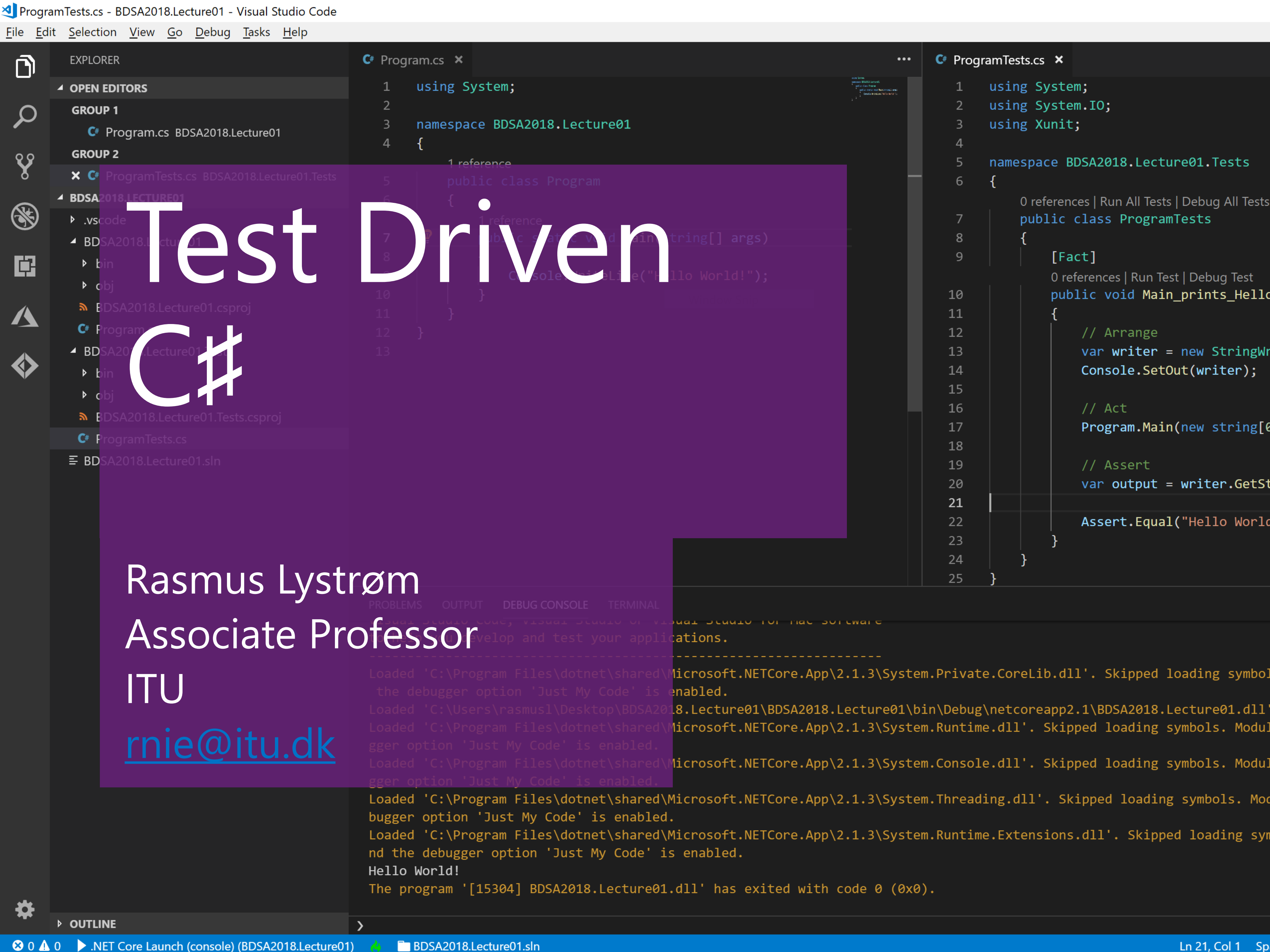


Test Driven C#

Rasmus Lystrøm
Associate Professor
ITU
rnie@itu.dk

ProgramTests.cs - BDSA2018.Lecture01 - Visual Studio Code

Program.cs

```csharp
using System;

namespace BDSA2018.Lecture01
{
    1 reference
    public class Program
    {
        1 reference
        public static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```

ProgramTests.cs

```csharp
using System;
using System.IO;
using Xunit;

namespace BDSA2018.Lecture01.Tests
{
    0 references | Run All Tests | Debug All Tests
    public class ProgramTests
    {
        [Fact]
        0 references | Run Test | Debug Test
        public void Main_prints_Hello
        {
            // Arrange
            var writer = new StringWr
            Console.SetOut(writer);

            // Act
            Program.Main(new string[0

            // Assert
            var output = writer.GetSt

            Assert.Equal("Hello Worl
        }
    }
}
```

```
Visual Studio Code, Visual Studio or Visual Studio for Mac software
to develop and test your applications.
-------------------------------------------------------------------
Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\2.1.3\System.Private.CoreLib.dll'. Skipped loading symbol
 the debugger option 'Just My Code' is enabled.
Loaded 'C:\Users\rasmusl\Desktop\BDSA2018.Lecture01\BDSA2018.Lecture01\bin\Debug\netcoreapp2.1\BDSA2018.Lecture01.dll'
Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\2.1.3\System.Runtime.dll'. Skipped loading symbols. Modul
gger option 'Just My Code' is enabled.
Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\2.1.3\System.Console.dll'. Skipped loading symbols. Modul
gger option 'Just My Code' is enabled.
Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\2.1.3\System.Threading.dll'. Skipped loading symbols. Mod
bugger option 'Just My Code' is enabled.
Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\2.1.3\System.Runtime.Extensions.dll'. Skipped loading sym
nd the debugger option 'Just My Code' is enabled.
Hello World!
The program '[15304] BDSA2018.Lecture01.dll' has exited with code 0 (0x0).
```

Me

M.Sc. IT, ITU
Thesis: Forecalc – Developing a core spreadsheet implementation in F♯

Senior Consultant @ Microsoft
DevOps, Cloud, Security

Associate Professor @ ITU
Object-Oriented Programming, C♯, F♯, .NET Core

Captain @ Danish Army (Reserve)
Acting Battalion Chief of Staff,
Battalion Chief Operations Officer

Hobbies

CANNIBAL CORPSE

SLAYER

OBITUARY

SCALEXTRIC

MORBID ANGEL

COPENHELL

HELLFEST

Wife: Katrine, Children: Lærke (1), Laura (4) and Alma (11)
Origin: Aarhus, Current whereabouts: Vanløse, Copenhagen

Agenda

Exercises
Why C♯
Curriculum
TDD
.NET (Core)
C♯
Visual Studio Code
Visual Studio 2019

Exercises

Correction:

Exercise01 will count towards 8 of 10

Submission (individual)

Technical setup (VSCode, fork, git)

Group members

Why C♯

Stack Overflow Annual Developer Survey
2019 – 90,000 respondents

Popularity (professional developers):

- C♯: 31.9%
- ASP.NET: 27.2%
- .NET: 38.1%
- .NET Core: 24.5%
- Microsoft SQL Server: 34.4%
- Visual Studio Code: 34.9% (top 1)

Source: https://insights.stackoverflow.com/survey/2019

Why C#

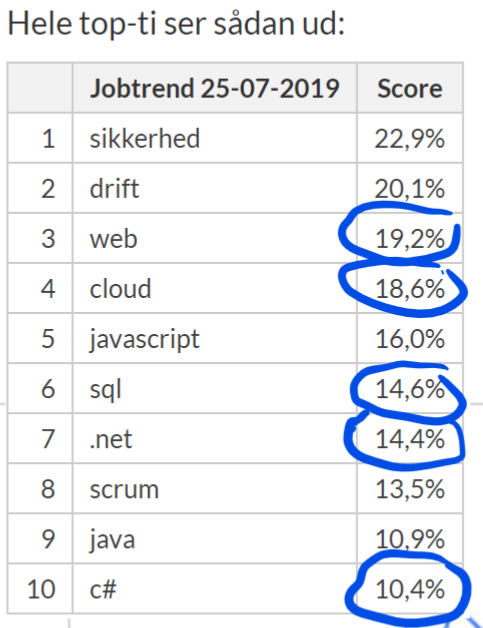
Love:

- C#: 67.0%
- ASP.NET: 64.9%
- .NET: 61.0%
- .NET Core: 77.2%
- Microsoft SQL Server: 57.5%

% of developers who are developing with the language or technology and have expressed interest in continuing to develop with it

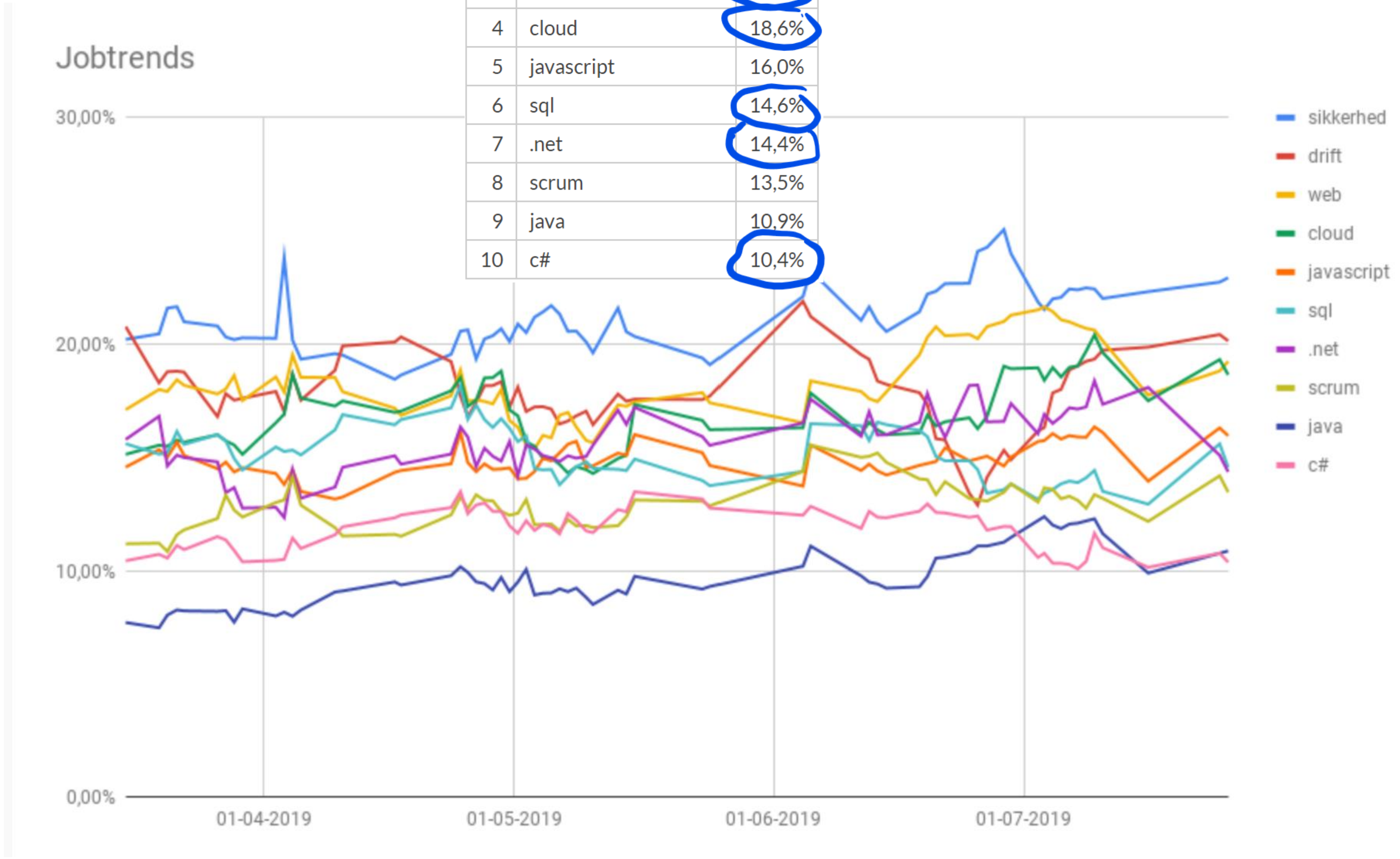
Source: https://insights.stackoverflow.com/survey/2019

Job trends



Hele top-ti ser sådan ud:

	Jobtrend 25-07-2019	Score
1	sikkerhed	22,9%
2	drift	20,1%
3	web	19,2%
4	cloud	18,6%
5	javascript	16,0%
6	sql	14,6%
7	.net	14,4%
8	scrum	13,5%
9	java	10,9%
10	c#	10,4%



Source: https://www.version2.dk/artikel/jobtrends-java-slaar-c-paa-maalfoto-1088564

Udvikler vild med C#: Ingen grund til at kode i Java nogensinde igen

(Illustration: Bigstock/Photosvit)

Seniorudvikler i konsulentfirma er krystalklar i mælet: C# og .Net er bare mindre bøvlet end Java. Og det er fordi, der kun er én leverandør bag platformen.

Source: https://www.version2.dk/artikel/udvikler-vild-med-c-ingen-grund-at-kode-java-nogensinde-igen-1088651

Curriculum

C♯ - Test Driven Development
Generics
Lambdas and Linq
Data access (SQL + Entity Framework)
Asynchronous and parallel processing
ASP.NET Core Web API
Design Patterns in Practice
Mobile apps with UWP and Xamarin.Forms
Security
Cloud

Test-Driven Development

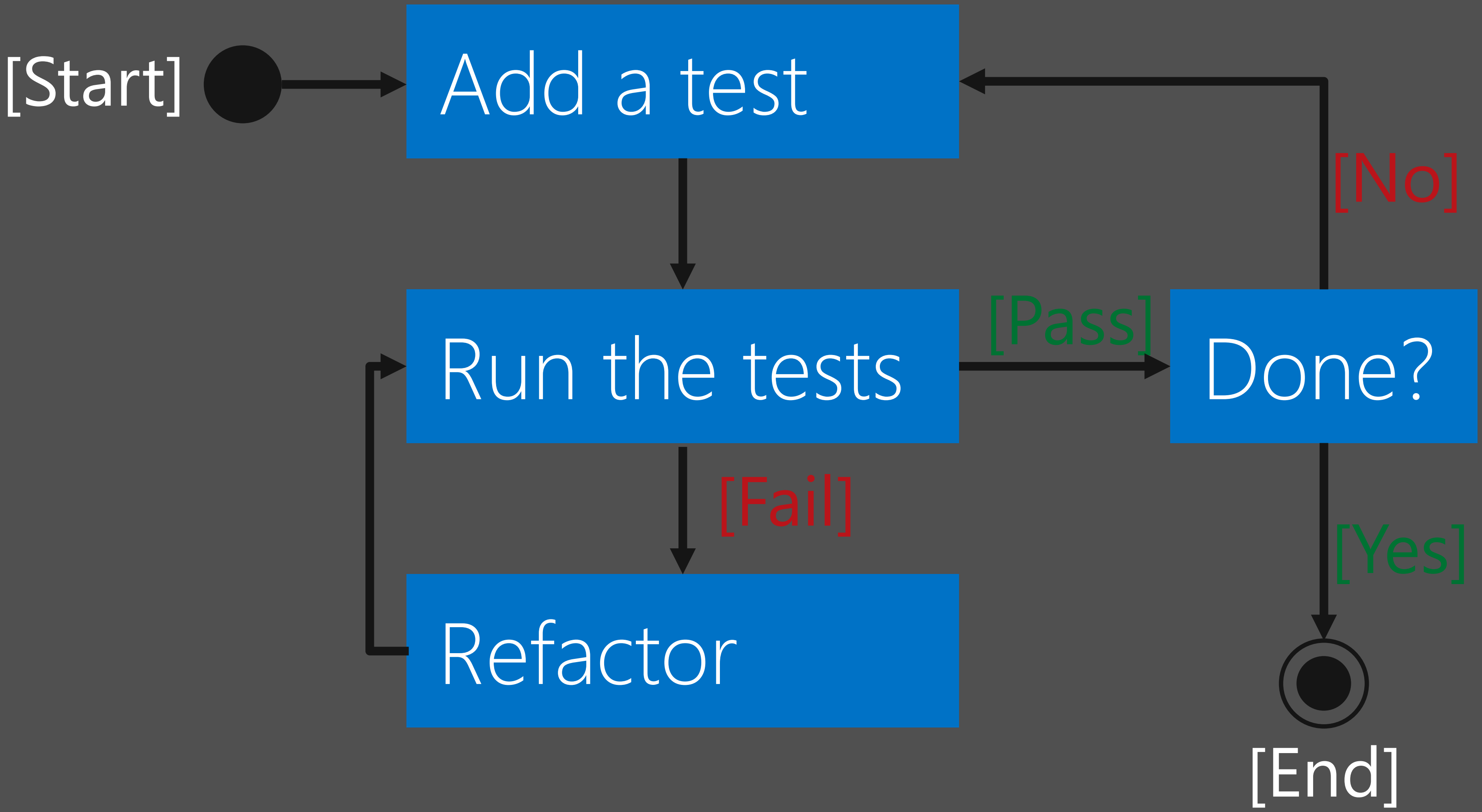
What?

Why?

How?

How?

RED-GREEN-REFACTOR



Microsoft
.NET

A brief introduction

.NET Framework

Versions

1.0 Visual Studio .NET (2002)

1.1 Visual Studio .NET 2003

2.0 Visual Studio 2005

3.0 (2006)

3.5 Visual Studio 2008 (2007)

4.0 Visual Studio 2010

4.5 Visual Studio 2012

4.5.1 Visual Studio 2013

4.6 Visual Studio 2015

4.7 Visual Studio 2017

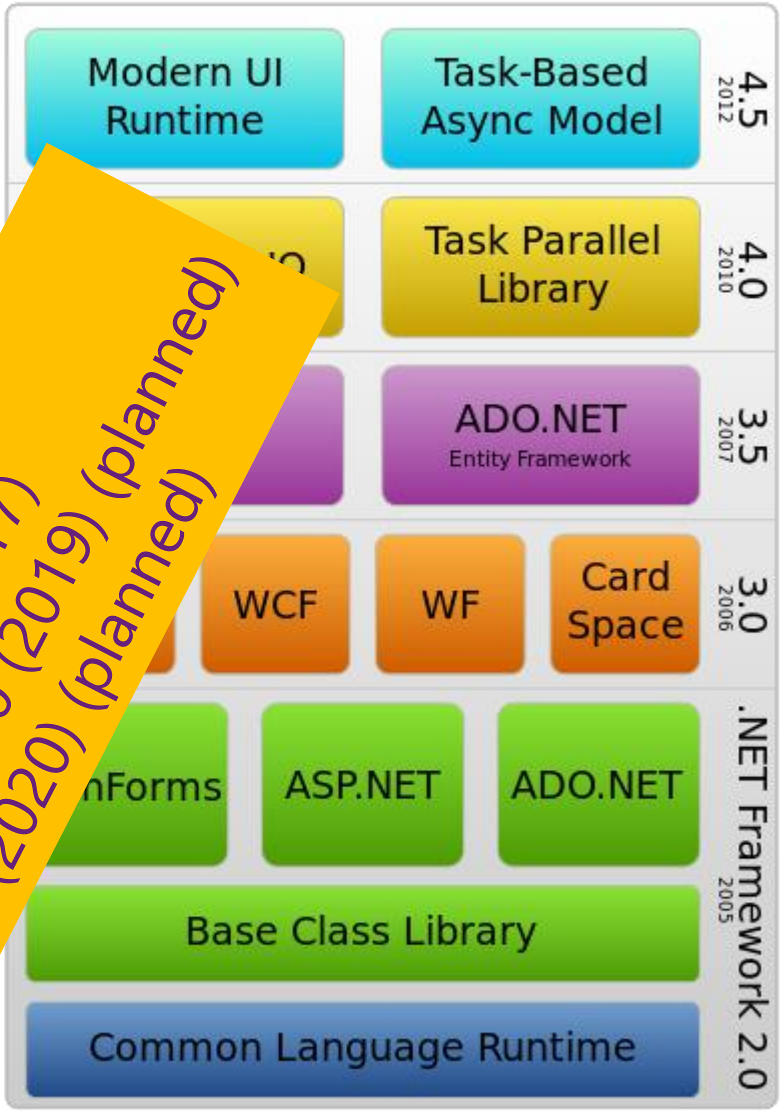
4.8 Visual Studio 2019

.NET Core 1.0 (2016)

.NET Core 2.0 (2017)

.NET Core 3.0 (2019) (planned)

.NET 5 (2020) (planned)



The .NET Framework Stack

C♯

Show me the CODE!!!

C♯ is intended t… mple, modern, general-purpose, o… nted programming language.

Ecma International (2006)

```
mkdir BDSA2019.Lecture01
cd BDSA2019.Lecture01

mkdir BDSA2019.Lecture01
mkdir BDSA2019.Lecture01.Tests

cd BDSA2019.Lecture01
dotnet new console
cd ..

cd BDSA2019.Lecture01.Tests
dotnet new xunit

dotnet new sln
dotnet sln add BDSA2019.Lecture01
dotnet sln add BDSA2019.Lecture01.Tests
dotnet add reference ..\BDSA2019.Lecture01

cd ..
code .
```

C♯

Language Basics

Naming Conventions

Composed names

`currentLayout`, `CurrentLayout`

Variables and fields

`vehicle`, `leftElement`

Private fields

`_vehicle`, `_leftElement`

Methods

`CurrentVehicle()`, `Size()`

Properties

`Pi`, `Name`, `Size`

Classes

`MyClass`, `List<T>`

Interfaces

`IException`, `IObserver`

https://docs.microsoft.com/en-us/dotnet/standard/design-guidelines/naming-guidelines

`Value Types can never be null!`

Value Types

Holds a value – assignment copies the value

Struct

- Numeric types
    - Integral types
    - Floating point types
    - Decimal
- bool

Integral
`byte, sbyte,`
`short, ushort,`
`Char,`
`int, uint,`
`long, ulong`
	Floating point
	`float`
	`Double`

	Decimal `decimal`

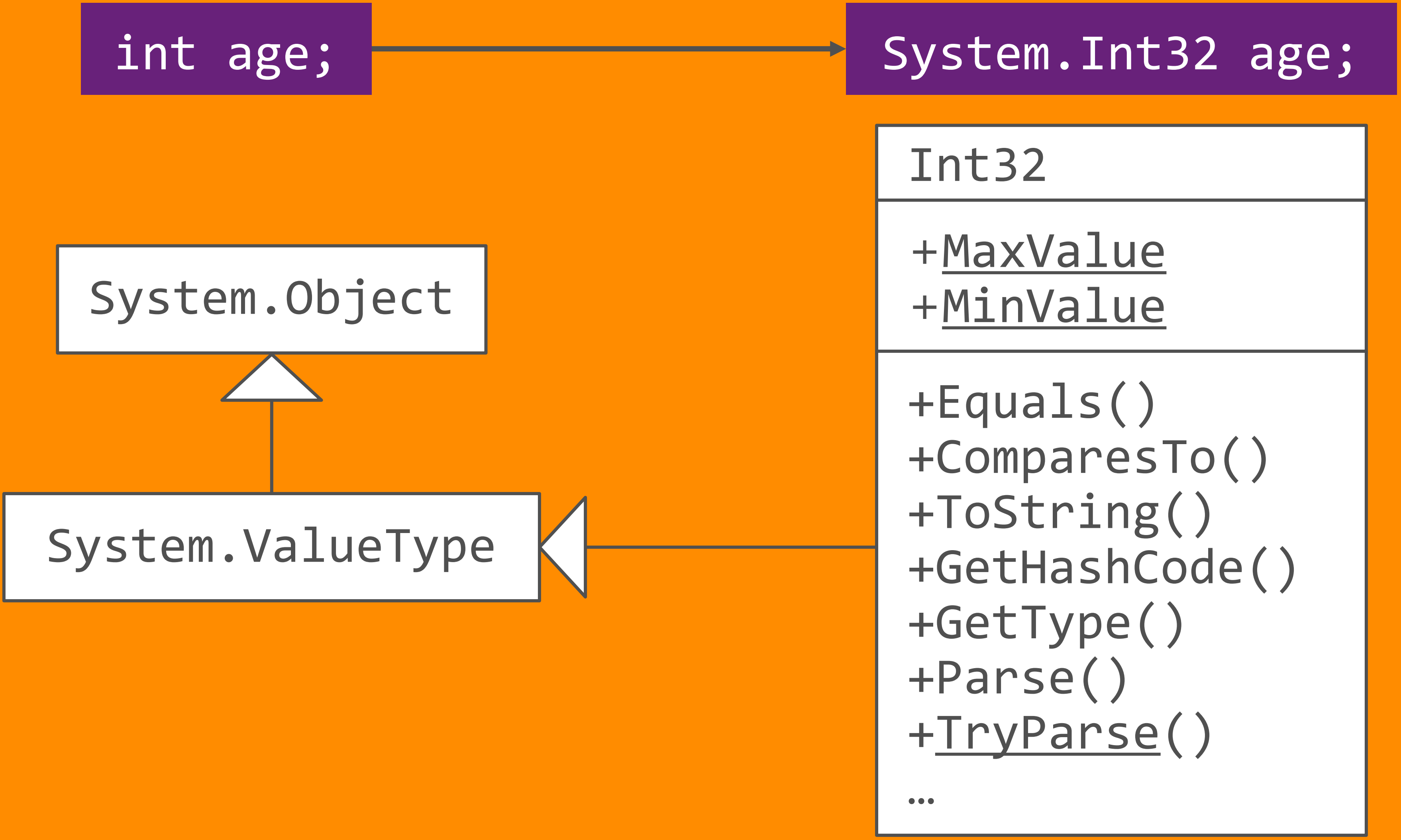
Enumeration

```
enum Days {Sat, Sun, Mon, Tue, Wed, Thu, Fri};
```

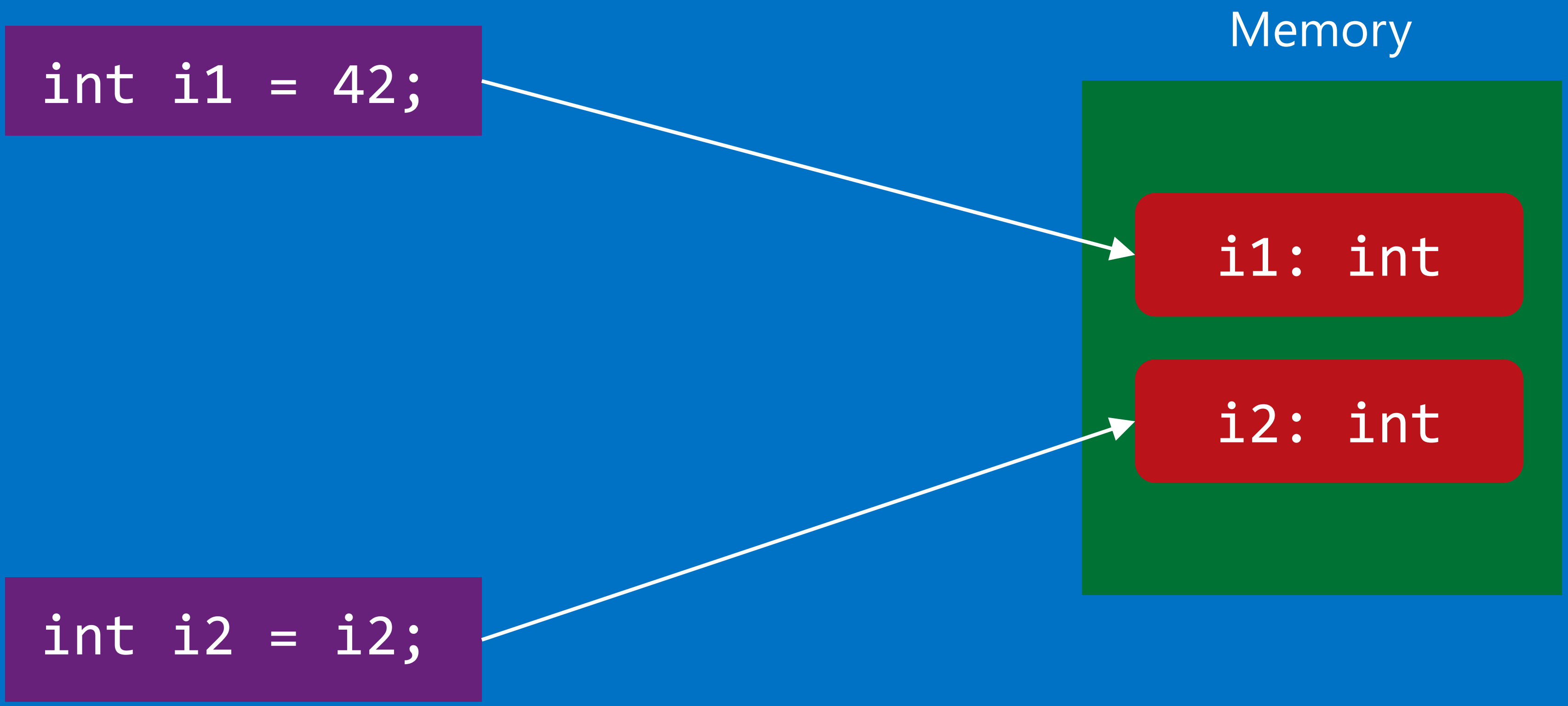
User defined structs

```
System.Guid
System.Drawing.Point
System.DateTime
System.Numerics.BigInteger
System.Numerics.Complex
```

Value Types


int age;
System.Int32 age;
Int32
+MaxValue
+MinValue
+Equals()
+ComparesTo()
+ToString()
+GetHashCode()
+GetType()
+Parse()
+TryParse()
…
System.Object
System.ValueType

Value Types

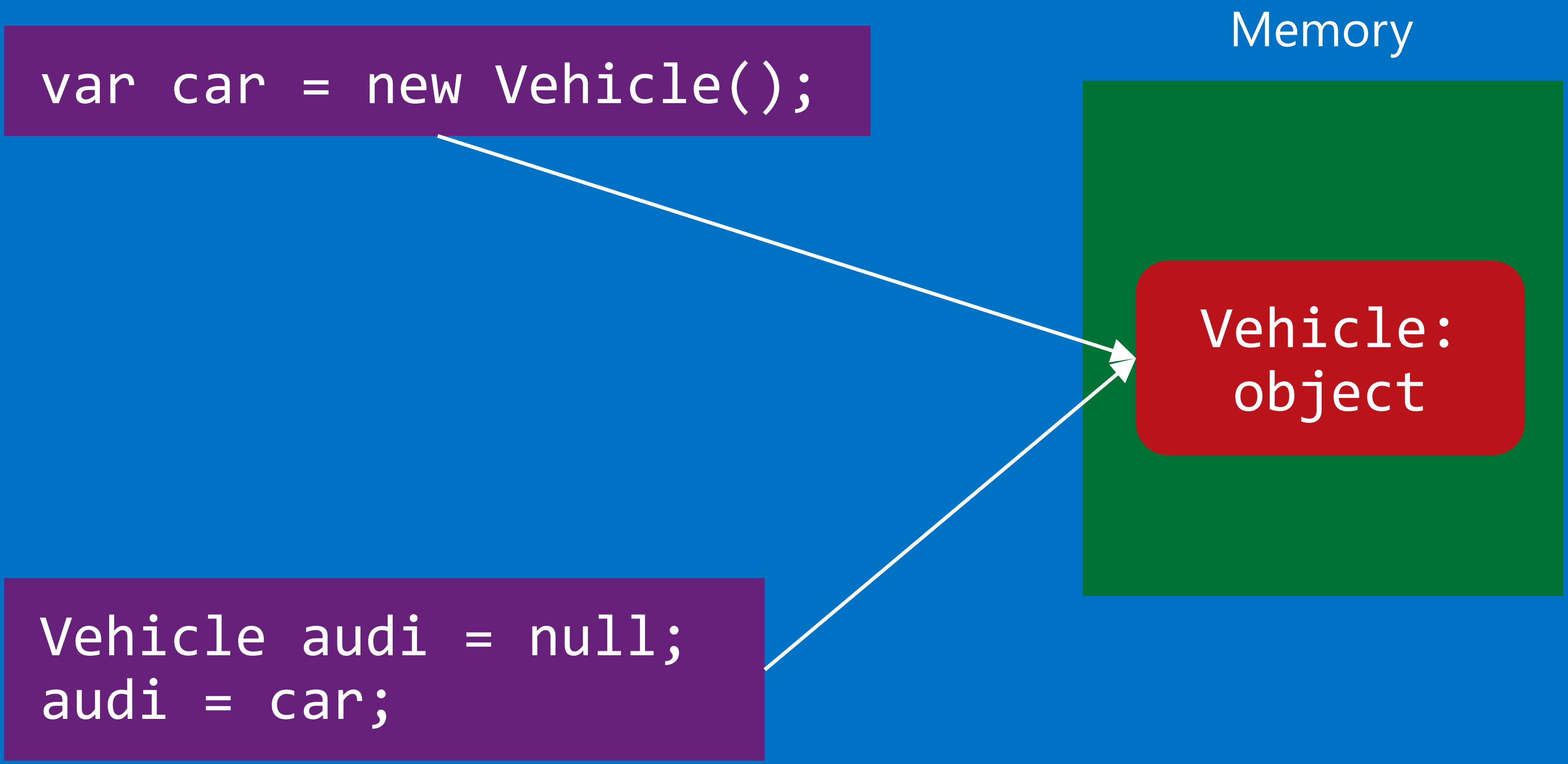


```
int i1 = 42;
```

```
int i2 = i2;
```

`ReferenceEquals` is always *`false`*

Reference Types



```
var car = new Vehicle();
```

```
Vehicle audi = null;
audi = car;
```

Reference Type Equality

ReferenceEquality:
Person p1 = new Person(“Joe”);
Person p2 = new Person(“Joe”);
Person p3 = p2;
ReferenceEquality(p1, p2) = false
ReferenceEquality(p2, p3) = true;

In C# the == operator is “equal” to reference equality. (Can be overridden)

Value Equality (for reference types)
p1.Equals(p2) = true; (Can be overridden)

Value Type Equality

Equals the same as for reference types

object.ReferenceEquality will always return false for value types

== operator is overridden so it does value equality

String Interning

```
string a = "Peter";
string b = "Peter";

a.Equals(b);     ==> true

a == b;          ==> true

object.ReferenceEquals(a, b); ==> true
```

String Interning

The String Type is Immutable – assigning creates a new value…

Local Variable Type Inference

var *identifier* = *expression*;

var is a keyword, not a type

Enumeration

```
public enum Day { Mon, Tue, Wed, Thu, Fri, Sat, Sun }

public enum Month : uint { Jan = 1, Feb, Mar, }

public enum Color : uint { Red = 0xFF0000,
                           Green = 0x00FF00,
                           Blue = 0x0000FF }

Vehicle car = new Vehicle();
car.Color = Color.Red;

Console.Write(Day.Mon);
```

Array stuff

```
int[] intArray = new int[4];

double[,] doubleArray = new double[4, 5];

int[,] array1 = {{1,2],{3,4]};

int value1 = intArray[0];

double value2 = doubleArray[0,1];

Console.WriteLine(array1[1,1]);
```

Arrays are 0-based

String stuff

```
public static void Main(string[] args)
{
    var name = "Anders";
    var argument = args[0];

    Console.WriteLine(10 + " hello " + name + argument);

    Console.WriteLine("Hello {0}", name);

    Console.WriteLine($"Hello {name}");

}
```

Automatic conversion

String Format

String Interpolation (preferred)

Compile, test, and run from the Command Line

```
$ dotnet build
$ dotnet test
$ dotnet run
```