



# Module 1 Day 5

Command-Line Programs

# What makes an application?

- Program Data

- ✓ Variables & .NET Data Types
- ✓ Arrays
- ❑ More Collections (list, dictionary, stack, queue)
- ❑ Classes and objects (OOP)

- Program Logic

- ✓ Statements and expressions
- ✓ Conditional logic (if)
- ✓ Repeating logic (for, foreach, do, while)
- ✓ Methods (functions / procedures)
- ❑ Classes and objects (OOP principles)
- ❑ Frameworks (MVC)

- Input / Output

- User
  - Console read / write
  - ❑ HTML / CSS
  - ❑ Front-end frameworks (HTML / CSS / JavaScript)
- Storage
  - ❑ File I/O
  - ❑ Relational database
  - ❑ APIs

# Command-Line Programs

- Console I/O (Standard I/O)
  - Console.Write method
  - Console.WriteLine method
    - Using Placeholders
  - Console.ReadLine method
- Converting a string to a number
  - Parse method

```
string numAsString = "123";  
int num = int.Parse(numAsString);
```

Let's  
Code

# Splitting and joining strings

- Split method

```
string numbersString = "1,2,3,4";  
string[] numbers = numbersString.Split(",");
```

- Separates the string into pieces, looking for the “separator” character
- Returns an array of strings

- Join method

```
string newString = string.Join('-', numbers);
```

- Kind of “the opposite of” Split
- Joins all elements of the array into a single string, inserting the “separator” character between them

Let's  
Code

# Creating a Command-Line Program

- Count, Average and Sum
- User enters a comma-delimited list of numbers
- Return to the user the Count, Sum and Average of the numbers
- Ask if they'd like to do another



Let's  
Code

# Creating a Command-Line Program

- “Proper-Nouner”
  - Accept a sentence from the user
  - Make every word start with a capital letter, remainder of the word lower case.
  - Make sure the sentence ends in a period
- Ask if they’d like to do another



Let's  
Code

# Creating a Command-Line Program

- Interest calculator
  - Initial Principal:  $p$
  - Interest Rate:  $r$
  - Investment time (years):  $t$
- Calculate balance after  $n$  years:
  - $\text{Balance} = p * (1 + r)^{**t}$
- Ask if they'd like to do another



Let's  
Code



# Pairs Exercises

- Pairs assigned
- Clone your repository
- Pair programming vs. parallel programming
- Make sure both partners get a chance to “drive”
  - Each partner should push their changes and pull their partner’s changes