2. Language Introduction

AGENDA

- Variables, Data Types & Constants
- Control Flow
  - code block
  - condition
  - loops
  - switch
- Arrays
- Operators
- Methods
- Documentation

LANGUAGE INTRODUCTION

Variables

- A variable:
    - holds a value
    - has a data type
    - is created in a declaration statement
- coding convention for naming:
    - starting with lower case
    - using camel case

Variables - Data Types

Define

- Size and location in memory
- Data range
- Valid operators

of/for a variable

> all built in types can be found here

LANGUAGE INTRODUCTION

Implicitly typed local variables

- declared without using a type

```
// i is compiled as an int
var i = 5;

// s is compiled as a string
var s = "Hello"
```

Constants

- using key-word `const`, e.g. `const int myInt = 10`

LANGUAGE INTRODUCTION

Control Flow

- Codeblock:
  - contains several lines of code
  - surrounded with `{ }`
- boolean expression:
  - returns a boolean value
  - operators: <, >, <=, >=, ==, !=

LANGUAGE INTRODUCTION

Control Flow

- Condition: if statement (if, else if, else)
- Loop:
  - for
  - foreach
  - while
  - do-while
- Switch: case statement
- break / continue

> more information and code examples can be find here

WISSENSQUIZ

- Nenne drei grundlegende Datentypen in C#? Gibt es Typen die sich stark ähneln?
- Was passiert, wenn Zahlen den Wertebereich ihres Datentypen übersteigen?
- Du ließt logs von einem Server in London, was musst du bei der Betrachtung der Zeitstempel beachten?
- Welche Möglichkeiten gibt es, in C# Schleifen zu programmieren?

LANGUAGE INTRODUCTION

Arrays

- can have one or more dimensions

```
string[] cars = {"Volvo", "BMW", "Ford", "Mazda"};
```

```
int[,] coordinates = {{1, 2}, {2,3}};
```

```
double[] doubleNumbers = new double[5];
```

> more information can be find here

LANGUAGE INTRODUCTION

Operators

- Arithmetic operators (+ - / * % ++ --) that perform arithmetic operations with numeric operands
- Comparison operators (< > == <= >=) that compare numeric operands
- Boolean logical operators (&& || ! ^ & |) that perform logical operations with bool operands
- Bitwise and shift operators (~ >> << | & >>>) that perform bitwise or shift operations with operands of the integral types
- Equality operators (== !=) that check if their operands are equal or not

> source

Methods, Parameters and structuring your code

- methods
  - are code block which only runs when they are called
  - use parameters to pass data into methods
  - can have a return value or void
  - are declared in a class or struct
- use methods in order to reuse code
- are also called functions

LANGUAGE INTRODUCTION

Methods, Parameters and structuring your code

```
class SimpleMathExtension
{
    public int DivideTwoNumbers(int number1, int number2)
    {
        return number1 / number2;
    }
}
```

```
class Class_Name
{
    <access modifier> <return type> Method_Name(Parameters)
    {
        //method statements
        return Return_Value;
    }
}
```

Main Method

Entry method which is called after start of the app

```
class SimpleMathExtension
{
    static void Main(string[] args)
    {
      int result = DivideTwoNumbers(9, 3);
      Console.WriteLine(result);
    }

    public int DivideTwoNumbers(int number1, int number2)
    {
        return number1 / number2;
    }
}
```

Method Overloading

```
/**
* Methods has the same name but different parameters
**/
class MethodOverloadExample
{
    public int addNumbers(int number1, int number2)
    {
        return number1 + number2;
    }

    public int addNumbers(int number1, int number2, int number3)
    {
        return number1 + number2 + number3;
    }

    public double addNumbers(double number1, double number2)
    {
        return number1 + number2;
    }
}
```

LANGUAGE INTRODUCTION

Local Scope Variables

```
public int DivideTwoNumbers(int number1, int number2)
{
    //local scoped variable - only available in method
    int returnValue = number1 / number2;
    return returnValue;
}
```

Optional Parameters

```
//number2 is an optional paramenter
public int DivideTwoNumbers(int number1, int number2 = 10)
{
    return number1 / number2;
}
```

WISSENSQUIZ

- Nenne arithmetische Operatoren. Welche Arten von Operatoren gibt es noch?
- Welche Zugriffsmodifizierer für Methoden sind dir bekannt?
- Wie heißt die Methode, die beim starten eines Projektes ausgeführt wird?
- Was sind lokale Variablen?
- Aus welchen drei Elementen besteht die Signatur einer Methode?

LANGUAGE INTRODUCTION

Documentation

- use **Logger** and not Console.WriteLine() to be able to redirect output e.g. in files by configuration. See here for detailed information
- use DocFx to provide api documentation for your programm

```
Rem installation
dotnet tool update -g docfx
Rem create docfx configuration
docfx init
Rem start webserver
docfx docfx.json --serve
```

WISSENSQUIZ

- Für wen schreibst du deinen Code?
- Wann und was würdest du dokumentieren?
- Welche log-Level gibt es?
- Welchen Vorteil haben Logger im Gegensatz zu direkten Consolenausgaben?