

```

// R_FatNumForName ( SKYFLATNAME )
// IF NO WOULD LEVELS, NOT EXIT
// on the current episode, and the game version.
// code == commercial
// gamemode == pack (tr)
// gamemode == pack (plut)
}

texture = R_TextureNumForName ("SKY3");
memset < 12
texture = R_TextureNumForName ("SKY11");
void B_GoCompleted (void)
{
    int i;
    if (gamemap < 21)
        skytexture = R_TextureNumForName ("SKY2");
    gamemode = ga_nothing;
}

// for time calculation
for (i=0; i<MAXPLAYERS; i++)
{
    if (i==0)
        // Player not level 0
    }

// This was quite messy with SPECIAL and commented parts.
// supposedly backs to make the latest edition work.
// it might not work properly.
if (episode < 1)
    episode = 1;

// if (gamemode == retail)
{
    if (episode < 4)
        episode = 4;
    else if (gamemode == shareware)
    {
        if (episode > 3)
            episode = 1; // only start episode 1 on shareware
        else
        {
            if (episode > 3)
                episode = 3;
        }
    }
    if (map < 1)
        map = 1;
    if (map > 90)
        map = 9;
    M_ClearRandom ();

    if (skill == sk_nightmare || respawnparm)
        respawnmonsters = true;
    else
        respawnmonsters = false;

    if (respawn || skill == sk_nightmare && gameskill != sk_nightmare)
    {
        for (i=S_SARG_RUN1; i<=S_SARG_PAIN2; i++)
            mobinfo[MT_BRUISERSHOT].speed = 20*FRACUNIT;
            mobinfo[MT_TROOPSHOT].speed = 20*FRACUNIT;
    }
    else if (skill != sk_nightmare && gameskill == sk_nightmare)
    {
        for (i=S_SARG_RUN1; i<=S_SARG_PAIN2; i++)
            mobinfo[MT_BRUISERSHOT].speed = 15*FRACUNIT;
            mobinfo[MT_TROOPSHOT].speed = 10*FRACUNIT;
    }

    // force players to be initialized upon first level load
    for (i=0; i<MAXPLAYERS; i++)
        players[i].playerstate = PST_REBORN;

    usergame = true; // will be set false if a demo
    paused = false;
    demoplayback = false;
    automapactive = false;
    viewactive = true;
    gamemode = episode;
    gamemap = map;
    gameskill = skill;
    viewactive = true;
}

```

Course Introduction, Test Driven C#

Rasmus Lystrom
External Associate Professor
ITU

Me

Rasmus Lystrøm, rasmus@lystroem.dk

39 yrs. Old

Wife Katrine,
married for 2
years

Daughters:
Alma, 8.

Laura, 1

From Aarhus

Live in
Vanløse

2013

Consultant @
Microsoft

2007-2016

[d&n](#)

2001-

[Lystrøm ApS](#)

2011-2013

Teacher @
DTU

2013-

Danish Army
(Reserve)

Captain

2008-2012

M.Sc. IT @
ITU

Thesis:
Forecalc –
Developing a
core
spreadsheet
implementation
in F#

1998-2001

B.Sc. Warfare
@ Hærens
Officersskole

1996-2008

Danish Army
1st Lieutenant

Agenda

TDD

.NET and the C# language

Visual Studio 2015

Test Driven C#

Test-Driven Development

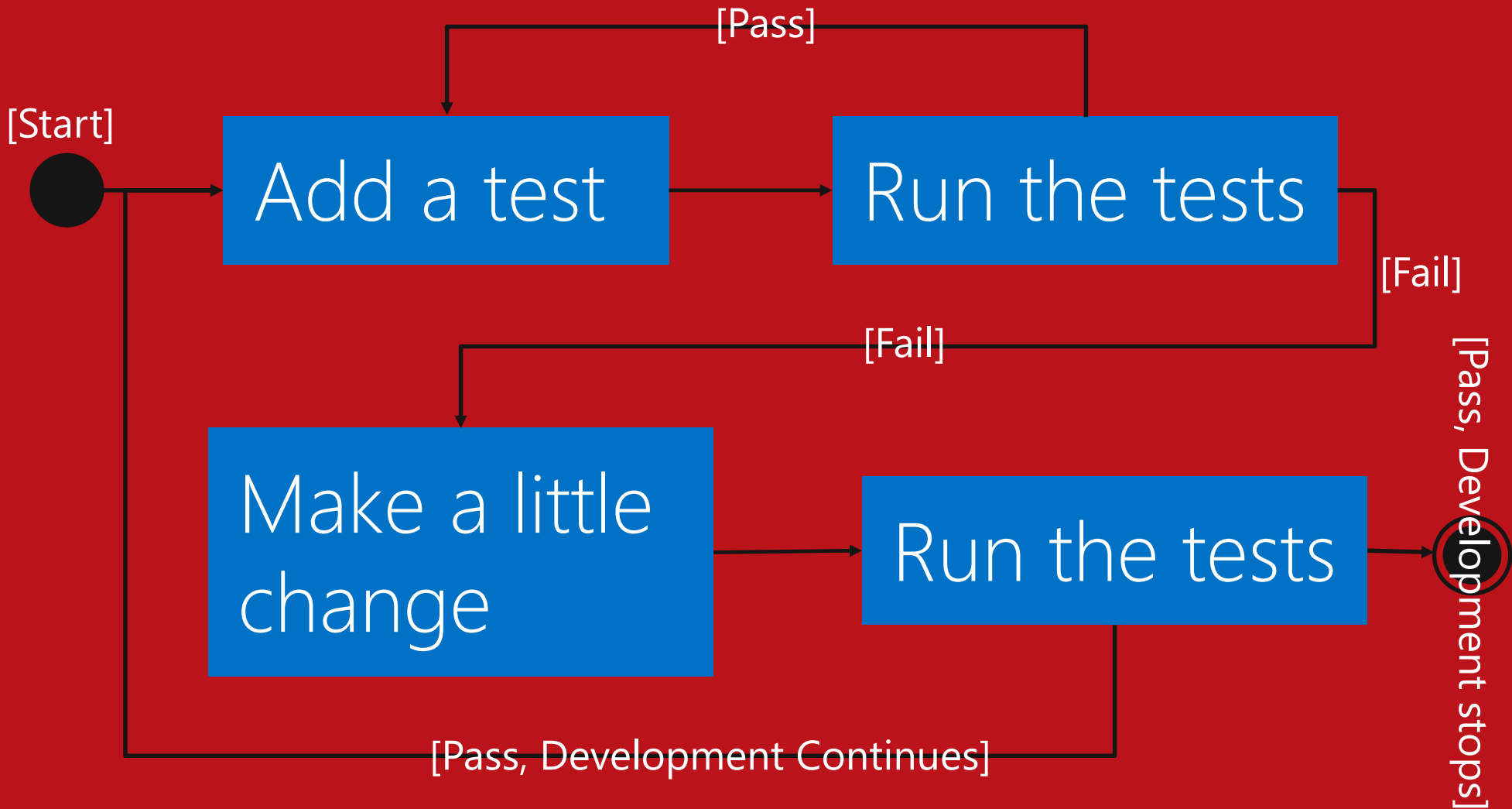
What?

Why?

How?

How?

RED-GREEN-REFACTOR

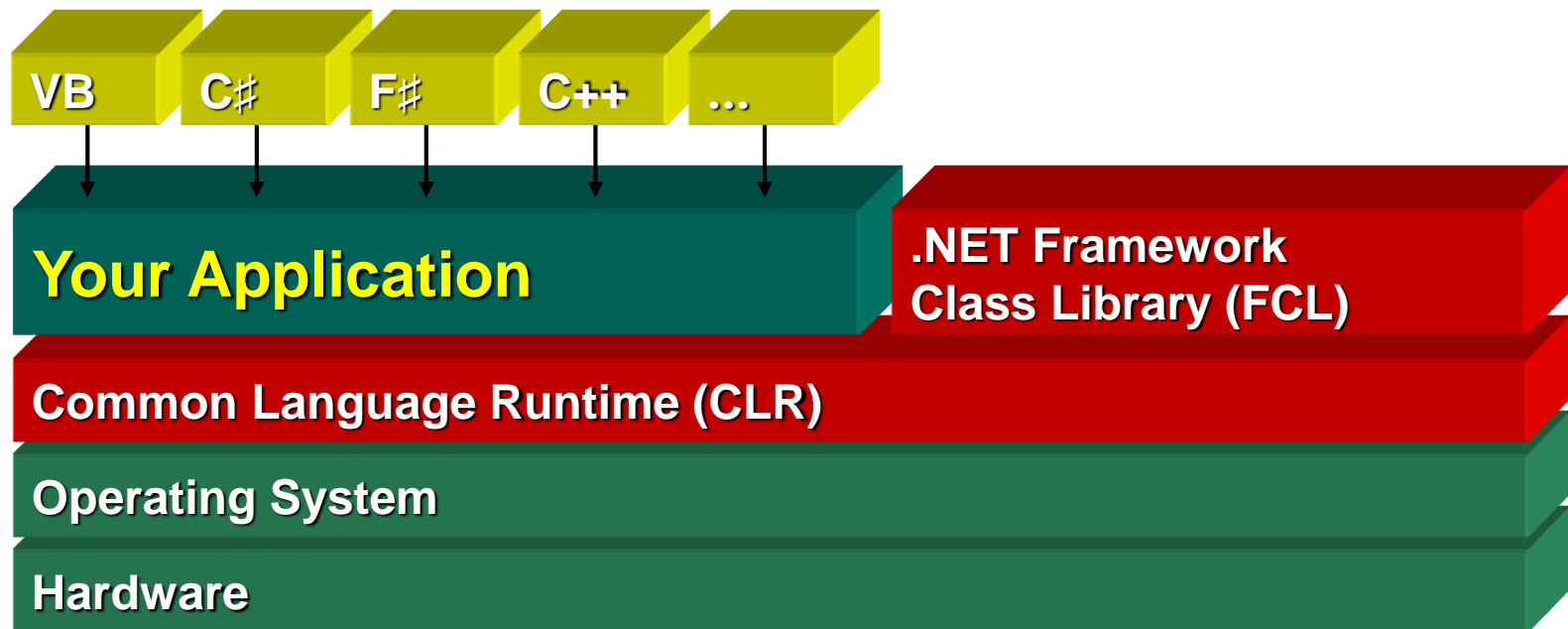




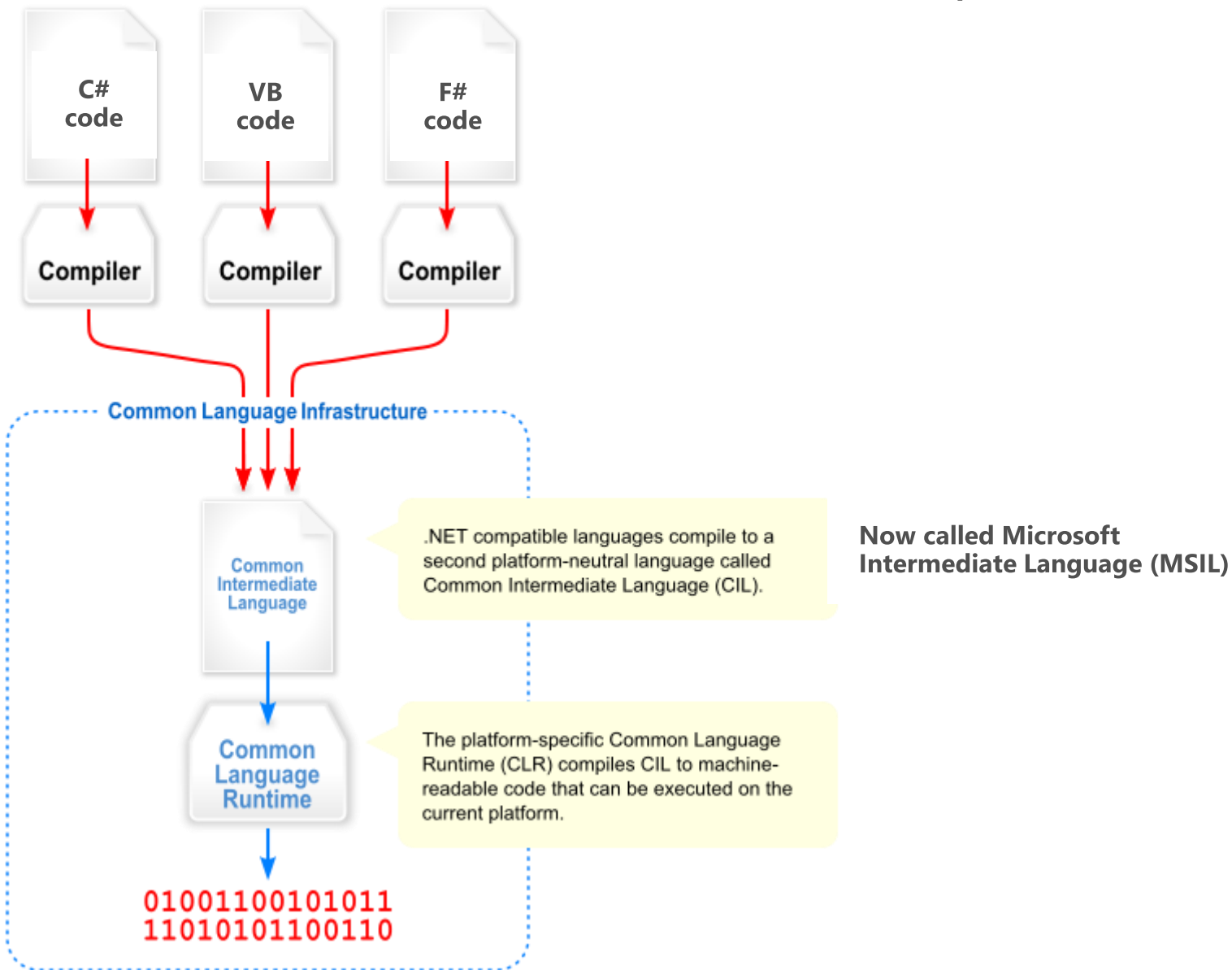
A brief introduction

.NET Framework

.NET offers multiple languages, a large class library, driven by a virtual machine



.NET Framework Compilation



.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

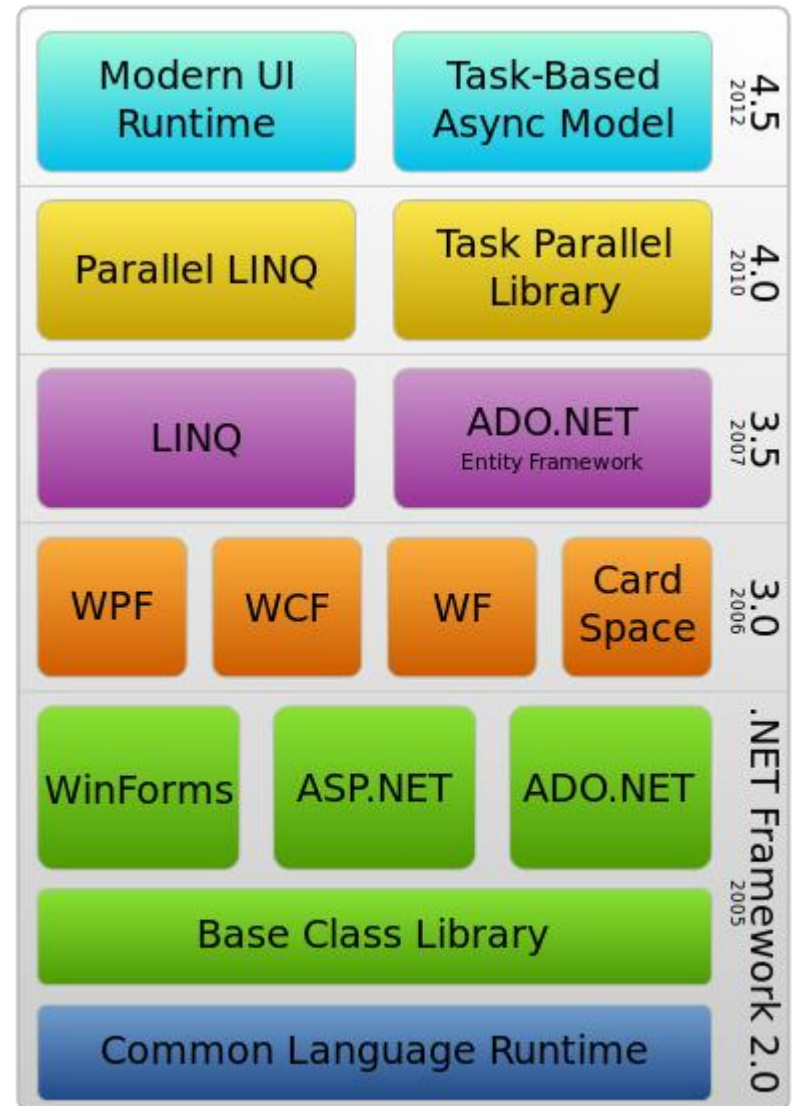
4.5.2 (2014)

4.6 Visual Studio 2015

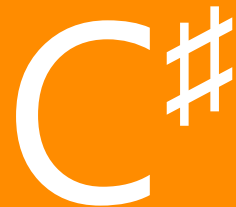
4.6.1 (2015)

4.6.2 (2016)

.NET Core 1.0 (2016)



The .NET Framework Stack



C# is intended to be a simple, modern, general-purpose, object-oriented programming language.

Ecma International (2006)

C#

show me the CODE!!!

Language Basics

Naming Conventions

Composed names

currentLayout, CurrentLayout

Properties

Pi, Name, Size

Variables and fields

vehicle, leftElement

Classes

MyClass, List<T>

Private fields

_vehicle, _leftElement

Interfaces

IException, IObservable

Methods

CurrentVehicle(), Size()

[http://msdn.microsoft.com/en-us/library/ms229002\(v=vs.110\).aspx](http://msdn.microsoft.com/en-us/library/ms229002(v=vs.110).aspx)

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;
```

System.Object

System.ValueType

Int32

+MaxValue
+MinValue

+Equals()
+CompareTo()
+ToString()
+GetHashCode()
+GetType()
+Parse()
+TryParse()
...

Value Types

```
int i1 = 42;
```

Memory

i1: int

i2: int

```
int i2 = i2;
```

ReferenceEquals is always *false*

Reference Types

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

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

Memory

Vehicle:
object



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
```

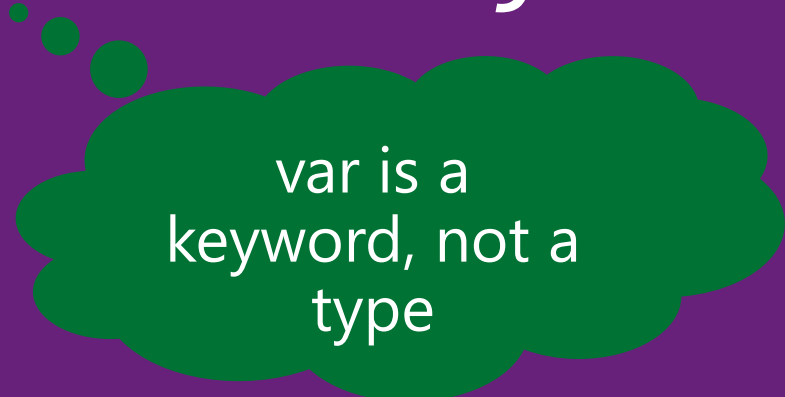


String
Interning

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);
}
```



Automatic
conversion

Compile from Command Line

Developer Command Prompt for VS2015:

```
C:\[program_dir]> csc
```

Command Line Options

Compiler Option	Short Form	Description
/target:exe /target:library	/t:exe /t:library	Compile to an executable (.exe file) (default) Compile to a library (.dll file)
/reference:Lib.dll	/r:Lib.dll	Include reference to library Lib.dll
/main:MyClass	/m:MyClass	Method Main in MyClass is the entry point
/debug	/d	Add debugging information

```
C:\Program Files (x86)\Microsoft Visual Studio 14.0>csc /?
```