

Espressif IoT SDK Json 命名准则

Status	Released
Current version	V0.1
Author	Liu Han
Completion Date	2014.6.19
Reviewer	Wu Jiangang
Completion Date	2014.6.19

☐ CONFIDENTIAL

☐ INTERNAL

☒ PUBLIC

版本信息

日期	版本	撰写人	审核人	修改说明
2014.3.7	0.1	刘 晗		初稿

免责声明和版权公告

本文中的信息，包括供参考的 URL 地址，如有变更，恕不另行通知。

文档“按现状”提供，不负任何担保责任，包括对适销性、适用于特定用途或非侵权性的任何担保，和任何提案、规格或样品在他处提到的任何担保。本文档不负任何责任，包括使用本文档内信息产生的侵犯任何专利权行为的责任。本文档在此未以禁止反言或其他方式授予任何知识产权使用许可，不管是明示许可还是暗示许可。

Wi-Fi 联盟成员标志归 Wi-Fi 联盟所有。

文中提到的所有商标名称、商标和注册商标均属其各自所有者的财产，特此声明。

版权归© 2014 乐鑫信息技术有限公司所有。保留所有权利。

目录

版本信息.....	2
目录.....	3
1.前言	4
2.准则	5
2.1.一般准则	5
2.1.1.注释	5
2.1.2.使用双引号	5
2.1.3.扁平化数据 VS 结构化数据	5
2.2.属性名准则	6
2.2.1.属性名格式	6
2.2.2.命名冲突	6
2.2.3.属性值准则	7

1.前言

本文主要介绍基于ESP_IOT lib库创建的JSON APIs而提供的指导性准则和建议。总体来讲，JSON APIs应遵循JSON.org上的规范。这份准则澄清和标准化了特定情况，适用于基于REST风格的API的JSON请求和响应。

CONFIDENTIAL

2. 准则

为了更好地实现这份的规范目的，下面几项需要说明：

属性(property) - JSON对象内的键值对(name/value pair)

属性名(property name) - 属性的名称

属性值(property value) - 分配给属性的值

示例：

```
{  
  // 一组键值对称作一个“属性”.  
  "propertyName": "propertyValue"  
}
```

2.1. 一般准则

2.1.1. 注释

JSON对象中不应该包含注释。

2.1.2. 使用双引号

如果(某个)属性需要引号，则必须使用双引号。所有的属性名必须在双引号内。字符类型的属性值必须使用双引号。其它类型值(如布尔或数字)不应该使用双引号。

2.1.3. 扁平化数据 VS 结构化数据

JSON中的属性元素应以扁平化方式呈现，不能为了方便而将数据任意分组。

但是，在某些情况下，结构化的方式对开发人员来讲更有意义。比如描述单一结构的一批属性，因它被用来保持结构层次，所以是有意义的，遇到这些情况应当慎重考虑。示例：

扁平化方式：

```
{  
  "ssid": "tenda_837R",  
  "password": ""  
}
```

结构化方式：

```
{  
  "Requese": {  
    "Station": {  
      "Connect": {  
        "ssid": "tenda_837R",  
        "password": ""  
      }  
    }  
  }  
}
```

```
}  
}
```

2.2.属性准则

2.2.1.属性名准则

选择有意义的属性名。必须遵循以下准则：

属性名应该是具有定义语义的有意义的名称；

属性名必须是驼峰式的，ASCII码字符串；

首字符必须是不能包含数字；

当一个属性名有子属性，而无属性值时，首字符大写；

当一个属性名无子属性，但有属性值时，小写；

随后的其他字符可以包含数字。

示例：

```
{  
  "Response": {  
    "status": 0  
  }  
}
```

2.2.2.命名冲突

新的属性可在将来被添加进保留列表中。如果存在命名冲突，可通过选择新的属性名或者版本化来解决这个问题。示例：

```
{  
  "version": "1.0",  
  "message": {  
    "sucessful": true,  
    "information": "flash.bin",  
    "data": ["sensor", "device"]  
  }  
}
```

如果希望将来把information列为保留字，可以通过下面两件事情来达成。

1. 选一个不同的名字

```
{  
  "version": "1.0",  
  "Message": {  
    "sucessful": true,  
    "information": "flash.bin",  
    "information0": "irom.bin",  
    "Data ": ["sensor", "device"]  
  }  
}
```

2. 在主版本上重新命名属性



```
{  
  "version": "2.0",  
  "Message ": {  
    "sucessful": true,  
    "information": "irom.bin",  
    "Data": ["sensor", "device"]  
  }  
}
```

2.2.3.属性值准则

JSON.org上的标准准确的说明了哪些类型的数据可以作为属性值。属性值必须是Unicode的booleans(布尔),数字(numbers),字符串(strings),对象(object),数组(arrays)或null等。

JavaScript表达式是不被接受的。APIs应该支持该准则,并为某个特定的属性选择最合适的数据类型。示例:

```
{  
  "message": null, // null  
  "sucessful": false, // boolean  
  "status": 42, // number  
  "name": "Bart", // string  
  "response": { }, // object  
  "data": [ ] // array  
}
```