Review of C# Syntax

What Is .NET?

- CLR
  - Robust and secure environment for your managed code
  - Memory management
  - Multithreading
- Class library
  - Foundation of common functionality
  - Extensible
- Development frameworks
  - WPF
  - Entity Framework
  - ASP.NET

What Is C#?

- A language on top of .NET
- Statements ending with a semi colon
- Case sensitive

Creating a .NET Application

1. In Visual Studio, on the **File** menu, point to **New**, and then click **Project**.
2. In the **New Project dialog** box, choose a template, location, name, and then click **OK**.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApplication1
{
   class Program
   {
       static void Main(string[] args) { }
   }
}
```

What are Data Types?

- int – whole numbers
- long – whole numbers (bigger range)
- float – floating-point numbers
- double - double precision
- decimal - monetary values
- char - single character
- bool - Boolean
- DateTime - moments in time
- string - sequence of characters

Declaring and Assigning Variables

- Declaring variables:

```
int price;
// OR
int price, tax;
```

- Assigning variables:

```
price = 10;
// OR
int price = 10;
```

- Implicitly typed variables:

```
var price = 20;
```

Expressions and Operators in Visual C#

Example expressions:

- + operator

```
a + 1
```

- / operator

```
5 / 2
```

- + and – operators

```
a + b - 2
```

- + operator (string concatenation)

```
"ApplicationName: " + appName.ToString()
```

Casting Between Data Types

- Implicit conversion:

```
int a = 4;
long b = 5;
b = a;
```

- Explicit conversion:

```
int a = (int) b;
```

- **Native Type conversion**

```
string possibleInt = "1234";
int count = int.Parse(possibleInt);
```

- **System.Convert** conversion:

```
string possibleInt = "1234";
int count = Convert.ToInt32(possibleInt);
```

Implementing Conditional Logic

- **if** statements

```
if (response == "connection_failed") {. . .}
else if (response == "connection_error") {. . .}
else { }
```

- **select** statements

```
switch (response)
{
   case "connection_failed":
      . . .
      break;
   case "connection_success":
      . . .
      break;
   default:
      . . .
      break;
}
```

Console and Interpolated Strings

- To output text to the console

```
Console.WriteLine("Some text");
```

- To read text from the console

```
var input = Console.ReadLine(); //input is a string here!
```

- Interpolated strings

```
Console.WriteLine($"Hello {firstName}!");
```

Parsing values from strings

- Parsing is the ability to generate a variable of their underlying type given a textual equivalent
  - Useful since all string input from the user arrives as string data

Lab 1

https://github.com/RolandGuijt/csharpworkshop

Implementing Iteration Logic

- **for** loop

```
for (int i = 0 ; i < 10; i++) { ... }
```

- **foreach** loop

```
string[] names = new string[10];
foreach (string name in names) { ... }
```

- **while** loop

```
bool dataToEnter = CheckIfUserWantsToEnterData();
while (dataToEnter)
{
    ...
    dataToEnter = CheckIfUserHasMoreData();
}
```

- **do** loop

```
do
{
    ...
    moreDataToEnter = CheckIfUserHasMoreData();
} while (moreDataToEnter);
```

Practicing iteration

- Create an int variable that is initialized to 0.
- Create a for loop that adds 1 to the int.
- Print out the int afterwards with a Console.WriteLine and observe it.

- Achieve the same functionality with a while loop.

Lab 2

Creating and Using Arrays

- Creating an array:

```
int[] arrayName = new int[10];
```

```
int[] arrayName = { 1, 2, 3, 4, .. };
```

- Accessing data in an array:
  - By index

```
int result = arrayName[2];
```

  - In a loop

```
for (int i = 0; i < arrayName.Length; i++)
{
    int result = arrayName[i];
}
```

Using Breakpoints in Visual Studio

- Breakpoints enable you to view and modify the contents of variables:
  - Immediate Window
  - Autos, Locals, and Watch panes
- Debug menu and toolbar functions enable you to:
  - Start and stop debugging
  - Enter break mode
  - Restart the application
  - Step through code

Lab 3