MOTOROLA
intelligence everywhere™

Vibe and Backlight API & FunLight API on Linux OS based mobile phones

October 31, 2005

WHITE PAPER

Vibe and Backlight API & FunLight API on Linux OS based mobile phones

By

Motocoder staff

1. Which Phone can support these API

This device matrix shows the existence of the Vibe and Backlight API & FunLight API on the Motorola Linux based phone.

	Vibe and Backlight API	FunLight API
A760/A768	NO	NO
E680/E680I	Support	Support
A780/A728	Support	NO

2.Vibe and Backlight API

The Vibe and Backlight API allows J2ME applications access to vibrator, backlight. This API gives a MIDlet the ability to turn these features on or off based on the applications needs. The MIDlet will use this API to enhance the activity being performed by the application.

Examples of this enhancement are the following:

a. When in a driving game application, the vibrator is turned on during a crash.

b. An alarm application would have access to turn the vibrator on and off.

c. A stock ticker application turns the backlight on and off when a specified stock hits a target price.

The following are code samples to show implementation of the Vibe and Backlight API:

Sample of code for calling of 'vibrate(int)' method of Display class:

```
int duration = 5000;//5 seconds;

boolean returnValue = display.vibrate(duration);


if (returnValue != false) {
        System.out.println("Invoke vibrate method with parameter = "+ duration + ", method
        returns : " + returnValue);
}
else {
        System.out.println("Failed: invoke vibrate(" + duration +"), method returns false");
```

}

The backlight on the phone can be turned on or off for the purpose of drawing the user's attention or providing special effects to enhance the user’s gaming experience. Backlight control on the E680/E680I/A780/A728 is achieved through the Display class' flashBacklight method as defined in the MIDP2.0 profile.

The flashBacklight method's declaration is below:

public boolean **flashBacklight**(int duration)

The duration parameter specifies the period of time in milliseconds that the backlight should be switched on. To specify that the backlight be switched off, the value of duration should be set to 0. An IllegalArgumentException is thrown if a negative value is passed in for the duration. The flashBacklight method returns either true or false depending on whether the backlight can be controlled i.e. for devices that can not control the backlight, this method returns false.

Sample of code for calling of ‘flashBacklight(int)’ method of Display class:

```
 int duration = 30000;// 30 seconds;
 boolean returnValue = display.flashBacklight(duration);
if (returnValue != false) {
    System.out.println("Invoke flashBacklight method with parameter = " + duration + ",
    method returns : " + returnValue);
}
else {
    System.out.println("Failed: invoke flashBacklight(" +
    duration + "), method returns false");
}
```

3. Fun Lights API

The Fun Lights API allows J2ME applications access to the Fun Lights feature on the Motorola E680/E680i. The Fun Lights feature gives a MIDlet the ability to turn the Fun Lights

feature on or off based on the applications needs. The MIDlet would use this API to enhance the activity being performed by the application.

Examples of this enhancement would be the following:

a. When in a driving game application, the Fun Lights turn on at varying light intensities and colors to warn of an impending crash.

b. A karaoke or jukebox application turns the Fun Lights on and off to the beat of the music.

c. An alarm clock or stopwatch application uses the Fun Lights in conjunction with a timer feature.

Fun Lights Regions

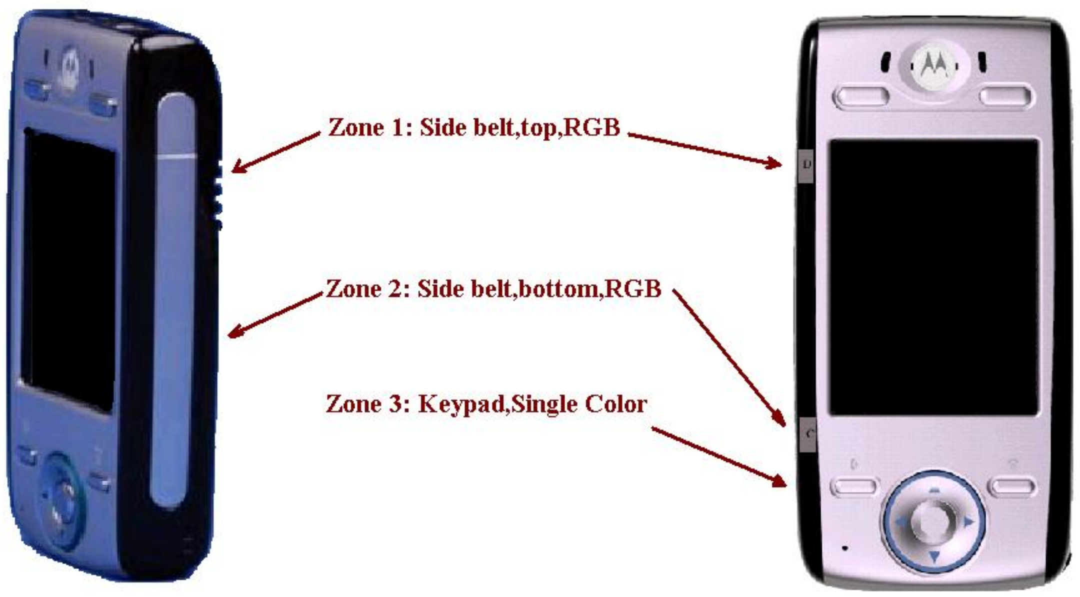
The Motorola E680/E680i has areas that light up or change colors based on an assigned region number. By assigning priority ratings to the regions, all products have the ability to use

Fun Lights. Querying the number of regions a handset has allows for Fun Light patterns to be mapped to a specific region. For example,

• A MIDlet will query the handset's native software to determine the number of light

regions and from there, can map Fun Lights features to each of the regions.

The image below is a graphic representation of Fun Lights on the E680:


Zone 1: Side belt,top,RGB
Zone 2: Side belt,bottom,RGB
Zone 3: Keypad,Single Color

The regions for the Motorola E680/E680i are outlined below:

Region	Description
1	Side belt,top RGB
2	Side belt ,bottom,RGB
3	Keypad,Single Color

The Fun Lights features that J2ME MIDlets access will include individual LED control – color changes of LEDs (seven colors) and light intensity of LEDs (two levels: ON/OFF).

Motorola will create the color table for the 8 colors (including BLACK) supported. If a color in a Fun Light feature is not supported by the handset, the closest color that matches the specified color shall be used.

Each of these LEDs can be set to on or off and due to human factor concerns the refresh rate will be less than 5 Hz. If the rate is set to more than 5 Hz, setColor () operation will be ignored.

The following are code samples to show implementation of the Fun Lights API:

Sample of operating regions one by one

```
/*
* Obtain the array of regions
*/
Region [] regions = FunLight.getRegions();

/*
* Set MAGENTA color for all regions Fun Lights API
*/

for(int i = 0; i < regions.length; i++) {
/*
* Obtain control of i's region
*/
int oc_result = regions[i].getControl();
/*
* Make sure control was obtained
*/
if(oc_result != FunLight.SUCCESS) {
/*
* Control was not obtained
*/
if(oc_result == FunLight.QUEUED) {
/*
* The request of obtaining region's control
was queued
* The control will be obtained when it
released by other application
```

```
*/
...
}
if(oc_result == FunLight.IGNORED) {
/*
* The request of obtaining region's control
was ignored
* It is related to 5Hz limit
*/
...
}
}
/*
* Set MAGENTA color for i's region
*/
int sc_result = regions[i].setColor(FunLight.MAGENTA);
/*
* Make sure the color was set
*/
if(sc_result != FunLight.SUCCESS) {
/*
* The color was not set
*/
if(sc_result == FunLight.QUEUED) {
/*
* The request of setting region's color wasqueued
* (it means the MIDlet does not have control of the region)
* The color will be set when the control of the regions will *be obtained
*/
...
}
if(sc_result == FunLight.IGNORED) {
/*
* The request of setting region's color was ignored
* It is connected with 5Hz limit
*/
...
}
}
/*
* Obtain the color of the region
*/
int color = regions[i].getColor();
/*
* Check the color of the region
*/
if(color != FunLight.MAGENTA) {
/*
* Maybe it's a region which supports only white color
*/
if(color != FunLight.WHITE) {
/*
* OK. We got a region which supports only white color
*/
...
} else {
/*
* Error situation: the color differs from MAGENTA and *WHITE
*/
...
}
}
/*
* Release the control of i's region
*/
regions[i].releaseControl();
}
```

Sample of operating regions all at once

```
/*
* Obtain control of all regions Fun Lights API
*/
int oc_result = FunLight.getControl();
/*
* Make sure the control of all regions was obtained
*/
if(oc_result != FunLight.SUCCESS) {
/*
* The control of at least one region was not obtained
*/
if(oc_result == FunLight.QUEUED) {
/*
* The request of obtaining region's control was queued for * at least one region
*/
...
}
if(oc_result == FunLight.IGNORED) {
/*
* The request of obtaining region's control was ignored for * at least one region
*/
...
}
}
/*
* Set color for all regions
*/
int sc_result = FunLight.setColor(FunLight.MAGENTA);
/*
* Make sure the color was set for all regions
*/
if(sc_result != FunLight.SUCCESS) {
/*
* The color of at least one region was not set
*/
if(sc_result == FunLight.QUEUED) {
/*
* The request of setting region's color was queued for at
* least one region
*/
...
}
if(sc_result == FunLight.IGNORED) {
/*
* The request of setting region's color was ignored for at
* least one region
*/
...
}
}
/*
* Release control of all regions
*/
FunLight.releaseControl();
```

4. Samples in SDK

The CoolFLTest on Motorola\SDK v5.2.1 for J2ME\demo\com\mot\j2me\midlets\CoolFLTest directory is a samples to show how to use Fun Lights API. You can run this samples on E680.

Reference:

1. Motorola E680 Handset J2ME™ Developer