



python



PowerShell

FPDF Library  
PDF generator

*Full credit is given to the above companies including the OS that this PDF file was generated!*

### ***PowerShell Get-Help on command 'Get-Culture'***

***PS C:\Users\wahid> Get-Help Get-Culture***

#### NAME

Get-Culture

#### SYNOPSIS

Gets the current culture set in the operating system.

#### SYNTAX

Get-Culture [<CommonParameters>]

#### DESCRIPTION

The `Get-Culture` cmdlet gets information about the current culture settings. This includes information about the current language settings on the system, such as the keyboard layout, and the display format of items such as numbers, currency, and dates.

You can also use the `Get-UICulture` cmdlet, which gets the current user interface culture on the system, and the Set-Culture (/powershell/module/international/set-culture)cmdlet in the International module. The user-interface (UI) culture determines which text strings are used

for user interface elements, such as menus and messages.

## PARAMETERS

### <CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer, PipelineVariable, and OutVariable. For more information, see [about\\_CommonParameters \(https://go.microsoft.com/fwlink/?LinkID=113216\)](https://go.microsoft.com/fwlink/?LinkID=113216).

----- Example 1: Get culture settings -----

### Get-Culture

LCID	Name	DisplayName
1033	en-US	English (United States)

This command displays information about the regional settings on the computer.

----- Example 2: Format the properties of a culture object -----

```
PS C:\> $C = Get-Culture
PS C:\> $C | Format-List -Property *
Parent           : en
LCID             : 1033
KeyboardLayoutId : 1033
Name             : en-US
IetfLanguageTag  : en-US
DisplayName      : English (United States)
NativeName      : English (United States)
EnglishName      : English (United States)
TwoLetterISOLanguageName : en
ThreeLetterISOLanguageName : eng
```

ThreeLetterWindowsLanguageName : ENU  
CompareInfo : CompareInfo - 1033  
TextInfo : TextInfo - 1033  
IsNeutralCulture : False  
CultureTypes : SpecificCultures, InstalledWin32Cultures,  
FrameworkCultures  
NumberFormat : System.Globalization.NumberFormatInfo  
DateTimeFormat : System.Globalization.DateTimeFormatInfo  
Calendar : System.Globalization.GregorianCalendar  
OptionalCalendars : {System.Globalization.GregorianCalendar,  
System.Globalization.GregorianCalendar}  
UseUserOverride : True  
IsReadOnly : False

PS C:\> \$C.Calendar

MinSupportedDateTime : 1/1/0001 12:00:00 AM  
MaxSupportedDateTime : 12/31/9999 11:59:59 PM  
AlgorithmType : SolarCalendar  
CalendarType : Localized  
Eras : {1}  
TwoDigitYearMax : 2029  
IsReadOnly : False

PS C:\> \$C.DateTimeFormat

AMDesignator : AM  
Calendar : System.Globalization.GregorianCalendar  
DateSeparator : /  
FirstDayOfWeek : Sunday  
CalendarWeekRule : FirstDay  
FullDateTimePattern : dddd, MMMM dd, yyyy h:mm:ss tt  
LongDatePattern : dddd, MMMM dd, yyyy  
LongTimePattern : h:mm:ss tt  
MonthDayPattern : MMMM dd

```

PMDesignator           : PM
RFC1123Pattern         : ddd, dd MMM yyyy HH':'mm':'ss 'GMT'
ShortDatePattern       : M/d/yyyy
ShortTimePattern       : h:mm tt
SortableDateTimePattern : yyyy'-'MM'-'dd'T'HH':'mm':'ss
TimeSeparator          : :
UniversalSortableDateTimePattern : yyyy'-'MM'-'dd HH':'mm':'ss'Z'
YearMonthPattern       : MMMM, yyyy
AbbreviatedDayNames    : {Sun, Mon, Tue, Wed...}
ShortestDayNames       : {Su, Mo, Tu, We...}
DayNames               : {Sunday, Monday, Tuesday, Wednesday...}
AbbreviatedMonthNames  : {Jan, Feb, Mar, Apr...}
MonthNames             : {January, February, March, April...}
IsReadOnly             : False
NativeCalendarName     : Gregorian Calendar
AbbreviatedMonthGenitiveNames : {Jan, Feb, Mar, Apr...}
MonthGenitiveNames     : {January, February, March, April...}

```

```
PS C:\> $C.DateTimeFormat.FirstDayOfWeek
```

```
Sunday
```

This example demonstrates the vast amount of data in the culture object. It shows how to display the properties and sub-properties of the object.

The first command uses the ``Get-Culture`` cmdlet to get the current culture settings on the computer. It stores the resulting culture object in the ``$C`` variable.

The second command displays all of the properties of the culture object. It uses a pipeline operator (``|``) to send the culture object in ``$C`` to the ``Format-List`` cmdlet. It uses the `Property` parameter to display all (``*``) properties of the object. This command can be abbreviated as ``$c | fl *``.

The remaining commands explore the properties of the culture object by using dot notation to display the values of the object properties. You can use this notation to display the value of any property of the object.

The third command uses dot notation to display the value of the Calendar property of the culture object.

The fourth command uses dot notation to display the value of the DateTimeFormat property of the culture object.

Many object properties have properties. The fifth command uses dot notation to display the value of the FirstDayOfWeek property of the DateTimeFormat property.

#### REMARKS

To see the examples, type: "get-help Get-Culture -examples".

For more information, type: "get-help Get-Culture -detailed".

For technical information, type: "get-help Get-Culture -full".

For online help, type: "get-help Get-Culture -online"