Brug af variabler i C#

Af Michell Cronberg (michell@cronberg.dk)

Grundlæggende teori

Hukommelsen i en C# applikation er helt overordnet betragtet opdelt to dele - stack (den statiske hukommelse) og heap (den dynamiske hukommelse). Afhængig af koden kan en applikation godt består af flere hukommelsesområder, men for at opnå en overordnet forståelse så antag, at hukommelsen på et givet tidspunkt kan beskrives i et konkret stack/heap diagram:

Stack	Heap
[variabler og data]	[data]

På stack'en findes variabler og data relateret til eksekvering af metoder, og den er opdelt i frames svarende til et hukommelsesområde (virkefelt/scope) for hver metode. Frames er placeret i en standard stak-struktur (LIFO : last in - first out), og runtime kan på den måde dels adskille variabler i forskellige virkefelter fra hinanden, og dels holde styr på eksekveringen af programmet og stille de rigtige variabler til rådighed på de rigtige tidspunkter.

På stack'en placeres variabler der enten kan indeholde værdier eller reference til data placeret på heap'en. Af værdibaserede variabeltyper kan eksempelvis nævnes bool, byte, int, long, double, enum og struct. Værdierne fra disse placeres direkte på stack'en hvilket gør eksempelvis tilgang og kopiering meget hurtig.

På heap'en placeres alle de dynamiske data - typisk strenge, objekter, arrays og delegates. Til alle data placeret på heap'en findes en variabel der holder referencen. Den kan enten være placeret på stack'en eller i en datastruktur på heap'en (eksempelvis et array af objekter eller et objekt der har en reference til et andet objekt). Hvis data på heap'en ikke er refereret bliver det automatisk fjernet og hukommelsen frigivet til anden brug.

Her er et eksempel på et diagram som viser både værdibaserede typer (i (int), j (double), k (bool),l (char) og m (struct)) samt referencebaserede typer (n (string) og o (objekt)).

Stack	Heap
i = 10 j = 200.34 k = true l = 's' m = { x: 10, y: 20 } n = [ref] o = [ref]	"s" { Id: 1, Navn: "Mikkel" }