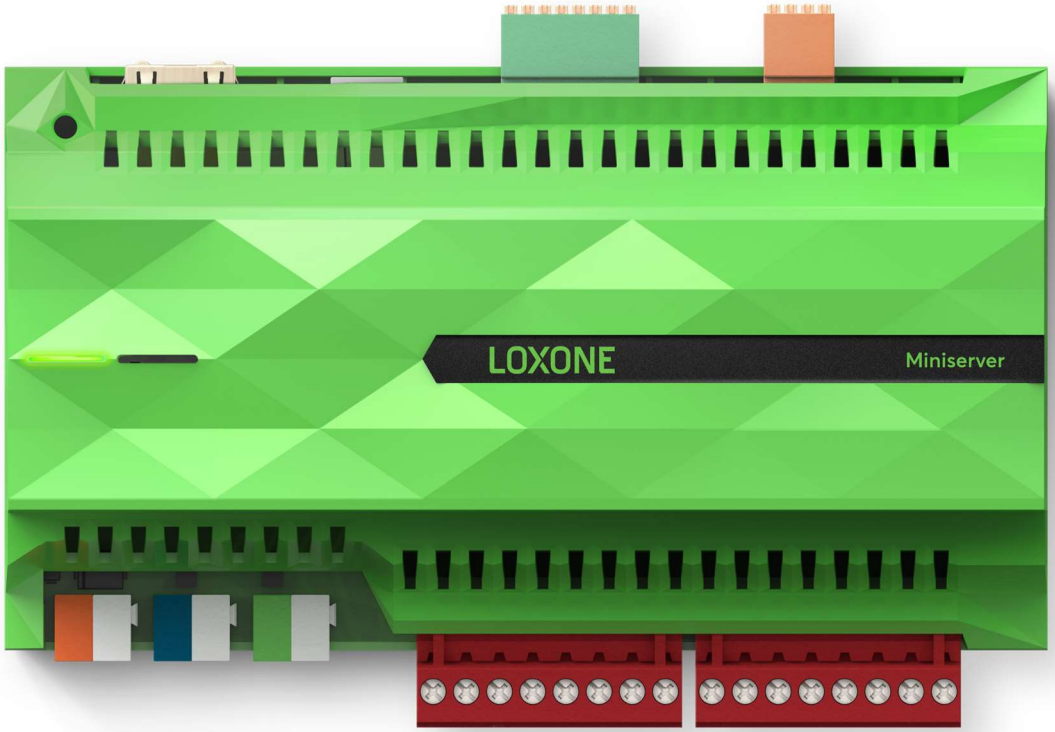
LOXONE

Usermanagement

31.3.2026



V17.0

This Document describes all webservices for creating new plus altering and deleting existing users.

Table of contents

Commands

- In front of each command there is written "/jdev/sps/".
- It describes the path to the available commands.
- Commands need to be performed by a user with a right to modify other users.

Get all available users

getuserlist2

- gets a list of all configured users

Response

```
{
        "name":"Administrator",
        "uuid":"089396d4-0207-0119-1900000000000000",
        "isAdmin":true,
        "userState": 0
```

```
},
{
        "name":"Feuerwehr",
        "uuid":"0a5fa72f-018b-0050-1900000000000000",
        "isAdmin":false,
        "userState": 0
},
{
        "name":"admin",
        "uuid":"0ee1424f-006b-57d5-ffffeee000880187",
        "isAdmin":true,
        "userState": 0,
        "expirationAction":{ACTION}
}
```


- expirationAction
  - ActionDeactivate = 0
  - ActionDelete = 1

Get all details for a user

getuser/{uuidUser}

This webservice returns a json with the full user-configuration.

Response


```
{
        "name":"admin",
        "desc:"",
        "uuid":"12eebb90-00a1-3073-ffff88c561c84c44",
        "userid":"1234 24 12 83",
        "firstname":"",
        "lastname":"",
        "email":"",
        "phone":"",
        "uniqueUserId":"",
        "company":"",
        "department":"",
        "personalno":"",
        "title":"",
        "debitor":"",
        "customField1":"",
        "customField2":"",
        "customField3":"",
        "customField4":"",
        "customField5":"",
        "lastedit":472141393,
        "userState":0,
        "isAdmin":true,
        "changePassword":true,
```

```
        "masterAdmin":true,
        "userRights":2047,
        "scorePWD":2,
        "scoreVisuPWD":-2,
        "trustMember": serialNumber,
        "validUntil":{SECONDS_SINCE_2009},
        "validFrom":{SECONDS_SINCE_2009},
    "expirationAction":{ACTION},
        "usergroups":[
                {
                        "name":"Administratoren",
                        "uuid":"12eebb90-00a1-3076-ffff88c561c84c44"
                },
                {
                        "name":"Alle",
                        "uuid":"12eebb90-00a1-307c-ffff88c561c84c44"
                }
        ],
        "nfcTags":[
                {
                        "name":"NFC-Tag",
                        "id":"12 34 56 78 90 98 76 54"
                }
        ],
        "keycodes":[
                {
                        "code":"6C3A7D85A4B196E37C5DADFDE7E0586ED4D7137C"
                }
        ],
        "customFields":[
           "Custom Field 1",
           "Custom Field 2",
           "Custom Field 3",
           "Custom Field 4",
           "Custom Field 5"
         ]
}
```


Get all configurable user-groups

getgrouplist

Lists all available user-groups and additional information

Response


```
[
        {
                "name":"Administratoren",
                "description":"Administratoren",
                "uuid":"12eebb90-00a1-3076-ffff88c561c84c44",
                "type":1,
```

```
                "userRights":4294967295
        },
        {
                "name":"Benutzer",
                "description":"Benutzer",
                "uuid":"12eebb90-00a1-307a-ffff88c561c84c44",
                "type":0,
                "userRights":1
        },
        {
                "name":"Alle",
                "description":"Alle",
                "uuid":"12eebb90-00a1-307c-ffff88c561c84c44",
                "type":2,
                "userRights":0
        },
        {
                "name":"Niemand",
                "description":"Niemand",
                "uuid":"12eebb90-00a2-3080-ffff88c561c84c44",
                "type":3,
                "userRights":0
        }
]
```


Create or Edit existing user

addoredituser/{json}

- {json}:
  - User configuration with all settings to be created / changed
  - see getuser-cmd for details on JSON structure
- json-content:
  - uuid: [optional]
    - if null, a new user will be created.
    - if provided and uuid is found, an existing user is adapted.
    - if not found, it will return with code 500, user not found.
  - usergroups
    - When during editing a user and no groups-array is set, the group-assignment will remain unchanged.
  - All other json attributes are optional.
- Return-Value:
  - Fully serialized user
  - Contains UUID of created or adapted user
  - errortext if failed

- Certain fields like username are validated, invalid characters are replaced by underscores. For further processing of a newly created user, it is recommended adding the uuid to the succeeding webservices.

Example - Add a new user

Request

jdev/sps/addoredituser/{"name":"A","userid":"1234","changePassword":true,"userState":4,"validUntil":371738510,"validFrom":371736410,"expirationAction":{ACTION}}

Please note, that here no UUID is specified - that is why a new user is created.

Response


```
{
        "name": "A",
        "uuid": "16282419-0233-2026-ffffeee000240011",
        "userid": "1234",
        "isAdmin": false,
        "changePassword": true,
        "masterAdmin": false,
        "userRights": 32,
        "scorePWD": -1,
        "scoreVisuPWD": -1,
        "userState": 4,
        "validUntil": 371738510,
        "validFrom": 371736410,
        "usergroups": [],
        "nfcTags": [],
        "keycodes": []
}
```


Example - Existing User → groups and NFC-tags are adapted


```
{
        "name":"ExistingUser",
        "uuid":"133cb125-0052-f7e9-ffff88c561c84c44",
        "usergroups":[
                "12eebb90-00a1-3076-ffff88c561c84c44",
                "12eebb90-00a1-307a-ffff88c561c84c44"
        ],
        "nfcTags":[
          { “name”: “Tag1”, “id”: “12 34 56 78 90 98 76 54” }
        ]
}
```


Example – Existing User → custom user fields are adapted


```
{
        "name":"ExistingUser",
        "uuid":"133cb125-0052-f7e9-ffff88c561c84c44",
        "customField1":"Text 1",
        "customField2":"Text 2",
        "customField3":"Text 3",
        "customField4":"Text 4",
        "customField5":"Text 5",
```

```
}
```

Create User

createuser/{username}

- creates an user with a given username
- result will contain the uuid of the new user

Delete User

deleteuser/{uuidUser}

- deletes an user with a given uuid
- All active websockets used by this user will be closed.
- When trying to delete the last admin, the Miniserver will respond with 403

Assign user to group

assignusertogroup/{uuidUser}/{uuidGroup}

- adds an existing user to a usergroup
- more information: Get all configurable user-groups

Remove user from group

removeuserfromgroup/{uuidUser}/{uuidGroup}

- removes an user from a usergroup

Update user password

updateuserpwdh/{uuidUser}/{value}

- adapts a password for an existing user
- {value}: hashed password-value
- optional: append password-scoring for strength of new password
  - {value} = {hash}|{score}
- return Value 504 if user is from a trust member and member cant be reached

Update user Visu-password

updateuservisupwdh/{uuidUser}/{value}

- adapts a visu-password for an existing user
- {value}: hashed visu-password
- optional: append password-scoring for strength of new password
  - {value} = {hash}|{score}

- return Value 504 if user is from a trust member and member cant be reached

Update user keycodes

updateuseraccesscode/{uuidUser}/{newAccessCode}
Sets a new keycode, updates, or removes an existing keycode for a user.

- to remove a keycode, simply pass in an empty string or invalid code
- {newAccessCode} → numeric code (0-9) with 2-8 digits, will be hashed once stored on the Miniserver, do not hash on client. This is required to ensure no duplicate codes are in use.

Return values:

- 200 → code unique, successfully changed or deleted
- 201 → code not unique and successfully changed
- 400 → error, {uuidUser} not found or not a user
- 403 → error, logon user has no User management right
- 409 → error, code already in use of NFC Authentication block
- 429 → error, 5 minutes lock after 5 requests from logon user without sufficient rights

Assign NFC tag to user

For assigning an NFC tag to a user, the tag must be read first using an NFC Code touch (see "Structure File" documentation, section "NFC Code Touch"). Once the NFC Tag ID is known it can be paired with the user. Please not that the token which authenticates this request needs addtionally the 0x20 permission.

addusernfc/{uuidUser}/{nfcTagId}/{name}

Remove NFC tag from user

The linking of an NFC tag and the user can be removed at any time.

removeusernfc/{uuidUser}/{nfcTagId}

Get all permissions for a control

This webservice returns a List of all Users and Groups which are directly assigned to a control with a given uuid

getcontrolpermissions/{uuid}

Get Names of Custom User Fields

Since Version 14.3

Returns a list of customizable user-fields 1-5

Request

jdev/sps/getcustomuserfields

Response


```
{
        "customField1": "Building",
        "customField2": "Parking Space",
        "customField3": "ICQ Number",
        "customField4": "Custom Field 4",
        "customField5": "Custom Field 5"
}
```


Get List of existing options for user-fields

Since Version 14.3

Returns an object that contains already configured options for various user-specific fields. The key is the same as in the **getuser** response, the value is an array of values that have already been assigned to this properties.

Request

jdev/sps/getuserpropertyoptions

Response


```
{
        "company": ["Loxone Smart Engineering","Loxone Electronics"],
        "department": ["Software Development","Marketing","Product Management"]
}
```


Get List of existing options for fields

Since Version 14.3

Returns the user with a given userID.

If user is not found, values are empty

Request

jdev/sps/checkuserid/[userid]

Response


```
{
        "name": "admin",
        "uuid": "19f4f3b3-038a-4172-ffffb91f6db8271b"
}
```

Required Rights

There are 4 levels of rights which allow different levels of editing. Make sure that the commands are executed with a token with the required rights. To request a token with such rights the user itself must also have those rights.
These levels are best described via User-Roles:

- Guest
  - is just a normal user
  - is not allowed to edit anything
- User
  - has enabled “Change own password” -checkbox
  - may edit own password or visu-password
- Usermanager
  - is Member of a group with group-right “Usermanagement”
  - “Change own password” is automatically enabled and may not be changed
  - may fully edit own user
  - may fully edit non-Admin-users
  - is not allowed to view or edit admin-users
  - is not allowed to add **any** user to an admin-group
- Administrator
  - is Member of group “All Access” or a group with group-right “Loxone Config”
  - may fully edit own user
  - may fully edit all other users
  - is allowed to add or remove any user to/from an admin-group

	1 Administrator	2 Usermanager	3 User	4 Guest
change own password	X	X	X	
change own access-code	X	X	X	
edit own nfc tag	X	X		
change Code / Tag / Pass of an administrator	X			
change Code / Tag / Pass of non-admin	X	X		
add or remove user from/to admin-groups	X			
add or remove user from/to common groups	X	X		

Detailed info on attributes

userstate
Indicates whether or not a user is active and may log in or get access (depending on the rights granted in config permission management).

- 0 = enabled, without time limitations
- 1 = disabled
- 2 = enabled until, disabled after that point in time
- 3 = enabled from, disabled before that point in time
- 4 = timespan, only enabled in between those points in time

type
- 0 = Normal
- 1 = Admin (deprecated)
- 2 = All
- 3 = None
- 4 = AllAccess → New "Admin Group"

userrights
- See "permissions" in the document Communicating with Miniserver

validUntil
- Only available/required if with userstate 2 and 4
- provided as seconds since 1.1.2009 00:00:00

validFrom
- Only available/required if with userstate 3 and 4
- provided as seconds since 1.1.2009 00:00:00

expirationAction
- Only available/required if with userstate 2 and 4
- Since V14.2.5.16
- Possible values
  - 0 = Deactivate
  - 1 = Delete

changePassword
Specifies whether or not a user is allowed to change its passwords from within the apps

name
When it comes to users, this is the username that is used to login via our app.

userid
May be empty, this is the id that will be returned by the NFC permission block when granting access. In Loxone Config, this field is configured as NFC Code Touch ID

uuid
Generated by the Miniserver, unique identifier.

isAdmin
- This flag is set if the user has administrative rights on the Miniserver.
- There must be at least one user who has isAdmin set, in order to still have config access.
- Any modification that would violate this rule will result in error 403 “delete of last admin not allowed”

masterAdmin (deprecated)
In config versions prior to 11.0, there used to be one main admin, which could not be removed.

usergroups

getuser/{uuidUser}
An array containing an object for each group the user is part of. Each group object contains both the name and the UUID of the group.

getgrouplist
With this request, additional infos on the group are available, such as the userrights and the type.

nfcTags
- An array with an entry for each NFC tag associated with this user.
- Each tag is represented by a name and the NFC tag id

keycodes
- Even though this is an array, currently there is only one keycode for each user.
- The only attribute of each keycode object is the code itself.
- The code is a hashed representation of the keycode stored.

scorePWD
Provides/sets info on how strong a password is.

- -2 = not specified
- -1 = empty password
- 0 = low strength
- 1 = good,
- 2 = very good,
- 3 = most powerful

scoreVisuPWD

Same like scorePWD but for visualization passwords. (additional password that has to be entered, even tough the connection itself is already authenticated - e.g. for disarming a burglar alarm).

hashing

Passwords are never transmitted, not even encrypted - they are hashed on the client and only the hash is being transmitted to the Miniserver.

Creating a hash for a password

- Use "jdev/sys/getkey2/{username}" to retrieve the {salt} and {hashAlg}
  - salt is user-specific, long-lived
  - hashAlg specifies which hashing algorithm to use (recent versions use SHA256)
  - A "key" property is also provided, but not used here, it is a temporarily valid key to be used for hashing when authenticating.
- Use the {hashAlg} provided and the long lived {salt} to create a **uppercase** {passHash} for the new {password}
  - e.g.: SHA256({password} + ":" + {salt}).toUpperCase() → {passHash}

Creating a hash for a visu password

Same as creating a hash for a password, but "dev/sys/getvisusalt/{username}" is to be used instead of "jdev/sys/getkey2/{username}"

trustMember

If a user originates from a Trust Member/Peer this attribute is added containing the serial of the source Miniserver
New with Version 12.1

disabledBySource

Boolean. When set user was disabled by Trust source. Since V15.3

Additional User-Fields

New with Version 14.3

- uniqueUserId
  - Do not confuse this one with "userid" - which is the "NFC Code Touch ID"
  - uniqueUserIdis called "User ID" in Config and must be unique!
- firstName
- lastName
- email
- phone
- company
- department
- personalno
- title

- debitor
- customField1 - customField5
    - Individually named custom fields
    - for generic usage via API, these fields are not addressed with a custom name
    - for Title of custom fields, see Get Names of Custom User Fields

Trust

Available Peers

jdev/sps/trustusermanagement/peers

Response: Native answer json


```
{
  "peers": [
    {
      "serial": "504F94A00074",
      "name": "Loxone Miniserver",
      "intAddr": "192.168.5.199",
      "extAddr": "dns.loxonecloud.com"
    },
    {
      "serial": "504f94a00210",
      "name": "Home",
      "intAddr": "192.168.5.2",
      "extAddr": "home.aiglesberger.com"
    }
  ]
}
```


Discovery

jdev/sps/trustusermanagement/discover/{peerSerial}

- {peerSerial}:
  - Serial number of the peer you want to search on

Return-Value:

- UUID of created or adapted user
- errortext if failed

Request

jdev/sps/trustusermanagement/discover/504F94A00210

Response


```
{
  "serial": "504F94A00210",
```

```
"users": [
  {
    "name": "admin",
    "uuid": "09155ec7-0187-003c-1900000000000000",
    "used": false
  },
  {
    "name": "timo",
    "uuid": "096bfc48-01bb-0001-1900000000000000",
    "used": true
  },
  {
    "name": "Loxone",
    "uuid": "144087d0-0218-217b-ffff504f94a00074",
    "used": false
  },
  {
    "name": "Siri",
    "uuid": "14bc350e-009e-2f3f-ffffeee000d80cfd",
    "used": false
  },
  {
    "name": "Kerstin",
    "uuid": "15f45069-01db-3b8a-ffff504f94a00210",
    "used": false
  },
  {
    "name": "Test",
    "uuid": "16d29cb0-01e1-54b6-ffffeee000d80cfd",
    "used": true,
    "locked": true
  }
 ]
}
```


locked -> user used in usergroup and can therefore not be removed

Create user

jdev/sps/trustusermanagement/add/{peerSerial}/{userUuid}

- {peerSerial}
  - Serial number of the peer from which you want to add a user

- {userUuid}
  - Uuid of the user as specified in the discovery answer

Return-Value:

- errortext if failed

Request

jdev/sps/trustusermanagement/add/504F94A00210/15f45069-01db-3b8a-ffff504f94a00210

Response

```
{}
```

Response - Failed

```
{
  "error": "add failed"
}
```

Remove user

jdev/sps/trustusermanagement/remove/{peerSerial}/{userUuid}

- {peerSerial}
  - Serial number of the peer from which you want to remove a user
- {userUuid}
  - Uuid of the user as specified in the discovery answer

Return-Value:

- errortext if failed

Request

jdev/sps/trustusermanagement/remove/504F94A00210/15f45069-01db-3b8a-ffff504f94a00210

Response

```
{}
```

Response - Failed

```
{
  "error": "unknown user"
}
```

Edit users

jdev/sps/trustusermanagement/edit/{json}

- {json}
  - Json containing list of users that need to be removed/added
  - Array of objects for each peer, array contains an object for each user with the attribute 'used' and the uuid
- Request may be a Post request! json is in post data.

Return-Value:

- errortext if failed

Request

jdev/sps/trustusermanagement/edit/{"peerSerial":[{"uuid":"...", "used":true}]}

Response

```
{}
```

Response - Failed

```
{
  "error": "invalid command"
}
```

Revision History

V16.1

- ∉ Updated NFC Tag example

V14.3

- Additional User-Fields
- Webservice GetCustomUserFields

V14.2 - 2023.06.16

- User expiration action added (validUntil)

V13.0 - 2022.07.20 - Improved hashing doc

- documentation on hashing was confusing, clarified.

V12.1 - Trust

- Add / Remove Trust users
- Login with Host (Miniserver Name or Serialnumber)

V17.0