

Sela.

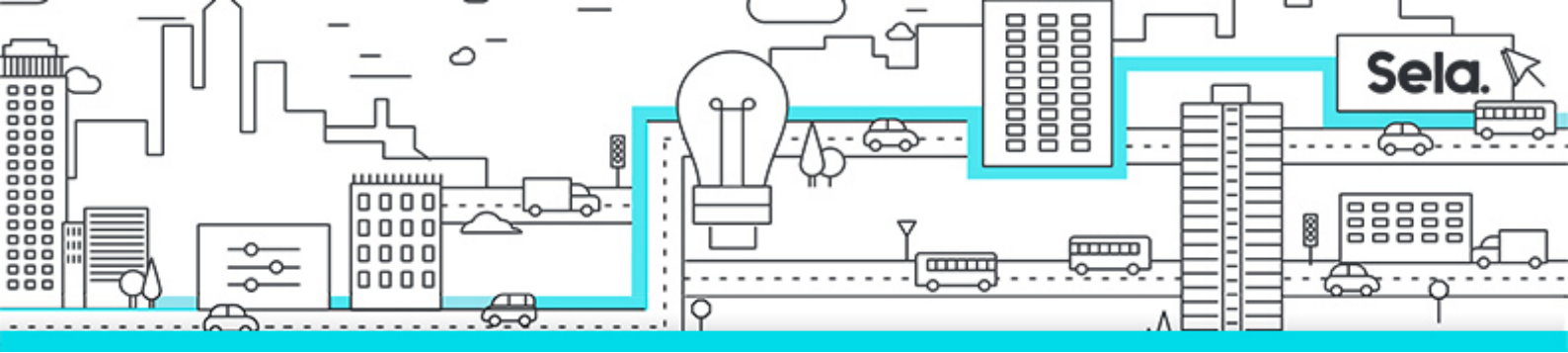
50153

.NET Performance

college@sela.co.il

03-6176666





.NET Performance

50153 - Version: 3

3 days course

Description:

This three-day instructor-led course provides students with the knowledge and skills to develop high-performance applications with the .NET Framework. Building high-performance applications with the .NET Framework requires deep understanding of .NET memory management (GC), type internals, collection implementation and most importantly - tools for measuring application performance. The course features numerous performance measurement scenarios, optimization tricks, deep focus on .NET internals and high-performance development guidelines.

Intended Audience:

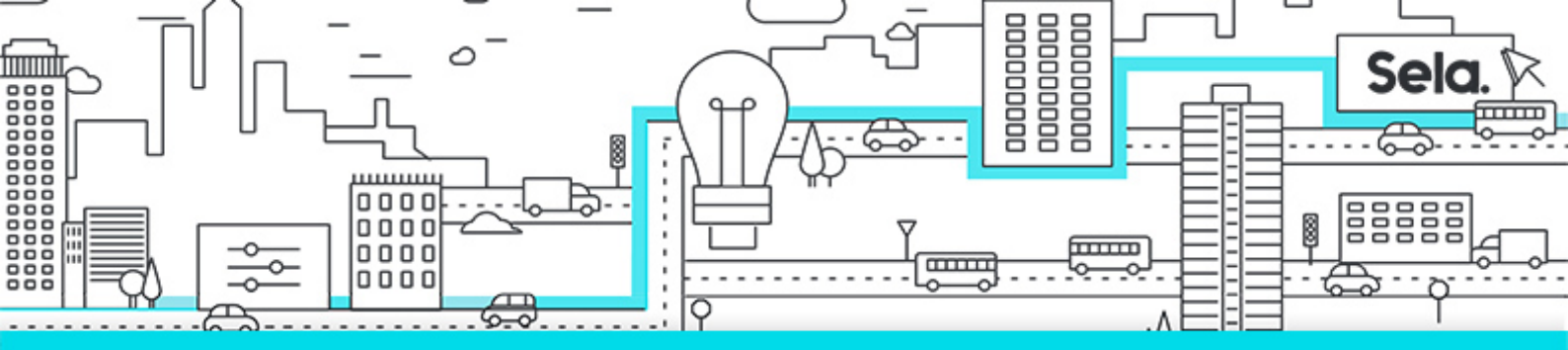
This course is intended for C# developers with practical experience of at least a year with the .NET framework.

Prerequisites:

- Working knowledge of C# 3.0
- Working knowledge of the .NET Framework
- Familiarity with operating systems topics: Threads, paging, file-system cache, memory management
- Familiarity with computer organization topics: CPU, cache, memory

Objectives:

- Measure the performance of .NET applications on the Windows platform.
- Avoid performance pitfalls in all kinds of managed applications.
- Improve application memory management performance by properly



interacting with the .NET garbage collector.

- Choose the right collection implementation for managed applications.
- Choose between reference types and value types, virtual and non-virtual methods.

Topics:

- **Module 1 - Introduction**

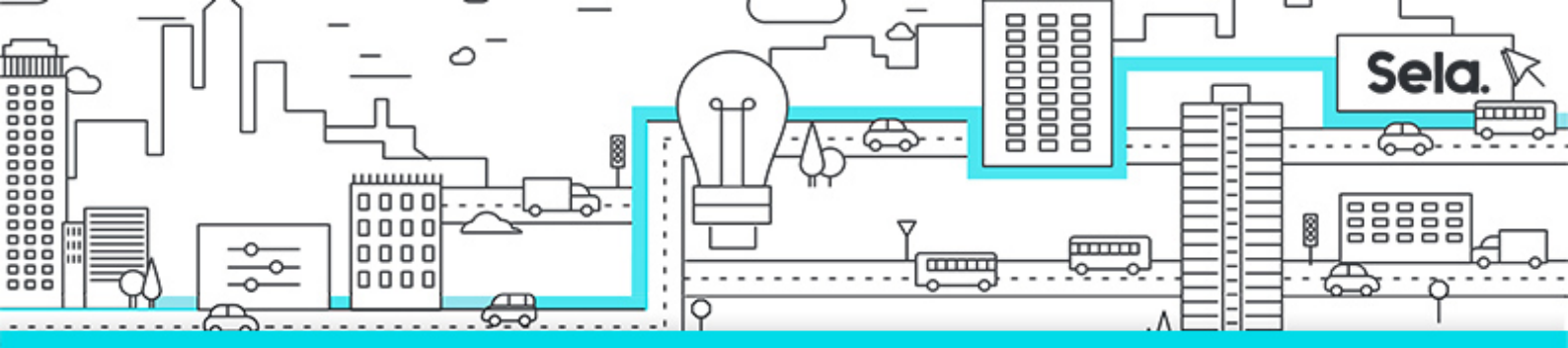
- **Module 2 - Performance Measurement**

- Performance measurement metrics - what can be measured?
- Windows performance counters
- CPU profilers - sampling and instrumentation
- Memory allocation profiling
- Memory leak profiling
- Concurrency profiling
- Event Tracing for Windows
- Windows Performance Toolkit and PerfView
- Micro-benchmarking
- LAB: Measuring CPU time and wall-clock time
- LAB: Profiling memory allocations
- LAB: Diagnosing a memory leak
- LAB: Profiling CPU cache misses

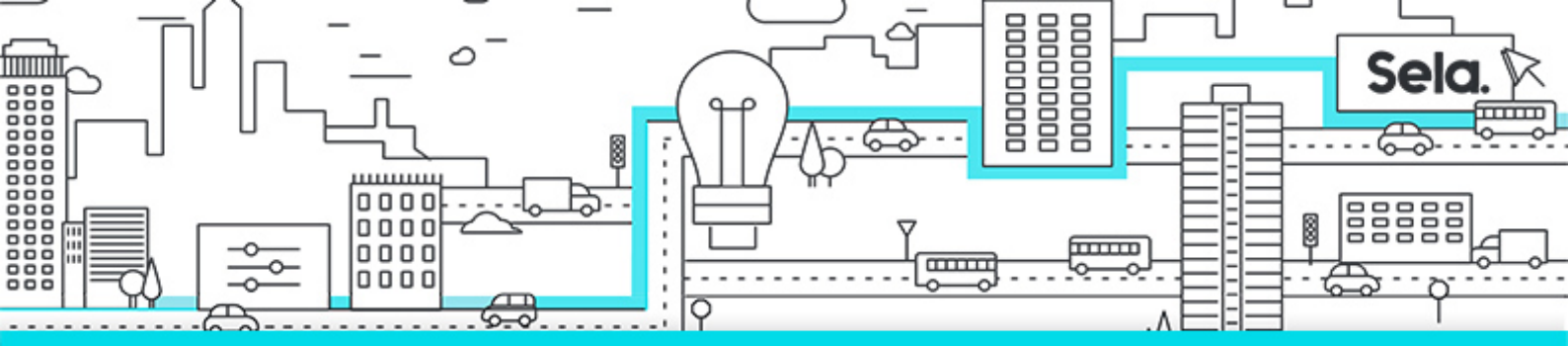
- **Module 3 - Type Internals**

- Differences between value types and reference types
- Reference type memory layout - type object pointer, sync block index
- Invoking virtual vs. non-virtual methods, the sealed modifier
- Value type memory layout, boxing
- Implementing value types correctly - Equals and GetHashCode

- **Module 4 - Garbage Collection**



- Reference counting vs. tracing GC
 - The managed heap and the next object pointer (NOP)
 - Mark and sweep GC model, GC roots
 - GC flavors - workstation GC, server GC
 - Thread suspension for GC
 - Pinning objects referenced by unmanaged code
 - Generations and inter-generation references
 - GC segments and virtual memory
 - Managed GC APIs
 - Finalization internals and deterministic finalization
 - Weak references
 - Best practices for interacting with the GC
- **Module 5 - Generics**
 - Motivation and generic constraints
 - Implementation of generics at runtime
 - .NET generics compared to Java generics and C++ templates
 - **Module 6 - Unsafe Code**
 - The Marshal class, accessing unmanaged memory
 - Copying data from unmanaged structures
 - C# pointers, the unsafe keyword, pinned pointers
 - LAB: Implementing memory copy with unsafe code
 - LAB: Improving upon code-generation approaches
 - **Module 7 - Collections**
 - .NET Collections
 - Choosing a Collection
 - Cache Considerations
 - Custom Collections



• **Module 8 - JIT Optimizations**

- Multi-Core Background JIT
- NGen
- MPGO
- RyuJIT
- ILMerge
- .NET Native
- Method Inlining
- Range Check Elimination
- Microsoft.Bcl.Simd