C#: The Big Picture

Discovering C#

Mike Woodring
Programmer | Learner | Teacher

@mcwoodring linkedin.com/in/woodring

A Little Bit About You

- A developer, newbie ←→ veteran
- Curious about C#
- Skeptical about C#
- Aspiring C# developer/communicator

A Little Bit About Me

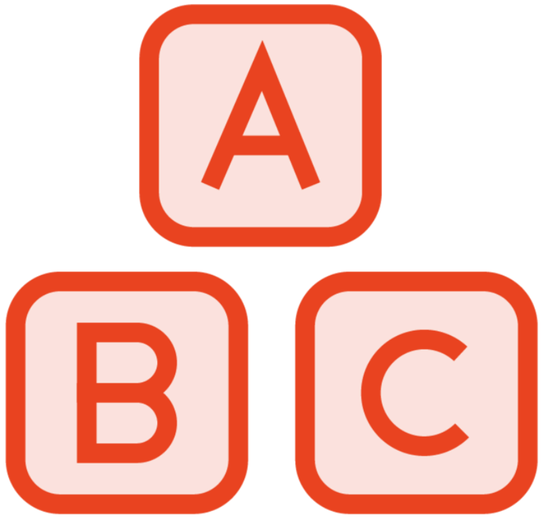
A (gray-haired) developer

Curious about programming languages

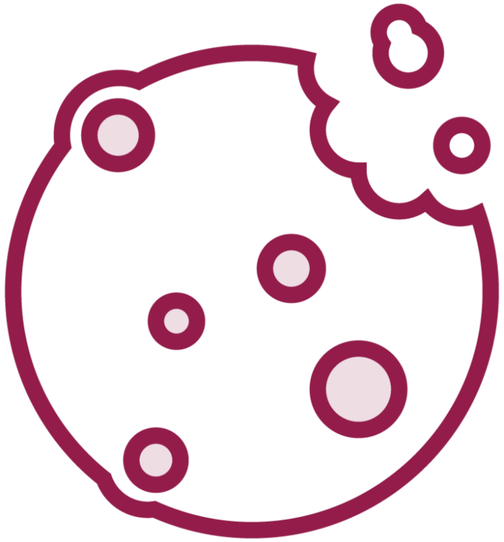
Skeptical about bold claims

Appreciative C# developer

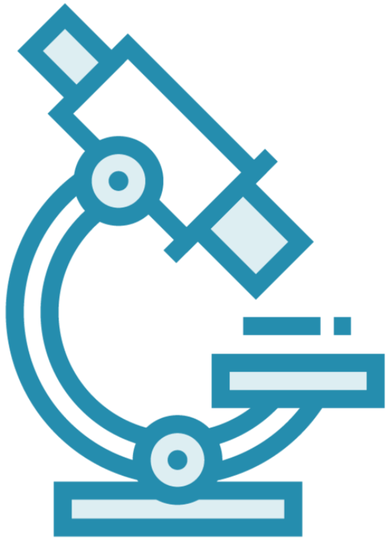
A Little Bit (More) About This Course



Elemental



Flavorful



Revealing

Directional

Tools Used in This Course

C# 9

.NET 5

Visual Studio 2019 Community

The Essence of C#

Rivalry


```cpp
#include <stdio.h>

int main() {
  printf("Hello, world!");
  return 0;
}
```


```java
class HelloWorld {
  public static int main(String[] args) {
    System.out.println("Hello, World!");
    return 0;
  }
}
```

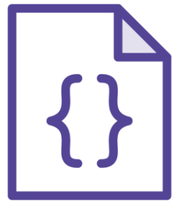
Approachable
(to C++ & Java developers)

C# Is Approachable

(to some)

```
using System;

class Program
{
    static int Main(string[] args)
    {
        Console.WriteLine("Hello, world!");

        for (int n = 0; n < args.Length; n++)
        {
            Console.WriteLine("arg[{0}] = {1}", n, args[n]);
        }

        return 0;
    }
}
```

```
c:\> program.exe 30 12

Hello, world!
arg[0] = 30
arg[1] = 12
```



Approachable
(to C++ & Java developers)

Strongly typed
(with compiler inference)

C# Is Strongly (Statically) Typed

(with compile-time type inference)

```
using System;

class Program
{
    static int Main(string[] args)
    {
        Console.WriteLine("Hello, world!");

        for (int n = 0; n < args.Length; n++)
        {
            Console.WriteLine("arg[{0}] = {1}", n, args[n]);
        }

        return 0;
    }
}
```

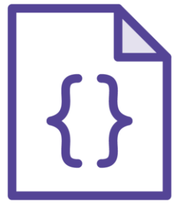
```
c:\> program.exe 30 12

Hello, world!
arg[0] = 30
arg[1] = 12
```

C# Is Strongly (Statically) Typed

(with compile-time type inference)

```
using System;

class Program
{
    static int Main(string[] args)
    {
        Console.WriteLine("Hello, world!");

        for (var n = 0; n < args.Length; n++)
        {
            Console.WriteLine("arg[{0}] = {1}", n, args[n]);
        }

        return 0;
    }
}
```

```
c:\> program.exe 30 12

Hello, world!
arg[0] = 30
arg[1] = 12
```



Approachable
(to C++ & Java developers)

Strongly typed
(with compiler inference)

Resilient & safe
(with native performance)

C# Is Resilient & Safe

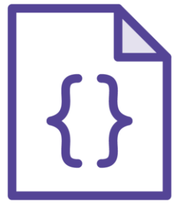
(with native performance)

```
class Program
{
    static void Main()
    {
        var numbers = new int[] { 1, 2, 3, 4, 5 };
        var sum = 0;

        for (var n = 0; n < numbers.Length; n++)
        {
            sum += numbers[n] ;
        }

        System.Console.WriteLine(sum);
    }
}
```

```
c:\> program.exe

15
```

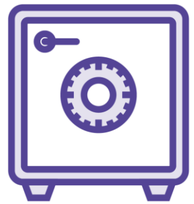
DEMO - Safety

DEMO - Resilience

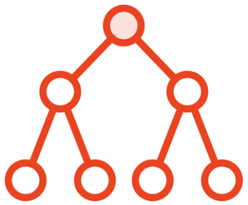


Approachable
(to C++ & Java developers)

Strongly typed
(with compiler inference)



Resilient & safe
(with native performance)



Object-oriented
(with functional features)

C# Is Object-Oriented

(with functional features)

```
class Program
{
    static void Main()
    {
        var numbers = new int[] { 1, 2, 3, 4, 5 };
        var type = numbers.GetType();

        do
        {
            System.Console.WriteLine(type.FullName);
            type = type.BaseType;
        }
        while (type != null);
    }
}
```

```
c:\> program.exe

System.Int32[]
System.Array
System.Object
```

C# Is Object-Oriented

(with functional features)

```
class Program
{
    static void Main()
    {
        var numbers = new int[] { 1, 2, 3, 4, 5 };
        var sum = 0;

        for (var n = 0; n < numbers.Length; n++)
        {
            sum += numbers[n] ;
        }

        System.Console.WriteLine(sum);
    }
}
```

```
c:\> program.exe

15
```

C# Is Object-Oriented

(with functional features)

```
using System;
using System.Linq;

class Program
{
    static void Main()
    {
        var numbers = new int[] { 1, 2, 3, 4, 5 };
        var sum = numbers.Aggregate(
            0,
            (total, num) => total + num
        );

        Console.WriteLine(sum);
    }
}
```

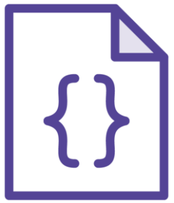
```
c:\> program.exe

15
```

C# Is Object-Oriented

(with functional features)

```
using System;
using System.Linq;

class Program
{
    static void Main()
    {
        var numbers = new int[] { 1, 2, 3, 4, 5 };
        var sum = numbers.Aggregate(
            0,
            (total, num) => {
                Console.WriteLine("total = {0}, num = {1}“, total, num);
                return total + num;
            }
        );

        Console.WriteLine(sum);
    }
}
```

General purpose
(desktop, mobile, web, game)

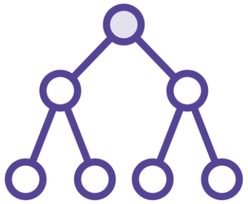


Approachable
(to C++ & Java developers)

Strongly typed
(with compiler inference)

Open source & cross-platform
(using .NET 5)

Resilient & safe
(with native performance)



Object-oriented
(with functional features)

Summary

The Big Picture

- Approachable
- Strongly typed
- Resilient to change/runtime type safety
- Object-oriented, with functional features
- Open-source & cross-platform
- General purpose

Courses Referenced

Paolo Perotta, C#: Getting Started

Gill Cleeren, Introduction to the C# Type System

Elton Stoneman, C# Extension Methods

Paul D. Sheriff, C# Language-Integrated Query (LINQ)

Up Next:
Exploring Managed Execution in C#