

PowerShell — inwentaryzacja sprzętu skryptem

Czas: 2 × 45 min

praca indywidualna

Cel: Napisanie skryptu PowerShell, który zbiera dane o sprzęcie stanowiska i zapisuje je jako raport CSV i HTML, oraz porównanie raportów komputera stacjonarnego i laptopa.

Po zajęciach potrafisz:

- odczytuje dane sprzętu poleceniami Get-CimInstance, Get-ComputerInfo i Get-PhysicalDisk
- wybiera i przelicza właściwości obiektów (Select-Object, właściwości obliczane)
- eksportuje wyniki do CSV i HTML
- bezpiecznie uruchamia skrypt .ps1 z ustawieniem zasad wykonywania tylko dla bieżącej sesji
- porównuje konfiguracje dwóch komputerów na podstawie raportów

Sprzęt i materiały

- komputer stacjonarny z Windows 11
- laptop szkolny z Windows 11 (bez otwierania obudowy)
- pendrive (przeniesienie skryptu i raportów)

Oprogramowanie

- **Windows PowerShell 5.1 (wbudowany w Windows 11)** — uruchamiaj jako zwykły użytkownik; nie wymaga administratora
- **Visual Studio Code z rozszerzeniem PowerShell (opcjonalnie)** — wygodny edytor; wystarczy też Notatnik lub PowerShell ISE
code.visualstudio.com/download
- **HWiNFO (portable)** — kontrola poprawności danych ze skryptu
hwinfo.com/download

Przygotowanie przed zajęciami

- Przypomnij sobie z zadania 2 (rok 1), jakie dane zapisuje się w karcie inwentaryzacyjnej komputera.

Bezpieczeństwo

- Nie zmieniaj zasad wykonywania skryptów dla całego komputera (`-Scope LocalMachine`). Używaj wyłącznie `-Scope Process` — ustawienie znika po zamknięciu okna.
- Uruchamiaj tylko skrypty, które napisałeś sam lub których treść przeczytałeś i rozumiesz. Plików .ps1 z internetu nie odblokowuj bez sprawdzenia.
- Raport zawiera numery seryjne sprzętu szkoły — przechowuj go na pendrivie lub udziale szkolnym, nie publikuj w internecie.

Przebieg zadania

1 Polecenia w konsoli

Najpierw sprawdź polecenia ręcznie, potem przeniesiesz je do skryptu.

1. Uruchom PowerShell i wpisz `Get-ComputerInfo -Property CsName, OsName, OsVersion, CsManufacturer, CsModel`.
2. Odczytaj procesor: `Get-CimInstance Win32_Processor | Select-Object Name, NumberOfCores, NumberOfLogicalProcessors, MaxClockSpeed`.
3. Odczytaj pamięć: `Get-CimInstance Win32_PhysicalMemory | Select-Object BankLabel, Manufacturer, PartNumber, @{n='GB';e={$_.Capacity/1GB}}, ConfiguredClockSpeed`.
4. Odczytaj dyski i BIOS: `Get-CimInstance Win32_DiskDrive | Select-Object Model, SerialNumber, @{n='GB';e={[math]::Round($_.Size/1GB)}} , Get-PhysicalDisk | Select-Object FriendlyName, MediaType, BusType, HealthStatus` oraz `Get-CimInstance Win32_BIOS | Select-Object Manufacturer, SMBIOSBIOSVersion, ReleaseDate, SerialNumber`.
5. Sprawdź, jakie inne właściwości ma obiekt: `Get-CimInstance Win32_Processor | Get-Member -MemberType Property`. Wybierz jedną dodatkową, przydatną właściwość.

2 Skrypt raport.ps1

Zapisz skrypt na pendrivie jako `raport.ps1`.

1. Na początku skryptu ustaw folder i nazwę raportu: `$nazwa = $env:COMPUTERNAME` i `$folder = Join-Path $PSScriptRoot 'raporty'`, a następnie utwórz folder: `New-Item -ItemType Directory -Path $folder -Force | Out-Null`.
2. Zapisz wyniki poleceń z kroku 1 do zmiennych (`$cpu`, `$ram`, `$dyski`, `$bios`, `$os`).
3. Wyeksportuj listę modułów RAM i dysków do CSV: `$ram | Export-Csv "$folder\$nazwa-ram.csv" -NoTypeInformation -Delimiter ';' -Encoding UTF8` (analogicznie dla dysków).
4. Zbuduj raport HTML z fragmentów: `$html = @($os, $cpu, $ram, $dyski, $bios) | ForEach-Object { $_ | ConvertTo-Html -Fragment }, potem ConvertTo-Html -Title "Raport $nazwa" -PreContent "<h1>$nazwa</h1><p>$(Get-Date)</p>" -PostContent $html | Out-File "$folder\$nazwa.html" -Encoding UTF8.`
5. Dodaj nagłówki sekcji (`-PreContent '<h2>Procesor</h2>'` przy każdym fragmencie) i komentarze `#` opisujące każdą część skryptu.

3 Uruchomienie i kontrola

1. Sprawdź bieżące zasady: `Get-ExecutionPolicy -List`. Zapisz wynik.
2. Ustaw zasady tylko dla tej sesji: `Set-ExecutionPolicy -Scope Process -ExecutionPolicy RemoteSigned` i uruchom skrypt: `.\raport.ps1`. Jeśli plik pochodzi z innego komputera i jest blokowany, sprawdź jego treść i użyj `Unblock-File .\raport.ps1`.
3. Otwórz raport HTML w przeglądarce, a CSV w arkuszu. Porównaj trzy wartości (procesor, RAM, model dysku) z HWiNFO.

4 Laptop i porównanie

1. Uruchom ten sam skrypt na laptopie (z pendrive'a, w ten sam sposób).
2. Porównaj raporty w tabeli: procesor, pamięć (liczba modułów, pojemność, taktowanie), typ i magistrala dysku (`MediaType`, `BusType`), wersja BIOS.
3. Zapisz, których danych skrypt nie odczytał (puste pola) i dlaczego mogło tak się stać.

W sprawozdaniu umieść

- ☐ pełny kod skryptu `raport.ps1` z komentarzami
- ☐ zrzuty raportu HTML ze stacjonarki i laptopa

- ☐ tabela porównawcza obu komputerów
- ☐ porównanie z HWiNFO i opis pustych lub błędnych pól
- ☐ odpowiedzi na pytania kontrolne

Pytania kontrolne

1. Czym różni się `Format-Table` od `Select-Object` i dlaczego przed `Export-Csv` nie używa się `Format-Table`?
2. Co oznacza ustawienie `RemoteSigned` i dlaczego zmieniamy je tylko z `-Scope Process`?
3. Dlaczego pojemność pamięci dzielimy przez `1GB` i co by się stało bez tego przeliczenia?
4. Do czego w serwisie lub w firmie przydaje się automatyczny raport stanowiska zamiast ręcznej karty?

Kryteria oceny

Kryterium	Punkty
Działający skrypt tworzący CSV i HTML	7
Czytelny kod z komentarzami i sekcjami raportu	3
Raport z laptopa i tabela porównawcza	4
Kontrola danych z HWiNFO i analiza braków	3
Odpowiedzi na pytania kontrolne	3
Razem	20

Sprawozdanie przygotuj na wzorze dołączonym do zadania (plik .docx o tym samym numerze).