Why is there a limit of 15 shell icon overlays?

devblogs.microsoft.com/oldnewthing/20190313-00

March 13, 2019

Raymond Chen

There is a limit of fifteen shell icon overlays. Why fifteen?

The value 15 came from the corresponding limit for image lists. The `ImageList_SetOverlayImage` function supports up to 15 image list overlays per image list. (Hey, it used to be worse. The limit used to be only 3!)

Okay, but why only 15? Why not more?

The overlay image is one of the pieces of information used when drawing an image from an image list. The options are encoded in the `fStyle` parameter, and when the bits were divided up for various purposes, four bits were available to be used to specify the overlay image. (You get 15 overlay images instead of 16 because you lose one of the values in order to specify "no overlay.")

Okay, but the values in the `fStyle` parameter use only the bottom 16 bits. What about the upper 16 bits? There's plenty of room there.

The 16-bit limit was carried over from the 16-bit version of the common controls (which still needed to be supported in Windows 95). Of course, nowadays, nobody cares about the 16-bit version of the common controls, so why not start using the upper bits?

There's an unsatisfying explanation: The code internally that manages the `fStyle` still uses a `WORD` in some places, so all the code that manages the `fStyle` would have to be revised. This occurs in multiple modules across Windows, so a synchronized change would have to be made across multiple components. This is a breaking change at the binary level because the interfaces are no longer compatible. Breaking changes are procedurally difficult to coordinate: The affected code may not be visible to the shell team because they are sitting in a far-away leaf branch that has not yet RI'd to the trunk. It might be that expanding `fStyle` from a `WORD` to a `DWORD` has far-reaching consequences for some component.

Like I said, this is unsatisfying. Basically it boils down to "It would be a lot of work and we are lazy."

Another unsatisfying explanation is that the image list serialization format would be affected. This is one of the far-reaching consequences I hinted at: Changing a file format means that you also need to come up with a compatibility story. What happens if an old program encounters a file in the new format?

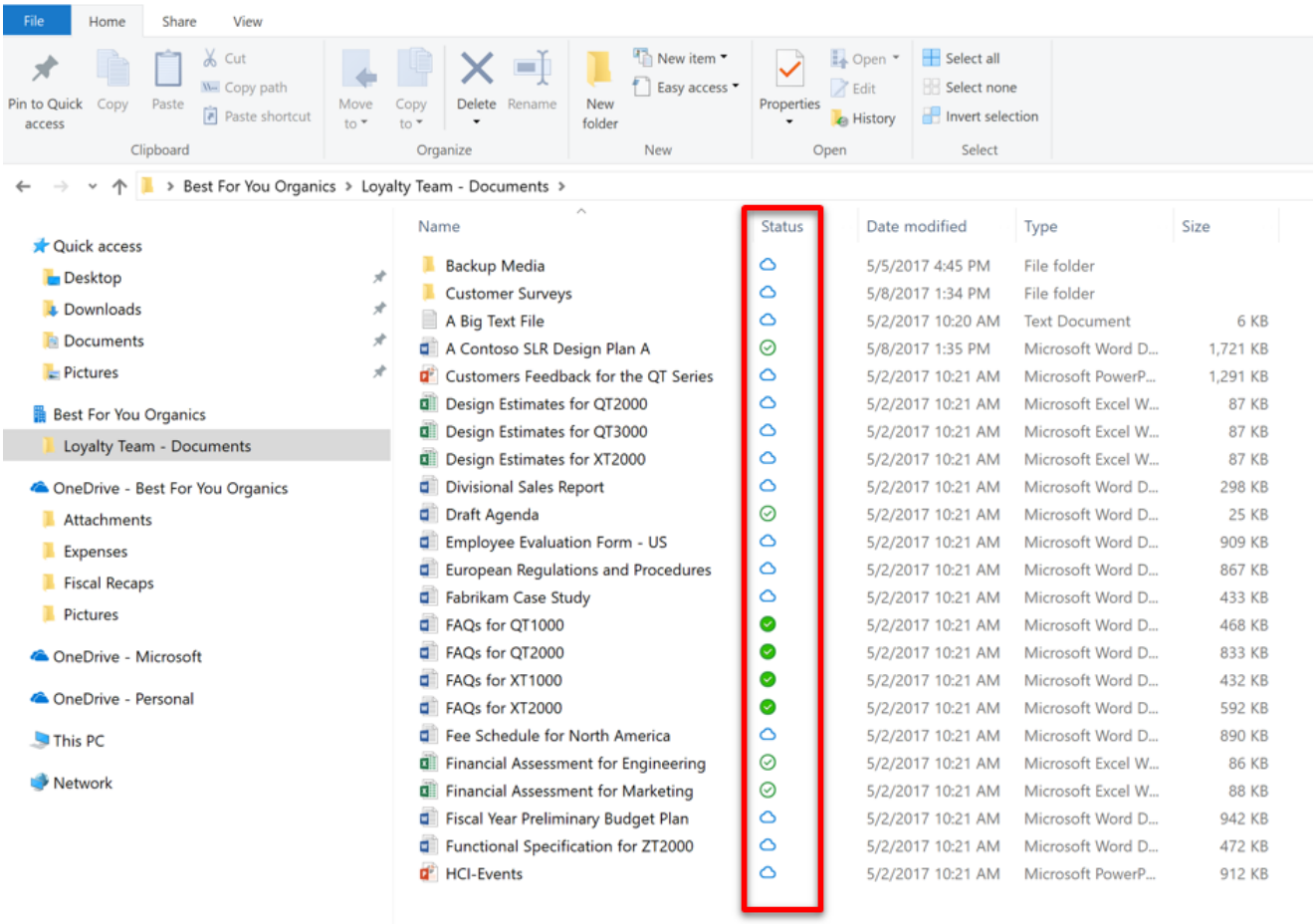
Okay, so maybe we leave image lists the way they are, but change Explorer so that it internally allows more than 15 overlays by, say, "paging" overlays in and out of the system image list. But this would require taking a lock during image list drawing operations, which currently can be performed concurrently. But the system image list is exposed to applications so that they can draw out of them directly, and applications won't know about the extra serialization that applies only to the system image list. So that's a non-starter, too.

A bigger problem is that the overlay indices are exposed to applications via the `SHGetFileInfo` function. If you ask for `SHGFI_OVERLAYINDEX`, then the function returns the overlay index to the application, and if the overlay index is greater than 15, older applications won't know how to draw with it.

Now, these problems aren't completely insurmountable. For example, we could invent a new flag to `SHGetFileInfo` that says, "I understand overlays greater than 15," and if the caller doesn't set the flag, then the function just pretends that those overlays don't exist. The calling application doesn't draw the icon correctly (it's missing the overlay), but at least it doesn't crash.

But the main reason this doesn't get done is simply that overlays are not recommended as the way to convey metadata about a file to the user. An icon can have only one overlay, which means that if multiple overlays apply, then the system will arbitrarily choose one to show, and the others will lose.

Windows 10 takes a step away from icon overlays by moving the OneDrive file synchronization status indicator from an icon overlay to a separate *Status* column.



Since it's a separate column, there's room to put more than one status icon there.

Raymond Chen

Follow