Memory Management in .NET

Understanding How .NET handles memory

Mirolim Majidov

What is RAM in a computer? What does it do?

Value and Reference types

Value types:

Examples: **int, float, struct, enum, Span<T> …**
Stored on the **Stack**

Reference types:

Examples: **class, interface, string, array, delegate, …**
Stored on the **Heap**

More info

Stack and Heap

Stack:

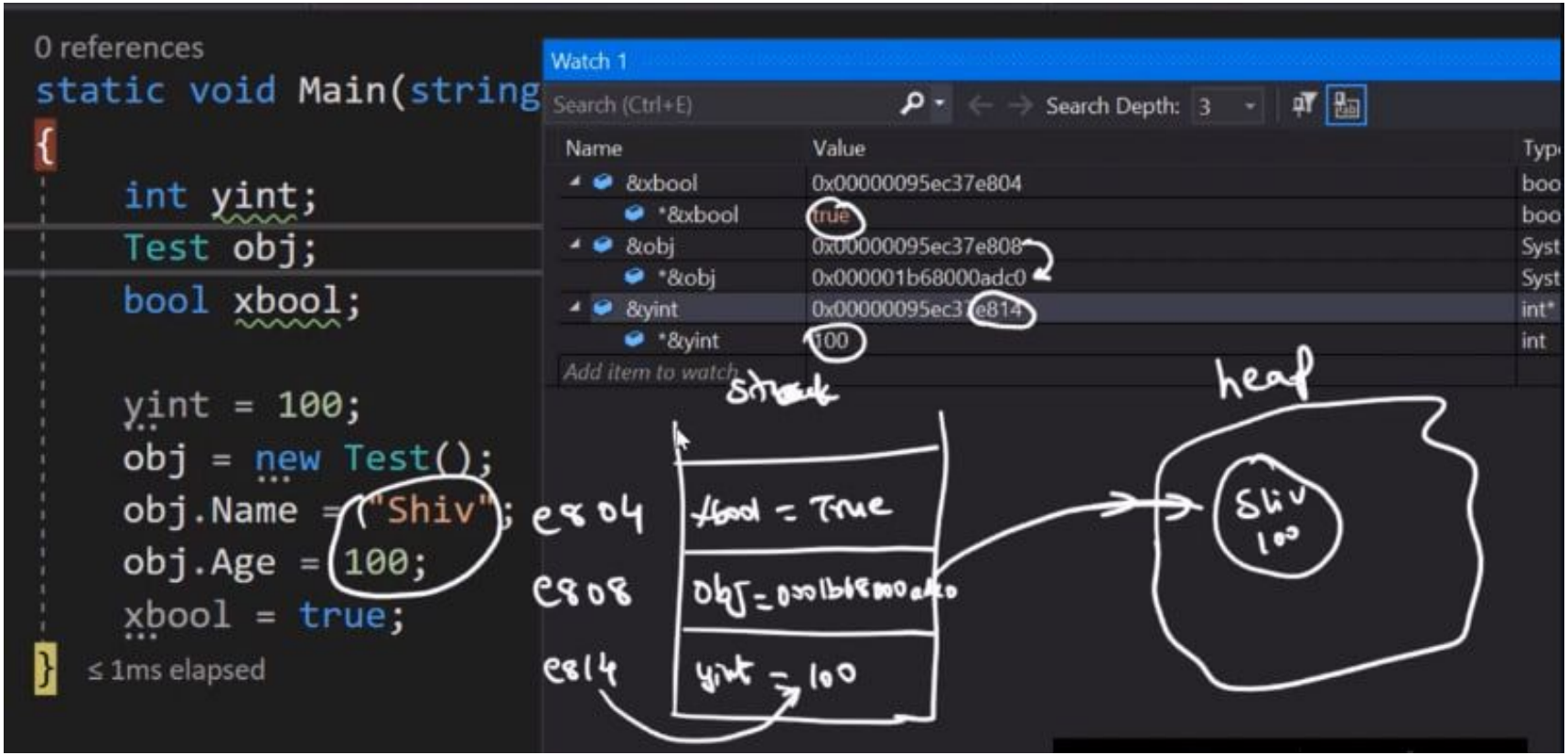
Managing the execution of program code, including method calls and returns.
Fast access, limited in size (1MB for 32-bit and 4MB for 64-bit).
One Stack will be created **per thread**.
The data will be removed with **LIFO** (Last in first out)

Heap:

Dynamic memory allocation.
Slower access, larger in size.
One a heap for the **whole application**.
The **Garbage collector** clean all reference data that does not have reference.

More info

Stack and Heap



Mutable and Immutable classes

Mutable classes:

Data **can be changed** after creation.
In C#, all reference types are designed to be mutable by default.
Examples: Arrays, StringBuilder, Collections (List, Dictionary<,>)

Immutable classes:

Data **cannot be changed** after creation.
In C#, all **value types** are designed to be immutable by default.
Examples: Integer, Boolean, String, DateTime

Example: Concatenation with String and StringBuilder, First example Second example

Boxing and Unboxing

Boxing:

Converting a value type to a reference type.
Can lead to performance overhead.

Unboxing:

Converting a reference type back to a value type.
Can cause runtime exceptions if not done correctly.

More info

Garbage collector and Garbage collection

Garbage Collector:

Manages the allocation and deallocation of memory in .NET.

Garbage Collection:

Process of automatically identifying and reclaiming unused memory.

Small and Large Object Heap

Small Object Heap (SOH):

This segment for storing small objects in 3 generations (Gen 0, Gen 1, Gen 2).

Large Object Heap (LOH):

This segment for storing large objects, usually more than 85,000 bytes(85 KB) in size.

More info

Managed and Unmanaged resources

Managed resources:

Resources that are automatically handled by the .NET runtime through the garbage collector.

Unmanaged resources:

Resources that are not handled by the garbage collector and need to be manually managed and released (e.g., file streams, database connections).

Dispose Pattern

The Dispose Pattern is a pattern used to cleanup of unmanaged resources, such as file handles, database connections, or native memory. It involves implementing the IDisposable interface to release resources explicitly when they are no longer needed.

More info

Memory Investigation

Techniques and tools to analyze and investigate memory-related issues, such as memory profilers (**VS Tools**, **dotTrace**, **dotMemory**), memory dumps, and performance monitoring tools.