

## Lab. 1.2. Aplikacja konsolowa bez interakcji (tryb wsadowy)

przekazywanie argumentów z linii komend konsoli do programu, argumenty funkcji Main(), uruchamianie aplikacji w trybie wsadowym, konfiguracja środowiska Visual Studio do pracy z aplikacją wsadową

### C#

15.05.2018

KRZYSZTOF MOLENDY - MICROSOFT IMAGINE ACADEMY@WSEI

1

Strona 1 / 13 | - Q +

#### C#

### Cel

- › W ramach tego ćwiczenia zmodyfikujesz opracowaną aplikację w Lab. 1.1. tak, aby wymagane dane (imię, nazwisko, wiek) wprowadzone zostały do programu z poziomu linii komend konsoli Windows (w podobny sposób jak polecenia systemu operacyjnego).
  - › Podstawowe pojęcia:
    - *Tryb wsadowy, przetwarzanie wsadowe* – wykonywanie serii zadań (programów) przez komputer. Zazwyczaj kolejne zadania są ze sobą powiązane: dane wyjściowe jednego programu przekazywane są kolejnemu programowi, któremu służą jako dane wejściowe itd. [Wikipedia]
    - *Program wsadowy, plik wsadowy* (ang. *batch file*) – program komputerowy wykonywany w trybie wsadowym, tj. bez wpływu użytkownika na przebieg programu. Plik wsadowy to plik, w którym zapisany został program wsadowy. Programy wsadowe mogą być wykonywane w różnych systemach operacyjnych, w szczególności w systemach Unix, Linux, DOS, Windows i innych.
- Program wsadowy z założenia nie oczekuje współpracy (interakcji) z użytkownikiem w trakcie wykonywania się programu. Po uruchomieniu programu użytkownik nie ma wpływu na przebieg programu wsadowego, może jedynie oczekiwać na jego zakończenie lub ewentualnie przerwać jego działanie. Jednak jeśli program wsadowy wymaga podania parametrów potrzebnych do działania, np. opcji, nazwy pliku, katalogu czy nazwy użytkownika, należy podać je w chwili uruchamiania programu w wierszu poleceń jako argumenty wywołania programu. [Wikipedia]

Strona 2 / 13 | - Q +

### C#

### Tworzenie nowego projektu

- › Do istniejącej solucji dodaj nowy projekt:
  - W oknie *Solution Explorer* prawokliknij myszką na *Solution 'Lab01'* i z menu kontekstowego wybierz



Add→New Project ...

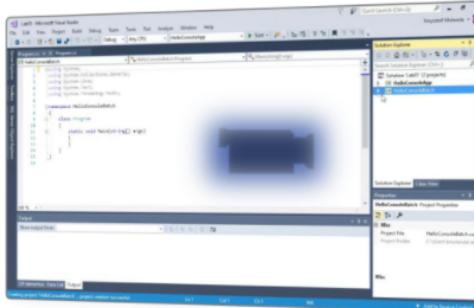
- Utwórz projekt według szablonu **Visual C# → Console App (.NET Framework)** nadając mu nazwę **HelloConsoleBatch**
- (ważne: szablon nie **.Net core**)

screencast→



- W efekcie, w ramach solucji będą dostępne 2 projekty, ale tylko jeden może być w danym momencie projektem głównym (tzn. nadającym się do uruchamiania i testowania).
  - Uczyń nowo utworzony projekt aktywnym (**StartUp Project**): w oknie **Solution Explorer** prawokliknij na nazwie nowo utworzonego projektu i wybierz **Set as StartUp Project**.
  - Projekt aktywny jest **wytłuszczone**.

screencast→



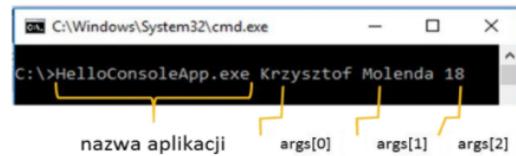
3

## Aplikacja wsadowa – podstawy

- Aplikację konsolową uruchamiamy z poziomu linii komend podając nazwę programu, po której wystąpią parametry programu oddzielone „białymi znakami” (np. spacją).
- Parametry te można przechwycić do swojego programu za pomocą tablicy parametrów określonej w deklaracji funkcji Main:
 

```
static void Main(string[] args)
```
- Zapis ten wskazuje, że funkcja Main może otrzymać parametry wywołania programu w formie tablicy napisów **string[]**. Tablica ta nazywa się **args**. Proces przechwytywania danych ilustruje rysunek (uwaga: w C# komórki tablic numerowane są od 0).

```
C:\Windows\System32\cmd.exe
C:\>HelloConsoleApp.exe Krzysztof Molenda 18
```



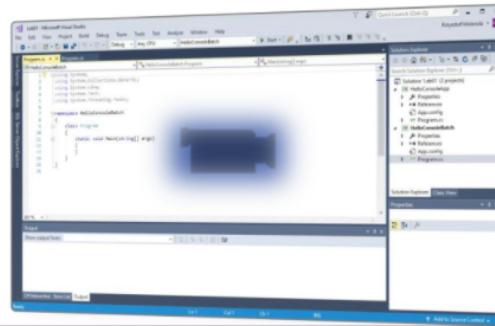
Zatem modyfikacja poprzedniego programu polegała będzie na zastąpieniu wczytywania odpowiednich danych (czyli zbędnych **Console.WriteLine()** oraz **Console.ReadLine()**) przypisaniem do zmiennych odpowiednich komórek tablicy **args**:

```
string imie = args[0];
string nazwisko = args[1];
...
int wiek = Convert.ToInt32(args[2]);
```

## Aplikacja wsadowa – kod

- Skopiuj odpowiedni fragment kodu z **Program.cs** z projektu **HelloConsoleApp** do **Program.cs** nowo utworzonego projektu **HelloConsoleBatch**.

screencast→



- Następnie dokonaj omówionych **wzorców korrek**

## WŁOCZSZEJ KOLEJKI.

```
string imie = args[0];
string nazwisko = args[1];
...
int wiek = Convert.ToInt32( args[2] );
```

pełny kod→

```
class Program
{
    static void Main(string[] args)
    {
        //tu piszemy kod
        Console.WriteLine("Program na powitanie.");

        string imie = args[0];
        string nazwisko = args[1];

        Console.WriteLine(" Witaj " + imie + " " + nazwisko);
        Console.WriteLine(" Witaj {0} {1}. Czy Pan {0} rzeczywiście nazywa się {1}?", imie, nazwisko);

        int wiek = Convert.ToInt32(args[2]); //możliwe zgłoszenie wyjątku

        if (wiek < 67)
        {
            Console.WriteLine(" do emerytury zostało Ci {0} lat", 67 - wiek);
        }
        else
        {
            Console.WriteLine(" jesteś emerytem");
        }

        //zatrzymanie konsoli
        //Console.WriteLine("... naciśnij dowolny klawisz, aby zakończyć");
        //Console.ReadKey();
        //podpowiedź:
        //F5 - uruchamianie z debuggerem - zatrzymanie konsoli przez ReadKey()
        //Ctrl-F5 - uruchamianie bez debuggera - nie ma potrzeby zatrzymywania konsoli
    }
}
```

KRZYSZTOF MOLENDY - MICROSOFT IMAGINE ACADEMY@WSEI

5

```
static void Main(string[] args)
{
    //tu piszemy kod
    Console.WriteLine("Program na powitanie.");

    string imie = args[0];
    string nazwisko = args[1];

    Console.WriteLine(" Witaj " + imie + " " + nazwisko);
    Console.WriteLine(" Witaj {0} {1}. Czy Pan {0} rzeczywiście nazywa się {1}?", imie, nazwisko);

    int wiek = Convert.ToInt32(args[2]); //możliwe zgłoszenie wyjątku

    if (wiek < 67)
    {
        Console.WriteLine(" do emerytury zostało Ci {0} lat", 67 - wiek);
    }
    else
    {
        Console.WriteLine(" jesteś emerytem");
    }

    //zatrzymanie konsoli
    //Console.WriteLine("... naciśnij dowolny klawisz, aby zakończyć");
    //Console.ReadKey();
    //podpowiedź:
    //F5 - uruchamianie z debuggerem - zatrzymanie konsoli przez ReadKey()
    //Ctrl-F5 - uruchamianie bez debuggera - nie ma potrzeby zatrzymywania konsoli
}
```

15.05.2018

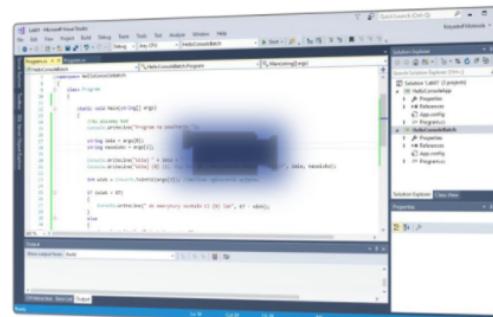
KRZYSZTOF MOLENDY - MICROSOFT IMAGINE ACADEMY@WSEI

6

## Aplikacja wsadowa – kompilacja i uruchomienie

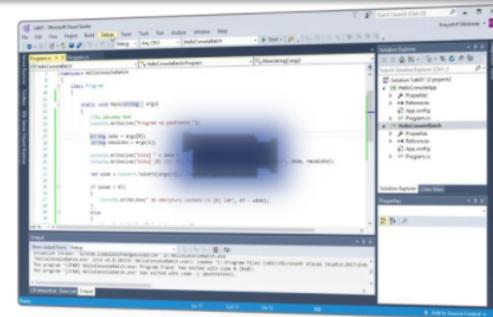
- Aby skompilować projekt, w *Solution Explorer* należy prawokliknąć na nazwie projektu i wybrać **Build**.
- Jeśli nie będzie błędów – wytworzony zostanie kod wynikowy aplikacji.
- Próba uruchomienia aplikacji (**F5** – bez debuggera) zakończy się zgłoszeniem wyjątku.

screencast→



- Próba uruchomienia aplikacji (**Ctrl-F5** – z debuggerem) również zakończy się zgłoszeniem wyjątku.
- Aplikacja przerwa swoje działanie z powodu zgłoszenia wyjątku **IndexOutOfRangeException**.
- Przyczyną błędu jest niepodanie odpowiedniej liczby poprawnych parametrów.

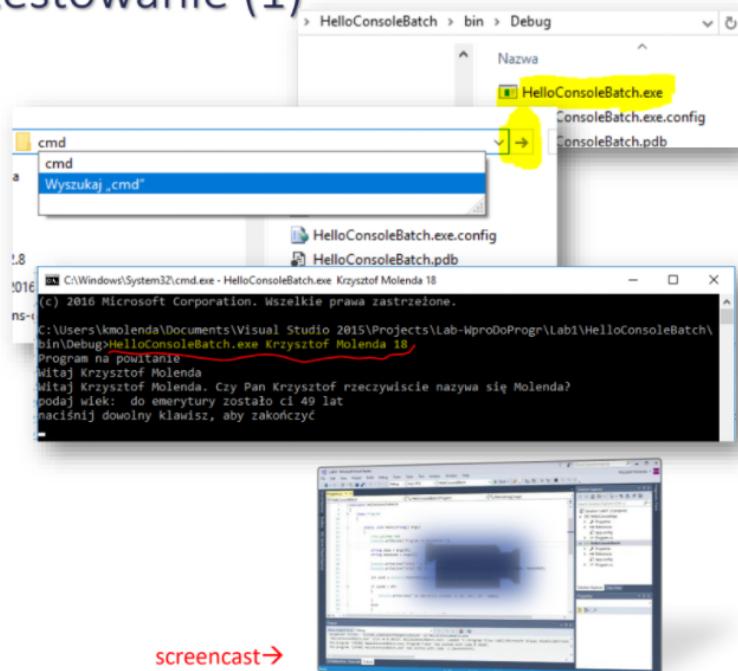
screencast→



## Aplikacja wsadowa – testowanie (1)

Aplikację wsadową testujemy, uruchamiając ją w oknie konsoli Windows.

- Za pomocą eksploratora Windows należy przejść do podfolderu `bin\Debug` projektu. W folderze tym znajduje się skompilowany, wykonywalny poza środowiskiem Visual Studio plik programu `HelloConsoleBatch.exe`
- W pasku ścieżki dostępu eksploratora należy wpisać polecenie `cmd` i zatwierdzić klawiszem ENTER lub strzałki w prawo. Otworzy się konsola systemu Windows we wskazanej lokalizacji.
- Wpisując `Hell` i naciskając klawisz TAB autouzupełniamy do tekstu `HelloConsoleBatch.exe` (jeśli konsola nie podpowie właściwie, naciskaj klawisz TAB do skutku albo po prostu wpisz nazwę Twojego programu). Następnie, oddzielając spacją wprowadzamy 3 parametry: imię, nazwisko i wiek i uruchamiamy program.



screencast→

Microsoft Imagine Academy  
Program Member15.05.2018  
Strona 8 / 13

KRZYSZTOF MOLENDY - MICROSOFT IMAGINE ACADEMY@WSEI

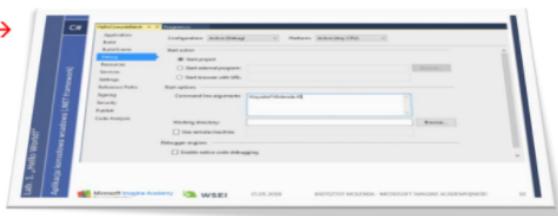
8

## Aplikacja wsadowa – testowanie (2)

Visual Studio dostarcza specjalnego trybu umożliwiającego testowanie aplikacji wsadowych, bez przechodzenia do okna konsoli Windows.

- Należy prawokliknąć na nazwie projektu w Solution Explorer i z menu kontekstowego wybrać **Properties**. Otworzy się nowa zakładka/formularz z parametrami projektu.
- W kategorii **Debug** → **Start options** → **Command line arguments**: należy wpisać parametry, które chcemy przekazać aplikacji wsadowej. Zmiany należy zapisać (**Ctrl-S**)
- Teraz, bez opuszczania Visual Studio, można testować aplikację wsadową – zarówno bez debugera (**Ctrl-F5**) jak i z debuggerem (**F5**).
- W tym drugim przypadku należy odcommentować kod blokujący zamknięcie się konsoli.
- Program działa, ale tylko w sytuacji poprawnego jego użycia – podana jest wymagana liczba argumentów i trzeci parametr jest ciągiem znaków, który zinterpretować można jako liczbę całkowitą. W każdym innym przypadku program przestanie działać i zgłosi błęd.**

podgląd→



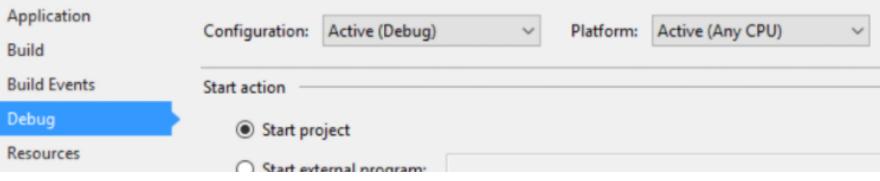
screencast→

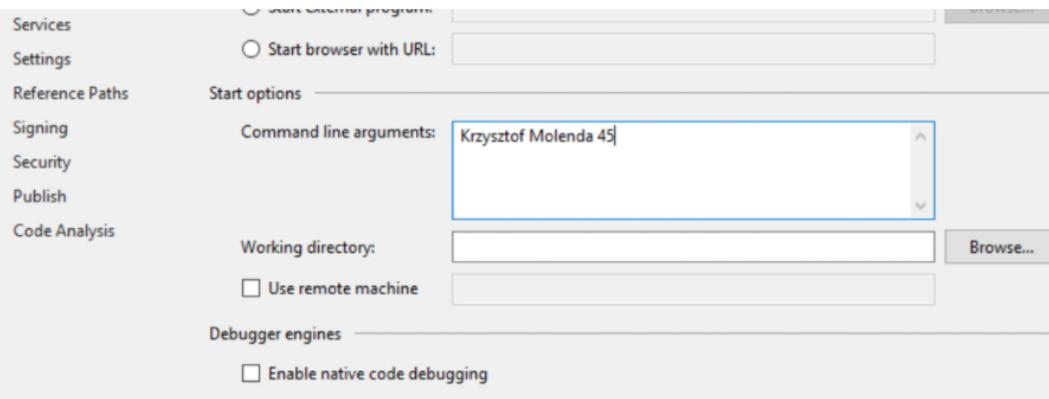
Microsoft Imagine Academy  
Program Member15.05.2018  
Strona 9 / 13

KRZYSZTOF MOLENDY - MICROSOFT IMAGINE ACADEMY@WSEI

9

## HelloConsoleBatch -> Program.cs





## Zadanie domowe 1

Zmodyfikuj program tak, aby w przypadku:

1. nie podania żadnego parametru wyświetlił komunikat: „Brak danych”,
2. podania tylko jednego parametru potraktował go jako imię i wyświetlił „ *Witaj imię*”,
3. podania dwóch parametrów – pierwszy potraktował jako imię, drugi jako nazwisko – i wyświetlił „ *Witaj imię nazwisko*”,
4. podania 3 parametrów zadziałał tak, jak przed modyfikacją,
5. podania więcej niż 3 parametrów – wziął pod uwagę tylko 3 pierwsze i zadziałał jak w punkcie poprzednim.

Zakładamy, że trzeci parametr (wiek) nie spowoduje błędu.

Podpowiedź:

- › By zrealizować to zadanie, będziesz musiał użyć konstrukcji warunkowego wyboru `if-then-else` oraz sprawdzić liczbę parametrów, badając wielkość tablicy `args` polecienniem `args.Length`

```
if (args.Length == 1)
{
    //kod
}
i.t.d.
```

Równość wartości w C# realizowana jest za pomocą podwójnego znaku „równa się” (`==`).

- › Zasięg (widoczności) zmiennej lokalnej rozciąga się od momentu jej zadeklarowania do końca bloku, w którym została zadeklarowana. Był może na początku kodu będziesz musiał zadeklarować wszystkie zmienne, a wartości przypiszesz dopiero wtedy, gdy będziesz pewien, że zostały przekazane za pośrednictwem tablicy `args`.

```
string imie = "";
if (args.Length == 1)
{
    imie = args[0];
}
```



## Zadanie domowe 2

Utwórz nowy projekt o nazwie `HelloConsoleAppBatch`. Połącz w jedną aplikację rozwiązanie z Lab. 1.1 oraz z Lab. 1.2 tak, aby Twój program w przypadku:

1. nie podania żadnego parametru zadziałał interaktywnie, czyli tak, jak w Lab. 1.1 (pytając użytkownika o imię, nazwisko i wiek)
2. podania tylko jednego parametru potraktował go jako imię, oraz interaktywnie poprosił o podanie brakujących: nazwiska oraz wieku (jak w Lab. 1.1)
3. podania dwóch parametrów – pierwszy potraktował jako imię, drugi jako nazwisko – i w tym przypadku poprosił o podanie brakującego

Podpowiedź:

- › Prawdopodobnie pojawi się zagnieżdżenie instrukcji warunkowego wyboru (jeden „`if`” wewnętrz innego „`if`”). Kontroluj logikę aplikacji i odpowiednie domykanie nawiasów klamrowych.
- › Kopij i wklejaj kod poprawny z innych projektów.
- › Dbaj, aby kod był czytelny i samodokumentujący się. Często wykonuj `Edit→Advanced→Format Document`, aby kod zachował standardowy wygląd (wcięcia).
- › Przetestuj wszystkie możliwe warianty wywołania aplikacji.

4. podania 3 lub więcej parametrów zadziałał tak, jak w Lab. 1.2 – czyli o nic nie pytał.

Zakładamy, że trzeci parametr (wiek) nie spowoduje błędu.

## Podsumowanie – pytania kontrolne

- › Podaj charakterystyczne cechy programu wsadowego.
- › W jaki sposób parametry z linii komend przekazywane są do programu wsadowego i w jaki sposób są obsługiwane?
- › Jaki może być efekt, przy uruchomieniu programu, nie podania właściwej liczby parametrów lub podania ich w niewłaściwej kolejności (np. najpierw wiek, później imię i nazwisko)?
- › W jaki sposób testujemy programy wsadowe.