NUnit Assert class is used to determine whether a particular test method gives expected result or not.

In a test method, we write code the check the business object behavior. That business object returns a result. In Assert method we match the actual result with our expected result. If result comes according to our expect result then our test case is passed else failed.

**Constraint Model**

NUnit provides a new Constraint Model to improve the test method readability. In constraint model, we use a single method "That" and specify constraints to check our expected response.

That method apply a constraint to actual value. If constraint is satisfied the our test case is succeed else failed.

**Helper Classes**

Below are helper classes to provide a constraint to assert method.

1. Is
2. Has
3. Contains
4. Does
5. Throws

**Constraint Categories**

These constraints can be divided into eight categories:

1. [**Comparison**](http://dotnetpattern.com/nunit-assert-examples#ComparisonConstraints)
   * [**Equal**](http://dotnetpattern.com/nunit-assert-examples#Equal_Example)
   * [**Not Equal**](http://dotnetpattern.com/nunit-assert-examples#NotEqual_Example)
   * [**Greater Than**](http://dotnetpattern.com/nunit-assert-examples#GreaterThan_Example)
   * [**Less Than**](http://dotnetpattern.com/nunit-assert-examples#LessThan_Example)
   * [**Ranges**](http://dotnetpattern.com/nunit-assert-examples#Ranges_Example)
2. [**String**](http://dotnetpattern.com/nunit-assert-examples#NUnit_String_Constraints)
   * [**Equals and Ignore Case**](http://dotnetpattern.com/nunit-assert-examples#String_Equal_OR_Not_Equal_Example)
   * [**Sub string**](http://dotnetpattern.com/nunit-assert-examples#Assert_Substring_Example)
   * [**Empty**](http://dotnetpattern.com/nunit-assert-examples#String_Empty_Example)
   * [**Starts With / Ends With**](http://dotnetpattern.com/nunit-assert-examples#String_StartsWith_EndsWith_Example)
   * [**Regex**](http://dotnetpattern.com/nunit-assert-examples#Regex_Example)
3. [**Collection**](http://dotnetpattern.com/nunit-assert-examples#NUnit_Collection_Constraints)
   * All Items
     + [**Not Null**](http://dotnetpattern.com/nunit-assert-examples#Not_Null_Example)
     + [**All Greater Than**](http://dotnetpattern.com/nunit-assert-examples#Collection_Elements_Greater_Than_Example)
     + [**All Less Than**](http://dotnetpattern.com/nunit-assert-examples#Collection_Elements_Less_Than_Example)
     + [**Instance Of**](http://dotnetpattern.com/nunit-assert-examples#Collection_Elements_InstanceOf_Example)
   * [**No Items**](http://dotnetpattern.com/nunit-assert-examples#Collection_No_Elements_Example)
   * [**Exactly n Items**](http://dotnetpattern.com/nunit-assert-examples#Collection_Exactly_N_Items_Example)
   * [**Unique Items**](http://dotnetpattern.com/nunit-assert-examples#Collection_Unique_Items_Constraint_Example)
   * [**Contains**](http://dotnetpattern.com/nunit-assert-examples#Collection_Contains_Items_Example)
   * Ordered
     + [**Ascending**](http://dotnetpattern.com/nunit-assert-examples#Collection_Ascending_Elements_Example)
     + [**Descending**](http://dotnetpattern.com/nunit-assert-examples#Collection_Descending_Elements_Example)
     + [**By Single Property**](http://dotnetpattern.com/nunit-assert-examples#Collection_OrderBy_SingleProperty_Example)
     + [**Multiple Properties**](http://dotnetpattern.com/nunit-assert-examples#Collection_OrderBy_MultipleProperties_Example)
   * [**Superset/Subset**](http://dotnetpattern.com/nunit-assert-examples#Collection_Is_SuperSet/Subset_Example)
4. [**Conditional**](http://dotnetpattern.com/nunit-assert-examples#NUnit_Conditional_Constraints)
   * [**Null**](http://dotnetpattern.com/nunit-assert-examples#Null_Constraint_Example)
   * [**Boolean (True / False)**](http://dotnetpattern.com/nunit-assert-examples#Boolean_TrueFalse_Example)
   * [**Empty**](http://dotnetpattern.com/nunit-assert-examples#Empty_Element_Example)
5. [**Compound**](http://dotnetpattern.com/nunit-assert-examples#NUnit_Compound_Constraints)
   * [**And**](http://dotnetpattern.com/nunit-assert-examples#Items_And_Example)
   * [**Not**](http://dotnetpattern.com/nunit-assert-examples#Items_Not_EqualTo_Example)
   * [**Or**](http://dotnetpattern.com/nunit-assert-examples#Items_Or_Example)
6. [**Directory/File**](http://dotnetpattern.com/nunit-assert-examples#NUnit_Directory_File_Constraints)
   * [**File or Directory Exits**](http://dotnetpattern.com/nunit-assert-examples#FileDirectory_Exists_Example)
   * [**Same Path**](http://dotnetpattern.com/nunit-assert-examples#FileDirectory_SamePath_Example)
   * [**Empty Directory**](http://dotnetpattern.com/nunit-assert-examples#EmptyDirectory_Example)
7. [**Type/Reference**](http://dotnetpattern.com/nunit-assert-examples#NUnit_Type_Constraints)
   * [**Instance Of**](http://dotnetpattern.com/nunit-assert-examples#InstanceOf_Example)
   * [**Same Type**](http://dotnetpattern.com/nunit-assert-examples#ExactSameType_Example)
   * [**Assignable to another Type**](http://dotnetpattern.com/nunit-assert-examples#AssignableToAnotherType_Constraint_Example)
8. [**Exceptions**](http://dotnetpattern.com/nunit-assert-examples#NUnit_Exceptions_Constraints)
   * [**Is Exception throws**](http://dotnetpattern.com/nunit-assert-examples#IsExceptionThrowsByMethod_Example)
   * [**Expected Exception**](http://dotnetpattern.com/nunit-assert-examples#Exception_Type_Example)
   * [**Exception Message Comparison**](http://dotnetpattern.com/nunit-assert-examples#Exception_Message_Comparison_Example)

**Comparison Constraints**

**Equal Constraint Example**

|  |  |
| --- | --- |
| 1 | Assert.That(result, Is.EqualTo(5)); |

**Not Equal Constraint Example**

|  |  |
| --- | --- |
| 1 | Assert.That(result, Is.Not.EqualTo(7)); |

**Greater Than Constraint Example**

|  |  |
| --- | --- |
| 1  2 | Assert.That(result, Is.GreaterThan(2));  Assert.That(result, Is.GreaterThanOrEqualTo(5)); |

**Less Than Constraint Example**

|  |  |
| --- | --- |
| 1  2 | Assert.That(result, Is.LessThan(9));  Assert.That(result, Is.LessThanOrEqualTo(5)); |

**Ranges Example**

|  |  |
| --- | --- |
| 1 | Assert.That(result, Is.InRange(5, 10)); |

**String Constraints**

**String Equal / Not Equal Constraint Example**

|  |  |
| --- | --- |
| 1  2 | Assert.That(result, Is.EqualTo("abcdefg"));  Assert.That(result, Is.Not.EqualTo("mnop")); |

**String Equal Ignore Case Example**

|  |  |
| --- | --- |
| 1 | Assert.That(result, Is.EqualTo("ABCDEFG").IgnoreCase); |

**Substring Constraint Example**

|  |  |
| --- | --- |
| 1  2 | Assert.That(result, Does.Contain("def").IgnoreCase);  Assert.That(result, Does.Not.Contain("igk").IgnoreCase); |

**Empty Example**

|  |  |
| --- | --- |
| 1  2 | Assert.That(result, Is.Empty);  Assert.That(result, Is.Not.Empty); |

**Starts With / Ends With Examples**

|  |  |
| --- | --- |
| 1  2  3  4  5 | Assert.That(result, Does.StartWith("abc"));  Assert.That(result, Does.Not.StartWith("efg"));    Assert.That(result, Does.EndWith("efg"));  Assert.That(result, Does.Not.EndWith("mno")); |

**Regex Constraint Example**

|  |  |
| --- | --- |
| 1  2  3  4 | string result = "abcdefg";    Assert.That(result, Does.Match("a\*g"));  Assert.That(result, Does.Not.Match("m\*n")); |

**Collection Constraints**

All Items Examples

|  |  |
| --- | --- |
| 1 | int[] array = new int[] { 1, 2, 3, 4, 5 }; |

**Not Null Example**

|  |  |
| --- | --- |
| 1 | Assert.That(array, Is.All.Not.Null); |

**All Greater Than Example**

|  |  |
| --- | --- |
| 1 | Assert.That(array, Is.All.GreaterThan(0)); |

**All Less Than Example**

|  |  |
| --- | --- |
| 1 | Assert.That(array, Is.All.LessThan(10)); |

**Instance Of Example**

|  |  |
| --- | --- |
| 1 | Assert.That(array, Is.All.InstanceOf<Int32>()); |

**No Items Example**

|  |  |
| --- | --- |
| 1  2 | Assert.That(array, Is.Empty);  Assert.That(array, Is.Not.Empty); |

**Exactly n Items Example**

|  |  |
| --- | --- |
| 1 | Assert.That(array, Has.Exactly(5).Items); |

**Unique Items Example**

|  |  |
| --- | --- |
| 1 | Assert.That(array, Is.Unique); |

**Contains Item**

|  |  |
| --- | --- |
| 1 | Assert.That(array, Contains.Item(4)); |

**Ordered Examples**

**Ascending**

|  |  |
| --- | --- |
| 1 | Assert.That(array, Is.Ordered.Ascending); |

**Descending**

|  |  |
| --- | --- |
| 1 | Assert.That(array, Is.Ordered.Descending); |

**By Single Property**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | List<Employee> employees = new List<Employee>();  employees.Add(new Employee { Age = 32 });  employees.Add(new Employee { Age = 49 });  employees.Add(new Employee { Age = 57 });    Assert.That(employees, Is.Ordered.Ascending.By("Age"));  Assert.That(employees, Is.Ordered.Descending.By("Age")); |

**By Multiple Properties**

|  |  |
| --- | --- |
| 1 | Assert.That(employees, Is.Ordered.Ascending.By("Age").Then.Descending.By("Name")); |

**SuperSet / SubSet Examples**

|  |  |
| --- | --- |
| 1  2  3 | int[] array = new int[] { 1, 2, 3, 4, 5 };  int[] array2 = { 3, 4 };  Assert.That(array2, Is.SubsetOf(array)); |

**Conditional Constraints**

**Null Constraint Examples**

|  |  |
| --- | --- |
| 1  2 | Assert.That(array, Is.Null);  Assert.That(array, Is.Not.Null); |

**Boolean (True / False)**

|  |  |
| --- | --- |
| 1  2  3  4 | bool result = array.Length > 0;  Assert.That(result, Is.True);    Assert.That(result, Is.False); |

**Empty Constraint Example**

|  |  |
| --- | --- |
| 1 | Assert.That(array, Is.Empty); |

**Compound Constraints**

**AND constraint example**

|  |  |
| --- | --- |
| 1 | Assert.That(result, Is.GreaterThan(4).And.LessThan(10)); |

**OR example**

|  |  |
| --- | --- |
| 1 | Assert.That(result, Is.LessThan(1).Or.GreaterThan(4)); |

**NOT example**

|  |  |
| --- | --- |
| 1 | Assert.That(result, Is.Not.EqualTo(7)); |

**Directory / File Constraints**

**File or Directory exists.**

|  |  |
| --- | --- |
| 1  2  3  4  5 | Assert.That(new FileInfo(path), Does.Exist);  Assert.That(new FileInfo(path), Does.Not.Exist);    Assert.That(new DirectoryInfo(path), Does.Exist);  Assert.That(new DirectoryInfo(path), Does.Not.Exist); |

**Same Path Example**

|  |  |
| --- | --- |
| 1 | Assert.That(path, Is.SamePath(@"c:\documents\imp1").IgnoreCase); |

**Empty Directory. Is.Empty returns true when directory has no files.**

|  |  |
| --- | --- |
| 1 | Assert.That(new DirectoryInfo(path), Is.Empty); |

**Type / Reference Constraints**

**Instance of example**

|  |  |
| --- | --- |
| 1  2  3  4 | IEmployee emp = new Employee();    Assert.That(emp, Is.InstanceOf<IEmployee>());  Assert.That(emp, Is.Not.InstanceOf<string>()); |

**Exact Same Type Constraint**

|  |  |
| --- | --- |
| 1 | Assert.That(emp, Is.TypeOf<Employee>()); |

**Assignable to another Type.** For e.g. interface to implemented class.

|  |  |
| --- | --- |
| 1 | Assert.That(emp, Is.AssignableTo<Employee>()); |

**Exceptions Constraints**

**Is Exception Throws By Method**

|  |  |
| --- | --- |
| 1  2  3  4 | IEmployee emp = new Employee();  emp.Age = 0;    Assert.That(emp.IsSeniorCitizen(), Throws.Exception); |

**Expected / Same Type Exception**

|  |  |
| --- | --- |
| 1 | Assert.That(emp.IsSeniorCitizen(), Throws.TypeOf<ArgumentException>()); |

**Exception Message Comparison**

|  |  |
| --- | --- |
| 1  2 | Exception ex = Assert.Throws<ArgumentException>(() => emp.IsSeniorCitizen());  Assert.That(ex.Message, Is.EqualTo("Age can not 0.")); |