# Asynchronous Computing and Composing Asynchronous and Event-Based

# Asynchronous Computing and Composing Asynchronous and Event-Based

RXTDF - Version: 1

## ⏱ 5 days Course

## Description:

5 days that target the different approach to parallelism and
computation of asynchronous events using
Async / Await, Reactive Extension (RX) and TPL Dataflow.
Rx is a functional programming library designed to handle complex event processing. The
course deep-dives into the library's concept and guidelines.
It will cover topics like exception handling, testing, remote processing, and scheduling.
Students will master both practice and theory and become familiar with numerous RX
operators.
TPL Dataflow is an agent-based library designed to achieve high throughput and low latency
for both I/O- and CPU-bound operations. The course deep-dives into the library's concept
and guidelines. Students will master both practice and theory.

## Intended audience:

.NET developers or team leaders with:
•at least 2 years of experience with C#;
•Programming experience and some practice with LINQ Query;
•Programming experience and some practice with multithreading;
•Familiarity with TPL (.NET 4) is recommended.

**Prerequisites:**

**Objectives:**

Appreciate the architectural and design principles of Async, RX and TPL Dataflow programming model

Practice complex event and messaging processing

Learn to handle exceptions

Test complex event pipeline

Know how to design RX flow and TPL Dataflow

Learn to combine different approach together

Master design principles and guidelines of the agent base model

Practice the complex parallel flow

Understand TPL Dataflow blocks and performance tuning

**Topics:**

## Introduction

- Moore's Law
- Amdahl's Law
- Thread safety
- .NET Parallel History
- Task and Task<T>
- Custom Task
- Continuation
    - º Tasks vs. APM
- Async and Await
- Cancellation
- Exception Handling

- Concurrent Collection

## Rx Introduction

- What is Rx?
- Push standard
- LINQ-able
- Like Events but better
- Course Goals
- Why Rx

  º Push vs. pull

  º Test Case: Cloud search

## Get started

- NuGet

## Concept

- Producer / Consumer

## Library structure

• Different offering
• Enlighten concept

## Marble Diagram

• Concept
• Select
• Where

## Built-in factories

• Interval
• Timer
• Range
• Return
• Create
• Generate

## Monitoring

• Do operator
• Visual RX

## Concurrency model

- Scheduler
- Built-in Scheduler
- ObserveOn
- SubscribeOn

## Backwards compatibility

- Composite events

## Exception Handling

- Retry
- Catch
- Finally
- SubscribeSafe

## Producer nature

- Hot Vs. Cold Observables
- Publish
- RefCount

## Subjects

- Subject
- Replay and ReplaySubject

## Operators

- District
- DistinctUntilChanged
- Sample
- Aggregate
- Scan

## Common Combinators

- Merge
- Zip
- CombineLatest
- Amb

## Splitting

- Buffer
- Window
- Select Many
- Group By

## Time-oriented

- Interval
- Timeout
- Timestamp
- TimeInterval
- Throttle
- Sample
- Take and Skip
- Generate Delay
- Delay Subscription

## Time-based Combinators

- Join
- Group Join

## Disposables

- Create
- CompositeDisposable
- RefCountDisposable
- CancellationDisposable
- BooleanDisposable
- ContextDisposable
- ScheduledDisposable
- SingleAssignmentDisposable

- MultipleAssignmentDisposable
- SerialDisposable

º LAB 04

## Scheduling

- Custom Scheduler
- Virtual time
- Historical Scheduler

## Testing

- Notification
- TestScheduler
- CreateHotObservable / CreateColdObservable

## Custom Operations

- Defer
- Extending the framework

# Remote

- Rx-Remoting
- IQbservable

# Introduction to Async / Await (C#5)

- Using
- Loops

# TPL Advance

- Scheduling
- Async and UI

# Parallel I/O

- What makes I/O operations different?
- I/O completion port

# What is TPL Dataflow?

- The agent base concept
- Evolution
- Goals

# Getting started

- Namespace
- NuGet

## Contract

- Source API
- Target API
- Block API
- Push vs. Pool

## Blocks

- Blocks categories

## Action block

- Structure
- Functionality
- Throttling

## Buffer Block

- Structure
- Functionality
- Push and Pool
- Bounded capacity

## Broadcast Block

- Structure
- functionality
- What makes it different than Buffer Block?

## Transform Block

- Structure
- Functionality

## Transform Many Block

- Structure
- Functionality

## TPL Dataflow and Async

- Using async / await with TDF
- Processing I/O operations

º LAB 06

## Performance tuning

- MaxMessagesPerTask

# Batch Block

- Structure
- Functionality

# Join Block

- Structure
- Functionality

# Greediness

- Targeting complex scenarios and correlation
- ConsumeMessage API
- Two-phase commit
- ReserveMessage API
- ReleaseReservation API

# Greediness and built-in blocks

- BatchBlock
- JoinBlock

# BatchedJoinBlock

- Structure
- Functionality

# WriteOnceBlock

- Structure
- Functionality

º LAB 08

# Rx vs. TDF

- Design differences
- Better together

º Summary