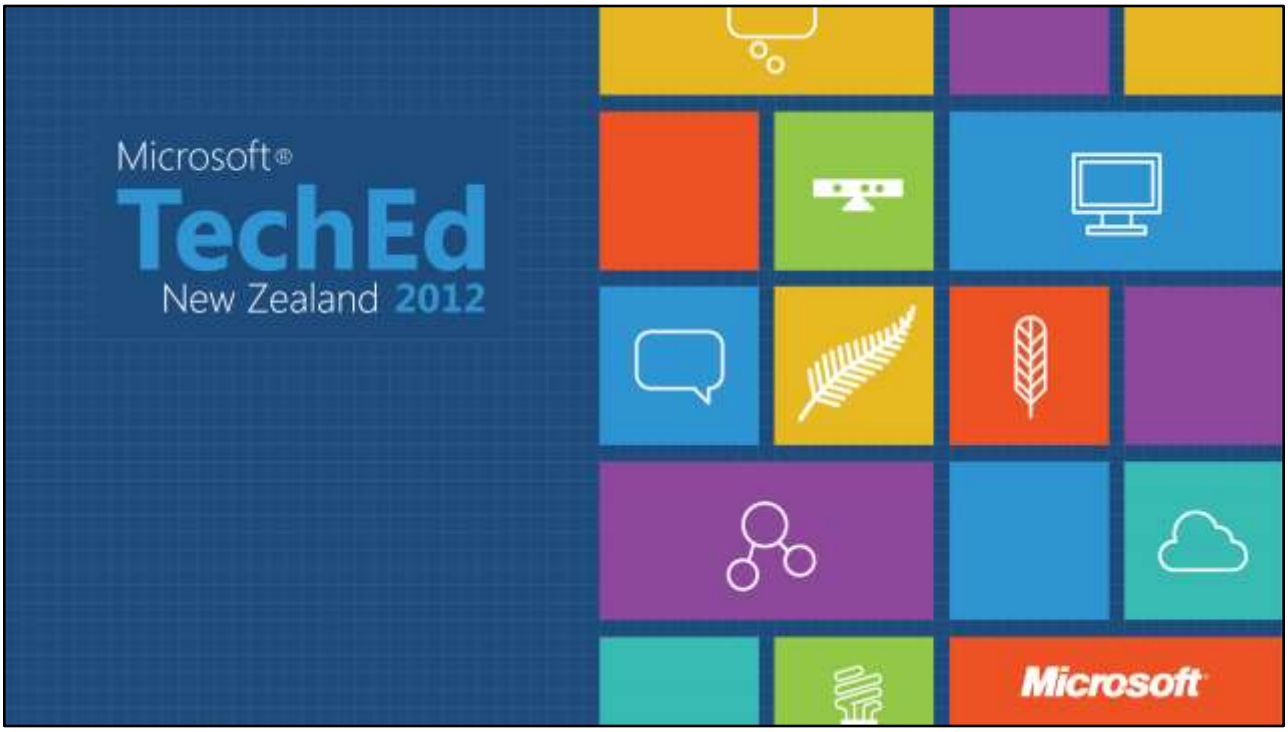

Microsoft TechEd New Zealand 2012

DEV307: Single Line Solutions in .NET

Bevan Arps
Senior Analyst Programmer
Reserve Bank of New Zealand

Microsoft TechEd 2012

Agenda

Framework
Third party
Do it yourself

Microsoft TechEd 2012

Goals

One to use
One to try
One to write

What should you take away from this presentation?

One improvement that you know you can use

One improvement that you want to try

One improvement that you want to write yourself.

"You have a finite number of keystrokes in your hands before you die"

Scott Hanselman

Microsoft TechEd 2012

Property binding

Implementing INotifyPropertyChanged

INotifyPropertyChanged

2.0
.NET VERSION

```
public string FullName
{
    get { return mFullName; }
    set
    {
        if (!Equals(mFullName, value))
        {
            mFullName = value;
            OnPropertyChanged("FullName");
        }
    }
}
```

Very common

Includes magic strings, but very hard to avoid them in .NET 2.0

```
private void OnPropertyChanged(string property)
{
    var handlers = mPropertyChanged;
    if (handlers != null)
    {
        handlers(this, new PropertyChangedEventArgs(property));
    }
}
```

Here's the classic implementation – passed the property name, triggers the event

INotifyPropertyChanged

3.5
.NET VERSION

```
public string FullName
{
    get { return mFullName; }
    set
    {
        if (!Equals(mFullName, value))
        {
            mFullName = value;
            OnPropertyChanged(() => FullName);
        }
    }
}
```

Lambda expressions in .NET 3.5 give us another way

Instead of being compiled to code, the compiler gives us a data structure we can manipulate

```
private void OnPropertyChanged<T>(
    Expression<Func<T>> propertyDelegate)
{
    var m = (MemberExpression)propertyDelegate.Body;
    var handlers = mPropertyChanged;
    if (handlers != null) {
        handlers( this,
            new PropertyChangedEventArgs(m.Member.Name));
    }
}
```

Mostly the same implementation, but with the lambda expression data structure passed in instead of a string.

INotifyPropertyChanged

4.5
.NET VERSION

```
public string FullName
{
    get { return mFullName; }
    set
    {
        if (!Equals(mFullName, value))
        {
            mFullName = value;
            OnPropertyChanged();
        }
    }
}
```

In .NET 4.0 we can write it this way.

How? Compiler magic!

```
private void OnPropertyChanged(
    [CallerMemberName]string property = null)
{
    var handlers = mPropertyChanged;
    if (handlers != null)
    {
        handlers(this, new PropertyChangedEventArgs(property));
    }
}
```

The new attribute [CallerMemberName] asks the compiler to automatically fill in the name of the calling method. There are a whole bunch of these call site attributes, including CallerFilePath and CallerLineNumber,

Microsoft TechEd 2012

Nullable<T>

Working with optional values

Nullable<T>

```
DateTime searchStart;
if (Start.HasValue)
{
    searchStart = Start.Value;
}
else
{
    searchStart = new DateTime(1900,1,1);
}
```

An alternative to having a static member marked with [ThreadStatic] attribute – has the advantage that the value is both local to the object, and local to the thread.

Nullable<T>

```
var searchStart
    = Start.HasValue ? Start.Value : new DateTime(1900,1,1);
```

An alternative to having a static member marked with [ThreadStatic] attribute – has the advantage that the value is both local to the object, and local to the thread.

Nullable<T>

2.0
.NET VERSION

```
var searchStart
    = Start ?? new DateTime(1900,1,1);
```

An alternative to having a static member marked with [ThreadStatic] attribute – has the advantage that the value is both local to the object, and local to the thread.

Nullable<T>

2.0
.NET VERSION

```
var searchStart
    = Start.GetValueOrDefault(new DateTime(1900,1,1));
```

An alternative to having a static member marked with [ThreadStatic] attribute – has the advantage that the value is both local to the object, and local to the thread.

Microsoft TechEd 2012

Files & paths

Special folders and path manipulation

Environment.GetFolderPath

4.0
.NET VERSION

```
var programFilesFolder
    = Environment.GetFolderPath(
        Environment.SpecialFolder.ProgramFiles);
```

New method to obtain special folders on the current machine – Program Files, Common Files, Windows, Desktop, Application Data and so on.

Path.Combine

4.0
.NET VERSION

```
var programFilesFolder
    = Environment.GetFolderPath(
        Environment.SpecialFolder.ProgramFiles);
var applicationFolder = "WordTutor";
var resourcefile = "Example.txt";
var filePath
    = Path.Combine(
        programFilesFolder, applicationFolder, resourcefile);
```

New overload for .NET 4 – takes three path segments and stitches them together. Two parameter version has been around since .NET 1.1

Useful because it handles leading and trailing \ properly

Path.GetFullPath

1.1
.NET VERSION

```
var baseDir = "C:\\Windows\\System32";
var relativeDir = "..\\..\\Program Files";
var filePath = Path.Combine(baseDir, relativeDir);
var actualPath = Path.GetFullPath(filePath);
```

Gives a rational, direct (optimal) path, to a file or directory. Handles funky edge cases properly.

Microsoft TechEd 2012

Strings

String manipulation made easy

String Constructor

2.0
.NET VERSION

```
// new String(char c, int count)
var line = new String('-', 32);
```

Useful when you need a padding string of a particular length, or for well formatted output to the console.

String.Concat

4.0
.NET VERSION

```
// Generates string: HewyDewyLouie
var children = new List<string> { "Hewy", "Dewy", "Louie" };
var result = String.Concat(children);
```

Joins a sequence of strings together into a single string.

Alternative to the version that makes an array

String.Join

4.0
.NET VERSION

```
// Generates string: Hewy, Dewy, Louie
var children = new List<string> { "Hewy", "Dewy", "Louie" };
var result = String.Join(", ", children);
```

Joins a sequence of strings together, with a specified delimiter between each value. New overload in .NET 4 takes an IEnumerable<string>, avoiding the need to convert to an array

String.IsNullOrWhitespace

4.0
.NET VERSION

```
string value = GetValue();
if (string.IsNullOrWhiteSpace(value))
{
    // value is blank
}
```

Test a string to see if it contains content – like IsNullOrEmpty, but also handles whitespace

Microsoft TechEd 2012

Linq

More than Select() and Where()

Enumerable.Repeat

3.5
.NET VERSION

```
// Generate a sequence containing "sample" 5 times
var sequence = Enumerable.Repeat("sample", 5);
```

Repeat a given value a number of times

Enumerable.Zip

4.0
.NET VERSION

```
// Generates sequence:
// Hewy; Dewy; Louie;
var children = new List<string> { "Hewy", "Dewy", "Louie" };
var sequence
    = children.Zip(
        Enumerable.Repeat("; ", 3),
        (name, sep) => name + sep);
```

Combine two sequences together – each pair of items gets combined by the passed lambda expression

Enumerable.TakeWhile

4.0
.NET VERSION

```
List<Player> players = FindPlayers();
var greatPlayers
    = players.OrderByDescending(p => p.Score)
        .TakeWhile(p => p.Score > 1000);
```

Take items from the start of the sequence as long as a predicate is true. Includes the last item that passes the test.

Microsoft TechEd 2012

Xml

<path>Easy</path>

XElement

3.5
.NET VERSION

```
var result = new Person()
{
    FullName = (string)element.Attribute("fullName"),
    FamilyName = (string)element.Attribute("familyName"),
    IsAlive = (bool)element.Attribute("isAlive"),
    BirthDate = (DateTime?)element.Attribute("birthDate")
};
```

No need to parse attribute (or element!) values – all the work has been done for you, by conversions written on the classes

XPathEvaluate

3.5
.NET VERSION

```
var namedSubsections
    = document.XPathEvaluate("section/subsection[@name]");
```

When working with XML, Xpath is a natural technique for queries and searches – and now it's easy to use.

Microsoft TechEd 2012

Helper Classes

ThreadLocal and Lazy

Lazy<T>

4.0
.NET VERSION

```
public Serializer Serializer
{
    // Creating a Serializer is expensive, so
    // we put it off until we really need it
    get { return mSerializerCache.Value; }
}

private Lazy<Serializer> mSerializerCache
    = new Lazy<Serializer>(() => new Serializer());
```

Easy way to defer creation of expensive resources until required

Lazy<T>

```
public Serializer Serializer
{
    // Creating a Serializer is expensive, so
    // we put it off until we really need it
    get { return mSerializerCache.Value; }
}

private Lazy<Serializer> mSerializerCache
    = new Lazy<Serializer>(
        () => new Serializer(),
        LazyThreadSafetyMode.ExecutionAndPublication);
```

And you can make it threadsafe with just an extra parameter

ThreadLocal

4.0
.NET VERSION

```
public WidgetFactory Factory
{
    get { return mFactory.Value; }
}

private static ThreadLocal<WidgetFactory> mFactory
    = new ThreadLocal<WidgetFactory>();
```

An alternative to having a static member marked with [ThreadStatic] attribute – has the advantage that the value is both local to the object, and local to the thread.

Microsoft TechEd 2012

MoreLINQ

Linq extensions from Jon Skeet

http://code.google.com/p/morelinq/

MaxBy

```
var players
    = new List<Player>
    {
        new Player("Adam", 1100),
        new Player("Brian", 500),
        new Player("Caleb", 750),
        new Player("Darcy", 950)
    };

// Find the player with the highest score
var winner = players.MaxBy(p => p.Score);
```

Find an item from the sequence that has a maximal score generated by the lambda

Concat

```
var uncle = "Donald";
var nephews = new List<string> { "Hewy", "Dewy", "Louie" };

// Create sequence with all four ducks
var ducks = uncle.Concat(nephews);
```

Add a single item to the start or end of a sequence

ForEach

```
List<Player> players = FindPlayers();

// Log all players who've played more than 100 games
players.Where(p => p.GamesPlayed > 100)
    .ForEach(p => LogPlayer(p));
```

Iterate over all the items in the sequence, applying an action

Pipe

```
List<Player> players = FindPlayers();

// Log all players who've played more than 100 games
// and average their scores
var averageScoreOfExperiencedPlayers
    = players.Where(p => p.GamesPlayed > 100)
        .Pipe( p => LogPlayer(p))
        .Average(p => p.Score);
```

Apply an action to every item in the sequence, while some other action pulls the items through. Does not alter the sequence.

Very useful for logging/tracing what's going on in the middle of a complex linq expression.

Note that Pipe() is lazy, where ForEach() is eager

Microsoft TechEd 2012

DIY

Building your own extensions

AsCount

```
public static string AsCount(
    this int count, string singular, string plural)
{
    return string.Format(
        "{0:0}{1}",
        count,
        count == 1 ? singular : plural);
}
```

An easy way to format your messages in a way that respects the English language.

However – don't use this if you're going to localise your application, as some languages don't do things the way english does. For example, some languages have a different term for "two" people vs one vs many.

So, best for in-house applications.

```
var message
    = string.Format(
        "Delete {0}?",
        items.Count.AsCount(" record", " records"));
```

Exception.AsEnumerable

```
public static IEnumerable<Exception>
    AsEnumerable(this Exception exception)
{
    var e = exception;
    do
    {
        yield return e;
        e = e.InnerException;
    } while (e != null);
}
```

Convert a single exception into a sequence by drilling into the nested inner exceptions.

Exception.AsStrings

```
public static IEnumerable<string> AsStrings(
    this Exception exception)
{
    yield return exception.Message;
    if (exception.Data != null) {
        foreach (DictionaryEntry e in exception.Data)
            yield return string.Format(
                "{0} : {1}", e.Key, e.Value);
    }
}
```

Extract all the information included in an exception.

Did you know that (almost) every exception includes “Data” – a dictionary of extra information about what happened.

Warning: The ExecutionEngineException, OutOfMemoryException, StackOverflowException and ThreadAbortException classes always return null for the value of the Data property.

http://msdn.microsoft.com/en-us/library/system.exception.data.aspx

```
try
{
    // ...
}
catch (Exception ex)
{
    var messages = from exception in ex.AsEnumerable()
                   from message in exception.AsStrings()
                   select message;
    foreach (var m in messages)
        mLogger.Error(m);
}
```

In reality, would put this code in the logger so we didn't repeat it

With & Return

```
var membershipDescription
    = person.Membership.MembershipType.Description;
```

Avoids lots of tests for nulls

```
var membershipDescription = string.Empty;
if (person != null
    && person.Membership != null
    && person.Membership.MembershipType != null)
{
    membershipDescription
        = person.Membership.MembershipType.Description;
}
```

Cumbersome and repetitive

With

```
public static TResult With<TInput, TResult>(
    this TInput o, Func<TInput, TResult> evaluator)
    where TResult : class
    where TInput : class
{
    if (o == null)
    {
        return null;
    }

    return evaluator(o);
}
```

Safe way to drill into an object while handling nulls by returning null

Return

```
public static TResult Return<TInput, TResult>(
    this TInput o,
    Func<TInput, TResult> evaluator,
    TResult defaultValue)
    where TInput : class
{
    if (o == null)
    {
        return defaultValue;
    }

    return evaluator(o);
}
```

Drill into an object, with a default value if it's null

```
var membershipDescription
    = person.With( p => p.Membership )
        .With( m => m.MembershipType )
        .Return( t => t.Description, string.Empty );
```

Avoids lots of tests for nulls

Microsoft TechEd 2012

Summary

Framework
Third party
Do it yourself

Microsoft TechEd 2012

Goals

- - - - →

One to use
One to try
One to write

What should you take away from this presentation?

One improvement that you know you can use

One improvement that you want to try

One improvement that you want to write yourself.

Related Content

- Breakout Sessions
  - DEV201: What's new in Visual Studio 2012
  - DEV311: Turbocharge your Productivity with Visual Studio
  - DEV314: What's New for Unit Testing in Visual Studio 2012
- Contact Me Later By...
  - Email: bevan@nichesoftware.co.nz
  - Twitter: @unrepentantgeek
  - Blog: http://www.nichesoftware.co.nz

Microsoft TechEd 2012

Track Resources

- Resource 1
- Resource 2
- Resource 3
- Resource 4

Required Slide

*delete this box when your slide is finalised

Track Owners will supply the content for this slide, which will be inserted during the final scrub.

Microsoft TechEd 2012

Resources

Microsoft TechEd New Zealand 2012

Connect. Share. Discuss.
newzealand.msteched.com

Microsoft | Learning

Microsoft Certification & Training Resources
microsoft.com/learning

Microsoft | TechNet

Resources for IT Professionals
microsoft.co.nz/technet

msdn

Resources for Developers
microsoft.co.nz/msdn


DEVELOP WOW
Visual Studio
Prepare to code greatness.
See how as we launch the new
Microsoft Visual Studio 2012.
Code Camp and Visual Studio 2012 Launch – this weekend – 9 September at Auckland University Business School - FREE
mscommunities.co.nz/codecamp.aspx
Save the date – 13 September NZ time – and REGISTER NOW to attend the free virtual event
visualstudiolaunch.com
Visual Studio


Visual Studio
MICROSOFT STORE
OPEN FOR BUSINESS 24 X 7
MICROSOFTSTORE.CO.NZ

Microsoft®
TechEd
New Zealand 2012

Win instant prizes by evaluating this session:

aka.ms/mobile

Microsoft®