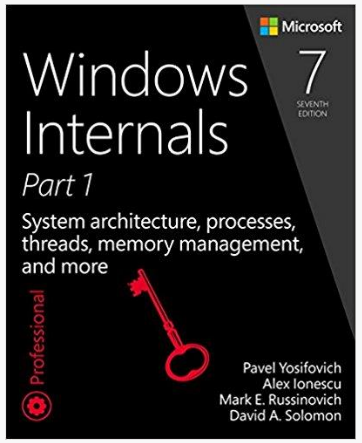
What's New in C# 7.x and C# 8.0?

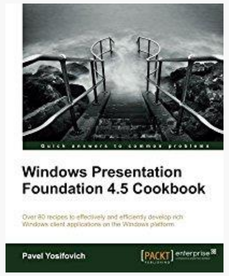
Pavel Yosifovich
http://blogs.microsoft.co.il/pavely
@zodiacon

Learn. Imagine. Build.
.NET Conf

About Me

- Developer, Author, Trainer and Speaker
- Tech Lead and Architect at Cybereason
- Co-author of *Windows Internals 7th edition, Part 1* (2017)
  - Author of ***WPF Cookbook*** (2012), ***Mastering Windows 8 C++ App Development*** (2013)
- Pluralsight author
  - "Windows Internals" series and ".NET Interoperability Fundamentals"
- Microsoft MVP
- Author of several open-source tools (http://github.com/zodiacon)
- Blog: http://blogs.microsoft.co.il/pavely
- Twitter: @zodiacon


Microsoft
Windows Internals 7 SEVENTH EDITION
Part 1
System architecture, processes, threads, memory management, and more
Professional
Pavel Yosifovich
Alex Ionescu
Mark E. Russinovich
David A. Solomon


Windows Presentation Foundation 4.5 Cookbook
Pavel Yosifovich
[PACKT] enterprise

Mastering Windows 8 C++ App Development
A practical guide to developing Windows Store apps with C++ and XAML
Pavel Yosifovich
PACKT

Agenda

- C# Road(map)
- C# 7.0
- C# 7.1
- C# 7.2 & 7.3
- C# 8.0
- Q&A

.NET Conf

C# Road(map)

C# 1.0 →	Managed Code
C# 2.0 →	Generics
C# 3.0 →	LINQ
C# 4.0 →	Dynamic Typing
C# 5.0 →	Async Programming

.NET Compiler Platform ("Roslyn")

C# 6.0
Many (small) features

C# 7.0
Pattern Matching

C# 7.x
Minor Features

C# 8.0
Major Features

Recap: C# 6.0

- Available with Visual Studio 2015

Null conditional operator	Auto property initializers
Read only auto properties	Named expressions
Using static	Expression bodied members
String interpolation	Exception filters
Await in catch and finally blocks	Index initializers
Extension methods for collection initializers	Improved overload resolution

C# 7.0

- Available with Visual Studio 2017

Out variables	Tuples
Pattern matching	Ref locals and returns
Local functions	More expression bodied members
Throw expressions	Generalized async return types
Numeric literals syntax improvements	

Out Variables

- Prior to C# 7.0
  - Out variables must be declared before passed to methods
- C# 7.0
  - Out variables can be declared “inline” within the method called

Demo: Out Variables

Tuples

- Prior to C# 7.0
  - Tuples exist as `System.Tuple<>` set of types
  - Members are called `Item1`, `Item2`, …
  - Reference types
  - No special language support
- C# 7.0
  - A new tuple type: `System.ValueTuple`
  - Value type
  - Available with a Nuget package
  - Language support

Demo: Tuples

Ref locals and returns

- Prior to C# 7.0
  - Returning a reference to a value is not allowed
  - Unless unsafe code is used with pointers
- C# 7.0
  - Methods can return references to values
  - Caller chooses ref or value call
  - Compiler protects against illegal usage
    - Returning a reference to a value that with a method limited lifetime

Demo: Ref locals and returns

Pattern Matching

- Pattern matching expressions
- Enhanced `is` operator
- Enhanced `switch` statements
  - `switch` can except non-integral types

Demo: Pattern Matching

Local functions

- Functions defined within existing function
- Private to the enclosing function
- Have access to local variables in the enclosing scope
- What about delegates?

Demo: Local functions

.NET Conf

More Expression Bodied Members

- C# 6.0 allows read only properties and methods bodies to be written as lambda expressions
- C# 7.0 extends this
  - Full properties, indexers, constructors, finalizers

Demo: More expression bodied members

Throw Expressions

- Prior to C# 7.0
  - `throw` statements are… well… statements
- C# 7.0
  - `throw` calls are expressions
  - Can be placed in new locations

Demo: throw expressions

Generalized async return types

- Prior to C# 7.0
  - async methods must return `void`, `Task` or `Task<T>`
- C# 7.0
  - Async methods can return anything that adheres to the “awaiter pattern”
  - Simplest example: `ValueTask` type vs. `Task`

Numerical syntax improvements

- Binary numbers literals supported with ***`0b`*** prefix
- Underscore characters can be used as convenient separators
  - For any numeric literal

Demo: Numeric literals

C# 7.1

- Available with Visual Studio 2017 version 15.3
- Not enabled by default
- Features
  - **`Main`** method may be `async`
  - **`default`** literal expressions
  - Inferred tuple element names

.NET Conf

Demo: C# 7.1

.NET Conf

C# 7.2 and C# 7.3

- Expose CLR's "protected and internal" to C#
  - `private protected`
- Low-level interop
  - Currently, unsafe code (with pointers) is required in many cases
- Non-trailing named arguments
- Read-only references

C# 8.0

- Async streams
  - `IAsyncDisposable`
  - `IAsyncEnumerable<T>`, `IAsyncEnumerator<T>`
- Nullable reference types
- Default interface implementation
  - Static members allowed
  - No instance state allowed
- Records

Q&A

THANK YOU

.NET