**JSON**

JavaScript Object Notation

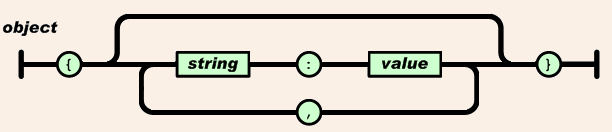
拓薪教育

刘洋

1. 初始JSON
   1. JSON介绍(熟练)

* JSON 指的是 JavaScript 对象表示法（JavaScript Object Notation）
* JSON 是轻量级的**文本数据交换格式**
* JSON 独立于语言：JSON 使用 Javascript语法来描述数据对象，但是 JSON 仍然独立于语言和平台。JSON 解析器和 JSON 库支持许多不同的编程语言。
* JSON 具有自我描述性，更易理解
  1. JSON格式
     1. 键值对集合

“名称/值”对的集合（A collection of name/value pairs）。



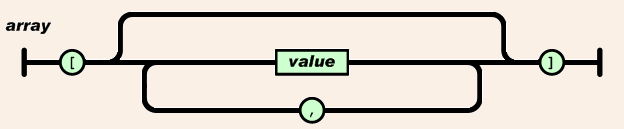
对象是一个无序的“‘名称/值’对”集合。一个对象以“{”开始，“}”结束。每个“名称”后跟一个“:”,“‘名称/值’ 对”之间使用“,”分隔。

示例：var json = {“name”:”Jack”,”age”:90,”Marray”:true};

{"name":"Jack","Marray":true,"age":90}

* + 1. 有序列表

值的有序列表（An ordered list of values）。在大部分语言中，它被理解为数组（array）。



数组是值（value）的有序集合。一个数组以“[”开始，“]”结束。值之间使用“,”分隔。

示例：var json = [“Jack”,”Rose”,”Tom”,89,true,false];

1. FastJson (精通)
   1. Java对象转JSON字符串/JSON对象
      1. Map

|  |
| --- |
| ***private static void* map2JSON(){** Map**<*String*,*Object*> *map* = *new* HashMap<>();  *map*.put("id",*1*);  *map*.put("name","张三");  *map*.put("birthday",*new* Date());  *String jsonString* = *JSON*.*toJSONString*(*map*);  *JSONObject jsonObject* = (*JSONObject*) *JSON*.*toJSON*(*map*);  *System*.*out*.println(*jsonString*);  *System*.*out*.println(*jsonObject*); }** |

{"birthday":1548642714656,"name":"张三","id":1}

* + 1. List

|  |
| --- |
| ***private static void* list2JSON(){** List**<*String*> *list* = *new* ArrayList<>();  *list*.add("a");  *list*.add("b");  *list*.add("c");  *String jsonString* = *JSON*.*toJSONString*(*list*);  *JSONArray jSONArray* = (*JSONArray*) *JSONArray*.*toJSON*(*list*);  *System*.*out*.println(*jsonString*);  *System*.*out*.println(*jSONArray*); }** |

["a","b","c"]

* + 1. JavaBean

|  |
| --- |
| ***private static void* bean2JSON(){  *Person p* = *new* Person(*1*,"张三",*new* Date());  *String jsonString* = *JSON*.*toJSONString*(*p*);  *JSONObject jsonObject* = (*JSONObject*) *JSON*.*toJSON*(*p*);  *System*.*out*.println(*jsonString*);  *System*.*out*.println(*jsonObject*); }** |

{"birthday":1548643135123,"id":1,"name":"张三"}

* 1. JSON字符串/JSON对象转Java对象
     1. Map

|  |
| --- |
| ***private static void* JSON2Map(){  *JSONObject jsonObject* = *new* JSONObject();  *jsonObject*.put("a",*1*);  *jsonObject*.put("b","bb");  *jsonObject*.put("c",*new* Date());  *String jsonString* = *jsonObject*.toJSONString();** Map**<*String*,*Object*> *map1* = *JSON*.*parseObject*(*jsonString*,** Map**.*class*);  *System*.*out*.println(*map1*); }** |

{a=1, b=bb, c=1548645386767}

* + 1. List

|  |
| --- |
| ***private static void* JSON2Array(){  *JSONArray jsonArray* = *new* JSONArray();  *jsonArray*.add("a");  *jsonArray*.add("b");  *jsonArray*.add("c");  *jsonArray*.add("d");  *String jsonString* = *jsonArray*.toJSONString();** List**<*String*> *list* = *JSONArray*.*parseArray*(*jsonString*, *String*.*class*);  *System*.*out*.println(*list*); }** |

[a, b, c, d]

* + 1. JavaBean

|  |
| --- |
| ***private static void* JSON2Person(){  *JSONObject jsonObject* = *new* JSONObject();  *jsonObject*.put("id",*1*);  *jsonObject*.put("name","张三");  *jsonObject*.put("birthday",*new* Date());  *String jsonString* = *jsonObject*.toJSONString();  *System*.*out*.println(*jsonString*);  *Person person* = *JSONObject*.*parseObject*(*jsonString*, *Person*.*class*);  *System*.*out*.println(*person*); }** |

{"birthday":1548645582282,"name":"张三","id":1}

Person{id=1, name='张三', birthday=Mon Jan 28 11:19:42 CST 2019}

* + 1. 复杂数据类型

|  |
| --- |
| ***private static void* JSON2ListInMap(){** // 创建一个map对象,模拟前台请求返回的数据  Map**<*String*,*Object*> *result* = *new* HashMap<>();  *result*.put("status","success");  *result*.put("code","200");** List**<*Person*> *list* = *new* ArrayList<>();  *list*.add(*new* Person(*1*,"张三",*new* Date()));  *list*.add(*new* Person(*2*,"李四",*new* Date()));  *list*.add(*new* Person(*3*,"王五",*new* Date()));  *list*.add(*new* Person(*4*,"赵六",*new* Date()));  *result*.put("data",*list*);** // 将map对象转换成JSON字符串  ***String jsonString* = *JSON*.*toJSONString*(*result*);  *System*.*out*.println(*jsonString*);** // 对JSON字符串进行解析,获取Map对象   // 直接转换的话,data内的集合为JSONArray,集合内存储的为JSONObject  Map**<*String*,*Object*> *map* = *JSONObject*.*parseObject*(*jsonString*,** Map**.*class*);  *System*.*out*.println(*map*);** // 先把JSON字符串转换成JsonObject  ***JSONObject jsonObject* = (*JSONObject*)*JSONObject*.*parse*(*jsonString*);** // 获取JSONArray对象  ***JSONArray jsonArray* = (*JSONArray*) *jsonObject*.get("data");** // 解析JSONArray对象获取Person集合  List**<*Person*> *array* = *JSONArray*.*parseArray*(*jsonArray*.toJSONString(), *Person*.*class*);** // 重新赋值到map  ***map*.put("data",*array*);  *System*.*out*.println(*map*); }** |

{"code":"200","data":[{"birthday":1548646719839,"id":1,"name":"张三"},{"birthday":1548646719839,"id":2,"name":"李四"},{"birthday":1548646719839,"id":3,"name":"王五"},{"birthday":1548646719839,"id":4,"name":"赵六"}],"status":"success"}

{code=200, data=[{"birthday":1548646719839,"name":"张三","id":1},{"birthday":1548646719839,"name":"李四","id":2},{"birthday":1548646719839,"name":"王五","id":3},{"birthday":1548646719839,"name":"赵六","id":4}], status=success}

{code=200, data=[Person{id=1, name='张三', birthday=Mon Jan 28 11:38:39 CST 2019}, Person{id=2, name='李四', birthday=Mon Jan 28 11:38:39 CST 2019}, Person{id=3, name='王五', birthday=Mon Jan 28 11:38:39 CST 2019}, Person{id=4, name='赵六', birthday=Mon Jan 28 11:38:39 CST 2019}], status=success}

1. JS操作JSON对象(熟悉)

|  |
| --- |
| **<*script* type="text/javascript" >** // 创建一个数组并赋值  ***var list* = *new Array*();  *list*[*0*] = "a";  *list*[*1*] = "b";  *list*[*2*] = "c";** // 创建一个JSON对象并赋值  ***var obj* = {  "id" : *1*,  "name" : "张三",  "data" : *list*,  "createDate" : *new Date*()  }** // 控制台打印JSON对象  ***console*.**log**(*obj*);** *alert***(*obj*.*id*);** *alert***(*obj*.*name*);** *alert***(*obj*.*createDate*);** // 获取JSON对象中的数组  ***var data* = *obj*.*data*;** *alert***(*data*);** //遍历数组  ***for*(*var i*=*0*;*i*<*data*.**length**;*i*++){** *alert***(*data*[*i*]);  } </*script*>** |