WorldNet Windows Integration
Quick Start

Using NuGet, Import IDTechSDK_STD 3.2.4.398 (or later) into the project.

Implement the MessageCallback

```
private void MessageCallBack(IDTechSDK.IDT_DEVICE_Types type, DeviceState state, byte[]
data, IDTTransactionData cardData, EMV_Callback emvCallback, RETURN_CODE
transactionResultCode, string ident)
```

Set the MessageCallback as the SDK callback

```
IDT_Device.setCallbackIP(MessageCallBack);
```

Plug in a USB device and monitor the Connect/Disconnect callback to understand device connection status

```
case DeviceState.Connected:
   output( "Connected " + IDTechSDK.Profile.IDT_DEVICE_String(type, connect,
ident),ident);
   break;
case DeviceState.Disconnected:
   output( "Disconnected " + IDTechSDK.Profile.IDT_DEVICE_String(type, connect,
ident),ident);
    break;
```

Please refer to the main SDK documentation for more advance topic like communicating with multiple devices and connecting through Serial or IP

Create a data object and populate it with transaction details

```
if (wn == null) wn = new ();

 WorldNet.WorldNetData data = new WorldNet.WorldNetData();

 data.terminal = " 4480001";
 data.apiKey =
"afae1c3fa41612acca86b435f65edcf3c930e3fe84b4dab4ef14321eb242890e09919c6e45e0a0506f523
db0b5e6da2589c5182b1aca06b91558100619576e88";
data.license = " 2y2r0h6sorxb-5ufv0z2ao3jb"; //if live/production transaction, a valid
license must be provided
 data.transactionType = WorldNet.TRANSACTION_TYPE.TRANSACTION_TYPE_SALE;

//specify Sale, Pre-Auth, Completion, Void or Return
// data.transactionType =  WorldNet.TRANSACTION_TYPE.TRANSACTION_TYPE_PRE_AUTH;
// data.transactionType =  WorldNet.TRANSACTION_TYPE.TRANSACTION_TYPE_COMPLETION;
// data.transactionType =  WorldNet.TRANSACTION_TYPE.TRANSACTION_TYPE_VOID;
// data.transactionType =  WorldNet.TRANSACTION_TYPE.TRANSACTION_TYPE_RETURN;

 data.amount = "1.00";   //specify transaction Amount
 data.timeout =  30;  //Timeout for transaction
 data.isTest = true;  //if test transaction, set to TRUE.  If live transactions, set
to FALSE
 data.transactionID = ""; //required if COMPLETION, REFUND, VOID.  Otherwise, leave
blank
 data. orderID = ""; //required value.  If blank, SDK will put in random number
 data.deviceID = ""; // set to device identifier to target specific device, or leave
empty to target first/only device.
 data.Operator = false;  // operator for Completion, Void or Return. Blank will put
value "VP3300"
data.refundReason = "Refund for overpayment";   // for Refund, must provide a reason.
If blank, transaction ID will be used
data.additionalTags = "";   // used for passing additional tags at the start of a
transactions


 wn.returnRequest = true; //set to TRUE if you would like the request packet returned
with the response
```

Define the callback that will receive the transaction results. The cardData is an optional element if you wanted to understand what initial card data was captured.

```
void WorldNetCallback(IDT_DEVICE_Types sender, string jsonResponse, RETURN_CODE
resultCode, EMV_RESULT_CODE emvCode, string ident, IDTTransactionData cardData)
        {

            output("\r\n\r\nResponse received from WorldNet: " + resultCode.ToString() +
"\r\n\r\n", ident);
            output(jsonResponse, ident);
        }
```

Execute the transaction, passing the data object, the callback, and request flag

```
bool requestOnly = false; // If this is TRUE, the SDK will return the request packet
instead of contacting  with it. For integrations that want to control communication with
servers

string ident = data.deviceID; // If you want to target a specific device, you specify
either as a parameter you pass to executeTransaction, or through data.deviceID


RETURN_CODE rt = wn.executeTransaction(data, WorldNetCallback, requestOnly, ident);

if (rt == RETURN_CODE.RETURN_CODE_DO_SUCCESS)
{
     output("Start Transaction Successful\r\n", IDT_Device.currentIdent());
}
else
{
     output("Start transaction failed Error Code: " + "0x" + String.Format("{0:X}",
(ushort)rt) + ": " + IDTechSDK.errorCode.getErrorString(rt) + "\r\n",
IDT_Device.currentIdent());
}
```

Await and parse the results in the WorldNetCallback.

```
void WorldNetCallback(IDT_DEVICE_Types sender, string jsonResponse, RETURN_CODE
resultCode, EMV_RESULT_CODE emvCode, string ident, IDTTransactionData cardData,)
        {
            output("\r\n\r\nResponse received from Worldpay: " + resultCode.ToString() +
"\r\n\r\n", ident);
            output(jsonResponse, ident);
        }
```

The device must be configured with proper terminal settings, AID's, and CAPKs. Please load the configuration file WorldNet_Device_Config.json to your device before running transactions.
You can use SDK Demo, or UpdateIDT to perform the update
The device must have the correct key. If a test device, it must have demo BDK 0123456789ABCDEFFEDCBA9876543210, and if production, it must have a secure data encryption key

Going from Demo Mode to Production Mode

Once you are successful in performing test transactions, to perform live transactions, you will need to change the following four fields:

```
data.terminal = "";  //needs production value
data.apiKey = "";   //needs production value
data.isTest = false;  //must be false to perform live transactions
```

```
data.license = "";  //needs production license
```

You must provided production values for terminal and API Key. In addition, you must change isTest to false so your transaction will be directed to the live production server at WorldNet.