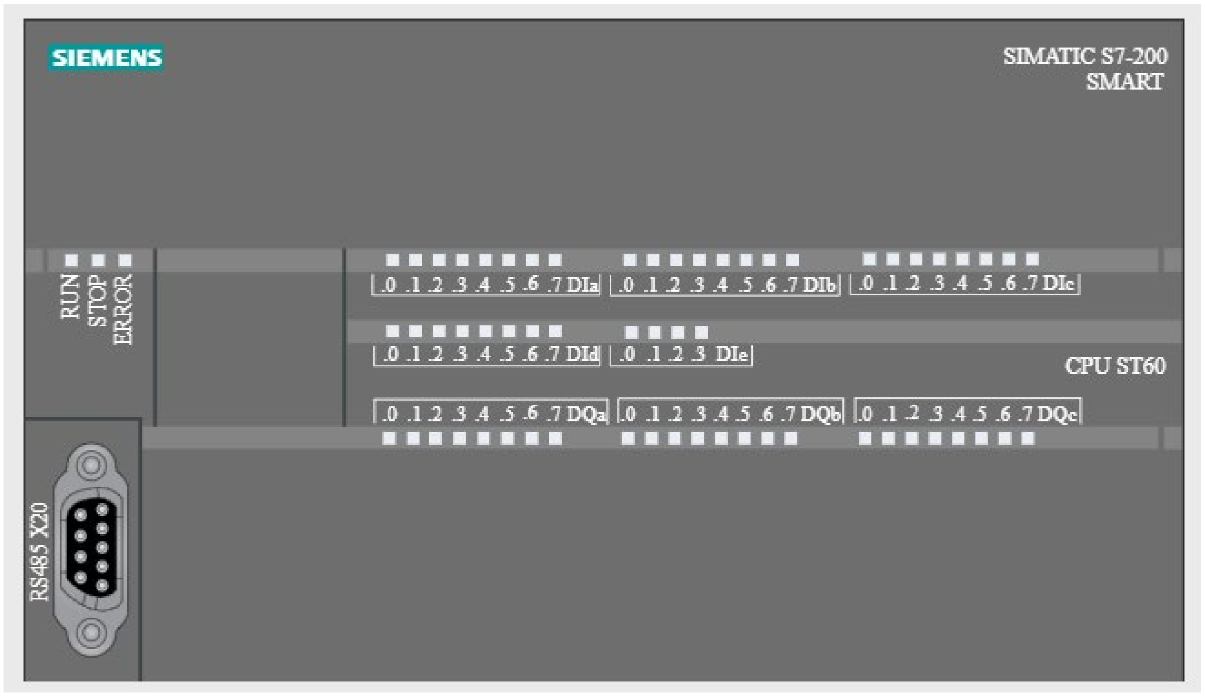
S7-200 SMART Web API development manual

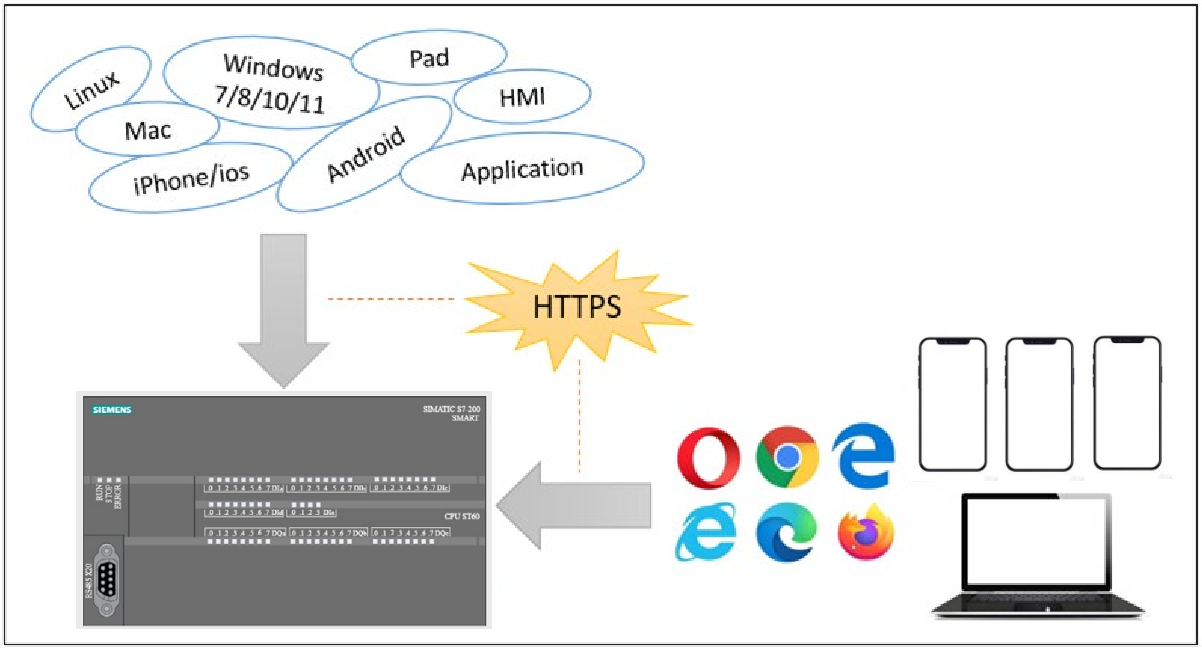

SIEMENS
SIMATIC S7-200 SMART
RUN
STOP
ERROR
.0 .1 .2 .3 .4 .5 .6 .7 DIa
.0 .1 .2 .3 .4 .5 .6 .7 DIb
.0 .1 .2 .3 .4 .5 .6 .7 DIc
.0 .1 .2 .3 .4 .5 .6 .7 DId
.0 .1 .2 .3 DIe
CPU ST60
.0 .1 .2 .3 .4 .5 .6 .7 DQa
.0 .1 .2 .3 .4 .5 .6 .7 DQb
.0 .1 .2 .3 .4 .5 .6 .7 DQc
RS485 X20

Contents

SIMATIC S7-200 SMART SR/ST 0AA1 CPUs support Web API since version 2.7!
Let’s take a look at this new feature!

1.What is Web API?

Before we understand what Web API is, let's see what is an API: In computer programming, an application programming interface (API) is some kind of interface which has a set of functions that allow programmers to access specific features or data of an application, operating system or other services.
Web API, as the name suggests, is an API over the Web server which can be accessed using HTTP protocol, a relatively broad concept.
In SIMATIC S7-200 SMART CPUs series, Web API we implement is a standard JSON-RPC process, specifically follow JSON-RPC 2.0 specification.



JSON (java-script object notation): a lightweight data-interchange format, easily read, write, parse, generate.
JSON-RPC 2.0: a kind of open-api framework, a stateless and light weight remote procedure call protocol.

2.What can Web API do?

Web APIs provides the basic ability to **remotely access CPU data**, according to the authority control system, you can read and write CPU data.
So you can integrate Web API into self-defined Web pages, terminals, applications, or shell scripts to manage your CPU.

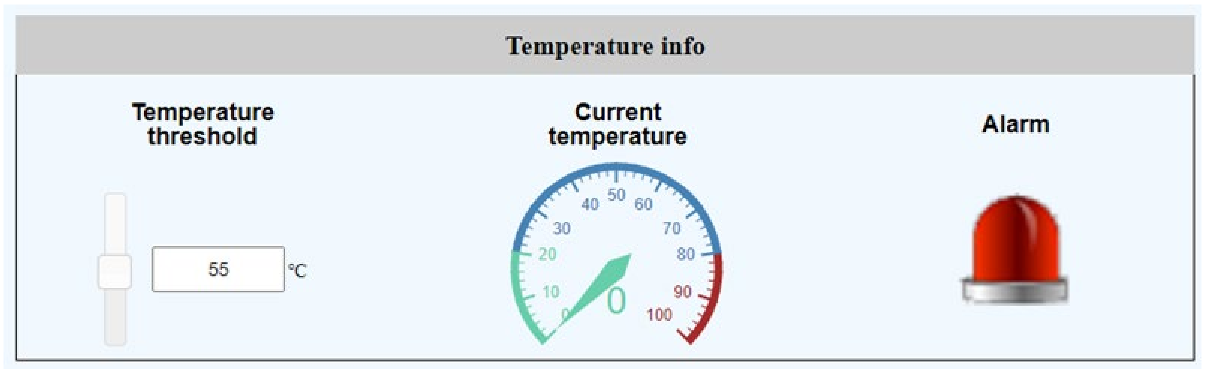
2.1 Real-time Control

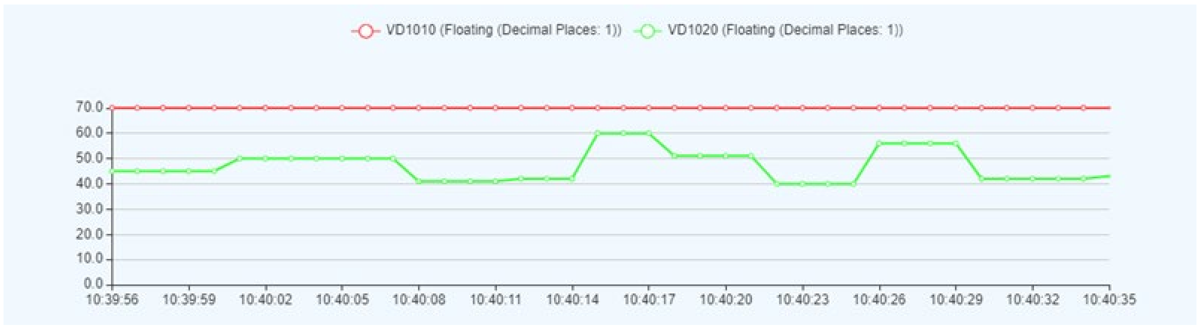
- Read/Write CPU data.
- Adjust CPU time.

your benefits:

- You can read/write CPU's memory runtime, so you can timely adjust data for user program.
- You can trigger/remove signals manually, interact with your program, such as, alarm lights.
- You can synchronize your own clock, such as, dead battery causes time lost.

more...





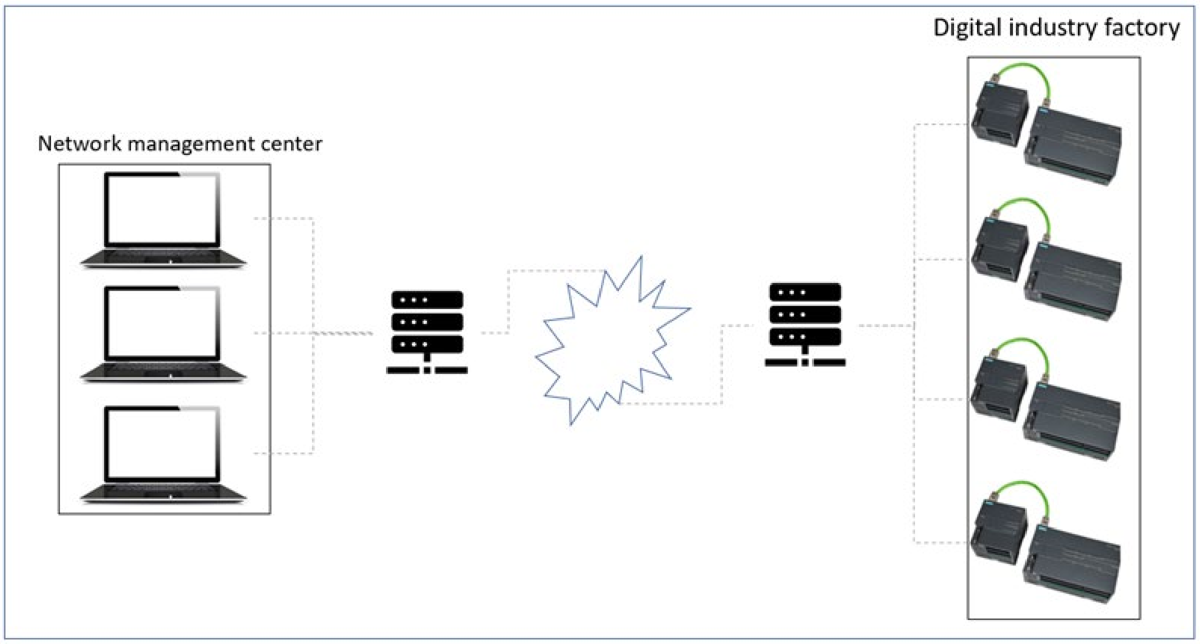
2.2 Network management

- Remote monitoring.
- Centralized monitoring.
- Multi-terminal monitoring.

your benefits:

- You can fix some issue or logical request remotely, so don't need to go to factory area on-site.
- You can share device control with other managers remotely, with different accounts and authorities.
- You can control Multi-devices at the same time remotely.

more...



3.Configure Web API

To use Web API ability in a new CPU, you should do some necessary configuration through STEP 7-Micro/WIN.

Notes: you should surely upgrade both your CPU and STEP 7-Micro/WIN to at least V2.7!

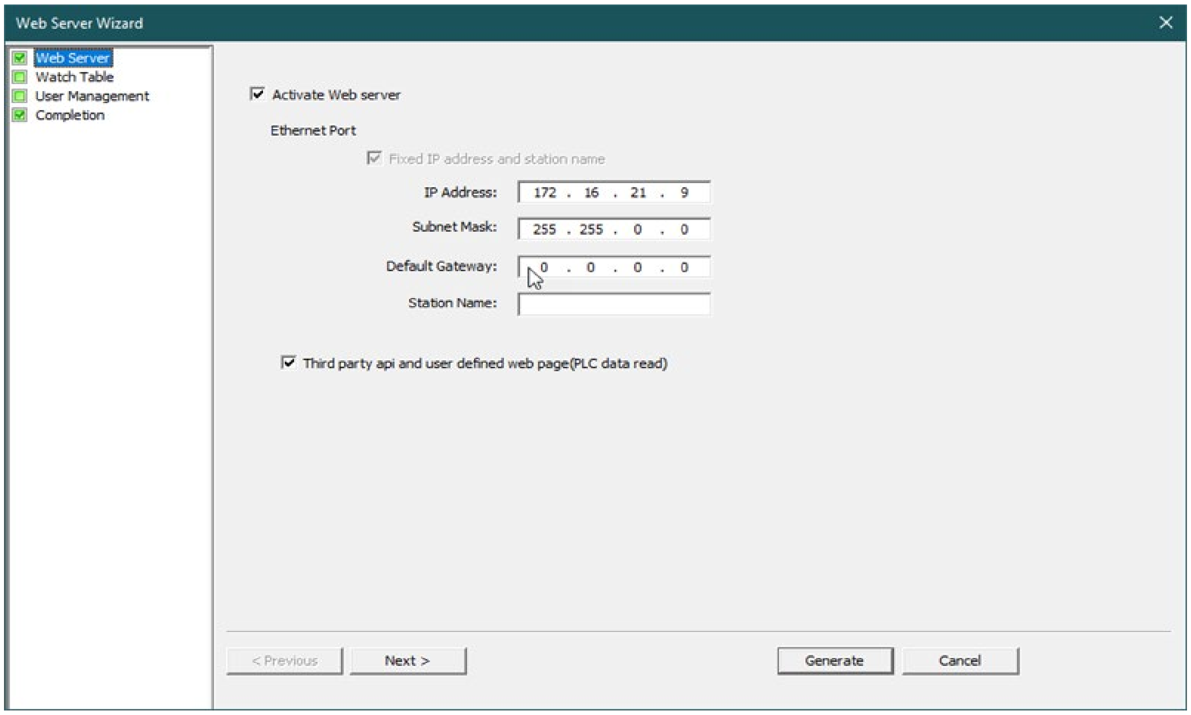
1. Connect to CPU through STEP 7-Micro/WIN.

Ensure that your Communication Interface and PLC connector cable for Ethernet is working.

2. Enable Web server and Web api.

   1) Open the Web server wizard.
   2) In the Web Server window, select "Activate Web Server".
   3) Enter the IP address and station name (optional) of the CPU module to which you intend to connect.

4) Select "Third party api and user defined Web page (PLC data read)" to set read permission of the connected CPU data on user defined Web pages.
5) Click "Next" to continue the Web server configuration.



☑ Activate Web server

Ethernet Port

☑ Fixed IP address and station name

IP Address: 172 . 16 . 21 . 9

Subnet Mask: 255 . 255 . 0 . 0

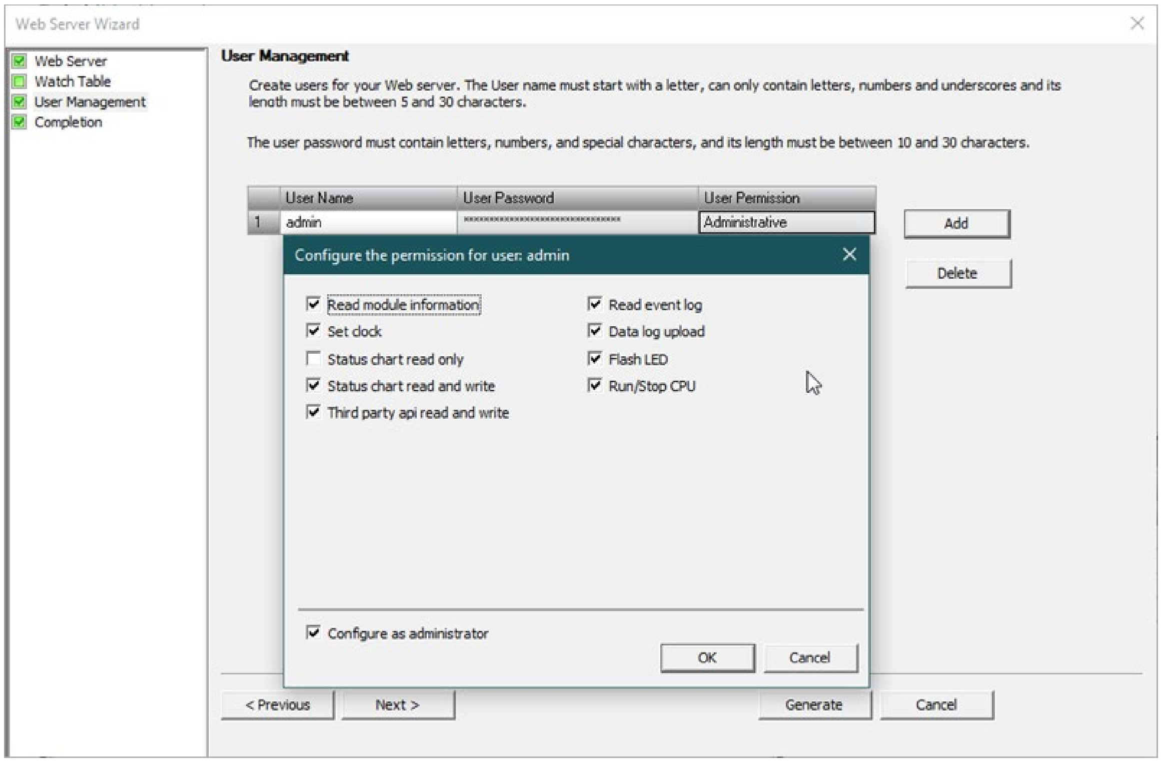
Default Gateway: 0 . 0 . 0 . 0

Station Name:

☑ Third party api and user defined web page(PLC data read)

3. Configuring Web server users.

1) Click "User Management" in the navigation panel to enter the User management page.
2) Click "Add" to add a line for each Web server user. You can add up to four Web server users.
3) Enter usernames and passwords for the user logins that you want to provide.
4) Click "Generate" to save the configuration.


Web Server Wizard

- ☑ Web Server
- ☐ Watch Table
- ☑ User Management
- ☑ Completion

User Management

Create users for your Web server. The User name must start with a letter, can only contain letters, numbers and underscores and its length must be between 5 and 30 characters.

The user password must contain letters, numbers, and special characters, and its length must be between 10 and 30 characters.

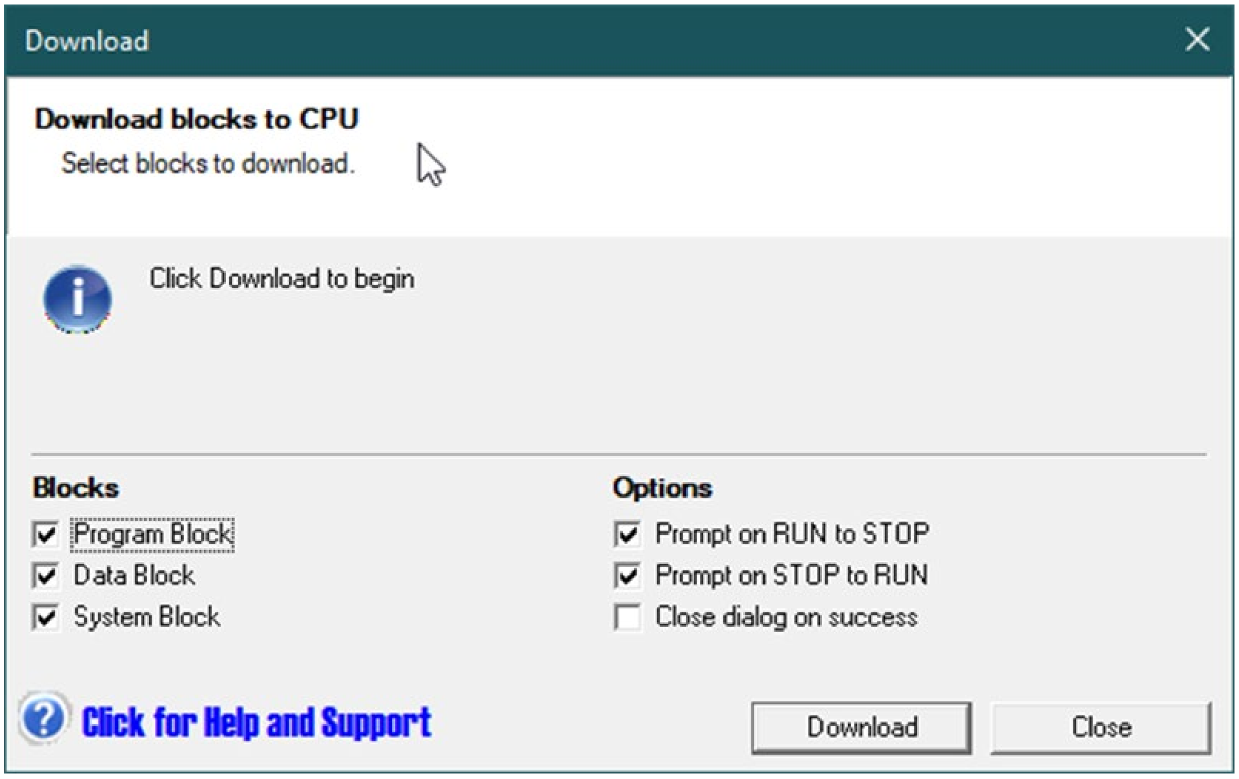
	User Name	User Password	User Permission
1	admin	******************************	Administrative

Configure the permission for user: admin

- ☑ Read module information
- ☑ Set clock
- ☐ Status chart read only
- ☑ Status chart read and write
- ☑ Third party api read and write
- ☑ Read event log
- ☑ Data log upload
- ☑ Flash LED
- ☑ Run/Stop CPU

☑ Configure as administrator

4. Download Web configuration to CPU.


Download

Download blocks to CPU

Select blocks to download.

Click Download to begin

Blocks

- ☑ Program Block
- ☑ Data Block
- ☑ System Block

Options

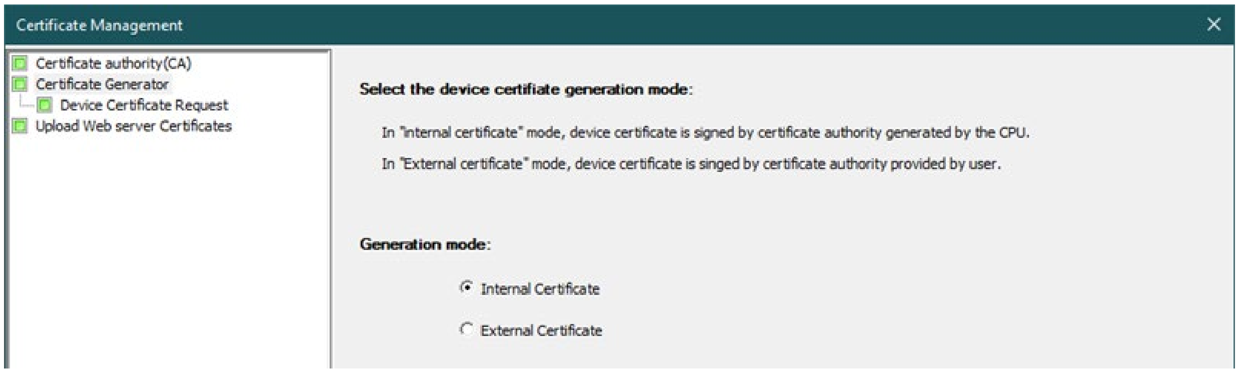
- ☑ Prompt on RUN to STOP
- ☑ Prompt on STOP to RUN
- ☐ Close dialog on success

5. Generate a new TLS certificate.

For using Web server of S7-200 SMART CPU, you need to configure the certificates in certificate management wizard in STEP 7-Micro/WIN SMART first.

STEP 7-Micro/WIN SMART provides two modes for downloading the certificate:

- In "External certificate" mode, device certificate is singed by certificate authority provided by user
- In "internal certificate" mode, device certificate is signed by certificate authority (CA) generated by the CPU


Certificate Management
Certificate authority(CA)
Certificate Generator
Device Certificate Request
Upload Web server Certificates

Select the device certifiate generation mode:

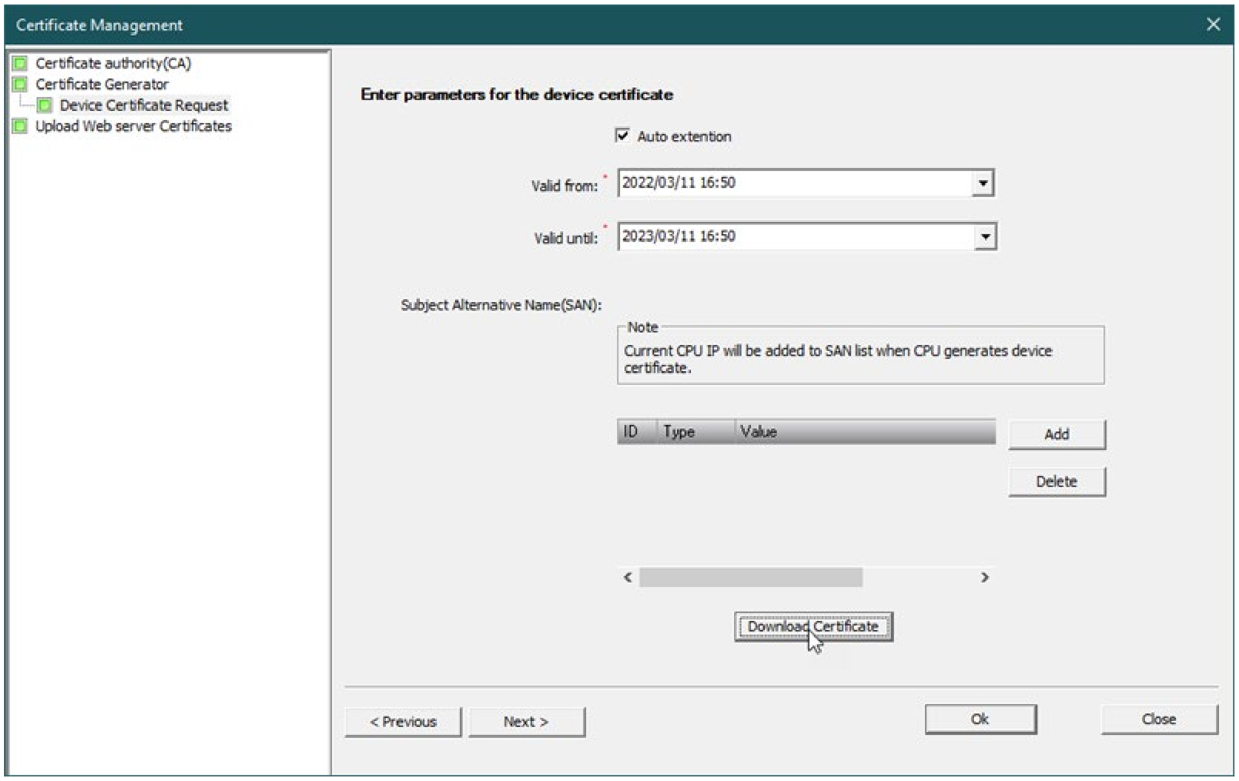
In "internal certificate" mode, device certificate is signed by certificate authority generated by the CPU.

In "External certificate" mode, device certificate is singed by certificate authority provided by user.

Generation mode:

[✓ Internal Certificate]

External Certificate



Enter parameters for the device certificate

☑ Auto extention

Valid from: * 2022/03/11 16:50

Valid until: * 2023/03/11 16:50

Subject Alternative Name(SAN):

Note
Current CPU IP will be added to SAN list when CPU generates device certificate.

ID	Type	Value

4.Web API basic functions

There are things you always need to know before using Web APIs.

4.1 API list

All APIs supported by Web API V1.0.0.

List	Description
login	Login a user, then you can visit CPU with access control.
logout	Logout a user, the user cookie does not work anymore from now on.
get_permission	After the successful login, returns a list of actions for whose execution the user is authorized.
read	Read from CPU with valid user token.
write	Write to CPU with valid user token.
browse	Browse API list supported by CPU.

4.2 API using suggestions

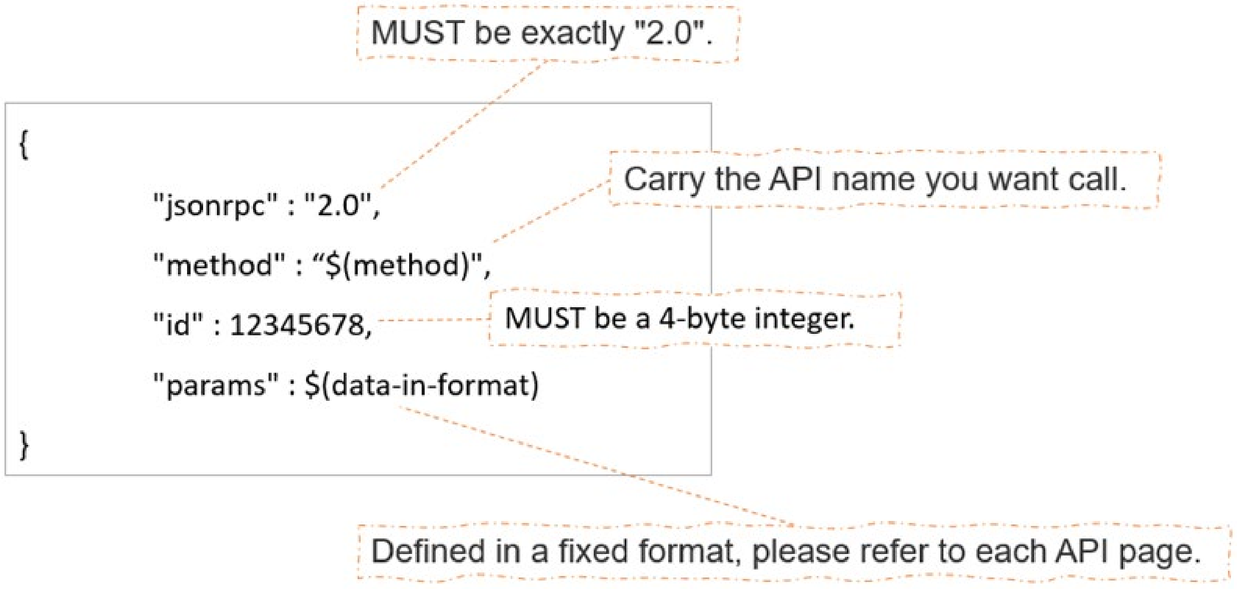
Items	Suggests
Call frequency.	It is recommended to call less than once a second. Web API call process takes up a lot of CPU resource. Network communication, TLS encryption, JSON-RPC parse, business check, request too fast may cause subsequent to be blocked.
Compress JSON text.	It is recommended to use compressed JSON text. Compressed JSON has a very small size and high transmission efficiency.
Efficient use Web request.	It is recommended to request more data in one request. Each Web request has very large buffer and can read/write with max to 32 nodes of data. Reduce interaction frequency and improve single request efficiency.
Use long connections.	It is recommended to use keep-alive connections. Building a new TLS chain will take up a lot of resources, it is very slow. But there is a high efficiency interaction in existing connections.

4.3 API general format

1. All API data need be encoded in the UTF-8 character encoding.
2. The overall payload shall not exceed 15KB.

The following is the JSON-RPC 2.0 API general template.

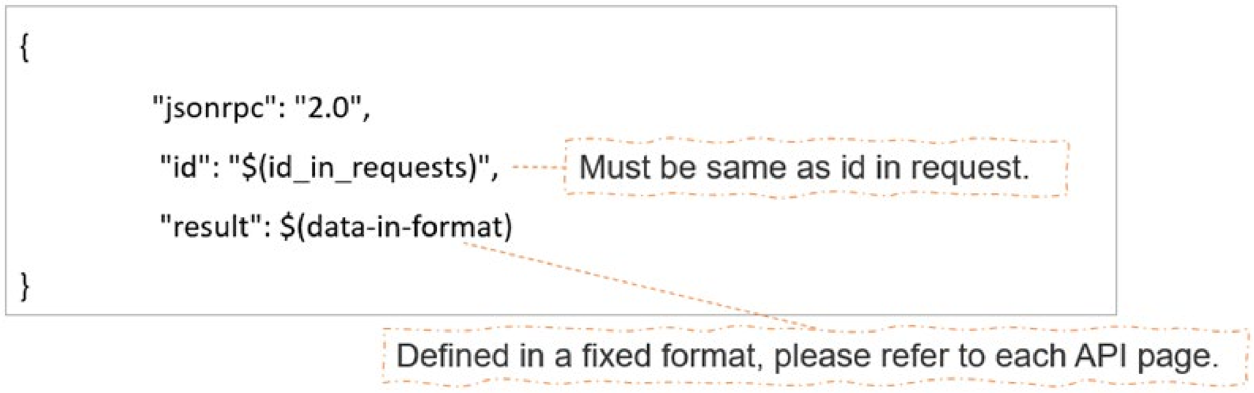
4.3.1 JSON-RPC request template



```
{
        "jsonrpc" : "2.0",
        "method" : “$(method)",
        "id" : 12345678,
        "params" : $(data-in-format)
}
```

MUST be exactly "2.0".
Carry the API name you want call.
MUST be a 4-byte integer.
Defined in a fixed format, please refer to each API page.

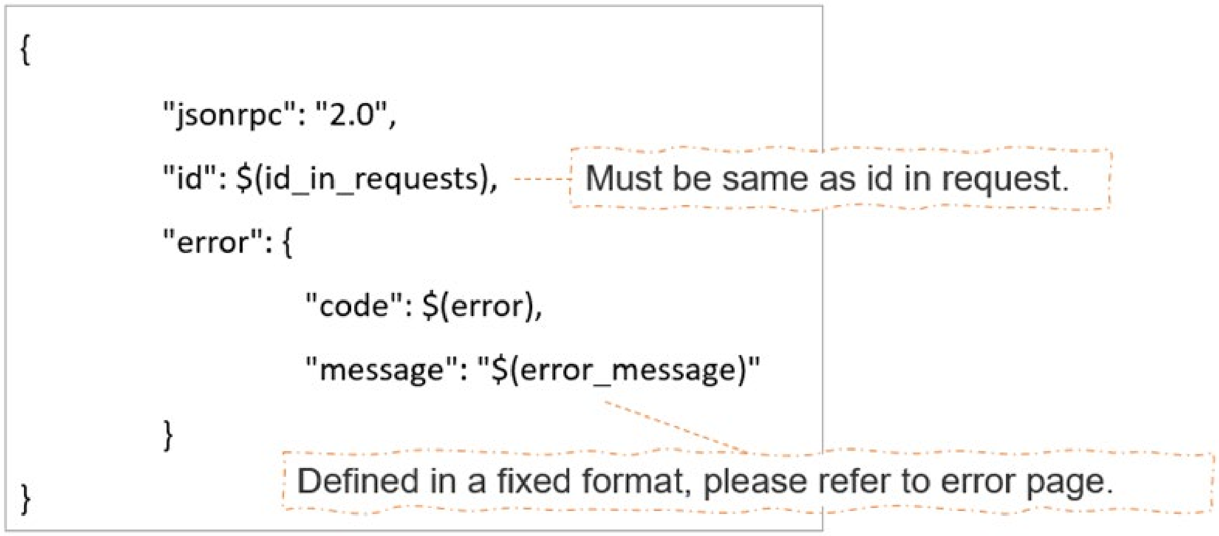
4.3.2 JSON-RPC success response template



```
{
        "jsonrpc": "2.0",
        "id": "$(id_in_requests)",
        "result": $(data-in-format)
}
```

Must be same as id in request.
Defined in a fixed format, please refer to each API page.

4.3.3 JSON-RPC fail response template



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

Must be same as id in request.
Defined in a fixed format, please refer to error page.

4.4 API limitations

1. URL of http request MUST be "https://ip_address/Web_api".
2. ALL http data MUST be encoded in the UTF-8 character encoding.
3. Content-Type of http header MUST be "application/json".
4. Http method MUST be POST.
5. Current version do not support JSON-RPC batch.

5.login

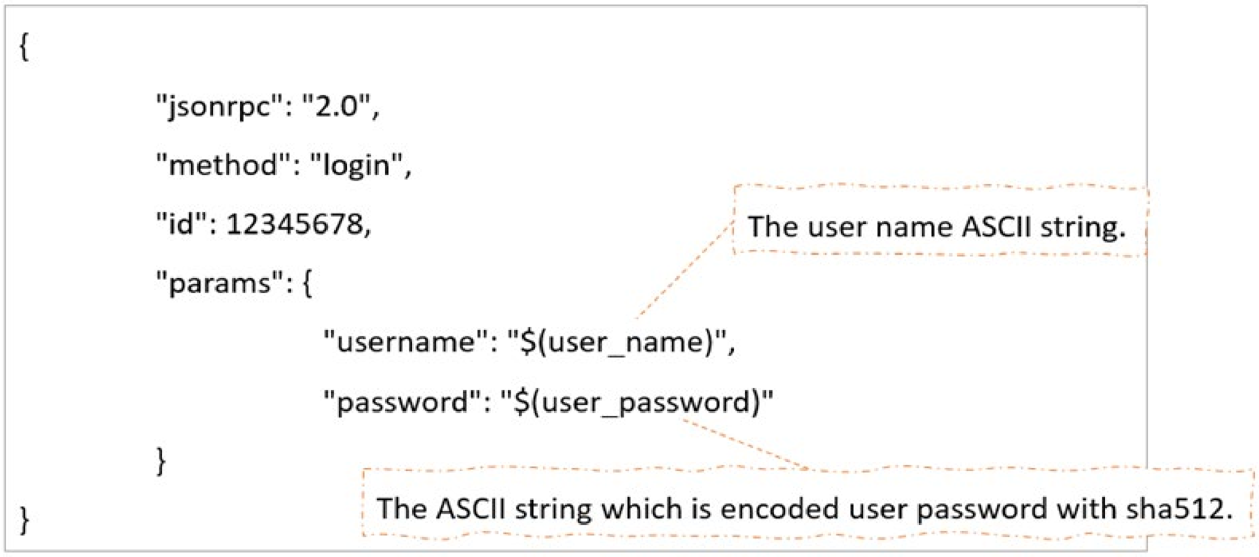
The login method checks the login data of the user and on successful verification return a user token for afterwards Web api actions.

5.1 Notes

a. The Web user shall be configured in Web server wizard.
b. The previous user token will be invalid after this successful login.
c. When there is no operation within half an hour, this user will be logout automatically.

5.2 Http request body format

Attributes	remarks
username	The username should already be configured in Web server wizard.
password	The password should be ASCII string which is encoded user password with sha512.



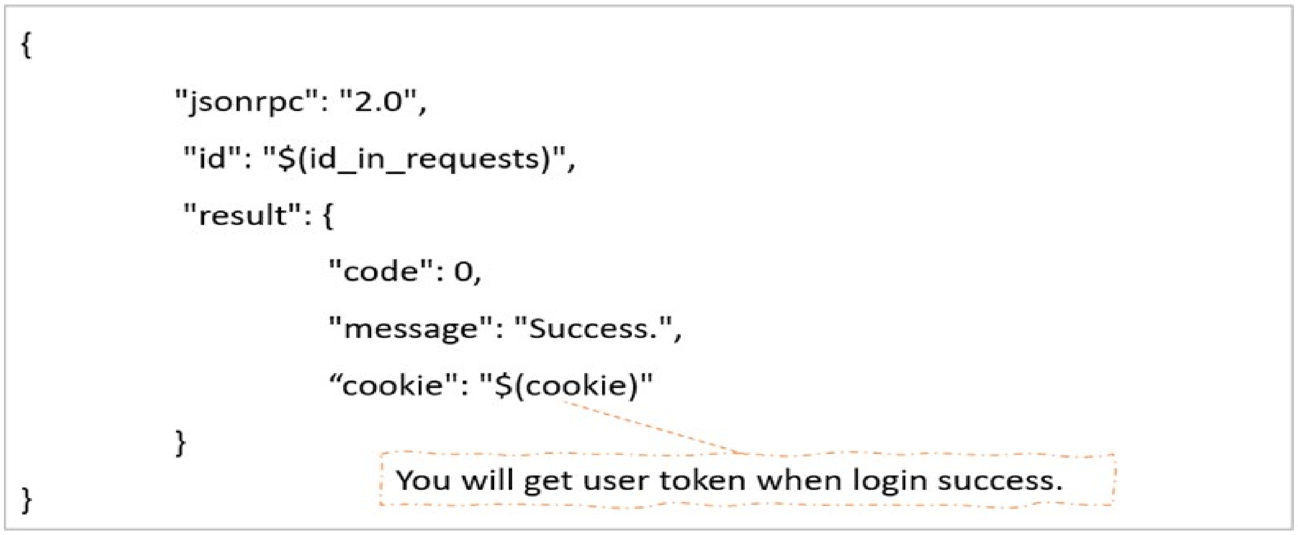
```
{
        "jsonrpc": "2.0",
        "method": "login",
        "id": 12345678,
        "params": {
                "username": "$(user_name)",
                "password": "$(user_password)"
        }
}
```

The user name ASCII string.
The ASCII string which is encoded user password with sha512.

5.3 Http response format

5.3.1 login success

Attributes	remarks
cookie	The user's token of login, use this token in subsequent API request to represent this user. The user's token will expire after half an hour without operation.



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

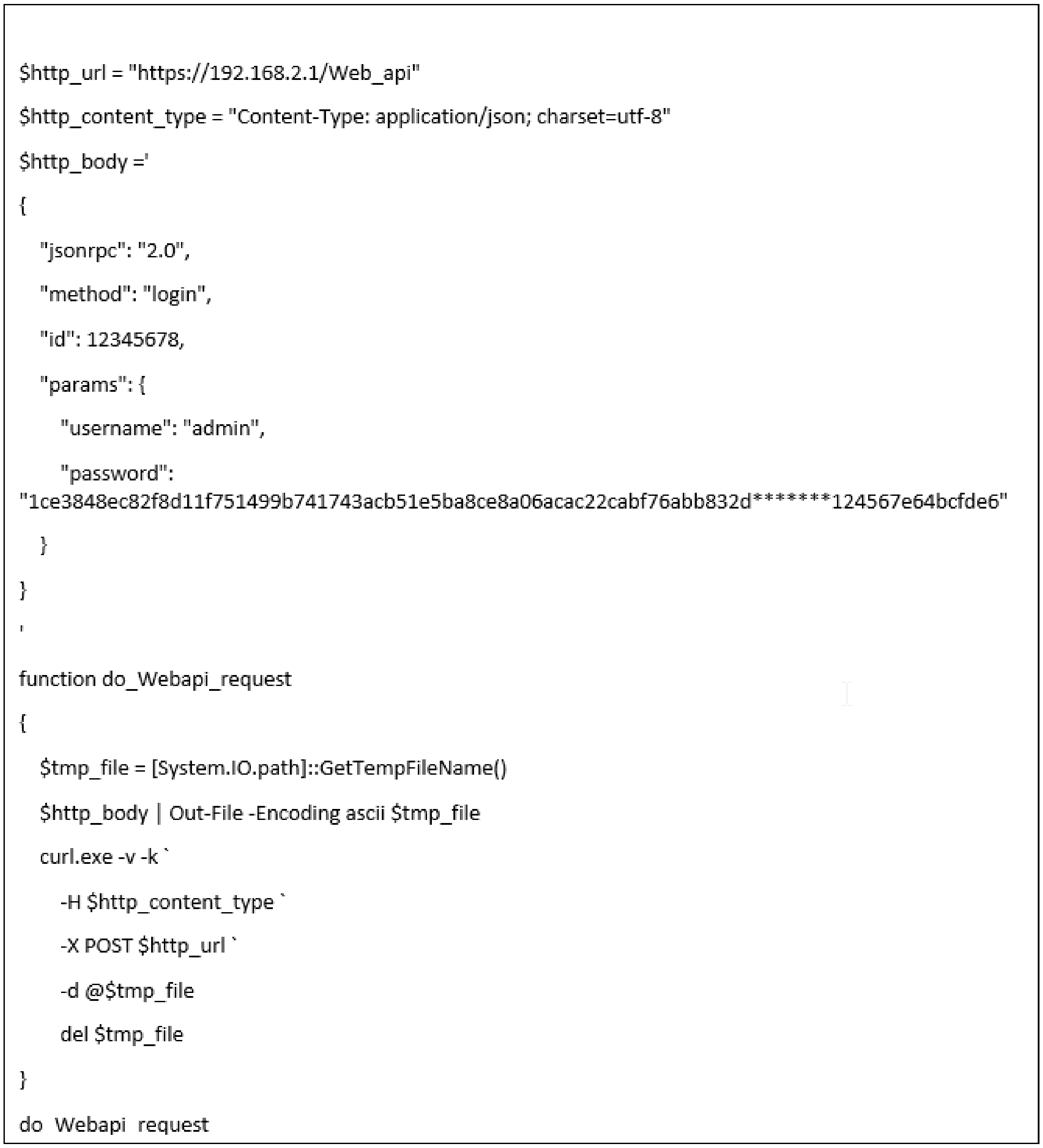
5.3.2 login fail

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

5.4 Example

Before testing:

- This is a windows powershell terminal example, you MUST send the example through powershell.
- This is a curl command; you MUST check your curl tools in your machine.
- This is a testing user and IP; you MUST modify to a correct account and IP.



```
$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

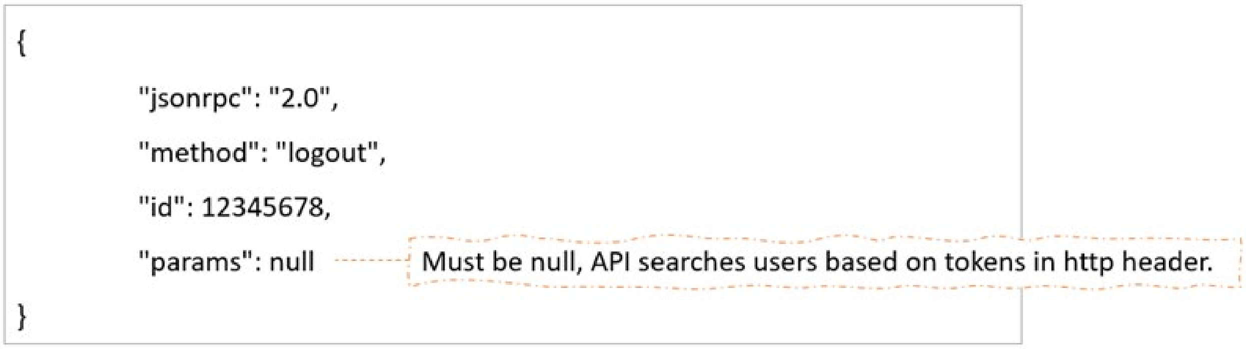
To logout an online user.

6.1 Notes

a. Logout a user who is not online or existing will still get a success.

b. You MUST set user's token into the "cookie" of HTTP header to logout the user.

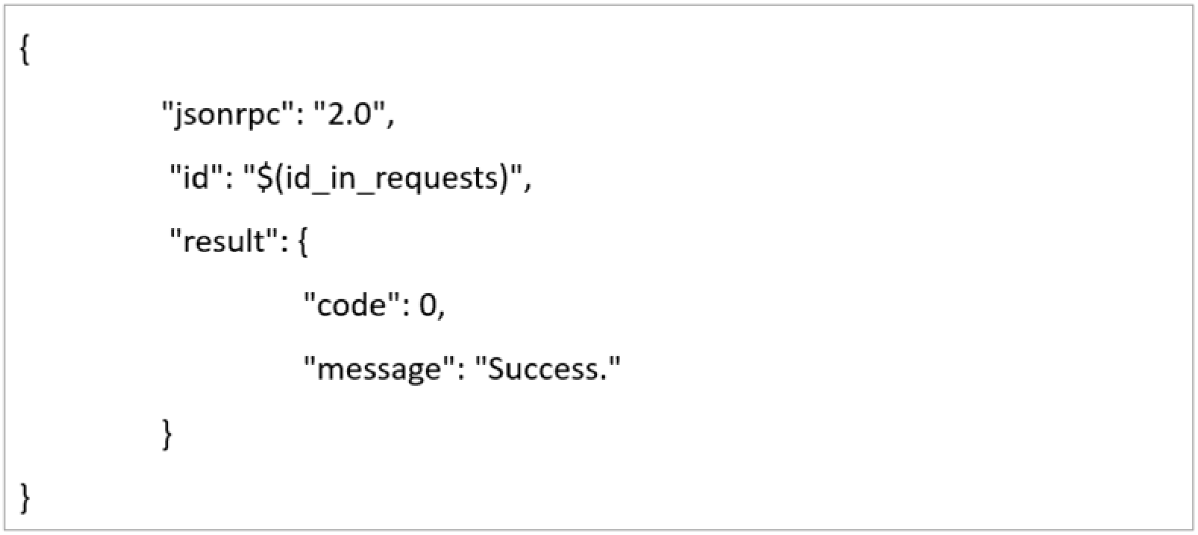
6.2 Http request format



```
{
        "jsonrpc": "2.0",
        "method": "logout",
        "id": 12345678,
        "params": null      Must be null, API searches users based on tokens in http header.
}
```

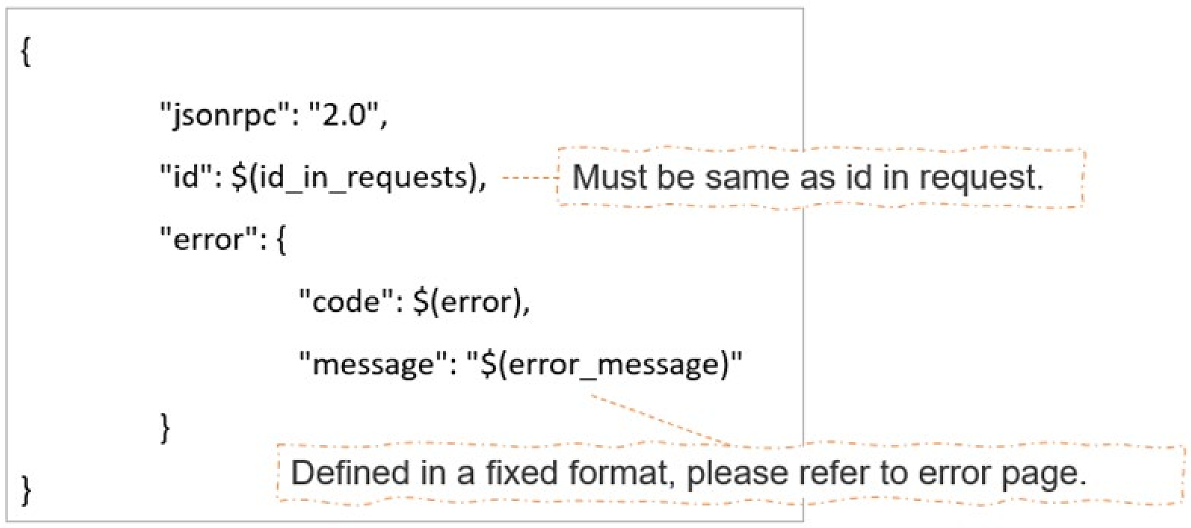
6.3 Http response format

6.3.1 logout success



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

6.3.2 logout fail



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

Must be same as id in request.

Defined in a fixed format, please refer to error page.

6.4 Example

Before testing:

- This is a windows powershell terminal example, you MUST send the example through powershell.
- This is a curl command; you MUST check your curl tools in your machine.
- 3. This is a testing token and IP; you MUST modify to a correct token and IP.

```
$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
```

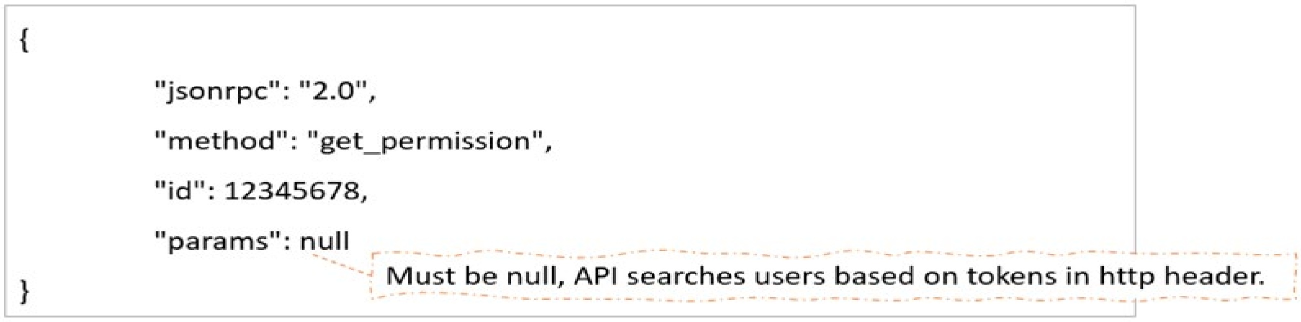
7.get_permission

To get a login user's permission list.

7.1 Notes

a. The user should login firstly and you had got the token.
b. You MUST set user's token into the "cookie" of HTTP header to logout the user.

7.2 Http request body format



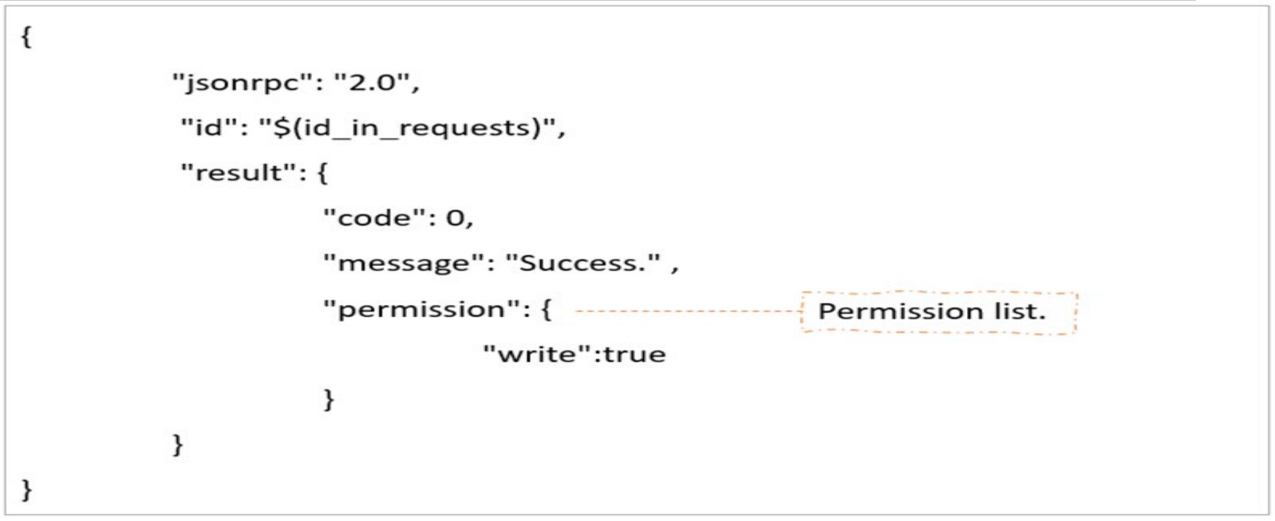
```
{
            "jsonrpc": "2.0",
            "method": "get_permission",
            "id": 12345678,
            "params": null
}
```

Must be null, API searches users based on tokens in http header.

7.3 Http response body format

7.3.1 get permission success

Attributes	remarks
permission	The permission list of login user.



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

Permission list.

7.4 Example

Before testing:

- This is a windows powershell terminal example, you MUST send the example through powershell.
- This is a curl command; you MUST check your curl tools in your machine.

- 3. This is a testing token and IP; you MUST modify to a correct token and IP.

```
$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
}
```

8.read

To read data from a specified address.

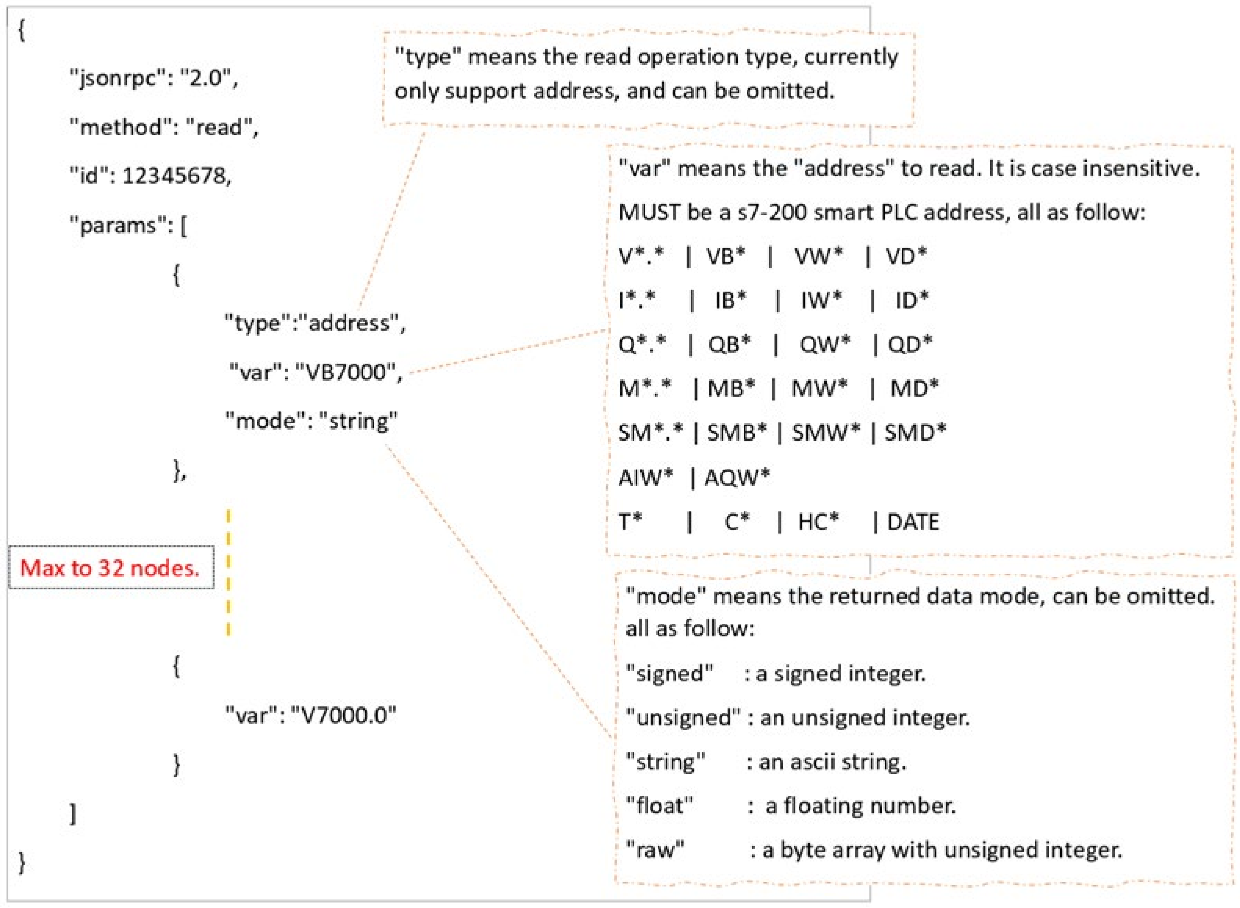
8.1 Notes

a. You MUST login before reading.

b. You MUST set user's token into the "cookie" of HTTP header to logout the user.
c. You can read with max to 32 addresses of data in one request.
d. You can send an empty request in order to get system status.

8.2 Http request body format

Attributes	remarks
params	A JSON array object inside with reading nodes, max to 32 nodes.
type	"type" means the read operation type, currently only support address, and can be omitted.
var	"var" means the "address" to read. It is case insensitive. MUST be a S7-200 SMART CPU address, all as follow: 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" means the returned data mode, and can be omitted. all as follow: "signed": a signed integer. This is default type if omit to specify data mode. "unsigned": an unsigned integer. "string": an ascii string. "float": a floating number. "raw": a byte array with unsigned integer.



```
{
    "jsonrpc": "2.0",
    "method": "read",
    "id": 12345678,
    "params": [
        {
            "type":"address",
            "var": "VB7000",
            "mode": "string"
        },

        {
            "var": "V7000.0"
        }
    ]
}
```

Max to 32 nodes.

"type" means the read operation type, currently only support address, and can be omitted.

"var" means the "address" to read. It is case insensitive.
MUST be a s7-200 smart PLC address, all as follow:
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" means the returned data mode, can be omitted. all as follow:

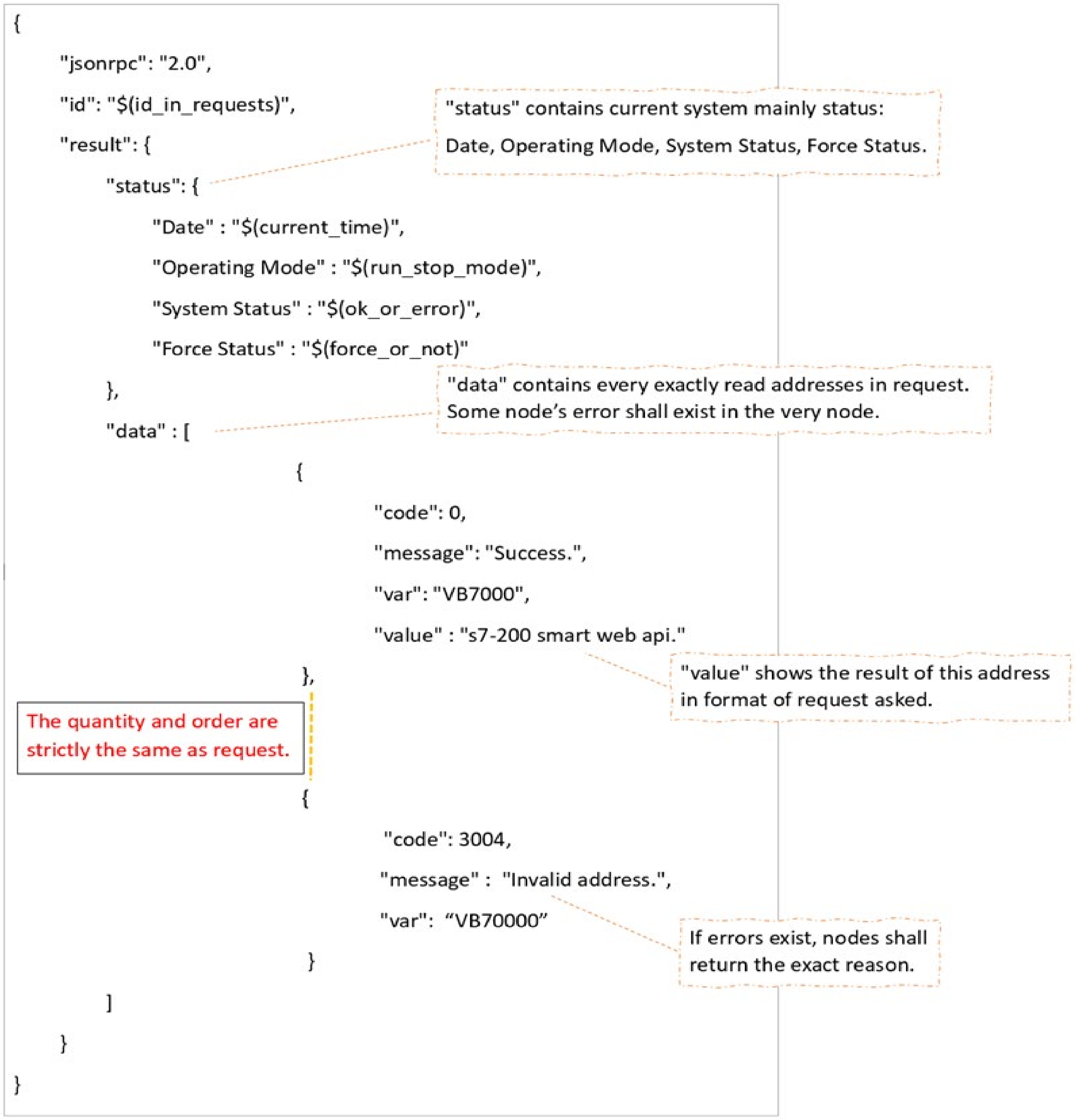
"signed"	: a signed integer.
"unsigned"	: an unsigned integer.
"string"	: an ascii string.
"float"	: a floating number.
"raw"	: a byte array with unsigned integer.

8.3 Http response body format

8.3.1 read success

Attributes	remarks
status	"status" contains current system mainly status: Date: time string in format of "YYYY-MM-DD HH:MM:SS". Operating Mode: RUN, STOP. System Status: OK, Error. Force Status: Forced, Not Forced.
data	A JSON array object inside with read result. The data array, which quantity and order are strictly the same as request.
- code & message	"code" & "message" shows the result of reading this address.

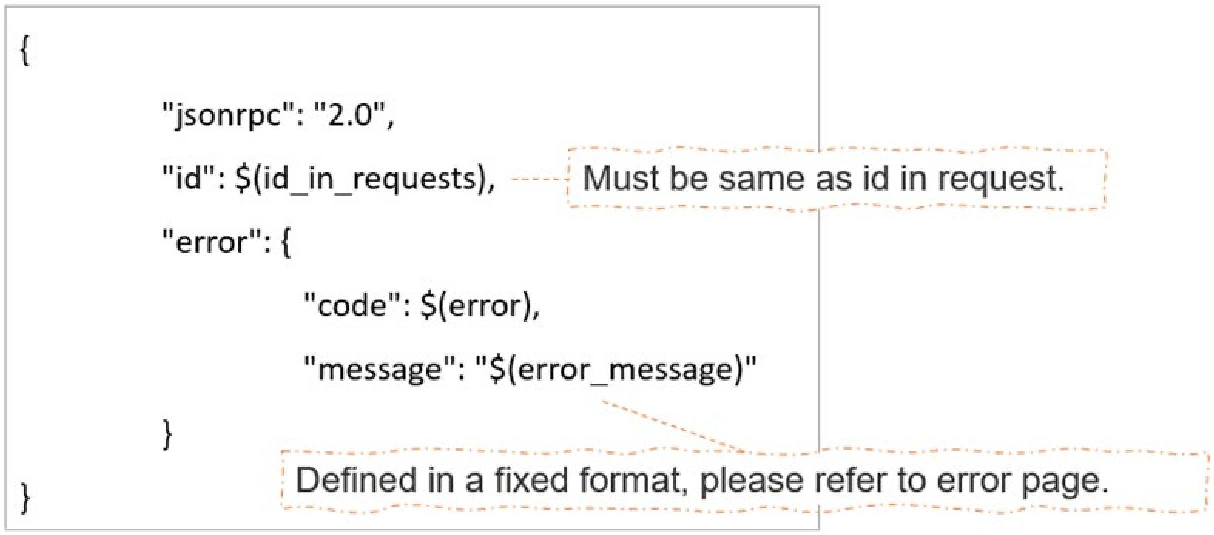
- var	"var" is exactly the same as request.
- value	"value" shows the result of reading this address in format of request asked.



```
{
        "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" contains current system mainly status: Date, Operating Mode, System Status, Force Status.
"data" contains every exactly read addresses in request. Some node's error shall exist in the very node.
"value" shows the result of this address in format of request asked.
The quantity and order are strictly the same as request.
If errors exist, nodes shall return the exact reason.

8.3.2 Read fail



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

Must be same as id in request.
Defined in a fixed format, please refer to error page.

8.4 Example

Before testing:

- This is a windows powershell terminal example, MUST be send the example through powershell.
- This is a curl command; you MUST check your curl tools in your machine.
- 3. This is a testing token and IP; you MUST modify to a correct token and IP.

```
$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

To write data to a specified address.

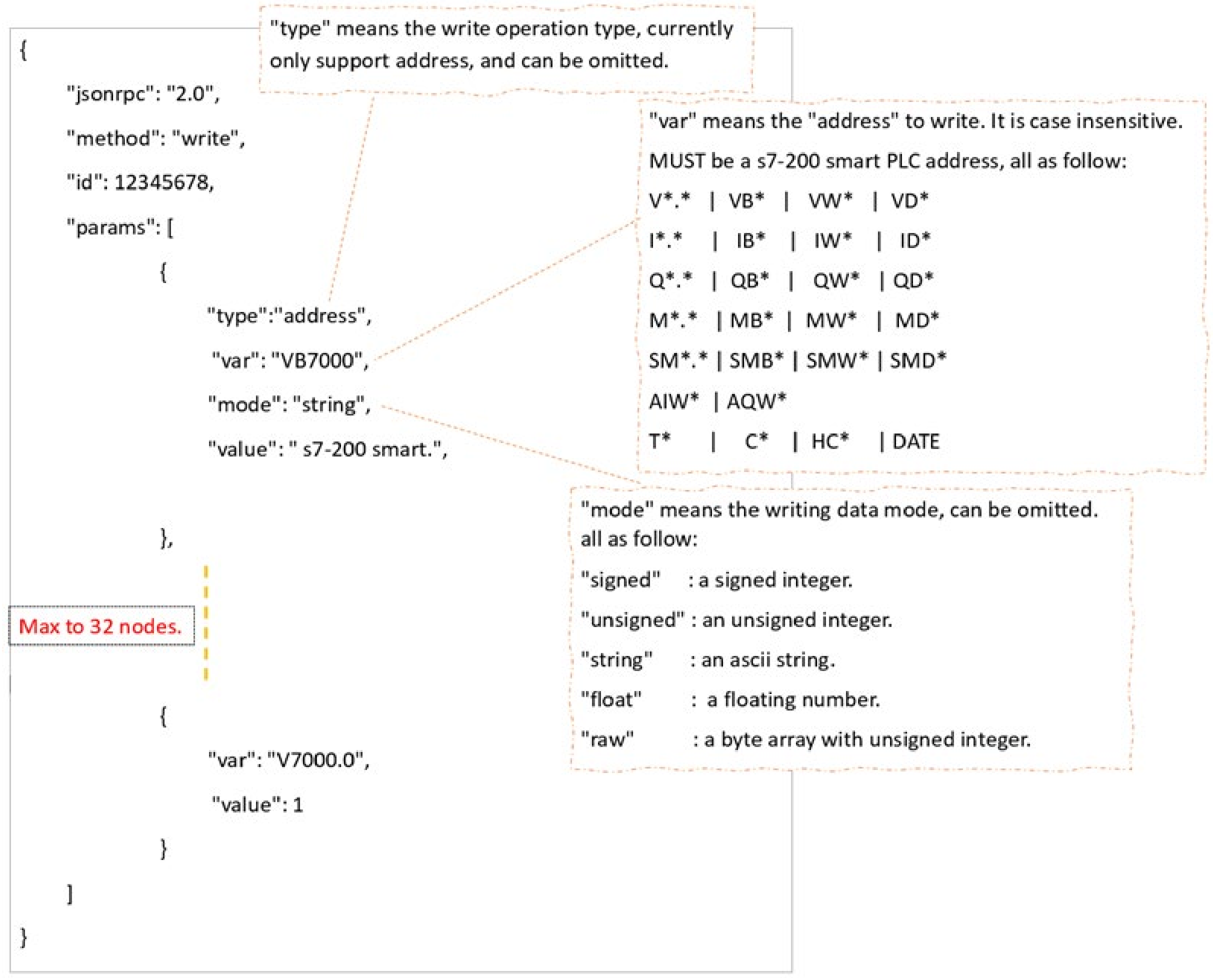
9.1 Notes

a. You MUST login before writing.
b. You MUST set user's token into the "cookie" of HTTP header to logout the user.
c. You can write with max to 32 addresses of data in one request.
d. You can ONLY write data while CPU in run.
e. You can write an empty request to get system status.

9.2 Http request body format

Attributes	remarks
params	A JSON array object inside with writing nodes, max to 32 nodes.
type	"type" means the write operation type, currently only support address, and can be omitted.
var	"var" means the "address" to write. It is case insensitive. MUST be a S7-200 SMART CPU address, all as follow: 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" means the writing data mode, and can be omitted. all as follow: "signed": a signed integer. This is default type if omit to specify data mode. "unsigned": an unsigned integer. "string": an ascii string. "float": a floating number. "raw": a byte array with unsigned integer.

value	"value" shall take the data in format of "mode".



```
{
    "jsonrpc": "2.0",
    "method": "write",
    "id": 12345678,
    "params": [
        {
            "type":"address",
            "var": "VB7000",
            "mode": "string",
            "value": " s7-200 smart.",

        },

        {
            "var": "V7000.0",
            "value": 1
        }
    ]
}
```

"type" means the write operation type, currently only support address, and can be omitted.

"var" means the "address" to write. It is case insensitive. MUST be a s7-200 smart PLC address, all as follow:

```
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" means the writing data mode, can be omitted. all as follow:

"signed"	: a signed integer.
"unsigned"	: an unsigned integer.
"string"	: an ascii string.
"float"	: a floating number.
"raw"	: a byte array with unsigned integer.

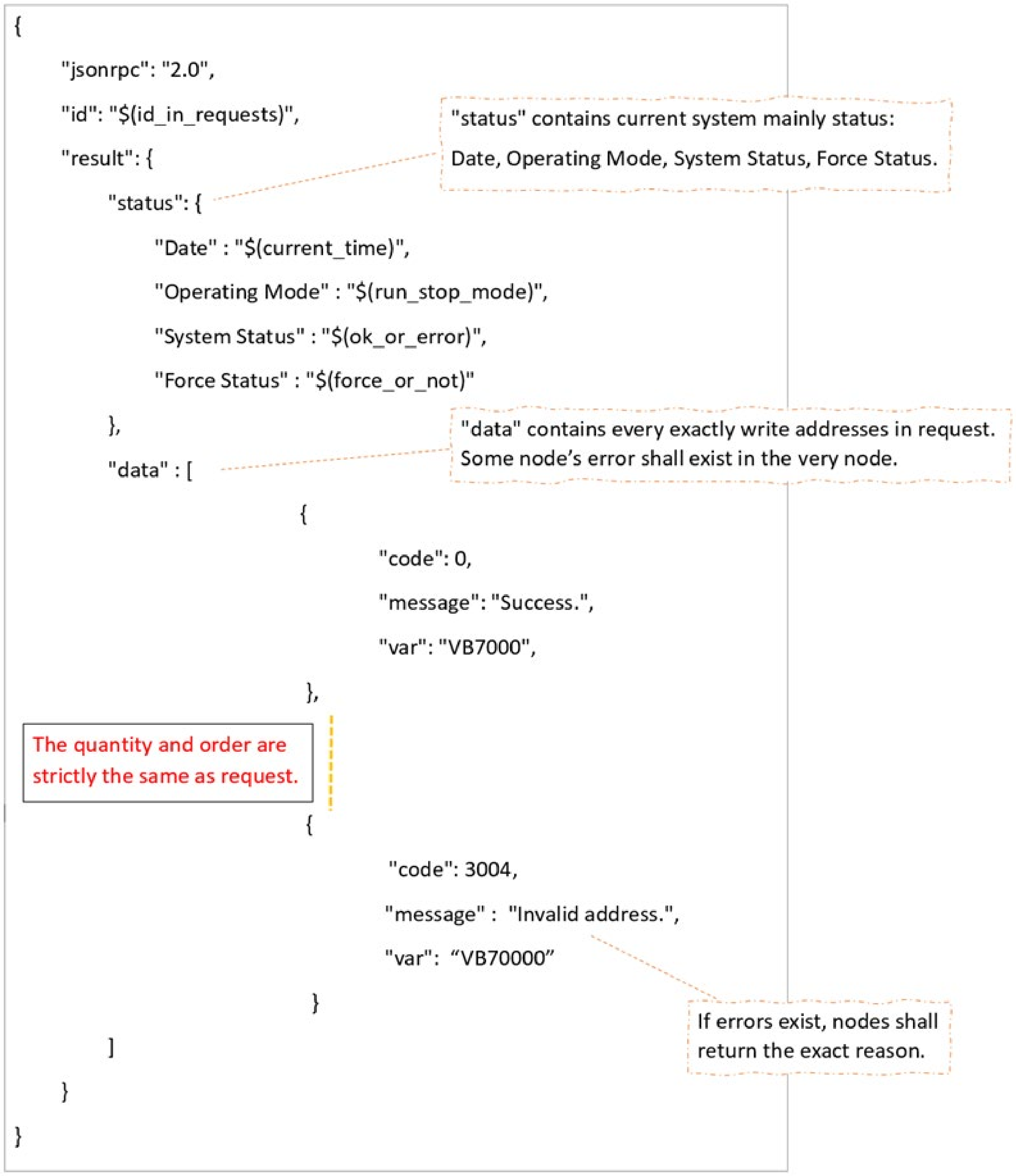
Max to 32 nodes.

9.3 Http response body format

9.3.1 write success

Attributes	remarks
status	"status" contains current system mainly status: Date: time string in format of "YYYY-MM-DD HH:MM:SS". Operating Mode: RUN, STOP. System Status: OK, Error. Force Status: Forced, Not Forced.
data	A JSON array object inside with write result. The data array, which quantity and order are exactly the same as request.

- code & message	"code" & "message" shows the result of writing this address.
- var	"var" is exactly the same as request.



```
{
    "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"
                    }
        ]
    }
}
```

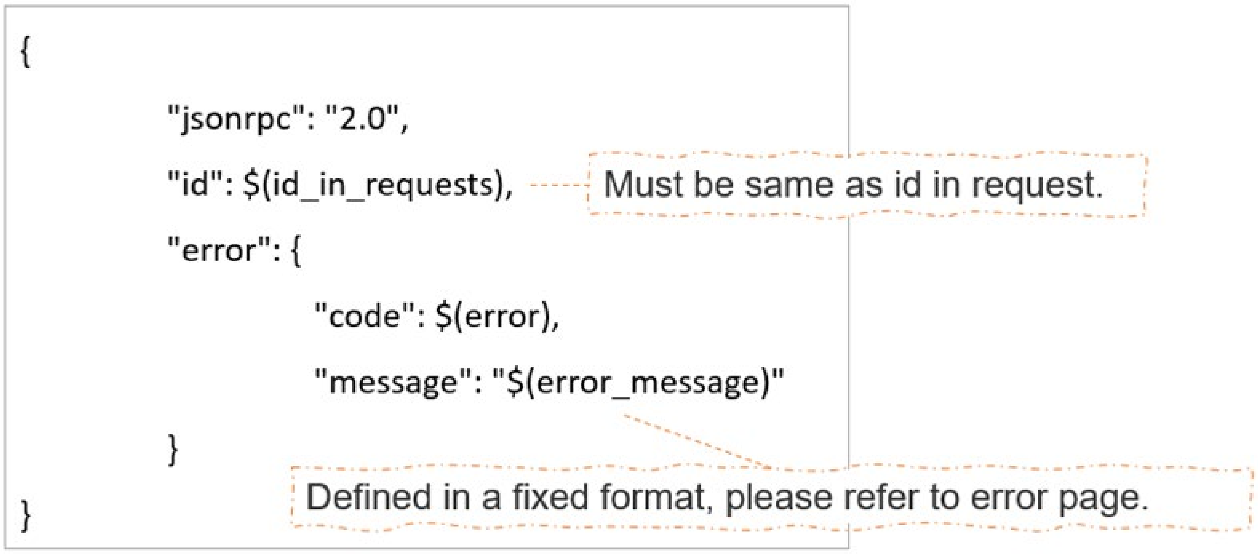
"status" contains current system mainly status: Date, Operating Mode, System Status, Force Status.

"data" contains every exactly write addresses in request. Some node's error shall exist in the very node.

The quantity and order are strictly the same as request.

If errors exist, nodes shall return the exact reason.

9.3.2 write fail



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

Must be same as id in request.
Defined in a fixed format, please refer to error page.

9.4 Example

Before testing:

- This is a windows powershell terminal example, you MUST send the example through powershell.
- This is a curl command; you MUST check your curl tools in your machine.
- This is a testing token and IP; you MUST modify to a correct token and IP.

```
$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

Get supported API list.

10.1 Notes

a. no need to login.

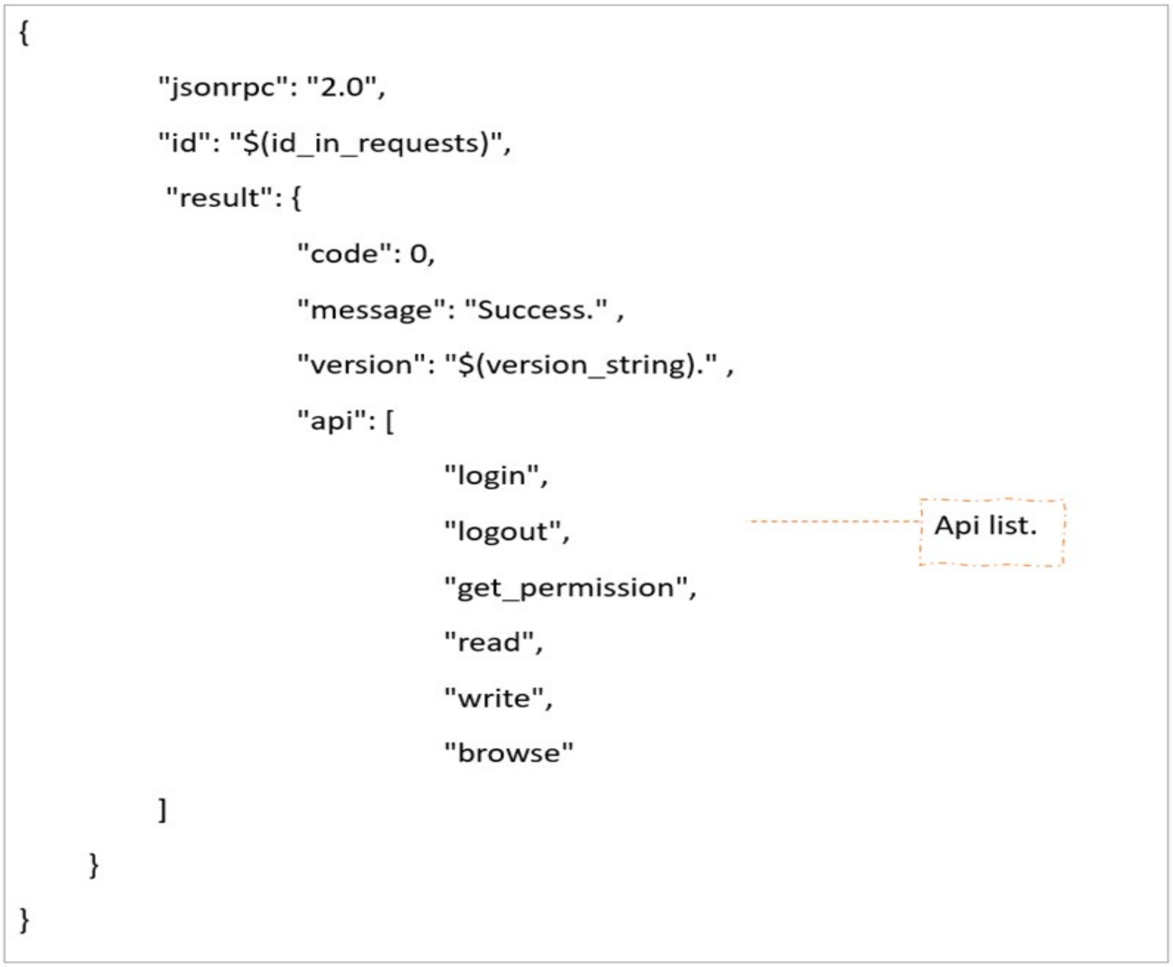
10.2 Http request body format

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

10.3 Http response body format

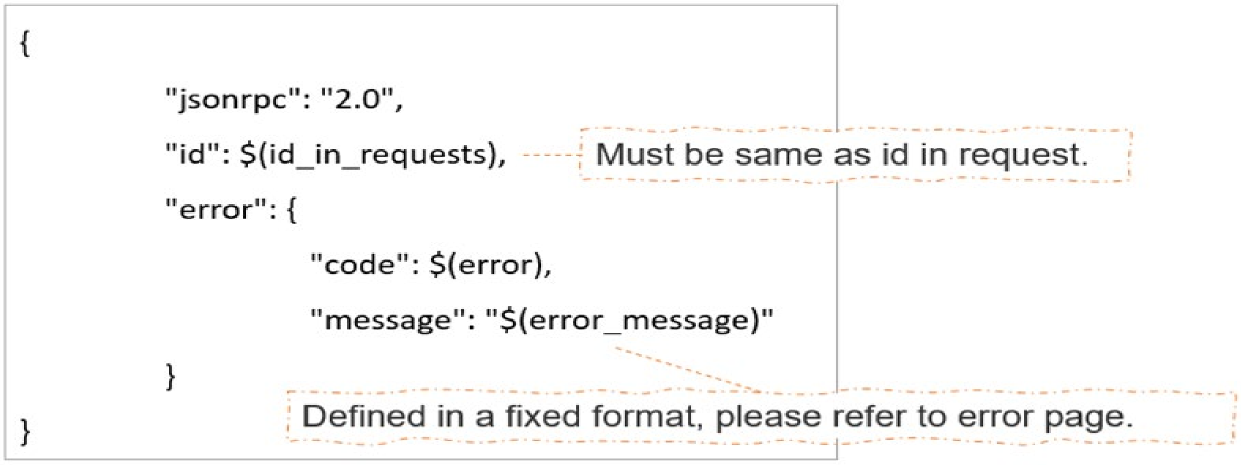
10.3.1 browse success

Attributes	remarks
version	"version" shows the WEB API version string.
api	A JSON array object inside with APIs list.



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

10.3.2 browse fail



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

Must be same as id in request.
Defined in a fixed format, please refer to error page.

10.4 Example

Before testing:

- This is a windows powershell terminal example, you MUST send the example through powershell.
- This is a curl command, you MUST check your curl tools in your machine.
- This is a testing user and IP, you MUST modify to a correct account and IP.

```
$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
```