.NET Performance and Debugging Workshop

DNWSH - Version: 2.3

6 days Course

Description:

The .NET Performance and Debugging Workshop is a practical workshop for experienced .NET developers willing to develop high-performance .NET applications and debug them in the development and production environments. In this eight-day workshop, you will obtain practical knowledge about the performance characteristics of the .NET framework and common language constructs; learn about the relevant internal details of the .NET type system, garbage collector and synchronization mechanisms; and practice debugging scenarios that arise in the development or production environment of .NET applications. This workshop is limited to 8-12 participants to ensure that instructor attention is properly distributed during the practical labs, and to provide an opportunity for specific questions to be brought up after classroom hours.

Intended audience:

This workshop is intended for experienced .NET developers with working knowledge of C#.

Prerequisites:

Working knowledge of C# 3.0
Working knowledge of the .NET Framework, including threading, synchronization mechanisms, application domains
Familiarity with the C++ programming language (preferred but not a must)
Familiarity with operating systems concepts: paging, virtual memory, processes and threads
Familiarity with computer organization concepts: CPU cache, registers, main memory

Objectives:

Develop high-performance .NET applications
Expose custom performance and monitoring data from .NET applications
Analyze the performance of existing applications and tune them appropriately
Diagnose memory leaks, deadlocks, crashes and other scenarios
Use a variety of external tools to monitor your application's behavior in production environments

Topics:

.NET Performance

- 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

- º The managed heap and the next object pointer (NOP)
- º Mark and sweep GC model, GC roots
- º GC flavors - workstation GC, server GC
- º Thread suspension for GC
- º Pinning objects referenced by unmanaged code
- º Generations and inter-generation references
- º GC segments and virtual memory
- º Managed GC APIs
- º Finalization internals and deterministic finalization
- º Weak references
- º Best practices for interacting with the GC

• Module 5 - Generics
- º Motivation and generic constraints
- º Implementation of generics at runtime
- º .NET generics compared to Java generics and C++ templates

• Module 6 - Unsafe Code
- º The Marshal class, accessing unmanaged memory
- º Copying data from unmanaged structures
- º C# pointers, the unsafe keyword, pinned pointers
- º LAB: Implementing memory copy with unsafe code
- º LAB: Improving upon code-generation approaches

• Module 7 - Collections
- º .NET Collections
- º Choosing a Collection
- º Cache Considerations
- º Custom Collections

• Module 8 - JIT Optimizations
- º Multi-Core Background JIT
- º NGen
- º MPGO
- º RyuJIT
- º ILMerge
- º .NET Native
- º Method Inlining
- º Range Check Elimination
- º Microsoft.Bcl.Simd

.NET Debugging

- Module 01 - Exceptions and Dumps
  - Exception Handling
  - Debugging Symbols
  - Dump Files and Types
  - Generating Dumps
  - Automatic Dump Generation
  - Opening Dump Files
- Module 02 - Introduction to WinDbg
  - Basic WinDbg Commands
  - Smart Breakpoints
  - WinDbg Scripts
  - WinDbg Extensions
  - LAB: Getting Acquainted with WinDbg
  - LAB: Capturing Crash Dumps (x3)
- Module 03 - Debugging Tools
  - Performance Counters
  - Process Explorer
  - Process Monitor
  - Application Compatibility Toolkit
  - ETW and Xperf
  - GFlags
  - LAB: Profiling with Xperf
  - LAB: Process Monitor
  - LAB: Application Compatibility Toolkit
- Module 04 - Debugging in Visual Studio
  - Visual Studio Windows
  - Breakpoints and Tracepoints
  - Data Breakpoints, Function Breakpoints
  - Threads
  - Parallel Stacks, Parallel Tasks
  - Static Code Analysis
  - LAB: Runtime Checks
- Module 05 - SOS
  - Setting Smart Breakpoints

- Analyzing Memory Leaks
- Inspecting Objects
- Inspecting Threads and Stacks
- Advanced Commands
- LAB: Getting Acquainted with SOS
- LAB: Capturing Crash Dumps (x2)
- LAB: Deadlock (x2)
- LAB: Memory Leak (x4)

• Module 06 - .NET Debugging Tools
- Managed Debugging Assistants
- IntelliTrace
- Visual Studio Profiler
- CLR Profiler
- ANTS Memory Profiler
- Assembly Loading Diagnostics
- LAB: Fusion Diagnostics
- LAB: IntelliTrace

• Appendix 01 - Assembly Language Fundamentals
• Appendix 02 - Interop Debugging