How do I convert a WIC bitmap back to a GDI HBITMAP?

devblogs.microsoft.com/oldnewthing/20250528-00/?p=111225

May 28, 2025

The Windows Imaging Component (WIC) has a function for creating an `IWICBitmap` from a GDI `HBITMAP`. But how do you go the other way?

One way is to lock the bitmap bits and then use `SetDIBits` to place them into a DIB section. We saw this trick when we were debugging code that converts a Direct2D bitmap to a GDI `HBITMAP`. The downside of this is that it may result in a double-copy if the `IWICBitmapSource` has to generate the pixels on the fly (due to being the end of a chain of transforms).

Instead, we can ask `CopyPixels` to generate the pixels directly into our DIB section.

```
HRESULT ConvertWICBitmapSourceToHBITMAP(
    IWICBitmapSource* source,
    HBITMAP* bitmap)
{
    *bitmap = nullptr;

    // Convert to 32bpp if necessary.
    wil::com_ptr<IWICBitmapSource> source32bpp;
    RETURN_IF_FAILED(
        WICConvertBitmapSource(GUID_WICPixelFormat32bppBGR, source,
            source32bpp.put()));

    // Create a 32bpp DIB section of matching size
    UINT width, height;
    RETURN_IF_FAILED(source32bpp->GetSize(&width, &height));

    // Make sure the size won't create integer overflow.
    RETURN_HR_IF(HRESULT_FROM_WIN32(ERROR_ARITHMETIC_OVERFLOW),
                 width > (MAXDWORD / 4 / height));

    BITMAPINFO bi{};
    bi.bmiHeader.biSize = sizeof(BITMAPINFOHEADER);
    bi.bmiHeader.biWidth = width;
    bi.bmiHeader.biHeight = -static_cast<LONG>(height); // top-down
    bi.bmiHeader.biPlanes = 1;
    bi.bmiHeader.biBitCount = 32;
    bi.bmiHeader.biCompression = BI_RGB;
    bi.bmiHeader.biSizeImage = width * height * 4;

    void* bits;
    wil::unique_hbitmap result(CreateDIBSection(nullptr, &bi,
            DIB_RGB_COLORS, &bits, nullptr, 0));
    RETURN_LAST_ERROR_IF_NULL(result);

    // Copy the pixels into the bitmap
    RETURN_IF_FAILED(source32bpp->CopyPixels(nullptr, width * 4,
        bi.bmiHeader.biSizeImage, reinterpret_cast<BYTE*>(bits)));

    // All done
    *bitmap = result.release();
    return S_OK;
}
```

We convert the WIC bitmap to 32-bit BGR format, since that's the format that GDI uses for 32-bit DIBs. Then we create a 32-bit DIB section with the same dimensions as the WIC bitmap and use `CopyPixels` to tell the WIC bitmap to generate pixels into the DIB section buffer.

This code always creates a 32-bit DIB section. You could extend the function to query the WIC bitmap for its pixel format and create a matching GDI bitmap, but in practice almost nobody uses any GDI color formats besides 32-bit DIB and 32-bit DIB with premultiplied alpha. (32-bit DIB with straight alpha is also popular, but that's not a GDI color format.)