# LINQ Best Practices

#### Discovering the Power of LINQ



Mark Heath
Software Architect

@mark\_heath www.markheath.net



# LINQ Best Practices

Version Check



#### Version Check



#### This version was created by using:

- .NET 6
- LINQPad 7



#### Version Check



#### This course is 100% applicable to:

- .NET 6 through .NET 7

#### This course is 99% applicable to:

- All versions of .NET since 3.5
- IAsyncEnumerable
- Newer extension methods (e.g. MaxBy)

# Why LINQ?

It's been around a while ...

C#3 released in 2007

... but there's more to LINQ than you think!



### Language Features

Lambda Expressions

**Extension**Methods

Anonymous Types

Query Expression Syntax

Generics

yield and var Keywords

Best practice: Know the language features that power LINQ



```
// statement lambdas can contain several lines of code
customers.Where(c => {
    var hasEmail = c.Email != null;
    return hasEmail; })

customers.Where(c => c.Email != null)
```

#### Lambda Expressions

Easily pass anonymous functions to methods

```
static class StringExtensions
{
    public static string Shout(this string s)
    {
       return s.ToUpper() + "!!!!";
    }
}
```

#### Extension Methods

Extend any type with additional methods

LINQ provides extension methods on IEnumerable<T>

Connect these extension methods together into "pipelines"





#### More about extension methods:

**C# Extension Methods** 

Elton Stoneman

# Misconception: "LINQ pipelines are for show-offs"

LINQ should make your code more readable, not less



```
var x = new { Author = "Mark Seemann", Title = "Dependency
Injection in .NET" };
var y = new { Author = "Martin Fowler", Title = "Patterns
of Enterprise Architecture" };
var z = new { Author = "Robert Martin", Title = "Clean
Code", Pages = 245 }; // NOT the same type as x & y

var books = new[] { x, y };
```

#### Anonymous Types

Create new types without explicitly declaring a class

```
var author = "Elton Stoneman";
var title = "Docker on Windows";

var book = new { Author = author, Title = title };

// inferred property names:
var book = new { author, title };
```

## Anonymous Types

Can infer property names

Great for passing state through LINQ pipelines

Often preferable to tuples



Query Expression Syntax

Similar to SQL

Many new keywords



# Misconception: "LINQ is just for database queries"

There are several LINQ "providers".
e.g. LINQ to **Entities**LINQ to **Objects** 



## Query Expression Syntax

Can be used with any LINQ provider, including LINQ to objects

Sometimes easier to read than chained extension methods



```
var numbers = new List<int>();
var messages = new List<string>();
var customers = new List<Customer>();
```

## Generics and the yield Keyword

Create classes and methods that can work with any type



```
public static IEnumerable<T> DoubleUp<T>(this IEnumerable<T> source)
{
    foreach (var s in source)
    {
        yield return s;
        yield return s;
    }
}
```

## Generics and the yield Keyword

Create classes and methods that can work with any type

The LINQ extension methods are generic

You can create your own generic methods



#### Language Features

Lambda Expressions

**Extension**Methods

Anonymous Types

Query Expression Syntax

Generics

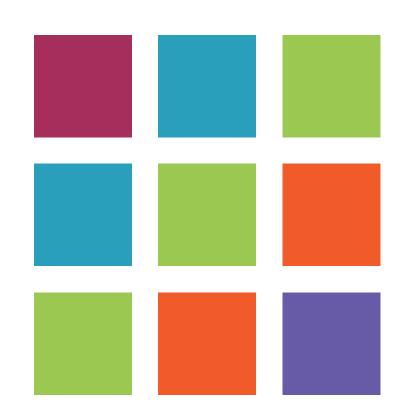
yield and var Keywords

**Expression Trees** 



# Collections are Everywhere!





In-memory objects

**Database queries** 

Algorithmically generated data

## LINQ and Functional Programming

LINQ applies many powerful functional programming techniques in C#

Learning LINQ will increase your understanding of functional programming



## No slides – LINQPad & VS

#### Summary



#### LINQ is awesome

#### Many powerful C# language features

- Learn how to use them!

Applicable to almost all programs



### What to Expect in the Rest of This Course

Thinking in Patterns

Unleashing the Power of Pipelines

Writing Clean and Readable Code

**Extending LINQ** 

Avoiding Unnecessary Work

Optimizing Performance

Test and Debug Effectively Working with Asynchronous Streams

**Embracing a Functional Style** 

Put it into practice with some "LINQ Challenges"

