**Naam : Jan Dammans (r0796935) Vak: TDD**

**Klasgroep: 1 Academiejaar : 2021-2022 Datum:20/09/2021**

**Onderwerp: Opdracht Software Testing**

# Onderzoek

## Tools

* Visual Studio
* C#
* Selenium
* xUnit

## Verschil xUnit – Nunit

1. xUnit beter voor nieuwe/lege projecten
2. xUnit propere/duidelijkere versie van NUnit
3. xUnit makkelijk uitbreid baar (zoals Xamarin)
4. xUnit is nieuwer dus veranderd soms nog
5. xUnit heeft beter geïsoleerde tests omdat iedere test wordt geïnstantieerd. Zo kunnen de testen elkaar niet beïnvloeden.
6. NUnit kunnen meerdere keren testen hetzelfde benoemen zoals SetUp wat voor complexiteit zorgt.
7. xUnit heeft duidelijkere namen zoals Assert.throw ipv ExpectedExceptation

## Cursus

1. <https://openclassrooms.com/en/courses/5670586-test-your-net-applications>

## Bronnen

1. <https://www.lambdatest.com/blog/xunit-testing-tutorial/>
2. <https://www.lambdatest.com/blog/setting-selenium-webdriver-for-xunit/>
3. <https://code-maze.com/automatic-ui-testing-selenium-asp-net-core-mvc/>
4. Boek: programmeren in c#: hoofdstuk testen

# Cursus

## Testfase

* Reactief: bugs vinden nadat ze zijn opgetreden
* Proactief: bugs voorkomen door te testen

Soorten testen:

* Manueel
  + Zelf gaan kijken of alles klopt
* Automatisch
  + Alles automatisch laten verlopen
  + Soorten automatische testen:
    - **Unit test**: Deze testen een deel van de code die geïsoleerd is van alle afhankelijkheden. Deze zijn eenvoudig te ontwikkelen, kunnen snel worden uitgevoerd, en stellen je in staat om kleine delen van je code direct te controleren.
    - **Mock tests** - Dit zijn unit tests die een "mock" van een deel van de code testen in plaats van het werkelijke deel van de code zelf.
    - **Integratie tests** - Deze controleren of al deze secties van geïsoleerde code correct samenwerken. Het duurt over het algemeen langer dan unit tests om ze uit te voeren en ze kunnen afhankelijkheden testen waar de ontwikkelaar controle over heeft, zoals databases of bestanden.

## Soorten automatische test

**Test-Driven development**

* Eerst testcases, dan code
* Blijven testen totdat alles werkt (proactief – vooral technisch)

**Behaviour-Driven Development**

* Verbeterd communicatie alle teams
* Vooral klantgericht (vooral output)

## TestPrincipe

**F**ast,

**I**solated,

**R**epeatable,

**S**elf-valifating,

**T**imely

## 3A stappenplan

* **Arrange:** Alle noodzakelijke randvoorwaarden en inputs voor het te testen object of de te testen methode regelen.
* **Act:** Voer de actie uit op het geteste object of de geteste methode met de gerangschikte parameters.
* **Assert:** Bevestigen (of verifiëren) dat de actie van het geteste object of de geteste methode zich gedraagt zoals verwacht.

## Tests

Eerst wordt de methode uitgewerkt in code (zie Calculator). Vervolgens maken we dan twee tests die nagaan of de code in de method werkt.

### Calculator

Afbeelding met tekst

Automatisch gegenereerde beschrijving

### Som van 2 elementen

Afbeelding met tekst

Automatisch gegenereerde beschrijving

### Som van meerdere element

Afbeelding met tekst

Automatisch gegenereerde beschrijving

### Resultaat

Bij het runnen van de tests zien we dat alles werkt. Bijgevolg kan de method Sum() in productie gebracht worden.

Afbeelding met tekst

Automatisch gegenereerde beschrijving