



# 가

BioStar SDK  
 BioStar  
 C++ SDK Example

Obj 가  
 BS2\_ReleaseObject 가

```
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 . SDK

SDK Context Context 가  
 BS\_SDK\_ERROR\_UNINITIALIZED

```
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 .

BioStar *server mode* *direct mode* . *server mode*  
 가 BioStar , *direct mode* BioStar  
 , *direct mode* 가

- IP Port

```
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);
}
```

- 

```
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 .

BioStar <sup>1)</sup> UI 가  
 BS2\_GetDeviceInfo . 가

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

### 4 .

### 가

BS2\_GetXXXConfig<sup>2)</sup> 가  
 Configuration API .

```
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 .

PIN, , , , 가

# 4 . 가 BS2SimpleDeviceInfo

## BS2\_EnrolUser

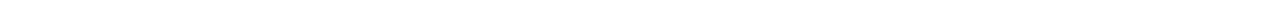
### User Management API

```

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
}

```



가 , BioStation 2  
 , PIN BioEntry Plus BioStar  
 가 UI

1 ~ 4294967295

```

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
    }
}
}

```

가

BioStar  
8

MIFARE, IClass

User header

BS2\_ScanCard

```

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;
}
```

---

가

User header

```
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
    }
}
}
```

---

가

User header

```
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[].faceindex = idx;
        userBlob.faceObjs = face[]
    }
}
}
}

```

6

BioEntry Plus, BioEntry W, BioLite Net, Xpass, Xpass S2 50,000  
, BioStation 2 3,000,000

[Log Management API](#)

가

가 [BS2\\_GetLog](#) [BS2\\_GetFilteredLog](#)  
[BS2\\_GetLog](#) [BS2\\_GetFilteredLog](#)

```

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);
}

```

[BioStar 2 Device SDK](#)      [OpenSSL](#)   [OpenSSL](#)      [Original SSLeay](#)  
[. OpenSSL](#)      [Original SSLeay](#)      [OpenSSL License](#)   [Original SSLeay License](#)

1) , Xpass 가

2) [BS2\\_GetFactoryConfig](#), [BS2\\_GetSystemConfig](#), [BS2\\_GetAuthConfig](#),  
[BS2\\_GetDisplayConfig](#), [BS2\\_GetIPConfig](#), [BS2\\_GetTNACConfig](#), [BS2\\_GetCardConfig](#),  
[BS2\\_GetFingerprintConfig](#)

From:  
<http://kb.supremainc.com/bs2sdk/> - **BioStar 2 Device SDK**  
 Permanent link:  
[http://kb.supremainc.com/bs2sdk/doku.php?id=ko:quick\\_guide](http://kb.supremainc.com/bs2sdk/doku.php?id=ko:quick_guide)  
 Last update: **2019/04/18 09:42**