Inside Out Networks

Anywhere USB Library API Specification

Revision 1.2

April 9, 2008

Revision History

Revision	Date	Comments
1.1	12/8/2003	Added AWUSB_STATUS_NOT_CONNECTED and AWUSBSYS_STATUS_DELETE_FAILURE return codes to AwUsbDisconnect.

Anywhere USB Library API Specification

Intro

This specification is for a set of utility functions for AnywhereUsb. These functions have been implemented in a Windows DLL named **AwUsbApi.dll**. The primary purpose of AwUsbApi.dll is to provide customers a Windows 2000/XP/2003/NT40 application program interface to acquire and relinquish connections to AwUsb hubs and to monitor their AwUsb network of Concentrator.

AwUsbApi.dll exports the following functions:

1. **AwUsbConnect**
2. **AwUsbDisconnect**
3. **AwUsbGetConnectionStatus**

The following is the details of these functions.

AwUsbConnect

This function is called to establish a connection with an AwUsb Hub.

```
AWUSB_STATUS
AwUsbConnect (
   IN    LPCWSTR                 Hub,
   IN    PAWUSB_STATUS           Status,
   IN    DWORD                   Timeout,
   IN    HANDLE                  hEvent OPTIONAL
);
```

Parameters:

Hub
[in] AwUsb Hub IP Address in string format (e.g. “10.0.0.1”).

Status
[out] Updated upon return from function, and, when hEvent is non-NULL, after the connect operation completes. See *Return Values*.

Timeout
[in] Specifies how long to wait for connection to succeed, in milliseconds. Must be non-zero. Set value to INFINITE in order to wait forever. Returns AWUSB_STATUS_TIMEOUT if timeout interval elapses.

hEvent
[in] Handle to an event which will be set to the signaled state when the operation has been completed. If hEvent is NULL, this function will block until it completes. If hEvent is non-NULL, then the function returns immediately, usually with AWUSB_STATUS_PENDING unless an error has occurred. To create an event object, use the CreateEvent function. An application must wait until hEvent has been signaled before calling CloseHandle.

Return Values:

AWUSB_STATUS_CONNECTED
Indicates that a connection has been successfully established with the AwUsb Hub.

AWUSB_STATUS_ALREADY_CONNECTED
Indicates that a connection was already established to this host at the time this function was called.

AWUSB_STATUS_INVALID_PARAMETER
Indicates that one or more of the parameters was invalid such as an invalid IP address.

AWUSB_STATUS_TIMEOUT
Indicates that a connection was not established within the timeout interval.

AWUSB_STATUS_CANCELLED
Indicates that the connect operation was cancelled because the application called AwUsbDisconnect before the timeout elapsed.

AWUSB_STATUS_PENDING
Indicates that status is pending. This code is returned when the function would otherwise block but hEvent is non-NULL. When the operation completes, *Status* is updated with the completion status and hEvent is signaled.

Remarks:

This function when successful is functionally equivalent to plugging in a USB hub to the PC. All devices attached to the AwUsb hub will be enumerated by the Plug and Play manager. When the application is finished with any devices attached to the hub, it should call AwUsbDisconnect to make the hub available to other hosts.

If this function fails, no further attempts will be made to connect to the hub until a subsequent AwUsbConnect is reissued.

The connection established by AwUsbConnect only persists until:

1) The AwUsb hub is disconnected by calling AwUsbDisconnect
2) The host machine is rebooted
3) The connection is unexpectedly lost (e.g. hub disconnected from network). Note that the application can recognize a lost connection by periodically calling AwUsbGetConnectionStatus (described below).

The application must call AwUsbDisconnect if it wishes to cancel AwUsbConnect before timing out. *hEvent* (if non-NULL) will be signaled and *Status* will be updated to AWUSB_STATUS_CANCELLED. AwUsbDisconnect may also be called from another thread to cancel AwUsbConnect is it is synchronously blocking a thread (hEvent is NULL) in which case AwUsbConnect will return with a status of AWUSB_STATUS_CANCELLED.

AwUsbConnect will keep trying to connect to the hub even if it is currently connected to another host. If desired, the application can call AwUsbGetConnectionStatus before attempting to connect in order to determine if a hub is already in use by another host.

AwUsbDisconnect

This function is called to disconnect an AwUsb Hub from the local host and make it available to other hosts.

```
AWUSB_STATUS
AwUsbDisconnect (
   IN LPCWSTR        Hub
);
```

Parameters:

Hub
[in] AwUsb Hub IP Address in string format.

Return Values:

AWUSB_STATUS_SUCCESS
Indicates that disconnect has succeeded. AwUsbDisconnect will stop any further attempts to connect to the AwUsb hub until a subsequent AwUsbConnect is issued.

AWUSB_STATUS_INVALID_PARAMETER
Indicates that one or more of the parameters was invalid such as an invalid IP address.

AWUSBSYS_STATUS_DELETE_FAILURE
Indicates attempt to delete a persistent AwUsb hub connection. This connection must be deleted by the AnywhereUSB Configuration Utility.

AWUSB_STATUS_NOT_CONNECTED
Indicates an attempt to disconnect an AwUsb hub which was not connected. AwUsbDisconnect will stop any further attempts to connect to the AwUsb hub until a subsequent AwUsbConnect is issued.

Remarks:

This function when successful is functionally equivalent to unplugging a USB hub from the PC. The Device Manager will remove the devices attached to the AwUsb hub. AwUsbDisconnect can also be used to cancel AwUsbConnect (see AwUsbConnect section).

AwUsbGetConnectionStatus

This function gets the connection status of an AwUsb Hub.

```
AWUSB_STATUS
AwUsbGetConnectionStatus (
    IN      LPCWSTR           Hub,
    OUT     PDWORD            IpAddress
    OUT     PAWUSB_STATUS     Status,
    IN      DWORD             Timeout,
    IN      HANDLE            hEvent OPTIONAL,
);
```

Parameters:

Hub
[in] AwUsb Hub IP Address in string format.

IpAddress
[out] 32-bit IP Address of host (if any) to which hub is connected. *IpAddress* is in host byte order and can be converted to TCP/IP network byte order using the WinSock *htonl* function. *IpAddress* is only valid when *Status* is AWUSB_STATUS_CONNECTED or AWUSB_STATUS_IN_USE.

Status
[out] Updated upon return from function, and, when hEvent is non-NULL, after the status operation completes. See *Return Values*.

Timeout
[in] Specifies how long to wait for status from the AwUsb hub, in milliseconds. Must be non-zero. Maximum timeout is 10,000 ms (10secs). Returns AWUSB_STATUS_TIMEOUT if interval elapses.

hEvent
[in] Handle to an event which will be set to the signaled state when the operation has been completed. If hEvent is NULL, this function will block until it completes. If hEvent is non-NULL, then the function returns immediately, usually with AWUSB_STATUS_PENDING unless an error has occurred. To create an event object, use the CreateEvent function.

Return Values:

AWUSB_STATUS_CONNECTED
Hub at AwUsbHubAddress is connected to this Host. The IP address of the local host which is connected to the AwUsb hub is returned in *IpAddress*.

AWUSB_STATUS_CONNECTING
This Host is in the process of connecting to hub

AWUSB_STATUS_IN_USE
Hub was found, but is unavailable. The IP address of the host which is connected to the AwUsb hub is returned in *IpAddress*.

AWUSB_STATUS_AVAILABLE
Hub was found and is available. It is not connected to a host. *IpAddress will be 0.*

AWUSB_STATUS_TIMEOUT
Indicates that the hub was not found within timeout interval.

AWUSB_STATUS_INVALID_PARAMETER
Indicates that one or more of the parameters was invalid such as an invalid IP address.

AWUSB_STATUS_PENDING
Indicates that status is pending. This code is returned when the function would otherwise block but hEvent is non-NULL. When the operation completes, *Status* is updated with the completion status and hEvent is signaled.

Remarks:

This function returns the status of an AwUsb hub. Function returns immediately if the local host is connected to the hub or the local host is attempting to connect to the hub. Otherwise the function attempts to contact the hub and query its availability status.

Note that it is possible for another host to acquire the hub between receiving a status of AWUSB_STATUS_AVAILABLE from AwUsbGetConnectionStatus and calling AwUsbConnect.