# 목차

**User Management API** ........ 1
*구조체* ........ 1
BS2User ........ 1
BS2UserSetting ........ 3
BS2UserPhoto ........ 4
BS2UserBlob ........ 5
BS2Job ........ 6
BS2UserBlobEx ........ 6
BS2UserSmallBlob ........ 7
BS2UserSmallBlobEx ........ 8
BS2UserSettingEx ........ 9
BS2UserFaceExBlob ........ 11

# User Management API

사용자 등록 및 삭제할 수 있는 함수를 제공하는 API입니다.

- BS2_GetUserList: 등록된 사용자 ID 리스트를 가져옵니다.
- BS2_RemoveUser: 사용자를 삭제합니다.
- BS2_RemoveAllUser: 모든 사용자를 삭제합니다.
- BS2_GetUserInfos: 주어진 사용자 ID의 정보를 가져옵니다.
- BS2_GetUserInfosEx: [+ 2.4.0] 주어진 사용자 ID의 정보를 가져옵니다. (Job code, User phrase 포함)
- BS2_EnrolUser: 새로운 사용자를 등록합니다.
- BS2_EnrolUserEx: [+ 2.4.0] 새로운 사용자를 등록합니다.(Job code, User phrase 포함)
- BS2_EnrollUser: [+ 2.6.3] 새로운 사용자를 등록합니다.
- BS2_EnrollUserEx: [+ 2.6.3] 새로운 사용자를 등록합니다.(Job code, User phrase 포함)
- BS2_GetUserDatas: 사용자 정보의 일부를 선별적으로 가져옵니다.
- BS2_GetUserDatasEx: [+ 2.5.0] 사용자 정보의 일부를 선별적으로 가져옵니다.(Job code, User phrase 포함)
- BS2_GetSupportedUserMask: 장치에서 지원하는 사용자 설정을 가져옵니다.
- BS2_EnrollUserSmall: [+ 2.6.3] 효율적으로 메모리를 사용하여, 새로운 사용자를 등록합니다.
- BS2_EnrollUserSmallEx: [+ 2.6.3] 효율적으로 메모리를 사용하여, 새로운 사용자를 등록합니다.
- BS2_GetUserSmallInfos: [+ 2.6.3] 효율적으로 메모리를 사용하여, 주어진 사용자 ID의 정보를 가져옵니다.
- BS2_GetUserSmallInfosEx: [+ 2.6.3] 효율적으로 메모리를 사용하여, 주어진 사용자 ID의 정보를 가져옵니다.
- BS2_GetUserSmallDatas: [+ 2.6.3] 효율적으로 메모리를 사용하여, 사용자 정보의 일부를 선별적으로 가져옵니다.
- BS2_GetUserSmallDatasEx: [+ 2.6.3] 효율적으로 메모리를 사용하여, 사용자 정보의 일부를 선별적으로 가져옵니다.
- BS2_EnrollUserFaceEx: [+ 2.7.1] `FaceStation F2`새로운 사용자를 등록합니다.
- BS2_GetUserInfosFaceEx: [+ 2.7.1] `FaceStation F2`주어진 사용자 ID의 정보를 가져옵니다.
- BS2_GetUserDatasFaceEx: [+ 2.7.1] `FaceStation F2`사용자 정보의 일부를 선별적으로 가져옵니다.

## 구조체

### 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 infoMask;
    uint32_t authGroupID;
    uint32_t faceChecksum;
} BS2User;
```

1. *userID*
사용자 식별자로 문자열 숫자이며, 1 ~ 4294967295의 범위를 가집니다.

2. *formatVersion*
사용되지 않음.

3. *flag*
사용자의 상태를 나타내는 flag값으로 OR 연산이 가능합니다. 마스크값은 아래와 같습니다.

| 값 | 설명 |
|---|---|
| 0x00 | None |
| 0x01 | 사용자가 생성됨 |
| 0x02 | 사용자가 갱신됨 |
| 0x04 | 사용자가 삭제됨 |
| 0x80 | 사용자가 비활성화됨 |

4. *version*
사용되지 않음.

5. *numCards*
사용자에게 맵핑된 카드 개수입니다.

6. *numFingers*
사용자에게 맵핑된 지문 개수입니다.

7. *numFaces*
사용자에게 맵핑된 얼굴 개수입니다.

8. *infoMask*
**사용자 정보의 취득**
사용자 정보 취득 시, infoMask는 현재 사용자에 어떤 정보들이 할당되어 있는지를 나타냅니다.

**사용자 정보의 갱신**
infoMask에 변경하고자 하는 mask를 지정하여 선별적으로 갱신이 가능합니다.

**`Credential`(카드/지문/얼굴) 정보의 갱신**
BS2User의 numCards, numFingers, numFaces가 0인지, fingerObjs, cardObjs, faceObjs, faceExObjs가 NULL여부 등, credential 정보의 지정 여부를 제일 먼저 검토하고, 부가적으로 infoMask를 검토합니다.
카드/지문/얼굴 정보를 0보다 큰 값으로 지정하고, infoMask에 mask를 설정하면 장치의 credential 정보를 갱신 할 수 있습니다.
예를들어, 장치에 대상이 되는 사용자의 지문이 2개가 있는 상태에서, infoMask에 BS2_USER_INFO_MASK_FINGER를 masking하고, numFingers = 1, fingerObjs에 지문을 할당하여 내려준다면, 장치는 새로 내려준 1개 지문 만을 갖습니다.
만일 지문의 추가 갱신이 목적이라면, 기존 2개의 지문에 새롭게 더해질 지문 1개가 더해진, 총 3개의 지문이 내려져야만 합니다.

**`Credential`(카드/지문/얼굴) 정보의 유지**
카드/지문/얼굴 각각의 credential 정보를 0으로 하고, infoMask의 mask를 켜주면, 장치는 기존에 가지고 있는 credential 정보들을 유지합니다.

**`Credential`(카드/지문/얼굴) 정보의 삭제**
카드/지문/얼굴 각각의 credential 정보를 0으로 하고, infoMask를 unmasking 하면, 장치는 각각

credential에 해당되는 정보를 삭제합니다.

| 값 | 설명 |
|---|---|
| 0x01 | BS2_USER_INFO_MASK_PHRASE |
| 0x02 | BS2_USER_INFO_MASK_JOB_CODE |
| 0x04 | BS2_USER_INFO_MASK_NAME |
| 0x08 | BS2_USER_INFO_MASK_PHOTO |
| 0x10 | BS2_USER_INFO_MASK_PIN |
| 0x20 | BS2_USER_INFO_MASK_CARD |
| 0x40 | BS2_USER_INFO_MASK_FINGER |
| 0x80 | BS2_USER_INFO_MASK_FACE |

9. *authGroupID*
얼굴 그룹 매칭을 사용할시 사용자에게 할당 할 그룹의 ID.

10. *faceChecksum*
사용되지 않음.

## BS2UserSetting

**도움말**
**FaceStation F2 이외**
**FaceStation F2**는 BS2UserSettingEx를 사용해 주십시오.

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

1. *startTime*
사용자 인증이 가능한 시작 시간을 의미합니다.
**978307200 (2001-01-01 00:00:00) 보다 큰 값**을 입력하여야 하며, **0으로 설정 시 제한 없음**을 의미합니다.

2. *endTime*
사용자 인증이 가능한 마지막 시간을 의미합니다.
**1924991999 (2030-12-31 23:59:59) 보다 작은 값**으로 입력하여야 하며, **0으로 설정 시 제한 없음**을 의미합니다.

3. *fingerAuthMode*
사용자 인증을 위한 지문 인증 설정 모드입니다.

| 값 | 설명 |
|---|---|
| 0 | 지문 인증만 사용 |
| 1 | 지문과 PIN 인증 사용 |
| 254 | 사용할 수 없음 |
| 255 | 정의되지 않음(시스템에 정의된 모드로 동작) |

4. *cardAuthMode*
사용자 인증을 위한 카드 인증 설정 모드입니다.

| 값 | 설명 |
|---|---|
| 2 | 카드 인증만 사용 |
| 3 | 카드와 지문 인증 사용 |
| 4 | 카드와 PIN 인증 사용 |
| 5 | 카드 인증 후 지문이나 PIN 인증 사용 |
| 6 | 카드, 지문, PIN 사용 |
| 254 | 사용할 수 없음 |
| 255 | 정의되지 않음(시스템에 정의된 모드로 동작) |

5. *idAuthMode*
사용자 인증을 위한 ID 인증 설정 모드입니다.

| 값 | 설명 |
|---|---|
| 7 | 사용자 ID 입력 후 지문 인증 사용 |
| 8 | 사용자 ID 입력 후 PIN 인증 사용 |
| 9 | 사용자 ID 입력 후 지문이나 PIN 인증 사용 |
| 10 | 사용자 ID 입력 후 지문과 PIN 인증 사용 |
| 254 | 사용할 수 없음 |
| 255 | 정의되지 않음(시스템에 정의된 모드로 동작) |

6. *securityLevel*
지문 인증이나 얼굴 인식을 위해 필요한 보안 수준입니다.

| 값 | 설명 |
|---|---|
| 0 | 시스템에 정의된 기본 값 |
| 1 | 최고 낮은 보안 수준 |
| 2 | 낮은 보안 수준 |
| 3 | 기본 보안 수준 |
| 4 | 높은 보안 수준 |
| 5 | 최고 높은 보안 수준 |

## BS2UserPhoto

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

1. *size*

사용자 프로파일 이미지 데이터의 크기입니다.

2. *data*
프로파일 이미지의 데이터이며, 최대 16kb까지 저장할 수 있습니다.

## 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*
사용자의 기본 정보를 정의한 구조체입니다.

2. *setting*
사용자 식별을 위한 설정값을 정의한 구조체입니다.

3. *name*
사용자 이름이며 문자열 인코딩은 UTF-8입니다.

4. *photo*
사용자 프로파일 이미지이며 Jpeg 이미지만 지원합니다.

5. *pin*
PIN 값이며 반드시 *BS_MakePinCode* 함수를 통해 암호화된 문자열을 입력해야 합니다.

6. *cardObjs*
사용자 인증을 위한 카드 리스트로 반드시 **user.numCards**만큼 존재해야 합니다. 데이터 형식은 Smartcard API를 참고하십시오.

7. *fingerObjs*
사용자 인증을 위한 지문 템플릿 리스트로 반드시 **user.numFingers**만큼 존재해야 합니다. 데이터 형식은 Fingerprint API를 참고하십시오.

8. *faceObjs*
사용자 인증을 위한 얼굴 템플릿 리스트로 반드시 **user.numFaces**만큼 존재해야 합니다. 데이터 형식은 Face API를 참고하십시오.

9. *accessGroupId*
사용자가 속한 출입 그룹을 나열한 리스트로 최대 16개까지 설정할 수 있습니다.

## 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*
사용자에 매핑된 Job 개수입니다.

2. *reserved*
예약된 공간입니다.

3. *jobs*
T&A에 사용자 Job 리스트입니다.

## 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*
사용자의 기본 정보를 정의한 구조체입니다.

2. *setting*
사용자 식별을 위한 설정값을 정의한 구조체입니다.

3. *name*
사용자 이름이며 문자열 인코딩은 UTF-8입니다.

4. *photo*
사용자 프로파일 이미지이며 Jpeg 이미지만 지원합니다.

5. *pin*

PIN 값이며 반드시 *BS_MakePinCode* 함수를 통해 암호화된 문자열을 입력해야 합니다.

6. *cardObjs*
사용자 인증을 위한 카드 리스트로 반드시 **user.numCards**만큼 존재해야 합니다. 데이터 형식은 Smartcard API를 참고하십시오.

7. *fingerObjs*
사용자 인증을 위한 지문 템플릿 리스트로 반드시 **user.numFingers**만큼 존재해야 합니다. 데이터 형식은 Fingerprint API를 참고하십시오.

8. *faceObjs*
사용자 인증을 위한 얼굴 템플릿 리스트로 반드시 **user.numFaces**만큼 존재해야 합니다. 데이터 형식은 Face API를 참고하십시오.

9. *job*
근태모드에서 사용자의 작업코드입니다.

10. *phrase*
인증시 장치 UI에서 표시되는 개인 메시지입니다.

| 지원 모델 | 지원 버전 |
|---|---|
| FaceStation 2 | V1.0.0 이상 |
| FaceStation F2 | V1.0.0 이상 |
| X-Station 2 | V1.0.0 이상 |

11. *accessGroupId*
사용자가 속한 출입 그룹을 나열한 리스트로 최대 16개까지 설정할 수 있습니다.

## 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*
사용자의 기본 정보를 정의한 구조체입니다.

2. *setting*
사용자 식별을 위한 설정값을 정의한 구조체입니다.

3. *name*
사용자 이름이며 문자열 인코딩은 UTF-8입니다.

4. *photo*
사용자 프로파일 이미지이며 Jpeg 이미지만 지원합니다.

5. *pin*
PIN 값이며 반드시 *BS_MakePinCode* 함수를 통해 암호화된 문자열을 입력해야 합니다.

6. *cardObjs*
사용자 인증을 위한 카드 리스트로 반드시 **user.numCards**만큼 존재해야 합니다. 데이터 형식은 Smartcard API를 참고하십시오.

7. *fingerObjs*
사용자 인증을 위한 지문 템플릿 리스트로 반드시 **user.numFingers**만큼 존재해야 합니다. 데이터 형식은 Fingerprint API를 참고하십시오.

8. *faceObjs*
사용자 인증을 위한 얼굴 템플릿 리스트로 반드시 **user.numFaces**만큼 존재해야 합니다. 데이터 형식은 Face API를 참고하십시오.

9. *accessGroupId*
사용자가 속한 출입 그룹을 나열한 리스트로 최대 16개까지 설정할 수 있습니다.

## 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*
사용자의 기본 정보를 정의한 구조체입니다.

2. *setting*
사용자 식별을 위한 설정값을 정의한 구조체입니다.

3. *name*
사용자 이름이며 문자열 인코딩은 UTF-8입니다.

4. *photo*
사용자 프로파일 이미지이며 Jpeg 이미지만 지원합니다.

5. *pin*

PIN 값이며 반드시 *BS_MakePinCode* 함수를 통해 암호화된 문자열을 입력해야 합니다.

6. *cardObjs*
사용자 인증을 위한 카드 리스트로 반드시 **user.numCards**만큼 존재해야 합니다. 데이터 형식은 Smartcard API를 참고하십시오.

7. *fingerObjs*
사용자 인증을 위한 지문 템플릿 리스트로 반드시 **user.numFingers**만큼 존재해야 합니다. 데이터 형식은 Fingerprint API를 참고하십시오.

8. *faceObjs*
사용자 인증을 위한 얼굴 템플릿 리스트로 반드시 **user.numFaces**만큼 존재해야 합니다. 데이터 형식은 Face API를 참고하십시오.

9. *job*
근태모드에서 사용자의 작업코드입니다.

10. *phrase*
인증시 장치 UI에서 표시되는 개인 메시지입니다.

| 지원 모델 | 지원 버전 |
|---|---|
| FaceStation 2 | V1.0.0 이상 |
| FaceStation F2 | V1.0.0 이상 |
| X-Station 2 | V1.0.0 이상 |

11. *accessGroupId*
사용자가 속한 출입 그룹을 나열한 리스트로 최대 16개까지 설정할 수 있습니다.

## BS2UserSettingEx

**도움말**
**FaceStation F2 만**
FaceStation F2 **이외의 장치**는 BS2UserSetting를 사용해 주십시오.

```
typedef struct {
    uint8_t faceAuthMode;
    uint8_t fingerprintAuthMode;
    uint8_t cardAuthMode;
    uint8_t idAuthMode;
    uint8_t reserved[28];
} BS2UserSettingEx;
```

`FaceStation F2`

1. *faceAuthMode*
사용자 인증을 위한 얼굴 인증 설정 모드입니다.

| 값 | 1단계 인증 | 2단계 인증 | 3단계 인증 | 4단계 인증 |
|---|---|---|---|---|
| 11 | 얼굴 | | | |
| 12 | 얼굴 | 지문 | | |
| 13 | 얼굴 | PIN | | |
| 14 | 얼굴 | 지문 또는 PIN | | |
| 15 | 얼굴 | 지문 | PIN | |
| 254 | 사용할 수 없음 | | | |
| 255 | 정의되지 않음(시스템 정의 모드) | | | |

2. *fingerprintAuthMode*
사용자 인증을 위한 지문 인증 설정 모드입니다.

| 값 | 1단계 인증 | 2단계 인증 | 3단계 인증 | 4단계 인증 |
|---|---|---|---|---|
| 16 | 지문 | | | |
| 17 | 지문 | 얼굴 | | |
| 18 | 지문 | PIN | | |
| 19 | 지문 | 얼굴 또는 PIN | | |
| 20 | 지문 | 얼굴 | PIN | |
| 254 | 사용할 수 없음 | | | |
| 255 | 정의되지 않음(시스템 정의 모드) | | | |

3. *cardAuthMode*
사용자 인증을 위한 카드 인증 설정 모드입니다.

| 값 | 1단계 인증 | 2단계 인증 | 3단계 인증 |
|---|---|---|---|
| 21 | 카드 | | |
| 22 | 카드 | 얼굴 | |
| 23 | 카드 | 지문 | |
| 24 | 카드 | PIN | |
| 25 | 카드 | 얼굴 또는 지문 | |
| 26 | 카드 | 얼굴 또는 PIN | |
| 27 | 카드 | 지문 또는 PIN | |
| 28 | 카드 | 얼굴 또는 지문 또는 PIN | |
| 29 | 카드 | 얼굴 | 지문 |
| 30 | 카드 | 얼굴 | PIN |
| 31 | 카드 | 지문 | 얼굴 |
| 32 | 카드 | 지문 | PIN |
| 33 | 카드 | 얼굴 또는 지문 | PIN |
| 34 | 카드 | 얼굴 | 지문 또는 PIN |
| 35 | 카드 | 지문 | 얼굴 또는 PIN |

| 값 | 1단계 인증 | 2단계 인증 | 3단계 인증 |
|---|---|---|---|
| 254 | 사용할 수 없음 | | |
| 255 | 정의되지 않음(시스템 정의 모드) | | |

4. *idAuthMode*
사용자 인증을 위한 ID 인증 설정 모드입니다.

| 값 | 1단계 인증 | 2단계 인증 | 3단계 인증 |
|---|---|---|---|
| 36 | ID | 얼굴 | |
| 37 | ID | 지문 | |
| 38 | ID | PIN | |
| 39 | ID | 얼굴 또는 지문 | |
| 40 | ID | 얼굴 또는 PIN | |
| 41 | ID | 지문 또는 PIN | |
| 42 | ID | 얼굴 또는 지문 또는 PIN | |
| 43 | ID | 얼굴 | 지문 |
| 44 | ID | 얼굴 | PIN |
| 45 | ID | 지문 | 얼굴 |
| 46 | ID | 지문 | PIN |
| 47 | ID | 얼굴 또는 지문 | PIN |
| 48 | ID | 얼굴 | 지문 또는 PIN |
| 49 | ID | 지문 | 얼굴 또는 PIN |
| 254 | 사용할 수 없음 | | |
| 255 | 정의되지 않음(시스템 정의 모드) | | |

5. *reserved*
예약된 공간입니다.

## BS2UserFaceExBlob

```
typedef struct
{
    BS2User user;
    BS2UserSetting setting;
    BS2_USER_NAME user_name;
    BS2UserPhoto* user_photo_obj;
    BS2_USER_PIN pin;
    BS2CSNCard* cardObjs;
    BS2Fingerprint* fingerObjs;
    BS2Face* faceObjs;                  // FS2, FL
    BS2Job job;
    BS2_USER_PHRASE phrase;
    BS2_ACCESS_GROUP_ID accessGroupId[BS2_MAX_NUM_OF_ACCESS_GROUP_PER_USER];
    BS2UserSettingEx settingEx;          // F2
    BS2FaceEx* faceExObjs;               // F2
} BS2UserFaceExBlob;
```

1. *user*
사용자의 기본 정보를 정의한 구조체입니다.

2. *setting*
사용자 식별을 위한 설정값을 정의한 구조체입니다.

3. *name*
사용자 이름이며 문자열 인코딩은 UTF-8입니다.

4. *photo*
사용자 프로파일 이미지이며 Jpeg 이미지만 지원합니다.

5. *pin*
PIN 값이며 반드시 *BS_MakePinCode* 함수를 통해 암호화된 문자열을 입력해야 합니다.

6. *cardObjs*
사용자 인증을 위한 카드 리스트로 반드시 **user.numCards**만큼 존재해야 합니다. 데이터 형식은 Smartcard API를 참고하십시오.

7. *fingerObjs*
사용자 인증을 위한 지문 템플릿 리스트로 반드시 **user.numFingers**만큼 존재해야 합니다. 데이터 형식은 Fingerprint API를 참고하십시오.

8. *faceObjs*
FaceStation2, FaceLite 사용자 인증을 위한 얼굴 템플릿 리스트로 반드시 **user.numFaces**만큼 존재해야 합니다. 데이터 형식은 Face API를 참고하십시오.

9. *job*
근태모드에서 사용자의 작업코드입니다.

10. *phrase*
인증시 장치 UI에서 표시되는 개인 메시지입니다.

| 지원 모델 | 지원 버전 |
|---|---|
| FaceStation 2 | V1.0.0 이상 |
| FaceStation F2 | V1.0.0 이상 |
| X-Station 2 | V1.0.0 이상 |

11. *accessGroupId*
사용자가 속한 출입 그룹을 나열한 리스트로 최대 16개까지 설정할 수 있습니다.

12. *settingEx*
FaceStation F2 개인인증모드를 설정할 수 있습니다. 지문과 얼굴을 함께 조합한 더 다양한 인증모드의 조합이 가능하게 되었습니다.

13. *faceExObjs*
FaceStation F2 사용자 인증을 위한 얼굴 템플릿 리스트로 반드시 **user.numFaces**만큼 존재해야 합니다. 데이터 형식은 Face API를 참고하십시오.

From:
https://kb.supremainc.com/bs2sdk./ - **BioStar Device SDK**

Permanent link:
**https://kb.supremainc.com/bs2sdk./doku.php?id=ko:user_management_api&rev=1661388626**

Last update: **2022/08/25 09:50**