BASTA!

.NET, VISUAL STUDIO & MORE

Share Connect 2011

SQLCON 2011

Oliver Sturm

Async

Oliver Sturm

thinktecture Associate

ineta

MVP Microsoft Most Valuable Professional

Oliver Sturm (@olivers)

Consultant and Trainer, Author
Associate Consultant at thinktecture

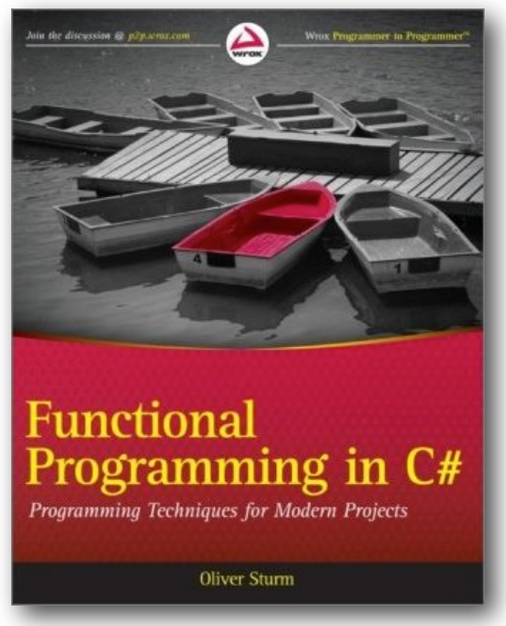
.NET Application System Architecture

- User Interfaces
- Data Handling / Data Access Architectures
- Programming Languages
- DevExpress Component/Framework Products

Microsoft MVP for C#
INETA Europe Speaker

Services: http://www.oliversturm.com
Blog: http://www.sturmnet.org/blog

oliver@oliversturm.com



Agenda

What is “Async” and why do I care?
Doing it yourself
Making it happen automatically
How does it work?

Things we've seen

"Asynchronous design pattern" -
.NET 1.1

- BeginInvoke/EndInvoke
- AsyncCallback
- IAsyncResult

Examples - many, all around networking, general IO, even delegates

Things we've seen

"Event-based asynchronous pattern" - .NET 4.0

- DoSomethingAsync
- DoSomethingCompletedEventHandler
- CancelAsync

Examples: WebClient, BackgroundWorker

Purposes of Async Patterns

Reducing latency - making things *appear* to run in parallel
Allow actual parallel execution when hardware is available, and depending on the asynchronous operation itself

Demo

Jumping right in:
Implementing Asynchrony manually

Thank <deity of choice> for lambdas and closures!

Meanwhile, elsewhere...

Parallel Extensions (PFX) introduced to .NET 4.0
Brilliant shiny new way of organizing parallelization
Task and Task<T> are available to represent outstanding pieces of work, with and without results
WinRT uses IAsyncResult<T> and Task<T>, JavaScript "Promises"

TAP (<- meaning no. 128 according to acronymfinder.com)

"Task-based asynchronous pattern" -
Async CTPs and now VS 11 DP

- await
- async
- Hidden stuff:
  - GetAwaiter
  - IsCompleted, OnCompleted, GetResult
  - *Forget BeginAwait, EndAwait!*
- Continuations for the Masses!

Demo

Async/Await

WOW!

Right?

Seriously:

No structural changes!
No delegates!
Absolutely freakin' nothing other than a few keywords in the right places!

Demo

... and a few more features

How does it work?

- The `await` keyword makes the compiler generate code to call into the GetAwaiter/BeginAwait pattern, passing a continuation
- The “default” implementation of that pattern in the CTP is the TaskAwaiter
- `async` is just for fun :-)

Demo

... and a few more features

Terminology

`await` does NOT mean

“wait at this point”

`async` does NOT mean

“run in the background”

At this time it seems like MS expect people to figure it out. VS 11 DP has the same keywords.

Final words

Why did it have to be new keywords in the compiler... ah well.

Summary

Async/await - perhaps not perfect, but very cool for certain purposes

Check it out in VS 11 DP!

Thank you!

Please feel free to contact me about the content anytime!

oliver@oliversturm.com