

# AdomdTests how to configure in Windows Server

## Tools needed:

- .Net Framework;
- Ant (apache-ant-1.9.3-bin);
- Java (jdk-7u45-windows.i586);
- Jenkins Server (jenkins-1.546);
- Nunit Jenkins Plugin;
- Nunit (NUnit-2.6.3).

## Installation:

- Install .Net Framework;
- Install Visual Studio (optional);
- Install Java Jdk;
- Install Ant;
- Install Jenkins;
- Install Nunit;
- Download pentaho biserver;

## Prepare System Environment:

Configure system variables and paths, with a command line tools or windows system environment. Start by creating:

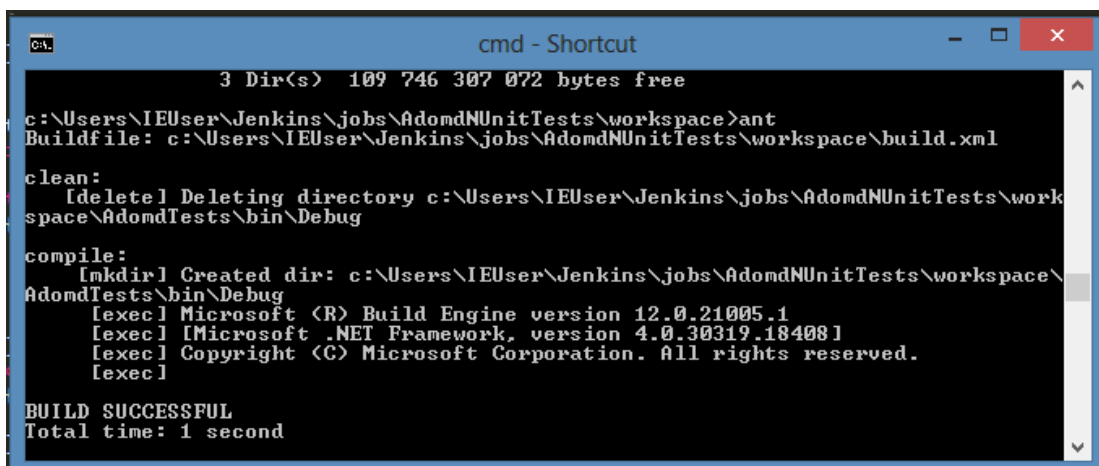
- JAVA\_HOME variable path to JDK folder (ex: C:\ProgramFiles\Java\jdk1.7.0\_45);
- ANT\_HOME variable path to Ant folder (ex: C:\ant);
- MS\_BUILD variable path to MSBuild folder (ex: C:\ProgramFiles\MSBuild);
- NUNIT variable path to Nunit path folder (ex: [C:\NUnit-2.6.3](#));

and finally:

- PATH variable to respective binary folders (Ex: PATH=%PATH%;%JAVA\_HOME%\bin;%ANT\_HOME%\bin;%MS\_BUILD%\bin;%NUNIT%\bin). May have to restart windows for those changes to take effect. Test commands MSBuild, ant, nunit-console or nunit, java -version.

## Project:

- Chekout the project AdomdTests from repository (<https://github.com/portisheadaff/adomdtest>);
- Run ant in the project folder, it should compile tests and creates the test dll library.



```
cmd - Shortcut
3 Dir(s) 109 746 307 072 bytes free
c:\Users\IEUser\Jenkins\jobs\AdomdNUnitTests\workspace>ant
Buildfile: c:\Users\IEUser\Jenkins\jobs\AdomdNUnitTests\workspace\build.xml

clean:
[delete] Deleting directory c:\Users\IEUser\Jenkins\jobs\AdomdNUnitTests\workspace\AdomdTests\bin\Debug

compile:
[mkdir] Created dir: c:\Users\IEUser\Jenkins\jobs\AdomdNUnitTests\workspace\AdomdTests\bin\Debug
[exec] Microsoft (R) Build Engine version 12.0.21005.1
[exec] [Microsoft .NET Framework, version 4.0.30319.18408]
[exec] Copyright (C) Microsoft Corporation. All rights reserved.
[exec]

BUILD SUCCESSFUL
Total time: 1 second
```

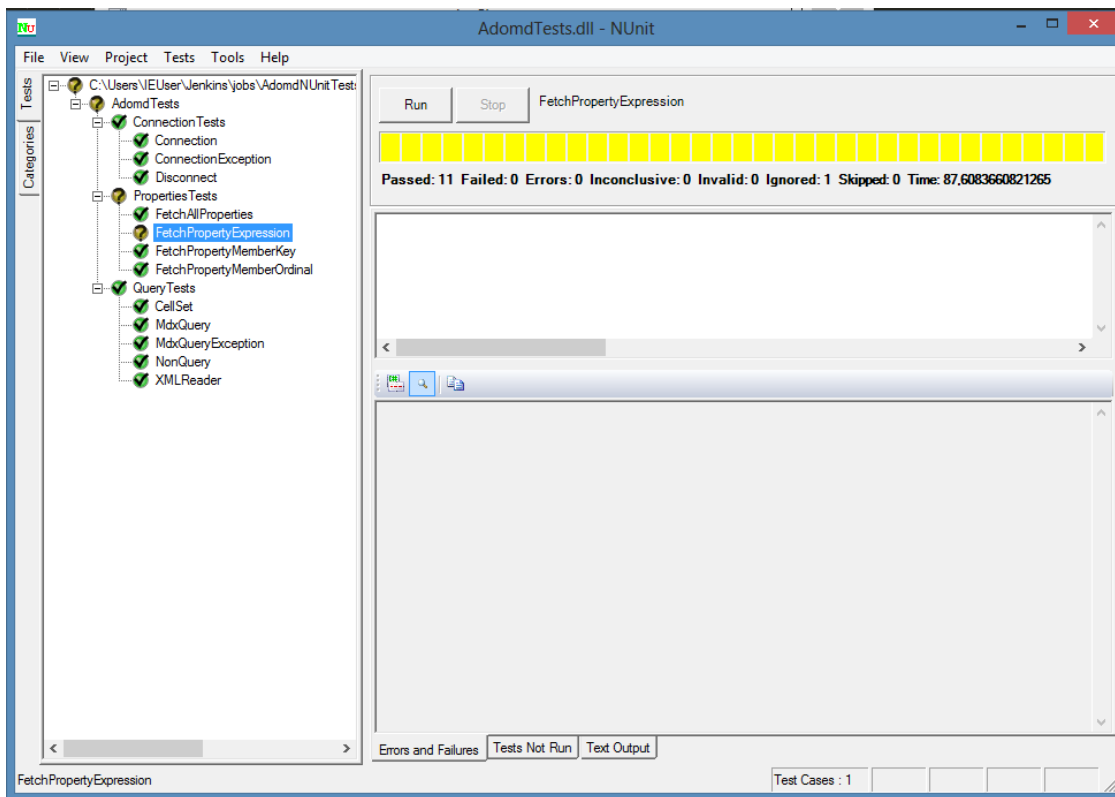
- Before you can run the tests configure biserver on port **8089** change the configuration of tomcat/config/server.xml, because jenkins may be running in port 8080.
- Start biserver running the command start-pentaho.bat.

- With server started run ant tests command line;

If everything worked has expected it's time to configure Jenkins, but before this let's know about NUnit app it's a graphical tool to run tests, it also can be integrated with visual studio (optional).

## NUnit app:

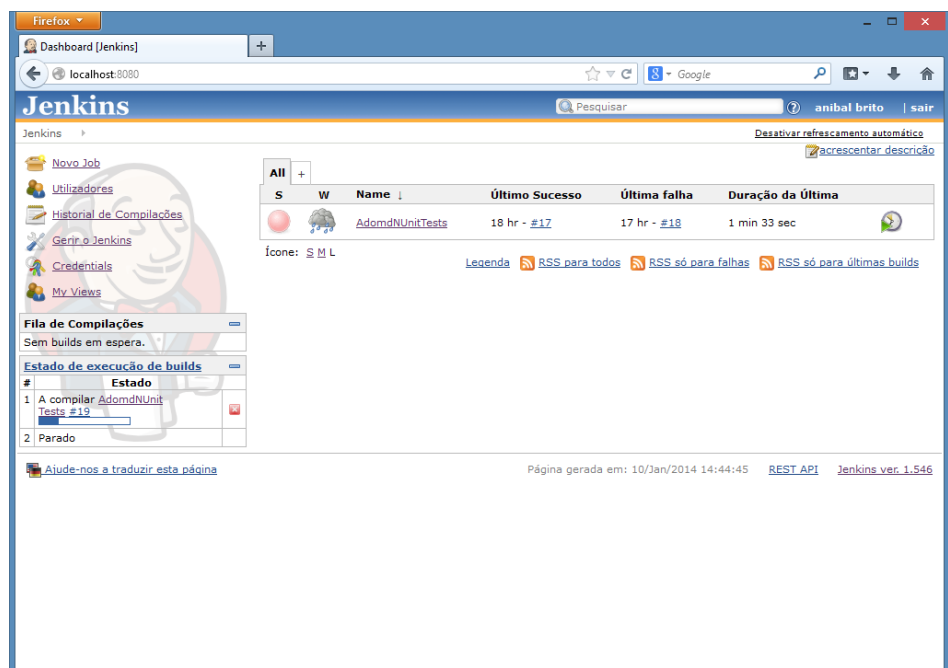
The NUnit app is an application to run tests in standalone mode. You can open the AdomdTests.dll and start running the tests.



Very similar to other Unit testing frameworks, you can run all the tests, suites of tests or standalone tests, remember the biserver must be running or all the tests will fail. You can have a plugin of NUnit installed in your visual studio.

## Jenkins configuration:

After installation of Jenkins you should have a windows service running. If that was your installation option.



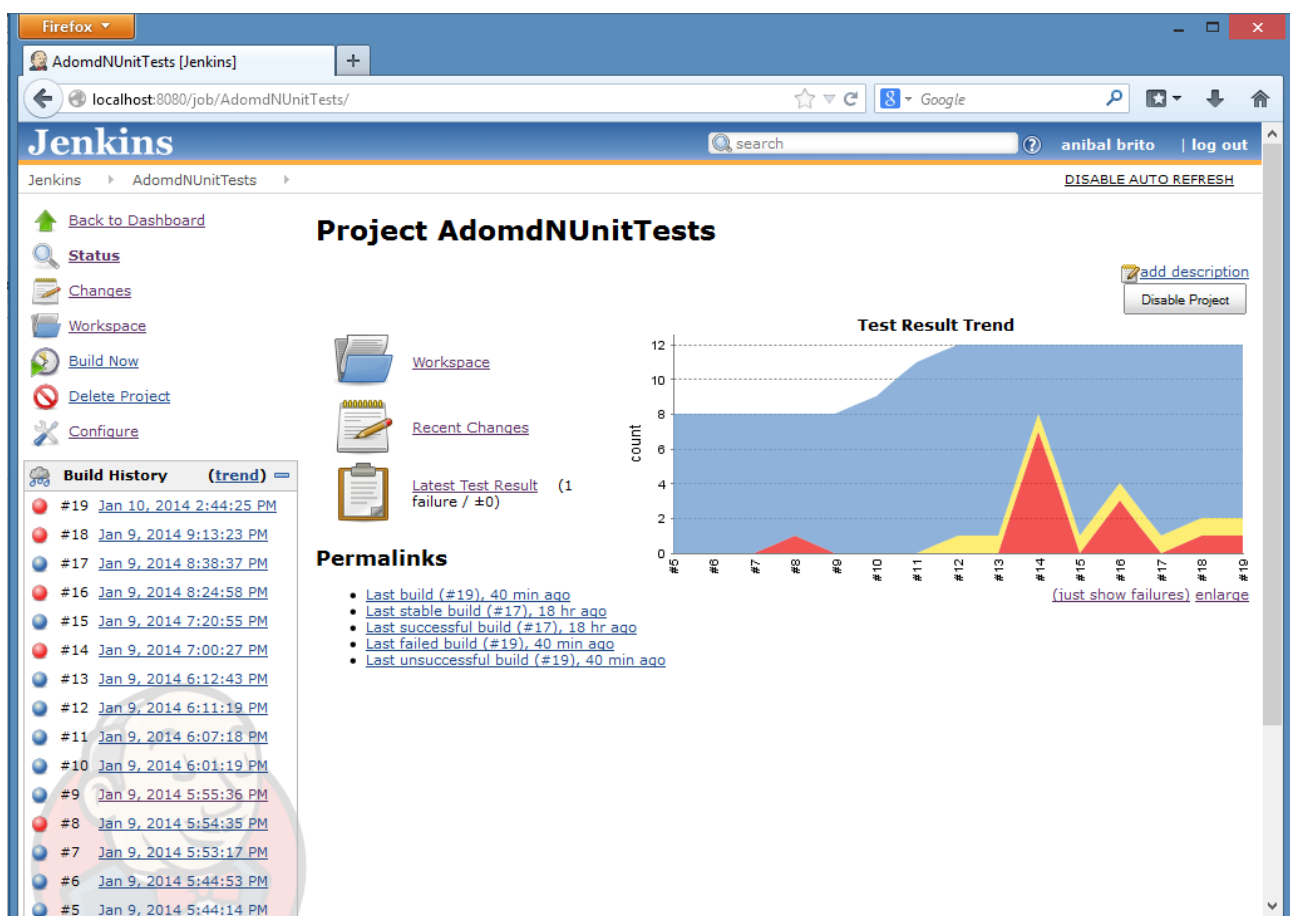
Go to Manage Jenkins, then choose manage plugins, search for Jenkins NUnit plugin and install it.

After plugin installation in the tab “Installed” of manage plugins it should appear this line.

|                                     |   |                      |  |                            |
|-------------------------------------|---|----------------------|--|----------------------------|
|                                     | Netbeans cvsclient.   |                      |  |                            |
| <input checked="" type="checkbox"/> | <a href="#">Jenkins NUnit plugin</a><br>This plugin transforms <a href="#">JUnit</a> test reports so they can be recorded by Jenkins' JUnit features. | <a href="#">0.15</a> |  | <button>Uninstall</button> |
|                                     | <a href="#">Jenkins SSH Slave plugin</a>  |                      |  |                            |

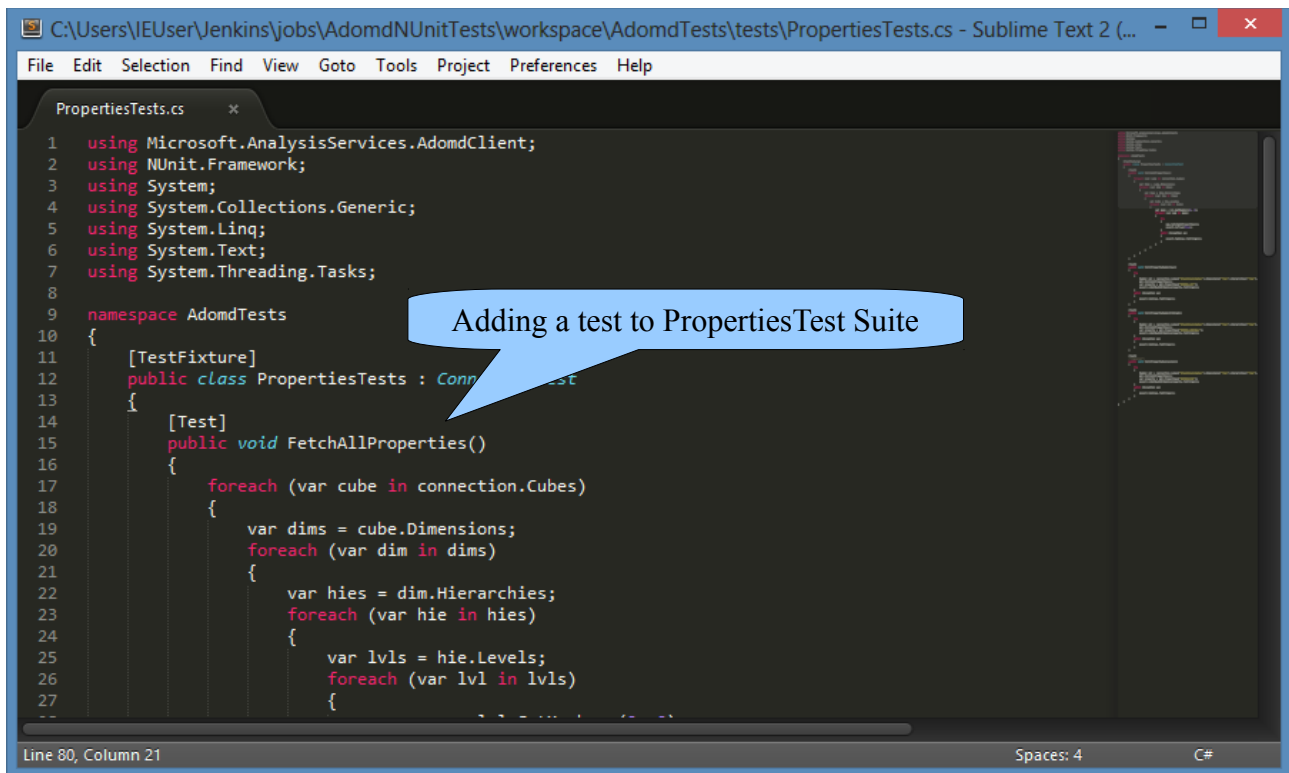
After this, configure a Job in Jenkins and put it all together.

- Create a new Job in Jenkins (ex: AdomdNUnitTests);
- In Build add “Execute Windows Batch Command” and type in command text input: c:\biserver-location\start-pentaho.bat;
- In Build add “Invoke Ant Task” and type in Targets text input: tests, this will run the clean, compile and tests target ant tasks;
- In Build add “Execute Windows Batch Command” and type in command text input: c:\biserver-location\stop-pentaho.bat;
- In Post-build Actions, add “Publish NUnit test report result” and type TestResults.xml;
- Save the Job;
- Upload the project to workspace;
- Run the Job.



## How to add more tests:

- NUnit is based on JUnit test framework so is simple to add more tests.
- If you want to write a new Test Suite just create new class and add the tests.
- If you want to write a test in Existing Test Suites just add the test.
- Using Sublime Text 2 or other text application to add a test:



The screenshot shows the Sublime Text 2 editor with the file `PropertiesTests.cs` open. The code is in C# and defines a test suite. A blue callout bubble with the text "Adding a test to PropertiesTest Suite" points to the `FetchAllProperties()` method. The code includes several using statements, a namespace, and a test fixture class.

```
1 using Microsoft.AnalysisServices.AdomdClient;
2 using NUnit.Framework;
3 using System;
4 using System.Collections.Generic;
5 using System.Linq;
6 using System.Text;
7 using System.Threading.Tasks;
8
9 namespace AdomdTests
10 {
11     [TestFixture]
12     public class PropertiesTests : ConnectionTest
13     {
14         [Test]
15         public void FetchAllProperties()
16         {
17             foreach (var cube in connection.Cubes)
18             {
19                 var dims = cube.Dimensions;
20                 foreach (var dim in dims)
21                 {
22                     var hies = dim.Hierarchies;
23                     foreach (var hie in hies)
24                     {
25                         var lvls = hie.Levels;
26                         foreach (var lvl in lvls)
27                         {
28                             // Test logic here
29                         }
30                     }
31                 }
32             }
33         }
34     }
35 }
```

- run ant command line and use NUnit app if you want a graphical environment;
- run ant tests command line if not using NUnit app;
- Using Visual Studio, you can configure NUnit Plugin for tests (several plugins) or use NUnit app to run the tests, everytime you build the solution tests are automatically updated inside NUnit app.