他客户端使用，减少重复劳动。Lua脚本使用C语言写成的，执行速度很快，并天然具有可移植性，也是代码重用的很好体现。一个完整的Lua解释器，不过200k，在目前所有脚本引擎中，Lua的速度是最快的。这一切都决定了Lua是作为嵌入式脚本的最佳选择。  
  
**三、基本操作**  
  
    **1、在Linux环境下安装CJSON和简单使用**  
  
        cjson是一个类型转换工具类，可以把一个字符串转换为Lua的类型，如果类型嵌套多层，还可以转换为Lua的table类型，所以这个工具很重要，在编写Lua脚本的过程中经常使用，所以我们先来介绍cjson这个工具类的安装和使用。  
  
**1.1、下载CJSON软件包，官网地址：https://www.kyne.com.au/~mark/software/lua-cjson.php。**

[root@linux cjson]# pwd

/root/software/download/cjson

[root@linux cjson]# wget https://www.kyne.com.au/~mark/software/download/lua-cjson-2.1.0.tar.gz

                         
  
**1.2、解压到当前目录，当然也可以解压到其他目录，解压到其他目录，比如：/usr/local，命令如下：tar zxvf lua-cjson-2.1.0.tar.gz -C(大写的) /usr/local**

[IMG_257](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

//当前目录 [root@linux cjson]# pwd

/root/software/download/cjson

//显示当前目录下的文件和目录 [root@linux cjson]# ls

lua-cjson-2.1.0.tar.gz

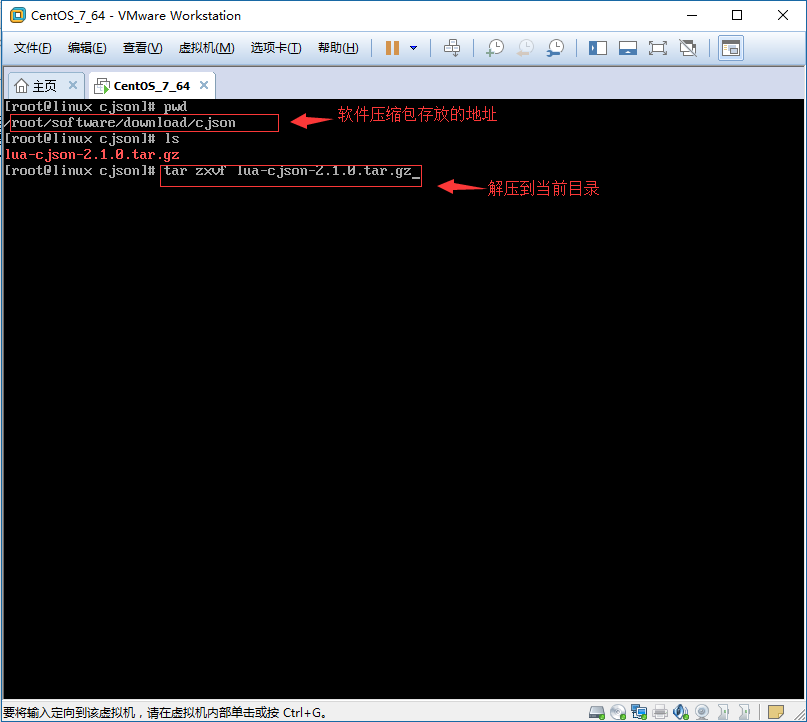
//解压到当前目录

[root@linux cjson]# tar zxvf lua-cjson-2.1.0.tar.gz [-C /usr/local]

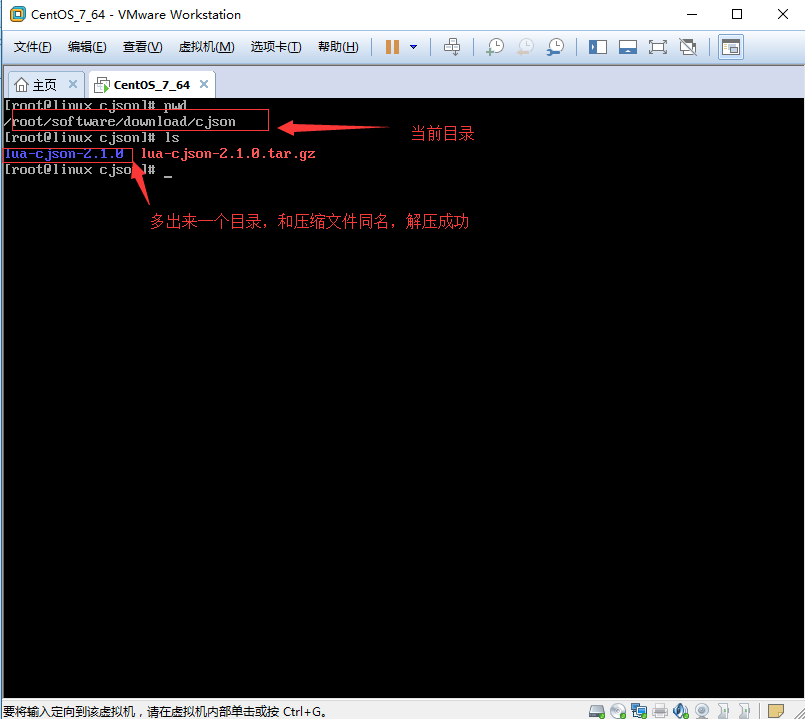
//...

complete!

[IMG_258](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

                     解压成功后的目录结构：

                        
  
**1.3、进入到解压目录的根目录，准备修改Makefile文件，用vim打开的修改文件的时候，注意名称大小写拼写正确。**

[IMG_261](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

//当前目录 [root@linux cjson]# pwd

/root/software/download/cjson

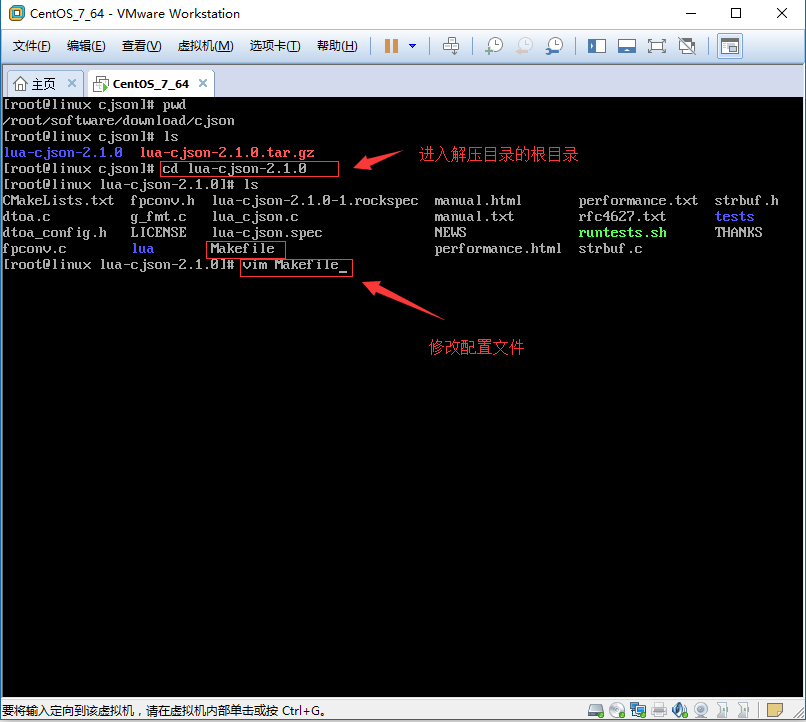
//进入到CJSON的解压根目录

[root@linux cjson]# cd lua-cjson-2.1.0

//进入到根目录，修改Makefile文件

[root@linux lua-cjson-2.1.0]# vim Makefile

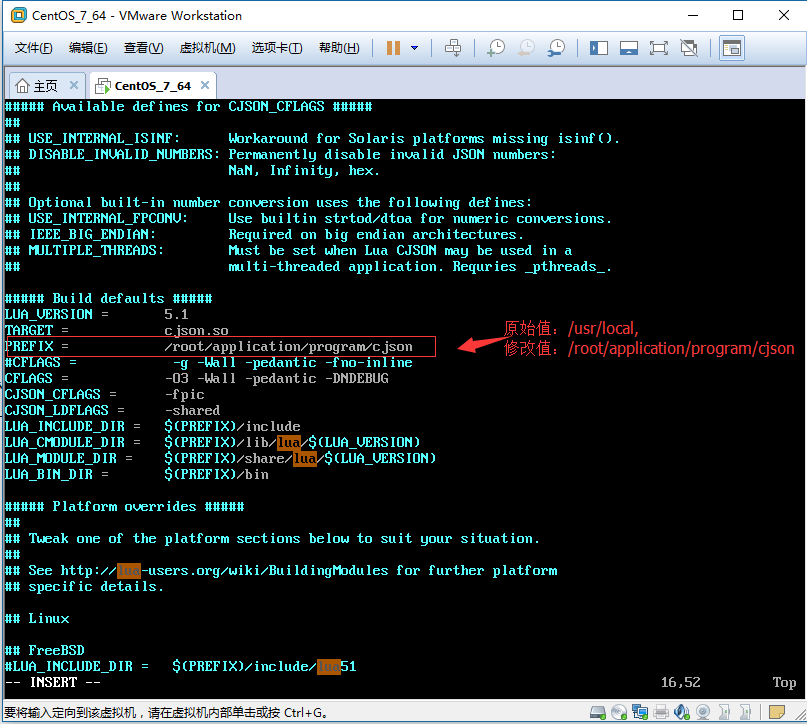
[IMG_262](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

                          
  
**1,4、vim 进入Makefile文件，修改PREFIX选项，修改完毕，按ESC按钮，再按：,最后按wq，保存退出。**

//当前目录

[root@linux lua-cjson-2.1.0]# pwd

/root/software/download/cjson/lua-cjson-2.1.0

                          
  
**1.5、使用make命令在解压文件的根目录下进行编译。**

[IMG_265](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

//当前目录

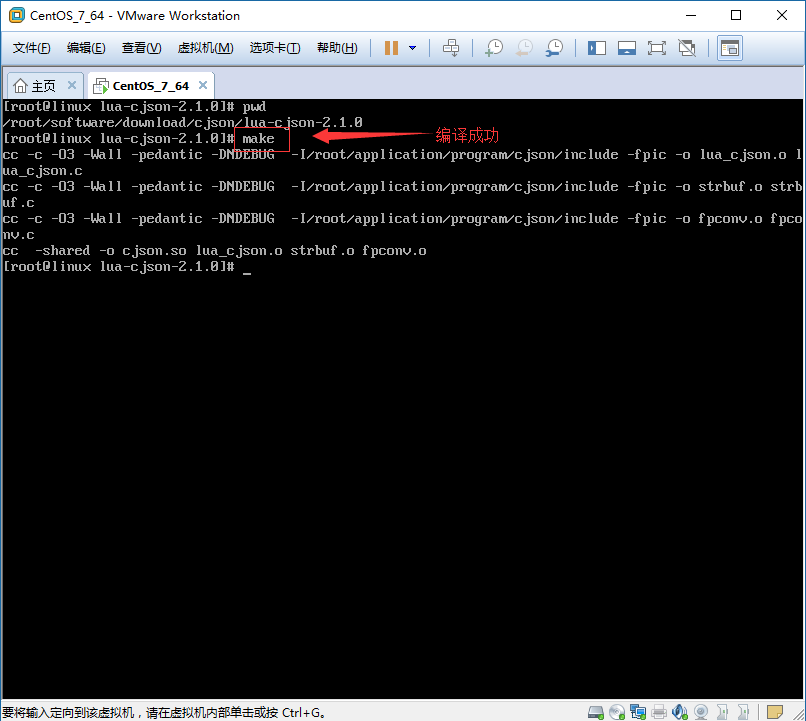
[root@linux lua-cjson-2.1.0]# pwd

/root/software/download/cjson/lua-cjson-2.1.0

[root@linux lua-cjson-2.1.0]# make

//开始编译

[IMG_266](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

  
  
**1.6、最后使用 make install 安装cjson软件包。**

[IMG_268](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

//当前目录

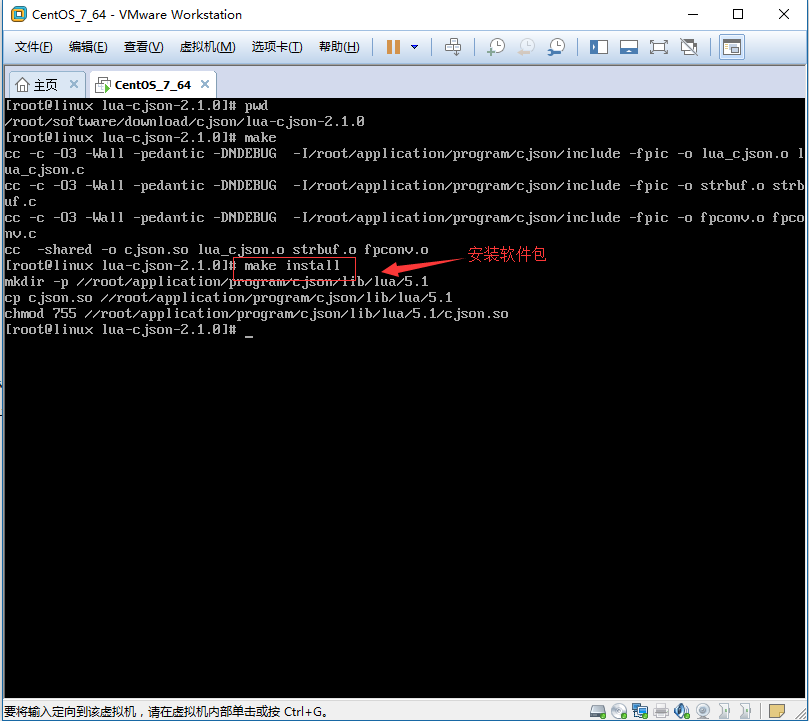
[root@linux lua-cjson-2.1.0]# pwd

/root/software/download/cjson/lua-cjson-2.1.0

[root@linux lua-cjson-2.1.0]# make install

//开始安装

[IMG_269](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

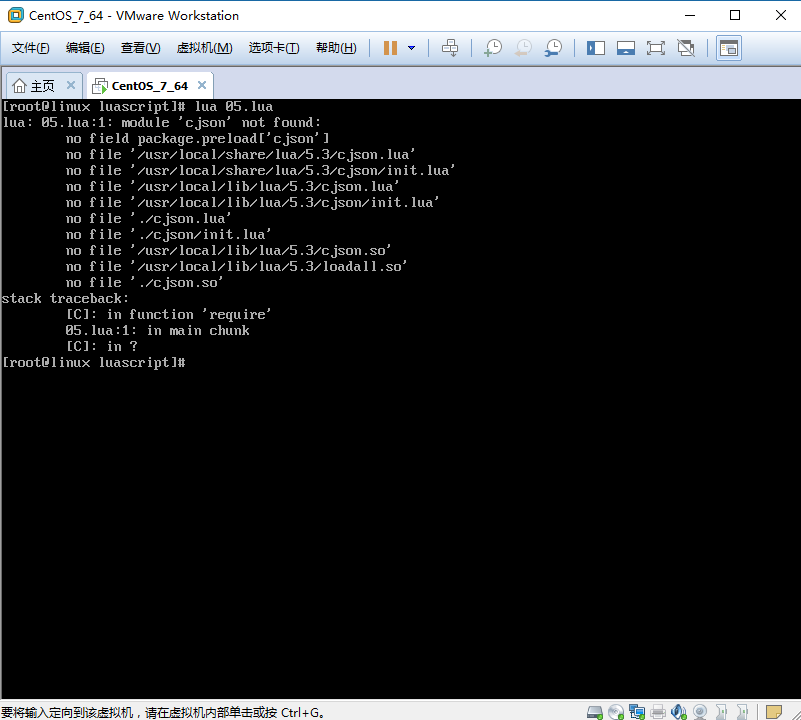
  
  
**1.7、cjson.so文件很重要，是核心运行文件，如果出现错误，在某某目录没有找到cjson这个文件，就在其他目录找到，并拷贝到目标目录，可以解决所出现的错误。**

[root@linux program]# pwd

/root/application/program

//拷贝sjson.so文件到指定目录

[root@linux program]# sudo cp -r cjson/lib/lua/5.1/cjson.so /usr/local/lib/lua/5.3

                     **错误如图：**  
  
                       

**只需要把cjson.so文件拷贝到【/usr/local/lib/lua/5.3（这个版本具体情况要看具体情况）/】目录下就可以了**  
  
**1.8、安装完成，开始测试cjson安装是否成功，此步骤有两步，必须全部都完成。**

[IMG_272](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

//1、从Linux命令行模式进入lua命令行测试安装是否正确

[root@linux lua-cjson-2.1.0]# lua

Lua 5.3.4 Copyright (C) 1994-2017 Lua.org PUC-Rio

>local cjson=require("cjson")

>local cjson2=require "cjson.safe"

//以上所有操作没有提示任何错误

//2、我们在Linux命令行下在执行lua命令，执行lua脚本，测试是否安装成功

//05.lua脚本文件的源码

local cjson=require "cjson"

local smapleJson=[[{"age":"23","testArray":{"array":[8,9,1,14,15]},"baidu":"www.baidu.com"}]];

local data=cjson.decode(sampleJson)

print(data["age"]);

print(data["testArray"]["array"][1])

//没有提示任何错误，安装成功  
  
 //在Linux环境下执行刚才的Lua脚本

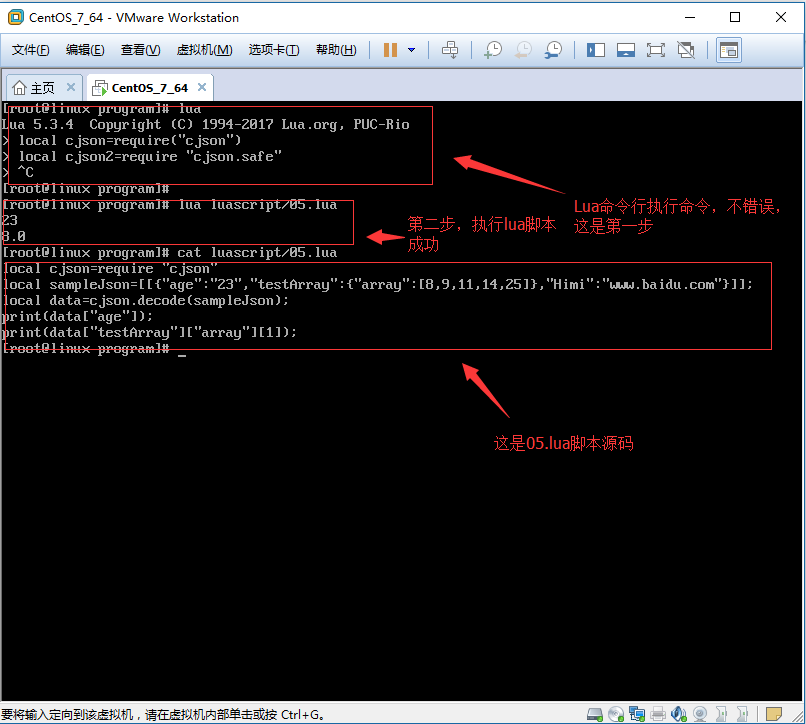
[root@linux lua-cjson-2.1.0]# lua /root/application/program/luascript/05.lua

23

8.0

//安装成功

[IMG_273](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

  
   
  
**2、在Linux环境下，在Lua脚本里面使用cjson来解析类型。**

[IMG_275](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

1 //cjsonDemo.lua的源代码如下： 2 3 local cjson = require "cjson" 4 local retTable = {}; --最终产生json的表 5 6 --顺序数值 7 8 local intDatas = {}; 9 intDatas[1] = 100;10 intDatas[2] = "100";11 12 --数组13 14 local aryDatas = {};15 aryDatas[1] = {};16 aryDatas[1]["键11"] = "值11";17 aryDatas[1]["键12"] = "值12";18 aryDatas[2] = {};19 aryDatas[2]["键21"] = "值21";20 aryDatas[2]["键22"] = "值22";21 22 --对Table赋值23 24 retTable["键1"] = "值1";25 retTable[2] = 123;26 retTable["int\_datas"] = intDatas;27 retTable["aryDatas"] = aryDatas;28 29 --将表数据编码成json字符串30 31 local jsonStr = cjson.encode(retTable);32 print(jsonStr);33 34 return retTable;35 36 //以上为Lua源码

[IMG_276](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

[IMG_277](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

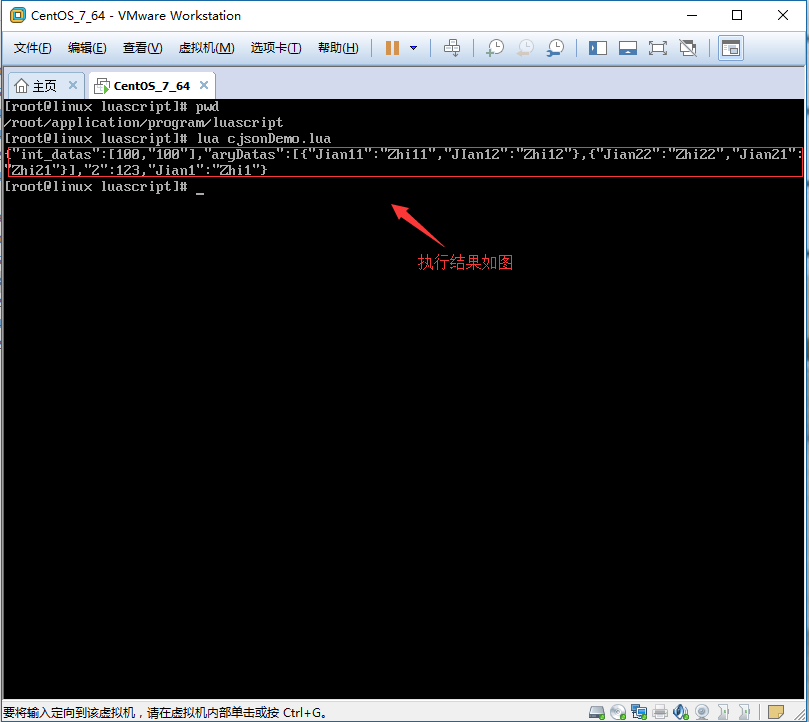
//开始执行命令 [root@linux luascript]# pwd

/root/application/program/luascript //lua脚本存放的位置

[root@linux luascript]# lua cjsonDemo.lua

{"int\_datas":[100,"100"],"2":123,"键1":"值1","aryDatas":[{"键12":"值12","键11":"值11"},{"键21":"值21","键22":"值22"}]}

[IMG_278](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

          执行效果如图：  
            
                    
  
   **3、在Windows环境下，用C#执行lua脚本**  
  
            **3.1、在VS2015中建一个C#控制台应用程序，并添加LuaInterface.dll的引用**  
  
                       LuaInterface下载地址：[http://luaforge.net/projects/luainterface/](http://luaforge.net/projects/luainterface/" \t "http://www.cnblogs.com/PatrickLiu/p/_blank) （下载luainterface-1.5.3，这里面的资源比较多）  
  
　　               LuaInterface.Lua类是CLR访问Lua解释器的主要接口，一个LuaInterface.Lua类对象就代表了一个Lua解释器（或Lua执行环境），Lua解释器可以同时存在多个，并且它们之间是完全相互独立的。  
                      
  
**3.2、在Windows环境下，Lua脚本文件使用上面提到的源文件 cjsonDemo.lua。**  
  
                        文件所在地址如下：C:\Users\Administrator\Desktop\Redis\LuaScript\cjsonDemo.lua

[IMG_280](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

//源码如下：

local retTable = {}; --最终产生json的表

--顺序数值

local intDatas = {};

intDatas[1] = 100;

intDatas[2] = "100";

--数组

local aryDatas = {};

aryDatas[1] = {};

aryDatas[1]["键11"] = "值11";

aryDatas[1]["键12"] = "值12";

aryDatas[2] = {};

aryDatas[2]["键21"] = "值21";

aryDatas[2]["键22"] = "值22";

--对Table赋值

retTable["键1"] = "值1";

retTable[2] = 123;

retTable["int\_datas"] = intDatas;

retTable["aryDatas"] = aryDatas;

return retTable;

[IMG_281](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

         **3.3、测试代码如下：**

[IMG_282](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

using LuaInterface; // 引入的dll

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CSharpPinvokeLuaDemo

{

class Program

{

static void Main(string[] args)

{

// 新建一个Lua解释器，每一个Lua实例都相互独立, 一个global全局域

Lua lua = new Lua();

//---------------------------------------------------

// Lua的索引操作[]可以创建、访问、修改global域

lua["age"] = 20;

lua["name"] = "Mr.huang";

string luaCode = "print(\"This is Lua code\")";

lua.DoString(luaCode);//执行lua脚本代码

object[] result=lua.DoFile(@"C:\Users\Administrator\Desktop\Redis\LuaScript\cjsonDemo.lua");//执行lua脚本文件，这里我直接用了绝对定位

double age = (double)lua["age"];

Console.WriteLine("age = {0}", age);

Console.WriteLine("width = {0}", lua["width"]);

Console.ReadKey();

}

}

}

[IMG_283](http://www.cnblogs.com/PatrickLiu/p/javascript:void(0);)

**3.4、有可能抛出的异常：FileLoadException，异常内容：其他信息: 混合模式程序集是针对“v2.0.50727”版的运行时生成的，在没有配置其他信息的情况下，无法在 4.0 运行时中加载该程序集。**  
  
                 该问题解决不是很难，只要在配置文件里增加一点配置就能解决。配置代码如下：

<startup useLegacyV2RuntimeActivationPolicy="true">

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5.2" />

<supportedRuntime version="v2.0.50727"/>

</startup>

              在原来.NET2.0，.NET3.5的时候，由于程序运行环境本质还是.NET2.0，而到了.NET4.0由于整个程序集的版本更新，以前使用.NET2.0所编写的程序集与.NET4.0的程序集继续互操作的时候就会出现上面所说的兼容性问题。通过MSDN，我们可以知道，startup配置节中的useLegacyV2RuntimeActivationPolicy属性是在.NET4.0中新增的，默认是false，表示：使用默认的 .NET Framework 4 激活策略，该激活策略将加载 .NET Framework 4 通过使用公共语言运行时 (CLR) 版本 4 所创建的程序集，以及 CLR 早期版本通过使用受支持的低于版本 4 的最高 CLR 版本所创建的程序集。  
  
             现在如果当程序在.NET4.0环境下要使用.NET2.0及.NET3.5的程序时就必须将useLegacyV2RuntimeActivationPolicy设置为true，同时还要注意，需要在startup配置节的字节中添加supportedRuntime配置节，并指定为“v4.0”，表示使用.NET4.0运行时来运行程序。  
  
  
**四、结束**  
  
       今天就写到这里了，就给今天做一个总结吧。今天主要完成了lua-cjson在Linux下的安装和使用，当然也有出现问题的解决。同时也测试了在Linux环境下，在Lua脚本中使用cjson工具类完成table类型的解析。最后也测试了一下如何使用C#来调用lua脚本，并执行。但是有一个事情，还没做，就是在Windows环境下，如何在Lua脚本里使用lua-cjson来解析类型，里面涉及到的内容很多，一篇文章无法写完，只好把相关的东西放到下一片文章吧。