Using WM_SETREDRAW to speed up adding a lot of elements to a control

devblogs.microsoft.com/oldnewthing/20140407-00

April 7, 2014

Raymond Chen

Today's Little Program shows one way you can implement a better version of `WM_SETREDRAW`. Our first version doesn't use `WM_SETREDRAW` at all.

Start with the scratch program and make the following changes:

```
HFONT g_hfList;
int g_yOrigin;
int g_cyLine;
int g_cLinesPerPage;
int g_cItems = 100;
// GetTrackPos incorporated by reference
// ScrollTo incorporated by reference
// ScrollDelta incorporated by reference
// OnSize incorporated by reference
// OnVscroll incorporated by reference + modifications
// OnCreate incorporated by reference
// OnDestroy incorporated by reference
// This is the same as the earlier version of PaintSimpleContent
// with one tiny change: Draw the items in reverse order so the effect
// is more noticeable.
void
PaintSimpleContent(HWND hwnd, PAINTSTRUCT *pps)
{
 HFONT hfPrev = SelectFont(pps->hdc, g_hfList);  /* Use the right font */
 int iMin = max(pps->rcPaint.top / g_cyLine, 0);
 int iMax = min((pps->rcPaint.bottom + g_cyLine - 1) / g_cyLine, g_cItems);
 for (int i = iMin; i < iMax; i++) {
  char szLine[256];
  int cch = wsprintf(szLine, "This is line %d", g_cItems - i);
  TextOut(pps->hdc, 0, i * g_cyLine, szLine, cch);
 }
 SelectFont(pps->hdc, hfPrev);
}
// PaintContent incorporated by reference
void AddItem(HWND hwnd)
{
 g_cItems++;
 InvalidateRect(hwnd, 0, TRUE);
 ScrollDelta(hwnd, 0);
}
void OnChar(HWND hwnd, TCHAR ch, int cRepeat)
{
 switch (ch) {
 case TEXT('1'):
  AddItem(hwnd);
  break;
 case TEXT('2'):
  for (int i = 0; i < 10000; i++) {
   AddItem(hwnd);
  }
  break;
 case TEXT('3'):
  SetWindowRedraw(hwnd, FALSE);
  for (int i = 0; i < 10000; i++) {
   AddItem(hwnd);
  }
  SetWindowRedraw(hwnd, TRUE);
```

```
  InvalidateRect(hwnd, nullptr, TRUE);
 }
}
 HANDLE_MSG(hwnd, WM_VSCROLL, OnVscroll);
 HANDLE_MSG(hwnd, WM_CHAR, OnChar);
```

Most of this program was stolen from my scroll bar series. The interesting new bits are that you can add one new item by hitting `1`, or you can add ten thousand items by hitting `2`, or you can add ten thousand items with redraw disabled by hitting `3`.

I drew the items in reverse order so that adding an item forces everything to change position, so that the effect of the redraw is more noticeable.

Observe that adding one item is fast, but adding ten thousand items with redraw enabled is slow; you can watch the scroll bar as it slowly shrinks. But adding ten thousand items with redraw disabled is not too bad.

But we can do better.

```
BOOL g_fRedrawEnabled = TRUE;
void AddItem(HWND hwnd)
{
 g_cItems++;
 if (g_fRedrawEnabled) {
  InvalidateRect(hwnd, 0, TRUE);
  ScrollDelta(hwnd, 0);
 }
}
void OnSetRedraw(HWND hwnd, BOOL fRedraw)
{
 g_fRedrawEnabled = fRedraw;
 if (fRedraw) {
  InvalidateRect(hwnd, 0, TRUE);
  ScrollDelta(hwnd, 0);
 }
}
void
OnPaint(HWND hwnd)
{
 if (g_RedrawEnabled) {
  PAINTSTRUCT ps;
  BeginPaint(hwnd, &ps);
  PaintContent(hwnd, &ps);
  EndPaint(hwnd, &ps);
 } else {
  ValidateRect(hwnd, nullptr);
 }
}
 HANDLE_MSG(hwnd, WM_SETREDRAW, OnSetRedraw);
```

We have a custom handler for the `WM_SETREDRAW` message that updates a flag that indicates whether redraw is enabled. When adding an item, we do the visual recalculations (updating the scroll bar, mostly) only if redraw is enabled. If a paint message comes in while redraw is disabled, we merely validate the window to say "It's all good, don't worry!" When redraw is re-enabled, we ask for a fresh repaint and update the scroll bars.

With this version of the program, adding ten thousand items with redraw disabled is lightning fast.

Notice that `g_fRedrawEnabled` is not reference-counted. It's a simply `BOOL`. In other words, if you send the `WM_SETREDRAW` message twice to disable redraw, you still only need to enable it once. Disabling redraw on a window where redraw is already disabled has no effect.

Exercise: Compare the behavior of `WM_SETREDRAW` with (the incorrect) `LockWindowUpdate` for this program.

Raymond Chen

Follow