

## Frequently asked Questions and Answers

**Q.** I have a number of devices configured in UPDD. My application needs to determine which is the first enabled device. How do I do this in C++?

**A.** The following code will do just that:-

```
int aDeviceId; unsigned long lEnabled= 0;

for (int i=0;;i++)
{
    if ((aDeviceId = TBApiGetRelativeDevice(i)) != 0)
    {
        if (TBApiGetSettingDWORD(aDeviceId,_T("Enabled"), &lEnabled))
            if (lEnabled) break;
    }
    else
    {
        break;
    }
}
```

**Q.** I need my device to be disabled on startup. When my application runs, it needs to enable the device. How do I achieve this?

**A.** The device can be disabled on startup by setting the tbupdd.ini setting **InitialMousePortEnabled** in [\[UPDD\]](#) to zero. Your application should call [TBApiMousePortInterfaceEnable](#) to enable the device.

**Q.** How can I determine if the touch device is currently enabled?

**A.** In C++

```
// To determine if the first device is enabled
DWORD bEnabled ;
TBApiGetSettingDWORD(TBApiGetRelativeDevice(0),_T("Enabled",&bEnabled);
```

```
// To enable the first device
TBApiSetSettingDWORD(TBApiGetRelativeDevice(0),_T("Enabled"),1);
TBApiApply();
```

```
// To disable the first device
TBApiSetSettingDWORD(TBApiGetRelativeDevice(0),_T("Enabled"),0);
TBApiApply();
```

**Q.** Does your API work with C# and if so, how

**A.** The following example demonstrates how to call a (C++) DLL from C Sharp, in this case a function (func\_dll) in a DLL called MinGW\_dll

```
using System.Runtime.InteropServices;
using System;

class call_dll {

    [StructLayout(LayoutKind.Sequential, Pack=1)]
    private struct STRUCT_DLL {
        public Int32 count_int;
        public IntPtr ints;
    }

    [DllImport("mingw_dll.dll")]
    private static extern int func_dll(
        int an_int,
        [MarshalAs(UnmanagedType.LPArray)] byte[] string_filled_in_dll,
        ref STRUCT_DLL s
    );

    public static void Main() {

        byte[] string_filled_in_dll = new byte[21];

        STRUCT_DLL struct_dll = new STRUCT_DLL();
        struct_dll.count_int = 5;
        int[] ia = new int[5];
        ia[0] = 2; ia[1] = 3; ia[2] = 5; ia[3] = 8; ia[4] = 13;

        GCHandle gch = GCHandle.Alloc(ia);
        struct_dll.ints = Marshal.UnsafeAddrOfPinnedArrayElement(ia, 0);

        int ret=func_dll(5,string_filled_in_dll, ref struct_dll);

        Console.WriteLine("Return Value: " + ret);
        Console.WriteLine("String filled in DLL: " + System.Text.Encoding.ASCII.GetString(string_filled_in_dll));
```

```
}  
}
```

Good article is also available here:

[http://www.codeproject.com/KB/cs/C\\_DLL\\_with\\_Csharp.aspx](http://www.codeproject.com/KB/cs/C_DLL_with_Csharp.aspx)