Windows Sandbox

Przygotowanie skryptów pod konkretne wymagania, wiążą się z potrzebą dokładnego sprawdzenia ich działania. W większości przypadków skrypty te wykonują pewne operacje na systemach Windows. Ich testowanie na stacji wiązałoby się z potrzebą nie jednej reinstalacji systemu. Rozwiązanie, które przechodzi pierwsze na myśl to oczywiście Hyper-V i wirtualna maszyna, ale czy jedyne?
Poniżej, pokaże Ci funkcjonalność Windows Sandbox w "dziesiątce", która może być świetną alternatywą dla tego typu potrzeb (ale nie tylko).
Czym jest Windows Sandbox?
Windows Sandbox to rozwiązanie umożliwiające w szybki sposób uruchomienie zwirtualizowanego środowiska na swojej stacji, które co prawda użyje lokalnego systemu operacyjnego, ale w izolowany sposób. Dzięki temu podejrzane pliki, szkodliwe aplikacje, wadliwe skrypty nie będą miały wpływu na system.
Po zamknięciu piaskownicy całe oprogramowanie wraz ze wszystkimi plikami i stanem zostanie trwale usunięte. Jest to szybszy i prostszy sposób niż konfiguracja Hyper-V i wirtualizacja systemu.
Dodatkowo co jest zaletą, piaskownica zajmie tylko około 100MB przestrzeni dyskowej.

A. Instalacja Windows Sandbox

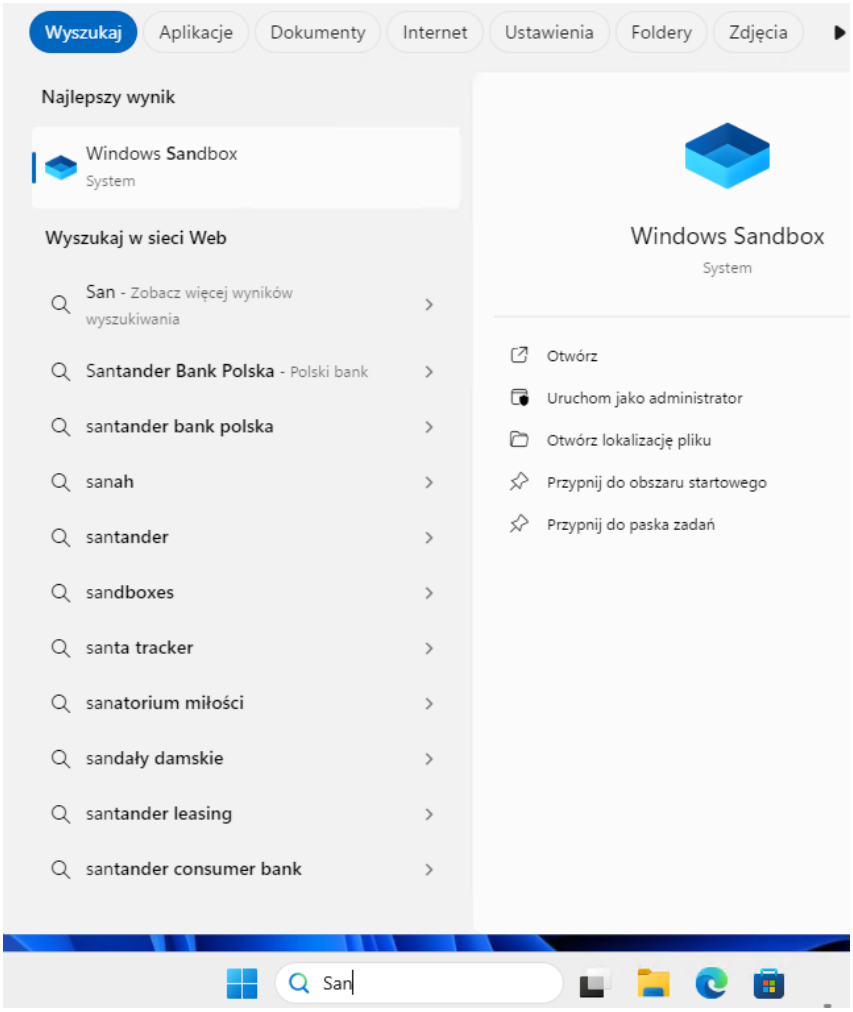
Przed instalacja spójrz na wymagania wstępne:
Co najmniej Windows 10 Pro lub Enterprise Insider build 18342 lub nowszy
Architektura 64-bitowa
Funkcje wirtualizacji włączone w systemie BIOS
Co najmniej 4GB RAM (rekomendowane 8GB)
Co najmniej 1 GB wolnej przestrzenidyskowej (rekomendowany dysk SSD)
Co najmniej 2 rdzenie CPU (rekomendowane 4 rdzenie z hyperthreadingiem)
Jeżeli chcesz wykonać kolejne czynność w maszynie wirtualnej Hyper-V to musisz włączyć wirtualizację zagnieżdżoną (jeśli będziesz je wykonywał w szkole to jest ona włączona):

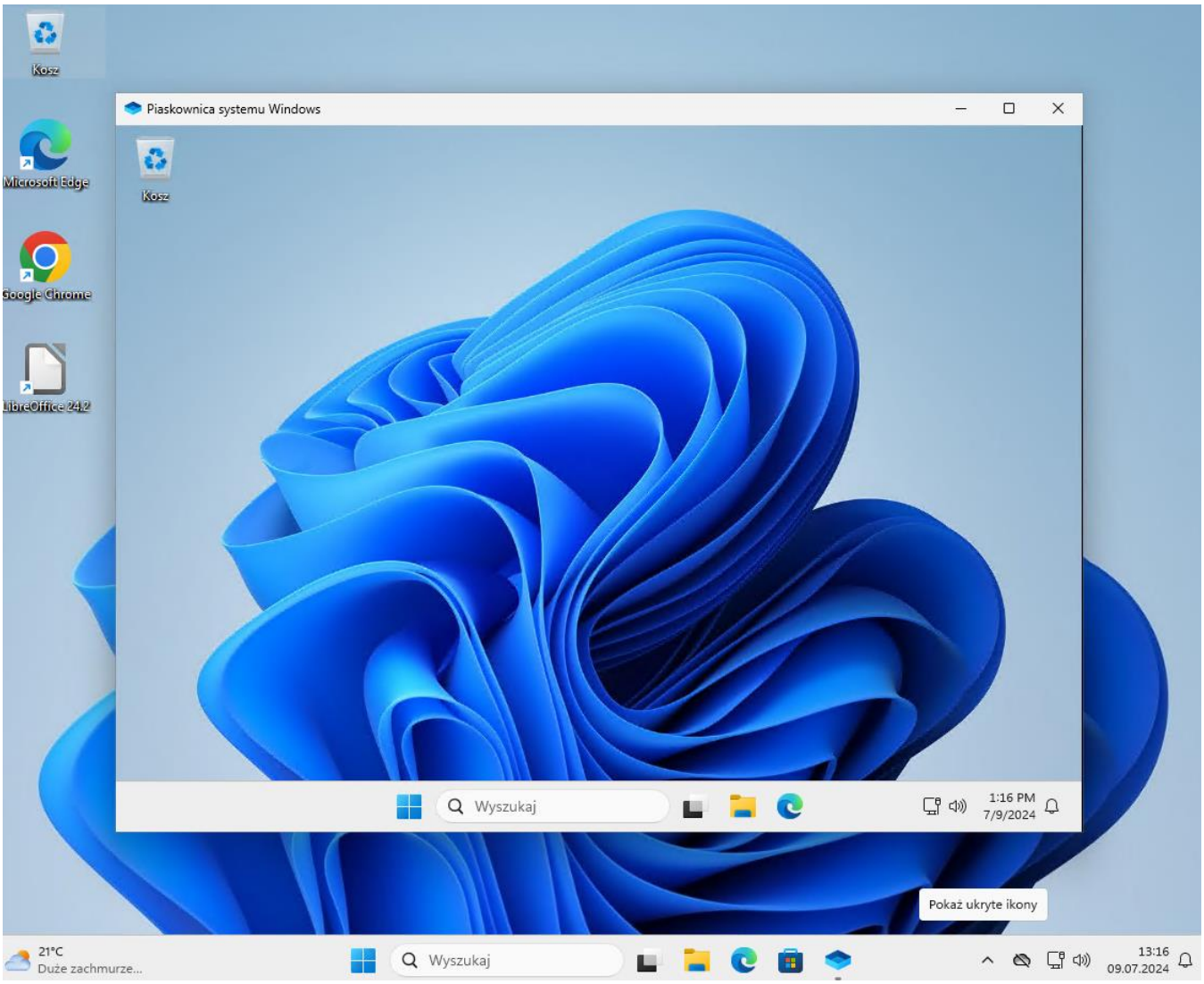
```
Set-VMProcessor -VMName "nazwa sytemu" -ExposeVirtualizationExtensions $true -Count 2
```

Jeśli sprawdziłeś i nie ma przeciwskazań to wykonaj poniższe polecenie PowerShell:

```
if([System.Environment]::OSVersion.Version.Build -gt 18305)
{
    Enable-WindowsOptionalFeature -FeatureName "Containers-DisposableClientVM" -Online -
ErrorAction Stop -NoRestart
}
Restart-Computer -Force
```

Powyższy skrypt zadziała, jeśli masz Windows 11 z numerem wersji Build większym niż 18305.
Komenda Enable-WindowsOptionalFeature włączy funkcję Windows Sandbox o nazwie Containers-DisposableClientVM. Po uruchomieniu tej komendy jest wymagane ponowne uruchomienie komputera.
Startujemy z Windows Sandbox
Po ponownym uruchomieniu, piaskownica jest gotowa do działania. Windows Sandbox znajdziesz w menu lub w konsoli PowerShell wykonaj WindowsSandbox.exe.


Wyszukaj
Aplikacje
Dokumenty
Internet
Ustawienia
Foldery
Zdjęcia
Najlepszy wynik
Windows Sandbox
System
Wyszukaj w sieci Web
San - Zobacz więcej wyników wyszukiwania
Santander Bank Polska - Polski bank
santander bank polska
sanah
santander
sandboxes
santa tracker
sanatorium miłości
sandały damskie
santander leasing
santander consumer bank
Windows Sandbox
System
Otwórz
Uruchom jako administrator
Otwórz lokalizację pliku
Przypnij do obszaru startowego
Przypnij do paska zadań
San



B. Personalizacja Windows Sandbox

Standardowo piaskownica za każdym razem uruchamia się czysta jako nowa instalacja systemu Windows. Jednak nie musi tak być, za pomocą plików konfiguracyjnych możesz wpływać na ustawienia w 4 zakresach:

1. vGPU (zwirtualizowany procesor graficzny)

Włącz lub wyłącz zwirtualizowany procesor GPU. Jeśli vGPU jest wyłączone, Sandbox użyje WARP (rasteryzatora oprogramowania).

2. Sieć

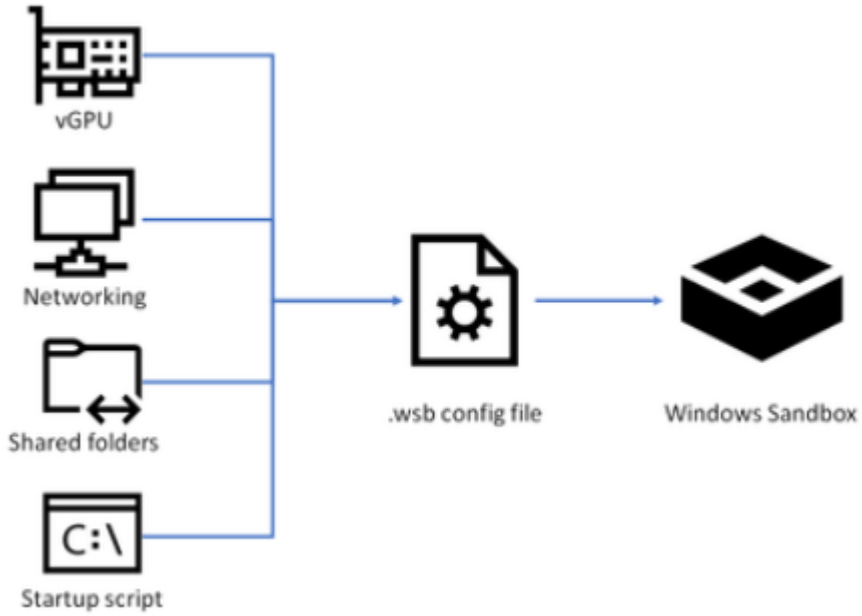
Włącz lub wyłącz dostęp sieciowy do piaskownicy.

3. Udostępnione foldery

Udostępniaj foldery z hosta z uprawnieniami do odczytu lub zapisu. Należy pamiętać, że ujawnienie katalogów hostów może pozwolić złośliwemu oprogramowaniu na wpłynięcie na system lub kradzież danych (widoczne na pulpicie – C:\Users\WDAGUtilityAccount\Desktop).

4. Skrypt startowy

Akcja logowania do piaskownicy.



Pliki konfiguracyjne Windows Sandbox są sformatowane jako XML i używają rozszerzenia pliku .wsb.
Przetestuj przykładowe pliki konfiguracyjne.
Przykładowy plik konfiguracyjny - 0.
Poniżej przykład pliku konfiguracyjnego .wsb, który przygotowuję od razu piaskownicę do pracy z Visual Studio Code.
Dzięki takiej konfiguracji folderów mam dostęp (tylko odczyt) do niezbędnych zasobów ze skryptami oraz dodatkowy katalog Out(zapis), który jest moim miejscem pracy w VSCode.

Przetestuj przykładowy plik konfiguracyjny - 1.
Poniższego pliku konfiguracyjnego można użyć do łatwego testowania pobranych plików w piaskownicy. W tym celu skrypt wyłącza sieć i vGPU, a w kontenerze ogranicza udostępniony folder Pobrania do dostępu tylko do odczytu. Dla wygody polecenie logowania otwiera folder pobierania w kontenerze podczas uruchamiania.
Utwórz Downloads1.wsb

```
<Configuration>
  <vGpu>Disable</vGpu>
  <Networking>Disable</Networking>
  <MappedFolders>
    <MappedFolder>
      <HostFolder>C:\Users\Public\Downloads</HostFolder>
      <SandboxFolder>C:\Users\WDAGUtilityAccount\Downloads</SandboxFolder>
      <ReadOnly>true</ReadOnly>
    </MappedFolder>
  </MappedFolders>
  <LogonCommand>
    <Command>explorer.exe C:\Users\WDAGUtilityAccount\Downloads</Command>
  </LogonCommand>
</Configuration>
```

Zgłoszenie 1
Przywróć pierwszy punk kontrolny wykonaj instalacje Windows Sandbox

Przetestuj przykładowy plik konfiguracyjny - 2.

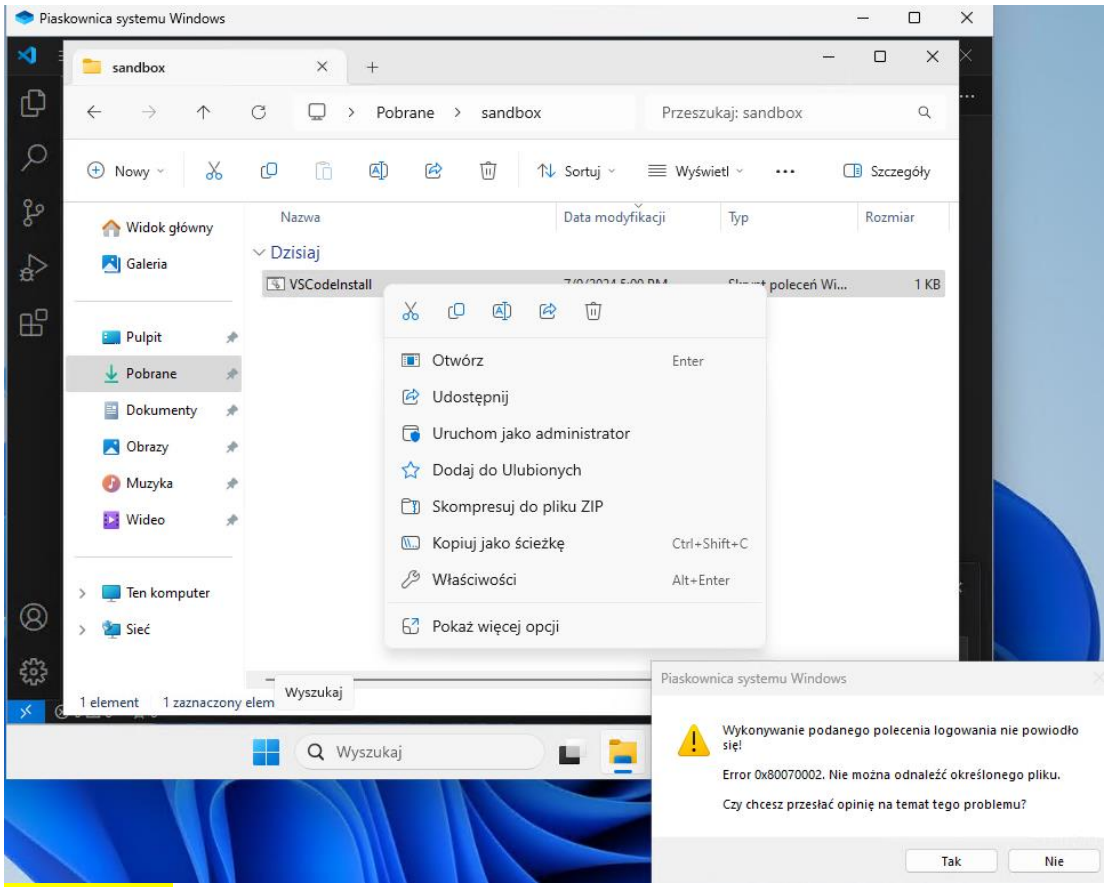
Poniższy plik konfiguracyjny instaluje kod Visual Studio w kontenerze, co wymaga nieco bardziej skomplikowanej konfiguracji LogonCommand.
W C:\ są dwa foldery; pierwszy (SandboxScripts) zawiera VSCodeInstall.cmd, który instaluje i uruchamia VSCode. Zakłada się, że drugi folder (CodingProjects) zawiera pliki projektu, które deweloper chce zmodyfikować za pomocą VSCode.
Za pomocą skryptu instalatora VSCode, który jest już zmapowany do kontenera, LogonCommand może się do niego odwoływać.
Utwórz VSCodeInstall.cmd

```
REM Download Visual Studio Code
curl -L "https://update.code.visualstudio.com/latest/win32-x64-user/stable" --output
C:\Users\WDAGUtilityAccount\Downloads\vscode.exe
REM Install and run Visual Studio Code
C:\Users\WDAGUtilityAccount\Downloads\vscode.exe /verysilent /suppressmsgboxes
```

Utwórz VSCode2.wsb

```
<Configuration>
  <MappedFolders>
    <MappedFolder>
      <HostFolder>C:\SandboxScripts</HostFolder>
      <SandboxFolder>C:\Users\WDAGUtilityAccount\Downloads\sandbox</SandboxFolder>
      <ReadOnly>true</ReadOnly>
    </MappedFolder>
    <MappedFolder>
      <HostFolder>C:\CodingProjects</HostFolder>
      <SandboxFolder>C:\Users\WDAGUtilityAccount\Documents\Projects</SandboxFolder>
      <ReadOnly>false</ReadOnly>
    </MappedFolder>
  </MappedFolders>
  <LogonCommand>
    <Command>C:\SandboxScripts\VSCodeInstall.cmd</Command>
  </LogonCommand>
</Configuration>
```

Teraz wystarczy dwukrotnie kliknąć VSCode2.wsb i uruchomić Windows Sandbox. Zaletą tego jest to, że można utworzyć kilka konfiguracji dla sposobu działania piaskownicy. Właściwie całkiem praktyczne. Można mieć tylko nadzieję, że Microsoft z czasem rozszerzy możliwości pliku konfiguracyjnego. Może być konieczne odsunięcie komunikatu piaskownicy i ręczne uruchomienie pliku cmd



Zgłoszenie 2

Przywróć pierwszy punk kontrolny wykonaj instalacje Windows Sandbox

Przetestuj przykładowy plik konfiguracyjny - 3

Utwórz skrypt PowerShell korzystając z poniższego kodu i zapisz go w katalogu C:\sandbox jako SwapMouse3.ps1.

```
[Reflection.Assembly]::LoadWithPartialName("System.Windows.Forms") | Out-Null
$SwapButtons = Add-Type -MemberDefinition @'
[DllImport("user32.dll")]
public static extern bool SwapMouseButton(bool swap);
'@ -Name "NativeMethods" -Namespace "PInvoke" -PassThru
$SwapButtons::SwapMouseButton(!([System.Windows.Forms.SystemInformation]::MouseButtonsSwapped))
```

Utwórz SwapMouse3.wsb

```
<Configuration>
  <MappedFolders>
    <MappedFolder>
      <HostFolder>C:\sandbox</HostFolder>
      <SandboxFolder>C:\sandbox</SandboxFolder>
      <ReadOnly>True</ReadOnly>
    </MappedFolder>
  </MappedFolders>
  <LogonCommand>
    <Command>powershell.exe -ExecutionPolicy Bypass -File C:\sandbox\SwapMouse3.ps1</Command>
  </LogonCommand>
```

```
</Configuration>
```
Wyjaśnienie:
SwapMouse.ps1: Ten skrypt PowerShell zmienia podstawowy przycisk myszy, aby dostosować go dla użytkowników leworęcznych. Wczytuje niezbędne biblioteki, definiuje funkcję SwapMouseButton z biblioteki user32.dll i wywołuje tę funkcję, aby zamienić przyciski myszy.
SwapMouse.wsb: Jest to plik konfiguracyjny Windows Sandbox, który mapuje katalog C:\sandbox na hoście do katalogu C:\sandbox w sandboxie jako tylko do odczytu. Po zalogowaniu się do sandboxa uruchamia skrypt SwapMouse.ps1 za pomocą PowerShell z pominięciem polityki wykonywania skryptów.
Użycie:
Stwórz katalog C:\sandbox na komputerze lokalnym i umieść tam skrypt SwapMouse.ps1.
Utwórz plik SwapMouse.wsb z powyższą zawartością i zapisz go w dowolnym miejscu.
Uruchom plik SwapMouse.wsb poprzez dwukrotne kliknięcie, co spowoduje uruchomienie Windows Sandbox z zmapowanym katalogiem i wykonanie skryptu przy logowaniu.
To rozwiązanie automatycznie skonfiguruje przyciski myszy w środowisku Windows Sandbox dla leworęcznych użytkowników.
Zgłoszenie 3
Przywróć pierwszy punk kontrolny wykonaj instalacje Windows Sandbox

C. Instalacja Visual Studio Code w Windows Sandbox
Napisz skrypt PowerShell, który instaluje Visual Studio Code, przenosi ustawienia oraz wybrane rozszerzenia z lokalnego hosta w Windows Sandbox, należy uwzględnić specyfikę środowiska Sandbox.
Skrypt PowerShell do instalacji Visual Studio Code w Windows Sandbox oraz przenoszenia ustawień i rozszerzeń z lokalnego hosta.
Ustawienia ścieżek i listy rozszerzeń
```
$sourceSettingsPath = "C:\users\WDAGUtilityAccount\Desktop\User"
$destinationSettingsPath = "C:\users\WDAGUtilityAccount\AppData\Roaming\Code\User"
$sourceExtensionsPath = "C:\users\WDAGUtilityAccount\Desktop\extensions"
$destinationExtensionsPath = "C:\users\WDAGUtilityAccount\.vscode\extensions"
$extensionsList = @("ms-vscode.powershell-2020", "coenraads.bracket-pair-colorizer")
```
Zmiana polityki wykonywania skryptów
```
Set-ExecutionPolicy Bypass -Scope Process -Force
```
Instalacja Chocolatey
```
[System.Net.ServicePointManager]::SecurityProtocol =
[System.Net.ServicePointManager]::SecurityProtocol -bor 3072
Invoke-Expression ((New-Object
System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))
```
Instalacja Visual Studio Code
```
choco install vscode -y
```
Przenoszenie ustawień
```
if (Test-Path $sourceSettingsPath) {
    New-Item -ItemType Directory -Path $destinationSettingsPath -Force
    Copy-Item "$sourceSettingsPath\*" -Destination $destinationSettingsPath -Recurse -Force
} else {
    Write-Host "Ścieżka źródłowa dla ustawień nie istnieje: $sourceSettingsPath"
}
```
Przenoszenie rozszerzeń
```
if (Test-Path $sourceExtensionsPath) {
    foreach ($ext in $extensionsList) {
        $extension = Get-ChildItem $sourceExtensionsPath | Where-Object { $_.Name -like "*$ext*" }
```

```
        if ($extension) {
            Copy-Item -Path $extension.FullName -Destination
"$destinationExtensionsPath\$($extension.BaseName)" -Recurse -Force
        } else {
            Write-Host "Rozszerzenie $ext nie znaleziono w ścieżce źródłowej."
        }
    }
} else {
    Write-Host "Ścieżka źródłowa dla rozszerzeń nie istnieje: $sourceExtensionsPath"
}
```

Uruchomienie Visual Studio Code

```
Start-Process "C:\Program Files\Microsoft VS Code\Code.exe" -ArgumentList
"C:\users\WDAGUtilityAccount\Desktop\out"
```

Wyjaśnienie skryptu:

1. Ustawienia ścieżek i listy rozszerzeń:

sourceSettingsPath: Ścieżka źródłowa dla ustawień VS Code.
destinationSettingsPath: Ścieżka docelowa dla ustawień VS Code.
sourceExtensionsPath: Ścieżka źródłowa dla rozszerzeń VS Code.
destinationExtensionsPath: Ścieżka docelowa dla rozszerzeń VS Code.
extensionsList: Lista rozszerzeń do przeniesienia.

2. Zmiana polityki wykonywania skryptów:

Umożliwia wykonanie skryptów PowerShell, co jest wymagane do instalacji Chocolatey.

3. Instalacja Chocolatey:

Ustawia protokół TLS i pobiera skrypt instalacyjny Chocolatey, a następnie go wykonuje.

4. Instalacja Visual Studio Code:

Używa Chocolatey do instalacji VS Code.

5. Przenoszenie ustawień:

Sprawdza czy istnieje ścieżka źródłowa, tworzy ścieżkę docelową, a następnie kopiuje pliki.

6. Przenoszenie rozszerzeń:

Sprawdza czy istnieje ścieżka źródłowa, przeszukuje ją pod kątem rozszerzeń z listy i kopiuje znalezione rozszerzenia do ścieżki docelowej.

7. Uruchomienie Visual Studio Code:

Uruchamia VS Code z określonymi argumentami.

Jak używać:

1. Utwórz plik .ps1:

Skopiuj powyższy skrypt do pliku .ps1, np. install_vscode.ps1.

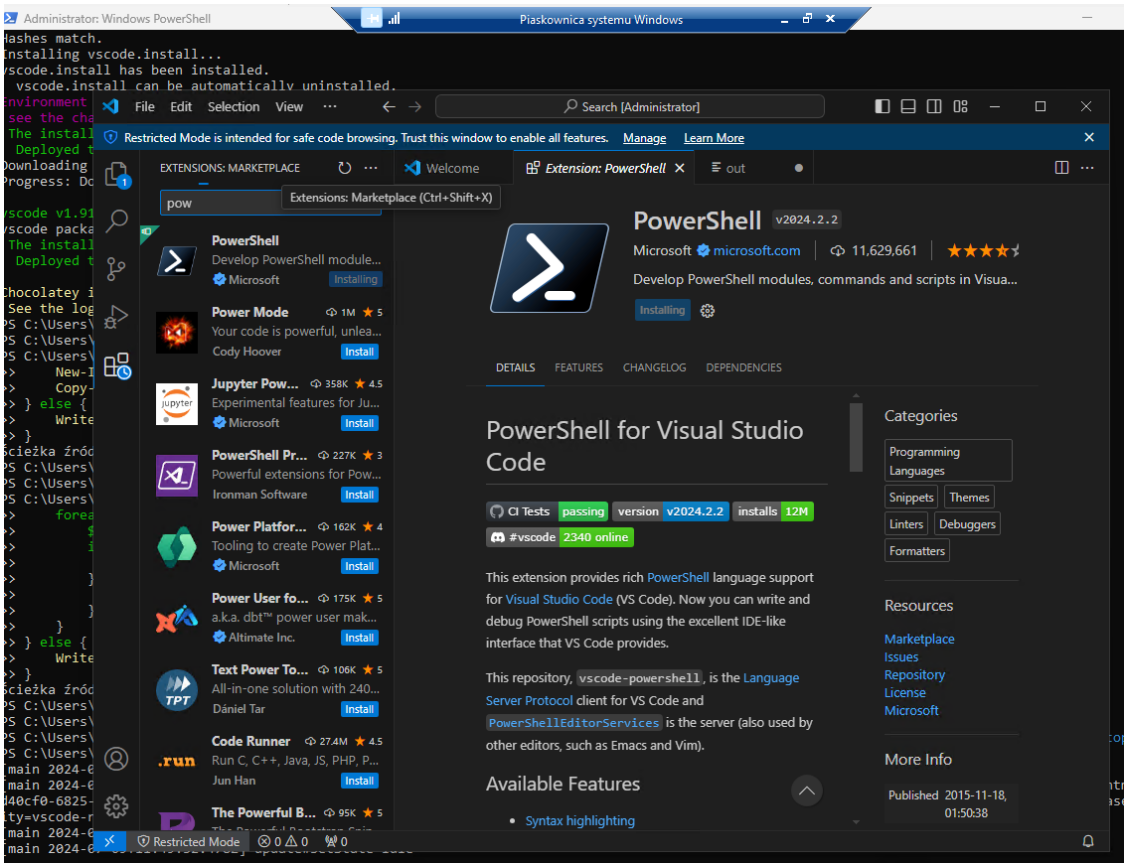
2. Uruchom skrypt w Windows Sandbox:

Skrypt musi być uruchomiony w Windows Sandbox z odpowiednimi uprawnieniami administracyjnymi.

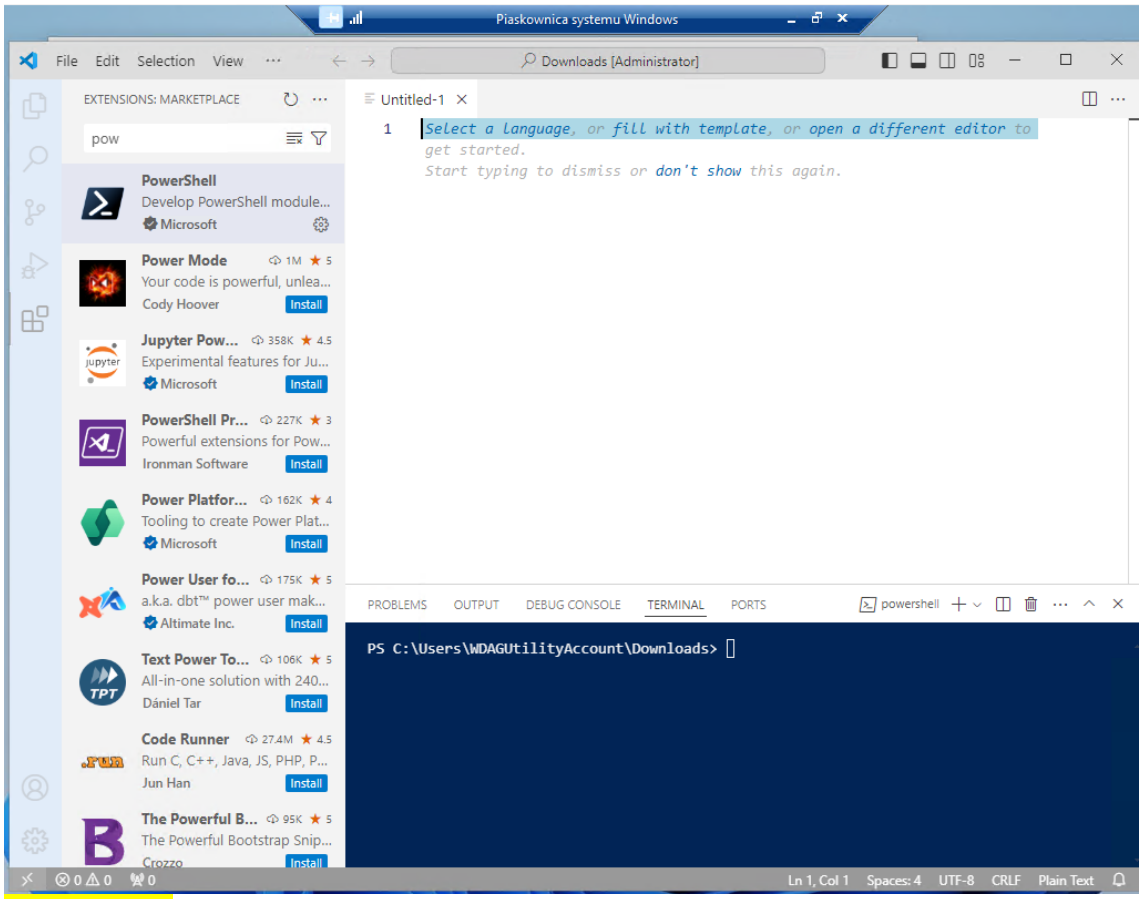
Uwaga:

Powyższy skrypt zakłada, że pliki ustawień i rozszerzeń znajdują się na pulpicie konta WDAGUtilityAccount w Windows Sandbox. Upewnij się, że te pliki są dostępne w odpowiednich lokalizacjach przed uruchomieniem skryptu.

Wykonaj instalację PowerShell w VSC uruchamiającą powyższy skrypt:



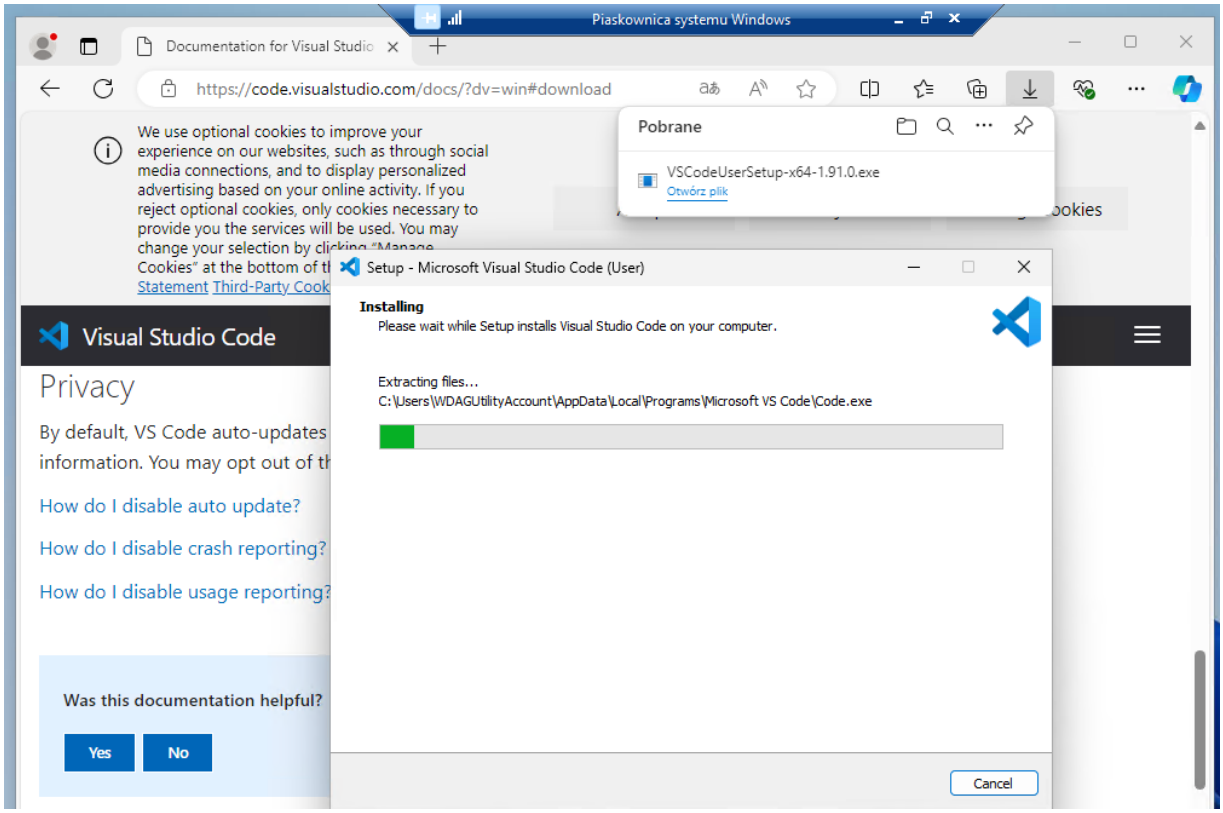
Efekt:

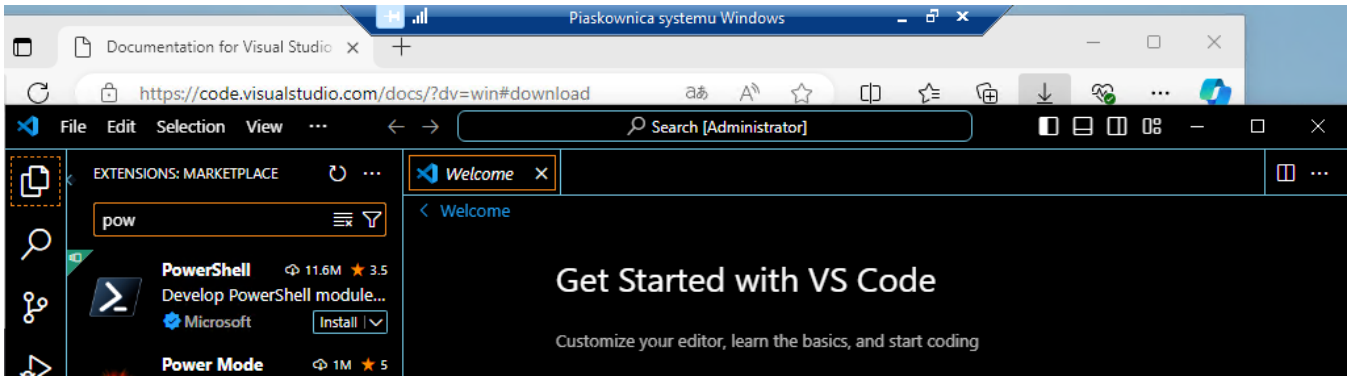


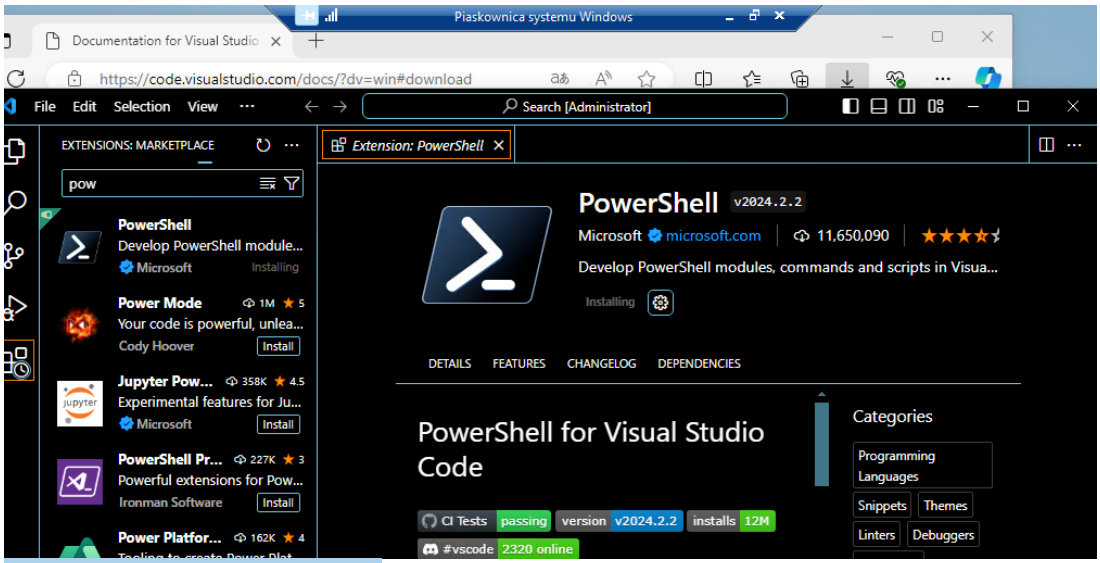
Zgłoszenie 4

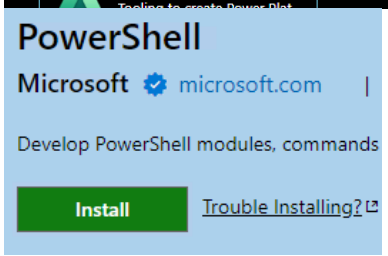
Przywróć pierwszy punk kontrolny wykonaj instalacje Windows Sandbox

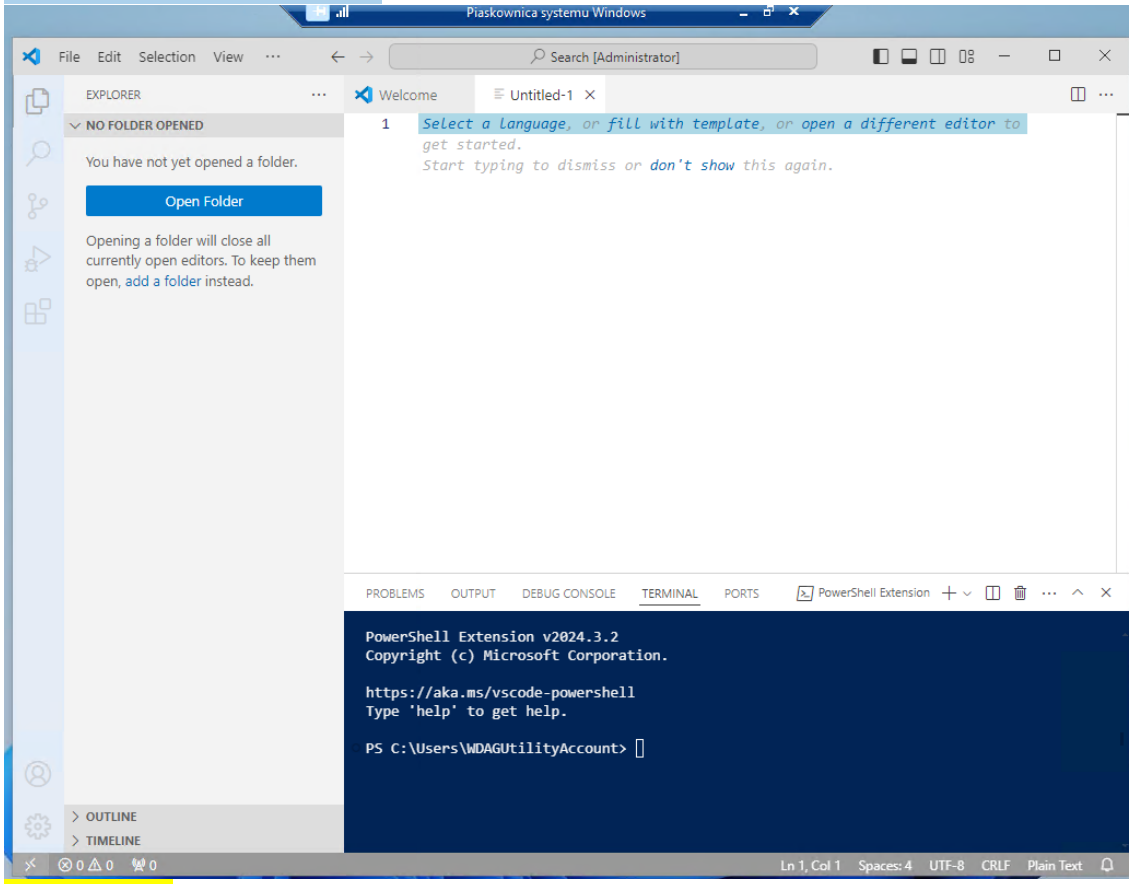
3. Instalacja w Windows Sandbox aplikacji Visual Studio Code z pobraniem z przeglądarki
Pobranie https://code.visualstudio.com/docs/?dv=win i zainstalowalnie
Dodanie rozszerzenia do PowerShell











Zgłoszenie 5

Przywróć pierwszy punk kontrolny.

Podaj wnioski z powyższych czynności.
Wnioski z wykonanych czynności:

1. **Windows Sandbox - Szybka i bezpieczna izolacja**:
    - Windows Sandbox oferuje szybkie uruchomienie izolowanego środowiska, idealnego do testowania skryptów, instalacji aplikacji oraz innych operacji, które mogą być potencjalnie niebezpieczne dla głównego systemu.
    - Po zamknięciu sandboxa, wszystkie dane i stan systemu są trwale usuwane, co zapewnia wysoki poziom bezpieczeństwa.
2. **Instalacja Windows Sandbox**:
    - Wymaga posiadania systemu co najmniej Windows 10 Pro lub Enterprise z odpowiednią wersją build.
    - Włączenie funkcji wirtualizacji w BIOSie oraz posiadanie odpowiednich zasobów sprzętowych jest niezbędne.
    - Instalacja jest prosta i obejmuje włączenie odpowiednich funkcji Windows za pomocą PowerShell.
3. **Konfiguracja Windows Sandbox**:
    - Możliwość personalizacji sandboxa poprzez pliki konfiguracyjne .wsb, które mogą określać ustawienia takie jak vGPU, dostęp do sieci, udostępnione foldery oraz skrypty startowe.
    - Przykłady plików konfiguracyjnych pokazują różne scenariusze użycia, takie jak testowanie plików z wyłączoną siecią, instalacja i uruchamianie aplikacji takich jak Visual Studio Code, czy dostosowywanie ustawień systemu.
4. **Przykładowe pliki konfiguracyjne**:
    - Przykłady plików .wsb pokazują, jak można przygotować sandboxa do specyficznych zadań, takich jak testowanie plików pobranych z internetu, instalacja i konfiguracja Visual Studio Code, czy zmiana ustawień systemu za pomocą skryptów PowerShell.
    - Przykład z instalacją VS Code obejmuje użycie skryptu .cmd do pobrania i instalacji aplikacji, a także mapowanie folderów między hostem a sandboxem w celu przenoszenia ustawień i rozszerzeń.
5. **Instalacja Visual Studio Code w Windows Sandbox**:
    - Skrypt PowerShell do instalacji VS Code w sandboxie uwzględnia specyfikę środowiska, takie jak zmiana polityki wykonywania skryptów, instalacja Chocolatey, kopiowanie ustawień i rozszerzeń oraz uruchamianie aplikacji.
    - Skrypt jest praktycznym przykładem, jak automatyzować proces instalacji i konfiguracji oprogramowania w izolowanym środowisku.

Podsumowanie:

Windows Sandbox jest praktycznym narzędziem do szybkiego i bezpiecznego testowania aplikacji, skryptów i innych operacji, które mogą być potencjalnie niebezpieczne dla głównego systemu. Dzięki łatwej instalacji, personalizacji za pomocą plików konfiguracyjnych oraz możliwości automatyzacji zadań za pomocą skryptów PowerShell, Sandbox staje się niezastąpionym narzędziem dla deweloperów, testerów oraz administratorów systemów.

Zgłoszenie 6
Przywróć pierwszy punk kontrolny.