# How do I get a radio button control to render its text transparently?

devblogs.microsoft.com/oldnewthing/20100628-00

June 28, 2010

Raymond Chen

Commenter Andrei asks via the Suggestion Box for help with making the text transparent using `WM_CTLCOLORSTATIC`. “Instead of the radio button now there’s a black background.”

Let’s look at this problem in stages. First, let’s ignore the transparent part and figure out how to render text without a black background. The background color of the text comes from the color you selected into the DC when handling the `WM_CTLCOLORSTATIC` message. And if you forget to set a background color, then you get whatever color is lying around in the DC, which might very well be black. Start with the scratch program and make these changes, which I’m going to write in the way I think Andrei wrote it, even though it doesn’t fit the style of the rest of the scratch program.

```
HBRUSH g_hbr;
BOOL
OnCreate(HWND hwnd, LPCREATESTRUCT lpcs)
{
    g_hwndChild = CreateWindow(TEXT("button"), TEXT("Bingo"),
        WS_CHILD | WS_VISIBLE | BS_RADIOBUTTON,
        0, 0, 0, 0, hwnd, (HMENU)1, g_hinst, 0);
    g_hbr = CreateSolidBrush(RGB(0xFF, 0x00, 0xFF)); // hot pink
    return TRUE;
}
void
OnDestroy(HWND hwnd)
{
    if (g_hbr) DeleteObject(g_hbr);
    PostQuitMessage(0);
}
// add to WndProc
  case WM_CTLCOLORSTATIC:
    if (GetDlgCtrlID(
             GET_WM_CTLCOLOR_HWND(wParam, lParam, uiMsg)) == 1) {
      return (LRESULT)g_hbr; // override default background color
    }
    break;
```

If you run this program, the radio button's background is indeed hot pink, well except for the text, where the color is, I dunno, it's white on my machine, but who knows what it is on yours. Since we didn't specify a color, the result is undefined. The bug here is that we handled the `WM_CTLCOLORSTATIC` message incompletely. The `WM_CTLCOLOR` family of messages requires that the message handler do three things:

1. Set the DC text color.
2. Set the DC background color.
3. Return a background brush.

We got so excited about the background brush that we forgot the other two steps. Let's fix that.

```
case WM_CTLCOLORSTATIC:
    if (GetDlgCtrlID(
             GET_WM_CTLCOLOR_HWND(wParam, lParam, uiMsg)) == 1) {
      HDC hdc = GET_WM_CTLCOLOR_HDC(wParam, lParam, uiMsg);
      SetTextColor(hdc, RGB(0xFF, 0xFF, 0x00)); // yellow
      SetBkColor(hdc, RGB(0xFF, 0x00, 0xFF)); // hot pink
      return (LRESULT)g_hbr; // override default background color
    }
    break;
```

(Just for fun, I chose yellow as the text color.) Now that we specified the text color and the background color, the text appears in the correct colors.

Note that we didn't actually do anything transparently here. We just made sure that the background color we told the control to use for text matches the color we told the control to use for erasing the background. The effect looks transparent since the two colors match.

But what if you really wanted transparency instead of fake transparency? To illustrate, let's give the control a background that is not a solid color:

```
BOOL
OnCreate(HWND hwnd, LPCREATESTRUCT lpcs)
{
    g_hwndChild = CreateWindow(TEXT("button"), TEXT("Bingo"),
        WS_CHILD | WS_VISIBLE | BS_RADIOBUTTON,
        0, 0, 0, 0, hwnd, (HMENU)1, g_hinst, 0);
    g_hbr = CreatePatternBrushFromFile(
                          TEXT("C:\\Windows\\Gone Fishing.bmp"));
    return TRUE;
}
```

When you run this version of the program, the radio button background consists of the Gone Fishing bitmap. (Of course, if you don't have that bitmap, then feel free to substitute another bitmap. I can't believe I had to write that.) But the text is still yellow on pink. How do we get it to be yellow on the complex background?

By setting the background mix mode to `TRANSPARENT`.

```
case WM_CTLCOLORSTATIC:
    if (GetDlgCtrlID(
             GET_WM_CTLCOLOR_HWND(wParam, lParam, uiMsg)) == 1) {
      HDC hdc = GET_WM_CTLCOLOR_HDC(wParam, lParam, uiMsg);
      SetTextColor(hdc, RGB(0xFF, 0xFF, 0x00)); // yellow
      SetBkColor(hdc, RGB(0xFF, 0x00, 0xFF)); // hot pink
      SetBkMode(hdc, TRANSPARENT);
      return (LRESULT)g_hbr; // override default background color
    }
    break;
```

According to the documentation, the background mix mode "is used with text, hatched brushes, and pen styles that are not solid lines." It's the text part we care about here. When the control does its `TextOut` to draw the control text, the background mix mode causes the text to be rendered transparently.

**Exercise**: There's actually one more thing you need to do, but I conveniently arranged the program so you didn't notice. What other step did I forget?