What is the inverse of AdjustWindowRect and AdjustWindowRectEx?

devblogs.microsoft.com/oldnewthing/20131017-00

October 17, 2013

Raymond Chen

We saw over a decade ago (my goodness I've been doing this way too long) that the `AdjustWindowRect` and `AdjustWindowRectEx` functions do not take menu wrapping into account because they don't take a window handle parameter, so they don't know what menu to test for wrapping. Still, they are useful functions if you aren't worried about menu wrapping because they let you do window size calculations without a window handle (say, before you create your window).

But those functions take a proposed client rectangle and return the corresponding non-client rectangle by inflating the rectangle by the appropriate borders, caption, scroll bars, and other non-client goo. But how do you go the other way? Say you have a proposed window rectangle and you want to know what client rectangle would result from it?

`AdjustWindowRect` and `AdjustWindowRectEx` can do that too. You just have to apply a negative sign.

The idea here is that we use the `AdjustWindowRectEx` function to calculate how much additional non-client area gets added due to the styles we passed. To make the math simple, we ask for a zero client rectangle, so that the resulting window is all non-client.

We pass in the empty rectangle represented by the dot in the middle, and the `AdjustWindowRectEx` expands the rectangle in all dimensions. We see that it added ten pixels to the left, right, and bottom, and it added fifty pixels to the top. (Numbers are for expository purposes. Actual numbers will vary.)

From this we can perform the reverse calculation: Instead of expanding the rectangle, we shrink it.

```
BOOL UnadjustWindowRectEx(
    LPRECT prc,
    DWORD dwStyle,
    BOOL fMenu,
    DWORD dwExStyle)
{
  RECT rc;
  SetRectEmpty(&rc);
  BOOL fRc = AdjustWindowRectEx(&rc, dwStyle, fMenu, dwExStyle);
  if (fRc) {
    prc->left -= rc.left;
    prc->top -= rc.top;
    prc->right -= rc.right;
    prc->bottom -= rc.bottom;
  }
  return fRc;
}
```

Note that the top and left are subtracted, so that the two negative signs cancel out.

Raymond Chen

Follow