Request #: HUTRR117
Title: Keyboard Backlight
Spec Release: 1.6
Requester: Matthew Williams
Company: Microsoft

Current Status: **Approved**
Priority: Normal

Required Voter: Logitech
Required Voter: Intel
Required Voter: Google

Voting Begins: 2nd April 2025
Voting Ends: 9th April 2025
Voting Result: 7-0

Summary:
Adding new Usages to support the control of keyboard backlight devices.

Background:
Many available keyboards support a controllable backlight, illuminating keys in low-light conditions.

The existing ‘HUTRR73-FnKeyKeyboardBacklightBrightness’ described many backlight Usages, available to inform the system when a device-controller brightness was changed (for assistive technologies). There is no provision however to describe a device where the backlight is controlled by the system itself.

It is proposed to introduce new Usages to describe a system-controllable keyboard backlight.

Proposal:

Add/modify to table 15.1 Consumer Page

UsageId	**Usage Name**	**Usage Type**
0x07	**Keyboard Backlight**	CA
0x515	Keyboard Brightness Next	OSC
0x516	Keyboard Brightness Previous	OSC
0x517	Keyboard Backlight Level Suggestion	SV

Add/modify 15.22 Keyboard Backlight Controls

Usage Type	Usage Type	Description
Keyboard Backlight	CA	Identifies a device to control the backlight of a specific keyboard. *It is expected that this device is defined alongside a Keyboard or Keypad CA.*
Keyboard Brightness Next	OSC	Indicates to the system to cycle to the 'next' brightness-level (as defined by the system) for this device. This differs from 'Keyboard Brightness Increment', which indicates a strictly single-step monotonically increasing brightness-level for each invocation. *For example, a system may choose to increase brightness by irregular steps (for each invocation), overflowing to minimum brightness once a maximum is reached.*
Keyboard Brightness Previous	OSC	Indicates to the system to cycle to the 'previous' brightness-level (as defined by the system) for this device. This differs from 'Keyboard Brightness Decrement', which indicates a strictly single-step monotonically decreasing brightness-level for each invocation. *For example, a system may choose to decrease brightness by irregular steps (for each invocation), underflowing to maximum brightness once a minimum is reached.*
Keyboard Backlight Level Suggestion	SV	Indicates to the Host, the device's brightness-level *preferences* to cycle through when 'Keyboard Brightness Next', or Keyboard Brightness Previous' is actuated. *For example, a device may suggest 3 levels (3nits, 10nits, 15nits), indicating when the 'Keyboard Brightness Next' is actuated, the Host would cycle from 3nits, 10nits, 15nits to 3nits (etc…)* **Note: This is strictly an indication of preference to the Host; the device cannot take a dependency on what level the Host will ever set.**

Sample Descriptor .wara:

A simple ‘Keyboard Backlight’ device capable of emitting 255nits. Three brightness-levels are suggested to the Host (between 0 and 255bits).

Expected to be paired with an actual Keyboard device.

*Below document authored with **Waratah** (https://github.com/microsoft/hidtools)*

```
[[unit]]
name = 'meter'
centimeter = [100.0, 1.0]

[[unit]]
name = 'nits'
candela = [1.0, 1.0]
meter = [1.0, -2.0]

[[usagePage]]
name = 'Consumer'

    [[usagePage.usage]]
    id = 0x07
    name = 'Keyboard Backlight'
    types = ['CA']

    [[usagePage.usage]]
    id = 0x515
    name = 'Keyboard Brightness Next'
    types = ['OSC']

    [[usagePage.usage]]
    id = 0x516
    name = 'Keyboard Brightness Previous'
    types = ['OSC']

    [[usagePage.usage]]
    id = 0x517
    name = 'Keyboard Backlight Level Suggestion'
    types = ['SV']

[[applicationCollection]]
usage = ['Consumer', 'Keyboard Backlight']

    # Simple key to cycle through available brightness-levels.
    # The Host determines the brightness-level to set, assertion only indicates that the 'next' should be
set.
    [[applicationCollection.inputReport]]

        [[applicationCollection.inputReport.variableItem]]
        usage = ['Consumer', 'Keyboard Brightness Next']
        logicalValueRange = [0, 1]
        reportFlags = ['Relative', 'PreferredState']

        [[applicationCollection.inputReport.variableItem]]
        usage = ['Consumer', 'Keyboard Brightness Previous']
        logicalValueRange = [0, 1]
        reportFlags = ['Relative', 'PreferredState']

    # Keyboard provided brightness-levels suggestions for Host.
```

```
# Here, the device suggests 3 brightness-levels, which the Host free to override as it sees fit.
[[applicationCollection.featureReport]]

    [[applicationCollection.featureReport.variableItem]]
    usage = ['Consumer', 'Keyboard Backlight Level Suggestion']
    sizeInBits = 8
    logicalValueRange = 'maxUnsignedSizeRange'
    unit = 'nits'
    reportFlags = ['constant']
    count = 3

# Host control of the keyboard brightness-level.
# Device has been previously calibrated to understand what LED 'power-level' corresponds to emitted nits.
# Here, the device supports 0-16 nits, in single-nit increments.
[[applicationCollection.outputReport]]

    [[applicationCollection.outputReport.variableItem]]
    usage = ['Consumer', 'Keyboard Backlight Set Level']
    sizeInBits = 8
    logicalValueRange = 'maxUnsignedSizeRange'
    unit = 'nits'
```

Sample Descriptor:

Below document output from above .wara contents

```
------------------------------------------------------------------------
0x05, 0x0C,                    // UsagePage(Consumer[0x000C])
0x09, 0x07,                    // UsageId(Keyboard Backlight[0x0007])
0xA1, 0x01,                    // Collection(Application)
0x85, 0x01,                    //     ReportId(1)
0x0A, 0x15, 0x05,              //     UsageId(Keyboard Brightness Next[0x0515])
0x0A, 0x16, 0x05,              //     UsageId(Keyboard Brightness Previous[0x0516])
0x15, 0x00,                    //     LogicalMinimum(0)
0x25, 0x01,                    //     LogicalMaximum(1)
0x95, 0x02,                    //     ReportCount(2)
0x75, 0x01,                    //     ReportSize(1)
0x81, 0x06,                    //     Input(Data, Variable, Relative, NoWrap, Linear, PreferredState,
NoNullPosition, BitField)
0x95, 0x01,                    //     ReportCount(1)
0x75, 0x06,                    //     ReportSize(6)
0x81, 0x03,                    //     Input(Constant, Variable, Absolute, NoWrap, Linear, PreferredState,
NoNullPosition, BitField)
0x0A, 0x17, 0x05,              //     UsageId(Keyboard Backlight Level Suggestion[0x0517])
0x67, 0xE1, 0x00, 0x00, 0x01,  //     Unit('nits', SiLinear, Centimeter:-2, Candela:1)
0x55, 0x04,                    //     UnitExponent(10,000)
0x26, 0xFF, 0x00,              //     LogicalMaximum(255)
0x95, 0x03,                    //     ReportCount(3)
0x75, 0x08,                    //     ReportSize(8)
0xB1, 0x03,                    //     Feature(Constant, Variable, Absolute, NoWrap, Linear, PreferredState,
NoNullPosition, NonVolatile, BitField)
0x09, 0x7B,                    //     UsageId(Keyboard Backlight Set Level[0x007B])
0x95, 0x01,                    //     ReportCount(1)
0x91, 0x02,                    //     Output(Data, Variable, Absolute, NoWrap, Linear, PreferredState,
NoNullPosition, NonVolatile, BitField)
0xC0,                          // EndCollection()
```