# 20/09 – 2U

Ik heb vandaag nagedacht over wat mij erg zou interesseren om testen voor te schrijven. Ik ben dan tot de conclusie gekomen dat ik c# .net zou kiezen sinds we dit niet leren in onze opleiding en ik geïnteresseerd ben in c# .net. Ik heb een cursus gevonden bij openclassrooms waar de bron van terug te vinden is onderaan dit document bij Bronnen.

# 27/09 – 4U

Ik heb vandaag al het eerste en tweede hoofdstuk van de cursus doorlopen.

Hierbij bij zou ik bij aanvang vooral begrijpen waarom testing nuttig was in de .net omgeving en dit is ook duidelijk geworden.

Ik heb vandaag onderstaande spullen geleerd.

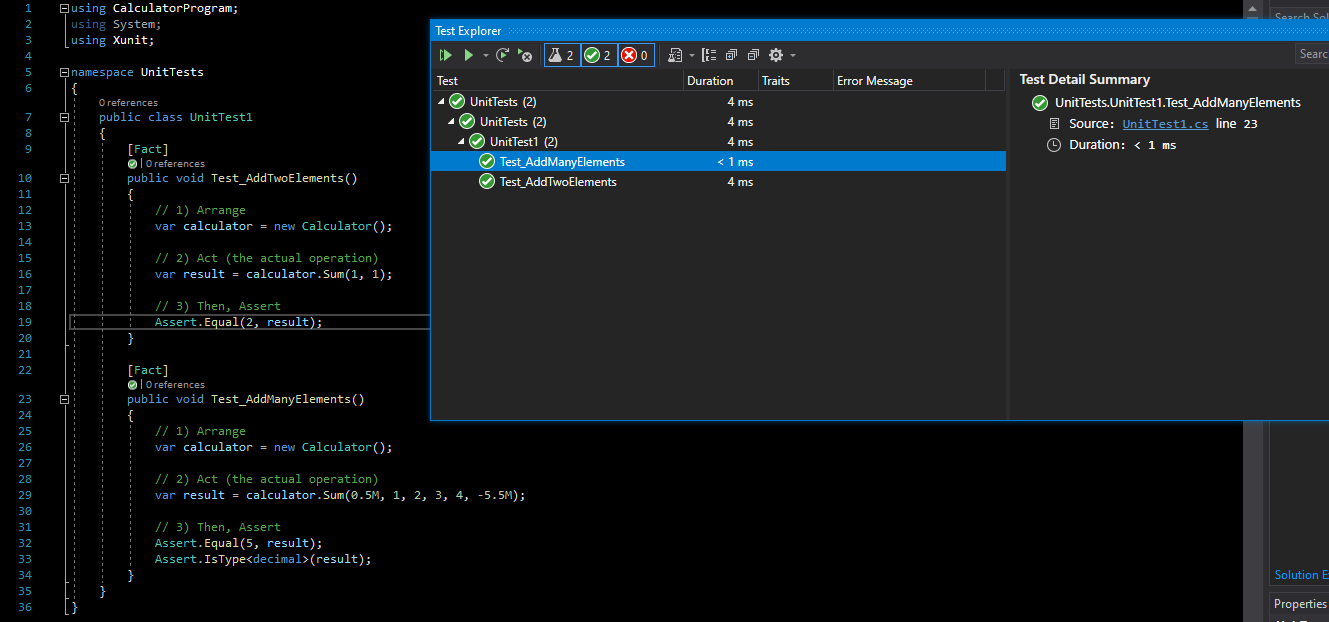
Er bestaan 3 soorten automatische testen namelijk: Unit, Mock en Integration tests.

Hierna werd het verschil tussen Test en behavior driven development uitgelegd. Bij TDD wordt eerst een test geschreven en daarna de code zodat deze zeker juist is bij BDD wordt er meer gekeken naar hoe je de werking kan tonen in normale tekst zodat iedereen dit kan begrijpen.

Vervolgens ging het over hoe je een goeie unit test kan schrijven Deze gaan volgens het FIRST principe dit betekent Fast, Isolated, Repeatable, Self-validating, Timely. Dit betekent dus vooral dat de unit testen snel moeten zijn maar vooral geïsoleerd zodat ze niet afhangen van andere code. En natuurlijk moet de testen blijven kunnen runnen.

Hierna was er het 2e hoofdstuk hier werden aan de hand van code uitgelegd hoe je testen moet schrijven. Ik heb deze dan gevolg en deze kunt u vinden met de naam CalculatorProgram.

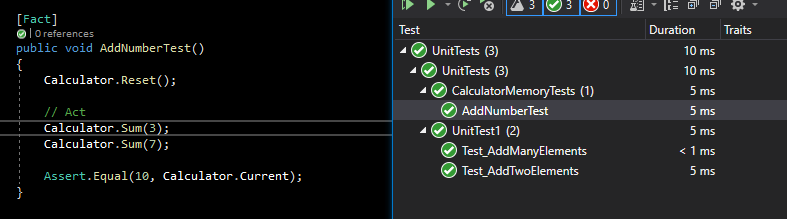
Hieronder kun je de tests zien en ook hun succesvolle test.



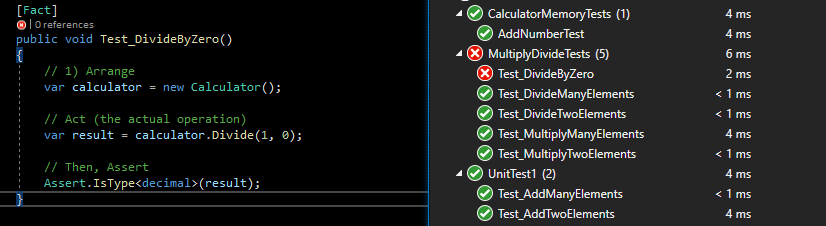
# 04/10 – 3U

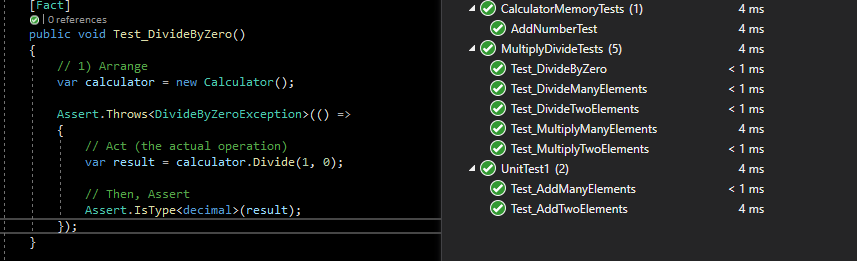
De cursus ging nu verder met het CalculatorProgram. Hier hebben we een nieuwe test ontwikkeld namelijk AddNumberTest. Hier gaan we eerst de current value resetten en dan 2 getallen optellen en dan kijken we met de test of deze 10 zijn wat de optelling zou moeten zijn.

Zoals u hieronder kan zien verloopt deze test succesvol.

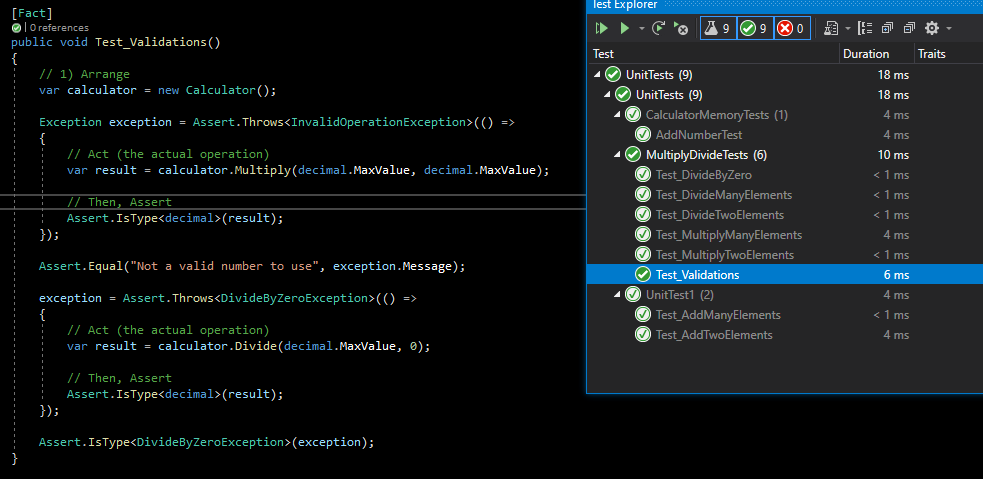


Hierna hebben ze eerst weer wat testjes laten zien in plaats van optellen en aftrekken is het nu met vermenigvuldigen en delen. Hier is er 1 test die mislukt namelijk DivideByZero wat te verwachten was. Vervolgens wordt de test case aangepast met een DivideByZeroException zodat die test niet meer faalt.





Hierna wordt er geleerd hoe je in plaats van exceptions per test te schrijven dat je de gegevens ook kan valideren in een test. Daarmee weet je dan of er geen expections gaan komen in de andere tests.



# Bronnen

https://openclassrooms.com/en/courses/5670586-test-your-net-applications/5945960-why-is-testing-important