Custom HotKey Control

The Settings project provides a custom hotkey control which consumes key presses. This control can be used to set the hotkey of any PowerToy.

HotKey Control in FancyZones

 Image of hotkey control

Hotkey related files

HotkeySettingsControlHook.cs

- This function initializes and starts the `keyboardHook` for the hotkey control.

```
        public HotkeySettingsControlHook(KeyEvent keyDown, KeyEvent keyUp, IsActive
isActive, FilterAccessibleKeyboardEvents filterAccessibleKeyboardEvents)
        {
            _keyDown = keyDown;
            _keyUp = keyUp;
            _isActive = isActive;
            _filterKeyboardEvent = filterAccessibleKeyboardEvents;
            _hook = new KeyboardHook(HotkeySettingsHookCallback, IsActive,
FilterKeyboardEvents);
            _hook.Start();
        }
```

HotkeySettingsControl.xaml.cs

- The function of this class is to update the state of the keys being pressed within the custom control. This information is stored in `internalSettings`.
- It provides the following callbacks to the `HotKeySettingsControlHook`:
    - `KeyUp`: Resets the key state in `internalSettings` when a key is released.
    - `KeyDown`: Updates the user facing text of the hotkey control as soon as a key is pressed.
    - `isActive`: Sets the current status of the keyboard hook.
    - `FilterAccessibleKeyboardEvents`: This function is used to ignore the `Tab` and `Shift+Tab` key presses to meet the accessibility requirements.

HotkeySettings.cs

- Contains the structure of a HotKey where it is represented as a combination of one of the modifier keys (`Alt`, `Shift`, `Win` and `Ctrl`) and a non-modifier key.

Note

- The control displays all key presses to the user (except Tab and Shift+Tab which move focus out of the control). However, when the focus is being lost from the control, the `lastValidHotkeySettings` is set as the user facing text.