PSCalendar

This module contains a few functions for displaying a calendar in the PowerShell console. The primary function is based on code originally published by Lee Holmes at http://www.leeholmes.com/blog/2008/12/03/showing-calendars-in-your-oof-messages/. However, `v2.0.0` of this module contains a complete rewrite of the core function.

Installation

You can install this module from the PowerShell Gallery.

```
Install-Module PSCalendar [-scope currentuser]
```

> Installing this module will also install the `ThreadJob` module from the PowerShell Gallery, as that is module dependency if you want to use `Show-GuiCalendar`.

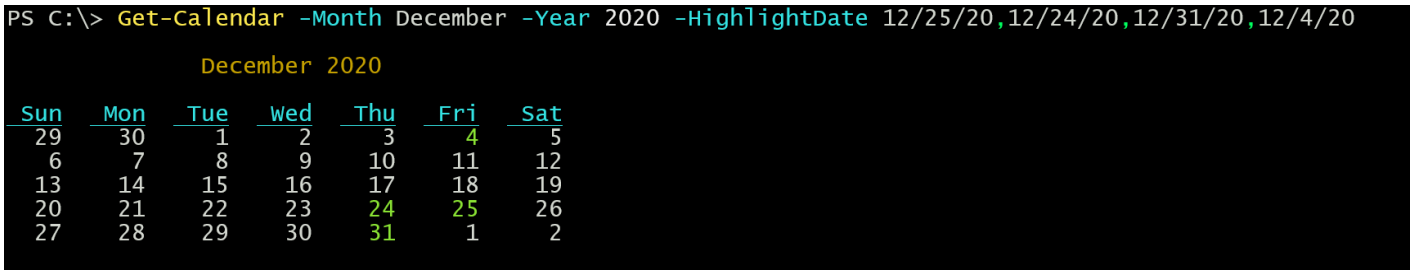
The commands in this module have been tested on PowerShell 7 both under Windows and Linux and there is no reason these commands should not work. Commands and aliases that are incompatible with non-Windows platforms are not exported.

Note: If you are upgrading to v2.0.0 or later of this module, and have older versions installed, it is recommended that you uninstall the older versions.

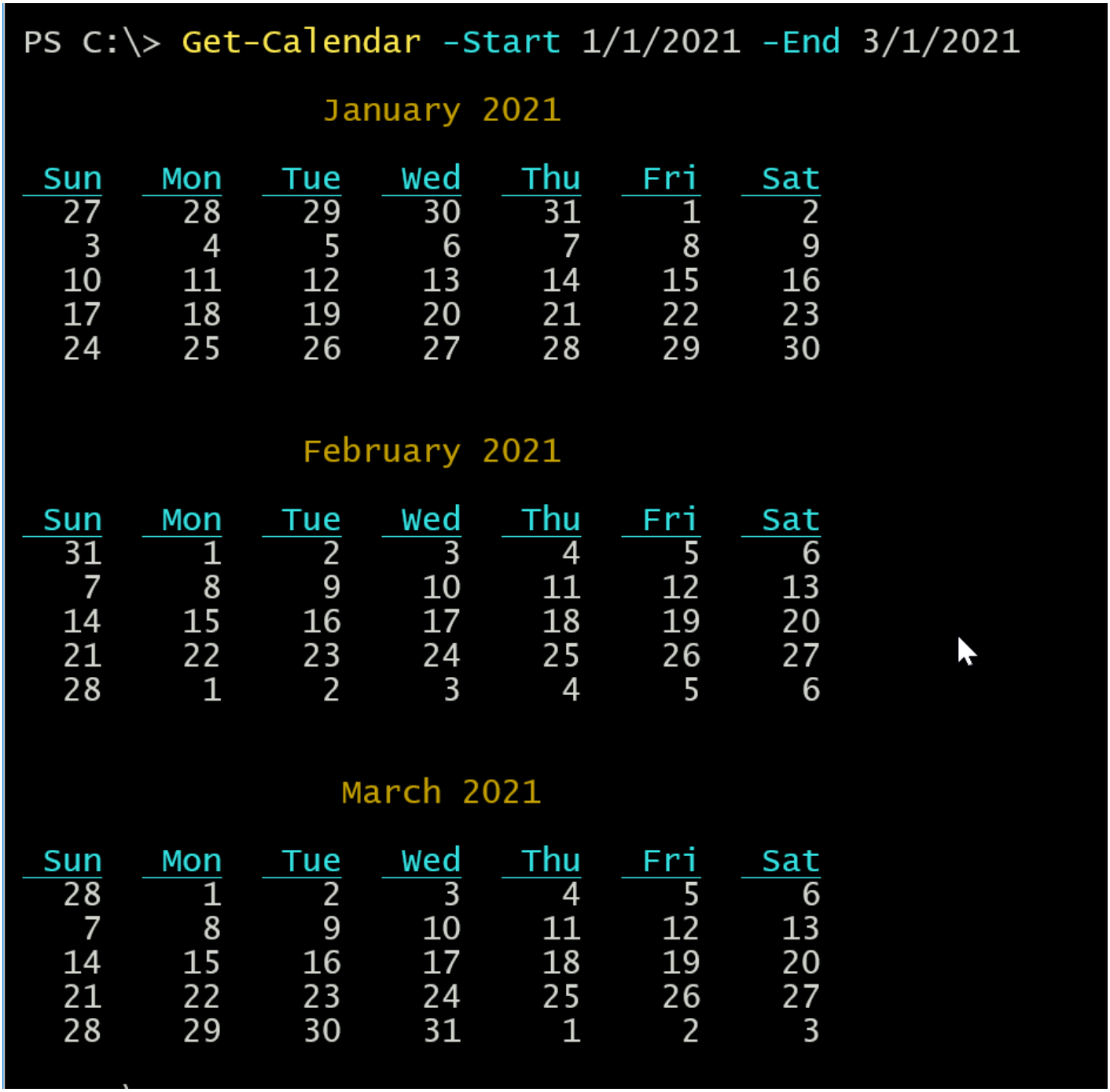
Get-Calendar

The commands in this module have been updated to take advantage ANSI escape sequences. The main function, Get-Calendar, will display the current month in the console, highlighting the current date with an ANSI escape sequence.

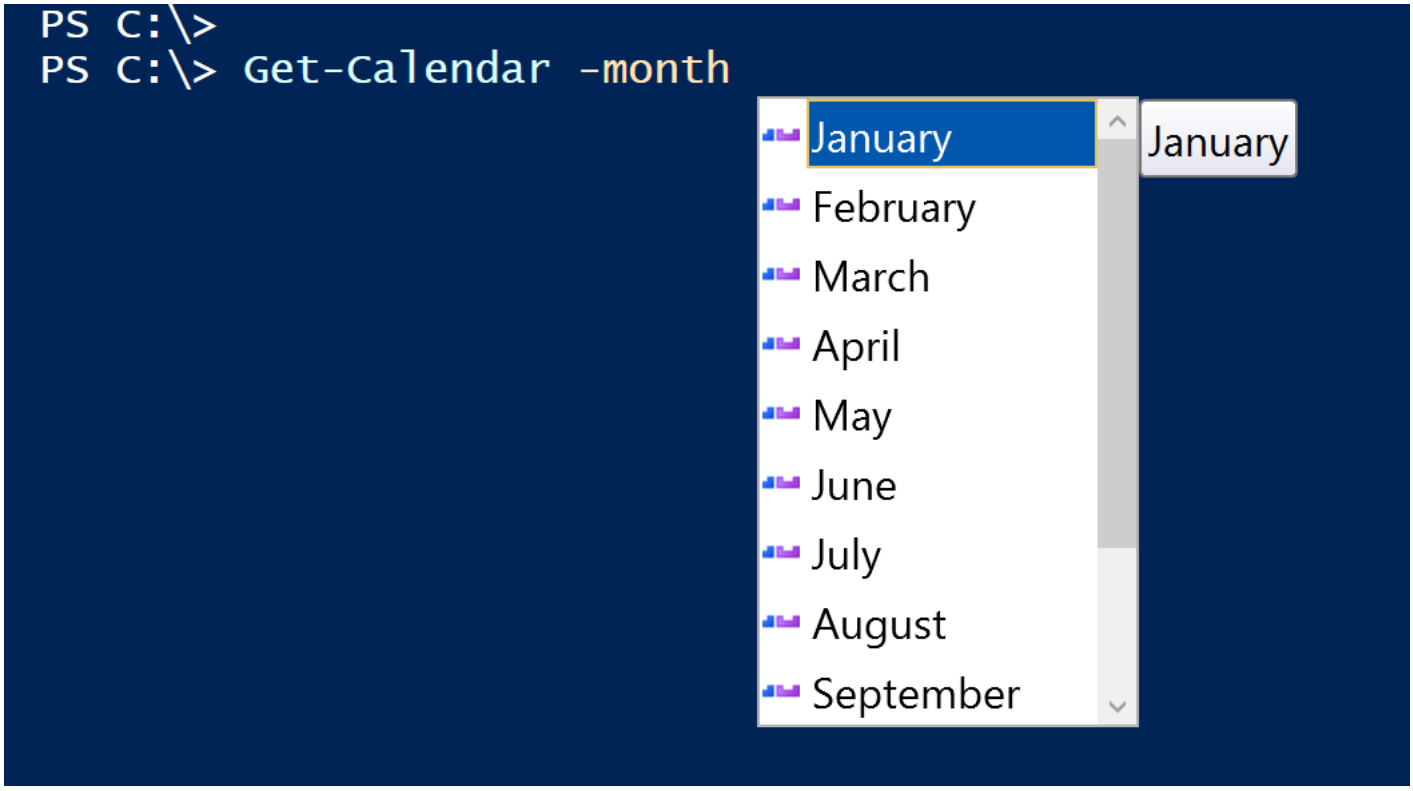
But you can also specify a calendar by month and year.



In this example you can see that I specified dates to highlight. Or you can specify a range of months.



The function should be culturally aware. The commands in this module that have a `-Month` parameter should autocomplete to culture-specific month names.



There is a similar autocompletion for `-Year` that begins with the current year and then the next 5 years. Although nothing prevents you from entering any year you want.

Show-Calendar

In previous versions of this module, there was a command called `Show-Calendar` which wrote a colorized version of the calendar to the host using `Write-Host`. This command has been rewritten and now is essentially a wrapper for `Get-Calendar`. The primary difference is that you can position the calendar.

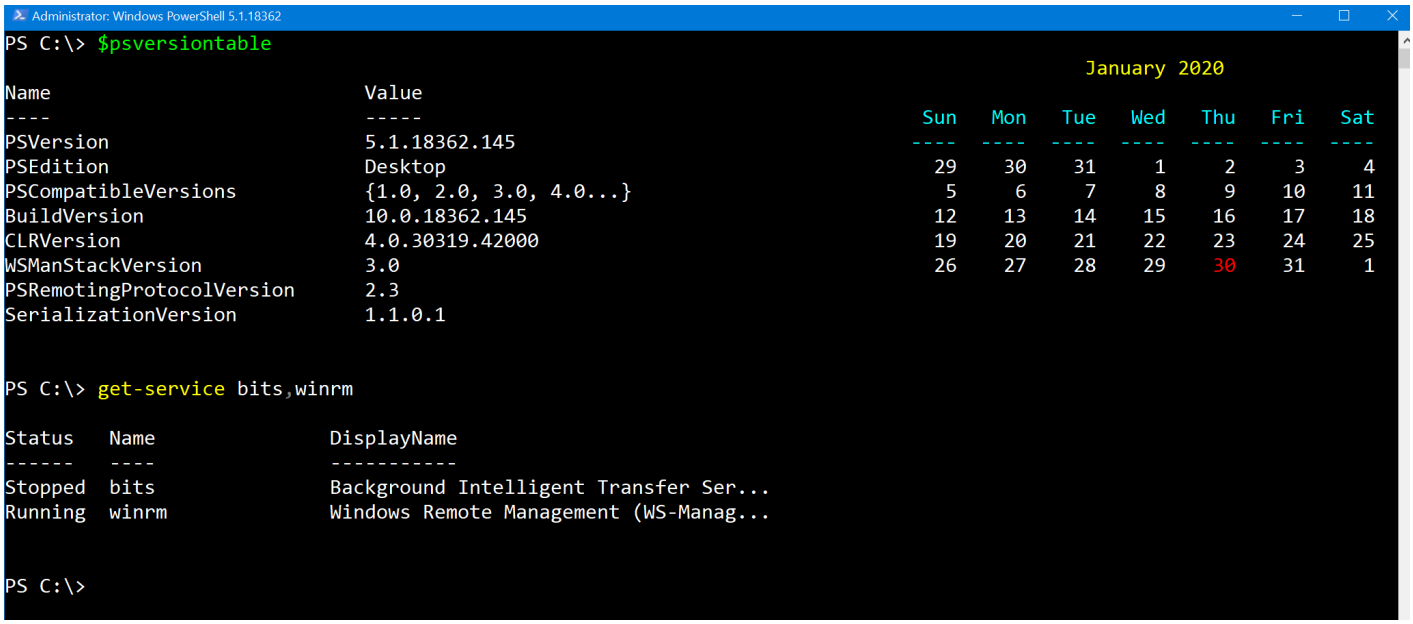
A Console Calendar Prompt

One way you might want to use this is in your PowerShell console. You can use the prompt function like this:

```
#requires -modules @{ModuleName="PSCalendar";ModuleVersion="2.1.0"}
```

```
Function prompt {

  #define a buffercell fill
  $fill = [system.management.automation.host.buffercell]::new("
",$host.ui.RawUI.BackgroundColor,$host.ui.RawUI.BackgroundColor,"complete")

  #define a rectangle with an upper left corner X distance from the edge
  $left =$host.ui.RawUI.WindowSize.width - 42

  #need to adjust positioning based on buffer size of the console
  #is the cursor beyond the window size, ie have we scrolled down?
    if ($host.UI.RawUI.CursorPosition.Y -gt $host.UI.RawUI.WindowSize.Height) {
        $top = $host.ui.RawUI.CursorPosition.Y - $host.UI.RawUI.WindowSize.Height
    }
    else {
        $top = 0
    }
  #    System.Management.Automation.Host.Rectangle new(int left, int top, int right,
int bottom)
  $r = [System.Management.Automation.Host.Rectangle]::new($left, 0,
$host.ui.rawui.windowsize.width,$top+10)

  #clear the area for the calendar display
  $host.ui.rawui.SetBufferContents($r,$fill)

  #show the calendar in the upper right corner of the console
  $pos = [system.management.automation.host.coordinates]::new($left,0)
  Show-Calendar -Position $pos

  "PS $($executionContext.SessionState.Path.CurrentLocation)$('>' *
($nestedPromptLevel + 1)) ";

# .Link
# https://go.microsoft.com/fwlink/?LinkID=225750
# .ExternalHelp System.Management.Automation.dll-help.xml

}
```

Assuming the width of your console is at least 120, this code should work. Otherwise, you might need to tweak the positioning. This should also work in Windows Terminal. If you add some highlighted dates using `$PSDefaultParameterValues`, then you'll have a calendar right in front of you.

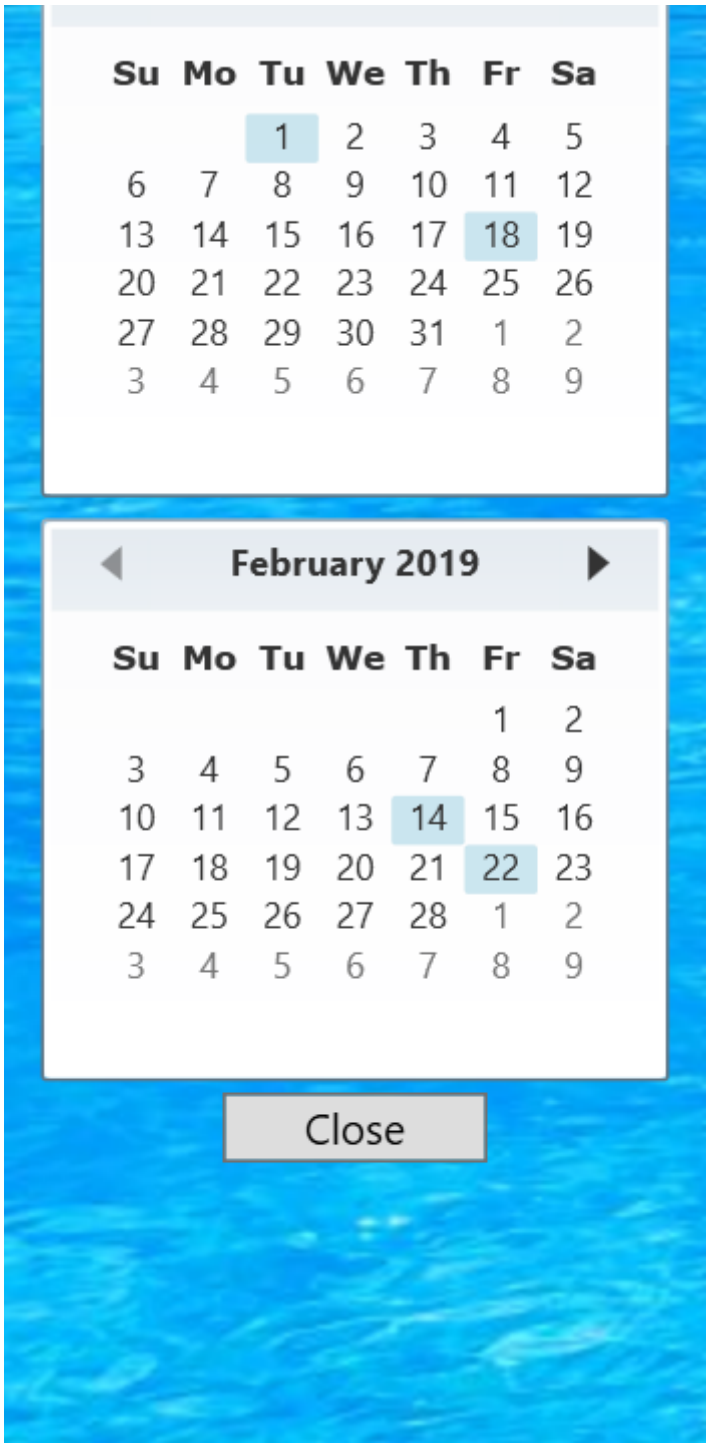


Note that any command output may be truncated because of the calendar display. This prompt function works as expected when using the Windows Terminal. Function needs work to behave as expected in a traditional PowerShell console where you might have a large buffer for scrolling.

Show-GUICalendar

Finally, you can display a graphical calendar using a Windows Presentation Foundation (WPF) based script. The function runs the calendar-related code in a runspace so it does not block your prompt. You can display up to 3 months and specify dates to highlight.

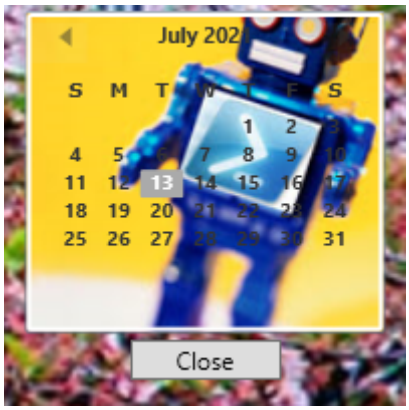
```
PS C:\> Show-GuiCalendar 12/2018 2/2019 -highlight
12/24/18,12/25/18,12/31/18,1/1/19,1/18/19,2/14/19,2/22/19
```



The calendar form is transparent. But you should be able to click on it to drag it around your screen. You can also use the + and - keys to increase or decrease the calendar's opacity. Be aware that if you close the PowerShell session that launched the calendar, the calendar too will close.

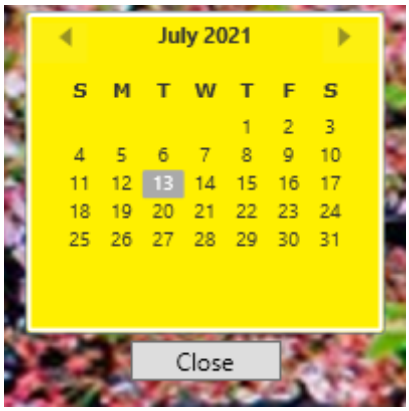
Beginning with module version 2.2.0 you can also customize the calendar background with an image:

```
Show-GuiCalendar -BackgroundImage D:\images\blue-robot-ps-thumb.jpg -Stretch
UniformToFill -FontWeight Bold
```



Or you can specify a color. You can specify a WPF brush color like Cornsilk or Wheat, or use a color code like `#FFF000`:

```
Show-GuiCalendar -BackgroundColor "#FFF000"
```



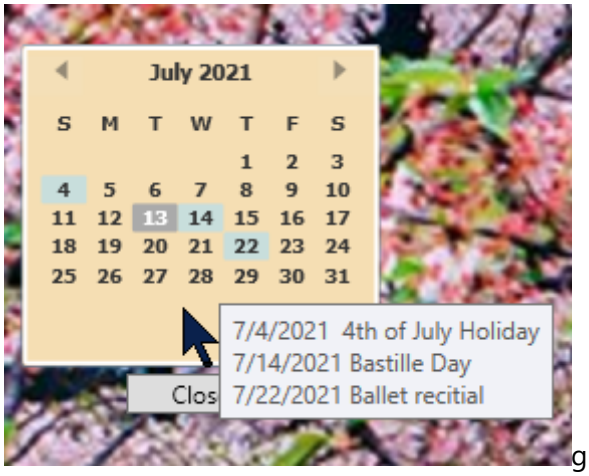
On Windows platforms, the `-BackgroundColor` parameter will autocomplete the available brush colors.

Highlight Dates with Notes

Beginning with v2.2.0, in addition to specifying an array of dates to highlight, you can also use a hashtable. The key should be the highlight date, and the value a brief description.

```
$h = @{"7/4/2021"="4th of July Holiday";"7/14/2021"="Bastille
Day";"7/22/2021"="Ballet recitial"}
Show-GuiCalendar -Start 7/1/2021 -HighlightDate $h -BackgroundColor wheat -
FontWeight Bold -Font Tahoma
```

When you pass a hashtable, you will get a tooltip popup when you hover the mouse over the month.

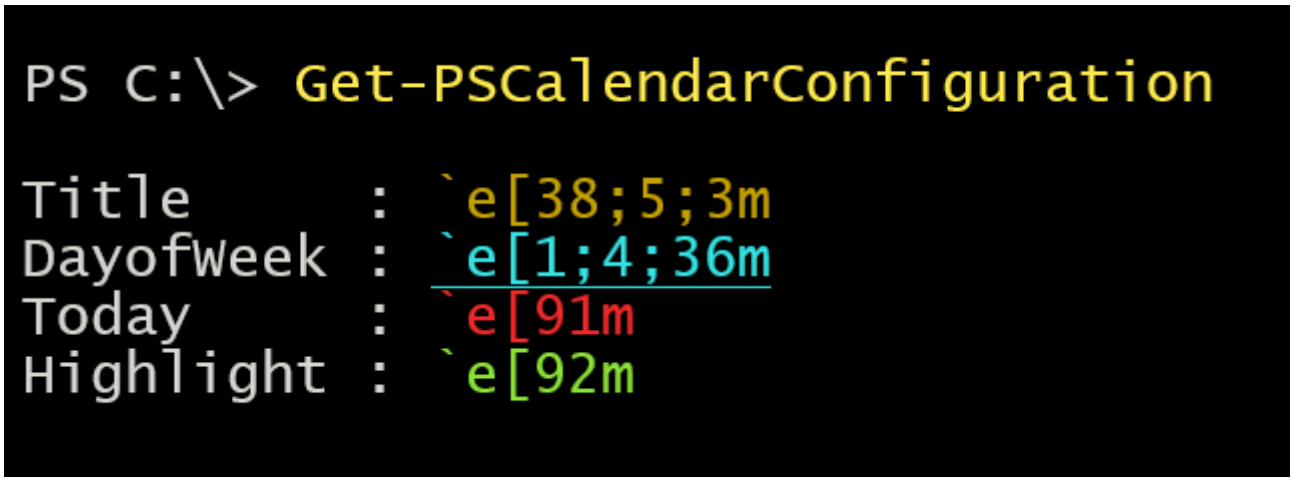


g

This function requires the WPF-related assemblies. It should work in Windows PowerShell and PowerShell 7. You will receive a warning if any incompatibility is detected.

Customizing the Calendar Appearance

Beginning with v2.0.0 of this module, ANSI escape sequences used to format the calendar are stored in module-scoped hashtable. You can use Get-PSCalendarConfiguration to view the current settings.



The output will show you the escape sequence appropriate for your PowerShell version. If you want to change a setting, you can use:

Set-PSCalendarConfiguration

You need to include the escape character but you do not need to include the closing escape sequence.

```
PS C:\> Set-PSCalendarConfiguration -title "`e[48;5;57m"
PS C:\> Get-PSCalendarConfiguration

Title      : `e[48;5;57m
DayofWeek  : `e[1;4;36m
Today      : `e[91m
Highlight  : `e[92m


PS C:\> Get-Calendar

               September 2020

 Sun    Mon    Tue    Wed    Thu    Fri    Sat
  30     31      1      2      3      4      5
   6      7      8      9     10     11     12
  13     14     15     16     17     18     19
  20     21     22     23     24     25     26
  27     28     29     30      1      2      3
```

This change lasts for the duration of your PowerShell session. If you want to make it more permanent, you will need to add the commands to your PowerShell profile script.

A Note on Culture

I've tried very hard to make the commands respect culture. Most commands now that string values to represent dates which are then treated as dates internally. For this reason, it is important that you follow the culture-specific short date format that you get from running this command:

```
(Get-Culture).datetimeformat.ShortDatePattern
```

In Windows PowerShell, all of the commands appear to respect culture settings. However, when running in PowerShell 7 there appears to be a bug in .NET Core and how it returns culture information for some cultures, specifically the first day of the week. If you run `Get-Calendar` or `Show-Calendar` and the week begins on the wrong day, use the `FirstDay` parameter to override the detected .NET values with the correct one.

```
PS C:\> Get-Calendar august -firstday Monday -highlight 1/8/2021,15,8,2021

               August 2021

 Mon    Tue    Wed    Thu    Fri    Sat    Sun
  26     27     28     29     30     31      1
   2      3      4      5      6      7      8
   9     10     11     12     13     14     15
```

```
    16    17    18    19    20    21    22
    23    24    25    26    27    28    29
    30    31     1     2     3     4
```

For example, if you are running under the `en-AU` culture, you would need to use this syntax.

Potential Issues

I have tried to make this module culture-aware. Testing across cultures is not an easy process. If you encounter a problem and are not running PowerShell under the `EN-US` culture, run the calendar command you are trying to use with `-Verbose` and post the results in a new issue. Or if you have both Windows PowerShell and PowerShell 7 installed, try the same command in both versions.

Last Updated 2021-07-13 13:50:40Z