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 expected 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 applies a constraint to the actual value. If a constraint is satisfied our test case is succeeded else failed.

Helper Classes

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

Is

Has

Contains

Does

Throws

Constraint Categories

These constraints can be divided into eight categories:

Comparison

Equal

Not Equal

Greater Than

Less Than

Ranges

String

Equals and Ignore Case

Sub string

Empty

Starts With / Ends With

Regex

Collection

All Items

Not Null

All Greater Than

All Less Than

Instance Of

No Items

Exactly n Items

Unique Items

Contains

Ordered

Ascending

Descending

By Single Property

Multiple Properties

Superset/Subset

Conditional

Null

Boolean (True / False)

Empty

Compound

And

Not

Or

Directory/File

File or Directory Exits

Same Path

Empty Directory

Type/Reference

Instance Of

Same Type

Assignable to another Type

Exceptions

Is Exception throws

Expected Exception

Exception Message Comparison

Comparison Constraints

Equal Constraint Example

Assert.That(result, Is.EqualTo(5));

Not Equal Constraint Example

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

Greater Than Constraint Example

Assert.That(result, Is.GreaterThan(2));

Assert.That(result, Is.GreaterThanOrEqualTo(5));

Less Than Constraint Example

Assert.That(result, Is.LessThan(9));

Assert.That(result, Is.LessThanOrEqualTo(5));

Ranges Example

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

String Constraints

String Equal / Not Equal Constraint Example

Assert.That(result, Is.EqualTo("abcdefg"));

Assert.That(result, Is.Not.EqualTo("mnop"));

String Equal Ignore Case Example

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

Substring Constraint Example

Assert.That(result, Does.Contain("def").IgnoreCase);

Assert.That(result, Does.Not.Contain("igk").IgnoreCase);

Empty Example

Assert.That(result, Is.Empty);

Assert.That(result, Is.Not.Empty);

Starts With / Ends With Examples

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

string result = "abcdefg";

Assert.That(result, Does.Match("a\*g"));

Assert.That(result, Does.Not.Match("m\*n"));

Collection Constraints

All Items Examples

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

Not Null Example

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

All Greater Than Example

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

All Less Than Example

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

Instance Of Example

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

No Items Example

Assert.That(array, Is.Empty);

Assert.That(array, Is.Not.Empty);

Exactly n Items Example

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

Unique Items Example

Assert.That(array, Is.Unique);

Contains Item

Assert.That(array, Contains.Item(4));

Ordered Examples

Ascending

Assert.That(array, Is.Ordered.Ascending);

Descending

Assert.That(array, Is.Ordered.Descending);

By Single Property

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

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

SuperSet / SubSet Examples

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

int[] array2 = { 3, 4 };

Assert.That(array2, Is.SubsetOf(array));

Conditional Constraints

Null Constraint Examples

Assert.That(array, Is.Null);

Assert.That(array, Is.Not.Null);

Boolean (True / False)

bool result = array.Length > 0;

Assert.That(result, Is.True);

Assert.That(result, Is.False);

Empty Constraint Example

Assert.That(array, Is.Empty);

Compound Constraints

AND constraint example

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

OR example

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

NOT example

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

Directory / File Constraints

File or Directory exists.

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

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

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

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

Type / Reference Constraints

Instance of example

IEmployee emp = new Employee();

Assert.That(emp, Is.InstanceOf<IEmployee>());

Assert.That(emp, Is.Not.InstanceOf<string>());

Exact Same Type Constraint

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

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

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

Exceptions Constraints

Is Exception Throws By Method

IEmployee emp = new Employee();

emp.Age = 0;

Assert.That(emp.IsSeniorCitizen(), Throws.Exception);

Expected / Same Type Exception

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

Exception Message Comparison

Exception ex = Assert.Throws<ArgumentException>(() => emp.IsSeniorCitizen());

Assert.That(ex.Message, Is.EqualTo("Age can not 0."));