

Digimaker 5.2 to Digimaker 6.0 Migration Guide.

Upgrading Digimaker 5.2 based web sites to Digimaker 6.0 is a **3 step** process. Follow the below mentioned procedure carefully to upgrade your web site to Digimaker 6.0.

Note: It is mandatory to follow the 3 steps in the exact order mentioned in this document for a smooth upgrade to Digimaker 6.0. You are also recommended to create a backup of the websites you intend to update, including the database.

Prerequisites

Before you can convert your application you will need to make sure that:

1. You have access to **Microsoft Visual Studio 2005**.
2. Your computer and target servers have the **Microsoft .NET Framework 2.0** installed and updated.
3. You have verified that all your existing **Digimaker 5.2** based websites are working correctly.

Step 1: Migrating an ASP.NET 1.1 web site to ASP.NET 2.0

To convert your ASP.NET 1.1 web site to ASP.NET 2.0:

1. Start the **Microsoft Visual Studio 2005 IDE**. Visual Studio 2005 has a built-in conversion wizard that will help you convert your ASP.NET 1.1 web site to ASP.NET 2.0. This wizard automates many of the basic steps necessary to make your application meet the new structural and coding requirements built into ASP.NET 2.0.



Figure 1: The Conversion Wizard

2. The conversion wizard is automatically invoked whenever you open an ASP.NET 1.1 Web application in Visual Studio 2005. The wizard detects the presence of a project file (e.g. *.vbproj or *.csproj) in the application directory and automatically starts the conversion process.

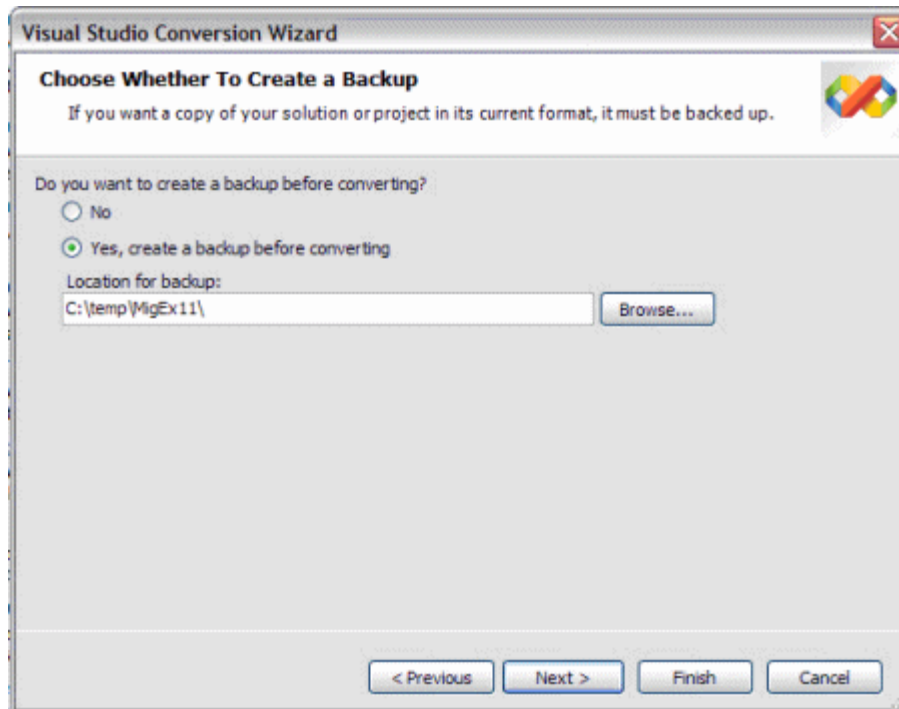


Figure 2: Backup your application

- 3 The first choice you have to make is whether to perform an in-place conversion or to create a backup of your application before conversion.

If you choose to create a backup, Visual Studio 2005 will automatically create a copy of your ASP.NET 1.x application in the directory of your choice.

Note: Please make a backup of your application, this will allow you to roll back changes if at all the conversion runs into problems.

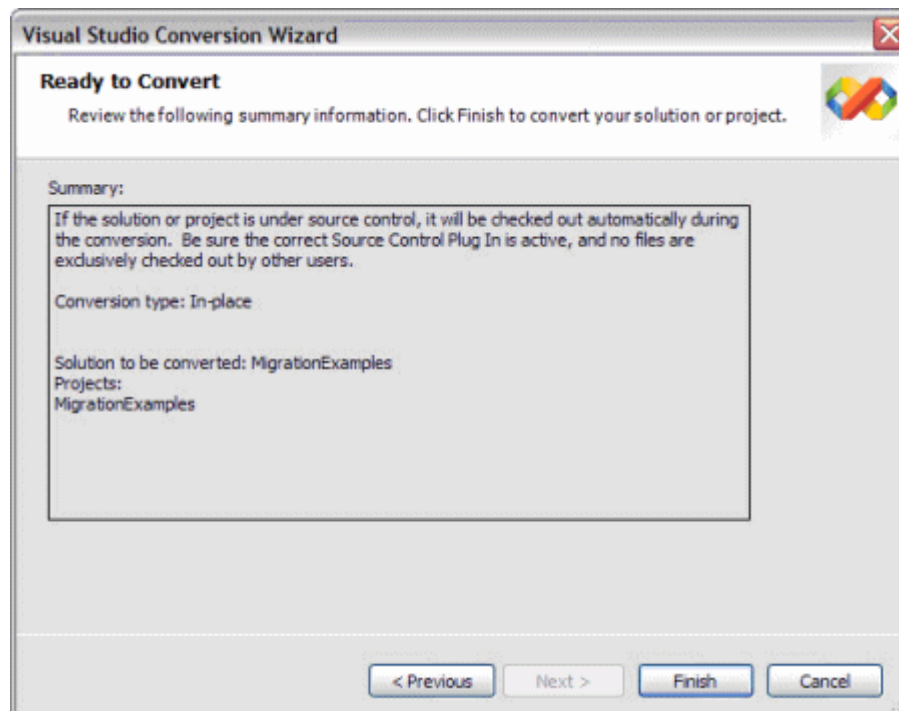


Figure 3: Summary screen

- 4 Next, you will see a summary screen of the conversion process and have one last opportunity to back out of the conversion.

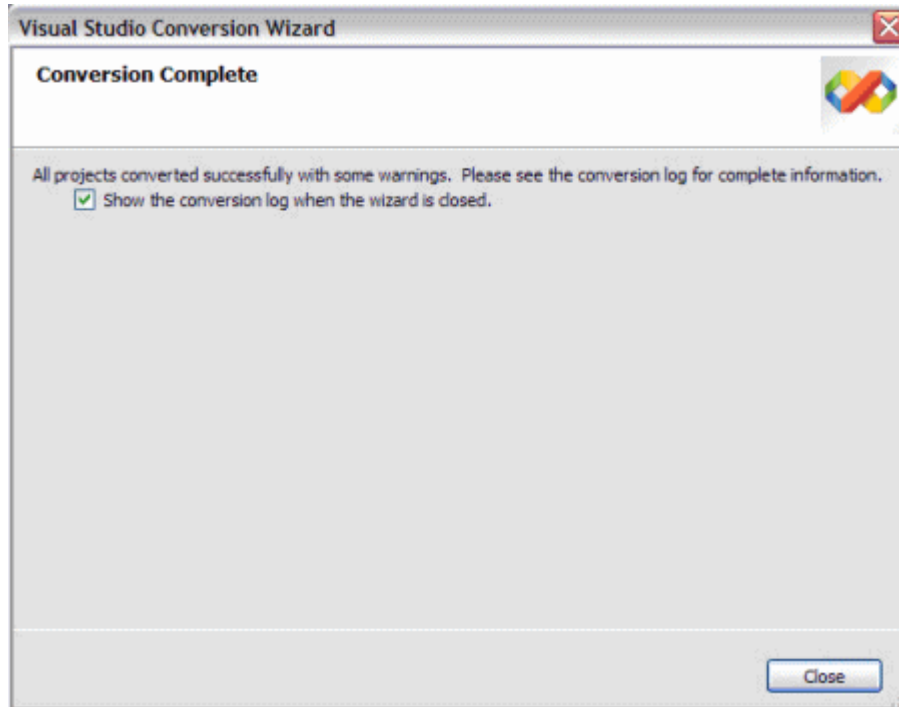


Figure 4: Conversion completed

- 5 The conversion may take a few minutes, depending on the size of your application. However, when it completes you will see a message indicating that your code was converted.

Conversion Report

You may also see a message about some warnings or errors. Warnings and errors occur when the conversion wizard made changes that may modify the behavior of your application, or when it can't completely update your application to ASP.NET 2.0.

After the conversion is complete, you are ready to look at the conversion report to see if you have to perform any additional steps to complete your conversion from ASP.NET 1.1 to 2.0. When the conversion wizard finishes upgrading your project, it automatically generates both an XML and a text version of the conversion report before displaying the XML version. This report will show you any issues encountered by the wizard and code areas where you may need to take additional steps to complete the conversion.

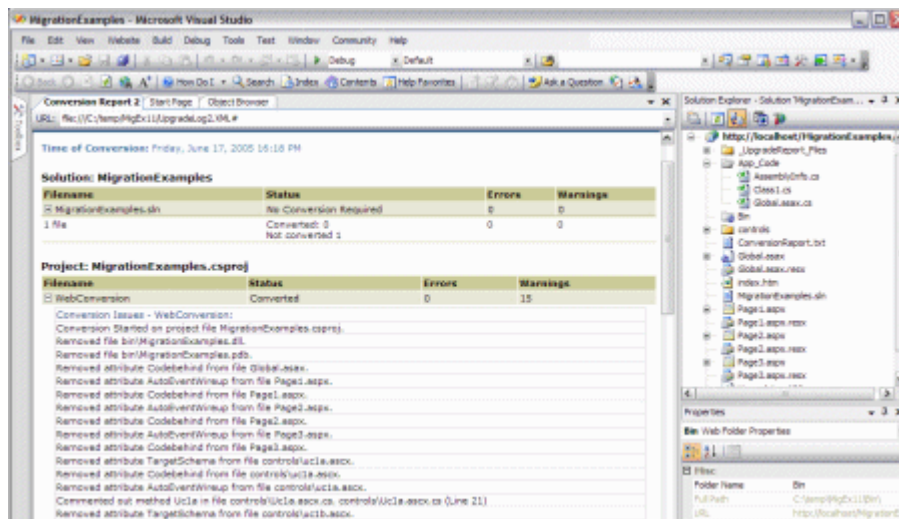


Figure 5: Conversion report

The report is divided into sections for each solution and project that was converted. The solution report will almost always be error free. However, the project sections may have multiple issues listed for each file in your project. If you close the conversion report, you can always find a text version at the top level of your converted project.

Notification Types

Each item in the report falls into one of three categories:

Notification - A notification item simply informs you of an action taken by the wizard. You will see many notifications about files that were deleted or moved, and code that was deleted or commented out. As you'll see in the following sections, the wizard performs certain standard operations on every file. These operations are necessary for the conversion but do not require any additional effort from you.

Warning - A warning item is generated whenever the wizard has to take an action that may cause a behavioral change in your application. Warnings are items you want to review, but may not need to act on. For example, if the wizard has to change access level on a piece of code, you will see a warning. You should review the change to make sure it doesn't have any unintended consequences. Generally, however, you can simply review warnings and then ignore them.

Error- An error item is generated if the wizard encounters something that cannot be automatically converted. These items require your effort to complete the conversion. Generally, an error is something that will generate a compilation error if you try to run the application.

Keep in mind the conversion report is a log file describing the changes made to your Web project. Most of the notifications can be ignored unless you are really curious about what changes were made to your Web application. You should review errors and warning first since these items indicate places where you may need to change your code to complete the conversion.

Converting an application from ASP.NET 1.x to ASP.NET 2.0 is generally a smooth process. However, you have to make sure that your development and deployment environments are properly configured. You also have to evaluate your conversion report to make sure that your converted application will function properly. You may also wish to review your application ahead of time and plan ahead to avoid known issues with the conversion.

Step 2: Upgrading the web site to Digimaker 6.0

The Digimaker server manager automates the process of converting web sites running on previous versions of Digimaker, to Digimaker 6.0.

To upgrade a website,

1. Start Digimaker Server Manager and right click on the desired web site node under the **Websites** tab.
2. Select **All Tasks > Upgrade Site** from the pop-up menu. This displays the **Digimaker Upgrade Utility** dialog b

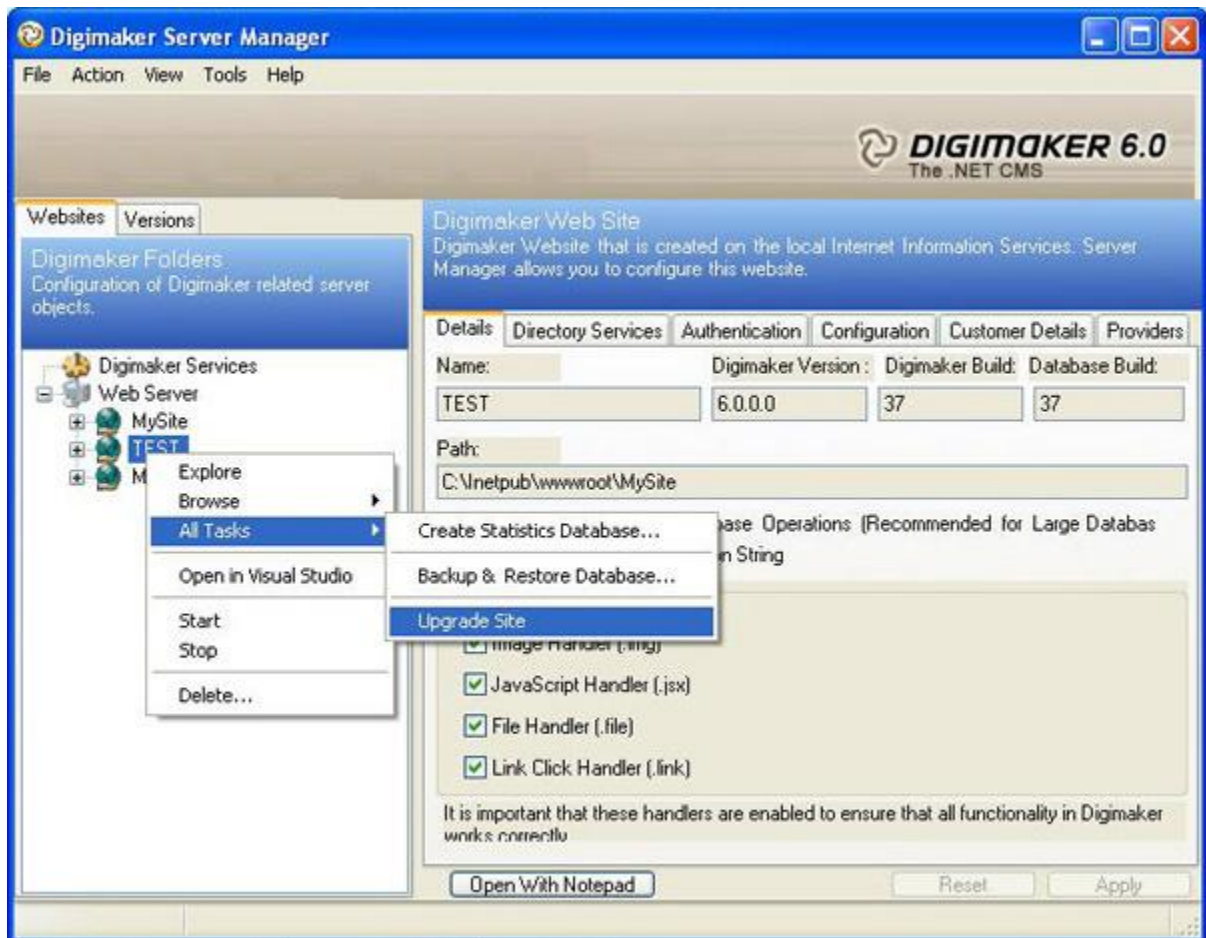


Figure 6: Upgrade Site

3. Upgrading can be done in phases. Check all the options to do a complete upgrade.

The Digimaker Upgrade Utility allows you to manually choose the upgrade setting, which can be used to segregate the operations based on the settings as well as hosting environment

EX: When your website is running live on a live server, and you want to test an upgrade, you don't want to risk the live site with the upgrade. So you can copy the website into the staging server and make the upgrade and test it.

4. Click the **Upgrade Now** to upgrade the web site to Digimaker 6.0.



Figure 7: Upgrading a website

The upgrade process performs the following tasks:

Upgrade Database

When this checkbox is checked and you click on the “**Upgrade Now**” button the web site database is automatically updated. i.e. running upgrade script.sql and patchscript.sql on the database server.

Backup Configurations

When this checkbox is checked and you click on the “**Upgrade Now**” button the **Digimaker.config** and **Web.config** files for the web site is backed up in the physical folder of the website as Digimaker_backDigimakerUpgrade.config and Web_backDigimakerUpgrade.config.

Upgrade Configurations

When this checkbox is checked and you click on the “**Upgrade Now**” button the assembly versions are modified in the Digimaker.config and Web.config file. In addition the newly introduced keys are automatically added into the configuration files. **Ex:**

- Appropriate entries are made in the Membership section.
- The WebResources section in digimaker.config is modified.

Register ASP.net with IIS

When this checkbox is checked and you click on the “**Upgrade Now**” button the site is registered with the appropriate version of ASP.NET in IIS.

Encryption Connection String

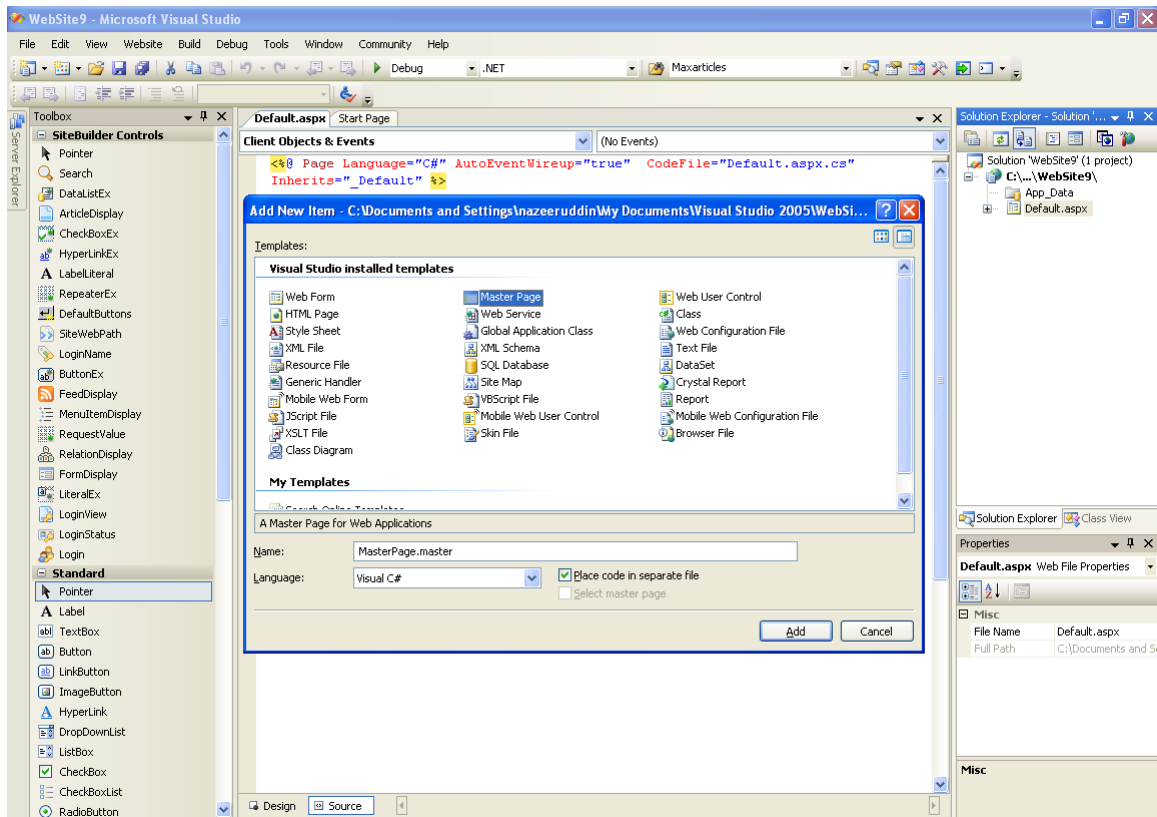
When this checkbox is checked and you click on the “**Upgrade Now**” button the connection-strings in Digimaker.config are encrypted.

Step 3: Converting the master page model

The last and final step in converting your web site to work with Digimaker 6.0 is upgrading from the Digimaker ASP.NET 1.1 master page model to ASP.NET 2.0 master page model.

To Convert the web site from the Digimaker ASP.NET 1.1 master page model to the ASP.NET 2.0 master page model,

1. Create a new master page in ASP.Net 2.0 using Visual Studio 2005. In Visual Studio 2005 IDE, choose **Add New Item** from **Website Menu** and select **Master Page** from the **Add new Item Dialog box**. A new master page will be created with the default name of MasterPage.master.



2. Now open the master.ascx file and copy the content leaving **@Control** directive as it is and paste it into the MasterPage.master page created in the previous step.

Example:

This is the content of the master.ascx file

```
<%@ Control Language="c#" Inherits="DigimakerDemo.Master" CodeFile="Master.ascx.cs" %>
<%@ Register TagPrefix="dm" Namespace="Digimaker.Web.Controls"
Assembly="Digimaker.Web.Controls" %>
<%@ Register TagPrefix="sb" Namespace="SiteBuilder.WebControls"
Assembly="SiteBuilder.WebControls" %>
<%@ Import namespace="SiteBuilder" %>
<%@ Import namespace="SiteBuilder.WebControls.Page" %>
<%@ Import namespace="SiteBuilder.WebControls.ContentToolkit" %>
<html>
  <dm:head>
    <title>.....</title>
  </dm:head>
  <body>
    <dm:form>
      .....
    </dm:form>
    ....
  </body>
</html>
```

So leaving the @Control directive

```
<%@ Control Language="c#" Inherits="DigimakerDemo.Master" CodeFile="Master.ascx.cs" %>
```

Copy all the content in Master.ascx and paste it into MasterPage.master, keeping the @ Master directive intact.

```
<%@ Master Language="C#" AutoEventWireup="true" CodeFile="MasterPage.master.cs"
Inherits="_MasterPage" %>
```

3. From the master.ascx.cs file copy all the code into the MasterPage.master.CS file.
4. Replace all occurrences of the Digimaker ContentPlaceHolder control, <dm: ContentPlaceHolder> tag with the <asp: ContentPlaceHolder>.

Example:

```
<dm:ContentPlaceHolder id="additionalHead" runat="server">
-To-
<asp:ContentPlaceHolder id="additionalHead" runat="server">
```

5. Do not change <dm: head> and <dm: form> since the page base and some other classes depend on these.
6. There's a new attribute "MasterPageFile" in the @Page directive, which is used to specify the master page file to be used by the Page. Add this attribute to point at the MasterPage.master file created in the step above.

Example:

```
<%@ Page language="c#" MasterPageFile="~/MasterPage.master"  
Inherits="DigimakerDemo._default" Debug="true" CodeFile="default.aspx.cs"  
CodeFileBaseClass="SiteBuilder.WebControls.Page.PageBase" %>
```

7. Remove the Digimaker master page control start tag <dm: masterPage> and end tag </dm:masterPage> from the aspx page.

Note: Don't remove the contents inside the Digimaker Master Page control.

8. Replace all occurrences of Digimaker Content control, <dm: Content> tag with the <asp: Content> tag.
9. Replace the "id" attribute of the Digimaker Content control to "ContentPlaceHolderId".

Example:

```
<dm:content id="additionalHead" runat="server">  
-To-  
<asp:content ContentPlaceHolderid="additionalHead" runat="server">
```