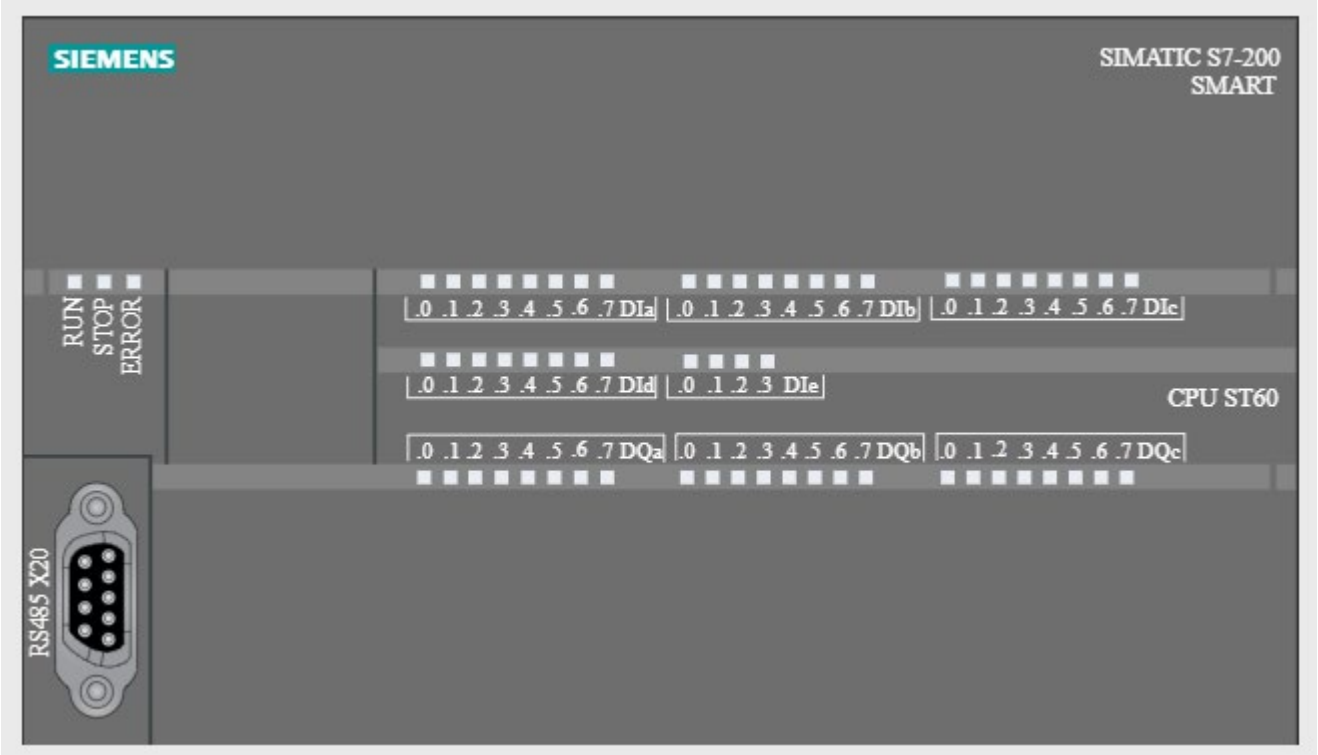
S7-200 SMART Web API 开发手册

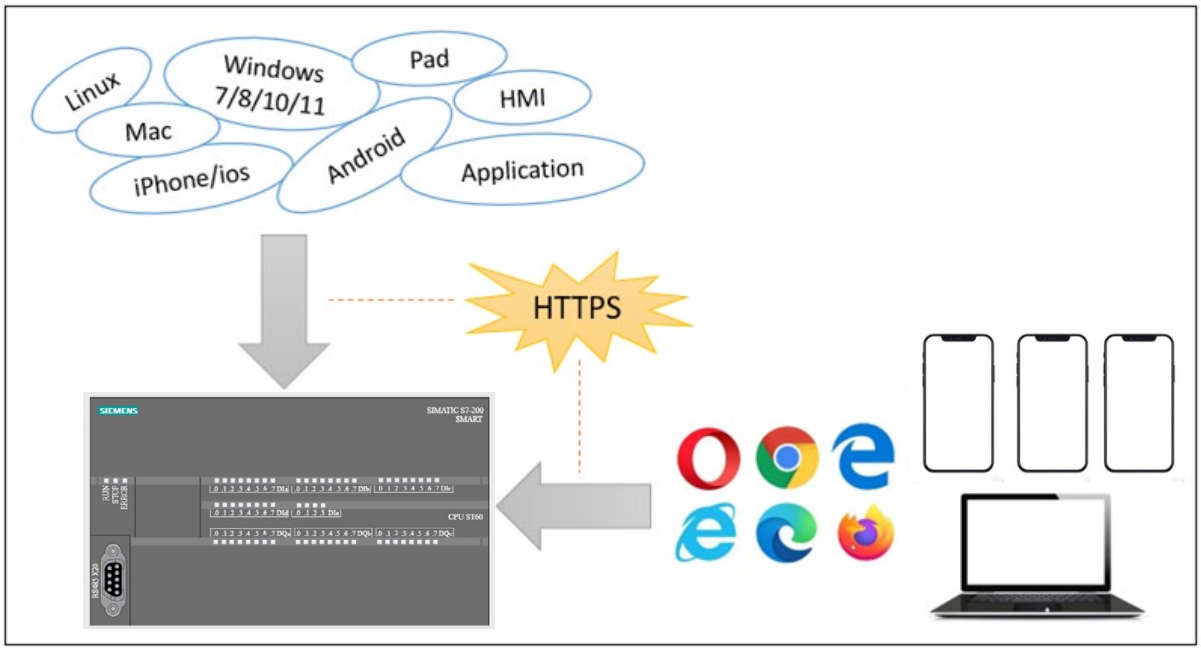


Contents

西门子 SIMATIC S7-200 SMART SR/ST 0AA1 系列 CPU 从固件 2.7 版本开始，都将支持 Web API！
让我们一起来看看这个新特性！

1 什么是 Web API？

在我们理解什么是 Web API 之前，我们先看看什么是 API：在计算机编程领域，API (application programming interface)是一类提供指定功能，提供特殊能力的函数集合，可以用来操作数据，系统和设备。
Web API 是一个很宽泛的概念，如同名字所示，是通过 Web 服务的 HTTP 协议通道来提供 API 能力的特性。
SIMATIC S7-200 SMART SR/ST 0AA1 系列 CPU 上，所实现的 Web API 是一个标准的 JSON-RPC 规范的 API 特性，遵循 JSON-RPC 2.0 规范.



JSON (java-script object notation): 一个轻量级的数据交换格式，简洁清晰，易于读取，易于解析。
JSON-RPC 2.0: 一种开放性 API 框架，也是一个轻量级、无状态的远程调用协议。

2 Web API 能做什么?

Web APIs 提供了最基本的远程访问 CPU 数据的能力，根据 Web 用户权限控制，你可以远程读写 CPU 的数据。

因此，你可以将 Web API 能力集成到你的自定义网页，终端脚本，应用程序中以管理你的 CPU。

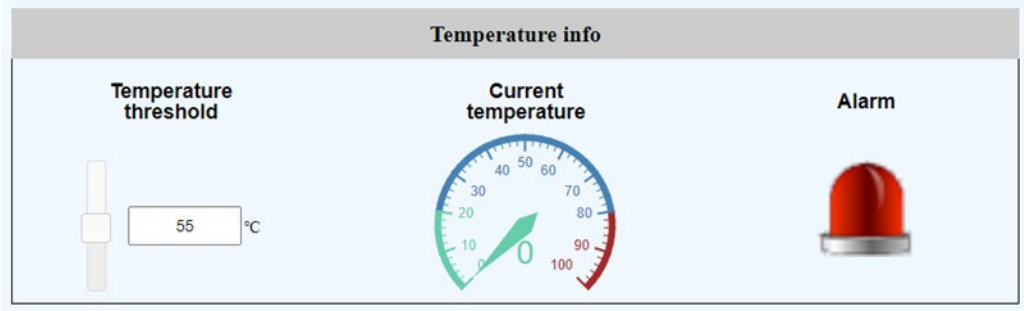
2.1 实时控制

- 读写 CPU 数据。
- 调整 CPU 时间。

能够：

- 你可以在 CPU 运行时读写 CPU 主要的数据区，因此你能够在运行时动态调整用户程序参数。
- 你可以手动触发、移除一些信号，与用户程序相配套时能够简化用户程序逻辑，比如，清除业务告警灯，调整温控限。
- 你可以同步时钟，比如，由于 CPU 电池没电导致的时钟丢失。

更多...



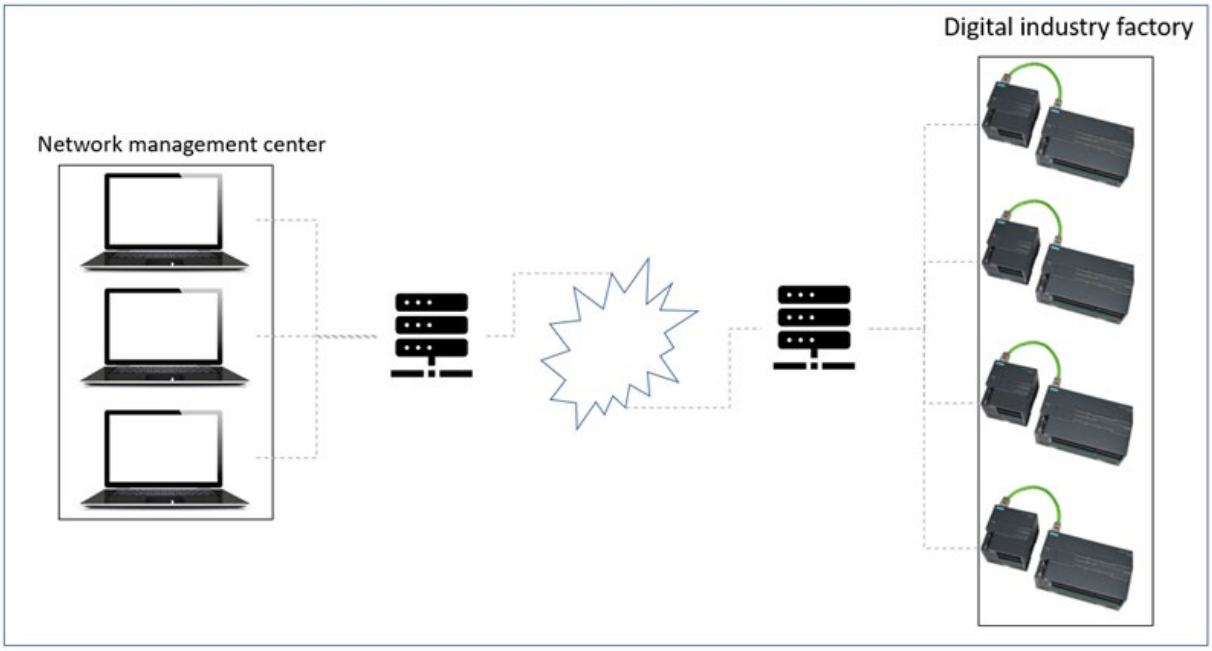
2.2 网络管理

- 远程监控。
- 中心化监控。
- 多终端监控。

能够：

- 你可以在远程修复一些错误问题、逻辑流程，而不需要去实地站点。
- 你可以和其他管理员，通过配置不同的用户和不同的权限，共享管理同一个 CPU。
- 你可以同时远程管理多个 CPU。

更多...



3 配置 Web API

要在一个全新的 CPU 上使用 Web API，你需要通过 STEP 7-Micro/WIN 做一些必要的 Web 配置。

提示：你需要确保将你的 CPU 和 STEP 7-Micro/WIN 都升级到 V2.7 或更高版本！

1. 通过 STEP 7-Micro/WIN 连接到 CPU

需要确保你的 STEP 7-Micro/WIN（PC）和 CPU 之间的通信是正常的。

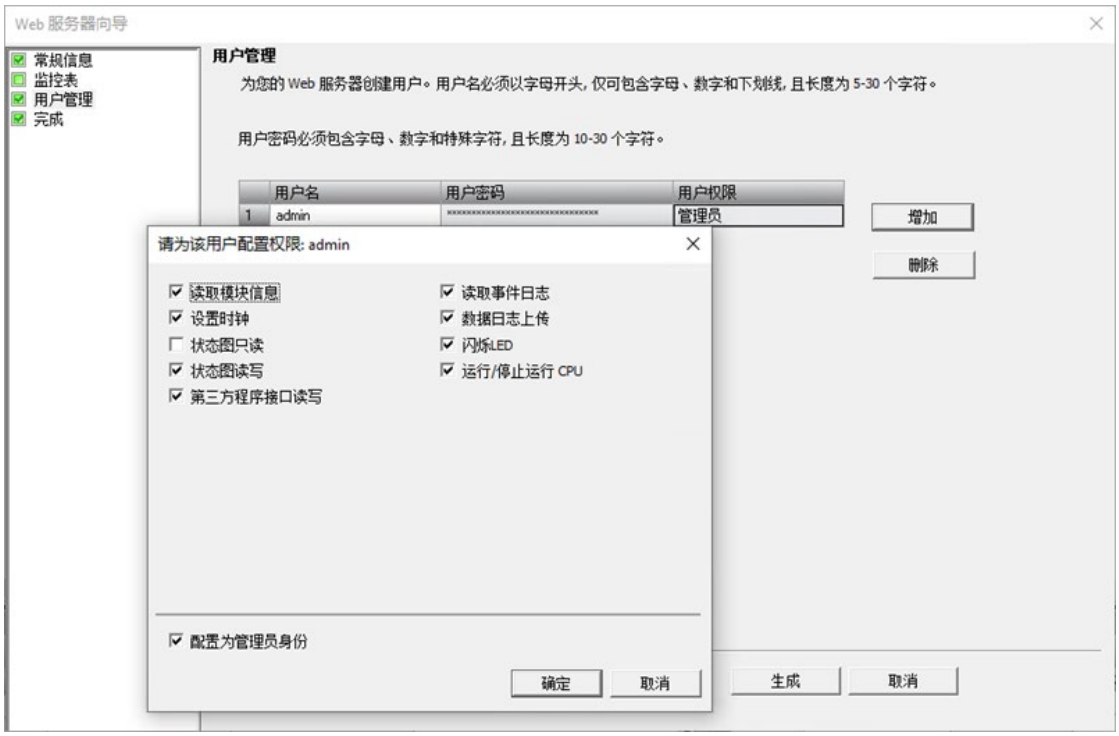
2. 激活 Web Server 和 Web API

1) 在 STEP 7-Micro/WIN 中打开 Web server 向导。
2) 在向导中，选中“激活 Web 服务器”。
3) 输入要连接的 CPU 模块的 IP 地址和站名称（可选）。
4) 选中“第三方程序接口和用户自定义网页（CPU 数据可读）”，设置用户自定义 Web 页面 CPU 数据的读取权限。

3. 配置 Web server 用户

1) 点击导航栏中的 “用户管理”，进入用户管理页面。
2) 点击 “增加” 添加新的 Web 用户。你可以最多添加 4 个 Web 用户。

3) 在弹出框中输入用户名和密码，在权限配置页面中配置你需要的权限。
4) 点击 “生成” 保存配置。



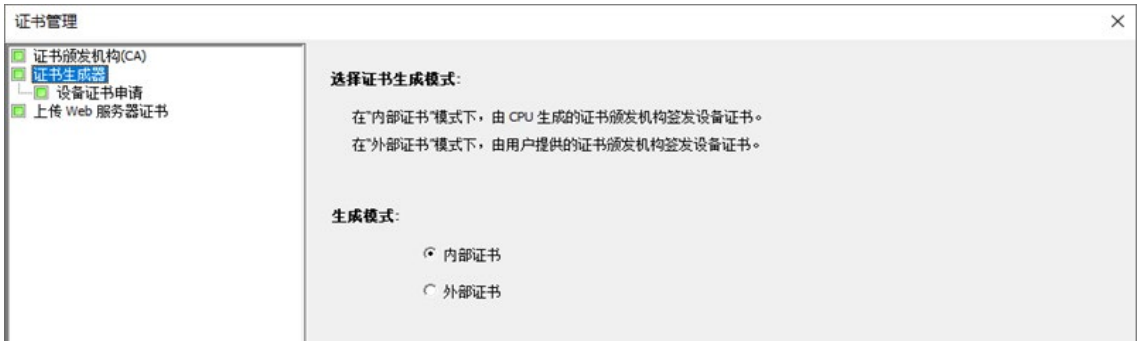
4. 下载配置到 CPU

5. 添加 TLS 证书

要使用 S7-200 SMART CPU 的 Web server 服务，你需要先在 STEP 7-Micro/WIN SMART 的证书管理向导中配置 TLS 证书。

STEP 7-Micro/WIN SMART 提供两种证书配置方案：

- “外部证书” 方案，设备证书由用户提供的证书颁发机构（CA）签名。
- “内部证书”方案，设备证书由 CPU 生成的证书颁发机构 (CA) 签名。



+

4 Web API 基本功能

在使用 Web API 之前，你需要先了解一下基本知识。

4.1 API 列表

当前 Web API 的版本是 V1.0.0。

列表	描述
login	用户登录，成功登录之后可以访问 CPU。
logout	用户登出，成功登出之后用户的临时 cookie 将失效。
get_permission	用户登录成功后，可用于获取用户的权限列表。
read	从 CPU 中读取数据。
write	向 CPU 中写入数据。
browse	访问 CPU 支持的 API 的列表。

4.2 API 使用建议

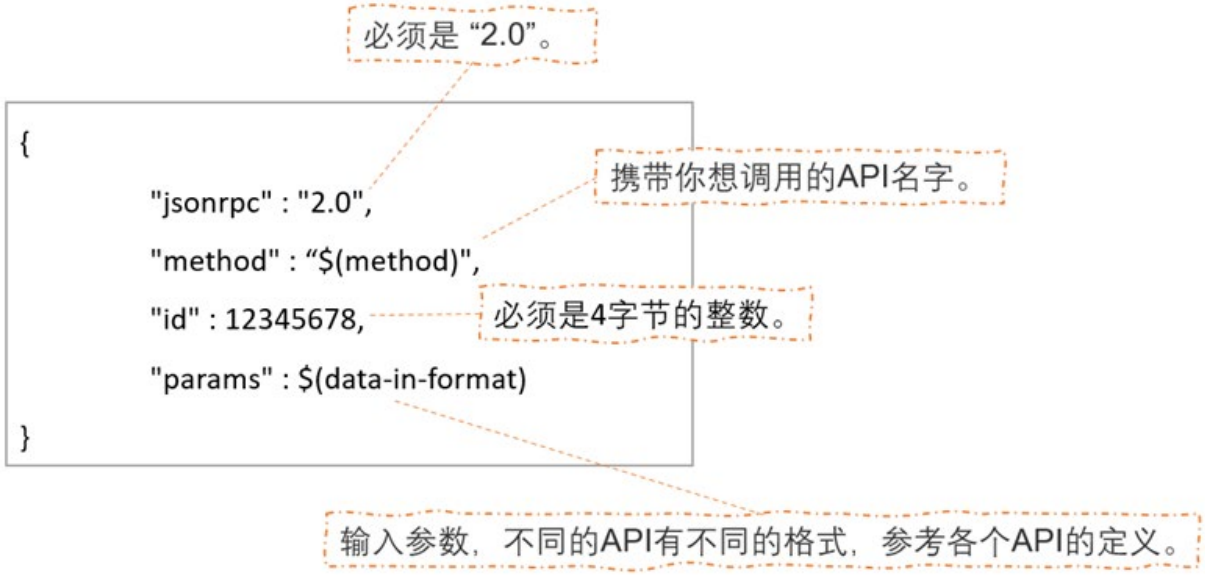
建议	建议描述
控制调用频率	推荐调用 API 最快一秒一次。 Web API 的处理会占用部分 CPU 资源：网络、TLS 加解密、JSON 解析、业务检查等，调用过快会导致请求阻塞。
压缩 JSON 文本	推荐使用压缩后的 JSON 文本。 JSON 压缩之后会变得更小，能提高信息传输效率。
高效利用 API 请求	推荐在一次 API 调用中，尽可能多的携带数据。 Web 请求支持很大的缓冲区，同时一次请求最多能支持到 32 个数据的读写处理，所以降低请求次数的频率，提升单次请求数据的效率。
使用 HTTP 长连接	推荐使用 HTTP 长连接。 新建 TLS 的连接会占用很多 CPU 资源，建立连接时间也会较长，在既有连接上传输数据的效率远高于新建连接。

4.3 API 通用模板

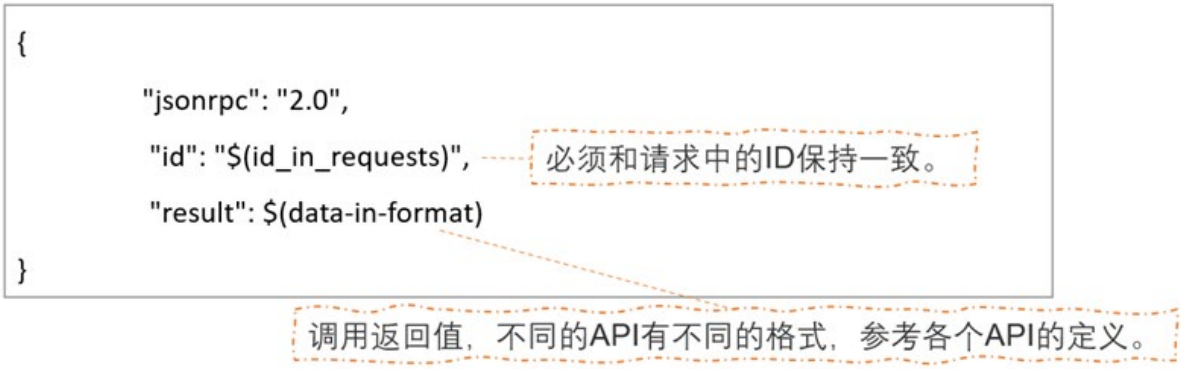
1. 所有 API 数据都需要使用 UTF-8 编码。
2. 整体 HTTP 负载长度不能超过 15 KB。

下面是 JSON-RPC 2.0 API 的通用模板。

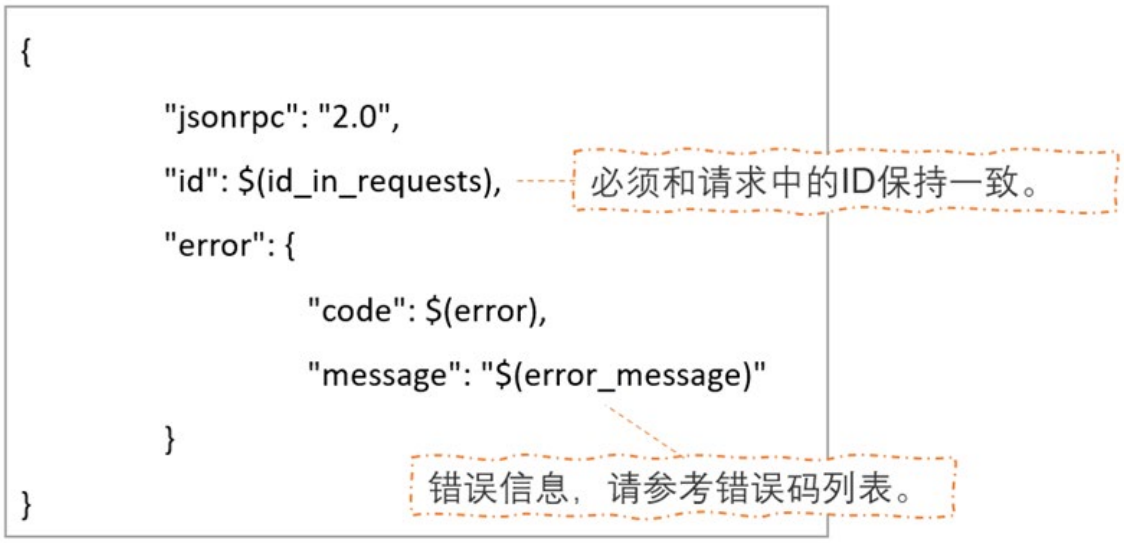
4.3.1 JSON-RPC 请求模板



4.3.2 JSON-RPC 处理成功模板



4.3.3 JSON-RPC 处理失败模板



4.4 API 使用限制

1. HTTP 的 URL 必须是"https://ip_address/Web_api"。
2. 所有的 HTTP 数据必须使用 UTF-8 来编码。
3. HTTP 头部的 Content-Type 必须是"application/json"。
4. HTTP 的方法必须是 POST。
5. 当前 Web API V1.0.0 版本不支持批量 API 请求模式。

5 login

调用登录接口时 CPU 会检查用户信息，如果登录成功会返回一个随机生成的用户 cookie，该 cookie 用于后续的 API 交互。

5.1 使用说明

a.需要提前配置好 Web 用户。
b. 重复登录同一用户，会导致该用户之前的 cookie 失效。
c. 已登陆用户半个小时内无操作会被自动登出。

5.2 API 请求格式

参数	说明
username	Web 用户名。
password	用户密码使用 SHA512 加密格式化后的 ASCII 字符串。

```
{
        "jsonrpc": "2.0",
        "method": "login",
        "id": 12345678,
        "params": {
                "username": "$(user_name)",
                "password": "$(user_password)"
        }
}
```

用户名。

将用户密码使用SHA512加密格式化的ASCII字符串。

5.3 API 回复格式

5.3.1 登录成功

参数	说明
cookie	成功登录之后的随机生成的用户标识，在后续请求时用来标识该用户。 用户标识将在主动登出或无操作半小时后自动失效。

```
{
        "jsonrpc": "2.0",
        "id": "$(id_in_requests)",
        "result": {
                "code": 0,
                "message": "Success.",
                "cookie": "$(cookie)"
        }
}
```

成功登录之后的随机生成的用户标识。

5.3.2 登录失败

```
{
        "jsonrpc": "2.0",
        "id": "$(id_in_requests)",
        "error": {
                "code": $(error),
                "message": "$(error_message)"
        }
}
```

5.4 使用样例

测试须知：

- 这是一个 Windows Powershell 脚本，所以只能在 Powershell 终端上使用。
- 脚本使用了 curl 命令，测试机器上需要集成 curl 工具。
- 脚本使用了假想的用户和 IP，你需要修改成真实的用户和 IP 地址。

```
$http_url = "https://192.168.2.1/Web_api"
$http_content_type = "Content-Type: application/json; charset=utf-8"
$http_body ='
{
   "jsonrpc": "2.0",
   "method": "login",
   "id": 12345678,
   "params": {
      "username": "admin",
      "password": "1ce3848ec82f8d11f751499b741743acb51e5ba8ce8a06acac22cabf76abb832d*******124567e64bcfde6"
   }
}
'
function do_Webapi_request
{
   $tmp_file = [System.IO.path]::GetTempFileName()
   $http_body | Out-File -Encoding ascii $tmp_file
   curl.exe -v -k `
      -H $http_content_type `
      -X POST $http_url `
      -d @$tmp_file
      del $tmp_file
}
do_Webapi_request
```

6 logout

主动登出用户。

6.1 使用说明

a. 登出一个不存在或不在线的用户，系统无报错。
b. 必须将用户 cookie 标识设置到 HTTP 头部的 cookie 字段。

6.2 API 请求格式

```
{
        "jsonrpc": "2.0",
        "method": "logout",
        "id": 12345678,
        "params": null
}
```

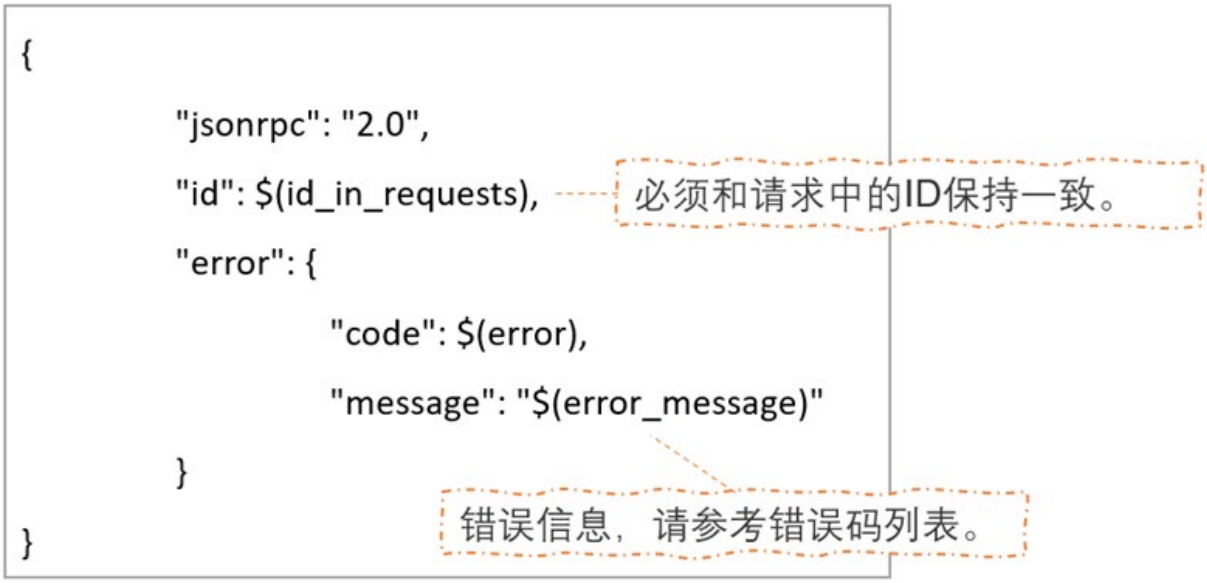
必须为null, API会通过cookie来查找登出用户。

6.3 API 回复格式

6.3.1 登出成功

```
{
        "jsonrpc": "2.0",
        "id": "$(id_in_requests)",
        "result": {
                "code": 0,
                "message": "Success."
        }
}
```

6.3.2 登出失败



6.4 使用样例

测试须知：

- 这是一个 Windows Powershell 脚本，所以只能在 Powershell 终端上使用。
- 脚本使用了 curl 命令，测试机器上需要集成 curl 工具。
- 脚本使用了假想的用户和 IP，你需要修改成真实的用户和 IP 地址。

```
$http_url = "https://192.168.2.1/Web_api"
$http_content_type = "Content-Type: application/json; charset=utf-8"
$user_token = "Session-Id=YWRtaW4uKdPEsyUwbVFqXY+HWZqrhQ=="
$http_body ='
{
   "jsonrpc": "2.0",
   "method": "logout",
   "id": 12345678,
   "params": null
}
'
function do_Webapi_request
{
   $tmp_file = [System.IO.path]::GetTempFileName()
   $http_body | Out-File -Encoding ascii $tmp_file
   curl.exe -v -k `
      -H $http_content_type `
      -X POST $http_url `
      -b $user_token `
      -d @$tmp_file
   del $tmp_file
}
do_Webapi_request
```

7 get_permission

获取用户的权限列表。

7.1 使用说明

a. 用户必须已经成功登录，并且已获取用户标识 cookie。
b. 必须将用户 cookie 标识设置到 HTTP 头部的 cookie 字段。

7.2 API 请求格式

```
{
        "jsonrpc": "2.0",
        "method": "get_permission",
        "id": 12345678,
        "params": null
}
```

必须为null, API会通过cookie来查找用户。

7.3 API 回复格式

7.3.1 获取权限成功

参数	说明
permission	用户权限列表。

7.4 使用样例

测试须知：

- 这是一个 Windows Powershell 脚本，所以只能在 Powershell 终端上使用。
- 脚本使用了 curl 命令，测试机器上需要集成 curl 工具。
- 脚本使用了假想的用户和 IP，你需要修改成真实的用户和 IP 地址。

```
$http_url = "https://192.168.2.1/Web_api"
$http_content_type = "Content-Type: application/json; charset=utf-8"
$user_token = "Session-Id=YWRtaW4uKdPEsyUwbVFqXY+HWZqrhQ=="
$http_body ='
{
   "jsonrpc": "2.0",
   "method": "get_permission",
   "id": 12345678,
   "params": null
}
'
function do_Webapi_request
{
   $tmp_file = [System.IO.path]::GetTempFileName()
   $http_body | Out-File -Encoding ascii $tmp_file
   curl.exe -v -k `
      -H $http_content_type `
      -X POST $http_url `
      -b $user_token `
      -d @$tmp_file
   del $tmp_file
}
do_Webapi_request
```

8 read

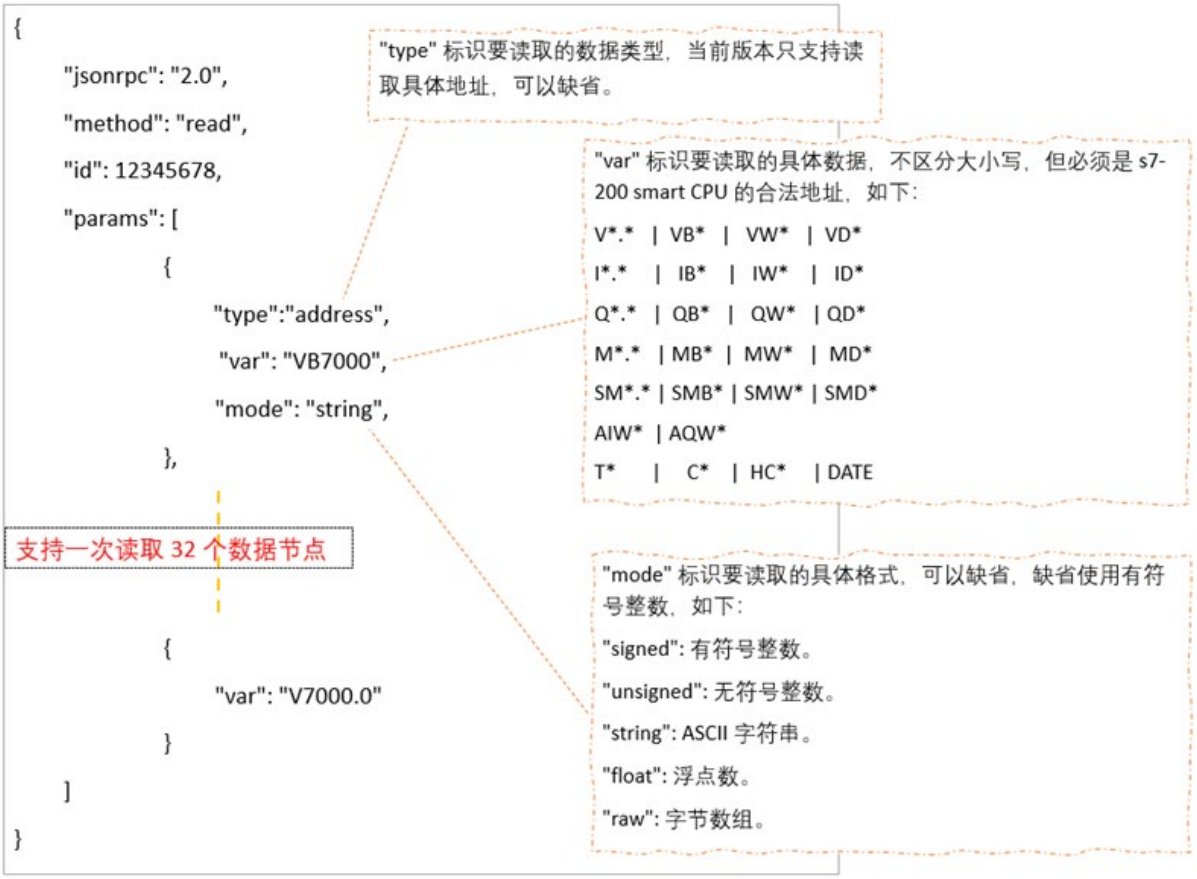
读取指定数据区。

8.1 使用说明

a. 用户必须已经成功登录，并且已获取用户标识 cookie。
b. 必须将用户 cookie 标识设置到 HTTP 头部的 cookie 字段。
c. 一次请求最多可以读取 32 个数据节点。
d. 可以发送空请求用于读取系统状态。

8.2 API 请求格式

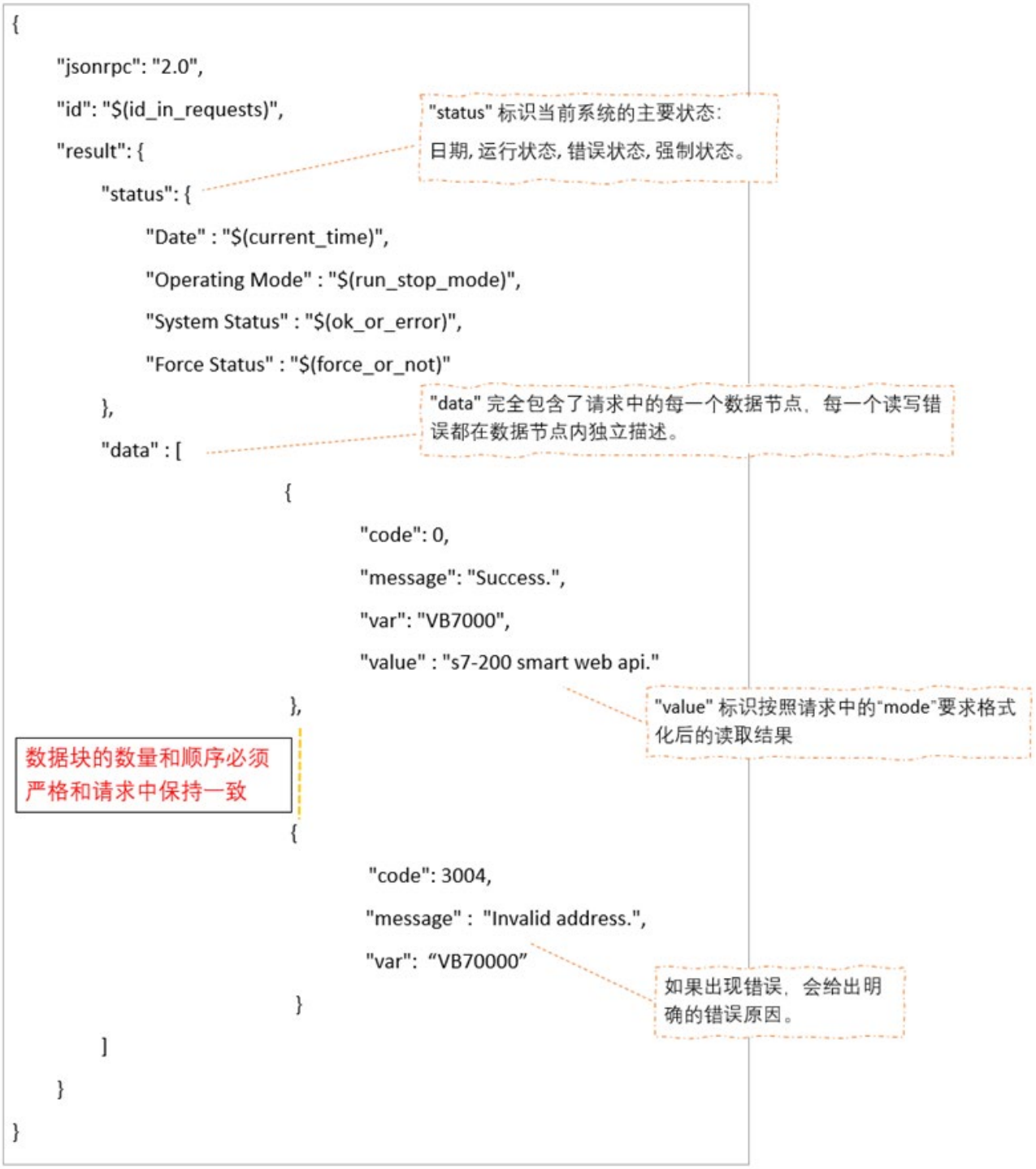
参数	说明
params	JSON 数组，每一个节点都标识了要读取的一个数据，最多可同时读取 32 个数据节点。
type	"type" 标识要读取的数据类型，当前版本只支持读取具体地址，可以缺省。
var	"var" 标识要读取的具体数据，不区分大小写，但必须是 S7-200 SMART CPU 的合法地址，如下： V*.* , VB* , VW* , VD* I*.* , IB* , IW* , ID* Q*.* , QB* , QW* , QD* M*.* , MB* , MW* , MD* SM*.* , SMB* , SMW* , SMD* AIW* , AQW* T* , C* , HC* , DATE
mode	"mode" 标识要读取的具体格式，可以缺省，缺省使用有符号整数，如下： "signed": 有符号整数。 "unsigned": 无符号整数。 "string": ASCII 字符串。 "float": 浮点数。 "raw": 字节数组。



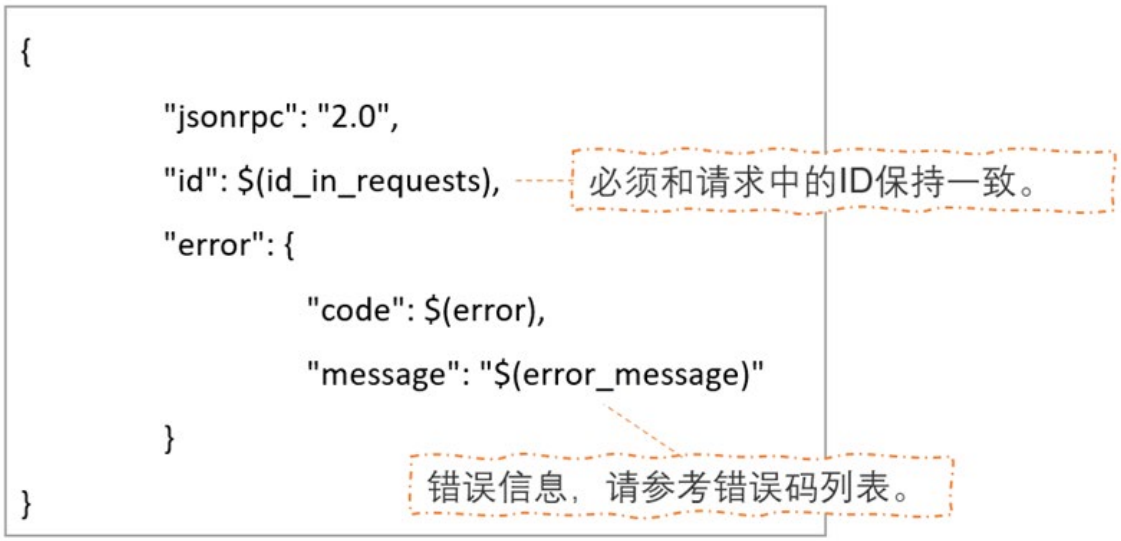
8.3 API 回复格式

8.3.1 读取成功

参数	说明
status	"status" 标识当前系统的主要状态： Date: 时间字符串，格式 "YYYY-MM-DD HH:MM:SS". Operating Mode: RUN/STOP 状态. System Status: OK, Error. Force Status: Forced, Not Forced.
data	在〞result〞字段中的一个 JSON 数组。 数组内节点的顺序和 API 请求中的顺序严格保持一致。
- code & message	"code" & "message"标识读取错误信息。
- var	"var" 内容和请求输入严格保持一致。
- value	"value" 标识按照请求中的"mode"要求格式化后的读取结果。


{
"jsonrpc": "2.0",
"id": "$(id_in_requests)",
"result": {
"status": {
"Date" : "$(current_time)",
"Operating Mode" : "$(run_stop_mode)",
"System Status" : "$(ok_or_error)",
"Force Status" : "$(force_or_not)"
},
"data" : [
{
"code": 0,
"message": "Success.",
"var": "VB7000",
"value" : "s7-200 smart web api."
},
{
"code": 3004,
"message" : "Invalid address.",
"var": "VB70000"
}
]
}
}
"status" 标识当前系统的主要状态:
日期, 运行状态, 错误状态, 强制状态。
"data" 完全包含了请求中的每一个数据节点，每一个读写错误都在数据节点内独立描述。
"value" 标识按照请求中的"mode"要求格式化后的读取结果
数据块的数量和顺序必须严格和请求中保持一致
如果出现错误，会给出明确的错误原因。

8.3.2 读取失败



8.4 使用样例

测试须知：

- 这是一个 Windows Powershell 脚本，所以只能在 Powershell 终端上使用。
- 脚本使用了 curl 命令，测试机器上需要集成 curl 工具。
- 脚本使用了假想的用户和 IP，你需要修改成真实的用户和 IP 地址。

```
$http_url = "https://192.168.2.1/Web_api"
$http_content_type = "Content-Type: application/json; charset=utf-8"
$user_token = "Session-Id=YWRtaW4uKdPEsyUwbVFqXY+HWZqrhQ=="
$http_body ='
{
   "jsonrpc": "2.0",
   "method": "read",
   "id": 123,
   "params": [
      {
         "var": "VB1000",
         "mode": "string"
      },
      {
         "var": "VB1000"
      },
      {
         "var": "DATE"
      }
   ]
}
'
function do_Webapi_request
{
   $tmp_file = [System.IO.path]::GetTempFileName()
   $http_body | Out-File -Encoding ascii $tmp_file
   curl.exe -v -k `
      -H $http_content_type `
      -X POST $http_url `
      -b $user_token `
      -d @$tmp_file
   del $tmp_file
}
do_Webapi_request
```

9 write

向 CPU 指定区域写入数据。

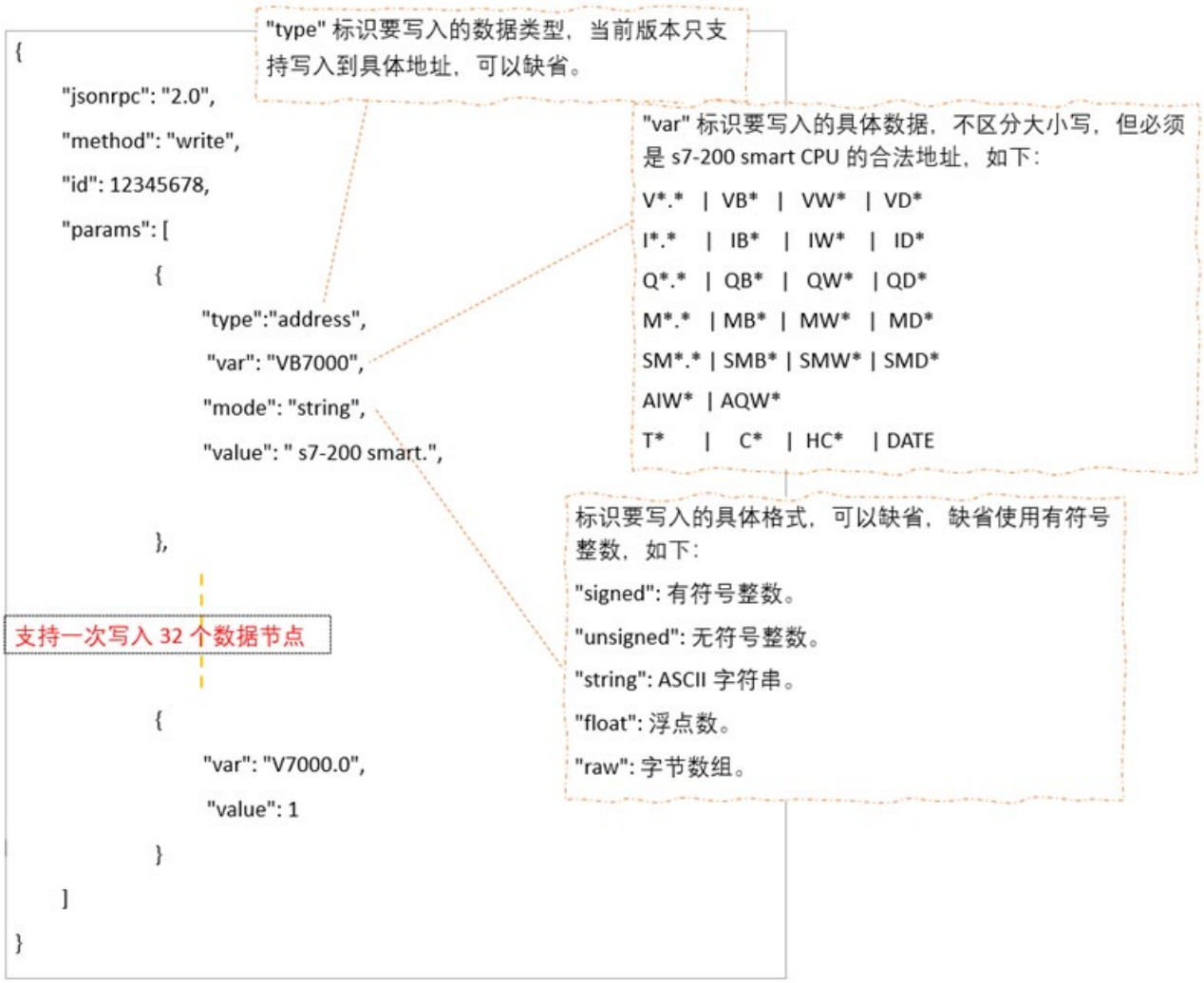
9.1 使用说明

a. 用户必须已经成功登录，并且已获取用户标识 cookie。
b. 必须将用户 cookie 标识设置到 HTTP 头部的 cookie 字段。
c. 一次请求最多可以写入 32 个数据节点
d. 只能在 RUN 状态下写入。
e. 可以发送空请求用于读取系统状态。

9.2 API 请求格式

参数	说明
params	JSON 数组，每一个节点都标识了要读取的一个数据，最多可同时写入 32 个数据节点。
type	"type" 标识要写入的数据类型，当前版本只支持写入到具体地址，可以缺省。
var	"var" 标识要写入的具体数据，不区分大小写，但必须是 S7-200 SMART CPU 的合法地址，如下： V*.* , VB* , VW* , VD* I*.* , IB* , IW* , ID* Q*.* , QB* , QW* , QD* M*.* , MB* , MW* , MD* SM*.* , SMB* , SMW* , SMD* AIW* , AQW* T* , C* , HC* , DATE

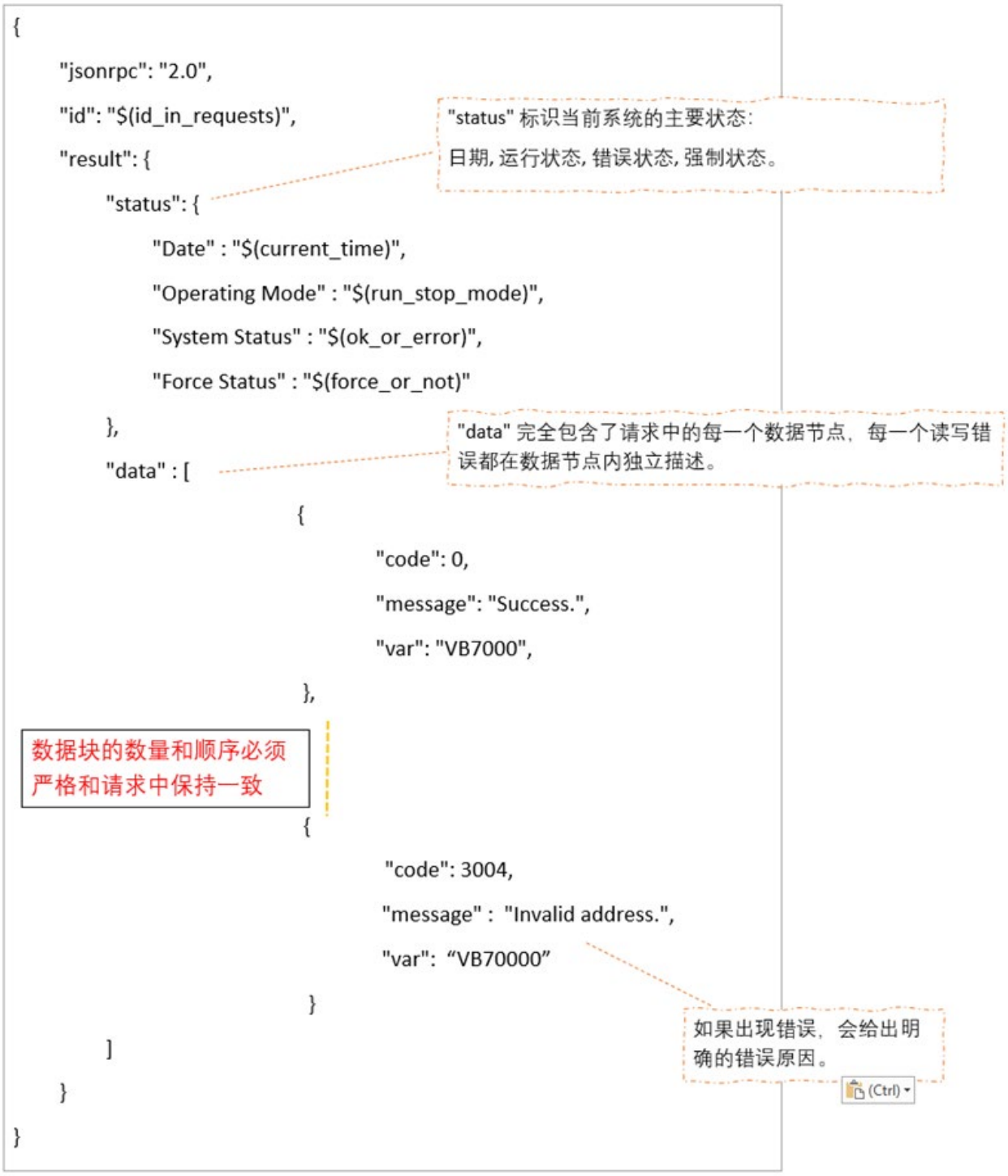
mode	"mode" 标识要写入数据的具体格式，可以缺省（缺省使用"signed"），缺省使用有符号整数，如下： "signed": 有符号整数。 "unsigned": 无符号整数。 "string": ASCII 字符串。 "float": 浮点数。 "raw": 字节数组。
value	"value" 标识按照请求中的“mode”要求格式化后的待写入数据。



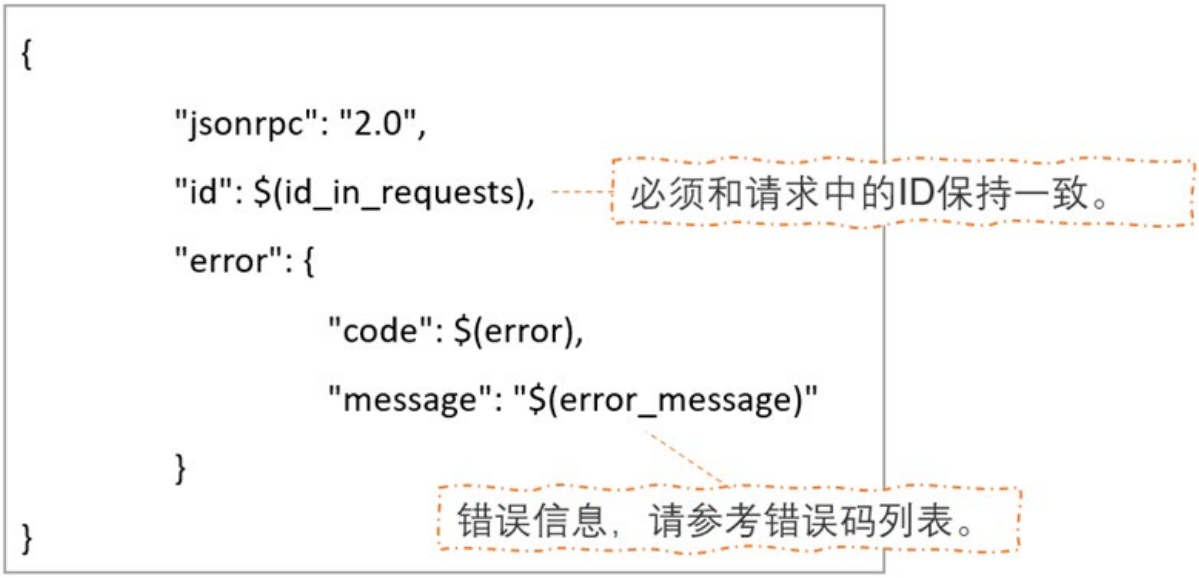
9.3 API 回复格式

9.3.1 写入成功

参数	说明
status	"status" 标识当前系统的主要状态： Date: 时间字符串，格式 "YYYY-MM-DD HH:MM:SS". Operating Mode: RUN/STOP 状态. System Status: OK, Error. Force Status: Forced, Not Forced.
data	在〞result〞字段中的一个 JSON 数组。 数组内节点的顺序和 API 请求中的顺序严格保持一致。
- code & message	"code" & "message"标识读取错误信息。
- var	"var" 内容和请求输入严格保持一致。


{
"jsonrpc": "2.0",
"id": "$(id_in_requests)",
"result": {
"status": {
"Date" : "$(current_time)",
"Operating Mode" : "$(run_stop_mode)",
"System Status" : "$(ok_or_error)",
"Force Status" : "$(force_or_not)"
},
"data" : [
{
"code": 0,
"message": "Success.",
"var": "VB7000",
},
{
"code": 3004,
"message" : "Invalid address.",
"var": "VB70000"
}
]
}
}
"status" 标识当前系统的主要状态:
日期, 运行状态, 错误状态, 强制状态。
"data" 完全包含了请求中的每一个数据节点，每一个读写错误都在数据节点内独立描述。
数据块的数量和顺序必须严格和请求中保持一致
如果出现错误，会给出明确的错误原因。
(Ctrl)

9.3.2 写入失败



9.4 使用样例

测试须知：

- 这是一个 Windows Powershell 脚本，所以只能在 Powershell 终端上使用。
- 脚本使用了 curl 命令，测试机器上需要集成 curl 工具。
- 脚本使用了假想的用户和 IP，你需要修改成真实的用户和 IP 地址。

```
$http_url = "https://192.168.2.1/Web_api"
$http_content_type = "Content-Type: application/json; charset=utf-8"
$user_token = "Session-Id=YWRtaW4uKdPEsyUwbVFqXY+HWZqrhQ=="
$http_body ='
{
   "jsonrpc": "2.0",
   "id": 12345678,
   "method": "write",
   "params": [
      {
         "type": "address",
         "var": "VB1000",
         "mode": "String",
         "value": "S7-200 SMART."
      },
      {
         "var": "VD5000",
         "value": 1000
      }
   ]
}
'
function do_Webapi_request
{
   $tmp_file = [System.IO.path]::GetTempFileName()
   $http_body | Out-File -Encoding ascii $tmp_file
   curl.exe -v -k `
      -H $http_content_type `
      -X POST $http_url `
      -b $user_token `
      -d @$tmp_file
   del $tmp_file
}
do_Webapi_request
```

10 browse

浏览 API 列表。

10.1 使用说明

a. 无需登录。

10.2 API 请求格式

```
{
        "jsonrpc": "2.0",
        "method": “browse",
        "id": 12345678,
        "params": null
}
```

10.3 API 回复格式

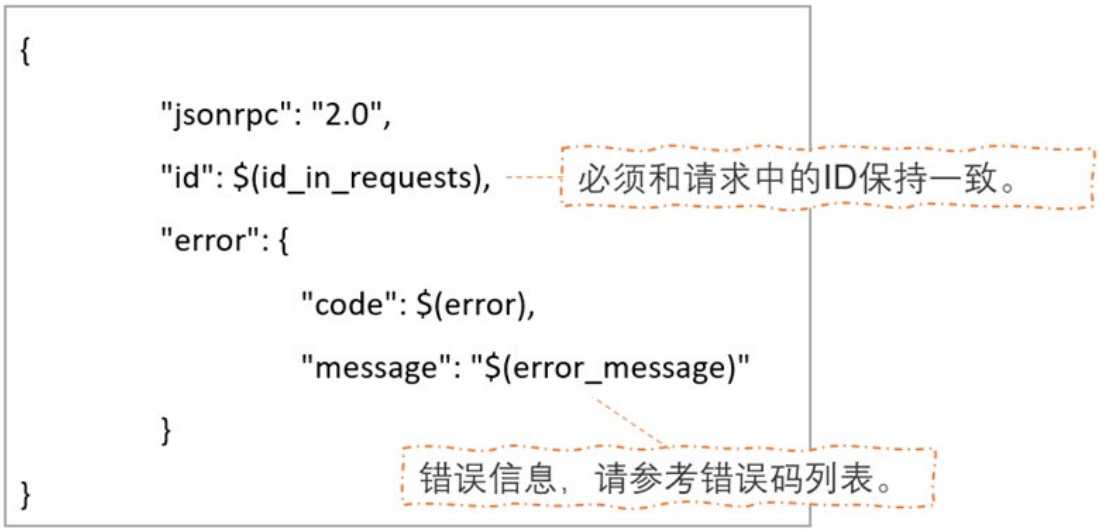
10.3.1 浏览成功

参数	说明
version	"version" 标识当前支持的 Web API 版本。
api	JSON 数组，标识当前支持的 API 列表。

```
{
        "jsonrpc": "2.0",
        "id": "$(id_in_requests)",
         "result": {
                "code": 0,
                "message": "Success." ,
                "version": "$(version_string)." ,
                "api": [
                        "login",
                        "logout",
                        "get_permission",
                        "read",
                        "write",
                        "browse"
        ]
    }
}
```

Api 列表。

10.3.2 浏览失败



10.4 使用样例

测试须知：

- 这是一个 Windows Powershell 脚本，所以只能在 Powershell 终端上使用。
- 脚本使用了 curl 命令，测试机器上需要集成 curl 工具。
- 脚本使用了假想的用户和 IP，你需要修改成真实的用户和 IP 地址。

```
$http_url = "https://192.168.2.1/Web_api"
$http_content_type = "Content-Type: application/json; charset=utf-8"
$http_body ='
{
   "jsonrpc": "2.0",
   "method": "browse",
   "id": 12345678,
   "params": null
}
'
function do_Webapi_request
{
   $tmp_file = [System.IO.path]::GetTempFileName()
   $http_body | Out-File -Encoding ascii $tmp_file
   curl.exe -v -k `
      -H $http_content_type `
      -X POST $http_url `
      -d @$tmp_file
   del $tmp_file
}
do_Webapi_request
```