Beschikbare MSTestv2 Assert methods

De vergelijking bij Assert methods houdt rekening met het gegevenstype: 10L (long) is bijvoorbeeld verschilllend van 10 (int). Naast de AreEqual() method, bestaan er nog vele andere. Probeer altijd de meest geschikte te gebruiken in functie van leesbaarheid en onderhoudbaarheid.

- **AreNotEqual()**: wanneer we wensen te verifieren dat twee objecten NIET gelijk zijn (de andere "Niet" methods bespreken we verder hier niet meer)
- **AreSame()**: controleert of twee objecten naar hetzelfde object verwijzen (vergelijkt met ==).
- **Equals()**: vergelijkt twee objecten met de Equals() method.
- **Fail()**: veroorzaakt het falen van de test; meestal gebruikt met een voorwaarde en optionele parameters zoals de foutmelding en parameters.
- **Inconclusive()**: vergelijkbaar met Fail(), maar wijst op het niet beslist zijn van de test.
- **IsFalse()**: verifieert of een uitdrukking False is.
- **IsInstanceOfType()**: verifieert of een object een instantie van de specifieke class is.
- **IsNull()**: verifieert dat de waarde null is.
- **IsTrue()**: verifieert dat een uitdrukking True is.
- **ReplaceNullChars()**: vervangt null karakters ("\0") door "\0". Veel gebruikt bij feedback.
- **ThrowsException()**: voert de opgegeven delegate uit en verifieert of er wel een uitzondering opgeworpen wordt van het type dat als generiek argument is meegegeven. Van deze method bestaat ook een asynchrone versie die noemt **ThrowsExceptionAsync()**.

Opgelet: de **ReferenceEquals()** method is een method die op alle classes bestaat en vormt geen onderdeel van deze set Assert() methods.

Initialisatie en opkuis

Wanneer je methods van een class test, kan het gebeuren dat je de class meermaals wenst te initialiseren, namelijk net voor de uitvoering van elke test. Hiervoor kan je het [TestInitialize] attribuut in MSTest (of ([SetUp] in NUnit of de constructur xUnit.NET) gebruiken.

Zo kunnen we ook gebruik maken van het [TestCleanup] attribuut in MSTest ([TearDown] in NUnit of een implementatie van de IDisposable interface in xUnit.NET) om objecten op te kuisen wanneer alle testen uitgevoerd zijn.

In het voorbeeld hieronder, waarin we vooruit lopen op volgende les, schrijven we de initialisatie slechts eenmaal en een instantie van de class zal opgezet worden elkens voor elke test uitgevoerd wordt:

```
ILogger _log;
IWordUtils _wordUtils;

[TestInitialize]
public void Initialize()
{
    _log = Mock.Of<ILogger<WordUtils>>();
    _wordUtils = new WordUtils(_log);
}

[TestMethod]
public void Reverse_ShouldBeWordInReverse_IfWordIsValid()
```

```
{
    string word = "mountain";
    string reverseWord = _wordUtils.Reverse(word);
    reverseWord.ShouldBe("niatnuom");
}

[TestCleanup]
public void Cleanup()
{
    // Optionally dispose or cleanup objects
    ...
}
```