

BS2_RemoveOsdpStandardDevice 1

..... 1

..... 1

..... 1

..... 1

(C++) 2

(C#) 3

BS2_AddOsdpStandardDevice
BS2_GetOsdpStandardDevice
BS2_GetAvailableOsdpStandardDevice
BS2_UpdateOsdpStandardDevice
BS2_RemoveOsdpStandardDevice
BS2_GetOsdpStandardDeviceCapability
BS2_SetOsdpStandardDeviceSecurityKey

(C++)

sample_bs2_removeosdpstandarddevice.cpp

```
BS2osdpStandardConfig config = { , };
vector<BS2_DEVICE_ID> removeData;

BS2_DEVICE_ID id = Utility::selectDeviceID(deviceList, false, false);
int sdkResult = cc.getOsdpStandardConfig(id, config);
if (BS_SDK_SUCCESS != sdkResult)
    return sdkResult;

uint32_t numOfActivated = cc.printOSDPDeviceID(config);
uint32_t numOfDevice = Utility::getInput<uint32_t>("How many devices do
you want to remove? (0~%u)", numOfActivated);
if ( < numOfDevice)
{
    for (uint32_t idx = ; idx < numOfDevice; idx++)
    {
        BS2_DEVICE_ID slaveID =
(BS2_DEVICE_ID)Utility::getInput<uint32_t>("[%u] Please enter the slave
ID to be removed.", idx + 1);
        removeData.push_back(slaveID);
    }

    vector<BS2osdpStandardDeviceResult> listResult;
    BS2osdpStandardDeviceResult* outResultObj = NULL;
    uint32_t outNumOfResult();
    int sdkResult = BS2_RemoveOsdpStandardDevice(context_, id,
const_cast<BS2_DEVICE_ID*>(removeData.data()), removeData.size(),
&outResultObj, &outNumOfResult);
    if (BS_SDK_SUCCESS != sdkResult)
    {
        printf("BS2_RemoveOsdpStandardDevice call failed: %d",
sdkResult);
    }

    if (outResultObj)
    {
        listResult.clear();
        for (uint32_t idx = ; idx < outNumOfResult; idx++)
```

```
        {
            listResult.push_back(outResultObj[idx]);
        }

        BS2_ReleaseObject(outResultObj);
    }
}

return sdkResult;
```

(C#)

[sample_bs2_removeosdpstandarddevice.cs](#)

```
BS20sdpStandardConfig config;
if (!CommonControl.getOsdpStandardConfig(sdkContext, deviceID, out
config))
    return;

UInt32 numOfActivated = ;
printOSDPDeviceID(ref config, ref numOfActivated);

string tempStr = String.Format("How many devices do you want to remove?
(0~{0})", numOfActivated);
Util.HighlightLineMulti(tempStr, "How many", "remove");
Console.Write(">>>> ");
int numOfDevice = Util.GetInput(1);
if ( < numOfDevice)
{
    List<UInt32> removeIDs = new List<UInt32>();
    for (int idx = ; idx < numOfDevice; idx++)
    {
        tempStr = String.Format(">>>> [{0}] Please enter the slave ID
to be removed.", idx + 1);
        Util.HighlightLine(tempStr, "device ID to be removed");
        Console.Write(">>>> ");
        removeIDs.Add(Util.GetInput((UInt32)));
    }

    List<BS20sdpStandardDeviceResult> listResult = new
List<BS20sdpStandardDeviceResult>();
    int structSize = Marshal.SizeOf(typeof(UInt32));
    IntPtr ptrArray = Marshal.AllocHGlobal(structSize *
removeIDs.Length);
    long ptrCurrent = ptrArray.ToInt64();
    BS2ErrorCode result = BS2ErrorCode.BS_SDK_SUCCESS;
    try
    {
```

```
int[] tempIDs = Array.ConvertAll(removeIDs, Convert.ToInt32);
Marshal.Copy(tempIDs, , ptrArray, tempIDs.Length);

IntPtr outResultObj = IntPtr.Zero;
UInt32 numOfResult = ;
result =
(BS2ErrorCode)API.BS2_RemoveOsdpStandardDevice(sdkContext, deviceID,
ptrArray, (UInt32)tempIDs.Length, out outResultObj, out numOfResult);
if (result != BS2ErrorCode.BS_SDK_SUCCESS)
{
    Console.WriteLine("Got error({0}).", result);
}
else
{
    IntPtr curResult = outResultObj;
    int resultSize =
Marshal.SizeOf(typeof(BS2OsdpStandardDeviceResult));
    for (UInt32 resultIdx = ; resultIdx < numOfResult;
resultIdx++)
    {
        BS2OsdpStandardDeviceResult item =
(BS2OsdpStandardDeviceResult)Marshal.PtrToStructure(curResult,
typeof(BS2OsdpStandardDeviceResult));
        //print(ref item, resultIdx);
        listResult.Add(item);
        curResult += resultSize;
    }

    API.BS2_ReleaseObject(outResultObj);
    Console.WriteLine("Call success.");
}
}
finally
{
    Marshal.FreeHGlobal(ptrArray);
}
}

return result;
```

From:

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

Permanent link:

https://kb.supremainc.com/bs2sdk./doku.php?id=ko:bs2_removeosdpstandarddevice

Last update: **2023/02/14 14:51**