

Table of Contents

- User Management API** 1
- Structure** 1
- BS2User 1
- BS2UserSetting 2
- BS2UserPhoto 4
- BS2UserBlob 4
- BS2Job 5
- BS2UserBlobEx 5
- BS2UserSmallBlob 6
- BS2UserSmallBlobEx 7

User Management API

API that provides functions to enroll and delete users.

- [BS2_GetUserList](#): Gets the enrolled user ID list.
- [BS2_RemoveUser](#): Deletes user.
- [BS2_RemoveAllUser](#): Deletes all users.
- [BS2_GetUserInfos](#): Gets the user information of the given user ID.
- [BS2_GetUserInfosEx](#): Gets the user information of the given user ID. ([+ 2.4.0] including Job code and User phrase)
- [BS2_EnrolUser](#): Enrolls new user.
- [BS2_EnrolUserEx](#): Enrolls new user. ([+ 2.4.0] including Job code and User phrase)
- [BS2_EnrolUser](#): [+ 2.6.3] Enrolls new user.
- [BS2_EnrolUserEx](#): [+ 2.6.3] Enrolls new user. (including Job code and User phrase)
- [BS2_GetUserDatas](#): Gets selected data of user. (+ [2.5.0])
- [BS2_GetUserDatasEx](#): Gets selected data of user. ([+ 2.5.0] including Job code, User phrase)
- [BS2_GetSupportedUserMask](#): Gets user settings supported by the device.
- [BS2_EnrollUserSmall](#): [+ 2.6.3] Enrolls new user with efficient use of memory.
- [BS2_EnrollUserSmallEx](#): [+ 2.6.3] Enrolls new user with efficient use of memory.
- [BS2_GetUserSmallInfos](#): [+ 2.6.3] Gets the user information of the given user ID with efficient use of memory
- [BS2_GetUserSmallInfosEx](#): [+ 2.6.3] Gets the user information of the given user ID with efficient use of memory
- [BS2_GetUserSmallDatas](#): [+ 2.6.3] Gets selected data of user with efficient use of memory.
- [BS2_GetUserSmallDatasEx](#): [+ 2.6.3] Gets selected data of user with efficient use of memory.

Structure

BS2User

```
typedef struct {
    char userID[BS2_USER_ID_SIZE];
    uint8_t formatVersion;
    uint8_t flag;
    uint16_t version;
    uint8_t numCards;
    uint8_t numFingers;
    uint8_t numFaces;
    uint8_t reserved2[1];
    uint32_t authGroupID;
    uint32_t faceChecksum;
} BS2User;
```

1. *userID*

User ID provided as string, and has a range of 1 ~ 4294967295.

2. *formatVersion*

Not Used.

3. *flag*

Flag that shows the user's status. OR operation is available and the mask value is listed below.

| Value | Description |
|-------|---------------|
| 0x00 | None |
| 0x01 | User enrolled |
| 0x02 | User updated |
| 0x04 | User deleted |
| 0x80 | User disabled |

4. *version*

Not Used.

5. *numCards*

Number of cards mapped to user.

6. *numFingers*

Number of fingerprint templates mapped to user.

7. *numFaces*

Number of face templates mapped to user.

8. *authGroupID*

ID of group when face group matching is enabled.

9. *faceChecksum*

Not Used.

BS2UserSetting

```
typedef struct {
    uint32_t startTime;
    uint32_t endTime;
    uint8_t fingerAuthMode;
    uint8_t cardAuthMode;
    uint8_t idAuthMode;
    uint8_t securityLevel;
} BS2UserSetting;
```

1. *startTime*

Start time that a user can identify. When the value is 0, there are no limitations.

2. *endTime*

End time that that a user can identify. When the value is 0, there are no limitations.

3. fingerAuthMode

Finger authentication mode for user authentication.

| Value | Description |
|-------|--|
| 0 | Uses only fingerprint authentication |
| 1 | Uses fingerprint and PIN authentication |
| 254 | Cannot use |
| 255 | Undefined(Operates as defined in system) |

4. cardAuthMode

Card authentication mode for user authentication.

| Value | Description |
|-------|---|
| 2 | Uses only card authentication |
| 3 | Uses card and fingerprint authentication |
| 4 | Uses card and PIN authentication |
| 5 | Uses fingerprint or PIN after card authentication |
| 6 | Uses card, fingerprint, and PIN authentication |
| 254 | Cannot use |
| 255 | Undefined(Operates as defined in system) |

5. idAuthMode

ID authentication mode for user authentication.

| Value | Description |
|-------|--|
| 7 | Uses fingerprint authentication after entering user ID |
| 8 | Uses PIN authentication after entering user ID |
| 9 | Uses fingerprint or PIN authentication after entering user ID |
| 10 | Uses fingerprint and PIN authentication after entering user ID |
| 254 | Cannot use |
| 255 | Undefined(Operates as defined in system) |

6. securityLevel

Security level for fingerprint identification or face recognition.

| Value | Description |
|-------|---------------------------------|
| 0 | Default value defined in system |
| 1 | Lowest security level |
| 2 | Low security level |
| 3 | Normal security level |
| 4 | High security level |
| 5 | Highest security level |

BS2UserPhoto

```
typedef struct {
    uint32_t size;
    uint8_t data[BS2_USER_PHOTO_SIZE];
} BS2UserPhoto;
```

1. *size*

Size of the user profile image data.

2. *data*

Data of the profile image, which can be stored up to 16kb.

BS2UserBlob

```
typedef struct {
    BS2User user;
    BS2UserSetting setting;
    uint8_t name[BS2_USER_NAME_SIZE];
    BS2UserPhoto photo;
    uint8_t pin[BS2_PIN_HASH_SIZE];
    BS2CSNCard* cardObjs;
    BS2Fingerprint* fingerObjs;
    BS2Face* faceObjs;
    uint32_t accessGroupId[BS2_MAX_NUM_OF_ACCESS_GROUP_PER_USER];
} BS2UserBlob;
```

1. **user**

Structure that defines the basic user information.

2. **setting**

Structure that defines the configuration value for user identification.

3. **name**

User name having UTF-8 for string encoding.

4. **photo**

User profile image, which supports only Jpeg images.

5. **pin**

Personal Identification Number(PIN). It should be entered through *BS_MakePinCode* function.

6. **cardObjs**

Card list for user authentication that needs to exist as much as **user.numCards**. Refer to [Smartcard API](#) for data format.

In case of Secure Credential card, cardObjs array of BS2UserBlob structure should be filled and the user should be updated after Secure Credential card issuing.

7. **fingerObjs**

Fingerprint template for user authentication that needs to exist as much as **user.numFingers**. Refer to [Fingerprint API](#) for data format.

8. *faceObjs*

Face template for user authentication that needs to exist as much as **user.numFaces**. Refer to [Face API](#) for data format.

9. *accessGroupId*

List of access groups where users belong to which can be configured up to 16 groups.

BS2Job

```
typedef struct {
    uint8_t numJobs;
    uint8_t reserved[3];

    struct {
        BS2_JOB_CODE code;
        BS2_JOB_LABEL label;
    } jobs[BS2_MAX_JOB_SIZE];
} BS2Job;
```

1. *numJobs*

Number of job codes allocated to the user.

2. *reserved*

Reserved Space.

3. *jobs*

List of jobs.

BS2UserBlobEx

```
typedef struct {
    BS2User user;
    BS2UserSetting setting;
    uint8_t name[BS2_USER_NAME_SIZE];
    BS2UserPhoto photo;
    uint8_t pin[BS2_PIN_HASH_SIZE];
    BS2CSNCard* cardObjs;
    BS2Fingerprint* fingerObjs;
    BS2Face* faceObjs;
    BS2Job job;
    BS2_USER_PHRASE phrase;
    uint32_t accessGroupId[BS2_MAX_NUM_OF_ACCESS_GROUP_PER_USER];
} BS2UserBlobEx;
```

1. user

Structure that defines the basic user information.

2. setting

Structure that defines the configuration value for user identification.

3. name

User name having UTF-8 for string encoding.

4. photo

User profile image, which supports only Jpeg images.

5. pin

Personal Identification Number(PIN). It should be entered through *BS_MakePinCode* function.

6. cardObjs

Card list for user authentication that needs to exist as much as **user.numCards**. Refer to [Smartcard API](#) for data format.

7. fingerObjs

Fingerprint template for user authentication that needs to exist as much as **user.numFingers**. Refer to [Fingerprint API](#) for data format.

8. faceObjs

Face template for user authentication that needs to exist as much as **user.numFaces**. Refer to [Face API](#) for data format.

9. job

Job code that will be allocated to user.

10. phrase

Private message that will be displayed when the user authenticates. (FaceStation 2 Only)

11. accessGroupId

List of access groups where users belong to which can be configured up to 16 groups.

BS2UserSmallBlob

```
typedef struct {
    BS2User user;
    BS2UserSetting setting;
    uint8_t name[BS2_USER_NAME_SIZE];
    BS2UserPhoto* photo;
    uint8_t pin[BS2_PIN_HASH_SIZE];
    BS2CSNCard* cardObjs;
    BS2Fingerprint* fingerObjs;
    BS2Face* faceObjs;
    uint32_t accessGroupId[BS2_MAX_NUM_OF_ACCESS_GROUP_PER_USER];
};
```

```
} BS2UserSmallBlob;
```

1. **user**

Structure that defines the basic user information.

2. **setting**

Structure that defines the configuration value for user identification.

3. **name**

User name having UTF-8 for string encoding.

4. **photo**

User profile image, which supports only Jpeg images.

5. **pin**

Personal Identification Number(PIN). It should be entered through *BS_MakePinCode* function.

6. **cardObjs**

Card list for user authentication that needs to exist as much as **user.numCards**. Refer to [Smartcard API](#) for data format.

7. **fingerObjs**

Fingerprint template for user authentication that needs to exist as much as **user.numFingers**. Refer to [Fingerprint API](#) for data format.

8. **faceObjs**

Face template for user authentication that needs to exist as much as **user.numFaces**. Refer to [Face API](#) for data format.

9. **accessGroupId**

List of access groups where users belong to which can be configured up to 16 groups.

BS2UserSmallBlobEx

```
typedef struct {
    BS2User user;
    BS2UserSetting setting;
    uint8_t name[BS2_USER_NAME_SIZE];
    BS2UserPhoto* photo;
    uint8_t pin[BS2_PIN_HASH_SIZE];
    BS2CSNCard* cardObjs;
    BS2Fingerprint* fingerObjs;
    BS2Face* faceObjs;
    BS2Job job;
    BS2_USER_PHRASE phrase;
    uint32_t accessGroupId[BS2_MAX_NUM_OF_ACCESS_GROUP_PER_USER];
} BS2UserSmallBlobEx;
```


1. user

Structure that defines the basic user information.

2. setting

Structure that defines the configuration value for user identification.

3. name

User name having UTF-8 for string encoding.

4. photo

User profile image, which supports only jpeg images.

5. pin

Personal Identification Number(PIN). It should be entered through *BS_MakePinCode* function.

6. cardObjs

Card list for user authentication that needs to exist as much as **user.numCards**. Refer to [Smartcard API](#) for data format.

7. fingerObjs

Fingerprint template for user authentication that needs to exist as much as **user.numFingers**. Refer to [Fingerprint API](#) for data format.

8. faceObjs

Face template for user authentication that needs to exist as much as **user.numFaces**. Refer to [Face API](#) for data format.

9. job

Job code that will be allocated to user.

10. phrase

Private message that will be displayed when the user authenticates. (FaceStation 2 Only)

11. accessGroupIid

List of access groups where users belong to which can be configured up to 16 groups.

From:

<https://kb.supremainc.com/bs2sdk/> - **BioStar 2 Device SDK**

Permanent link:

https://kb.supremainc.com/bs2sdk/doku.php?id=en:user_management_api&rev=1560403335

Last update: **2019/06/13 14:22**