Generational garbage collection

La maggior parte del garbage ha vita breve: è vantaggioso focalizzare il garbage collection su porzioni di memoria più piccole e dinamiche (contenenti gli oggetti più giovani), per ridurre le pause di garbage collection.

Suddividiamo l'heap in più *generazioni*, alle quali possiamo applicare algoritmi diversi con frequenza diversa. Gli oggetti avanzano alla prossima generazione se quella in cui si trovano si riempie e sono accessibili da un oggetto più vecchio.

Esempi

Java ha tre generazioni:

young (gen 0) algoritmo di Cheney;

tenured (gen 1) mark-sweep con meccanismi per evitare frammentazione;

permanent (gen 2) no GC.

.NET simile a Java ma la gen 1 non compatta i blocchi.