Pliki wsadowe

Budowa pliku wsadowego

Plik wsadowy jest to zwykły plik tekstowy zawierający rozszerzenie .bat lub .cmd. Wewnątrz takiego pliku znajdują się instrukcje, które są poleceniami systemu i generalnie umożliwiają wykonanie jakiejś czynności. Polecenia wewnątrz pliku wsadowego muszą być umieszczane każda w osobnej linii. Mogą to być polecenia wewnętrzne, polecenia zewnętrzne (programy) lub polecenia specyficzne dla pliku wsadowego (polecenia, które są dozwolone tylko w tym pliku).

Przykład prostego pliku wsadowego

```
REM Plik Przykład.cmd

ECHO Blablablabla

ECHO [autoexec.cmd]

TYPE plik1.cmd | MORE

ECHO [config.sys]

TYPE plik2.cmd | MORE

ECHO [Dysk twardy C:]

ECHO Naciśnij jakiś klawisz...

PAUSE
```

Korzystając z plików wsadowych, zwanych również programami wsadowymi lub skryptami, można uprościć rutynowe lub wielokrotnie wykonywane zadania.

Plik wsadowy jest niesformatowanym plikiem tekstowym zawierającym jedno lub więcej poleceń. Gdy nazwa pliku zostanie wpisana w wierszu polecenia, program Cmd.exe wykonuje kolejne polecenia w pliku.

Nazwa pliku wsadowego ma rozszerzenie cmd lub bat. Kiedy nazwa pliku zostaje wpisana w wierszu polecenia lub kiedy program wsadowy zostaje uruchomiony z innego komputera, polecenia programu wsadowego są przetwarzane sekwencyjnie. Programy wsadowe są również określane jako pliki wsadowe.

Termin skrypt oznacza: typ programu składający się z listy rozkazów aplikacji lub programu.
Skrypt najczęściej wyraża polecenia przy użyciu reguł i składni aplikacji lub narzędzia oraz prostych struktur sterujących, takich jak pętle i wyrażenia warunkowe typu if/then.

W pliku wsadowym można umieścić dowolne polecenia. Określone polecenia, takie jak for, goto i if, umożliwiają warunkowe przetwarzanie poleceń w pliku wsadowym.

Pliki wsadowe uruchamia się tak jak zwykłe pliki wykonywalne z rozszerzeniem .exe.
Mogą one również posiadać argumenty przekazywane do wnętrza pliku. Argumenty te podajemy po nazwie takiego pliku np. aby przekazać plikowi wsadowemu argumenty arg1 i arg2 wystarczy napisać:

W:\>przyklad.cmd arg1 arg2

Polecenia wsadowe

ECHO ON (włączenie wyświetlania poleceń)

ECHO OFF (wyłączenie wyświetlania poleceń)

ECHO komunikat (wypisanie komunikatu)

ECHO. (wyprowadzi pustą linię)

ECHO %ZMIENNA% (wypisze zawartość zmiennej)

@ECHO OFF symbol @ wyłączenie komunikatów od bieżącej linii

PAUSE wstrzymanie wykonywania poleceń pliku BAT do momentu wciśnięcia dowolnego klawisza

CHCP [nnn] zmiana numeru aktualnej strony kodowej na nnn (wartość domyślna 852).

UWAGA: Aby znaki były poprawnie wyświetlane notatnik powinien stosować również właściwą stronę kodową. Dla wartości domyślnej (852) – zestaw znaków OEM852.

GOTO etykieta wykonanie skoku do linii oznaczonej etykieta; jeżeli będzie GOTO koniec, to przejdzie do linii zaczynającej się :koniec)

IF [NOT] "napis1"=="napis2" polecenie

IF [NOT] ERRORLEVEL liczba polecenie (spełnienie warunku)

IF [NOT] EXIST plik1 polecenie (jako napis może być np. %CONFIG% lub od %1 do %9)

SHIFT przesunięcie w lewo o jedną pozycję parametrów wywołania pliku wsadowego

CALL PLIK.cmd [parametry] (wywołanie pliku BAT z zapewnieniem powrotu do pliku macierzystego)

CHOICE [/C[:]klawisze] [/N] [/S] [/T[:]domyslny,czas] [tekst] (umożliwia dokonanie wyboru podczas wykonywania pliku BAT, wykorzystuje zmienną ERRORLEVEL

FOR %zmienna IN (zbior) DO polecenie (pętla FOR w linii komend)

FOR %I IN (*.*) DO FIND "ALA" %I (zbiór to pliki w katalogu bieżącym)

FOR %%I IN (P1.TXT P2.TXT P3.TXT) DO COPY %%I PRN (pętla FOR w pliku BAT - są dwa procenty przy zmiennej)

PARAMETRY WYWOŁANIA

Jeżeli plik BAT jest wywołany z parametrami, to wewnątrz tego pliku parametry są dostępne za pomocą zmiennych od %1 do %9. Zmienna %0 przechowuje nazwę danego pliku BAT.

ĆWICZENIA

1. Na dysku W: utwórz katalog zadanie4

2. Uruchom program Notepad++

3. Poszczególne ćwiczenia zapisuj pod nazwami: plik1.cmd, plik2.cmd, …..

4. Wpisuj polecenia dla każdego pliku

5. Uruchomienie pliku: w oknie cmd wpisz nazwę uruchamianego pliku np. plik1 dodając opcjonalnie parametry.

plik1.cmd

```
ECHO.
ECHO Witaj!!!
ECHO.
PAUSE
```

plik2.cmd

```
@ECHO OFF

ECHO.

ECHO Witaj!!!

ECHO.

PAUSE
```

plik3.cmd

Korzystając z poleceń ECHO opisz różnicę działania pliku plik1.cmd i plik2.cmd.
Po wyświetleniu informacji plik oczekuje na naciśnięcie dowolnego klawisza.

plik4.cmd

```
@ECHO OFF

REM Plik Przyklad.cmd

ECHO Blablablabla

ECHO [autoexec.cmd]

TYPE plik1.cmd | MORE

ECHO [config.sys]

TYPE plik2.cmd | MORE

ECHO [Dysk twardy C:]

ECHO Naciśnij jakiś klawisz...

PAUSE
```

plik5.cmd

Korzystając z poleceń ECHO opisz działanie pliku plik4.cmd

plik6.cmd

Zapisz plik plik4.cmd pod nazwą plik6.cmd – dodaj polecenia, tak aby polskie znaki były wyświetlane poprawnie.

Komunikat Aby kontynuować, naciśnij dowolny klawisz . . . zastąp własnym np. Naciśnij jakiś klawisz - to zakończy program :)

plik7.cmd

Wyświetl na ekranie informację o podstawowych zmiennych środowiskowych w postaci:

Informacje o systemie:

nazwa komputera - komputer

nazwa użytkownika - JanKowalski

katalog główny – C:\

katalog ROOT - C:\Windows

katalog tymczasowy - C:\Users\JanKowalski\.....

serwer logowania - \\serwer01

Naciśnij dowolny klawisz ...

plik8.cmd

Napisz skrypt proszący użytkownika o podanie imienia, a następnie wyświetl to imię na ekranie. Dodaj opcję oczekiwania na naciśnięcie klawisza. Po zakończeniu działania skryptu usuń tymczasowe zmienne środowiskowe.

```
@ECHO OFF
@chcp 852
SET /p nazwa="Jak się nazywasz -> "
echo.
echo.
```

```
echo WIEDZIAŁEM - nazywasz się: %nazwa%

echo.

echo.

set nazwa=

pause
```

plik9.cmd

Napisz skrypt podający sumę dwóch podanych przez użytkownika liczb. Po podaniu wyniku dodaj opcję oczekiwania na naciśnięcie klawisza. Po zakończeniu działania skryptu usuń tymczasowe zmienne środowiskowe

plik10.cmd

Rozbudowanie zadania plik9.cmd o polecenie GOTO

Napisz skrypt podający sumę dwóch podanych przez użytkownika liczb. Po podaniu wyniku skrypt prosi o podanie następnych liczb. Skrypt działa do momentu naciśnięcia klawiszy Ctrl+C
Po zakończeniu działania skryptu usuń tymczasowe zmienne środowiskowe.

```
@ECHO OFF

@chcp 852

:start

SET /p a="Podaj wartość A="

SET /p b="Podaj wartość B="

SET /a w=a+b

echo.

echo.

echo Wynik: %a% + %b% = %w%

echo.
```

```
echo.

set a=

set b=

set w=

goto start
```

Opis: Skrypt oczekuje na wprowadzenie danych A oraz B. Następnie instrukcją SET sumuje wartość obu liczb podstawiając wynik do zmiennej W. Instrukcja echo Wynik: %a% + %b% = %w% wyświetla wynik na ekranie. Następnie usuwane są tymczasowe zmienne środowiskowe. Poleceniem goto start „przeskakujemy” do etykiety :start. Skrypt działa w nieskończoność. Przerwać go możemy kombinacją klawiszy Ctrl+C

plik11.cmd

Instrukcja warunkowa IF warunek polecenie Skrypt prosi o podanie hasła i sprawdza jego poprawność:

```
1. @ECHO OFF

2. @chcp 852

3. set wzor=abc123

4. SET /p haslo="Podaj hasło: "

5. IF %haslo% == %wzor% GOTO ok

6. GOTO error

7. :ok

8. echo.

9. echo.

10. echo Podano poprawne hasło

11. goto koniec

12. :error
```

```
13. echo.

14. echo.

15. echo Błędne hasło !!!

16. :koniec

17. echo.

18. echo.

19. set haslo=

20. set wzor=

21. pause
```

Opis: W linii 3 wprowadzamy wzorcowe hasło. Skrypt oczekuje na wprowadzenie hasła – linia 4
W linii 5 porównuje oba ciągi – jeśli hasło jest poprawne przeskakujemy do etykiety OK.
Wyświetlamy komunikat o poprawności hasła przeskakujemy do etykiety koniec – kończąc program.
Jeśli w linii 5 hasło jest błędne celem jest etykieta :error w linii 12. W linii 19 usuwamy tymczasową zmienną środowiskową haslo.

plik12.cmd

Instrukcja wyboru CHOICE Skrypt skrypt11.cmd zapiszmy jako skrypt12.cmd i dodajmy na końcu następujące linie:

```
choice /m "Czy kontynuować?  "

if errorlevel 2 goto koniec

if errorlevel 1 goto start

:koniec
```

oraz po linii 2 etykietę `:start`

Opis: Dodana instrukcja oczekuje na naciśnięcie klawisza T lub N. Po naciśnięciu klawisza zmienna errorlevel zawiera wartość numeryczną odpowiadającą kolejności oczekiwanych klawiszy. Tu dla T przyjmuje wartość 1 natomiast dla N wartość 2

Domyślne klawisze [T,N] można dowolnie zmienić, pokazuje to następny przykład.

plik13.cmd

Instrukcja wyboru CHOICE – zmiana klawiszy wyboru

```
@ECHO OFF
CHOICE /C:ABCD /M "Wybierz: [A]nna [B]eata [C]elina [D]orota"
ECHO.
IF ERRORLEVEL 4 GOTO dorota
IF ERRORLEVEL 3 GOTO celina
IF ERRORLEVEL 2 GOTO beata
IF ERRORLEVEL 1 GOTO anna
:anna
ECHO Wybrałeś: Anna
GOTO koniec
:beata ECHO Wybrałeś: Beata
GOTO koniec
:celina ECHO Wybrałeś: Celina
GOTO koniec
:dorota ECHO Wybrałeś: Dorota
:koniec
ECHO.
ECHO ********** Koniec skryptu **********
```

plik14.cmd

Zapisz plik plik13.cmd jako plik14.cmd i dodaj instrukcje, tak aby skrypt pytał czy dokonać kolejnego wyboru.

plik15.cmd

Porównanie numeryczne. Do porównywania wartości numerycznych stosujemy następujące operatory:

EQU – równe

NEQ – nierówne

LSS – mniejsze niż

LEQ – mniejsze niż lub równe

GTR – większe niż

GEQ – większe niż lub równe

Skrypt oczekuje na wprowadzenie dwóch liczb i wyświetla informację o ich zależności.

```
@ECHO OFF
@chcp 852
:start
set /p a=Podaj A= set /p b=Podaj B=
if %a% GTR %b% echo A jest większe od B
if %a% EQU %b% echo A jest równe B
if %a% LSS %b% echo A jest mniejsze od B
choice /m "Czy kolejne dane? "
IF ERRORLEVEL 2 GOTO end
IF ERRORLEVEL 1 GOTO start
:end
echo.
echo ********** Koniec skryptu **********
```

plik16.cmd

Napisz skrypt, który będzie prosił o podanie liczby do odgadnięcia. Po wyczyszczeniu ekranu CLS będzie oczekiwał na wprowadzenie liczby i wyświetlał komunikat: za duża, za mała lub zgadłeś. Następnie zapyta czy gramy dalej?

plik17.cmd

Napisz skrypt, który:

1. Poprosi o nazwę użytkownika.

2. Wyświetli powitanie: O! Witaj [imię użytkownika]

3. Następnie sam się przedstawi: Jestem programem napisanym przez autor

4. Poda aktualną godzinę i datę

5. Spyta czy to prawda prosząc o wpisanie [tak] lub [nie]

6. Jeśli użytkownik wpisze coś innego poprosi grzecznie o wpisanie poprawnej odpowiedzi.

7. Jeśli użytkownik odpowie [nie] poinformuje go również grzecznie, że się myli bo on ma zawsze rację i zakończy konwersację.

8. Jeśli użytkownik odpowie [tak] pochwali użytkownika, przypomni, że ma zawsze rację, jeszcze raz poda aktualną godzinę i datę i pożegna się.

plik18.cmd

Napisz skrypt, który wyświetli następujące menu:

[1]. Kalkulator

[2]. Systemowy notatnik

[3]. Paint

[Q]. Zakończ

Wybierz program do uruchomienia: [1,2,3,Q]

Po naciśnięciu wybranego klawisza uruchomi właściwą aplikację systemową.

Nazwy aplikacji:

Kalkulator – calc

Notatnik – notepad

Paint - mspaint

plik19.cmd

Napisz skrypt, który poprosi o dwie liczby, zapyta o działanie arytmetyczne do wykonania – poda wynik. Zapyta, czy następne działanie?

Podaj A=

Podaj B=

Działanie: [1]-dodawanie, [2]-odejmowanie [3]-mnożenie [4]dzielenie [1,2,3,4]?

Skrypty z parametrami

plik20.cmd

1. Przepisz poniższy kod.

```
@ECHO OFF
IF "%1"=="" GOTO error
REM Jeżeli nie podano parametru to podaje składnię.
REM Jeżeli podano parametr to wypisze zdanie.
ECHO Podano parametr %1.
GOTO koniec
:error ECHO Prawidłowa składnia: %0 parametr
:koniec
pause
```

2. Uruchom skrypt bez parametru: plik20

3. Uruchom skrypt z parametrem: plik20 100

plik21.cmd

1. Napisz skrypt, którego parametrami będą imię i nazwisko użytkownika. Przykładowe wywołanie: plik21 Jan Kowalski

2. Jeżeli podane zostaną oba parametry wyświetlony zostanie komunikat:

Witaj użytkowniku: Jan Kowalski

3. Jeśli nie zostanie wprowadzony jeden lub oba parametry pojawi się komunikat podający poprawną składnię wywołania:

Prawidłowa składnia: plik21 Imie Nazwisko

plik22.cmd

1. Napisz skrypt, który będzie sumował dwie liczby podane jako parametr.

Przykładowe wywołanie: plik22 10 20

2. Jeżeli podane zostaną oba parametry wyświetlony zostanie komunikat:

Wynik: 10 + 20 = 30

3. Jeśli nie zostanie wprowadzony jeden lub oba parametry pojawi się komunikat podający poprawną składnię wywołania: Prawidłowa składnia: plik22 liczba1 liczba2

plik23.cmd

1. Napisz skrypt, który będzie podawał wynik działania.

Przykładowe wywołanie: plik23 10 + 20

Wynik: 10 + 20 = 30

Przykładowe wywołanie: plik23 30 - 5

Wynik: 30 - 5 = 25

Przykładowe wywołanie: plik23 2 * 5

Wynik: 2 * 5 = 10

Przykładowe wywołanie: plik23 20 / 2

Wynik: 20 / 2 = 10

2. Zakładamy poprawność wprowadzenia parametrów (między elementami działania musi być spacja)

plik24.cmd

1. Przepisz poniższy kod.

```
@ECHO OFF
ECHO %0 ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9
ECHO %10 %11
pause
```

2. Uruchom skrypt z następującymi parametrami:

plik24 Kasia Ola Marek Ala Ela Jan Ania Kuba Kazio Franek

3. W pliku plik24_opis.txt zapisz wnioski z obserwacji działania skryptu

plik25.cmd

Przepisz poniższy kod.

```
1. @ECHO OFF
2. ECHO Skrypt: %0 wita w działaniu
3. ECHO.
4. ECHO.
```

5. :petla

6. IF "%1" == "" GOTO koniec

7. ECHO %1

8. SHIFT

9. GOTO petla

10. :koniec

11. pause

Uruchom skrypt z dowolną ilością parametrów.

Opis: Powyższy skrypt zawiera nowe polecenie SHIFT. Powoduje ono „przesunięcie" parametru z pozycji wyższej na niższą. Oznacz to, że parametr %1 staje się parametrem %0, %2 parametrem %1, itd. W skrypcie (linia 6) sprawdzamy istnienie parametru (konieczne jest zastosowanie cudzysłowu gdyż porównujemy ciągi) – jeśli istnieje wyświetlamy (linia 7), pobieramy następny parametr (linia 8 – SHIFT) podstawiamy za parametr %1 – wracamy do linii 5.

Uwaga: Ten sposób programowania pozwala na wprowadzanie dowolnej ilości parametrów.
Nie występuje tu ograniczenie do 9-ciu parametrów.

plik26.cmd

Zmodyfikuj skrypt plik25.cmd, tak aby numerował parametry i w podsumowaniu podawał ich ilość.

Wywołanie: plik26 Kasia Ola Marek

Wynik:

parametr 1: Kasia

parametr 2: Ola

parametr 3: Marek

Ilość wprowadzonych parametrów: 3

plik27.cmd

Zakładamy, że parametrami są dowolne liczby całkowite. Napisz skrypt, który poda ilość liczb: ujemnych, dodatnich i zer.

Wywołanie: plik27 1 -2 2 -3 8 0 -5 0 -2

Wynik:

dodatnie: 3

ujemne : 4

zera : 2

plik28.cmd

Przepisz poniższy kod.

```
1. @ECHO OFF
2. ECHO %~f0
3. ECHO %~d0
4. ECHO %~p0
5. ECHO %~n0
6. ECHO %~x0
7. pause
```

Uruchom skrypt bez parametrów.

Opis: Powyższy skrypt zawiera modyfikatory parametrów pozwalające na wyświetlenie wybranej informacji:

%~f Pełna ścieżka

%~d Tylko litera dysku

%~p Tylko katalog

%~n Tylko nazwa pliku

%~x Tylko rozszerzenie pliku

plik29.cmd

Napisz skrypt, który będzie podawał informację o wywoływanym pliku.

Wywołanie: plik29

Wynik:

Jestem plikiem: plik29.cmd

Moja lokalizacja to: W:\zadanie4\

plik30.cmd

Napisz skrypt, który w zależności od podanego argumentu (katalog, strona lub czas) będzie wykonywał następujące działania:

plik30 katalog – wyświetla zawartość bieżącego katalogu według wzoru: Lista plików w katalogu: \zadania4\ lista plików (tylko nazwy) posortowana alfabetycznie wygenerowano: data godzina

plik30 strona – wyświetla stronę technikum: http://www.zsl.gda.pl/

plik30 czas – w pliku data_czas.txt zapisuje informację o aktualnej dacie i czasie.

Plik: data_czas.txt start: czas rozpoczęcia

Plik: data_czas.txt stop: czas zakończenia

Zawartość pliku:

Aktualna data: data

Aktualna godzina: czas

wygenerowano przez: nazwa użytkownika

Jeśli podany zostanie inny parametr – wyświetlony zostaje komunikat o błędzie i przedstawiana poprawna składnia wywołania.

plik31.cmd

Sprawdzanie istnienia pliku podanego jako parametr i wyświetlanie jego zawartości.

Skrypt:

```
1. @ECHO OFF
2. ECHO Skrypt: %0 wita w działaniu
3. ECHO.
4. ECHO.
5. if "%1"=="" GOTO error
6. IF EXIST %1 (
7.  CLS
8.  ECHO Zawartość pliku: %1
9.  ECHO.
10.  ECHO.
11.  TYPE %1
12. ) ELSE (
13.  ECHO plik %1 nie istnieje
14. )
15. GOTO koniec
16. :error
17. ECHO Brak parametru
18. ECHO Właściwa składnia plik31 nazwa_pliku
19. :koniec
20. ECHO.
21. ECHO.
```

22. PAUSE

Opis: Powyższy skrypt zawiera nową konstrukcję: IF warunek (polecenia) ELSE (polecenia) oraz sprawdzanie warunku EXIST

Jeśli plik, podany jako parametr, istnieje wykonujemy polecenia z linii 7 do 11 (wyświetlenie zawartości pliku).

Jeśli plik nie istnieje ELSE – wyświetlamy komunikat o braku pliku.

plik32.cmd

Uruchamiając plik31: plik31 plik25 otrzymujemy komunikat o błędzie: plik: plik25 nie istnieje. Uruchamiając jednak skrypt plik25 otrzymujemy poprawny wynik.

Zmodyfikuj skrypt plik31.cmd (zapisując jako plik32.cmd), tak aby po wywołaniu: plik32 plik25 działał poprawnie.

PĘTLE FOR

plik33.cmd

Wyświetlenie zawartości bieżącego katalogu.

Skrypt:

1. @ECHO OFF

2. CLS

3. ECHO.

4. ECHO.

5. FOR %%z IN (*.*) DO echo %%z

6. ECHO.

7. ECHO.

8. PAUSE

Opis: Powyższy skrypt zawiera nową konstrukcję: FOR %%z IN (*.*) DO echo %%z

Działanie jest następujące:

- dla (FOR) zmiennej %%z
- ze zbioru (*.*)- (wszystkie pliki) przypisz ich nazwy
- wykonaj (DO) polecenie (echo %%z) wyświetlające te nazwy

plik34.cmd

Napisz skrypt (wykorzystujący pętlę FOR) wyświetlający pliki z rozszerzeniem dll z katalogu C:\Windows

plik35.cmd

Napisz skrypt (wykorzystujący pętlę FOR) liczący ilość plików z rozszerzeniem exe w katalogu C:\Windows

plik36.cmd

W skrypcie plik33.cmd zmień linię 5 (zapisz jako plik36.cmd) do następującej postaci:

```
FOR /D %%z IN (C:\*.*) DO echo %%z
```

Opis: Przełącznik /D powoduje wykonywanie operacji jedynie na katalogach

plik37.cmd

W skrypcie plik36.cmd zmień linię 5 (zapisz jako plik37.cmd) do następującej postaci:

```
FOR /L %%i IN (1,1,10) DO @echo %%i
```

Opis: Przełącznik /L powoduje, że jako zbiór wejściowy pobierany jest zakres liczb według schematu: (start, krok, koniec)

Przykłady:

(1, 1, 10) – od 1 do 10 co 1 wartości: 1,2,3,4,5,6,7,8,9,10

(1, 2, 10) – od 1 do 10 co 2 wartości: 1,3,5,7,9

(5, 6, 23) – od 5 do 23 co 6 wartości: 5,11,17,23

(20, -5, 0) – od 20 do 0 co 5 wartości: 20,15,10,5,0

plik38.cmd

Napisz skrypt wyświetlający:

- liczby parzyste <2, 40>
- liczby parzyste <11, 34)
- liczby nieparzyste (-3, 45>
- liczby malejące co 3 <21, -6)

plik39.cmd

Napisz skrypt tworzący w katalogu W:\zadanie4 (katalog ćwiczeniowy) katalogi o nazwach kat1 do katn gdzie n jest liczbą tworzonych katalogów.

Wywołanie: plik39 10 - utworzy 10 katalogów.

plik40.cmd

Dodaj „ograniczenia” w pliku plik39 (zapisz jako plik40), tak aby można było utworzyć jedynie od 3 do 20 katalogów. Podanie innej wartości generować będzie komunikat o błędzie.

plik41.cmd

Zmień plik40 (zapisz jako plik41), tak aby po wpisaniu jako parametru wartości mniejszej niż istniejąca ilość katalogów, pozostałe zostały usunięte (zakładamy, że katalogi są puste).

Przykład:

Jeśli istnieją katalogi: kat1 do kat20,

wywołanie skryptu: plik41 10

spowoduje usunięcie katalogów kat11 do kat 20