Powershell

Rahul R Birmiwal

Usage

1. To create a powershell (.ps1) script, say named *foo.ps1*, open Windows Powershell ISE.
2. To create an executable that can run *foo.ps1* then create a file in Notepad with the text:

```
Powershell.exe -executionpolicy remotesigned -File foo.ps1
Pause
```

1. Save this new file in the same directory as *foo.ps1* and Save As: it *run.bat*
2. Double click latter to execute the Powershell script

Set Up

Important libraries must be downloaded from the internet. These are "standard" open-source Powershell libraries

1. Copy the text below

```
Install-Module PSExcel -scope CurrentUser
Install-Module ImportExcel -scope CurrentUser
```

2. Paste into Notepad
3. Save as “install.ps1” (All File Types) , to say the documents folder.
4. Copy the text below

```
Powershell.exe -executionpolicy remotesigned -File install.ps1
Pause
```

5. Paste into a new Notepad
6. Save as “init.bat” (All File Types) to same folder where saved install.ps1

Functionality

Powershell is a very powerful way to automate tasks. It can do hundred and hundreds of things, and while we only discuss some of the most important Excel-related tasks below, one can simply search in Google: "how to do in powershell" and I am sure someone else has invented a solution to any arbitrary task.

Below we discuss

- Opening/closing Excel files
- Selecting particular columns
- *joining* worksheets/tables
- Writing new data to Excel
- Filtering data based on some user-defined consitions

The sample Powershell script below uses accompanying Excel files, which we show below (note that we use Python pandas to open and display the files)

In [3]:
```
import pandas as pd

enrollment = pd.read_excel("enrollment.xlsx")
lives_in = pd.read_excel("livesIn.xlsx")
major_counts = pd.read_excel("major_counts.xlsx")
college_locations = pd.read_excel("college_locations.xlsx")
```

In [2]:
```
college_locations
```

Out[2]:

	College	City	State
0	Harvard	Cambridge	MA
1	Washington	Seattle	WA
2	Berkeley	Berkeley	CA
3	MIT	Cambridge	MA
4	USC	Los Angeles	CA

In [4]:
```
lives_in
```

Out[4]:

	FirstName	LastName	City
0	Michael	Bradfor	Cambridge
1	Joe	Ashcroft	Seattle
2	Jill	Bennet	Los Angeles
3	Megan	Maroney	Los Angeles
4	Raj	Shiv	Cambridge
5	Carter	Lee	Cambridge
6	Ashley	Xu	Berkeley
7	Jill	Delaurentis	Berkeley

In [5]:
```
enrollment
```

Out[5]:

	ID	FirstName	LastName	Age	Major	College
0	0	Joe	Ashcroft	21	Computer Science	Washington
1	1	Jill	Bennet	18	Computer Science	USC
2	2	Ashley	Xu	19	Art History	Berkeley
3	3	Jill	Delaurentis	18	English	Berkeley
4	4	Michael	Bradfor	22	Physics	Harvard
5	5	Raj	Shiv	19	Electrical Engineering	MIT
6	6	Carter	Lee	21	History	MIT
7	7	Megan	Maroney	22	Economics	USC

In [6]:
```
major_counts
```

Out[6]:

	Major	Number in Major (All US)
0	Computer Science	67898556
1	Art History	676739
2	English	100089569
3	Physics	5636378
4	Electrical Engineering	67989999
5	History	217888888
6	Economics	34647888

With these tables in mind, the sample powershell (.ps1) script below walks through step-by-step how to handle some various tasks an actuarial dept might want to use with its own Excel files:

```
Import-Module PSExcel
Import-Module MergeCSV
Import-Module importexcel
Import-Module ImportExcel

#-------------------------------------------------------------------------------
-
# PARAMETERS #
[string]$WorkingDirectory = $PSScriptRoot #get the location of this file
[string]$pathname = Join-Path  -Path $WorkingDirectory -ChildPath "enrollment.xlsx"
<#
    Join-Path is a function that takes two arguments.
        1. -Path is the path of this folder, which we set to $WorkingDirectory
        2. -ChildPath is the name of the desired file, with file extension
#>
#-------------------------------------------------------------------------------
-

# Get a workbook
Write-Output 'Example 1'
$Workbook = Import-Excel  $pathname
$Workbook | Format-Table
#-------------------------------------------------------------------------------
-

# Get a workbook only selecting FirstName, LastName and Major
Write-Output 'Example 2'
$Workbook2 = Import-Excel  $pathname | Select FirstName, LastName, Major #where Fir
stName, LastName, Major are exact column names in the Excel File
$Workbook2 | Format-Table
#-------------------------------------------------------------------------------
-

# Inner Join enrollment.xlsx with college_locations.xlsx
Write-Output 'Example 3'
$enrollment = Import-Excel -Path (Join-Path -Path $WorkingDirectory -ChildPath "enr
ollment.xlsx")
<#
    In the above line, we set the "PATH" parameter of the function Import-Excel to
 the value within the parenthesis
    Note that the parenthesis is a must due to left-to-right order of operation exe
cution

#>

$locations = Import-Excel -Path (Join-Path -Path $WorkingDirectory -ChildPath "coll
ege_locations.xlsx")
```

```
$merged = Join-Object -Left $enrollment -Right $locations -LeftJoinProperty "Colleg
e" -RightJoinProperty "College" -Type  OnlyIfInBoth
$merged | Format-Table
#------------------------------------------------------------------------------
-



# Get only College Name and City and Display
Write-Output 'Example 4'
$merged | Select College, City | Format-Table
#------------------------------------------------------------------------------
-

# Get only UNIQUE College Name and City and Display
Write-Output 'Example 5'
$merged | Select -Unique College, City  | Format-Table
#------------------------------------------------------------------------------
-

# Select Person (firstname & lastname) and their current city
Write-Output 'Example 6'
$livesIn = $merged | Select -Unique FirstName, LastName, City
$livesIn | Format-Table
#------------------------------------------------------------------------------
-

#Export livesIn to a Excel file
$livesIn | Export-Excel -Path (Join-Path -Path $WorkingDirectory -ChildPath "livesI
n.xlsx")
#------------------------------------------------------------------------------
-

# Select only rows where city==Cambridge
Write-Output 'Example 7'
$livesIn | Where-Object {$_.City -eq 'Cambridge'} | Format-Table
#------------------------------------------------------------------------------
-
```