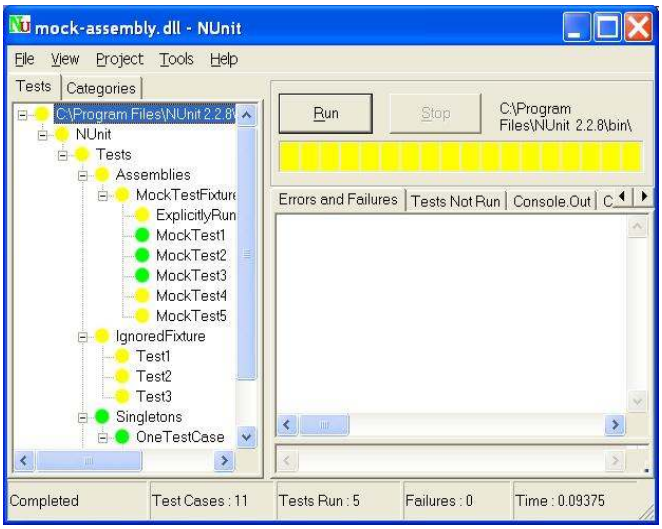
Integrating Nunit results in Aligned Elements

Unit testing as part of Test Driven Development

Test Driven Development (TDD) has appropriately received increasing attention and is nowadays regarded a best-practice in software development.



TDD advocates writing test before code and that tests shall be run repeatedly, quickly and often to detect errors early during development. Unit testing is the typical type of test that satisfies these criteria.

In the .NET community, Nunit has become one of the most widespread unit tests programs and provides an ever increasing set of features. Nunit is open source and can be run from either as a console application or as a GUI app. The results are either generated in an xml file or are displayed in the GUI application.

To perform the integration of the Nunit results in Aligned Elements the following items are needed:

- The xml result output from an Nunit test-run.
- An additional Aligned Elements Document Object template, in this example called "NunitResult" (NunitResult.rvt) as a container for the Nunit results in Aligned Element.
- The xsl transformation file Nunit2AE.xsl file that transforms Nunit results to importable document objects of NunitResult.rvt type.

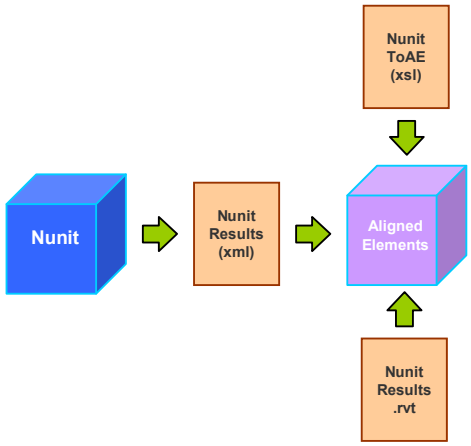
These mentioned items can be downloaded from the Aligned Elements website.

aligned

Transforming the Nunit data into Aligned Elements xml

The key concepts we are exploiting is that:

- Nunit can export unit tests results as xml and
- Aligned Elements can transform and import Document Object information from xml.



So all we need is to transform the Nunit xml to a format that is known by Aligned Elements.

```
- <results>
  - <test-case name="NUnit.Tests.Assemblies.MockTestFixture.ExplicitlyRunTest" executed="False">
    - <categories>
        <category name="Special"/>
      </categories>
    + <reason></reason>
    </test-case>
    <test-case name="NUnit.Tests.Assemblies.MockTestFixture.MockTest1" description="Mock Test #1" executed="True" success="True" time="0.000" asserts="0"/>
  - <test-case name="NUnit.Tests.Assemblies.MockTestFixture.MockTest2" executed="True" success="True" time="0.000" asserts="0">
    + <categories></categories>
    </test-case>
  - <test-case name="NUnit.Tests.Assemblies.MockTestFixture.MockTest3" executed="True" success="True" time="0.000" asserts="0">
    + <categories></categories>
    </test-case>
  - <test-case name="NUnit.Tests.Assemblies.MockTestFixture.MockTest4" executed="False">
```

We will apply the standard way to transform xml by using XSLT (Extensible Stylesheet Language Transformation). In the NunitResultsInAlignedElement package you will find the xsl file called Nunit2AE.xsl which is to be used for this transformation.

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" version="1.0" encoding="utf-8" indent="yes"/>
  <xsl:template match="/">
    <DocumentObjects xmlns="urn:RevisableObjectImportTemplate.xsd">
      <xsl:apply-templates />
    </DocumentObjects>
  </xsl:template>

  <xsl:template match="test-results">
    <xsl:apply-templates select="//test-case" />
  </xsl:template>

  <xsl:template match="test-case">
    <xsl:element name="DocumentObject" namespace="urn:RevisableObjectImportTemplate.xsd">
        <xsl:attribute name="ProjectName">NunitOutput</xsl:attribute>
        <xsl:attribute name="TypeName">NunitTest</xsl:attribute>
      <xsl:variable name="title">
        <xsl:call-template name="lastIndexOf">
```

By applying this xsl transformation to the nunitresult.xml file, we can now convert the output file to a xml-format known by Aligned Elements.

Defining a NunitResult.rvt template file

Our last step is to define a Document Object type in Aligned Elements that represents the Nunit Results. In this example, we have chosen to create a new type called NunitResult which has two Boolean attributes, namely "Success" and "Executed".

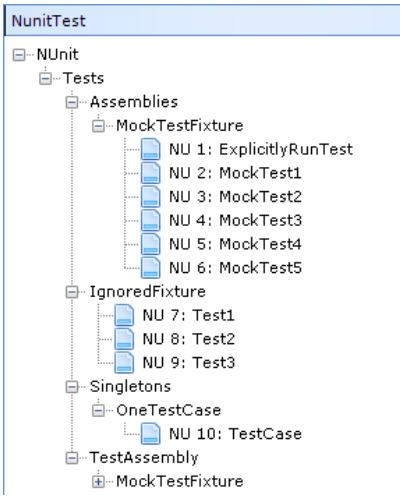
It is, of course, possible to extend the template with further attributes, such as "Description", "Number of Asserts" or any information available in the Nunit result xml file. Consequentially, the xsl file must be updated accordingly.

Importing the results into Aligned Elements

When importing the xml file into Aligned Elements, we are told that the xml file format is unknown to Aligned Elements and we are asked if we want to apply an xsl transformation.

Here, we can now apply our prepared "Nunit to AE" xsl file. The nunit results are automatically converted to a format known by Aligned Elements.

In the following import dialog, we select the option "Recreate Chapters" in the import dialog and complete the import.

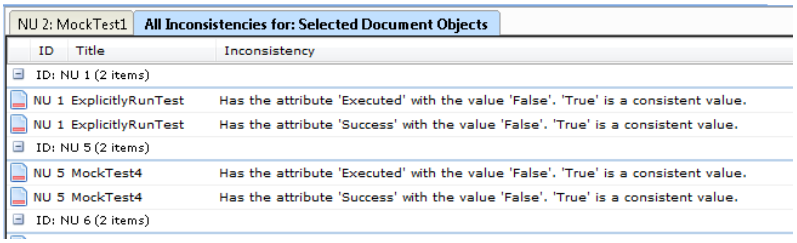


After the import, the Project Explorer displays the following state. We can here clearly see that the title of the Nunit Document Object is the same as the Nunit test function name. The chapter path is constructed by parsing the test-suite chain of each test.

Aligned Elements allows us to perform all regular Aligned Elements operation on these NunitTest Document Objects, including:

- Setting traces to and from the objects
- Use as objects in trace tables
- Take Snapshots on the current state of the objects
- If a Word Template is constructed, export the objects into Word Documents
- Perform Reviews on the objects
- Search among the objects using queries
- Add the objects to the DHF Index
- Perform Inconsistency Analysis on the objects

In our example, the NunitResult Document Object type has been set up to generate inconsistencies if any of the attributes "Executed" and "Success" are false. The result of an inconsistency analysis of the imported Nunit objects could look like this:



Update the Nunit results in Aligned Elements after next test run

It is not unlikely that we will run these Nunit tests several times and that we wish to update our Nunit test results in Aligned Elements with the results from the latest test run. For this purpose we can use the Aligned Elements "Synchronized Import"-feature to overwrite existing results with imported results.

In contrast to regular import, where new Document Objects are created at every import, Synchronized Import implies that the data of an existing Document Object is overwritten with the imported data. This is thus a one-way synchronization option of data from external systems with Document Object data in Aligned Elements.

The Synchronized Import can be performed "manually" for each object where the user explicitly points out which Document Objects that shall be overwritten with which imported data.

The more convenient approach is to let Aligned Elements automatically produce a "Sync Map File" at the first import and later use this map file to automatically find out which object shall be overwritten. The Sync Map File presupposes that the external data has some unique ID that can be mapped to the Aligned Elements Document Object ID in order for the match between external data and Aligned Elements Document Object to take place.

In our case the external ID consists of the Nunit xml "name"-attribute in the "test-case"-element which is a combination of the test-suite path and the test function name e.g.

<test-case name="**NUnit.Tests.Assemblies.MockTestFixture.MockTest1**" …

To generate the Sync Map File at the first import, we need to select the "Create Sync Map File" in the Import dialog. You will be prompted to save the file after the import is completed.

The second time we import the results, we select the option Synchronized Import , "Use Map File" and specify which map file to use. The synchronization mapping of the imported objects are not automatically assigned as specified in the import synch map file. We can perform this action each time we want to update our Aligned Elements Nunit objects with data from the latest test run.

Conclusion

Aligned Elements allows us to import Nunit test results and thereafter perform a large number of regular Aligned Elements-actions on these results. By combining the benefits of automated unit tests with the powerful analysis features of Aligned Elements, time is saved, quality is increased and trace coverage is expanded.