

# How to Create a MindManager Add-in With Visual Studio in 7 Steps

## Prerequisites:

MindManager 7, 8 or 9 installed

Visual Studio 2005, 2008 or 2010 installed

## Step One

The first thing to do is download this small Registry file. This contains the registry settings that allows VS 2005, 2008 and 2010 to list MindManager 7, 8 and 9 in the Shared Add-in Programs list when creating a Shared Add-in with VS.

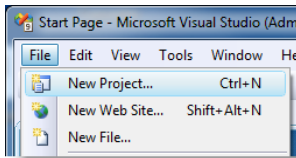
You can download the file from:

[http://www.olympic-limited.net/development/resource/vs\\_mindmanager\\_registry\\_settings.zip](http://www.olympic-limited.net/development/resource/vs_mindmanager_registry_settings.zip)

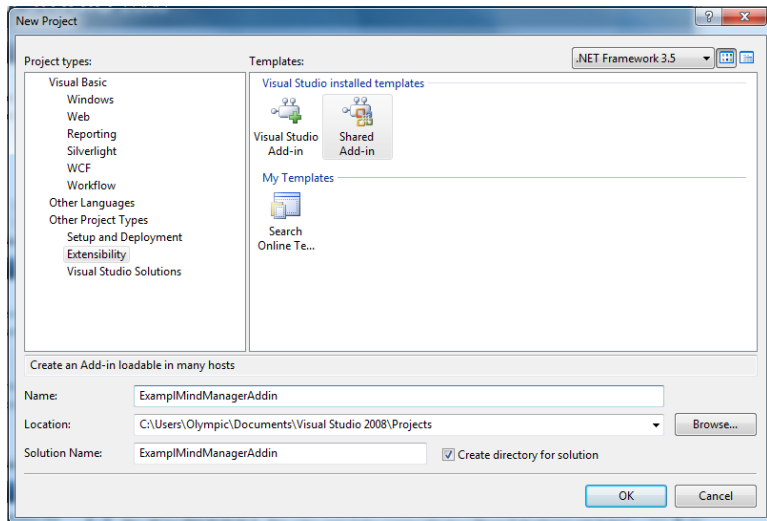
Once downloaded, extract the file to your desktop and then double-click the file to execute it. Allow the registry changes to proceed when prompted.

## Step 2

Fire up Visual Studio and select **New Project**.



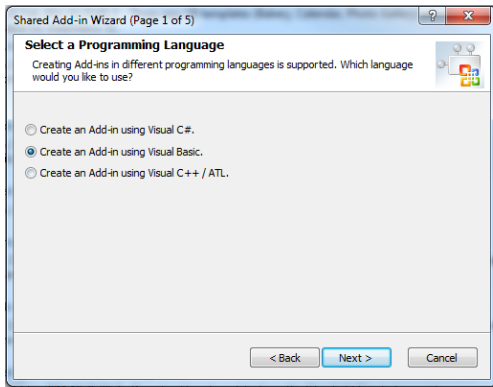
Then select **Shared Add-in** from the **Other Project Types** Option.



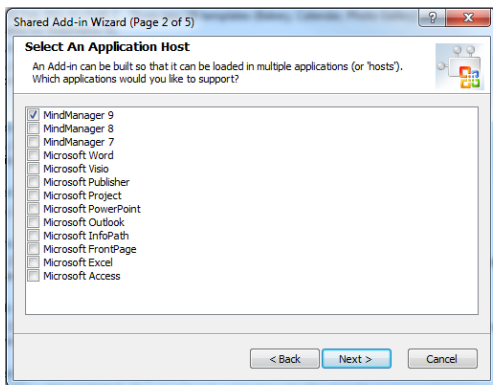
Give the add-in a name and select where to create the solution files. Selecting **OK** will invoke the **Add-in Wizard**.



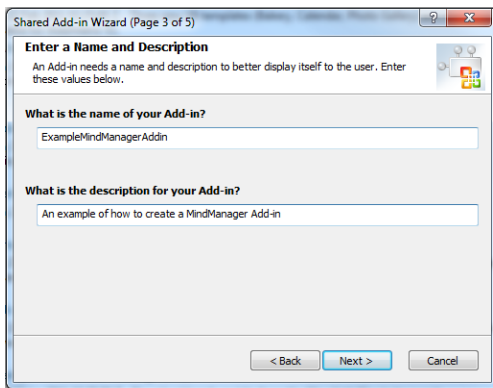
On the next dialog select the programming language you are going to use to develop your add-in. In this case we are using VB.



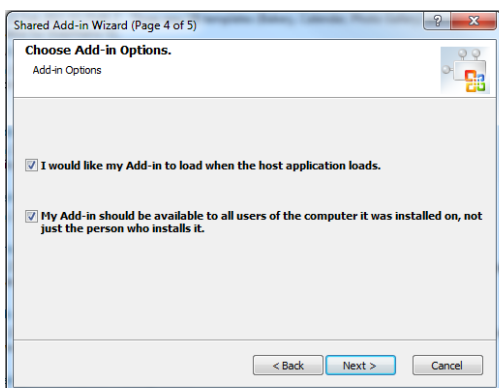
Next select the application your shared add-in will be hosted within. In this case **MindManager 9**. The MindManager entries appear in this list because of the Registry settings we added in Step 1.



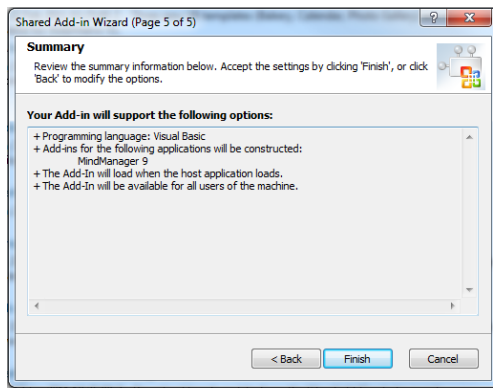
On the next dialog type the name of your add-in together with a description.



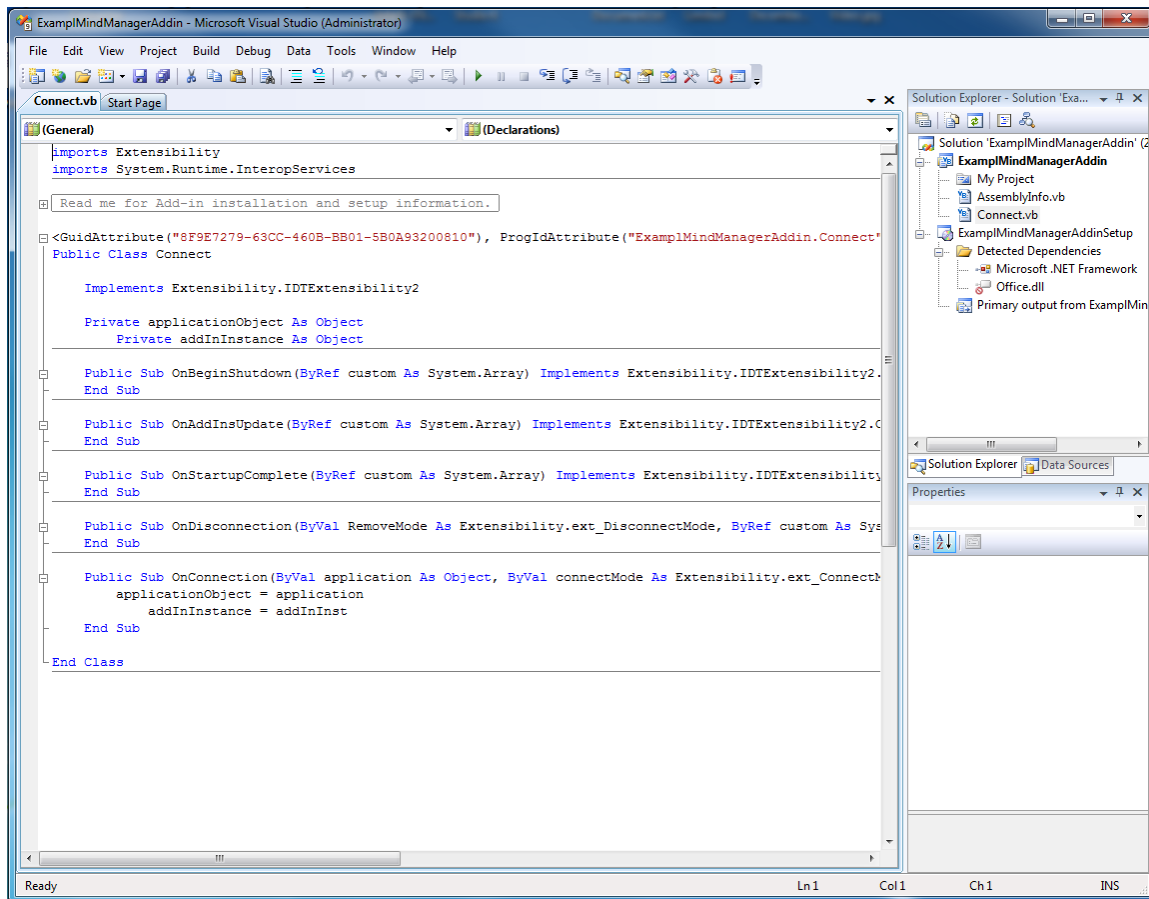
Next select the load and user availability options for the add-in. I have always found add-ins more reliable when both these options are selected at this point.



Click **Next** to see a short summary of the settings and then click **Finish** to create the add-in.

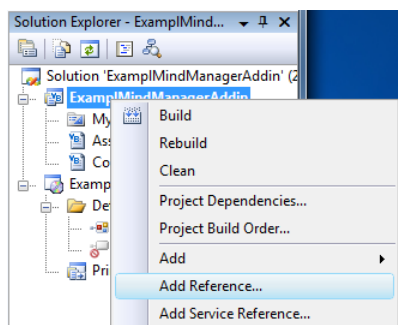


If everything goes according to plan VS will display the newly generated project as shown below.



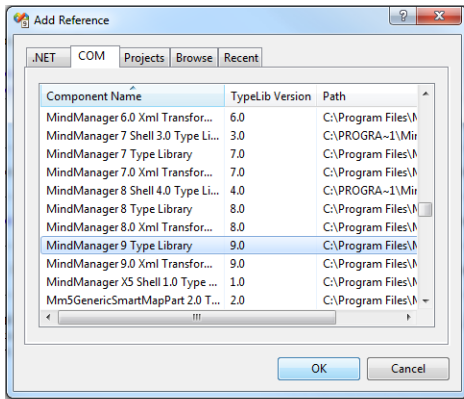
### Step 3

Now we need to add a reference to the **MindManager Type Library** we are going to need. In this example we are using MindManager 9 so we right-click the **Project** in the **Solution Explorer**.



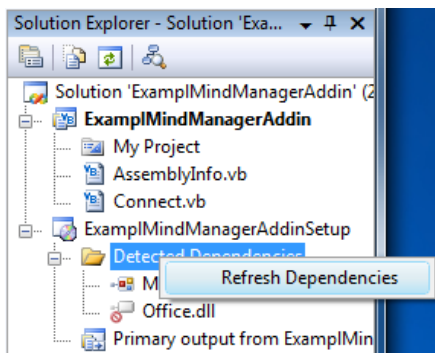
Select **Add Reference**.

When the **Add Reference** dialog is displayed, select the **COM** tab and scroll down the list until you find the **Type Library** entry for the version of **MindManager** you are developing your add-in for.

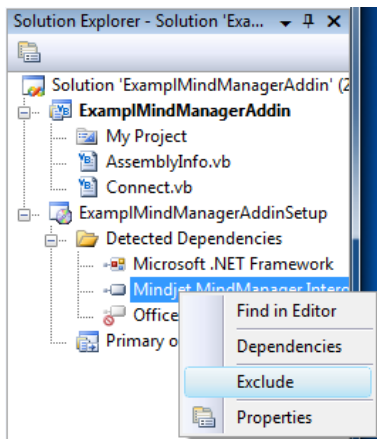


Click **OK**.

Right-click the **Detected Dependencies** entry in the **Set Up Project** within the **Solution Explorer** and select **Refresh Dependencies** to check that the **Reference** has been added successfully.



If it has you will see the **Mindjet.MindManager.Interop** dependency. Mindjet recommend that you exclude this from the **Set Up** so right-click and select **Exclude**.



## Step 4

Now we need to make some changes to the **Connect** code. Amend the automatically generated code for the **Connect Class** to resemble the following:

```
Public Class Connect

    Implements Extensibility.IDTExtensibility2

    Private m_applicationObject As Mindjet.MindManager.Interop.Application
    Private o_addInInstance As Object

    Public Sub OnBeginShutdown(ByRef custom As System.Array) Implements
Extensibility.IDTExtensibility2.OnBeginShutdown
        End Sub

    Public Sub OnAddInsUpdate(ByRef custom As System.Array) Implements
Extensibility.IDTExtensibility2.OnAddInsUpdate
        End Sub

    Public Sub OnStartupComplete(ByRef custom As System.Array) Implements
Extensibility.IDTExtensibility2.OnStartupComplete
        End Sub

    Public Sub OnDisconnection(ByVal RemoveMode As Extensibility.ext_DisconnectMode, ByRef
custom As System.Array) Implements Extensibility.IDTExtensibility2.OnDisconnection

        m_applicationObject = Nothing
        o_addInInstance = Nothing

        System.GC.Collect()

        MsgBox("Goodbye MindManager Development Universe!")

    End Sub

    Public Sub OnConnection(ByVal application As Object, ByVal connectMode As
Extensibility.ext_ConnectMode, ByVal addInInst As Object, ByRef custom As System.Array)
Implements Extensibility.IDTExtensibility2.OnConnection

        m_applicationObject = CType(application, Mindjet.MindManager.Interop.Application)
        o_addInInstance = addInInst

        MsgBox("Hello MindManager Development Universe!")

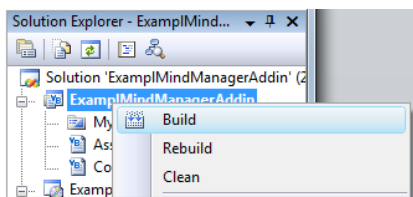
    End Sub

End Class
```

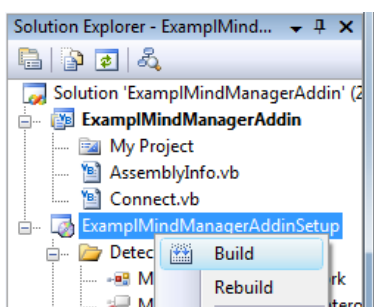
## Step 5

Now we need to build the Project.

Right-click the **Project** in the **Solution Explorer** and select **Build**.



Check that the build action is successful and then repeat for the **Set Up Project**.

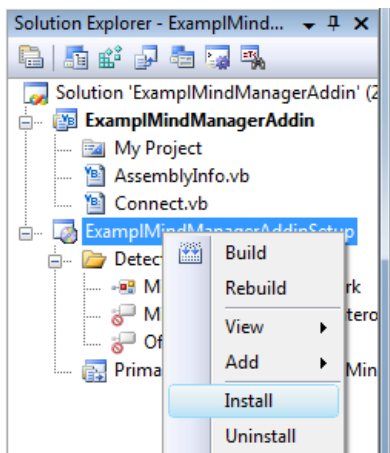


Again, check that the build action completes without any errors.

## Step 6

We now need to **Install** the add-in.

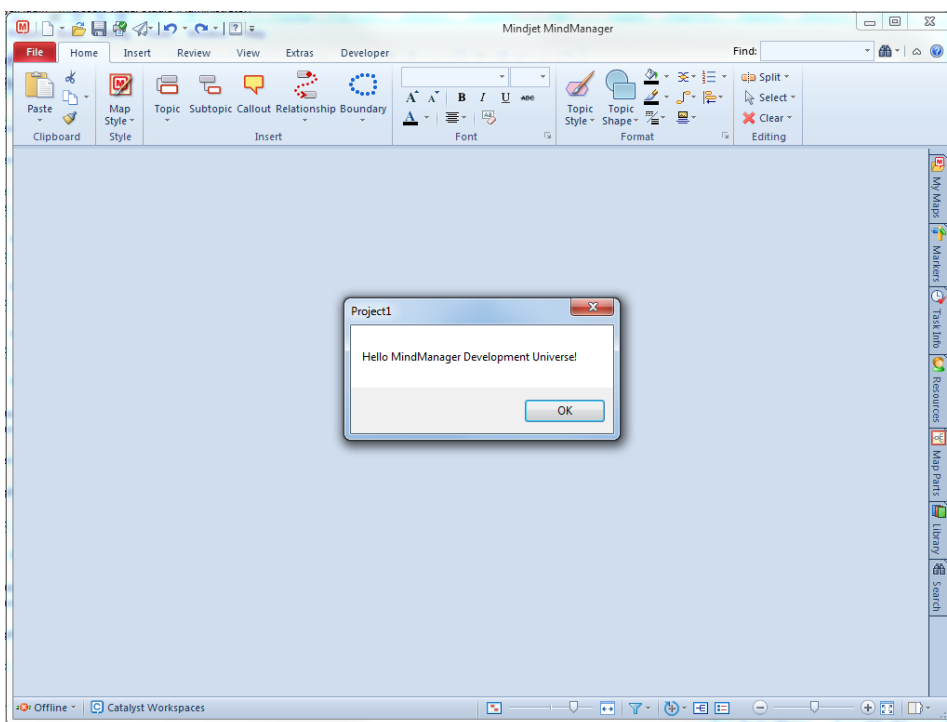
Right-click the **Set Up Project** from the **Solution Explorer** and select **Install**.



Run through the install screens selecting the default options.

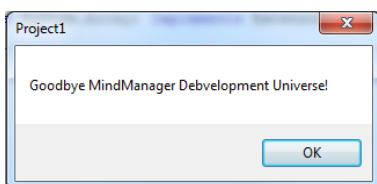
## Step 7

Fire up MindManager.



If everything has gone ok you should see the **Message Box** pop into view from the **On Connection Sub**.

Exiting MindManager should display the **Message Box** from the **On Disconnection Sub**.



I find it useful to have this **On Disconnection** message as it helps detect when MindManager has not closed properly which can happen without notification. If you close MindManager and don't see your add-in disconnect then something is wrong and MindManager will probably be still running in the background.