# Add Profiles To Jumplist

## Abstract

This spec outlines adding support for launching a profile from the jumplist. The main two points that will be detailed are:

1. Adding jumplist support
2. Adding the ability to open a profile from the jumplist

The profiles available in the jumplist will match the profiles available in the application from the + dropdown. The scope of the jumplist feature is limited to desktop devices.

## Inspiration

The inspiration is to be able to quickly open the terminal with one of the configured profiles.

## Terminology

- Jumplist: This is the menu that appears when the application is right-clicked on the taskbar or the start menu.

## Solution Design

### Overview

The jumplist has to be created/modified during the life-cycle of the application. The following is an outline of the steps to be performed (more details below)

1. Load settings / profiles
2. Create the jumplist with the initial profiles
3. Maintain sync between the jumplist and profiles throughout the life of the application

### Jumplist integration

UWP provides an API to access to the jumplist through the Windows.UI.StartScreen.JumpList class, however from previous attempts [1], the api does not work for the project architecture. Instead, we'll use the COM interface ICustomDestinationList [2] directly to create the jumplist. Since we are using Win32 apis, the work should be done in the `WindowsTerminal` project.

Using `ICustomDestinationList` is straightforward with a few additions discussed in this section below. Resources for using the jumplist can be found here and here.

The basic overview:

1. Get an instance of the `ICustomDestinationList` COM object
2. Call `ICustomDestinationList.BeginList`
3. Create an `IObjectCollection` COM object
4. Create `IShellLink` COM objects for each profile and add them to the `IObjectCollection`
5. Add the `IObjectCollection` to the `ICustomDestinationList`
6. Call `ICustomDestination.CommitList`

Note that the jumplist is replaced each time, instead of being updated.

**Approach**

One approach is to wrap the COM interfaces in a jumplist class and use that to add and remove items.

**What if there are multiple instances of the terminal application?**

There maybe multiple instances trying to update the jumplist at the same time.

## Adding profiles to the jumplist

To add the profiles to the jumplist there are some prerequisites that are needed

- Be able to access settings / profiles from the `WindowsTerminal` project.
- Detect when profiles are modified
  - Created: Add a new profile to the jumplist when a new profile is created by the user
  - Deleted: Remove the profile from the jumplist when the profile is removed by the user
  - Modified: We'll have to sync profile name changes / icon changes with the jump list

There are also different "categories" that we could add the profiles to. A jumplist can have items in a system category such as `Recent` or `Frequent` or a custom category. Jumplist items can also be a task. The difference between adding an item to a category or the task list is that items in a category can be `pinned` and `removed` from the jumplist by the user. Here is a summary of the two from the documentation [2]:

- `Destinations can be files, folders, websites, or other content-based items, but are not necessarily file-backed. Destinations can be thought of as things or nouns`
- `Tasks are common actions performed in an application that apply to all users of that application regardless of an individual's usage patterns. Tasks can be thought of as actions or verbs`

Following this, it would be more appropriate to add the profiles to the `Tasks` list as we are launching a profile. Essentially each item in the jumplist will be a shortcut that opens the terminal and provides command line arguments.

**Implementation notes**

When specifying the icon to use in the `IShellLink` object, it does not appear to be able to read `ms-appx://` URIs such as ones for the default profile icons and also only able to read `.ico` files. The way that the UWP api is able to do it is by adding additional `PropertyKey`s to the `IShellLink` object [3]. Using these tools JumpList and Lnk Explorer, we can examine what is different with UWP jumplists. When we examine the `.lnk` items that the jumplist uses, we can see that additional properties are added to the property store.

| Property Key (Format ID\Property ID) | Description | Example Value |
|---|---|---|
| {9F4C2855-9F79-4B39-A8D0-E1D42DE1D5F3}\28 | App User Model DestList Provided Description | |
| {9F4C2855-9F79-4B39-A8D0-E1D42DE1D5F3}\27 | App User Model DestList Provided Title | Windows PowerShell |
| {9F4C2855-9F79-4B39-A8D0- E1D42DE1D5F3}\30 | App User Model DestList Provided Group Name | Profiles |
| {9F4C2855-9F79-4B39-A8D0- | App User Model DestList | ms-appx:///ProfileIcons/{61c54bbd-c2c6- |

| | | |
|---|---|---|
| E1D42DE1D5F3}\29 | Logo Uri | 5271-96e7-009a87ff44bf}.png |
| {9F4C2855-9F79-4B39-A8D0-E1D42DE1D5F3}\5 | App User Model ID | WindowsTerminalDev_8wekyb3d8bbwe!App |
| {9F4C2855-9F79-4B39-A8D0-E1D42DE1D5F3}\20 | App User Model Activation Context | {61c54bbd-c2c6-5271-96e7-009a87ff44bf} |
| {436F2667-14E2-4FEB-B30A-146C53B5B674}\100 | Link Arguments | {61c54bbd-c2c6-5271-96e7-009a87ff44bf} |
| {F29F85E0-4FF9-1068-AB91-08002B27B3D9}\2 | (Description not available) | Windows PowerShell |

If we look at the property key `9f4c2855-9f79-4b39-a8d0-e1d42de1d5f3\29`, it specifies the uri for the icon and supports the `ms-appx://` scheme. So for icon support we can add that property key when creating the `IShellLink`. Also note that with this approach, it needs the `file://` uri scheme in the path for custom icons.

### Launching the terminal from the jumplist

The jumplist will launch the terminal by calling the executable alias `wt.exe` with arguments to indicate the profile. The command line arguments to use are tracked in issue #607

## UI/UX Design

No UI changes are needed. The jumplist is provided by Windows.

### Localization

Depending on how the jump list items will be worded.

### Profile order

The order of the profiles in the jumplist should match the order within the application.

## Capabilities

### Accessibility

Will be provided by what Windows supports for jump lists

### Security

Should not introduce any new security issues

### Reliability

Should not introduce any new reliability issues

### Compatibility

Will have to add the ability to get notified of changes to the profile settings.

### Performance, Power, and Efficiency

The jumplist will have to be saved each time a profile change occurs but the frequency would be expected to be low.

## Potential Issues

**Should it open a new instance of the terminal or open in a new tab?**

**What should happen if a non existent profile is launched**

The jumplist is only updated when the application is running so the profiles could be modified or deleted outside and the jumplist will not be updated. Handling will be done by whatever handles the command line parsing.

## Future considerations

Other things could potentially be added to the jumplist other than profiles.

## Resources

- [1] Original feature request: issue #576
- [2] ICustomDestinationList
- [3] Windows property system