自动化测试框架使用说明

测试程序：module\_test

测试用例脚本

module\_test的运行需要两份文件配合，一份是配置文件，默认为module\_test.conf定义了测试环境，测试配置等信息。另一份为测试用例脚本。

Ps: module\_test路径上午特殊要求，可在任意路径下运行

## 命令格式

module\_test的运行格式为

Usage: ./module\_test -hv | [-f configure\_file] -t test\_file

-f --config configure file default value is "module\_test.conf"

-t --test module test file

-h --help help information

-v –version version information

-f 为指导配置文件，如果不指定，则默认使用当前路径的module\_test.conf文件，如果该文件不存在将报错

-t 为准备运行的测试脚本

Ps：配置文件和测试脚本若不在module\_test同级目录下，则需指定相对路径或绝对路径

## 配置文件格式定义：

配置文件的json格式定义如下

{

// 测试平台配置信息，该项必须存在，用于配置测试平台的通讯信息的

"locale":{

"test\_service\_type":"control\_server", //测试平台内行， 必须存在于翻译表service中,测试时的使用服务类型，它决定了连接到哪些平台上进行测试

"ip":"xxx.xxx.xxx.xxx", // 测试平台本地ip

"domain":"xxxx", // 测试域

"group\_ip":"xxx.xxx.xxx.xxx", // 组播ip

"group\_port":xxxx // 组播端口

},

// 翻译表

// "trans"项 必须存在,它用于转换输入和输出的，当使用字符串作为数字输入，并指定了适用的转换表后，平台将尝试使用该转换表将字符串翻译成数字。同样的，如果输出返回项设置了转换表，平台将尝试使用该转换表将数字转换为字符串，如果翻译表中的数字和字符不是一一对应的关系，翻译会以找到的一项匹配项作为翻译基准（由于存储是乱序的，所以其实是随机的）

"trans":{

// 参数翻译表，必须存在

"param":

{

"domain":0,

"node\_name":1,

"node\_type":2,

....

},

// 服务翻译表，必须存在

"service":

{

"data\_server" : 1,

"control\_server" : 2,

"node\_client" : 3,

"storage\_server" : 4,

...

},

// 服务器状态表

"service\_mode":

{

"stop":0,

"warring":1,

"error":2,

"running":3

}

},

// 请求定义

// "request\_desc" 项，必须存在

// 请求接口描述表

// 数据类型描述 支持: uint, float, string, uint\_array, float\_array, string\_array, uint\_array\_array, float\_array\_array, string\_array\_array

// 基本结构为

// "query\_service\_type" :{ // 请求名

// "code": , // 请求代码

// "param\_list":{ 请求参数列表

// "type":{ 参数名,参数名必须在"param"翻译表中存在，

// "data\_type":"uint", 数据类型

// "trans\_tab":"service" 适用翻译表让参数或数据更直观

// "optional":true 代表是否可选，如无该项，则默认为False，即为必填项

// }

// },

// "respone\_list":{

// "type":{ // 返回项名,参数名必须在"param"翻译表中存在，

// "data\_type":"uint\_array", 数据类型

// "trans\_tab":"service" 适用翻译表 可以让参数或数据更直观

// }，

// "count":{// 返回项名,参数名必须在"param"翻译表中存在，

// "data\_type":"uint\_array\_array",数据类型

// }

// }

// }

// 如下例所示

"request\_desc":

{

"query\_service\_type":

{

"code": 92,

"param\_list":{},

"respone\_list":{

"type":{

"data\_type":"uint\_array","trans\_tab":"service"

},

"count":{"data\_type":"uint\_array\_array"}

}

},

...

}

## 测试脚本格式定义：

测试脚本的配置如下

{

// 测试任务

// "test\_task"关键字，必须存在，必须定义为数组

// 每个成员必须为以下形式

// {

// "request":"query\_service", 请求的命令

// "param":{"type":"control\_server", "group":"default"}, 请求的参数,参数可以为变量，但如果该变量未被设值，将报错

// "respone":{ 对respone的处理

// "server":{"set":["$abc"]} "server" 为respone中的"server"项，见参数定义，支持以下设定

"include"为检查该项值是否存在于include所列出的值序列中，否则将报错，终止测试

// }

// },

}

// 如下例所示

{

"test\_task":[

{

"request":"create\_server\_room",

"param":{"name":"new\_server\_room\_0",

"display":"room\_0\_display",

"description":"room\_0\_description!"

},

"respone":{

"uuid":{"set":["$room"]}

}

},

{

"request":"query\_server\_room",

"param":{},

"respone":{

"uuid":{"include":["$room"]},

"name":{"include":["new\_server\_room\_0"]}

}

}

]

}