Add-GpsToImages

Purpose:
Overlay JPG image files with text derived from GPS information in the image's EXIF data.
JPG images can contain extended information (so-called EXIF data). This data may contain GPS information such as the geographical position where the picture was taken.
The GPS location (latitude and longitude) can be used to look up the correspondig place name.
Using this script, the location text can be placed on the image itself.
Also, the filename can be changed to reflect date, location, or both.

The lookup of GPS information is done via the free Openstreetmap API, which is limited to about one request per second.
Depending on the place names, data may be given in non-Latin characters, e.g. 北京 instead of Beijing.
You can use a translation file to map such names to characters and languages of your choice

The script can be used in different ways:
1) process all images immediately
2) extract GPS information and create a text file mapping image names to GPS data.
   This file can be edited before actually writing data into the images
3) apply a translation file to the GPS file
4) apply the previously created GPS file and write the data into the images

As new user, be sure to check the descriptions of all parameters.

Usage:

To add GPS information directly to the images:
```
Add-GpsToImages.ps1 -source <String> -destination <String> [-translationFile <String>]
[-textFormat <String>] [-maxTextPercent <Int32>] [-renameFormat <String>]
```

To create a file with GPS information:
```
Add-GpsToImages.ps1 -source <String> [-translationFile <String>] -gpsFile <String>
```

To apply the data from the GPS file to the images:
```
Add-GpsToImages.ps1 -destination <String> [-translationFile <String>] -gpsFile <String>
[-textFormat <String>] [-maxTextPercent <Int32>] [-renameFormat <String>]
```

To apply the translation file to the GPS file:
```
Add-GpsToImages.ps1 -translationFile <String> -gpsFile <String>
```

Parameters:

-source
Path of the source files.
If you give the destination path as well, GPS data is written directly into the images.
If you give the gpsFile path, a GPS file is created.
See also -gpsFile and -translationFile parameters.

-destination
Path for the processed images, it MUST be different from the source folder.
If the source path is given as well, GPS data is written directly into the images.
If you give the gpsFile path, a GPS file is created.
See also -gpsFile and -translationFile parameters.

-translationFile
Place names are returned by the API in local script and language, e.g. 北京, Roma, or София.
A text file mapping may be used to map these into your desired script and language.
The file format is <originalname>;<translatedname>, e.g.

```
北京;Beijing
Roma;Rome
София;Sofia
```

If the source, gpsFile and translationFile parameters are given but no translation file exists, a new file is created containing all detected place names. If the translation file exists, place names that already have entries (and possibly translations) are not changed, but will be sorted alphabetically. New place names will be added to the end of the file.

If only the gpsFile and translation file parameters are given, the translation file is applied to the GPS file, i.e. all words appearing in the translation file will be replaced by their translations.

-gpsFile
A text file mapping image path and name to GPS data.
This file is created when the source and gpsFile parameters are specified, and its data is written into the images, when the gpsFile and destination parameters are specified.
After creation, the file can be edited with any text editor. The format is <path>;<filename>;<date>;<orientation>;<latitude>;<longitude>;<altitude>;<country>;<city>, e.g.
C:\pictures;IMG0001.JPG;2023:06:23 14:33:32;1;39.90;116.39;44.2;中国;北京
C:\pictures;IMG0055.JPG;2025:01:30 12:17:28;1;41.90;12.45;133,30;Italia;Roma

-textFormat
Specifies the format in which the GPS date will be written onto the image. The following placeholders will be replaced by the actual values, all other text will be written as specified:

!file	filename of the image
!date	date picture was taken in the format YYYY-MM-DD
!orien	orientation as given in EXIF data
!lat	latitude in degrees, minutes, and seconds, followed by N or S
!lon	longitude in degrees, minutes, and seconds, followed by W or E
!alt	altitude in metres
!country	name of country corresponding to GPS location
!place	name of village, town, city corresponding to GPS location
!monthyear	date picture was taken in the format <monthname>-YYYY
!time	time picture was taken

Example: the default value '!place (!country) !monthyear' will result in '北京 (中国) 2023-06-23' and 'Roma (Italia) 2025-01-30'

-maxTextPercent
Maximum width of the text in percent of the image width. If text would be longer than specified, the font size will be reduced. The default value is 75.

-renameFormat
When this parameter is NOT given, the destination file name is the same as the source file name.
If this parameter is given,the file name is defined by the specified place holders. Note that the resulting string must not contain characters which are forbidden in file names!
Allowed placeholders are:

y	year (four digits)
m	month (two digits)
d	day (two digits)
p	placename (see above comment on forbidden characters)
c	country (see above comment on forbidden characters)

A four-digit counter is added to any filename thus generated.

Example: the renameFormat 'y-m-' results in filenames like '2023-06-0001.jpg' and '2025-01-0001.jpg'

Get-ResourceCostDetails

Purpose:
Collect usage and cost data for resources of given resource type in Azure in given subscriptions.
Data is written to a set of CSV files (one file per resource type).

Usage:
```
Get-ResourceCostDetails -subscriptionFilter <subscription>[,<subscription>] -outFile <filename>
[-delimiter <character>] [-resourceTypes <type>[,<type>]] [-excludeTypes <type>[,<type>]]
[-billingPeriod <billingperiod>] [-totals] [-consolidate] [-consolidateOnly] [-showUsage]
[-showUnits] [-count]

Get-ResourceCostDetails [-subscriptionFilter <subscription>[,<subscription>]]
[-resourceTypes <type>[,<type>]] [-excludeTypes <type>[,<type>]] -WhatIf
```

Parameters:

```
-subscriptionFilter  Mandatory (optional when -WhatIf is used). Single filter or comma-separated
                     list of filters. All subscriptions whose name contain the filter expression
                     will be analysed.
-outFile             Mandatory except when -WhatIf is used. Write output to a set of CSV files.
                     Since the results have a different format for each resource type, results
                     are not written to a single CSV file, but to separate files, one for each
                     resource type. For each file, the resource type will be inserted into the
                     name before the final dot.
-delimiter           Separator character for the CSV files. Default is the list separator for
                     the current culture.
-resourceTypes       Single filter or comma-separated list of filters. Only resources with
                     matching types will be analysed. If '*' is given as filter, all resource
                     types will be evaluated, except those which are excluded by default and
                     except those listed with the excludeTypes.
-excludeTypes        Single resource type or comma-separated list of types, evaluation of these
                     types will be skipped. Some types will be excluded by default unless
                     specified in the resourceTypes parameter.
-billingPeriod       Collect cost for given billing period, format is 'yyyyMMdd',default is the
                     last month.
-totals              Display the total cost per resource as last column, i.e. the sum of all
                     cost metrics.
-consolidate         create an additional report file with a matrix of cost per subscription and
                     resource type
-consolidateOnly     create only the report file with a matrix of cost per subscription and
                     resource type, omitting the per-resource type files.
-showUsage           Display usage information for each cost item additionally to the cost.
-showUnits           Display the units for usages and cost as second header line.
                     This is useful with the -usage switch, as the metrics come in 1s, 10000s,
                     or so.
-count               Create an additional file containing the count of resources by subscription
                     and resource type
-WhatIf              Don't evaluate costs but show a list of resources and resource types which
                     would be evaluated.
```

Get-AzurePrice

Purpose:
Lists the price for a specified VM or managed disk

Usage:

```
Get-AzurePrice -VMType <vmtype> [-Reservation <reservation>] [-AHB] [-Currency <currency>]
Get-AzurePrice -DiskType <disktype> [-Redundancy <redundancy>] [-Currency <currency>]

    Get-AzurePrice -VMType [...] displays estimated monthly fee for a given VM type
    <vmtype>          must be written exactly as in the price table, including proper
                      capitalization. The space may be replaced by an underscore,
                      e.g. use "D4s v4" or "D4s_v5"
    <reservation>     is 0, 1, or 3 (years). If omitted, all three values will be reported,
                      separated by semicolons
    <currency> can be any valid three-letter currency code, default value is EUR

    -AHB              if this switch is given, prices are given without Windows license (i.e.
                      using Azure Hybrid Benefit)

Get-AzurePrice -DiskType [...] displays estimated monthly fee for a given disk type
    <disktype>        must be written exactly as in the price table, including proper
                      capitalization, e.g. "E10" or "S20"
    <redundancy>      is either LRS or ZRS, default is LRS
    <currency>        can be any valid three-letter currency code, default value is EUR
```

Get-ConnectedNICs

Purpose:
List network interfaces in given subscriptions with IP address, VNet name, network address and DNS record, also tests whether NIC is responding to pings

Usage:
```
Get-ConnectedNICs [-subscriptionFilter <filterexpression>] [-outFile <outfilename>] [-noPing]
    -subscriptionFilter   mandatory parameter, list NICs in subscriptions matching the filter
    -outFile              writes results to a semicolon-separated CSV format if this parameter
                          is given
    -noPing               skips testing whether the NIC is responding to a ping
```

Get-FileAccesses

Purpose:
provides count and size statistics about file extensions and ages on file storages

Usage:
```
Get-FileAccesses -serverName <servername> [-shareName <sharename>] [-onAccess] [-noAges]
[-noExtensions] [-priority (BelowNormal | Normal | AboveNormal | High | Realtime)]
    -serverName           Mandatory, evaluates data on given share(s) of <servername>
    -shareName            Analyzes files on the given share.
                          If omitted, analyzes data on all non-hidden shares of the given
                          server.
    -onAccess             Uses lastAccess for age calculation.
                          If omitted, uses lastWrite for age calculation
    -noAges               Ignores file age, lists by extensions only (if -noExtensions is not
                          given)
    -noExtensions         Ignores extensions, lists by age only (if -noAge is not given)
                          If BOTH -no... switches are given, script only returns total file
                          count and size
    -priority             Starts process with given priority. Use with care.
                          Possible values are BelowNormal | Normal | AboveNormal | High |
                          Realtime
```

```
Examples:
to list file count and size by age, based on last accessed date'
Get-FileAccesses -ServerName myserver -ShareName myshare -onAccess -noExtensions

to see all properties in table format, use ft -Property *
Get-FileAccesses -ServerName myserver -ShareName myshare | ft -Property *
```

Get-NSGRules

Purpose:
Lists the NSG rules in the given subscriptions in a summarized or detailed way

Usage:
```
Get-NSGRules -subscriptionFilter <filterexpression> [-details | -briefDetails]
    -subscriptionfilter   mandatory parameter, list NSGs in subscriptions matching the filter
    -details              list all rules in order of their priority
    -briefDetails         list every rule, but fewer details
                          if neither "details" switch is present, then all open ports are
                          listed, regardless of the actual source and target networks.
                          Since this mixes rules, it gives you an overview of ports but no
                          reliable information about security
```

Get-PrinterQueues

Purpose:
Lists all printer queues on all computers matching the filter.
The output will be for the each printer on the matching computer(s),
name of computer, printer, driver, and printer IP address.

Usage:
```
    Get-PrinterQueues -Filter <filter> [-details] [-ping] [-outFile <outfilename>]
```

```
    -Filter
              name(s) of computer(s) whose queues shall be displayed. Filter may contain
              wildcards
    -details  Give additional details for each printer: name of shared printer, location,
              comment, and port name
    -ping     will try to ping each printer and output an additional field 'IsLive'
    -outFile  if given, exports result into a semicolon-separated CSV file
```

Get-VirtualMachineInfo

```
Purpose:
Lists the VMs, their SKU, OS, OS version, and their disks

Usage:
Get-VirtualMachineInfos -subscriptionFilter <filterexpression>
[-disks [-asString | -aggregatedString]] [-ipAddresses] [-ping]
[-outFile <filename> [-separator <separator>]]
Get-VirtualMachineInfos -subscriptionFilter <filterexpression> -all
[-asString | -aggregatedString] [-outFile <filename> [-separator <separator>]]

Returns a list of all subscriptions, virtual machines, their SKU, IP addresses, and SKUs of
                  attached disks in subscriptions matching the filter
-all              includes -disks, -ipAddresses, -ping
-disks            show OS and data disk SKUs
-asString         shows the disks in string format
-aggregatedString shows the disks in an aggregated string format
-ipAddresses      show IP address(es)
-ping             ping VM to see whether it is live
-outFile          if given, exports result into a CSV file
-separator        separator for items in CSV file, default is semicolon
```

Get-VirtualNetworks

```
Purpose:
Lists all virtual networks, subnets, IP addresses and -ranges for the specified
subscription(s)

Usage:
Get-VirtualNetworks -subscriptionFilter <filterexpression> [-outFile <outfilename>]
[-includeSubnets]

-subscriptionFilter    mandatory. Lists networks in subscriptions matching the filter
-outFile               will write output into semicolon-separated CSV file,
                       otherwise output is a list of objects
-includeSubnets        list subnets for each VNet
```

Get-AzureResourceData

```
Purpose:
Returns a list of resources of selected type(s) in selected subscription(s), along
with some properties and metrics.

Usage:
Get-AzureResourceData -subscriptionFilter <filterexpression> [-VMs] [-SqlServer] [-DbAas]
                      [-Storage] [-ResourceList [-details] [-lastHours <hours>]
                      [-billingPeriod <billingperiod>]] [-outFile <filename> [-separator]]
Get-AzureResourceData -subscriptionFilter <filterexpression> [-all] ...

Parameters:
   -subscriptionFilter  Single filter or comma-separated list of filters. All subscriptions
                        whose name matches the filter expression will be analysed.
   -VM                  Show VMs and their properties
   -SqlServer           Show SQL server VM properties
   -DbAas               Show Azure SQL (databases aaS)
   -Storage             Show storage accounts
   -Snapshot            Show snapshots
   -all                 All of the above switches
   -lastHours           collect metrics within the given time period, default is 24 hours
   -ResourceList        Show count of resource types in subscription
```

```
-billingPeriod      Collect resource cost for given billing period, default is the last
                    month. Note that there may be no data available for the given billing
                    period.
-details            Show list of resources in subscription
-outFile            Export result into a CSV file rather than on the console
                    NOTE: separate files will be created for different resource types.
                    Two characters will be added to the file names to make them different.
-separator          Separator for items in CSV file, default is semicolon
-WhatIf             Just display the names of the subscriptions which would be analysed

 NOTE: you may adjust this script e.g. to add or remove metrics. These parts of the code
       are marked with # >>>>> and # <<<<<.
       Refer to the comments inside the code for further information.
```