Robert Haken
software & cloud architect, HAVIT, s.r.o.
haken@havit.cz, @RobertHaken, https://knowledge-base.havit.cz + .eu
Microsoft MVP: Development, MCT, MCPD: Web, MCSE: Cloud

.NET [Core] Internals
Garbage Collection

AGENDA

Architektura paměti .NET aplikací

Managed Heap a Garbage Collection

Roots

Generations

Large Object Heap

Finalization, IDisposable

Další optimalizace GC

Heap Inspection (02-ManagedHeap)

DEMO

Architektura paměti .NET aplikací

Instrukční paměť

Zásobník (Stack)

Halda (Heap)

- Managed Heap (.NET, Garbage Collector)
- Native Heap (Unmanaged Resources)

Halda (Managed Heap)

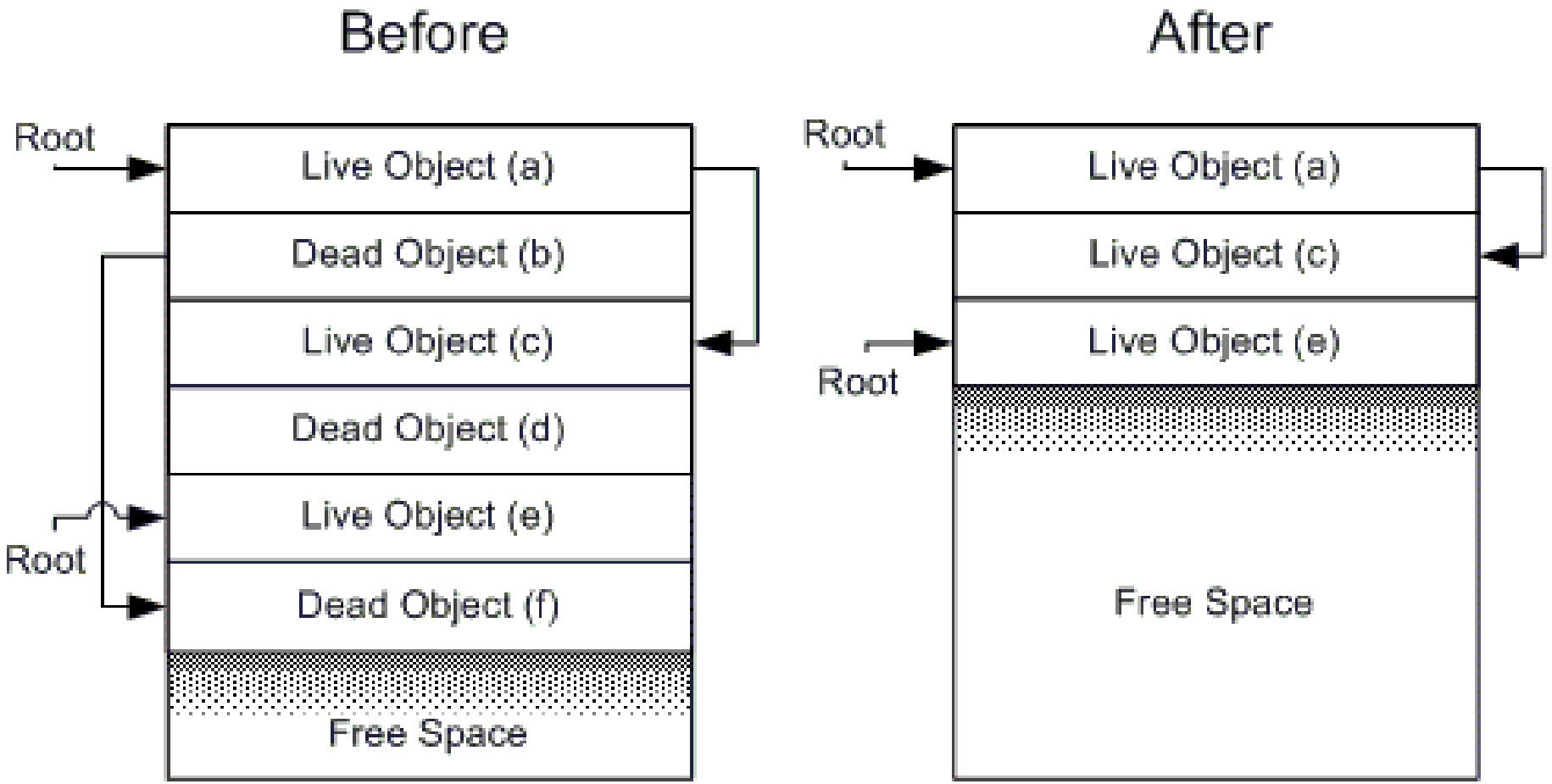
Garbage Collector ~ 1959 LISP

Garbage Collection

- překročení thresholdu generace při alokaci
- system-wide memory pressure
- GC.Collect() API (never-ever!)

bump pointer allocation vs. free-list

sesypání, update referencí


Before
After
Root
Live Object (a)
Dead Object (b)
Live Object (c)
Dead Object (d)
Live Object (e)
Dead Object (f)
Free Space
Root
Root
Live Object (a)
Live Object (c)
Live Object (e)
Free Space
Root

Garbage Collection (03-SimpleGarbageCollection)

DEMO

Roots

kořeny grafu objektů pro zjištění dosažitelnosti

- zásobník (lokální proměnné, parametry metod)
- GCHandles
  - globální statické fieldy
  - pinned objects
  - ...
- F-reachable queue

(WeakReference)

GC Roots (04-GCRoots)

DEMO

Generations

výkonová optimalizace

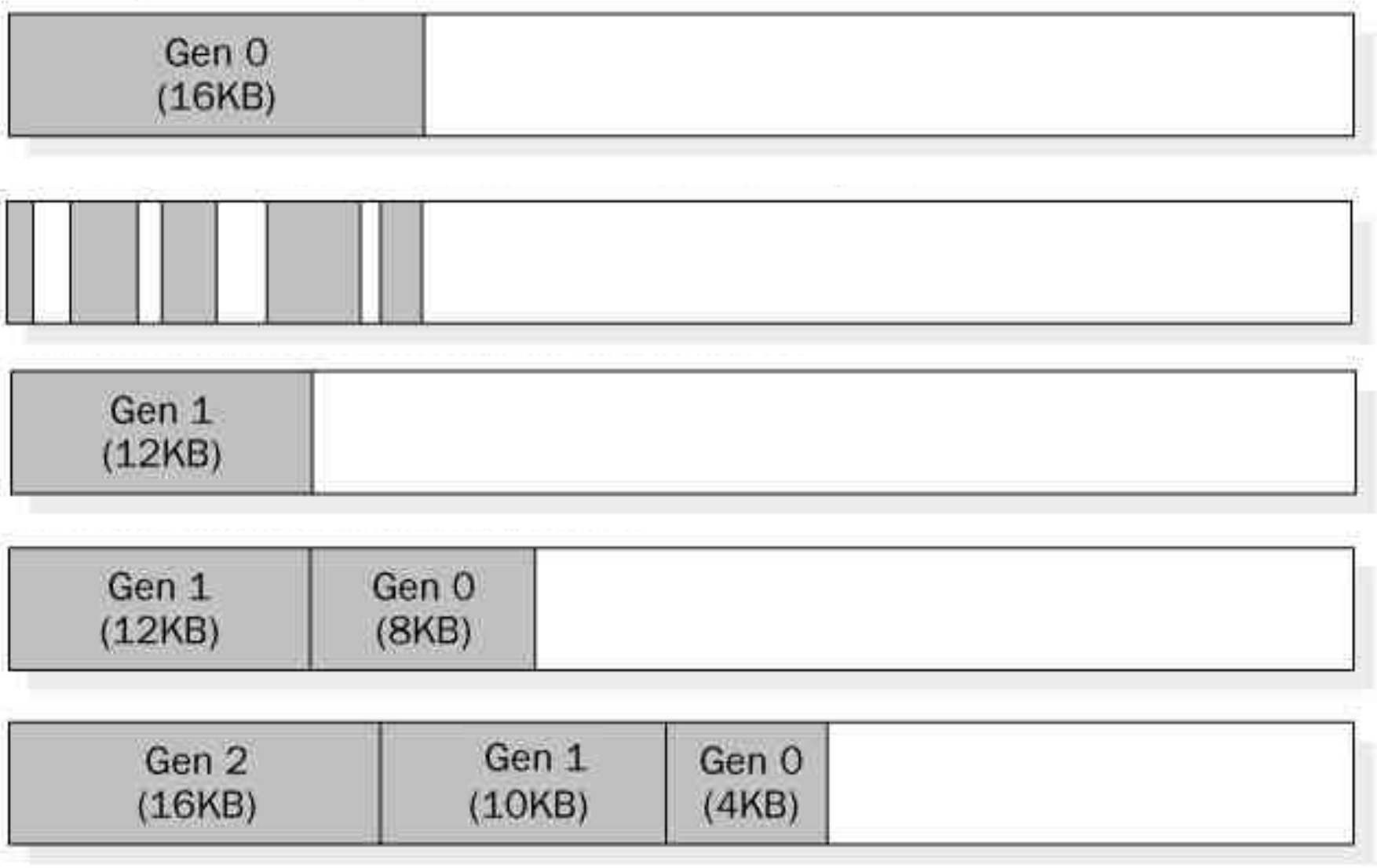
soustředí se objekty s krátkou životností

Gen 0 – nové, Gen 1 & 2 – přežily 1/více-krát

Gen 0 + 1 = ephemeral segment („fixed“ size)

Gen 2 = variable size

Generations



Generations (05-Generations)

DEMO

Finalization

explicitní úklid unmanaged zdrojů

Finalize() ~ C# destructor

Finalization Queue => do Gen 1

F-reachable Queue

Finalization Thread

IDisposable, ResourceWrapper pattern

DEMO

Large Object Heap

výkonová optimalizace

objekty větší než 85 000 bytů (default)

součást Gen 2 collection

nesetřásá se (NET 4.5.1+ lze jednorázově)

Free List

Large Object Heap (06-LargeObjectHeap)

DEMO

Garbage Collection Steps

1. Trigger
2. Suspend managed threads (vs. concurrent GC)
3. User-thread runs GC code
4. Select generation
5. Mark (GCRoots, Remembered Sets vs. Card Tables)
6. Plan (Compact/Sweep)
7. Sweep / Compact + Relocate
8. Resume managed threads

Verze Garbage Collectoru

Workstation (lag) vs. server (throughput)

configuration/runtime/gcServer enabled="true|false"

Background GC (NET4 wks only, NET4.5 svr)

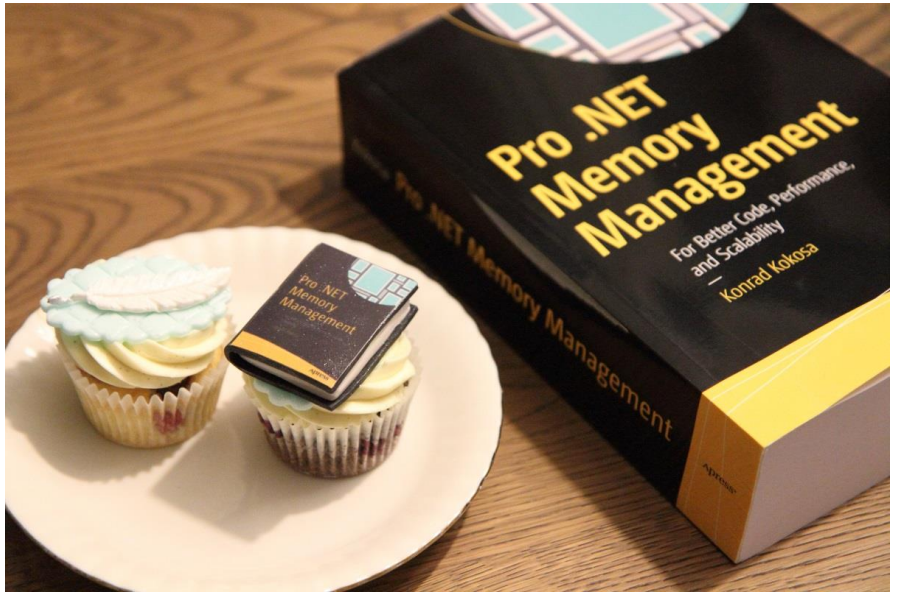
configuration/runtime/gcConcurrent enabled="true|false"

Objekt >2GB (x64, NET4.5)

LOH Free Lists Optimization (NET4.5)

LOH Heap Balancing (NET4.5)

LOH Compaction, explicit (NET4.5.1)



Q & A

Robert Haken

MVP ASP.NET/IIS, MCT

@RobertHaken, haken@havit.cz

http://knowledge-base.havit.cz