NPanday 1.1

v.

The NPanday Team

1

1.1 NPanday Conventions

The following sections describe the conventions used within NPanday itself. This section is useful for developers wishing to contribute to NPanday, as well as developers looking for a baseline for their own projects. These conventions are evolving and subject to change as better ideas emerge: Got better ideas?

- Artifact ID - specified within the pom - is equivalent to the project's module name.

  Project Structure

  ```
  |-- NPanday.Artifact
  |   `-- main
  |       `-- csharp
  |           `-- NPanday
  |               `-- Artifact
  |                   `-- ArtifactContext.cs
  |                   `-- Artifact.cs
   `-- pom.xml
  ```

  pom.xml file

  ```
  <project xmlns="http://maven.apache.org/POM/4.0.0">
    <modelVersion>4.0.0</modelVersion>
    <groupId>npanday.artifact</groupId>
    <artifactId>NPanday.Artifact</artifactId>
    <packaging>library</packaging>
    <version>0.9</version>
    <name>NPanday.Artifact</name>
  </project>
  ```

- If the module does not contain children modules, the Group ID is the same as the artifact ID.

  ```
  <project xmlns=&quot;http://maven.apache.org/POM/4.0.0&quot;>
    <modelVersion>4.0.0</modelVersion>
    <groupId>npanday.artifact</groupId>
    <artifactId>NPanday.Artifact<artifactId>
    <packaging>library</packaging>
    <version>0.9</version>
    <name>NPanday.Artifact</name>
  </project>
  ```

- If a module contains children modules, the child module Group ID should either be equivalent to a pluralized parent module Group ID or be a deriviative of the parent module Group ID.

  ```
  parent Group ID: NPanday.Model
  child Group ID: NPanday.Model, NPanday.Models or NPanday.Model.VSContent
  ```

- The directory structure of the source directory (typically src/main/csharp) will follow the same pattern as the group ID.

```
 |-- NPanday.Artifact
 |   `-- main
 |       `-- csharp
 |           `-- NPanday
 |               `-- Artifact
 |                   `-- ArtifactContext.cs
 |                   `-- Artifact.cs
  `-- pom.xml
```

```
<project xmlns=&quot;http://maven.apache.org/POM/4.0.0&quot;>
  <modelVersion>4.0.0</modelVersion>
  <groupId>npanday.artifact</groupId>
  <artifactId>NPanday.Artifact<artifactId>
  <packaging>library</packaging>
  <version>0.9</version>
  <name>NPanday.Artifact</name>
</project>
```

- If an assembly will only compile under a specific platform, those values should be specified within the compiler-config.

```
 <project xmlns="http://maven.apache.org/POM/4.0.0">
  <modelVersion>4.0.0</modelVersion>
  <groupId>npanday.plugins</groupId>
  <artifactId>NPanday.Plugins</artifactId>
  <packaging>pom</packaging>
  <version>0.9</version>
  <name>NPanday.Plugins</name>
  <build>
    <sourceDirectory>src/main/csharp</sourceDirectory>
    <testSourceDirectory>src/test/csharp</testSourceDirectory>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.dotnet.plugins</groupId>
        <artifactId>maven-compile-plugin</artifactId>
        <extensions>true</extensions>
        <configuration>
          <vendor>MONO</vendor>
          <frameworkVersion>2.0.50727</frameworkVersion>
          <vendorVersion>1.2.3.1</vendorVersion>
        </configuration>
      </plugin>
    </plugins>
  </build>
</project>
```

- Use the default setup within the npanday-settings.xml is to configure cross-platform builds.

2

2.1 NPanday Frequently Asked Questions

2.1.1 Why is NPanday named as such?

Since NPanday is a project that builds .NET Applications, we brainstormed for a name that would symbolize a great builder, a builder that would have more freedom from its predecessor.

`Panday` is a bisaya word for carpenter/builder and at the same time `Panday` is a fictional Filipino comic hero that would fight monsters using a dagger which magically turns into a sword when raised into the sky. The materials of the dagger came from a meteorite that struck down on earth during the reign of the monsters and supernatural beings. `Panday` fights to bring back freedom and peace to the people once more.

2.1.2 What are the requirements needed to run NPanday?

You would need to install the following:

- Nunit
- Java 1.5 or higher
- Apache Maven 2.0.4 or higher

2.1.3 Do you need to have intensive knowledge on the Apache Maven or Java in order to run NPanday?

No. Since NPanday aims to run on the back ground. The user will have minimal interaction with Apache Maven.

2.1.4 Can NPanday build projects outside of VS since it is a VS Addin?

Yes. Since NPanday creates a pom file from your corresponding .Net Project you can build your .Net Projects using Apache Maven commands.

2.1.5 Why use NPanday when you can build .Net Applications in Visual Studio?

- NPanday aims for Continuous Integration and Artifact Repository Management for your .Net Applications using Open Source Technology. By using NPanday, you can take advantage of existing development infrastructure that is compatible with Maven.

2.1.6 How can a custom `settings.xml` be used for the Visual Studio Addin?

Add the `-DsettingsFile=[path_to_custom_settings.xml_file]` parameter when executing `mvn npanday.plugin:maven-vsinstaller-plugin:install`. For example,

```
mvn npanday.plugin:maven-vsinstaller-plugin:install -DsettingsFile="C:
\settings.xml"
```

2.1.7 How do I set the root namespace for a Visual Basic assembly?

Add the `<rootNameSpace>` element under `<configuration>` inside the `maven-compile-plugin` plugin. Just like the following:

```
      <plugin>
        <groupId>npanday.plugin</groupId>
          <artifactId>maven-compile-plugin</artifactId>
          <extensions>true</extensions>
          <configuration>
          <language>VB</language>
          <rootNameSpace>my.company</rootNameSpace>
        </configuration>
      </plugin>
```

2.1.8 Where can I get the NPanday VS Addin installers?

NPanday VS Addin installers can be downloaded from the NPanday site, under the **Releases** tab.

In this page, you can see different files to download such as:

- **VS Addin Installer -** installs the VS Addin
- **VS Addin + Repository Installer -** installs the VS Addin with the NPanday repository
- **Source -** NPanday source codes for those who want to contribute to the community
- **Repository -** NPanday repository containing the artifacts commonly used

3

3.1 Starting and Stopping NPanday .NET Build Tool

This section provides a quick-reference for when you need to start and stop the NPanday .NET Build Tool (for example, if you want to start the tool after you have already been inside Visual Studio working, or if you want to stop the tool but keep Visual Studio open).

3.1.1 Starting the Build Tool

While you can go from the instructions for opening and loading your Project/Solution to using the NPanday .NET Build Tool with Visual Studio, there are times when you may want to start the tool after you have already been inside Visual Studio working. Also, if you load a new version of the NPanday .NET Build Tool, you should clear Visual Studio's cache before you start the tool. The table, below, explains how to start the tool in each of these situations:

If you want to clear Visual Studio's cache of any previous versions of the tool before you start,

1. Use the following command to clear Visual Studio's cache of any previous versions of the tool:
   ```
   devenv /ResetAddin NPanday.VisualStudio.Addin
   ```
2. Open Visual Studio.
3. Click Tools > NPanday Build System to load the NPanday add-in.
4. Continue by opening your project and then loading your solution, as explained in Loading Project/Solution.

If you are starting the NPanday .NET Build Tool from inside Visual Studio:

1. Open Visual Studio.
2. Click Tools > NPanday Build System to start the Maven embedded server.
3. Continue by opening your project and then loading your solution, as explained in Loading Project/Solution.

3.1.2 Stopping the Build Tool

There are two ways to stop the NPanday .NET Build Tool. The first is to click Tools>Add-in Manager then de-select the checkbox for NPanday .NET Build Tool to stop the Maven embedded server. Another way is to simply exit Visual Studio, which shuts down the Tool and the embedded Maven server automatically.

Note: If an abnormal shutdown of Visual Studio occurs, you must reboot to ensure both the Build Tool and the embedded Maven server have stopped.

4

4.1 Project Dependencies

This section defines the basic configuration of adding references into the project. To avoid runtime errors, make sure that the library to be loaded does not have conflicting instance(s) and/or is not yet loaded in the .Net Framework.

Note: Adding dependencies should be done using Visual Studio while manually adding is not recommended. If you manually add a dependency and re-import the project in Visual Studio, the recently added dependencies are deleted from the pom.xml file.

To add a system scope dependency, add a configuration similar to the following in the project's pom.xml:

```
<dependencies>
  [...]
  <dependency>
    <groupId>artifact_group_id</groupId>
    <artifactId>my.lib</artifactId>
    <version>2.2</version>
    <type>library</type>
    <scope>system</scope>
    <systemPath>C:\path\to\your\library.dll</systemPath>
  </dependency>
  [...]
</dependencies>
```

Supply the proper values for the dependency's `<groupId>`, `<artifactId>`, `<version>`, `<type>`, `<scope>`, and `<systemPath>`. Exact match of `artifactId` and `version` are required when manually adding dependencies into the project's `pom.xml`.

Dependency Types:

- normal – Local repository content artifacts that are added as dependencies.
- `gac_msil` – Artifacts that are installed in the GAC and used as references. Configuration similar to the following is used, `<type>gac_msil</type>`
- `system` – Artifact references to standard framework namespaces. Configuration similar to the following is added, `<scope>system</scope>` and the `<systemPath>[path_to_library]</systemPath>`
- `com_references` – Artifact references to Component Object Model that is installed in the system. Configuration similar to the following is used, `<type>com_reference</type>`

5

5.1 Installing Files Manually

This section provides procedures on installing a file manually into the local repository. In this process, `maven-install-plugin` is used.

Use the following command

```
mvn npanday.plugin:maven-install-plugin:install-file -Dfile=[path_to_file]
-DgroupId=[group_id] -DartifactId=[artifact_id] -DartifactVersion=[version]
-Dpackaging=[packaging_type]
```

and supply appropriate values:

- **file**: the location of the file to be installed.
- **groupId**: the groupId of the file.
- **artifactId**: artifact name of the file.
- **artifactVersion**: the version of file.
- **packaging**: the type of file to be installed.

6

6.1 Loading Project/Solution

This section explains how to load and use NPanday.

1 If it is not running, start Visual Studio from your shortcut Menu or the Windows Start menu.

2 If the NPanday Build System has not been started, from inside Visual Studio, select Tools>NPanday Build System.

After you start the NPanday Build System and it is running, it no longer appears as an option on the Tools menu (and reappears again when Visual Studio is stopped and started). Therefore, if you had previously started the NPanday Build System and it is still running, you do not need to start it again in this step.

3 If the Project/Solution was recently opened you can select it from the Recent Projects pane by clicking on it.

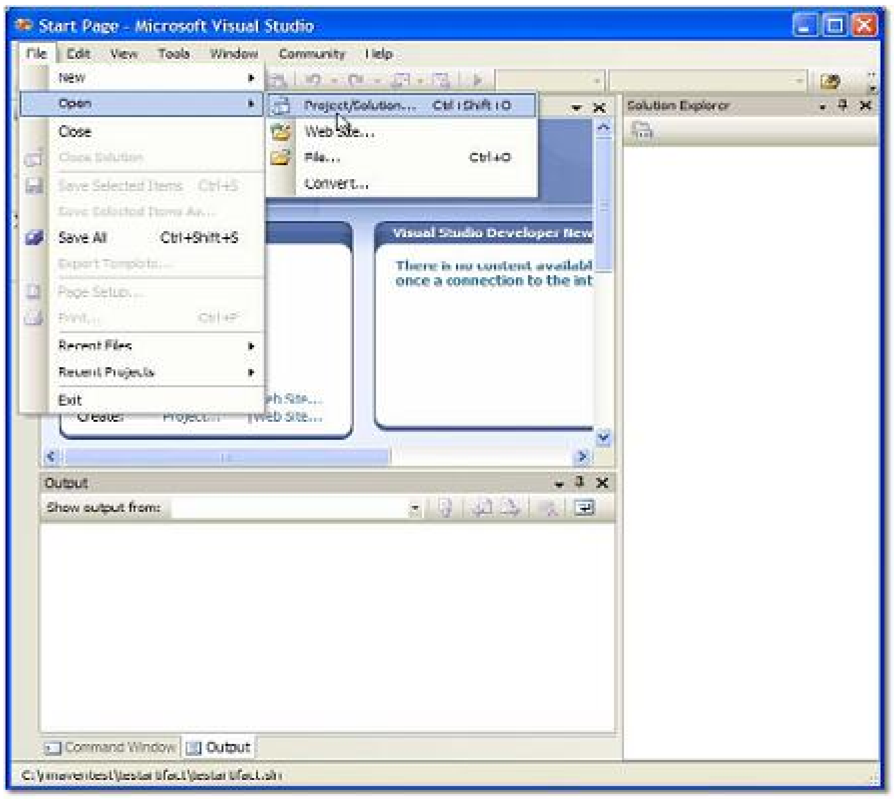
Or select File>Open>Project/Solution to open the Project/Solution. An Open Project window appears, which allows you to browse to find the specific Project/Solution you want to open.

For example, if you created the archetype C# or VB projects in the previous section of this document, from the Open Project window browse to the `NPanday.Test` directory (created previously) and locate the Project/Solution file (NPanday.Test.sln). Click to select the Project/Solution file, then click on Open to load the Project/Solution.

The `*.sln` file is located under the directory you created in the previous section of this document. In this sample procedure, you created a directory called npandaytest and the subsequent steps created the Project/Solution files under that directory. To find the `*.sln` file, you would browse to `C:\npandaytest\NPanday.Test` and find the file called `NPanday.Test.sln`.

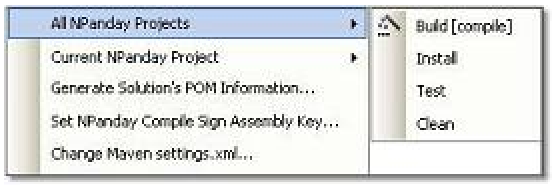
Upon opening your Project/Solution, you may see a window with the selection Load project normally. Select that option and continue. This window may appear twice, so select Load project normally both times. After you open the Project/Solution, the associated files and directories appear in the Solution Explorer window of Visual Studio.

4 In the Solution Explorer pane, right-click on the `NPanday.Test` folder.
.



Opening a Project Solution

5 In the Solution Explorer pane, right-click on the project and select All NPanday Projects or Current NPanday Project. Then, you may select the goal you want to perform, for example Install to install all the sub-projects in the parent project or the currently selected project respectively.


All NPanday Projects
Current NPanday Project
Generate Solution's POM Information...
Set NPanday Compile Sign Assembly Key...
Change Maven settings.xml...
Build [compile]
Install
Test
Clean

Maven goals menu

The goal will begin running, and status information appears in the Visual Studio Output pane. NPanday installs the artifacts in a .NET local repository, located in `C:\Documents and Settings\[user_home]\.m2\uac`.

For the NPanday.Test solution the artifact is placed here, `C:\Documents and Settings\[user_home]\.m2\uac\gac_msil\NPanday.Test\1.0-SNAPSHOT__NPanday\NPanday.Test.dll`

The contents in `C:\Documents and Settings\[user_home]\.m2\uac\gac_msil` SHOULD NOT be manually modified or the project build will fail due to artifacts that are not properly indexed and are not synchronized with the repository. In case the contents have been modified, delete the `C:\Documents and Settings\[user_home]\.m2\uac` directory then re-install the project.

If the build fails on nunit-console see the Pre-requisites section for information on installing NUnit 2.2+ and adding it to your PATH.

If you notice build errors in the Output pane, it may be because your PATH is not set correctly to find the compiler. The Pre-requisites section explains how to set the PATH correctly.

7

7.1 Making VS Project Files Portable

To be able to make VS project files portable, NPanday 1.1 provides the Resync References functionality. This will synchronize the references so that the project will still run in other user's machine. Added references will be stored in hidden folder named ".references". When the project is used in other user's machine, the Resync References functionality will download and store the references in the ".references" folder.

To use Resync References button, right click on the project and go to 'All NPanday Projects' or 'Current NPanday Project' in the context menu and select 'Resync References'.

Intra-project references are skipped on Resync. Since this behavior was just applied recently in Version 1.0.2, old POMs with intra-project references might generate an error during Resync or Import. To fix this, remove the reference and add it back again. Or simply delete the POM and re-import the project.

8

8.1 Quick Start Guide

This section is provides a quick way to install the NPanday .NET Build Tool and create a sample project. This section does not explain any of the steps - explanations can be found in NPanday .NET Built Tool Add-in Installation and subsequent sections.

Note: If you are upgrading to a new version of the add-in, we recommend you remove the old installation and start with a “clean” system. Follow the Uninstalling the NPanday .Net Build Tool procedure before installing the new version.

1. Make sure your `C:\Documents and Settings\[user_home]\.m2\settings.xml` file points to the repository that contains the .NET artifacts, where `[user_home]` is the home directory of the user performing these steps.
2. Set your `PATH` environment variable to find `csc` and `devenv` (the Visual Studio executables). Typically the `PATH` will be this:

   ```
   PATH=%PATH%;C:\Program Files\Microsoft Visual Studio 8\SDK\V2.0\Bin;C:
   \WINDOWS\Microsoft.NET\Framework\v2.0.50727; C:\Program Files\Microsoft
   Visual Studio 8\Common7\IDE;C:\Program Files\NUnit 2.4.3\bin
   ```

   If you are starting Visual Studio from the Windows Start menu, then you need to set the `PATH` environment variable in the System Properties:

   Start > My Computer > View System Information > Advanced > Environment variables

   If you are starting Visual Studio from a command shell using the `devenv` executable then you can set your `PATH` on the command line as shown above.
3. Create the Visual Studio add-in, by running the `maven-vsinstaller-plugin` to resolve (and download if needed) all of NPanday's dependencies. Execute the following command from a command shell from any directory:

   ```
   mvn npanday.plugin:maven-vsinstaller-plugin:install
   ```

   The add-in file will be created and put in the user's home directory, here:

   ```
   C:\Documents and Settings\[user_home]\My Documents\Visual Studio
   2005\Addins\NPanday.VisualStudio.AddIn
   ```
4. Start Visual Studio.
5. In Visual Studio start the NPanday Build System from Tools>NPanday Build System. You will see the NPanday embedded server starting in the Visual Studio Output pane.
6. To verify the add-in is working, create a simple project. Create an empty directory called, for example, `npandaytest`.
7. Go to the directory (`npandaytest` in our example) and execute:

   ```
   mvn archetype:create -DgroupId=npanday -DartifactId=NPanday.Test
   -DarchetypeArtifactId=maven-archetype-dotnet-simple -
   DarchetypeGroupId=org.apache.maven.dotnet -DarchetypeVersion=[version]
   ```
8. Go to the `NPanday.Test` directory (which was created in the previous step) and execute:

   ```
   mvn install
   ```

   ```
   mvn clean
   ```
9. Generate a solution file (also from the NPanday.Test directory):

   ```
   mvn NPanday.Plugins:NPanday.Plugin.Solution.JavaBinding:Solution
   ```
10. In Visual Studio, click File>Open>Project/Solution. Browse to the `NPanday.Test.sln` file (under the NPanday.Test directory created in step 4) and select it. Click Open.

11 In the Solution Explorer pane in Visual Studio, right-click the `NPanday.Test folder`. If you get the Security Warning pop-up screen select Load project normally and click OK. You may see this screen twice, so select Load project normally again and click OK the second time as well.

12 Select All NPanday Projects > Install. You should see the build running in the Output pane. The artifact, `NPanday.Test.dll`, will be placed here:

```
C:\Documents and Settings\[user_home]\.m2\uac\gac_msil
\NPanday.Test[version]__NPanday
```

9

9.1 Creating a Simple Project

Before you start this procedure, you must have all Pre-requisites in place and have successfully completed all steps in the previous section entitled Installing and Verifying NPanday .NET Build Tool.

1 Create an empty directory for your project. For example:

```
mkdir npandaytest
cd npandaytest
```

2 From a command shell go to the directory that contains the POM file, and execute the following command (In the example, execute the command from the `npandaytest` directory) to create a C# or VB project:

For C#: `mvn archetype:create -DgroupId=npanday -DartifactId=NPanday.Test -DarchetypeArtifactId=maven-archetype-dotnet-simple -DarchetypeGroupId=org.apache.maven.dotnet -DarchetypeVersion=[version]`

For VB: `mvn archetype:create -DgroupId=npanday -DartifactId=NPanday.Test -DarchetypeArtifactId=maven-archetype-vb-simple -DarchetypeGroupId=org.apache.maven.dotnet -DarchetypeVersion=[version]`

The command creates the project in the NPanday.Test directory, which will now contain the following for a C# project (and a VB project will be similar with a vb directory instead of the csharp directory shown):

```
.
|-- src
|  `-- main
|    `-- csharp
|      `-- Sample
|        `-- MyApp.cs
|    `-- resources
|  `-- test
|    `-- csharp
`- pom.xml
```

3 From inside the Solution directory (`NPanday.Test`), execute the following command to build and install the NPanday.Test DLL files into your repository:

```
mvn install
```

The `mvn install` command typically puts artifacts (installs them) into the repository here, `C:\Documents and Settings\[user_home]\.m2\repository`.

NPanday installs the artifacts in a .NET local repository, located in `C:\Documents and Settings\[user_home]\.m2\uac`.

For the `NPanday.Test` solution the artifact is placed here, `C:\Documents and Settings\[user_home]\.m2\uac\gac_msil\NPanday.Test\1.0-SNAPSHOT__NPanday\NPanday.Test.dll`

The contents in `C:\Documents and Settings\[user_home]\.m2\uac\gac_msil` SHOULD NOT be manually modified or the project build will fail due to artifacts that are not properly indexed and are not synchronized with the repository. In case the contents have been modified, delete the `C:\Documents and Settings\[user_home]\.m2\uac` directory then re-install the project.

If the build fails on nunit-console see the Pre-requisites section for information on installing NUnit 2.2+ and adding it to your PATH.

4 Then clean up the target directory and download dependencies by executing:

```
mvn clean
```

5 In a command shell, still from the Solution directory (`NPanday.Test` in this example), execute the following command to generate the Solution:

```
mvn npanday.plugin:NPanday.Plugin.Solution.JavaBinding:Solution
```

6 This command creates the Solution file (`*.sln`) that corresponds to the Solution directory you just created. For example, the Solution directory was `NPanday.Test`, so the Solution file is `NPanday.Test.sln`. After you execute the command, verify the Solution directory now contains the Solution file. If so, then you are ready to load the project and solution.

10

10.1 Managing Repository Artifacts

This feature allows the Visual Studio user to browse managed repositories and add and delete Maven dependencies to and from the repositories from within the Visual Studio IDE.

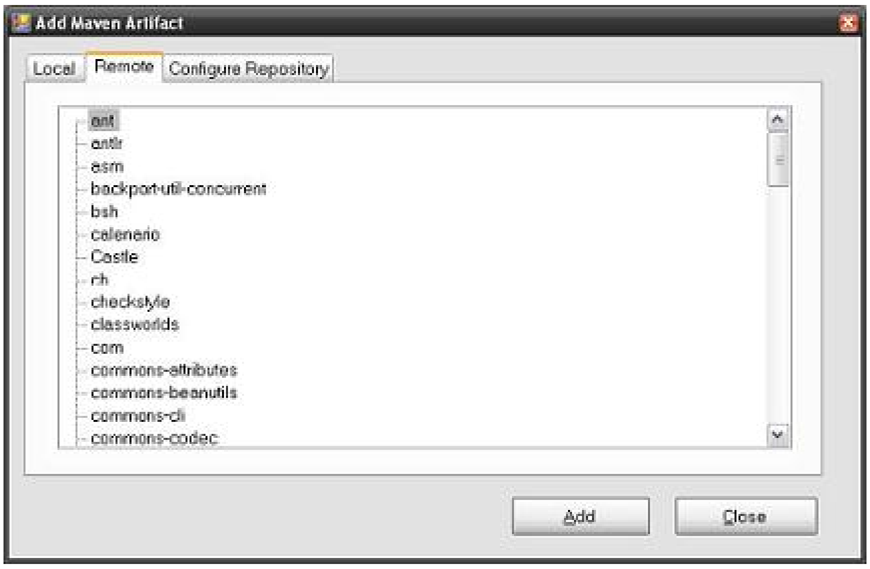
A remote repository must be configured before attempting to use this feature. Please see the Remote Repository section for instructions.

10.1.1 Adding Artifacts

When adding an artifact dependency into a web site project, there will be no changes in the `pom.xml`. This is because the artifacts are put directly into the project's `bin\` directory (one is created when this is missing), thus, `<dependencies>` in the `pom.xml` is not needed.

To add an artifact:

1 Right-click on a project and select Add Maven Artifact... from the menu.
2 In the Add Maven Artifact pop-up window there are tabs for each repository you have configured. Click the tab for the repository in which the desired artifact is located.
3 Scroll to select the desired artifact (must be a `DLL` file, not a `JAR` nor an `OCX` file).
.



Add Maven Artifact screen

4 Click Add to add the selected artifact and exit the form.

Or, double click on the artifact to be added. Repeat this step to add multiple artifacts. Then, click Close to exit the form.

The artifacts will be added as a reference to the project and added as a dependency in the POM file. It is also downloaded to the `C:\Documents and Settings\[user_home].\m2\uac` directory.

When adding an `.ocx` file into the repository, a warning message similar to the following will be displayed.
.

COM reference error message

10.1.2 Removing Artifacts

The Remove Artifact feature removes the artifact as a reference to the project and as a dependency in the POM file.

To remove an artifact, you can select the artifacts to remove and click the Delete key. This works for both Visual Basic and C# projects.

Alternatively, to remove Visual Basic project artifact:

1 Right-click on a project and select Properties from the menu.
2 Click on the References tab, and select the desired artifact.
3 Click Remove.

Alternatively, to remove C# project artifact:

1 Inside the project, browse through the References folder.
2 Right click on the desired artifact and select Remove.

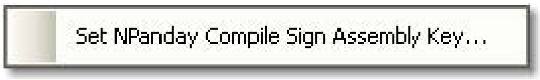
11

11.1 Setting Assembly Keys

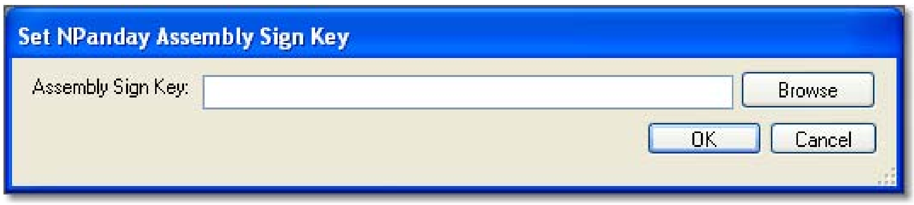
Assembly keys are used to uniquely identify the project to be installed in the Global Assembly Cache or also referred to as GAC.

To set an assembly key to a project:

1. Generate a strong name key by running the command `sn -k [filename] .snk`
2. Right click on the project and select Set NPanday Compile Sign Assembly Key...
   .



3. Put the strong name key that was generated in the Assembly Sign Key field.
   .



Setting assembly sign key

4. Click OK.

12

12.1 Command Line Release

The `<scm>` tag should be configured in the project's `pom.xml`. It should look similar to the following

```
<scm>
  <connection>scm:svn:[project_url]</connection>
  <developerConnection>scm:svn:[project_url]</developerConnection>
  <url>[URL of the project]</url>
</scm>
```

1. Execute the following command from the project directory.

   `mvn clean install or mvn install`
2. Before starting the release, invoke the release plugin by executing the following command from the project root directory of a single-module project, or from the module directory (that uses references) of a multi-module project.

   `mvn npanday.plugin:NPanday.Plugin.SysRef.JavaBinding:prepare`
3. Start releasing the project. The following commands prepares and releases the project respectively.

   `mvn release:prepare`

   `mvn release:perform`

   For Maven guide, you can refer to Better Builds with Maven.

13

13.1 Build Management Release

The <scm> tag should be configured in the project's pom.xml. It should look similar to the following

```
<scm>
  <connection>scm:svn:[project_url]</connection>
  <developerConnection>scm:svn:[project_url]</developerConnection>
  <url>[URL of the project]</url>
</scm>
```

1 Add the project in Apache Continuum -- Build Management.
2 Add the following build definition on the project module containing COM references by clicking the Add button under Build Definitions.

Field	**Value**
POM filename	`pom.xml`
Goals	`npanday.plugin:NPanday.Plugin.SysRef.JavaBin`
Arguments	`--batch-mode –non-recursive`
Schedule, Build Environment, Type	default values
Description	optional

3 Build the project by clicking the build icon of the clean install goal. Make sure this goal exists, if not, add the clean install goal.
4 Execute the `npanday.plugin:NPanday.Plugin.SysRef.JavaBinding:prepare goal` by clicking the build icon found to the right of the goal.
5 Start releasing the project.

For instructions on releasing projects using Apache Continuum, you can refer to Apache Continuum release documentation.

14

14.1 Executing Maven Goals

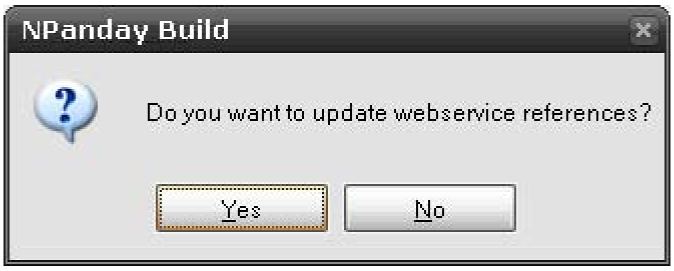
14.1.1 Compile

To use the NPanday .NET Build Tool to compile a project and produce a .NET-compliant output:

1. In the Solution Explorer, right-click on the name of the project you want to compile, then select `Current NPanday Project > Build` to build the selected project. Or, `All NPanday Projects > Build` to build the parent and its sub-projects.

   For projects with dependencies from remote repositories, the build might fail. As a workaround, re-import the project then, execute the build goal (`Current NPanday Project > Build` or `All NPanday Projects > Build`).

   For projects with web references, a dialog box is prompted for updating the Web Services Description Language (`wsdl`). Click Yes to continue with the update, or, No to skip updating.

   Do you want to update webservice references?

2. The NPanday .NET Build Tool performs the compile on the project and sends corresponding information to the Output window (including a message saying the build was successful). You can scroll up and down in the output window to display all the text.

   You can also execute the normal Visual Studio build on your projects. To invoke Maven to perform `build`/ `install`/ `test`/ `clean`, you must select the option from the All NPanday Projects or Current NPanday Project sub-menu.

   When compiling a web application project, the NPanday .NET Build Tool performs an extra step by calling the `Aspnet_compiler` to validate the ASP section of the project. The Build Tool performs this step right after calling the CSharp compiler and Visual Basic compiler.

14.1.2 Clean

To use the NPanday .NET Build Tool to clean a project (removes the `target\` directory containing the files that were generated at build-time from the project's working directory):

1. From the Solution Explorer, right-click on the name of the project you want to clean, then select Current NPanday Project > Clean to clean the selected project. Or, All NPanday Projects > Clean to perform the clean goal to the parent project and its sub-projects.
2. The NPanday .NET Build Tool performs the clean on the project and sends corresponding information to the Output window (including a message saying the build was successful). You can scroll up and down in the output window to display all the text.

14.1.3 Test

To use the NPanday .NET Build Tool to test a Project:

1. From the Solution Explorer, right-click on the name of the project you want to test, then select `Current NPanday Project > Test` to perform the test to the selected project. Or, `All NPanday Projects > Test` to perform the test to the parent project and its sub-projects.
2. The NPanday .NET Build Tool performs the tests on the project and sends corresponding information to the Output window (including a message saying the build was successful). You can scroll up and down in the output window to display all the text.

Note: When the test goal output result is not refreshed for C# test project, the project should be reimported, and the value of the test should also be changed. Then, perform `Current Project: Test` (or, `All NPanday Projects > Test`).

14.1.4 Install

To use the NPanday .NET Build Tool to perform an install on a Project:

1. From the Solution Explorer, right-click on the name of the project you want to build, then select Current NPanday Project > Install to install the selected project. Or, All NPanday Projects > Install to install the parent project and its sub-projects.

   The NPanday .NET Build Tool installs the artifacts into your local repository. For web projects with pom as packaging, the `target` directory is not created and only the parent `pom.xml` is put in the local repository which is the default behavior.

   For web application projects, the project zip file is produced during this phase, specifically during the `package` phase.

   Corresponding information is sent to the Output window (including a message saying the build was successful). You can scroll up and down in the output window to display all the text.
2. Verify the artifacts were placed into the .NET local repository under the following path:

   ```
   C:\Documents and Settings\[user_home]\.m2\uac
   \[gac_architecture]\artifactId\Version__GroupId
   ```

   For example:

   ```
   C:\Documents and Settings\[user_home]\.m2\uac\gac_msil\NPanday.Test
   \1.0__NPanday
   ```

   The following explains what each element in the path means:

   - `[user_home]` The user's home directory.
   - `gac_architecture` The architecture type such as `gac_msil`, `gac_32`, etc.
   - `artifactId` This is similar to the Maven `artifactId`. It is equivalent to the project's module name. In the example above, the artifactId is `NPanday.Test`.
   - `Version__GroupId` The version of the artifact and the group it is in. In the example above the version is `1.0` and the GroupId is `NPanday`.

   The contents in C:\Documents and Settings\[user_home]\.m2\uac\gac_msil SHOULD NOT be manually modified or the project build will fail due to artifacts that are not properly indexed and are not synchronized with the repository. In case the contents have been modified, delete the C:\Documents and Settings\[user_home]\.m2\uac directory then re-install the project.
3. Verify that the following are also created under the following paths:

   - `C:\Documents and Settings\[user_home]\.m2\pab`
   - `C:\Documents and Settings\[user_home]\.m2\npanday-settings.xml`

   Contents of npanday-settings.xml looks similar to the following:

```
<?xml version="1.0" encoding="utf-8"?>
<npandaySettings xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:x
  <operatingSystem>
    Microsoft Windows NT 5.1.2600 Service Pack 2
  </operatingSystem>
  <defaultSetup>
    <vendorName>MICROSOFT</vendorName>
    <vendorVersion>2.0.50727</vendorVersion>
    <frameworkVersion>2.0.50727</frameworkVersion>
  </defaultSetup>
  <vendors>
    <vendor>
      <vendorName>MICROSOFT</vendorName>
      <vendorVersion>1.1.4322</vendorVersion>
      <frameworks>
        <framework>
        <frameworkVersion>1.1.4322</frameworkVersion>
          <installRoot>
            C:\WINDOWS\Microsoft.NET\Framework\v1.1.4322
          </installRoot>
        </framework>
      </frameworks>
    </vendor>
    <vendor>
      <vendorName>MICROSOFT</vendorName>
      <vendorVersion>2.0.50727</vendorVersion>
      <frameworks>
        <framework>
        <frameworkVersion>2.0.50727</frameworkVersion>
          <installRoot>
            C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727
          </installRoot>
          <sdkInstallRoot>
            C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\
          </sdkInstallRoot>
        </framework>
      </frameworks>
    </vendor>
  </vendors>
</npandaySettings>
```

14.1.5 NPanday's Maven Phase Algorithm

Algorithm that differs the Visual Studio Execution from Console Command.

1 All Project - Is trigerred when the user right click in the solution explorer and select `All NPanday Projects > Build` or `All NPanday Projects > Clean` or `All NPanday Projects > Test` or All NPanday Projects > Install>>>

  A Save All Documents that are open in Visual Studio.

  B Retrieve the solution that is open in Visual Studio.

  C Loop through all the projects in the solution.

  D Check if it has a Web Reference, if it has it will update the Web Reference.

E Execute the pom.xml of the solution may it be Build/Clean/Test/Install.

2 Current Project - Is Triggered when the user right click in the solution explorer and select `Current NPanday Project > Build` or `Current NPanday Project > Clean` or `Current NPanday Project > Test` or `Current NPanday Project > Install`

A Save All Documents that are open in Visual Studio.

B Retrieve CURRENT pom.xml and CURRENT project.

a If pom.xml does not exist, set errorMsg to "Pom Not Found" Error.

b If the packaging in the pom.xml is "pom", set errorMsg to "Pom may not be Project's Pom" Error.

C Check if it has a Web Reference, if it has it will update the Web Reference.

D Execute the pom.xml of the solution may it be Build/Clean/Test/Install.

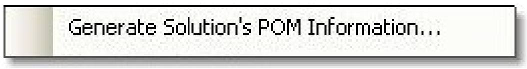
15

15.1 Importing Projects

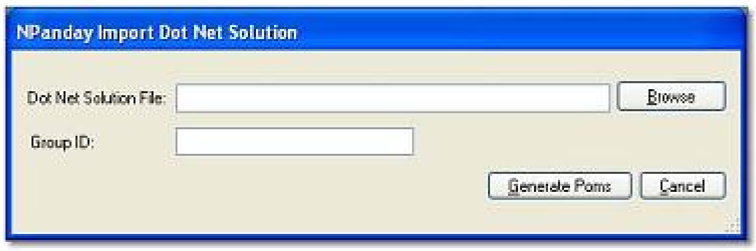
This section provides procedures on importing a Visual Studio project using Generate Solution's POM Information... feature. This feature will generate `pom.xml` files from an existing Visual Studio solution file. The generated `pom.xml` file will contain the latest versions of the artifact dependencies used for the project.

Note: ASP.NET MVC should be installed to get the proper behavior when importing projects especially those projects with unsupported project types. (Refer to Pre-requisites section.)

1 Right click on the project and select the Generate Solution's POM Information...
.



2 In the pop-up screen, Browse to the directory location of the parent solution file to use. Group Id is generated by default with the format of `[company_id].[solution_name]`.
.



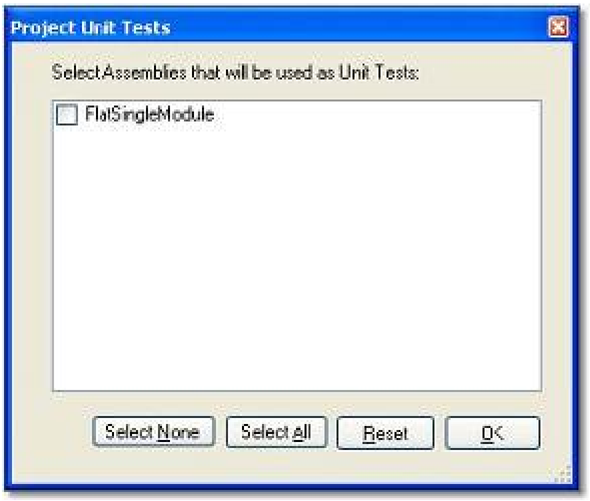
Importing parent solution file

3 Click Generate POMs.

Underscores and spaces in artifact's groupId are automatically deleted when added to the project's `pom.xml`. The table below shows the conversion of certain values of groupIds.

Artifact Group Id	Converted Group Id
The.Project	The.Project
the.project	the.project
the project	TheProject
the_project	TheProject

4 From the Project Units Tests window, select the test modules to be used for the project and click OK.
.



Project unit tests list

`pom.xml` files will be generated for all the projects that are included in the solution file: the parent project and its sub-projects, if there are any.

Note: The Units Tests window is not displayed when importing unsupported project types. Hence, an unsupported project type warning is displayed. For multi-module projects with supported and/or unsupported project types, the supported projects are included in the parent pom while for the unsupported project types, these are not included in the parent pom, and a warning is displayed.

For projects with web references, the `pom.xml` generated will contain `maven-wsdl-plugin` configuration similar to the following:

```
<plugin>
  <groupId>npanday.plugin</groupId>
  <artifactId>maven-wsdl-plugin</artifactId>
  <extensions>true</extensions>
  <executions>
    <execution>
      <goals>
        <goal>wsdl</goal>
      </goals>
    </execution>
  </executions>
  <configuration>
    <webreferences>
      <webreference>
        <namespace>AnotherWebService</namespace>
        <path>Web References/AnotherWebService/AnotherWS.wsdl</path>
        <output>Web References/AnotherWebService/</output>
      </webreference>
    </webreferences>
  </configuration>
</plugin>
```

The wsdl goal is the one responsible for building and creating webservice proxy classes. And, all the web references used in the project are listed inside the `<webreferences>` tag. For more instructions on adding, renaming, and deleting web references, refer to Web References section of this document.

15.1.1 Importing Projects with Resource File

If the project you are importing includes a resource file, the `maven-resgen` plugin is added to `pom.xml` during import.

In the plugin's configuration, you can find the list of resource (`resx`) files with corresponding names. By default, `RootNamespace` is the assembly name and `RootNamespace.Filename` is the filename of the `resx` files. To ensure that the plugin works, the following are required:

- The classname and the filename of the item that contains the resx files must have the same name. Example:

  `"public class frmMaestroName" = frmMaestroName.resx = frmMaestroName.cs`

  Note: Changing the codefile(.cs/.vb) automatically updates the resource(resx) file. In some instances, however, this does not hold true.
- The namespace of the project should be equal to the DefaultNamespace of the project assembly. To check the default name,
  - Right-click on the project and select Properties. This displays a project property window.
  - Select the Application tab.
  - In the DefaultNamespace field, you will find `RootNamespace`.

Once the above requirements are met, the importer automatically does the work and there is no need to edit `pom.xml`.

Removing the configuration part of the `maven-resgen` plugin allows the plugin to locate all the resx files and compile them as `ArtifactId.Filename`. If you set the namespace as the artifactId, you do not need to list all the resx files in the `pom.xml`.

Note: This feature does not support cultured esx files such as `eu-US`, `it`, `jp`, `de`, `eu-UK`, and even custom culture.

15.1.2 Importing WPF/WCF Projects with XAML files

If the project you are Importing is a WPF/WCF project and it contains a XAML file the MSBuildPlugin will be automatically added into that project's pom file.

```
<plugin>
      <groupId>npanday.plugin</groupId>
      <artifactId>NPanday.Plugin.Msbuild.JavaBinding</artifactId>
      <extensions>true</extensions>
      <executions>
        <execution>
          <goals>
            <goal>compile</goal>
          </goals>
        </execution>
      </executions>
</plugin>
```

This will generate the needed *.g.cs/*.g.vb files that are needed for those XAML files.

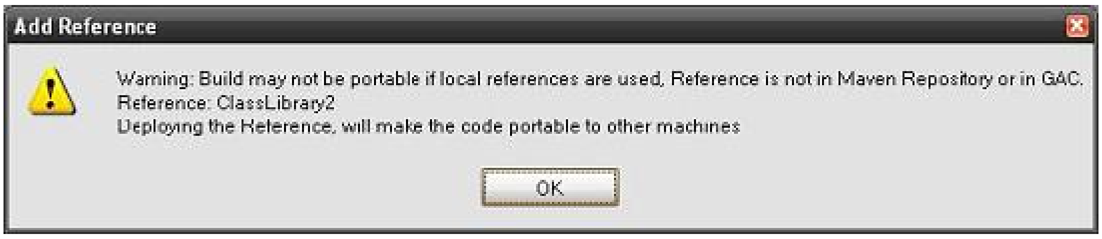
Note: The user would encounter double dependency errors, if the project was created with 3.0 .net framework. We suggest that you change your framework to 3.5 and then delete the references that are already included as defaults in NPanday. This will then allow your project to run successfully.

16

16.1 Project Dependencies

This section provides instructions on adding system installed artifacts as references to the project.

When adding artifact reference that is not installed in the system, NPanday will prompt a warning message then adds the reference in the `pom.xml` as a system scope dependency.



Warning: Build may not be portable if local references are used, Reference is not in Maven Repository or in GAC.
Reference: ClassLibrary2
Deploying the Reference, will make the code portable to other machines

Reference not installed in the system

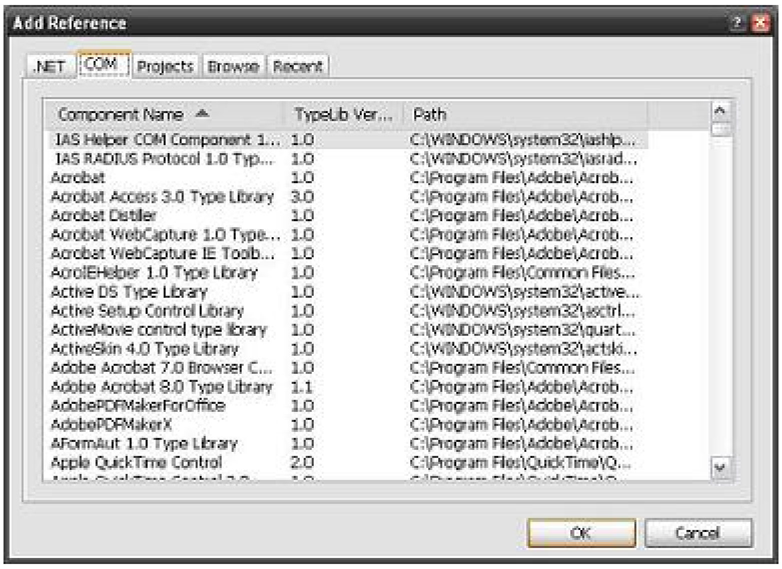
However, the above warning message is not displayed for web site projects even if the artifact reference to be added is not installed in the system. This is because the artifact references for web projects are directly added in the project's `bin\` directory (one is created when this is missing), thus, no need to add it in the `pom.xml`.

The following POM snippet shows a system dependency:

```
<dependency>
  <groupId>artifact_group_id</groupId>
  <artifactId>my.lib</artifactId>
  <version>2.2</version>
  <type>library</type>
  <scope>system</scope>
  <systemPath>C:\path\to\your\library.dll</systemPath>
</dependency>
```

To add references to the project:

1 Right click on the project that you want to add a reference to.
2 Select Add Reference... from the menu list.
3 Select the tab to which the reference belongs (**.Net**, **COM**, **Projects**).
  Or, Browse to look for the specific artifact to be reference and Recent for artifacts recently accessed.
4 Select the references to be added from the list as shown in the figure above. Scroll to select the desired object reference.
5 Click OK.



Component Name	TypeLib Ver...	Path
IAS Helper COM Component 1...	1.0	C:\WINDOWS\system32\iashlp...
IAS RADIUS Protocol 1.0 Typ...	1.0	C:\WINDOWS\system32\iasrad...
Acrobat	1.0	C:\Program Files\Adobe\Acrob...
Acrobat Access 3.0 Type Library	3.0	C:\Program Files\Adobe\Acrob...
Acrobat Distiller	1.0	C:\Program Files\Adobe\Acrob...
Acrobat WebCapture 1.0 Type...	1.0	C:\Program Files\Adobe\Acrob...
Acrobat WebCapture IE Toolb...	1.0	C:\Program Files\Adobe\Acrob...
AcroIEHelper 1.0 Type Library	1.0	C:\Program Files\Common Files...
Active DS Type Library	1.0	C:\WINDOWS\system32\active...
Active Setup Control Library	1.0	C:\WINDOWS\system32\asctrl...
ActiveMovie control type library	1.0	C:\WINDOWS\system32\quart...
ActiveSkin 4.0 Type Library	1.0	C:\WINDOWS\system32\actski...
Adobe Acrobat 7.0 Browser C...	1.0	C:\Program Files\Common Files...
Adobe Acrobat 8.0 Type Library	1.1	C:\Program Files\Adobe\Acrob...
AdobePDFMakerForOffice	1.0	C:\Program Files\Adobe\Acrob...
AdobePDFMakerX	1.0	C:\Program Files\Adobe\Acrob...
AFormAut 1.0 Type Library	1.0	C:\Program Files\Adobe\Acrob...
Apple QuickTime Control	2.0	C:\Program Files\QuickTime\Q...

Sample display of list of references

When adding a COM reference, the actual reference (DLL) is copied to `C:\WINDOWS\assembly\GAC_MSIL`. Wherein, the path to the reference found in this directory is used as the reference's `<systemPath>` in the `pom.xml`.

17

17.1 Releasing .NET Projects

This section provides details in preparing .Net projects with references (DLLs) for release. The procedures for command line execution and using Apache Continuum for build management in preparing the project are provided.

When invoking the release plugin using the command

```
mvn npanday.plugin:NPanday.Plugin.SysRef.JavaBinding:prepare
```

The reference (DLL) will be searched from `C:\WINDOWS\assembly\GAC_MSIL` directory and will be put in `C:\WINDOWS\Temp\NPanday` then renamed following the artifact filename format `[artifactId]-[version].[packaging]`.

Note: The above command should be invoked especially when releasing projects will references to `gac`.

After the reference is renamed, it will then be installed in the local repository `C:\Documents and Settings\[user_home]\.m2\repository` for it to be used when releasing the project.

The following are examples of references of different types which are renamed then installed in the local repository,

- **gac_msil** file: `Microsoft.JScript-8.0.0.0-b03f5f7f11d50a3a.gac_msil`
- **com_reference** file: `Acrobat-1.1.0.0-E64169B3-3592-47D2-816E-602C5C13F328-1.1-0.com_reference`

18

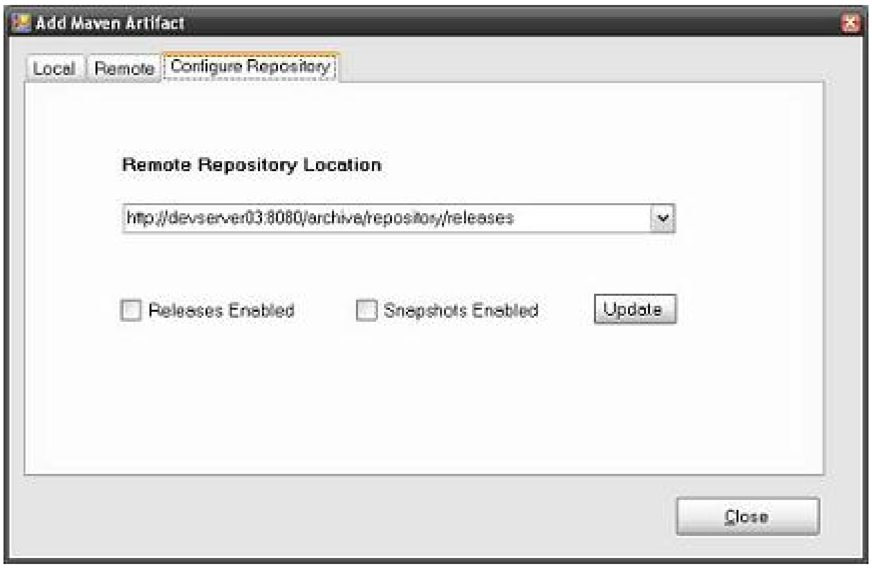
18.1 Remote Repository Configuration

A remote Maven repository can be configured using the Visual Studio IDE or it can be configured manually by adding information to the settings file.

18.1.1 Automatically Configure a Remote Maven Repository via the IDE

To access a remote Maven repository you can configure it from within Visual Studio. Visual Studio must be open, the NPanday Build System must be running, and you must have a project loaded. Then:

1. Right-click on a project and select Add Maven Artifact... from the menu.
2. In the Add Maven Artifact pop-up window, click the Configure Repository tab.
3. Select the URL of the repository in the pop-up window.
   .



http://devserver03:8080/archiva/repository/releases

Sample configuration for Remote Repository

4. If the remote repository allows snapshots or released artifacts to be stored there, then check the appropriate box.
5. Click Update. This will save the configuration to `C:\Documents and Settings\[user_home]\.m2\settings.xml`. To edit multiple repositories, repeat steps 3 to 5 by selecting another repository to be configured form the drop down list.
6. Click Close when done configuring the repository.

   NOTE: The repository is stored in a profile NPanday.id this profile is then added into the activeProfiles list as soon as an NPanday remote repository is added.

18.1.2 Manually Configure a Remote Maven Repository

To manually add a remote repository, add the following lines in your `C:\Documents and Settings\[user_home]\.m2\settings.xml` file within the `<profiles>` `</profiles>` tag. Modify the values for the `<repository>` and `<id>` elements with the repository url you want to access.

```
<profile>
      <activation>
        <activeByDefault>true</activeByDefault>
      </activation>
      <repositories>
        <repository>
          <releases>
            <enabled>false</enabled>
          </releases>
          <snapshots>
            <enabled>false</enabled>
          </snapshots>
          <id>http://repo1.maven.org/maven2/</id>
          <url>http://repo1.maven.org/maven2/</url>
        </repository>
      </repositories>
      <id>NPanday.id</id>
</profile>
```

If the remote repository allows the snapshots or release artifacts to be stored then add the following lines:

```
<profile>
      <activation>
        <activeByDefault>true</activeByDefault>
      </activation>
      <repositories>
        <repository>
          <releases/>
          <snapshots/>
          <id>http://repo1.maven.org/maven2/</id>
          <url>http://repo1.maven.org/maven2/</url>
        </repository>
      </repositories>
      <id>NPanday.id</id>
</profile>
```

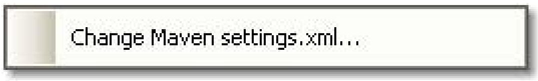
19

19.1 Using Custom Settings

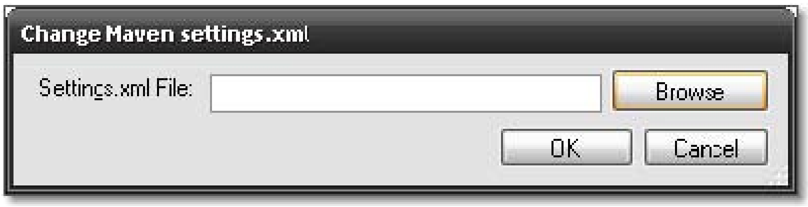
This feature allows the Visual Studio user to customize the location of the `settings.xml` file that points to the repository of NPanday artifacts.

To change the location:

1 Right click on the project and select the Change Maven settings.xml...
.



2 In the pop-up screen, click Browse to browse through the directory location of the `settings.xml` to use.
.



Configure settings.xml location

3 Click OK.

20

20.1 Integration Test

Integration test is run on projects that contains the plugin `maven-test-plugin` under the POM build configuration.

Since integration test itself is for testing, the `<testSourceDirectory>` is no longer needed and the `<sourceDirectory>` is enough to be able to run the project test successfully. The following POM snippet is an example build configuration of an integration test project.

```
<build>
  <sourceDirectory>./</sourceDirectory>
  <plugins>
    [...]
    <plugin>
    <groupId>npanday.plugin</groupId>
    <artifactId>maven-test-plugin</artifactId>
    <extensions>true</extensions>
    <configuration>
      <integrationTest>true</integrationTest>
    </configuration>
    </plugin>
  </plugins>
</build>
```

To run the test, simply import the project and execute the NPanday's test goal. You can refer to Importing Projects and Test sections for instructions on importing and executing test goal respectively.

21

21.1 Web References

Web references are artifacts that refers to the web service. These are usually in web format.

The following procedures in adding, renaming, and deleting web references assumes that the project in which the web references will be added, renamed, or deleted has an existing `pom.xml` file. Executing the following procedures will also make changes to the `pom.xml` file of the project.

21.1.1 Adding Web References

1. Right click on the project that you want to add a web reference to.
2. Select Add Web Reference... from the menu list.
   .

3. Select the web service you want to add, or you can choose to type the URL.
4. Supply a Name for the reference. This will be used to call the reference later in your code.

21.1.2 Renaming Web References

1. Right click on the web reference of the project application that you want to rename. Or, right click on the folder containing the wsdl file of the web service of the web project that you want to rename.
2. Select Rename.
3. Supply the new Name for the reference. This will be used to call the reference later in your code.

21.1.3 Removing Web References

1. Right click on the web reference of the project application that you want to delete. Or, right click on the folder where the wsdl file of the web service of the web project that you want to delete.
2. Select Delete.

22

22.1 Welcome to NPanday

NPanday provides Maven 2.x plugins to support building .NET applications at the command line or in your IDE. It enables you to use Visual Studio with a Maven-compatible development infrastructure.

NPanday is its own framework, the goal of which is to provide a common Maven build environment across .NET compilers (C#, VB), vendors (Microsoft, Novell, DotGNU), and platforms (Windows, Linux). This framework leverages Maven for the build lifecycle, making it a small compiler-plugin-framework on top of the larger Maven build-framework.

NPanday Maven Integration Features support console, window and web based applications as listed below.

C# & VB Project supported project types

- Windows Application
- Class Library
- Console Application
- Device Application
- Crystal Reports Application
- ASP .NET Web Application
- ASP .Net Web Service Application
- Windows Presentation Foundation (WPF)
- Windows Communication Foundation (WCF)

Web Site supported project types

- ASP.Net Web Site
- ASP .NET Web Service

To be more specific, NPanday supported project types are: CSharp (library, executable), VbDotNet (library, executable), Web_Site, and Web_Application, Windows_Presentation_Foundation__WPF, Windows_Communication_Foundation__WCF.

And the combination of types supported are: Web_Site + CSharp, Web_Site + VbDotNet, Web_Application + CSharp, and Web_Application + VbDotNet, Windows_Presentation_Foundation__WPF + CSharp, Windows_Presentation_Foundation__WPF + VbDotNet, Windows_Communication_Foundation__WCF + CSharp, Windows_Communication_Foundation__WCF + VbDotNet.

NPanday supported project structures are as follows:

- Flat Single Module Project
- Normal Single Project
- Flat Multi Module Project
- Normal Multi Module Project

Click here for more NPanday Information

22.2 Reporting Bugs/Requesting Features

- NPanday Issue Tracking

- Post to Mailing List

22.3 PDF

This documentation is also available in a single pdf.

23

23.1 NPanday .NET Build Tool Installation and Configuration

23.1.1 Pre-requisites

- Remove previous installation: We recommend starting with a “clean” system, so follow the procedure for Uninstalling the NPanday .Net Build Tool before you install the new version.
- Java Runtime Environment (JRE) 1.5 or greater: Must be installed. The `JAVA_HOME` environment variable must be set to the directory where the JRE is installed, for example, `C:\Program Files\Java\jre1.5.0_06.`. The Java executable (`%JAVA_HOME%\bin`) must also be added to `PATH`.
- Visual Studio 2005 or Visual Studio 2008: Must be installed. However, it must not be running when you install the NPanday add-in. If Visual Studio is currently running on your system, close it before continuing with this NPanday installation and configuration.
- Microsoft Service Pack 1: Must be installed. Most of the NPanday build tool features require this.

  To make sure you have the Service Pack 1, go to **Visual Studio Help** > **About** menu option. From the About screen, verify that the Service Pack version is enclosed in parenthesis after the Visual Studio Version.

  Another way to verify the Service Pack installation is to click **Copy Info** from the **About** screen and paste the information in a text editor. Verify that the SP installation information is in the list.

  For Visual Studio 2005

  ```
  Microsoft Visual Studio 2005 Professional
  Edition - ENU Service Pack 1 (KB926601)
  ```

  For Visual Studio 2008

  ```
  Microsoft Visual Studio 2008
  Version 9.0.30729.1 SP
  Microsoft .NET Framework
  Version 3.5 SP1
  ```
- Visual Studio .NET Framework Software Development Kit (SDK) version 2.0: Must be installed. The SDK usually installs automatically when you install Visual Studio 2005, but you might have to download it separately and put it into the right place.

  To make sure you have the SDK:

  Navigate to the following directory on your system:

  `C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0`

  Make sure the \v2.0 directory shown above contains all the pieces of the SDK (for example, the \Bin, \include, \CompactFramework, and \Lib directories, among others).

  Note: These are the default directories and it may be installed elsewhere on your system.

  To install the SDK if you don't already have it:

  If the directory structure/folder shown above does not exist or is empty, follow these instructions to get the SDK and put it into the correct location.

  Download the SDK from the following location:

http://msdn2.microsoft.com/enus/downloads/default.aspx

Note: The download places the SDK into a directory similar to the one shown below, instead of under Visual Studio 2005:

```
C:\Program Files\Microsoft.NET\SDK\v2.0
```

Note: It is not necessary to move the SDK after you download it (for example, to the location where Visual Studio puts it). Just make sure your PATH environment variable points to its location, as described in the PATH environment variable prerequisite information, below.

- .NET Install Root directory: Must be present. Verify that your system includes the following .NET Install Root directory:

  ```
  C:\WINDOWS\Microsoft.NET\Framework\v3.5
  ```

  Note: This directory should be in place by default from your Microsoft installation procedures. If it is not, refer to the appropriate Microsoft documentation for help.
- NUnit 2.2 or greater: Must be present for testing your projects. To download and install go to http://nunit.org/index.php?p=download. Follow the instructions there for installing it. Add the NUnit bin directory to your PATH (see below).
- `settings.xml` file: Must point to the repository that contains the .NET artifacts. This is typically found inside `[M2_HOME]/conf` directory, where `[M2_HOME]` is the Maven installation directory. Or, can also be found in `C:/Documents and Settings/[user_home]/.m2` where the `[user_home]` is the home directory of the user performing the installation.
- PATH environment variable: Must point to find the following:
  - The csc executable directory, the default path for which is: `C:\WINDOWS\Microsoft.NET\Framework\v3.5`
  - The devenv executable directory, the default path is typically: `C:\Program Files\Microsoft Visual Studio 9.0\Common7\IDE`
  - For testing your projects you will need the nunit-console executable on your path. For example, it can be found here for version 2.4.3 in a default installation: `C:\Program Files\NUnit 2.4.3\bin`.

  Typically, the the `PATH` environment variable will look similar to this:

  ```
  PATH=%PATH%;C:\Program Files\Microsoft Visual Studio 9.0\Common7\IDE;C:\WINDOW
  C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727;C:\Program Files\Microsoft Visua
  C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\bin;C:\Program Files\NUni
  ```

  Note:To successfully install the NPanday add-in on machines with only Visual Studio 2008 installed, add this in the PATH: `C:\Program Files\Microsoft SDKs\Windows\v6.0A\bin`.

  The method you use to set the PATH environment can vary, depending on how you usually launch Visual Studio:

  - If you are starting Visual Studio from the Windows Start menu, use the following to set the PATH environment variable in the System Properties: `Start > My Computer > View System Information > Advanced > Environment variables`
  - If you are running Visual Studio from a command shell (using the `devenv` executable), use the command line to set the PATH environment variable.
- ASP.NET MVC Release Candidate 2 which can be obtained from http://www.microsoft.com/downloads/details.aspx?FamilyID=ee4b2e97-8a72-449a-82d2-2f720d421031&displaylang=en.

This must be installed to skip the unsupported project types and be able to get the correct project type guid. Otherwise, the projects that are not supported will have a normal c# or vb project type guid and won't be skipped during project `Import`.

23.1.2 Installing NPanday .NET Build Tool

- Manual Installation
- Visual Studio Addin Installer

23.1.3 Verifying the NPanday .NET Build Tool

24

24.1 Visual Studio Addin Installer

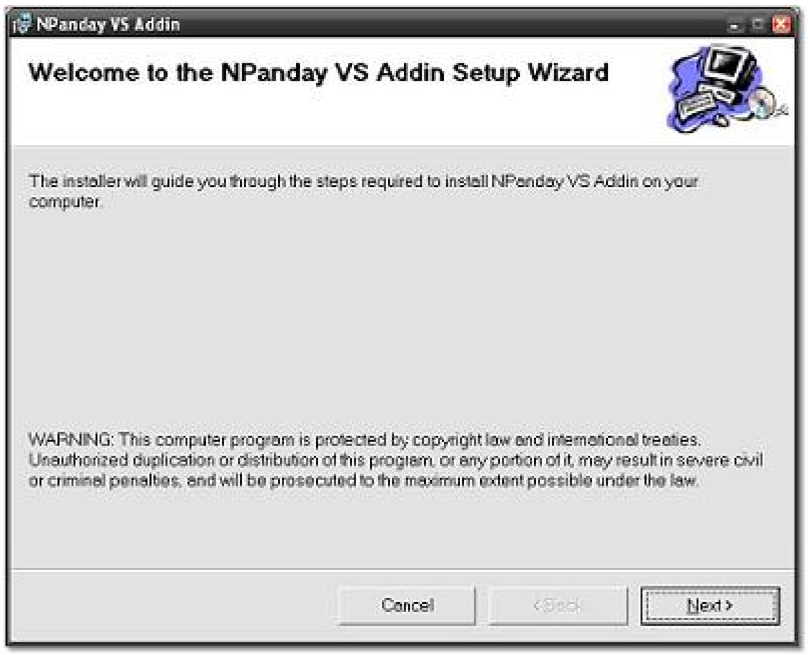
Visual Studio Addin installer does all the necessary configurations done when manually installing the build tool. The NPanday installers are available at the NPanday's site (click the VS Addin Installer or the VS Addin + Repository Installer link and accept the license agreement). NPanday releases link follows:

http://www.codeplex.com/npanday/Release/ProjectReleases.aspx

1 Extract the downloaded `NPanday.Installer.[version].zip` or `NPanday.Installer.[version]_with_repository.zip` file. This should contain the following files:

   - `NPanday.Setup.msi`
   - `setup.exe`

2 Double-click on the `NPanday.Setup.msi` or `setup.exe` file. This displays:
.



Welcome to the NPanday VS Addin Setup Wizard

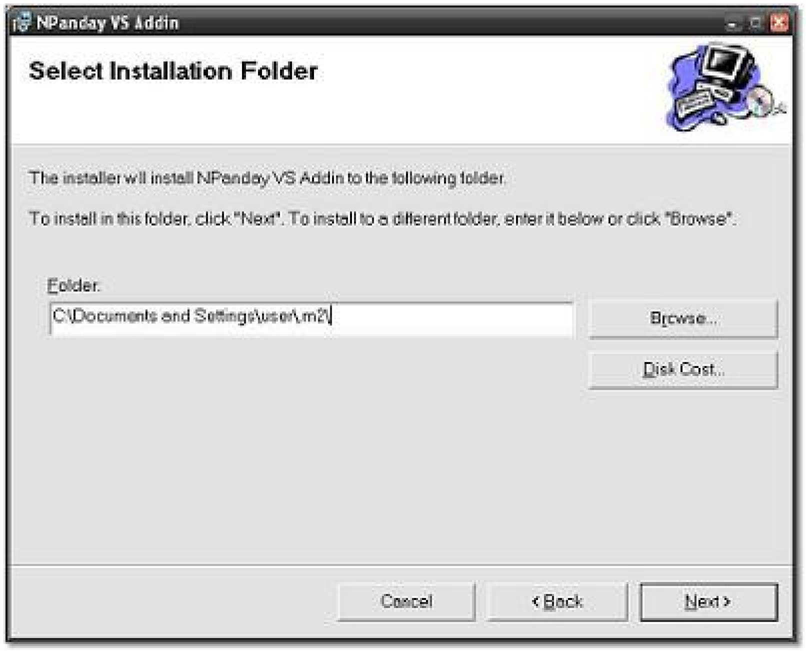
The installer will guide you through the steps required to install NPanday VS Addin on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

Install NPanday .NET Build Tool

Note: If NPanday .NET Build tool is already installed, double-clicking `NPanday.Setup.msi` or `setup.exe` file will display the NPanday VS Addin setup wizard which provides the options to repair and remove NPanday VS Addin. Repairing the add-in will re-install the corrupted and/or deleted files.

3 Click Next to proceed to Installation. Otherwise, click Cancel.
.



The installer will install NPanday VS Addin to the following folder.

To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".

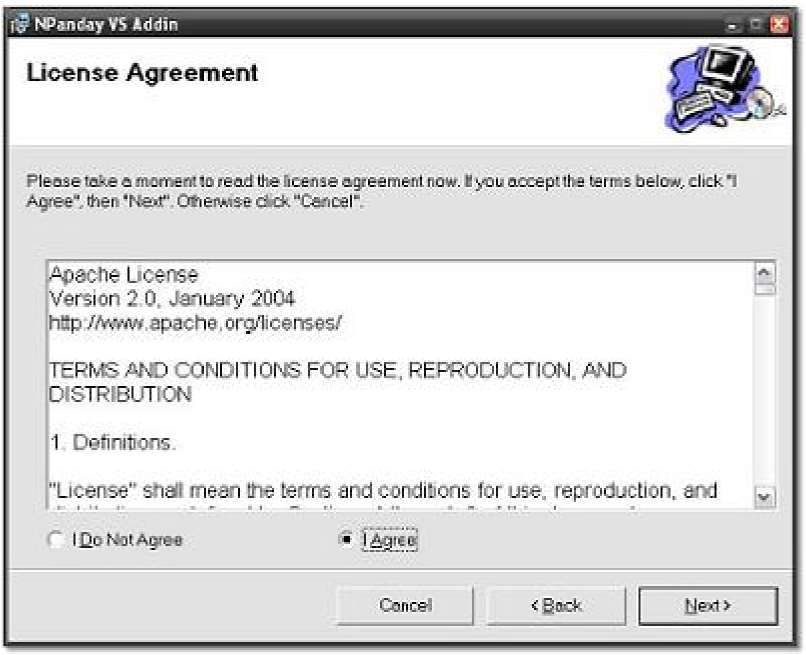
Folder:
C:\Documents and Settings\user\.m2\

Select Installation folder

4 Specify the folder where you want to install the NPanday .NET Build Tool then click Next. By default, the NPanday installation folder is set to `C:\Documents and Settings\[user_home]\.m2`.

5 Select I Agree to accept the license terms and click Next.

.



Please take a moment to read the license agreement now. If you accept the terms below, click "I Agree", then "Next". Otherwise click "Cancel".

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

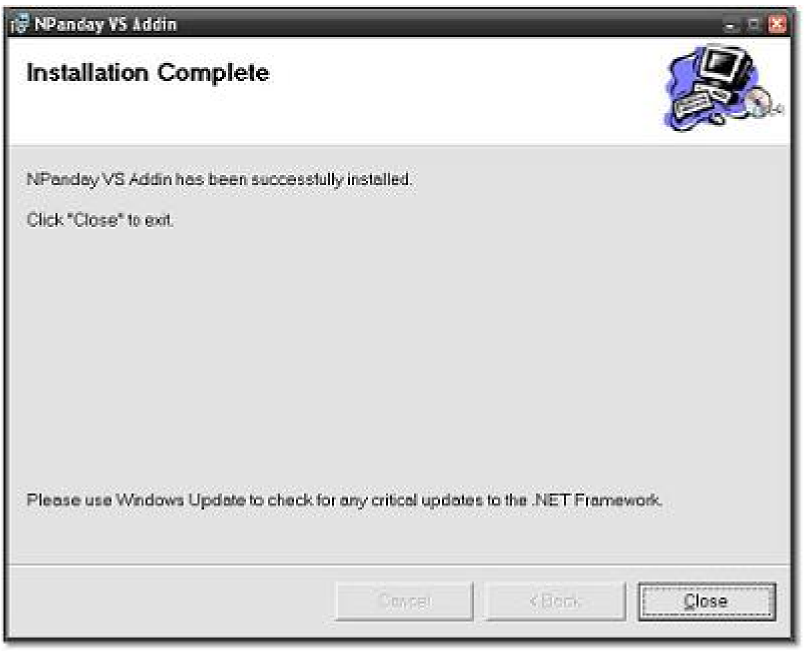
1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and

I Do Not Agree [✓ I Agree]

License Agreement

6 Wait for a few minutes while the installation is being initialized. Once the installation is completed, click Close.
.



Installation Complete

NPanday VS Addin has been successfully installed.

Click "Close" to exit.

Please use Windows Update to check for any critical updates to the .NET Framework.

Installation Complete Wizard

The table below lists the files and folders created after a successful installation of the NPanday VS Addin depending on the installer used.

For VS Addin Installer (`NPanday.Installer.[version].zip`), the following files and folders are installed:

- Visual Studio Addin
- `[NPanday_installation_folder]` which is set in "Select Installation folder". This should contain the following:
  - `license.rtf`
  - `\pab`
  - `\pab\NPanday.Setup.Ext`
  - `\pab\NPanday.VisualStudio.Addin`

For VS Addin Installer + Repository (`NPanday.Installer.[version]_with_repository.zip`), the following files and folders are installed:

- Visual Studio Addin
- `[NPanday_installation_folder]` which is set in "Select Installation folder". This should contain the following:
  - `license.rtf`
  - `\pab`
  - `\pab\repository` (contains NPanday artifacts)
  - `\pab\NPanday.Setup.Ext`
  - `\pab\NPanday.VisualStudio.Addin`

25

25.1 Uninstall Using the Installer

The NPanday VS Addin installer also provides the option to remove the add-in using the NPanday VS Addin setup wizard. The installers can be obtained from NPanday site (click the VS Addin Installer or the VS Addin + Repository Installer link and accept the license agreement).

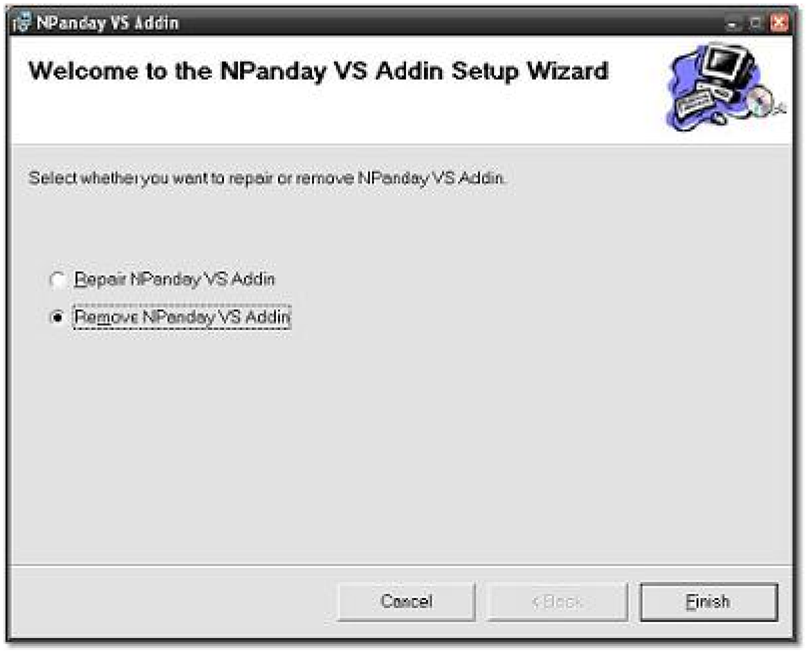
To remove the NPanday .NET Build Tool using the setup wizard, follow these steps:

1 Double click on the `NPanday.Setup.msi` or `setup.exe`.

If NPanday VS Addin is not found in the system, the wizard will guide you through the installation process.

2 Select Remove NPanday VS Addin to completely uninstall the add-in. Or, you can choose to Repair NPanday VS Addin to re-install corrupted and/or deleted add-in files.

.



NPanday VS Addin

Welcome to the NPanday VS Addin Setup Wizard

Select whether you want to repair or remove NPanday VS Addin.

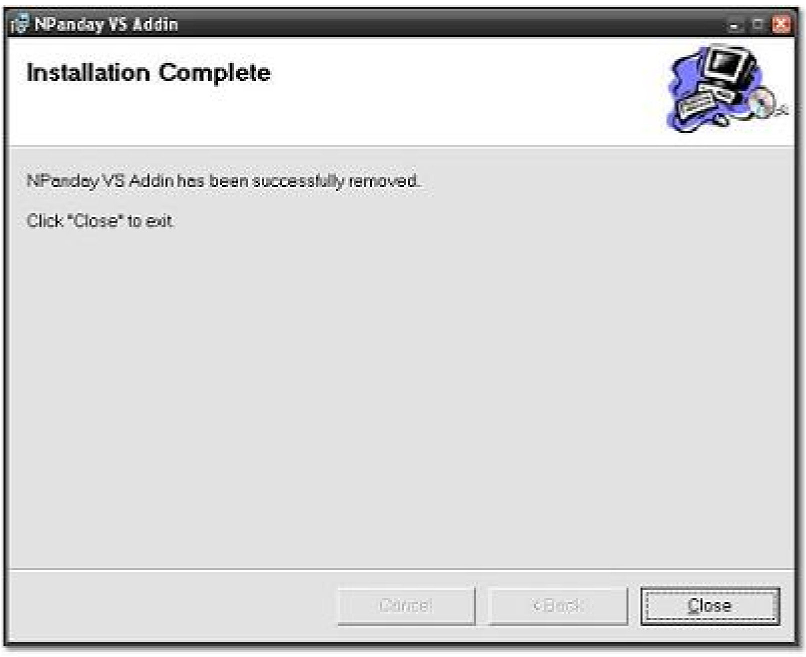
Repair NPanday VS Addin [✓ Remove NPanday VS Addin]

Remove NPanday VS Add-In

3 Click Finish. This will remove NPanday files installed. This may take a few minutes to complete.

4 When NPanday .NET Build Tool is completely removed from your machine, click Close.

.



Installation Complete

NPanday VS Addin has been successfully removed.

Click "Close" to exit.

Successfully Removed NPanday VS Add-In

Removing NPanday VS Addin deletes the following:

- VS Addin found in `C:\Documents and Settings\[user_home]\My Documents\Visual Studio 2005\Addins`
- Files (including the `license.rtf`) and folders in the NPanday installation folder. The installation folder is the location specified during the installation of the add-in. Otherwise, this is set to the default location, `C:\Documents and Settings\[user_home]\.m2\`.

26

26.1 Manual Installation

Manual installation would require the installation of the NPanday repository which can be downloaded from the NPanday site (click the Repository link and accept the license agreement).

1 Edit the `settings.xml` file so it points to the repository that contains the NPanday plugins. The settings.xml file is typically found at: `C:/Documents and Settings/[user_home]/.m2` or in `[M2_HOME]/conf` .

2 Set your PATH environment variable as explained in the Pre-requisites section. For example:

```
PATH=%PATH%;C:\Program Files\Microsoft Visual Studio 9.0\Common7\IDE;C:
\WINDOWS\Microsoft.NET\Framework\v3.5; C:\WINDOWS\Microsoft.NET
\Framework\v2.0.50727;C:\Program Files\Microsoft Visual Studio 9\SDK
\v3.5\bin; C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\bin;C:
\Program Files\NUnit 2.4.3\bin
```

3 Run the `maven-vsinstaller-plugin` to resolve (and download if needed) all of NPanday's dependencies and create the Visual Studio add-in file in the user's home directory. Run the `maven-vsinstaller-plugin` from any directory:

```
mvn npanday.plugin:maven-vsinstaller-plugin:[version]:install
```

Note: Close the Visual Studio Integrated Development Environment (IDE) before installing `maven-vsinstaller-plugin`.

4 This creates the NPanday .NET Build Tool add-in and puts it into your home directory. When you start Visual Studio it will be available for use. After running the `maven-vsinstaller-plugin` the add-in will be located here:

```
C:\Documents and Settings\[user_home]\My Documents\Visual Studio
2005\Addins\NPanday.VisualStudio.AddIn
```

27

27.1 Uninstalling Manually

To manually remove the NPanday .NET Build Tool, follow these steps:

1. Locate and delete your previous version of the NPanday .NET Build Tool for Visual Studio. It is typically found at the following location, where [user_home] is your home directory:

   `C:\Documents and Settings\[user_home]\My Documents\Visual Studio 2005\Addins\NPanday.VisualStudio.AddIn`
2. Clear the cache by typing the following in a command shell:

   `C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\mscorcfg.msc`
3. Go to My Computer > AssemblyCache, click View List. Delete the following two files:
   - `Npanday.Model.Pom`
   - `NPanday.Plugin`
4. Delete the following directories:
   - `C:\Documents and Settings\[user_home]\.m2\pab`
   - `C:\Documents and Settings\[user_home]\.m2\uac`
5. Remove the following file `C:\Documents and Settings\[user_home]\.m2\npanday-settings.xml`
6. Remove the `C:\Documents and Settings\[user_home]\.m2\repository` directory.

Note: This deletes all artifacts for all your projects, not just the NPanday ones. This is not a problem because when you re-build your projects, Maven will automatically recreate this directory and download the needed jars. The only time this is an issue is if you have manually downloaded jars, such as the Sun jar files. In that case you will have to manually download them again if this directory is removed.

28

28.1 Uninstalling NPanday .NET Build Tool

This section explains how to remove the NPanday .Net Build Tool from your system completely either manually or using the NPanday VS Addin installers.

Note: If you want to install a newer version of the tool, we recommend starting from a “clean” system. Follow this procedure for removing the tool, then perform the Installing the NPanday .NET Build Tool procedure.

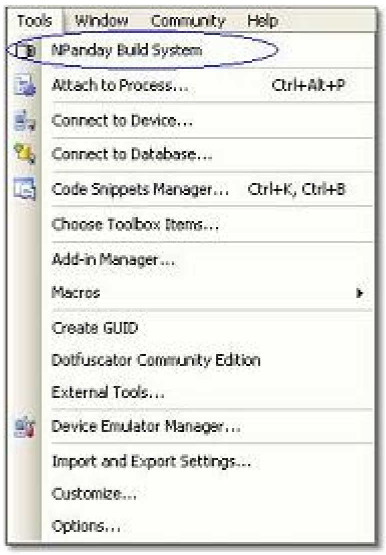
29

29.1 Verifying the NPanday .NET Build Tool Installation

To verify that the NPanday .NET Build Tool has been configured inside Visual Studio, perform the following instructions:

1. Launch Visual Studio 2005 (either from the command line or from the Windows Start> menu).
2. From inside Visual Studio, select Tools>NPanday Build System. This automatically starts the embedded Maven Build Server (the add-in) and displays a confirmation in the Visual Studio Output window at the bottom of the Visual Studio screen.

   You can set Visual Studio to automatically start NPanday Build System upon startup by checking NPanday.VisualStudio.Addin in Tools > Add-in Manager.

   NPanday Build System

   After the Maven embedded server starts, you will not see NPanday Build System listed in the Tools menu. That is, until the next time you start Visual Studio.
3. You are now ready to create a Solution, so proceed to the next section.

30

30.1 Available Plugins

- `aspx`
- `compile`
- `deploy`
- `fxcop`
- `install`
- `link`
- `mojo-generator`
- `repository`
- `resgen`
- `resolver`
- `test`
- `vsinstaller`
- `vstudio`
- `webapp`
- `wix`
- `wsdl`
- `xsd`
- `xsp`

31

31.1 Release Notes for NPanday 1.0

NPanday .NET Build Tool release is now available for download from the web site.

NPanday FAQs provides answers to common questions regarding NPanday .NET Build Tool.

31.1.1 NPanday 1.0

31.1.1.1 New Features

31.1.1.2 Improvements

31.1.1.3 Bug Fixes

32

32.1 NPanday List of Workarounds

- NPanday generated output files have matching versions of that specified in the project's pom.xml version. If the output file does not match that of the pom.xml you need to specify the version of the maven-compile-plugin to the match that of the version of the NPanday that you are using.
- Re-Importing a Project loses some references. Make sure that the added artifacts can be found in `gac` and/or local and/or remote repository, if not it won't be added in the pom though it is present in the References of VS.
- Visual Basic Test goal errors can be due to incompatibilities between NPanday and the NUnit version used in the project. Changing the version of NUnit dependency in the project's `pom.xml` to `<version><<2.2.9.0>></version>` will run the project successfully.
- Visual Studio Addin is unable to build (compile) without installing the project. As a workaround, execute `All NPanday Projects > Install` first before `All NPanday Projects > Build [compile]`.
- NPanday is not recognizing Visual Basic test codes. The project should be re-imported using `Generate Solution's POM Information...` option from the menu before executing the `All NPanday Projects > Test` or `All NPanday Projects > Install` goal.
- NPanday is unable to build projects with dependencies in remote repository. As a workaround, reimport the project.
- NPanday cannot release with references from `gac`. To make this work, execute `npanday.plugin:NPanday.Plugin.Sysref.JavaBinding:prepare` before releasing the project.
- Test goal output result is not refreshed for C# test project. To make this work, reimport the project and change the value of the test. Then, perform `Current Project: Test`.
- Addin Error with Test Goal is encountered. Verification can be done in two ways as listed below.

  I Use ProjectImporter. This creates two separate NPanday projects.

    1 Import project with a main and a test module.
    2 Perform `Maven Phase > All Project : Install`.
    3 Perform `Maven Phase > All Project : Test`.

  II Manually update the pom to specify the `<testSourceDirectory>`.

    1 Import project with a main and a test module.
    2 Update `<testSourceDirectory>` to point to the test project.
    3 Update `<testIncludeSources>` in the configuration element to add the test source codes individually.

    1 Perform `Maven Phase > All Project : Install`.
    2 Perform `Maven Phase > All Project : Test`.