# לימוד עצמי מבוא בPowershell

**שימו לב:**

**עקבו אחר התרגולים בל"ע ובצעו אותם על עמדת הקצה שלכם.**

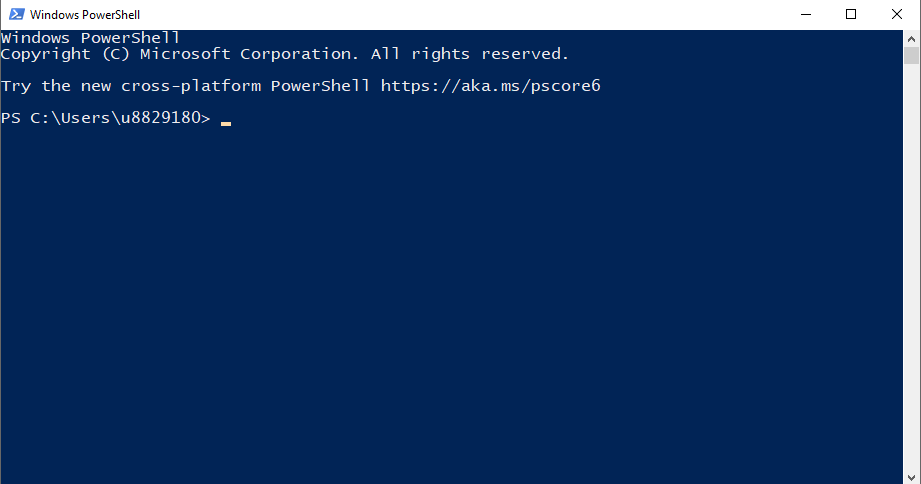
**ניתן ואפילו רצוי להשתמש באינטרנט!**

**שימוש ב-PowerShell**

על מנת להיכנס לכלי יש לפתוח את שורת ה-Run ולהקליד PowerShell.

חלון ה-PowerShell שנפתח הוא ה-Interpreter שמפרש כל פקודה ופקודה, וממיר את הפקודות לשפת מכונה על מנת להריץ אותן (זהו הShell ).

**\***מומלץ תמיד לפתוח את ממשק זה בהרשאות Administrator – לצורך כך נוכל לרשום "PowerShell" בחיפוש בתפריט הStart, ושם ללחוץ מקש ימני על PowerShell, ולבחור באפשרות "Run As Administrator".



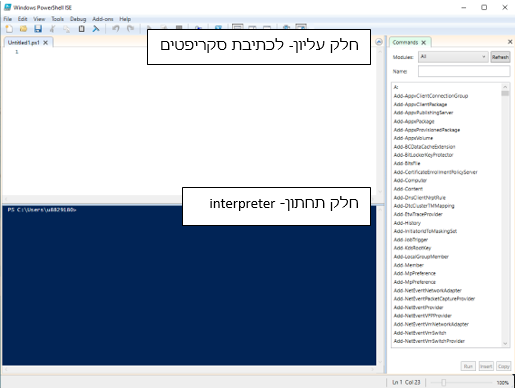


**Scripting**

(Integrated Scripting Environment) **PowerShell ISE**

ממשק מייקרוסופטי שמטרתו לשמש פלטפורמה נוחה לכתיבת והרצת **סקריפטים** בPS. היכנסו לממשק דרך powershell ise משורת ה-Run. הממשק מספק פיצ'רים כמו השלמת פקודות נוחה, צביעת קוד ואפשרות ל Debugging.

הכלי בנוי משני חלקים עיקריים:

1. **חלק תחתון** – הShell, ממשק אינטראקטיבי של PowerShell (כמו Idle ב-Python). Interpreter של השפה, כתיבת פקודה והרצתה באופן מידי.
2. **חלק עליון** – אזור לכתיבת סקריפטים. בו נכתוב קטעי קוד ארוכים. כאשר נריץ את הקוד עצמו- הוא ירוץ בחלק התחתון של הממשק. 

**אופן עבודה מומלץ בממשק ה-ISE**

1. בדיקת פקודה בודדת בממשק האינטראקטיבי.
2. במידה והפקודה מוציאה פלט תקין וטוב (ע"פ הצורך) יש לרשום אותה בחלק העליון ולהוסיף אותה לסקריפט שלנו.
3. הרצת הסקריפט כולו ובדיקת תקינותו.
4. חזרה על כל הפעולות עד סיום כתיבת הסקריפט.
5. שמירת הסקריפט בפורמט Ps1 (MyScript.Ps1).

**חזרה על פקודות בסיסיות**

**הצילו!**  
כמו בכל שפה שהיא, אנחנו נצטרך עזרה עם פקודות וחוקים של השפה.  
PowerShell זו שפה מתועדת היטב, ואנחנו לא נדרשים ללכת לאינטרנט לעזרה כל עוד אנחנו יכולים להריץ את 2 הפקודות הבאות: Get-Help ו- Get-Command.

**Get-Help** – מציגה לנו דפי עזרה שונים בנושא שנחפש.

* בכדי לקבל עזרה על פקודה ספציפית נשתמש בפקודה Get-Help באופן הבא:

**Get-Help <command>**

(במקום command תופיע פקודה כלשהי)

בצעו את הפקודה Get-Help Get-Help.  
**לפני שתענו על השאלות הבאות, פנו לנספח א' (נמצא בסוף הל"ע).**

באמצעות איזה פרמטר ניתן לראות דוגמאות עבור פקודה ב-PS?

באמצעות איזה פרמטר ניתן לקבל עזרה מלאה (כוללת דוגמאות ופרמטרים)?

|  |
| --- |
| -examples |
| -detailed |
|  |

**Get-Command** – מחזירה את רשימת הפקודות האפשריות ב-PowerShell

אפשר לחפש את כל הפקודות שמתחילות **בפועל** מסוים, או את כל הפקודות שמבוצעות על **אובייקט** מסוים, בעזרת הפרמטרים הבאים:

Get-Command –Verb Get

פקודה זו תיתן לנו את כל הפקודות שמתחילות ב-Get.

Get-Command –Noun Process

פקודה זו תיתן לנו את כל הפקודות שניתן לבצע על Processes.

**\*שימוש ב-Tab גם יכול לעזור בשימוש בשפה -** משלים פקודות ונתיבים.

**פרמטרים**

כדי להשתמש ברוב הפקודות, חייבים לתת להן קלט כלשהו לעבוד עליו. בשביל לספק לפקודה קלט(ארגומנט) יש לנו פרמטרים. פרמטרים קיימים כמעט בכל פקודה בPS, והם משפיעים על אופן ריצת הפקודה. פרמטר יתחיל ב – (מקף) ולאחר מכן שם הפרמטר.

לדוגמה:

Move-Item **-**Path C**:**\Temp **-**Destination C**:**\Logs

הפקודה הבאה מקבלת שני פרמטרים – מקור (Path) ויעד (Destination), ומעבירה את תוכן התיקייה C:\Temp לנתיב C:\Logs.

בשביל לחקור פרמטרים עבור פקודה מסוימת, ניתן להסתכל על דף העזרה של הפקודה. ניתן גם לכתוב את הפקודה, אחריה לכתוב "-" (בלי גרשיים) כאילו אנחנו הולכים לרשום שם של פרמטר, ואז ללחוץ TAB שישלים שם של פרמטר, כך נקבל את רשימת הפרמטרים שאפשר להכניס.

**פקודות בסיסיות**

כעת יש לנו את בסיס ההבנה של פקודות בPS באופן כללי, אפשר להתחיל להכיר פקודות שונות:

לפניכם מספר פקודות שימושיות. עבור כל פקודה, רשמו מה מבצעת הפקודה ודוגמה עבורה. ניתן להשתמש הן בGet-Help והן באינטרנט (Microsoft docs).

|  |  |  |
| --- | --- | --- |
| **Cmdlet** | **מה מבצעת** | **דוגמא** |
| Set-Location | מעבירה את מיקום(path) העבודה הנוכחי | Set-Location -Path "C:\" |
| New-Item | יוצרת פריט חדש | New-Item -Path "c:\" -Name "logfiles" -ItemType "directory" |
| Get-Content | מביאה את התכולה של פריט מסויים במקום מסויים | Get-Content -Path .\LineNumbers.txt |
| Get-ChildItem | מביאה את הפריט ואת "ילדיו" במקום אחד או יותר | Get-ChildItem -Path C:\Test -Name |
| Write-Host | כותבת פלט למארח(המשתמש\למסך) | Write-Host " string" |

בשאלות הבאות ניתן להיעזר בפקודה Get-Command בשביל למצוא את הפקודות המתאימות, וב-Get-Help למציאת הפרמטרים.

מצאו 3 פרמטרים שונים של הפקודה Get-ChildItem הסבירו על כל אחד מהם.

|  |
| --- |
| -name מביאה רק את שמם של הפריטים  -path פרמטר כדי לציין את הכתובת של הפריטים  -forceמכריחה את המערכת (אך בלי אפשרות לגשת למקום מוגבל) לגשת לקבצים (גם מוחבאים) |

צרו קובץ בנתיב C:\Course בשם מספר הקורס שלכם. אם נתיב התיקייה אינו קיים - צרו אותו באותה הפקודה.

|  |
| --- |
| New-Item -Name "7" -Path "C:\Course" -ItemType "file" -Force |

כתבו לתוך הקובץ את שם החיה האהובה עליכם.

|  |
| --- |
| Set-Content -Value "Red Panda" -Path "C:\Course\7" |

הדפיסו את תוכן הקובץ.

|  |
| --- |
| Get-Content -Path "C:\Course\7" |

**משתנים**

כמו בכל שפה, גם בPowerShell ניתן לשמור משתנים. הדרך לשמור משתנה היא באמצעות סימן ה-$. לדוגמה - $var = ‘value’ שומר את המחרוזת ‘value’ בתוך המשתנה בשם var. נוכל בPS לשמור מידע מסוגים שונים – מספרים, מחרוזות, מערכים וכו'.

**קלט ופלט**

רעיון שחוזר עליו בכל שפת תכנות הוא קלט ופלט.  
הרעיון הוא - "אנחנו רוצים שהסקריפט יקבל מידע מסוים, יעבוד עליו ויעשה אתו משהו, ולבסוף שיוציא את התוצרים כפלט". כדי לקבל מידע מהמשתמש זמינות לנו 2 פקודות עיקריות בPowerShell, Read- Host (קלט) ו- Write-Host (פלט). פקודת Read-Host מאפשרת לנו לקבל קלט מהמשתמש המריץ את הסקריפט. הערך שהמשתמש יכניס יודפס למסך. שימוש לדוגמה:

Read-Host -prompt “Enter the name of the best animal in the world”

בהרצה המשתמש יקבל את ההודעה שכתובה בפרמטר prompt ויכניס קלט. כשאנחנו אומרים קלט במקרה הזה, אנחנו מתכוונים למה שהמשתמש המריץ את הסקריפט יזין.  
בדרך כלל נרשום בפרמטר prompt משפט אינפורמטיבי ונוח, המסביר למשתמש בדיוק מה עליו להזין.

אם נריץ את הפקודה Read-Host באופן הסטנדרטי כפי שהגדרנו בדוגמה, הפקודה פשוט תפלוט את קלט המשתמש למסך, אבל לרוב נרצה לשמור את המידע הזה לצורך שימוש עתידי בסקריפט, נשתמש כמובן במשתנה:

$var = Read-Host -prompt “Enter the name of the best animal in the world”

כעת הקלט יהיה שמור במשתנה var.

פקודת Write-Host מאפשרת לנו להוציא פלט למשתמש. כשאנחנו מדברים על פלט, אנחנו מדברים על מידע שהתוכנית מוציאה למשתמש.

דוגמה: Write-Host -Object “Tate is the best”  
ידפיס למסך: “Tate is the best”  
ואם נכתוב את הפקודה הבאה:  
Write-Host –Object $var  
יודפס לנו התוכן של המשתנה var, שהוא כאמור הקלט שקיבלנו מפקודת Read-Host למעלה.  
  
קבלו מהמשתמש שם של צבע והכניסו אותו למשתנה בשם message\_color

|  |
| --- |
| $message\_color = Read-Host -Prompt "enter favorite color: " |

השתמשו בפקודה write-host לכתוב חזרה למשתמש את הצבע שבחר

|  |
| --- |
| Write-Host -Object $message\_color |

**אקסטרה**: כתבו למשתמש את הצבע שבחר, שנו את צבע הפלט לאותו הצבע.

|  |
| --- |
| Write-Host -Object $message\_color -ForegroundColor Black |

# נספח א' – עדכון Get-Help

במידה וכאשר הרצתם את הפקודה Get-Help Get-Help קיבלתם את ההודעה הבאה:  


עליכם לעדכן את הפקודה באמצעות הפקודה:

 Update-Help -Verbose -Force -ErrorAction SilentlyContinue

שימו לב, הפעולה לוקחת מספר דקות, המתינו בסבלנות.

לאחר שפעולת הפקודה הסתיימה (רואים שוב את ה-prompt עם הנתיב הנוכחי), בצעו את הפקודה Get-Help בשנית.

אם זה עדיין לא עובד לכם, השתמשו בפרמטר -Online בסוף הפקודה Get-Help על מנת לפתוח את עמוד הhelp באינטרנט.

למשל: 