JSR80 API Specification

Dan Streetman
ddstreet@us.ibm.com
October 23, 2002

Contents

List of Tables

List of Figures

1 Entry Point

The entry point for javax.usb is the *UsbHostManager* class. This class instantiates and provides access to the implementation's main access class, which is an implementation of *UsbServices*. This implementation class must provide a no-parameter constructor and it must be listed in the (required) properties file, *javax/usb/res/jusb.properties*, as the value for the key *javax.usb.os.UsbServices*.

The *UsbServices* provides add and remove methods for *UsbServicesListener*s, which monitors when devices are connected or added to the system. Also the class provides access to the root *UsbHub*. This hub is the root of the entire USB topology tree. It does not correspond to a physical hub, but instead is a virtual hub to which all physical Host Controller hubs are connected. Each of the physical Host Controller hubs contains the devices actually connected to the system.

2 Topology Tree

The USB topology tree is a representation of the physical device connection layout. It consists of a tree-based structure of *UsbDevice*s, some of which are *UsbHub*s. It starts at the virtual root *UsbHub*, to which devices (which are actually Host Controller hubs) are connected. Each of these hubs may have devices connected on one or more of its *UsbPort*s. These connected devices may actually be a hub, in which case they may have more devices connected to their port(s). This is the way that USB devices are connected to a system, with the exception of course of the virtual root hub. That is created and managed by the JSR80 implementation. See the section on *UsbHubs and UsbDevices* for details on the virtual root hub.

3 UsbHubs and UsbDevices

A *UsbHub* is an extended *UsbDevice*. It has methods specific to a hub. Those methods allow the user to access the hub's *UsbPort*s and attached devices.

A *UsbDevice* represents a physical USB device. It provides methods for accessing the parent port that the device is connected to, accessing the device's descriptor, accessing the device's configurations and strings, and adding or removing a *UsbDeviceListener* which monitors when the device is disconnected as well as when communication occurs on the Default Control Pipe. It also provides methods to communicate via the Default Control Pipe.

4 UsbDevice Components

Each *UsbDevice* is made up of several parts. These parts start with the device's configuration(s). A USB device is required to contain at least one configuration, and only one configuration may be active at once, meaning the device must be configured in one and only one certain way, that it defines. A device may also be in an unconfigured state, but normally the operating system software initializes the device to its default configuration, which is configuration number 1 (0 is reserved for the unconfigured state). Each configuration is represented by a *UsbConfig*. The device maintains which of its configs is active, if any.