|  |  |
| --- | --- |
|  | Microsoft Biology Initiative Onboarding Guide  Version 1.1 - Oct 2010 |

Abstract

The Microsoft® Biology Initiative (MBI) is a language-neutral bioinformatics toolkit—built as an extension to the Microsoft .NET Framework—that includes the Microsoft Biology Foundation (MBF) and Microsoft Biology Tools (MBT). MBI is available under an open source license.

This document describes the steps for onboarding to the open source project so that you can contribute code to the Microsoft Biology Initiative.

For updates to this document and the rest of the MBI documentation, see   
<http://mbf.codeplex.com/documentation>.

Contents

[Introduction 3](#_Toc274060506)

[Set Up Your Environment 3](#_Toc274060507)

[Install Visual Studio 2010 3](#_Toc274060508)

[Connect to the TFS Server 3](#_Toc274060509)

[Work with the Source Tree 8](#_Toc274060510)

[Build the MBI Project 11](#_Toc274060511)

[Generate the Installers 12](#_Toc274060512)

[Test the Code 13](#_Toc274060513)

[Build the API Reference CHM 14](#_Toc274060514)

[Manage Work Items 15](#_Toc274060515)

[Conclusion 18](#_Toc274060516)

Disclaimer: This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

© 2010 Microsoft Corporation. All rights reserved.

Microsoft, Azure, MSDN, Silverlight, Visual Studio, and Windows are trademarks of the Microsoft group of companies.  All other trademarks are property of their respective owners.

# Introduction

Welcome to the Microsoft Biology Initiative (MBI), an open source initiative to support bioinformatics research. The initiative consists of two components:

* Microsoft Biology Foundation (MBF) is an open source, reusable .NET Framework library and application programming interface (API) for bioinformatics research.
* Microsoft Biology Tools (MBT) is a set of bioinformatics applications.

Interested users can obtain the source code and contribute to either MBF or MBT. This document gets you started by describing how to connect to the project’s source code repository and work with the source tree. For more about MBI:

* For general information, see the MBI homepage: <http://research.microsoft.com/bio>
* For technical questions, use the MBI community forum: <http://www.getsatisfaction.com/bio>

**Note** This is a living document. If you find mistakes or areas that are confusing as you go through the process of getting setup, please log a work item to update the document. See “Create a Work Item” later in this document for details. Thanks!

# Set Up Your Environment

The MBI source code repository is hosted on a Microsoft extranet server and managed by Microsoft® Visual® Studio Team Foundation Server (TFS). MBI contributors can access this server and actively participate in the ongoing development of MBI.

The MBI project also periodically posts stable snapshots of the source tree to the [MBI CodePlex Web](http://mbf.codeplex.com/) site. If you are interested in MBI but do not want to submit code to the repository, you can obtain a copy of the source tree by downloading a snapshot. You don’t need partner or contributor status for such downloads, the snapshots are available to any interested user. See the MBI CodePlex Web site for details.

This section describes how to set up your system and connect to the active repository. It is intended primarily for contributors, and focuses on the tools and procedures that they must use.

## Become a Contributor and Partner

You must be an MBI contributor to access the active repository and submit code to the project. If you are interested in becoming a contributor, contact the MBI team through the [MBI Community Forum](http://getsatisfaction.com/mbi) or the MBI CodePlex Web site.

Contributors must also have partner credentials, which grants access the extranet servers that host the repository. If you have partner credentials from previous MBI TFS projects, you should already have access to the extranet servers. Otherwise, you can obtain partner credentials by contacting the MBI team, as described in the previous paragraph.

## Install Visual Studio 2010

The MBI project uses TFS to manage the development process in the active repository. TFS includes collaboration components that support team development, such as source code control and work item/bug tracking.

To be a contributor and participate fully in the MBI project, you must have Visual Studio 2010 and Visual Studio Team Explorer installed on your system. These two products provide you a complete development environment, including the source-control tools you need to interact with the MBI repository through the TFS server. For more information on installing these products, see [Getting Started with Team Foundation](http://msdn2.microsoft.com/en-us/library/ms181301(vs.90).aspx) on MSDN.

**Tip:** Visual Studio is a retail product. However if you are in Academia, you can obtain Visual Studio for free through the [DreamSpark](https://www.dreamspark.com/) program.

## Connect to the TFS Server

To access the MBI repository, you must select Visual Studio Team Foundation Server as your source control plug-in and connect to the MBI TFS server on the extranet. If you are already familiar with TFS, you can connect by using the following information.

Server: VSTF-EU-DUB-01.partners.extranet.microsoft.com

Path: TFS

Port: 8443

Protocol: https

Project: MBI

If you are new to TFS, the remainder of this section describes the process in detail

### Step 1: Specify TFS

If you haven’t already done so, you must specify TFS as your source-control plug-in.

Select TFS as the source-control plug-in

1. On the Visual Studio **Tools** menu, click **Options** to open the **Options** dialog box.

2. In the left panel of the **Options** dialog box.

Expand the Source Control item and click Plug-in Selection.

In the right panel, set **Current source control plug-in:** to Visual Studio Team Foundation Server. Click **OK**.

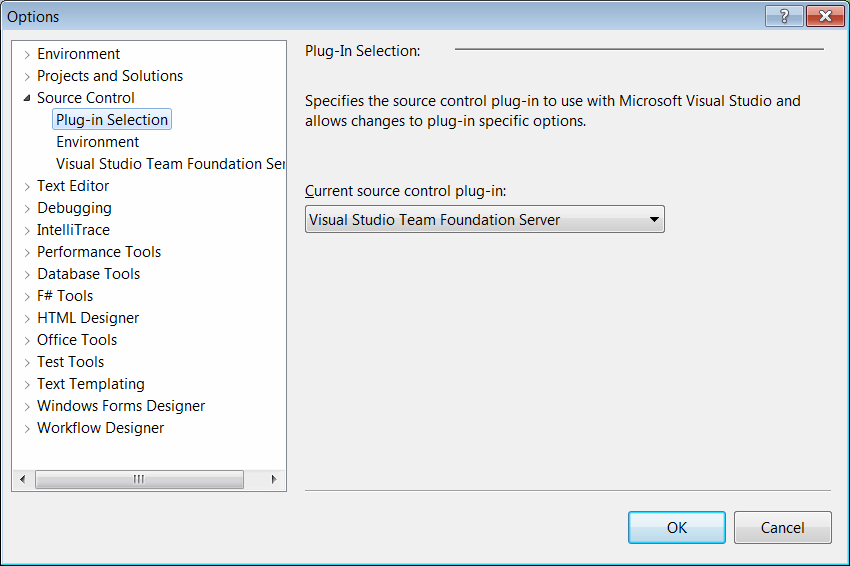


Figure 1. Visual Studio Options dialog box

### Step 2: Connect to the MBI server

You can now connect to the MBI server.

To connect to the MBI server

1. On the **Team** menu, select **Connect to Team Foundation Server**, which opens the **Connect to Team Project** dialog box

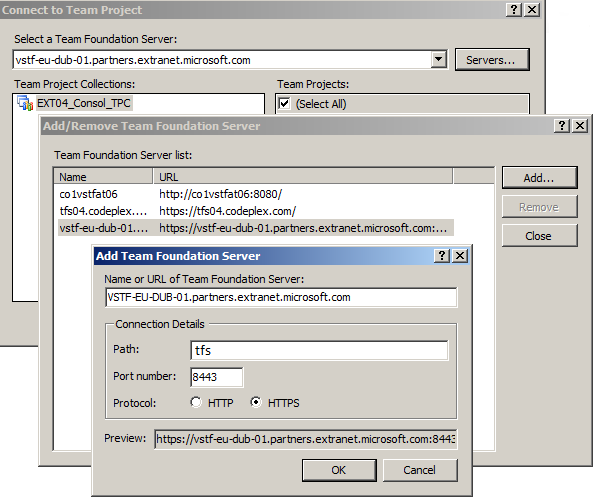


Figure 2. Connect to Team Project and related dialog boxes

2. In the Connect to Team Foundation Server dialog Box, click **Servers**.

3. In the **Add/Remove Team Foundation Server** dialog box, click **Add**.

4. In the **Add Team Foundation Server** dialog box:

Enter the name of the TFS server that hosts the project—VSTF-EU-DUB-01.partners.extranet.microsoft.com.

Set the Path to: TFS.

Set the Port number is set to 8443 and select the HTTPS protocol.

After you click **OK**, VSTF-EU-DUB-01.partners.extranet.microsoft.com should appear in your list of Team Foundation Servers.

5. Click **Close** to return to the **Connect to Team Foundation Server** dialog box.

6. Make sure that the **MBI** Team Project is checked, and then click **Connect**.

The project now appears in your **Team Explorer** window. If the **Team Explorer** window is not visible, press the key sequence **CTRL+\** followed by **CTRL+M** to activate it.

### Step 3: Create a Workspace

The next step is to create a **Workspace** that defines how you will work with the source on your machine.

To create a workspace

1. Create a root folder on your local system to host the source tree.

You can choose any convenient name, so the remainder of the paper refers to the root folder generically as [MBI Root].

**Caution:** The name that you choose for [MBI Root]should not have any spaces, which can interfere with some of the MBI tools.

2. In **Team Explorer**, double-click the **Source Control** tree item to launch the **Source Control Explorer** on the **MBI** project.

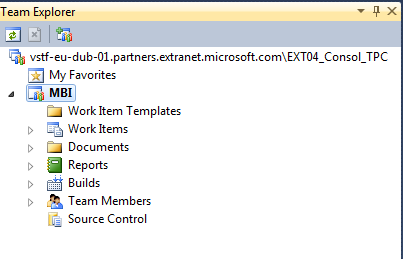


Figure 3. Team Explorer

3. In the **Source Control Explorer** window, click Workspaces in the dropdown menu to display the **Manage Workspaces** dialog box.

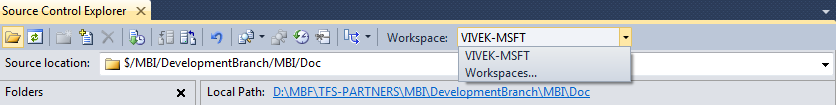


Figure 4. Source Control Explorer tool bar

TFS uses workspaces to put files that are under source control on your system, so you can work on them locally. Because you might have multiple enlistments in the source control system, you must to create a workspace name for each enlistment and tell TFS how to associate the files on the server with directories and files on your local hard drive.

4. Click **Edit** to open the Edit Workspace dialog box and create an MBI workspace.

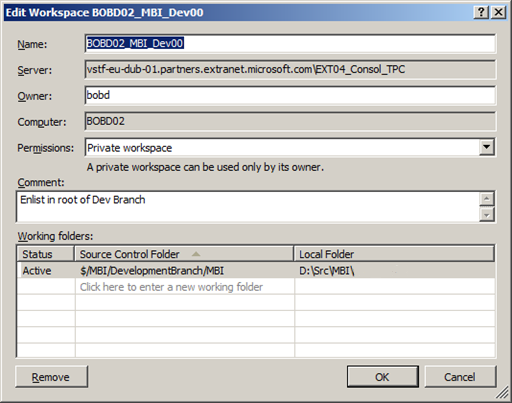


Figure 5. Edit Workspace dialog box

Most of these fields will already be filled. You will be working in the repository’s Development branch, so a good working model is to map that branch to the [MBI Root] folder on your system, which you created earlier. The complete MBI Development branch is small enough that we recommend that you map the entire branch.

To map the Development branch to [MBI Root]:

1. Click an empty line under the Source Control Folder heading in the Working folders pane and then click the B**rowse** […] button at the right edge of the line.

2. Select $/MBI/DevelopmentBranch/MBI.

3. Click an empty line under the Local Folder heading in the Working folders pane and then click the B**rowse** […] button at the right edge of the line.

4. Select the [MBI Root] folder that you created earlier and click **OK**.

If you do not map the entire Development branch, it is a *really* good idea to keep the path relationships the same in the part of the branch that you do map.Doing so reduces confusion and some projects have built-in tree-related assumptions.

### Step 4: Synchronize your Workspace with the Repository

The final step is to populate your workspace by synchronizing it with the Development Branch on the server.

To synchronize your workspace

* In the **Source Control Explorer** window, right-click the **MBI** project and click **Get Latest Version**.

Thiscopies the current version of the source files from the repository to your workspace.

**Tip:** Source control is an important part of the development process and you will be working with TFS frequently. It is important that you become well acquainted with it. For more information on using TFS, see [Getting Started with Team Foundation](http://msdn2.microsoft.com/en-us/library/ms181301(vs.90).aspx) on MSDN.

# Work with the Source Tree

You should now have the source tree on your system and be able to build the tree. Figure 6 shows the the source tree’s root folder in Source Control Explorer.

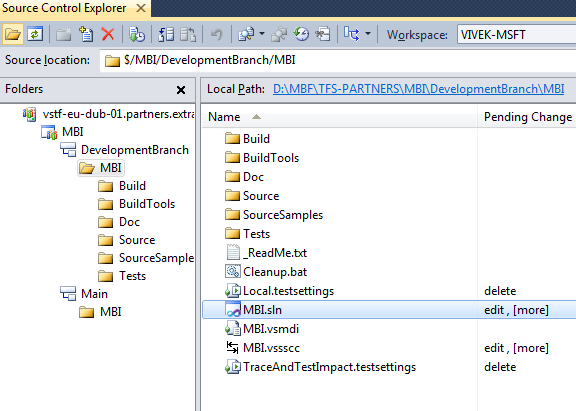


Figure 6. MBI project

**Source Control Explorer** allows you to manage your interactions with the repository, such as synchronizing the tree or checking in new code. To work on the project files on your local system, you use **Solution Explorer**.

To open the MBI project in **Solution Explorer**, double-click MBI.sln in the project’s **Source Control Explorer** root folder. To view the project in **Solution Explorer**, click the tab on the bottom left of the **Explorer** pane, next to the **Team Explorer** tab. Figure 7 shows the top level of the MBI tree in Solution Explorer, which shows the various projects that make up MBI, including the MBI project.

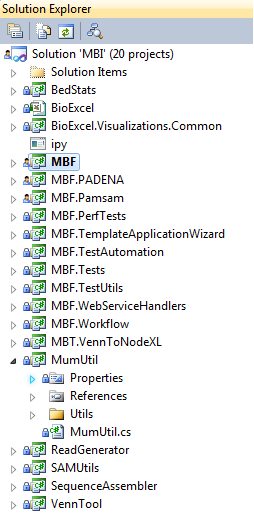


Figure 8. MBI project in Solution Explorer

For information on the various projects in Figure 8, see the [MBI Home page](http://research.microsoft.com/en-us/projects/bio/).

You can now use **Solution Explorer** to access the files in the source tree. For example, if you expand the Bio component and select the ISequence.cs item, **Solution Explorer** displays the file in the Visual Studio editor, as shown in Figure 9.

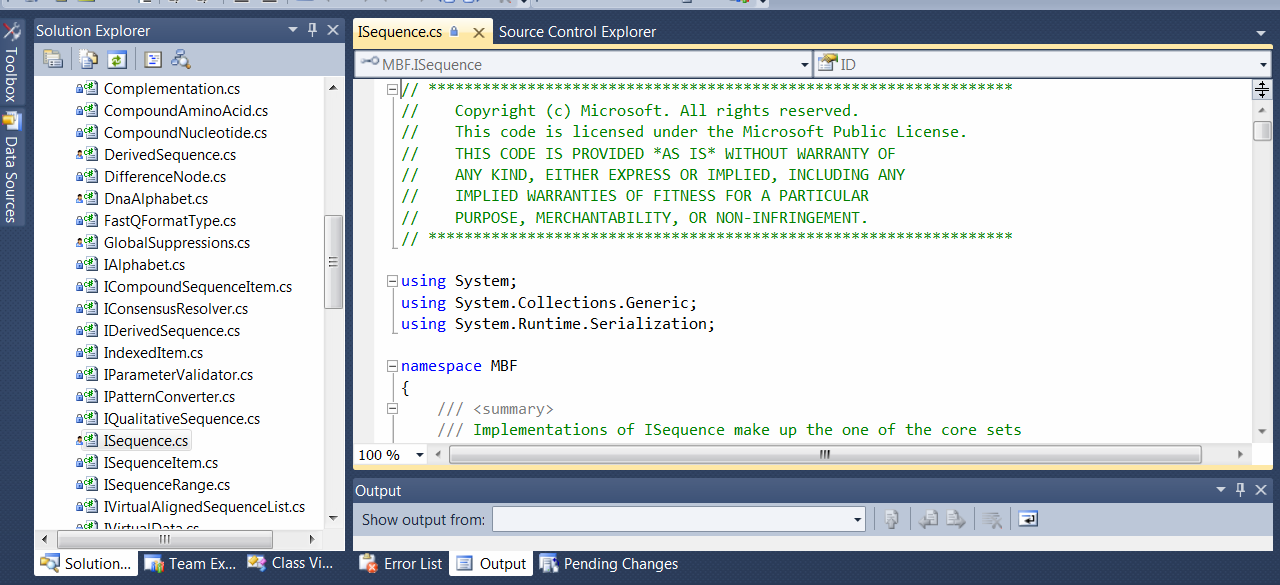


Figure 9. Viewing and editing source files

Visual Studio also provides quite a bit of navigational assistance through context menus. For example, if you right-click the term ISequence in the interface declaration, you can click **Find All References** in the popup menu to list the other places in the source tree that use this interface, as shown in Figure 10.

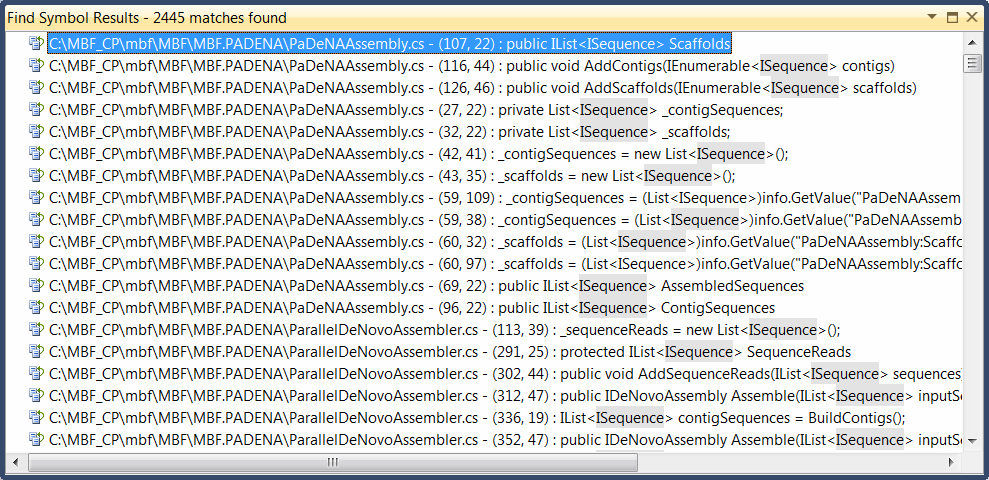


Figure 10. Locating references

You can then double-click any item in the list to jump to the location in the corresponding source file, which provides a bit more context and examples of how the interface can be used in your own code:

## Build the MBI Project

Now that you have loaded the MBI project, you can build it and produce the associated DLLs and executable.

First, choose whether to build a Release or Debug version of the code by clicking the appropriate item in the toolbar’s drop-down menu, as shown in Figure 11:

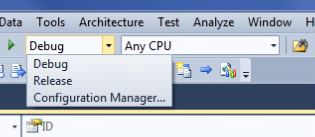


Figure 11. Specifying the build version

Similarly, choose the target build environment in the adjacent dropdown menu:

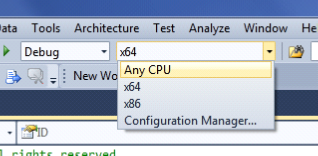


Figure 12. Specifying the build target

To build the entire MBI solution, open the **Build** menu and click **Build Solution**. You can also build individual projects by right-clicking the project in **Solution Explorer** and clicking **Build** in the popup menu.

To monitor the progress of the build, enable the **Output** window by clicking **Output** on the **View** menu. When the build is complete, the **Output** window shows the following information:

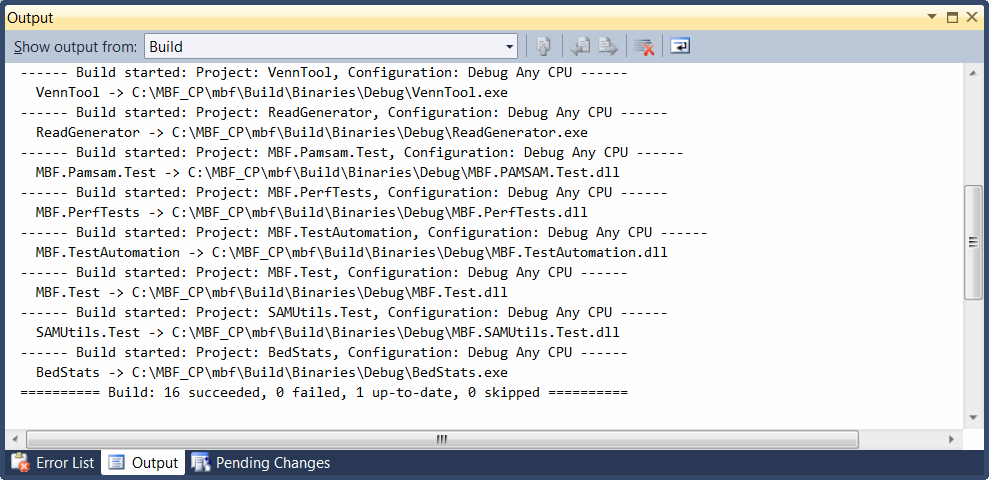


Figure 13. Output window

## Generate the Installers

To install MBI components, you must create an installer, which is packaged as a MSI file. The MBI project includes a set of batch files to build installers for various MBI components, which can be found in the following locations:

* Common scripts used by all installers: $\MBI\DevelopmentBranch\MBI\BuildTools\BuildScripts
* MBF installers build script: $\MBI\DevelopmentBranch\MBI\Source\MBF\Installer
* MBT installers build script: $\MBI\DevelopmentBranch\MBI\Source\MBT\Installer

To generate an installer, open a command window—sometimes called a DOS window. You can then create installers by running CreateSetup.cmd from the appropriate location, as shown in the following table

Table 1. MBI Installers

|  |  |
| --- | --- |
| **To build:** | **Run CreateSetup.cmd from:** |
| MBF.msi | [Root]\Source\MBF\Installer\ |
| MBF.msi + SequenceAssembler.msi | [Root]\Source\MBT\Installer\SequenceAssembler\ |
| MBF.msi + BioExcel.msi | [Root]\Source\MBT\Installer\BioExcel\ |
| MBF.msi + SequenceAssembler.msi + BioExcel.msi | [Root]\Source\MBT\Installer\ |

The installer and related build output are placed under: $\MBI\DevelopmentBranch\MBI\Build\LocalBuild,

## Test the Code

MBI uses VSTest as the primary means for ensuring code quality, which makes it easy to develop, integrate, and run test code against your contributions.

The MBI source tree includes VSTest tests that support both unit and automated tests that exercise each interface, function, and feature. The test code can be found in the MBF.Test and MBF.TestAutomation projects.

Important: If you develop any new code, or extend any existing code, you must implement a corresponding VSTest test and include it when you submit the changes. This ensures that subsequent development does not inadvertently cause a breaking change to another part of the code base. It also informs others about the intent of a particular piece of code—thus helping people who are new to the project to come up to speed on areas of interest.

To run the tests

1. Click **Windows** on the **Test** menu and open the **Test View** window.

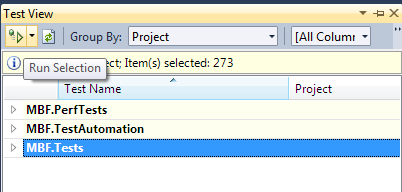
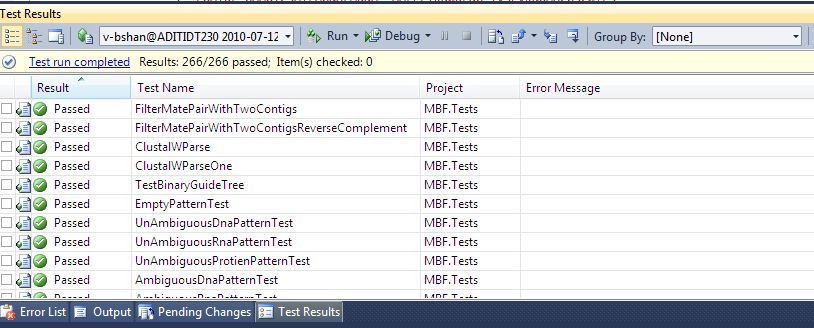
2. In the **Test View** window select MBF.Tests and click **Run Selection**—the green arrow in the upper left corner—as shown in Figure 14.

Figure 14. Test View window

All test cases should pass and VS Test should display a Test Results window that looks something like Figure 15.



## Figure 15. Test Results windowBuild the API Reference CHM

You can build a local version of the MBF API Reference Help file (CHM) that is synchronized with the source tree on your system by using Sandcastle. The required binaries are part of the source tree, so you do not need to install Sandcastle or HTML Help Workshop. Instead, run the following command:

[MBI Root]\BuildTools\BuildScripts\GenerateDocumentLocally.cmd

The GenerateDocumentLocally.cmd script checks for the required prerequisites—Visual Studio Tools for Office Runtime 4 and Microsoft Office Excel 2007 or 2010—builds the MBI solution and generates the API reference (MBF.chm). The CHM is placed in the [MBI Root]\Build\LocalBuild folder.**Note:** Before building the CHM, you must do a complete MBI build, which creates current versions of the necessary DLL and XML files and places them in the \Build folder.

# Manage Work Items

MBI uses TFS work items to manage the project’s workflow. Please see the MSDN article to create a work item <http://msdn.microsoft.com/en-us/library/ms181316.aspx>







# Conclusion

This document was intended to provide a quick introduction to get you started developing for MBI. Before you attempt to modify the code base, spend some time gaining a thorough understanding of how all the different parts of MBI work together and familiarize yourself with MBI Coding Guidelines document (MBF\_Coding\_Conventions.docx). A good background on the project will help to make sure your first code review goes smoothly.

Welcome to MBF!