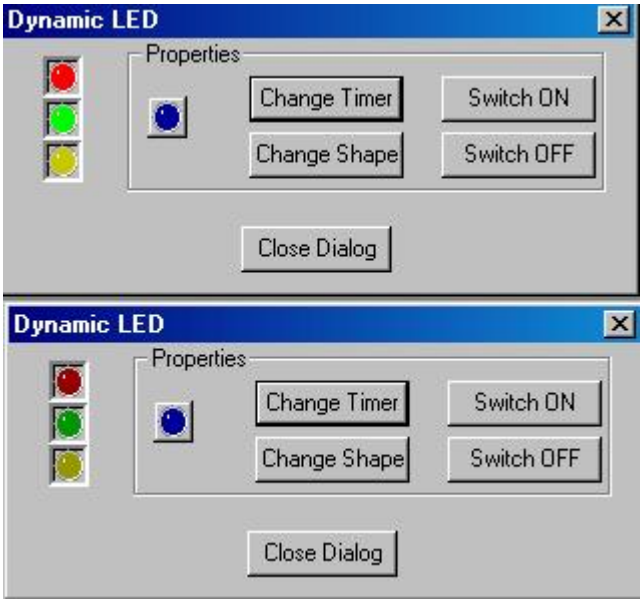
Dynamic LED Control

VGirish
30 Jul 2002

A blinking LED-style control

Download source files - 3 Kb

Download demo project - 28 Kb



Introduction

This class enables the user to have a blinking effect with a control over the blinking rate per second. They can set the timer of the control. Sometimes, it would be better to show the status of an ongoing operation or if there is any warning , it would be better to display it using an LED control. I faced this situation when developing for a client whose end users were security guards who had no knowledge of computers. So, the interface for them was designed like a TV remote control and I had to incorporate a lot of controls like this in order to make them feel at ease. I would like to tell the people who are going to use this class that its best suited for small controls rather than making it bigger where the effect is lost. It looks more like an ellipse than a LED at bigger proportions.

Implementation

To use the `CDynamicLED` class , you can follow the guidelines stated below.

1. Insert a new static frame where you want the LED. Setting the client edge property or the modal frame property for it looks better.

2. Rename the static frame to something other than `IDC_STATIC`. Name it something like `IDC_DYN_LED`.
3. Using the MFC ClassWizard, add a new member variable for `IDC__DYN_LED`. The category should be a control and the Variable Type should be `CDynamicLED`. If the `CDynamicLED` type does not show up in the Variable type dropdown combo, then you need to recreate the .clw file. Delete the .clw file and run the class wizard again.
4. Remember to add "*DynamicLED.h*" in your dialog's header file.

Operations

The various features in the `CDynamicLED` Class are outlined below.

1. `SetLED(CWnd *pWnd, UINT nIDColor, UINT nIDShape, int nTimerInterval)`

   where pWnd is your static window where you want the LED, nIDColor can be any of the following values

```
ID_LED_RED  <BR>
    ID_LED_GREEN    <BR>
    ID_LED_BLUE     <BR>
    ID_LED_YELLOW   <BR>
```

   NIDShape can be any of the following values

```
ID_SHAPE_ROUND<BR>
    ID_SHAPE_SQUARE<BR>
```

   Here, the `nIDShape` value determines whether the shape of the LED would be round or square.

   And the `nTimerInterval` parameter denotes the number of milliseconds. The LED would flash once in every period denoted by this parameter. You can either have a rapidly blinking LED by setting this parameter to 100 or have a normally blinking LED which blinks once per second by setting this value to 1000.

   This is the **only** function that you need to know to use this class.

2. In case you need more functionality to switch on or switch off the led , you have 2 functions named

   `SwitchOn` and `SwitchOff`

   These 2 functions don't need any parameters.

Now let's go to the implementation of this control in your dialog based application.

I have assumed that you have named your dialog class as `CMyDialog`

Remember that you have created a variable for this static frame. If you have forgotten about that, please refer to the implementation section above. Assuming that you have named your variable as `m_dynLEDRed` for a LED control which is round in shape, going to blink once every half a second and which is red in colour.

You have to add the following lines in your `OnInitDialog` function. I have also assumed that you have named your static frame `IDC_STATIC_LED_RED`.

```
CWnd *pWndRed = (CWnd *)GetDlgItem(IDC_STATIC_LED_RED);
m_dynLEDRed.SetLED(pWndRed,ID_LED_RED,ID_SHAPE_ROUND,500);
```

Incase I want to change the blinking interval of the LED at runtime from half a second to a full second, then you can use the following code.

```
// Change the time interval of the LED to 1000
CWnd *pWndRed = (CWnd *)GetDlgItem(IDC_STATIC_LED_RED);
m_dynLEDRed.SetLED(pWndBlue,ID_LED_BLUE, ID_SHAPE_ROUND,1000);
```

To change the shape of the LED from round to square or vice versa , you can follow this piece of code.

```
// Change the shape of the Blue LED from round to square
CWnd *pWndRed  = (CWnd *)GetDlgItem(IDC_STATIC_LED_RED);
m_dynLEDRed.SetLED(pWndBlue,ID_LED_BLUE, ID_SHAPE_SQUARE,1000);
```

If you want to turn off the LED (I mean switching it off) , you can use this .

```
// Switch OFF the Red LED
CWnd *pWndRed = (CWnd *)GetDlgItem(IDC_STATIC_LED_RED);
m_dynLEDRed.SwitchOff();
```

and to switch it on again, use

```
// Switch ON the Red LED
CWnd *pWndRed = (CWnd *)GetDlgItem(IDC_STATIC_LED_RED);
m_dynLEDRed.SwitchOn();
```

Thats all folks. All luck and have a great time.

With warm regards,
V.Girish

License

About the Author

VGirish No Biography provided
Founder
India

Comments and Discussions

First Prev Next

Rewritten...
Peter B. 28-Jun-05 6:08

Yes, this code may be taken somewhere.
And there are memory leaks, surely!

However, the result seems fine, so I worked with this
class for 2 hours and changed it completely.

I will not post an article, but here is my result:
(methods are more logical, no more memory leaks,...)

DynamicLED.h:

```
#if !defined(AFX_DYNAMICLED_H__7AA00BEC_B6E4_48A7_9719_3A15C0AB217A__INCLUDED_)
#define AFX_DYNAMICLED_H__7AA00BEC_B6E4_48A7_9719_3A15C0AB217A__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// DynamicLED.h : header file
//
#include "stdafx.h"

#define ID_SHAPE_ROUND 3001
#define ID_SHAPE_SQUARE 3002

/////////////////////////////////////////////////////////////////////////////
// CDynamicLED window
class CDynamicLED : public CStatic
{
// Construction
public:
CDynamicLED();
void SetColor(COLORREF on,COLORREF off = RGB(0,0,0));
void SetBlink(int iTime_in_ms); // 0 means: blink off
void SetOnOff(bool State);
virtual ~CDynamicLED();
void SetShape(int iShape);

private:

// The pens and brushes needed to do the drawing
CPen m_PenBright,m_PenDark;
CBrush m_BrushBright,m_BrushDark,m_BrushCurrent;

// This variable is used to store the shape and color
```

```
// set by the user for resetting the led later
UINT m_nShape;
BOOL m_bBright;

// Operations
public:

// Overrides
// ClassWizard generated virtual function overrides
//{{AFX_VIRTUAL(CDynamicLED)
//}}AFX_VIRTUAL



private:
unsigned int m_TimerHandle;
COLORREF m_OnColor;
COLORREF m_OffColor;
// Generated message map functions
protected:
//{{AFX_MSG(CDynamicLED)
afx_msg void OnPaint();
afx_msg void OnTimer(UINT nIDEvent);
//}}AFX_MSG

DECLARE_MESSAGE_MAP()
};

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.

#endif // !defined(AFX_DYNAMICLED_H__7AA00BEC_B6E4_48A7_9719_3A15C0AB217A__INCLUDED_)
```

DynamicLed.cpp:

```
// DynamicLED.cpp : implementation file
//

#include "stdafx.h"
#include "DynamicLED.h"


#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
```

```
// CDynamicLED

CDynamicLED::CDynamicLED()
{
// Initialize the variables
m_bBright = FALSE;
m_nShape = ID_SHAPE_ROUND;
m_TimerHandle=0;
}

CDynamicLED::~CDynamicLED()
{
}

BEGIN_MESSAGE_MAP(CDynamicLED, CStatic)
//{{AFX_MSG_MAP(CDynamicLED)
ON_WM_PAINT()
ON_WM_TIMER()
//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CDynamicLED message handlers


void CDynamicLED::OnPaint()
{
CPaintDC dc(this); // device context for painting
// If the timer value is zero, we dont want to do anything
// It means that the LED is in a switched off state. So just return

// Get the Device Context
// CClientDC dc(this);

// Get the rectangle of the window where we are going to draw
CRect rcClient;
GetClientRect(&rcClient);

// If the pen has been selected already, then we have to delete it
// so that it doesnt throw an assertion

if(m_PenBright.m_hObject!=NULL)
m_PenBright.DeleteObject();

if(m_BrushBright.m_hObject!=NULL)
m_BrushBright.DeleteObject();

if(m_PenDark.m_hObject!=NULL)
m_PenDark.DeleteObject();

if(m_BrushDark.m_hObject!=NULL)
m_BrushDark.DeleteObject();


m_PenBright.CreatePen(0,1,m_OnColor);
m_BrushBright.CreateSolidBrush(m_OnColor);
```

```
m_PenDark.CreatePen(0,1,m_OffColor);
m_BrushDark.CreateSolidBrush(m_OffColor);



if(m_bBright==TRUE)
{
// If we have to switch on the LED to display the bright colour select
// the bright pen and brush that we have created above

dc.SelectObject(&m_PenBright);
dc.SelectObject(&m_BrushBright);

m_BrushCurrent.m_hObject = m_BrushBright.m_hObject;

// m_bBright = FALSE;
}
else
{
// If we have to switch off the LED to display the dark colour select
// the bright pen and brush that we have created above

dc.SelectObject(&m_PenDark);
dc.SelectObject(&m_BrushDark);

m_BrushCurrent.m_hObject = m_BrushDark.m_hObject;

// m_bBright = TRUE;
}

// If the round shape has been selected for the control
if(m_nShape==ID_SHAPE_ROUND)
{
// Draw the actual colour of the LED
dc.Ellipse(rcClient);

// Draw a thick dark gray coloured circle
CPen Pen;
Pen.CreatePen(0,2,RGB(128,128,128));
dc.SelectObject(&Pen);
dc.Ellipse(rcClient);

// Draw a thin light gray coloured circle
CPen Pen2;
Pen2.CreatePen(0,1,RGB(192,192,192));
dc.SelectObject(&Pen2);
dc.Ellipse(rcClient);

// Draw a white arc at the bottom
CPen Pen3;
Pen3.CreatePen(0,1,RGB(255,255,255));
dc.SelectObject(&Pen3);

// The arc function is just to add a 3D effect for the control
CPoint ptStart,ptEnd;
ptStart = CPoint(rcClient.Width()/2,rcClient.bottom);
ptEnd = CPoint(rcClient.right,rcClient.top);
```

```
dc.MoveTo(ptStart);
dc.Arc(rcClient,ptStart,ptEnd);

CBrush Brush;
Brush.CreateSolidBrush(RGB(255,255,255));
dc.SelectObject(&Brush);

// Draw the actual ellipse
dc.Ellipse(rcClient.left+4,rcClient.top+4,rcClient.left+6,rcClient.top+6);
}
else if(m_nShape==ID_SHAPE_SQUARE)
{
// If you have decided that your LED is going to look square in shape, then

// Draw the actual rectangle
dc.FillRect(rcClient,&m_BrushCurrent);

// The code below gives a 3D look to the control. It does nothing more

// Draw the dark gray lines
CPen Pen;
Pen.CreatePen(0,1,RGB(128,128,128));
dc.SelectObject(&Pen);

dc.MoveTo(rcClient.left,rcClient.bottom);
dc.LineTo(rcClient.left,rcClient.top);
dc.LineTo(rcClient.right,rcClient.top);


// Draw the light gray lines
CPen Pen2;
Pen2.CreatePen(0,1,RGB(192,192,192));
dc.SelectObject(&Pen2);

dc.MoveTo(rcClient.right,rcClient.top);
dc.LineTo(rcClient.right,rcClient.bottom);
dc.LineTo(rcClient.left,rcClient.bottom);
}

}

void CDynamicLED::OnTimer(UINT nIDEvent)
{
// if(m_nTimerInterval==0)
// return;

if(m_bBright==TRUE)
{

m_bBright = FALSE;
}
else
{
m_bBright = TRUE;
}
RedrawWindow();
```

```
CStatic::OnTimer(nIDEvent);
}



void CDynamicLED::SetOnOff(bool State)
{
m_bBright = State;
RedrawWindow();
}

void CDynamicLED::SetBlink(int iTime_in_ms)
{

if(m_TimerHandle)
{
::KillTimer(this->m_hWnd,m_TimerHandle);
m_TimerHandle=0;
}

if(iTime_in_ms > 0)
m_TimerHandle = ::SetTimer(this->m_hWnd,1001,iTime_in_ms,NULL);

}

void CDynamicLED::SetColor(COLORREF on, COLORREF off)
{
m_OnColor = on;
m_OffColor = off;

}

void CDynamicLED::SetShape(int iShape)
{
m_nShape=iShape;
}
```

Re: Rewritten...
exJeff 2-May-07 12:48

anyway,thanks for you working

good idea!

Re: Rewritten...
BernardOfNewport 7-Jun-16 12:48

Thanks for finding the time to post your work. Needless to say yours works very well.

COPYING FROM OTHER'S ARTICLE ALSO A ART.RIGHT Mr.GIRISH?
Anonynmous 8-Dec-03 17:36

Hello!
This is not to discourage you ,but whatever you are writing ARTICLES all are just COPY CAT; copying from other's Articles and giving a small modification and post here . Do you like to take credit out of this.? Don't you have even small courtesey to mention of your thanks to where u copied from??? Is it your habbit?

Don't copy from others.. if you are really of some use.., post your own created articles.

Just bukllshit articles. Here is the original articlies http://codeguru.earthweb.com/controls/StaticLed.shtml for this Led control.

Get rid of from this and don't try to get fame out of this techniques.

Henery

COPYING FROM OTHER'S ARTICLE ALSO A ART.RIGHT Mr.GIRISH?
Anonymous 8-Dec-03 17:34

Hello!
This is not to discourage you ,but whatever you are writing ARTICLES all are just COPY CAT; copying from other's Articles and giving a small modification and post here . Do you like to take credit out of this.? Don't you have even small courtesey to mention of your thanks to where u copied from??? Is it your habbit?

Don't copy from others.. if you are really of some use.., post your own created articles.

Just bukllshit articles. Here is the original articlies http://codeguru.earthweb.com/controls/StaticLed.shtml for this Led control.

Get rid of from this and don't try to get fame out of this techniques.

Henery

Re: COPYING FROM OTHER'S ARTICLE ALSO A ART.RIGHT Mr.GIRISH?
Luca Crisi, MCP 22-Jun-09 19:09

What ever you migh think about it, this code ADDS a feature to the original one: it BLINKS the LED.

So, it's not just copy & paste, some work has been done.

Luca Crisi, MCP

Memory Leak
Anonymous 19-Sep-02 22:50

There appears to be a memory leak, 4k, about every other second, under Win2k.

Run the demo project then bring up windows task manager, switch to processes tab, look for DynhLED.exe and you'll notice mem usage increases 4k every other second.

Re: Memory Leak

DarrollWalsh 15-Feb-03 23:24

Has you figured out where it is at?

Darroll

Not one person lives in the present. Only the past. I can prove it.

Re: Memory Leak
Ton_B 25-Apr-05 21:45

It is probably a resource leak. Pens and brushes should not be deleted while they are selected in a DC. Given the other remarks here, this code should be reviewed and re-submitted!

Invisible control
11-Aug-02 23:51

This is an interesting class. The only problem I have is that you have to set the LED to blink first before it ever gets drawn. I need this to be drawn on creation. Visible but in the off position. To save myself time I would like to ask how this can be done with out researching the class. Also How do I get this to not blink at all? to be steady? I have 3 LEDs and want them to come on in sucesion of each other given some info ect. I hope you can help.

Darroll

Not one person lives in the present. Only the past. I can prove it.

Re: Invisible control
PJ Arends 12-Aug-02 1:12

The problem is that the control does all it's drawing in it's `OnTimer()` function. To fix it, move the drawing code from `OnTimer()` to `OnPaint(),` and in `OnTimer()` simply place a call to `RedrawWindow()`.

CPUA 0x5041

Sonork 100.11743 Chicken Little

"So it can now be written in stone as a testament to humanities achievments **"PJ did Pi at CP"**." Colin Davies

Within you lies the power for good - Use it!

Re: Invisible control
13-Aug-02 1:33

Thanks worked like a charm, just make sure to use the CPaint dc(this) in the on paint or it won't work correctly. I also added a function called Lit passing it a bool so that I can just turn the LED on and off as I like. lit looks like this.

```
void CDynamicLED::Lit(bool State)
{
m_bBright = State;
```

```
RedrawWindow();
}
```

Thanks for your help.

Darroll

Not one person lives in the present. Only the past. I can prove it.

Refresh 1

General News Suggestion Question Bug Answer Joke Praise Rant Admin

12 messages have been posted for this article Visit **https://www.codeproject.com/Articles/2654/Dynamic-LED-Control** to post and view comments on this article, or click **here** to get a print view with messages.