

How_to_create_a_color_fading_text_in_Java_ME

We'll implement a simple class to draw **text that changes multiple colors with a fading effect**. You can view it in action [here](#).



So, let's start defining our **ColorFadeText** class.

Source code: ColorFadeText class

We start defining the necessary **properties**:

```
//will hold the colors to fade
int[] colors = null;

//duration of a single fade
int fadeDuration = 0;

//effect start time
long startTime = 0;

//property used to check if effect has started
public boolean started = false;

//the text to be drawn
String text = null;
```

Now, we define the main **constructor**, that will accept the following arguments:

- the **text** to be drawn
- an **int[]** array containing the **colors** to fade into
- the **duration** of a single fade

```
public ColorFadeText(String text, int[] colors, int fadeDuration)
{
    if(colors.length == 0)
    {
        throw new IllegalArgumentException("You must define at least 1 color");
    }
    this.text = text;
    this.colors = colors;
    this.fadeDuration = fadeDuration;
}
```

The effect **start()** method will simply set the **startTime** value to current time and the **started** property to true

```
public void start()
{
    startstartTime = System.currentTimeMillis();

    started = true;
}
```

How_to_create_a_color_fading_text_in_Java_ME

Then, we have the **paint()** method, that will be used to paint the text on the given Graphics instance:

```
public void paint(Graphics g, int x, int y, int anchor)
{
    if(started)
    {
        long diff = System.currentTimeMillis() - startTime;

        int module = (int)(diff % fadeDuration);

        int colorIndex = (int)(diff / fadeDuration) % colors.length;

        int midColor = midColor(
            [(colorIndex + 1) % colors.length],
            [colorIndex], colors
            module,
            fadeDuration
        );

        setColor(midColor);
    }
    else
    {
        setColor(colors[0]);
    }

    drawString(text, x, y, anchor);
}
```

The following utility method will be the one actually used to get the current text color:

```
static int midColor(int color1, int color2, int prop, int max)
{
    int red =
        (((color1 >> 16) & 0xff) * prop +
        ((color2 >> 16) & 0xff) * (max - prop)) / max;

    int green =
        (((color1 >> 8) & 0xff) * prop +
        ((color2 >> 8) & 0xff) * (max - prop)) / max;

    int blue =
        (((color1 >> 0) & 0xff) * prop +
        ((color2 >> 0) & 0xff) * (max - prop)) / max;

    int color = red << 16 | green << 8 | blue;

    return color;
}
```

How to use it?

Using the **ColorFadeText** object is quite simple, since it'll be very similar to use a common String. Just follow this plain steps.

1. Create a ColorFadeText instance

How_to_create_a_color_fading_text_in_Java_ME

```
ColorFadeText text = new ColorFadeText (
    "I'M A FADING TEXT!",
    new int[] {0xff0000, 0x00ff00, 0x0000ff, 0xff00ff},
    1000
);
```

2. Start it...

```
text.start();
```

3. Paint it, using the same arguments used by Graphics drawString() method (adding a reference to the Graphics instance):

```
text.paint (g,
    getWidth
    2,
    Graphics.HCENTER | Graphics.TOP
);
```

Just a side note: since it's an animated effect, you'll need to **repaint it quite frequently**, so, for example, you can use a Thread to periodically call Canvas repaint() method.

Source code download

You can download **source code** of ColorFadeText, together with a sample Canvas that makes use of it, here: [Color Fading sources](#)