Meetrapport container

Florian Humblot - 1720570
Vera Schoonderwoerd - 1721202

Datum: 21 februari 2019

Inhoudsopgave

Doel

Het doel van dit experiment is het meten van het verschil in geheugengebruik tussen de default implementatie en de student implementatie.

Hypothese

De hypothese is als volgt: de studentimplementatie zal evenveel ruimte in beslag nemen in het geheugen als de default implementatie. Dit is onze verwachting omdat de containers in principe alleen pixels moeten opslaan en tenzij er een rekenfout gemaakt is bij het aanmaken van de buffer zouden ze gelijk moeten zijn.

Met opmerkingen [VS1]: Onderbouwing

Werkwijze

Om het geheugen dat in beslag wordt genomen te meten wordt de visual studio memory profiler gebruikt. De profiler houdt alle heap allocations bij tijdens een sessie en genereert vervolgens een bestand waarin de hoeveelheid allocaties en de totale grootte in terug te vinden is.

Voor deze test worden alle plaatjes uit testset A 2x door het programma heen gehaald. De eerste keer met de default container, en vervolgens met de student container. De compiler optimalisatie flag staat op -O2 en de applicatie wordt in release mode gebouwd.

Resultaten

Met opmerkingen [VS2]: Wat betekent deze table? Leg de kolommen uit.

Identifier	Count	Size (Bytes)
- ImageIO::loadImage	14	1,914,570
+ RGBImagePrivate::set	7	957,285
+ RGBImageStudent::set	7	957,285
- HereBeDragons::NoWantOfConscienceHoldItThatICall	14	513,076
+ IntensityImagePrivate::set	7	256,538
+ IntensityImageStudent::set	7	256,538

De identifier kolom laat zien welke functie hoe vaak iets heeft gedaan. De private functies “ImageIO::loadImage” en “HereBeDragons::NoWantOfConscienceHoldItThatICall” roepen in eerste instantie de private versie (dus de default implementatie) en vervolgens de student versie aan. In de volgende rij kun je zien hoe vaak de specifieke implementatie is aangeroepen.

In de rechter kolom zie je de grootte van het geheugen dat in beslag is genomen door de aanroep van de functie waarbij de rijen 1 en 4 een som zijn van de implementatie footprints.

Verwerking

Voor de verwerking van de resultaten zijn geen berekeningen gemaakt. Door naar de data uit de kolom “Size (Bytes)” te kijken zien wij dat het geheugengebruik van de twee containers identiek is op implementatie niveau (rijen 2&3 met elkaar vergelijken en rijen 5&6 met elkaar vergelijken).

Met opmerkingen [VS3]: Wel is er verschil bij de size rij 1 en 4. Wat is hier mee?

Conclusie

De hypothese was correct. De memory footprint van de student en de default implementaties zijn volkomen gelijk. Het doel van dit onderzoek is dus gehaald en onze implementatie is dus correct in de zin dat het geen overbodig geheugen gebruikt t.o.v. de default implementatie.

Met opmerkingen [VS4]: Foutenanalyse? Doel dus gehaald?

De kans op meetfouten is vrijwel 0 doordat dit gegevens zijn die de visual studio analyzer uit de heap haalt en de gegevens gebaseerd zijn op een snapshot uit het wekgeheugen.

Evaluatie

De containers zijn even groot, dit betekent dat de studentimplementatie, net als de default implementatie, precies genoeg ruimte reserveert voor de allocatie van de pixeldata uit het plaatje.