목차

퀵 가이드

이 페이지는 BioStar SDK를 이용하여 애플리케이션을 개발하는 방법을 설명합니다.
아래 내용은 BioStar 애플리케이션에서 자주 사용하는 인터페이스의 사용법을 기술하고 있으며, 예제 코드는 C++ 기반으로 작성되었습니다. 다른 언어의 예제는 SDK 패키지의 Example 폴더를 참고하십시오.

변수명이 Obj로 끝나는 파라미터는 동적으로 메모리가 할당되므로 더 이상 사용하지 않을 경우에는 BS2_ReleaseObject 함수를 사용하여 메모리를 해제하십시오.
아래는 메모리 해제가 필요한 함수 목록입니다.

```
int BS2_GetDevices(void* context, BS2_DEVICE_ID** deviceListObj,
uint32_t* numDevice);
int BS2_GetLog(void* context, BS2_DEVICE_ID deviceId, BS2_EVENT_ID
eventId, uint32_t amount, BS2Event** logsObj, uint32_t* numLog);
int BS2_GetFilteredLog(void* context, BS2_DEVICE_ID deviceId, char* uid,
BS2_EVENT_CODE eventCode, BS2_TIMESTAMP start, BS2_TIMESTAMP end, uint8_t
tnakey, BS2Event** logsObj, uint32_t* numLog);
int BS2_GetUserList(void* context, BS2_DEVICE_ID deviceId, char**
uidsObj, uint32_t* numUid);
```

1 단계. SDK 초기화

SDK를 이용하기 위해서는 먼저 Context를 생성하고 초기화해야 합니다. Context 초기화 과정을 수행하지 않고 다른 함수들을 호출할 경우에는 BS_SDK_ERROR_UNINITIALIZED 에러가 반환됩니다.

```
int main(int argc, char* argv[])
{
    void* context = NULL;

    context = BS2_AllocateContext();
    if(context != NULL)
    {
        int result = BS2_Initialize(context);
        if(result == BS_SDK_SUCCESS)
        {
            // do something ...
        }
    }
    else
    {
        printf("Out of memory\n");
    }

    if(context != NULL)
    {
```

```
        BS2_ReleaseContext(context);
    }
    return ;
}
```

2 단계. 장치 연결

BioStar 애플리케이션과 장치는 *server mode*와 *direct mode*로 연결할 수 있습니다. *server mode*는 장치가 BioStar 애플리케이션으로 신호를 보내서 연결하는 방식이고, *direct mode*는 BioStar 애플리케이션이 장치로 신호를 보내서 연결하는 방식입니다. 장치는 *direct mode*가 초기값으로 설정되어 있으며, *direct mode*로 연결하는 방법은 아래와 같습니다.

- IP 주소와 Port를 이미 알고 있는 경우

```
    const char* deviceAddress = "192.168.1.2";
    uint16_t devicePort = 51211;
    uint32_t deviceId = ;
    int result = BS2_ConnectDeviceViaIP(context, deviceAddress,
devicePort, &deviceId);
    if(result == BS_SDK_SUCCESS)
    {
        printf("The device ID while connected to the network is
%d\n", deviceId);
    }
    else
    {
        printf("Failed to connect to device. (error code : 0x%x)\n",
result);
    }
```

- 장치를 검색한 뒤 연결할 경우

```
    uint32_t* deviceListObj = NULL;
    uint32_t numDevice = ;
    uint32_t selectedDeviceId = ;
    int result = BS2_SearchDevices(context);
    if(result == BS_SDK_SUCCESS)
    {
        result = BS2_GetDevices(context, &deviceListObj,
&numDevice);
        if(result == BS_SDK_SUCCESS)
        {
            // TODO select proper device id
            selectedDeviceId = deviceListObj[];

            // free device list object
            BS2_ReleaseObject(deviceListObj);

            result = BS2_ConnectDevice(context, selectedDeviceId);
        }
```

```
    }
```

3 단계. 장치 지원 기능 확인

장치와 연결이 완료되었다면 장치 정보를 가져와야 합니다. 장치 종류에 따라 일부 기능이 지원되지 않기 때문에 BioStar 애플리케이션은 장치에 맞춰 UI를 구성해야 합니다.[1] 장치 정보를 가져오기 위해서는 BS2_GetDeviceInfo 함수를 사용합니다.

```
uint32_t deviceId = 1;
BS2SimpleDeviceInfo deviceInfo;
int result = BS2_GetDeviceInfo(context, deviceId, &deviceInfo);
if(result == BS_SDK_SUCCESS)
{
    //TODO Customizing the UI
}
```

4 단계. 장치 설정 정보 가져오기

장치의 설정 정보는 BS2_GetXXXConfig[2] 함수를 통해 가져올 수 있습니다. 장치 설정에 대한 자세한 정보는 Configuration API를 참고하십시오.

```
uint32_t deviceId = 1;
BS2SimpleDeviceInfo deviceInfo;
BS2TNAConfig tnaConfig;
BS2IpConfig ipconfig;

int result = BS2_GetIPConfig(context, deviceId, &ipconfig);
if(result == BS_SDK_SUCCESS)
{
    //TODO handle it
}

if(deviceInfo.tnaSupported)
{
    result = BS2_GetTNAConfig(context, deviceId, &tnaConfig);
    if(result == BS_SDK_SUCCESS)
    {
        //TODO handle it
    }
}
```

5 단계. 사용자 등록

장치에 사용자를 등록하려면 헤더 정보를 올바르게 설정해야 합니다. 사용자가 사용할 수 있는 크리덴셜 종류는 PIN, 스마트 카드, 지문 인식, 얼굴 인식이 있으며, 장치에서 지원하는 크리덴셜 종

류는 4 단계. 장치 설정 정보 가져오기에서 BS2SimpleDeviceInfo로 얻은 정보를 사용하십시오.

사용자 정보와 크리덴셜 정보는 BS2_EnrolUser 함수로 등록할 수 있습니다. 자세한 내용은 User Management API를 참고하십시오.

```
uint32_t deviceId = 1;
BS2UserBlob userBlob;

//TODO fill up user header
int result = BS2_EnrolUser(context, deviceId, &userBlob);
if(result != BS_SDK_SUCCESS)
{
    //TODO handle error
}
```

사용자 헤더

사용자 헤더 정보는 장치 종류에 따라 필드가 다릅니다. 예를 들어, BioStation 2는 사용자 식별자, 사용자 이름 그리고 PIN을 사용할 수 있지만 BioEntry Plus는 오직 사용자 식별자 필드만 사용할 수 있습니다. 장치에서 사용하지 않는 나머지 필드 값은 무시되므로 BioStar 애플리케이션 개발자는 반드시 장치가 지원하는 기능을 기반으로 UI를 생성해야 사용자의 혼란과 불편을 막을 수 있습니다.

> 사용자 식별자는 반드시 숫자로만 구성되어야 하며, 유효범위는 1 ~ 4294967295까지입니다.

```
uint32_t deviceId = 1;
BS2SimpleDeviceInfo deviceInfo;
BS2UserBlob userBlob;

memset(&userBlob, , sizeof(BS2UserBlob));

//setup user id
strcpy(userBlob.user.userID, "user1");
userBlob.setting.startTime = time(NULL);
userBlob.setting.endTime = userBlob.setting.startTime + 7*24*60*60;
// 1 week
userBlob.setting.idAuthMode = BS2_AUTH_MODE_NONE;
userBlob.setting.securityLevel = BS2_USER_SECURITY_LEVEL_DEFAULT;

if(deviceInfo.cardSupported)
{
    userBlob.setting.cardAuthMode = BS2_AUTH_MODE_CARD_ONLY;
}
else
{
```

```
    userBlob.setting.cardAuthMode = BS2_AUTH_MODE_NONE;
}

if(deviceInfo.fingerSupported)
{
    userBlob.setting.fingerAuthMode = BS2_AUTH_MODE_BIOMETRIC_ONLY;
}
else
{
    userBlob.setting.fingerAuthMode = BS2_AUTH_MODE_NONE;
}

if(deviceInfo.userNameSupported)
{
    strcpy(userBlob.user_name, "Joshua");
}

if(deviceInfo.pinSupported)
{
    const char* plaintext = "my password";
    int result = BS2_MakePinCode(context, plaintext, userBlob.pin);
    if(result != BS_SDK_SUCCESS)
    {
        //TODO handle error
    }
}
```

카드 추가

BioStar 시스템에서는 MIFARE, IClass 등과 같은 다양한 스마트 카드를 지원하며, 한 사람당 최대 8개의 카드를 할당할 수 있습니다. 사용자에게 스마트 카드를 할당하려면 BS2_ScanCard 함수를 이용하여 카드 정보를 읽은 뒤 카드 정보를 User header 구조체에 맵핑하십시오.

```
uint32_t deviceId = 1;
BS2SimpleDeviceInfo deviceInfo;
BS2UserBlob userBlob;
BS2Card cardList[BS2_MAX_NUM_OF_CARD_PER_USER];

if(deviceInfo.cardSupported)
{
    int idx = ;
    for(; idx < BS2_MAX_NUM_OF_CARD_PER_USER ; idx++)
    {
        int result = BS2_ScanCard(context, deviceId, cardList + idx,
NULL);
        if(result != BS_SDK_SUCCESS)
        {
            //TODO handle error
```

```
            break;
        }
    }

    userBlob.user.numCards = idx;
    userBlob.cardObjs = cardList;
}
```

지문 템플릿 추가

지문 템플릿 추출은 지문 이미지를 스캔하고 템플릿 데이터를 추출하는 과정과 등록된 지문 템플릿이 서로 매치되는지 확인하는 과정으로 이루어져 있습니다. 지문 템플릿 추출이 완료되면 지문 템플릿 정보를 User header 구조체에 맵핑하십시오.

```
uint32_t deviceId = 1;
BS2SimpleDeviceInfo deviceInfo;
BS2UserBlob userBlob;
BS2Fingerprint fingerprintList[BS2_MAX_NUM_OF_FINGER_PER_USER];

if(deviceInfo.fingerSupported)
{
    int idx = ;
    uint32_t templateIndex = ;
    uint32_t fingerprintQuality =
BS2_FINGER_TEMPLATE_QUALITY_STANDARD;
    uint8_t templateFormat = BS2_FINGER_TEMPLATE_FORMAT_SUPREMA;
    int result = BS_SDK_SUCCESS;
    for(; idx < BS2_MAX_NUM_OF_FINGER_PER_USER; idx++)
    {
        for(templateIndex =  ; templateIndex <
BS2_TEMPLATE_PER_FINGER ;)
        {
            result = BS2_ScanFingerprint(context, deviceId,
fingerprintList + idx, templateIndex, fingerprintQuality,
            fingerprintFormat, NULL);
            if(result != BS_SDK_SUCCESS)
            {
                if (result == BS_SDK_ERROR_EXTRACTION_LOW_QUALITY ||
                    result == BS_SDK_ERROR_CAPTURE_LOW_QUALITY)
                {
                    printf("Low quality. try again.\n");
                }
                else
                {
                    //TODO handle error
                    break;
                }
            }
            else
```

```
            {
                templateIndex++;
            }
        }

        if(result != BS_SDK_SUCCESS)
        {
            break;
        }
    }

    if(result == BS_SDK_SUCCESS)
    {
        result = BS2_VerifyFingerprint(context, deviceId,
fingerprintList);
        if(result == BS_SDK_SUCCESS)
        {
            userBlob.user.numFingers = idx;
            userBlob.fingerObjs= fingerprintList;
        }
        else
        {
            if(result == BS_SDK_ERROR_NOT_SAME_FINGERPRINT)
            {
                printf("The fingerprint doesn't match.\n");
            }

            //TODO handle error
        }
    }
}
```

얼굴 템플릿 추가

얼굴 템플릿 추출은 얼굴을 스캔하고 템플릿 데이터를 추출하는 과정과 등록된 얼굴 템플릿이 서로 매치되는지 확인하는 과정으로 이루어져 있습니다. 얼굴 템플릿 추출이 완료되면 얼굴 템플릿 정보를 User header 구조체에 맵핑하십시오.

```
uint32_t deviceId = 1;
BS2SimpleDeviceInfo deviceInfo;
BS2UserBlob userBlob;
BS2Face Face[BS2_MAX_NUM_OF_FACE_PER_USER];

if (deviceInfo.faceSupported)
{
    int idx = ;
    uint32_t templateIndex = ;
    byte enrollThreshold;
    int result = BS_SDK_SUCCESS;
```

```
    for(; idx < BS2_MAX_NUM_OF_FACE_PER_USER; idx++)
    {
            result = BS2_ScanFace(context, deviceId, Face,
enrollThreshold, NULL);
            if(result != BS_SDK_SUCCESS)
            {
                    //TODO handle error
                    break;
            }
            if(result == BS_SDK_SUCCESS)
            {
                     Face[].faceindext = idx;
                     userBlob.faceObjs= face[]
            }

    }
}
```

6 단계. 로그 제어

BioEntry Plus, BioEntry W, BioLite Net, Xpass, Xpass S2는 최대 50,000개의 로그를 저장할 수 있으며, BioStation 2는 최대 3,000,000개의 로그를 저장할 수 있습니다. 로그 정보는 순환 큐로 관리하므로 저장할 공간이 부족하면 자동으로 오래된 로그부터 삭제하여 저장 공간을 확보합니다. 로그 유형에 대한 자세한 정보는 Log Management API를 참고하십시오.

로그 가져오기

로그를 가져오는 함수는 BS2_GetLog와 BS2_GetFilteredLog를 사용합니다. 대부분의 경우 BS2_GetLog 함수로 사용하지만 조건 검색이 필요하다면 BS2_GetFilteredLog 함수를 이용하십시오.

```
uint32_t deviceId = 1;
BS2Event* logs = NULL;
uint32_t numLogs = ;
uint32_t endTime = time(NULL);
uint32_t startTime = endTime - 7*24*60*60; //last week

// Get all logs
int result = BS2_GetLog(context, deviceId, , , &logs, &numLogs);
if(result == BS_SDK_SUCCESS)
{
    uint32_t idx = ;
    for(idx =  ; idx < numLogs ; idx++)
```

```
    {
        // TODO handle it
    }

    BS2_ReleaseObject(logs);
}

// Filtering logs
result = BS2_GetFilteredLog(context, deviceId, NULL, , startTime,
endTime, , &logs, &numLogs);
if(result == BS_SDK_SUCCESS)
{
    uint32_t idx = ;
    for(idx =  ; idx < numLogs ; idx++)
    {
        // TODO handle it
    }

    BS2_ReleaseObject(logs);
}
```

오픈 소스 라이선스

BioStar 2 Device SDK에 포함된 OpenSSL은 OpenSSL 라이선스와 Original SSLeay 라이선스를 따릅니다. OpenSSL 라이선스와 Original SSLeay 라이선스는 OpenSSL License와 Original SSLeay License에서 확인할 수 있습니다.

1)
예를들어, Xpass는 지문 센서와 얼굴 인식 센서가 없으므로 지문 인식과 얼굴 인식 기능이 필요하지 않습니다.

2)
BS2_GetFactoryConfig, BS2_GetSystemConfig, BS2_GetAuthConfig, BS2_GetDisplayConfig, BS2_GetIPConfig, BS2_GetTNAConfig, BS2_GetCardConfig, BS2_GetFingerprintConfig