Opdracht JSF31, week 5

We werken deze week verder met de opdracht van week 4. Om de verwerking van de berekeningen te versnellen, gebruiken we 3 threads en bekijken we de extra problemen die daarbij optreden.

Doelstellingen

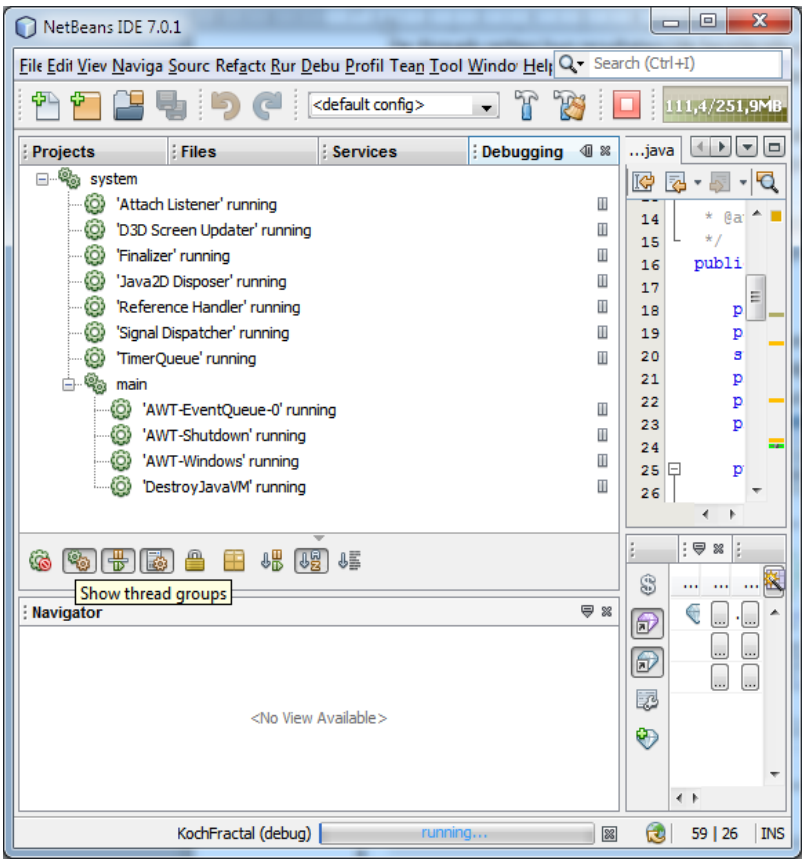
- meerdere threads laten samenwerken
- `synchronized` toepassen
- shared variables gebruiken

Opdracht

1. Pas de VM settings zo aan dat deze 4 CPU's gebruikt.

2. Maak een versie van de Koch Fractal applicatie die iedere keer als de fractal berekend moet worden (dus in methode `changeLevel`) drie threads opstart.
   Elke thread berekent 1 zijde van de initiële driehoek: `generateLeftEdge`, `generateBottomEdge`, en `generateRightEdge` worden nu dus ieder in een aparte thread uitgerekend. Hiervoor is een van `Runnable` afgeleide klasse nodig (of misschien geef je er de voorkeur aan om 3 klassen te maken). Gebruik de constructor van deze klasse(n) om waardes door te geven van de `KochManager` naar de thread.
   De threads moeten hun resultaten (de berekende edges) alle drie in dezelfde `ArrayList` (in de klasse `KochManager`) zetten die ook in Opdracht 4 werd gebruikt, en de Java FX Application Thread tekent de edges weer vanuit deze `ArrayList`. Let op: een `ArrayList` is niet threadsafe, dus als 2 threads daar tegelijk iets aan toe voegen kan het misgaan.
   Zorg er voor dat elke thread nu zijn eigen `KochFractal` object heeft. In dit geval treedt de van `Runnable` afgeleide klasse (en dus niet meer de `KochManager`) op als `Observer` van de `KochFractal` (die krijgt dus de `update` methode en moet met `addObserver` toegevoegd worden).
   Gebruik een gemeenschappelijke variabele `count` in de `KochManager` die initieel 0 is. Elke thread verhoogt deze variabele als hij klaar is, en wanneer alle threads klaar zijn, wordt de fractal getekend middels een aanroep van `application.requestDrawEdges()` vanuit de `KochManager`. Zorg dat de code threadsafe is door op de juiste punten `synchronized` te gebruiken.
   Let er op dat de Java FX Application Thread niet geblokkeerd wordt terwijl de threads aan het rekenen zijn.

3. Check met NetBeans of er inderdaad 3 extra threads draaien (start de applicatie in debugging mode, bekijk de debugging tab, en klik eventueel



onderin de button “Show thread groups”; zie screenshot op de volgende pagina); waarschijnlijk zie je de 3 threads pas bij de wat hogere levels, omdat bij de lagere levels de threads maar heel kort bestaan.

4. Voeg code toe om de totale rekentijd en de tekentijd te tonen. Vergelijk deze tijden met de single thread implementatie uit week 4.

<u>Toon het resultaat van vraag 4 aan de docent.</u>