Before beginning any application you should have certain standards in place. These include programming standards as well as development environment standards. By using good development standards, all programmers will know what is expected of them and how to create new applications from scratch. In addition, these standards help developers move from one project to another without having to learn how different programmers created a particular application.

## Computer Setup

It's important that you configure your development system appropriately. If you're working with multiple developers, this is especially important, as you want all computers to be configured similarly.

We've found that it's crucial for developers to be able to rebuild machines, from scratch, with a minimum of effort. This means that it's imperative that data be stored separately from the operating system, so you can "blow away" the operating system at any time without losing data.

We suggest the following strategy, which has worked well for us:

* Create a partition just for the operating system. This partition should be at least 150+ gigabytes. This partition will be your C: drive.
* Install an appropriate operating system (Windows 7, Windows 8, Windows Server 2012, etc.) onto your C: drive.
* Once Windows is installed and running, rename your CD-ROM (or DVD, or other optical drive used for installations) to a "high" drive letter (we use X:), so that other installed or temporary (USB) drives won't move the drive letter for the optical drive later on.
* This can be a separate physical drive or a logical partition.
* Create one more partition for your “D” drive. This is where you will store all your projects and other data.
  + Having a separate partition makes it very easy to backup all your data from this one drive to an external USB drive or to a network drive.
* Install all your standard applications. Make sure that any personal data files (such as an Outlook PST or OST file) are stored on the D: drive. (We suggest that you also set your My Documents folder to point to a folder on the D: drive.)

**NOTE**: You might also consider using VM Ware or Virtual PC to create a standard development environment. This makes it easy to use undo disks and to try out various environments without having to mess up your main machine.

## Developer Standards

Of course you are going to need lots of various software for developing with .NET. You will most likely need the following as a minimum.

* A version of Windows that supports Visual Studio
* The .NET Framework with any service packs applied
* Visual Studio 2010/2012 or later
  + Be sure to install any service packs
* Source Code Control software such as TFS
* SQL Server 2008/2012 or later
* PDSA .NET Productivity Framework
* Haystack
* PDSA Developer Utilities

### Create a Default Project Location

You should make sure that all developers create their projects under the same folder on the same drive letter to help people move their projects back and forth to each other. For example, you may decide that all programmers will create a folder on their **D** drives named \DotNetProjects (D:\DotNetProjects). This keeps everyone's project location consistent, which helps as they move projects back and forth between machines.

## Source Code Control

Use it! You will find that not only will it help you with the versioning of your changes, but it will also help you set up a new developer's machine with the correct virtual directory for your web projects, or just the correct directory structure for your other projects. The best way to start a new application is to have a single developer (typically the lead programmer for the project) create a new project on their machine. The lead programmer will then proceed to create the appropriate folders, Windows Forms, WPF windows, Web Pages, etc. for the application. When they have enough of the forms/pages created that they are ready to start assigning work to the rest of the team, they will then check in the entire project into your source code control system. This will allow the rest of the team to get the project from the source code control system and have it automatically set up the project on their machine.

#### Source Code Control for the Single Developer

Obviously source code control is essential for multi-developer projects, but you will also find that even for a single-developer project it can be a great productivity enhancer. It helps you keep track of what you have worked on and when, and helps you maintain a backup of all the code on another machine (assuming you install the source code control database on another machine). It also helps you roll back to a previous version of the code that you checked in. It takes a little more discipline to use it when you are working alone, but you will find the benefits far outweigh the disadvantages.

## Visual Studio Install

When installing Visual Studio, please select the C# development environment settings. If you have already installed Visual Studio, you can reset this by going to Tools | Import and Export Settings…

Select **Reset all settings** from the first dialog as shown below:

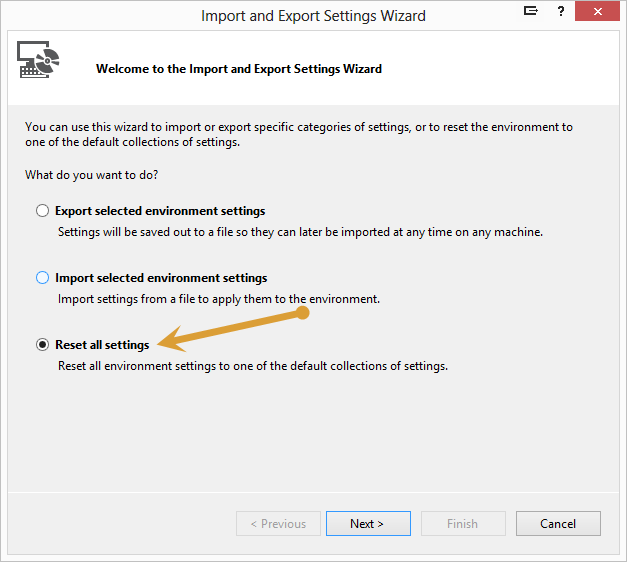


Figure . Reset all settings

Click the Next button

You should save your current settings in case you wish to go back.

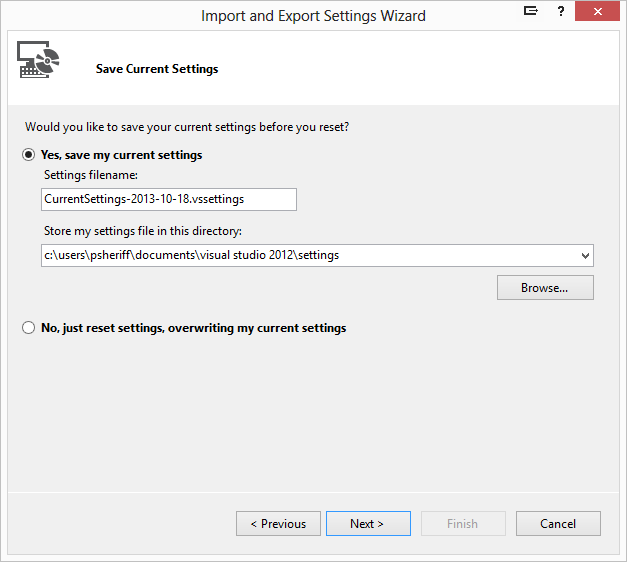


Figure . Save all your old settings (just in case)

Click Next

Choose the **Visual C# Development Settings** from the next list.

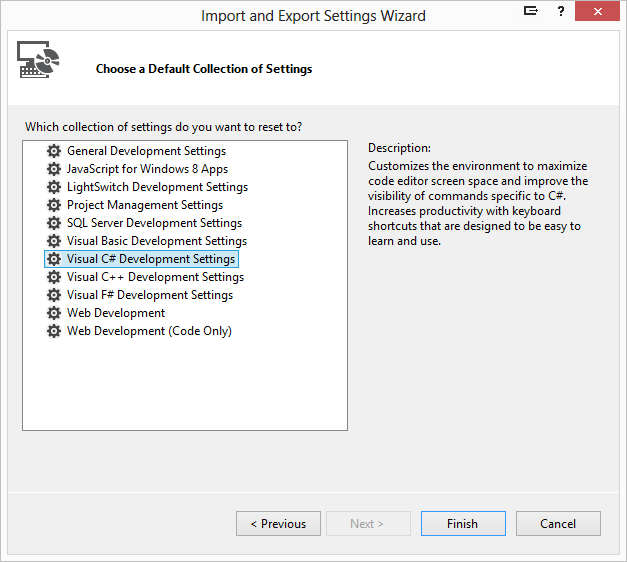


Figure . Select Visual C# Development Settings

## Setting Visual Studio.NET Options

Before you can begin coding, you'll need to set up your computer and Visual Studio .NET to make best use of your time and the tools. The next few sections discuss the details of choosing important Visual Studio .NET options. It is also very important that all developers choose the same options.

Visual Studio .NET includes many options that define the behavior of the environment and the features in new projects. You'll want to make sure that developers within your organization have set up their environments in the same way. The following sections highlight some settings within the Options dialog box that you will want to consider setting for each developer. (Use the Tools|Options menu item to display this dialog box.)

**NOTE**: Sometimes the options presented below change from one version of Visual Studio to another, so use the following as a guideline.

### Projects and Solutions | General

Set the **Visual Studio projects location** property to the location where you will be creating all your Visual Studio projects, for example D:\DotNet20Projects. This will ensure that the SLN file is located in the same location as your project files.

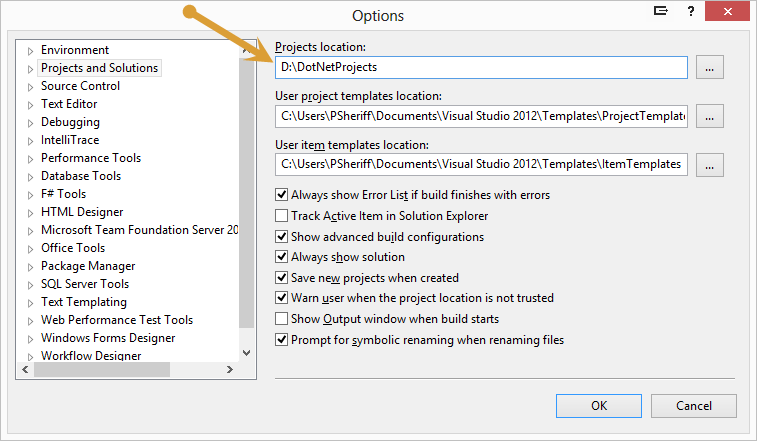


Figure . Environment | Projects and Solutions | General

### Projects and Solutions | VB Defaults

If you are using VB.NET, you should ensure that the Option Explicit and Option Strict setting are set to **On**.

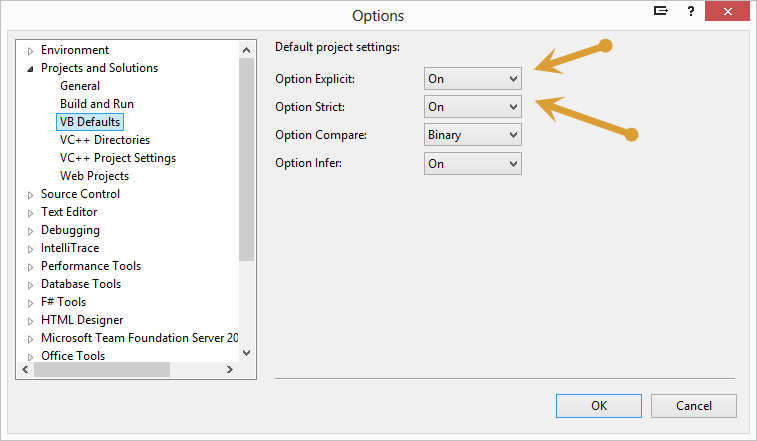


Figure . VB Defaults.

### Text Editor | All Languages | General

You should uncheck the Hide advanced members check box so the IntelliSense list displays all values.

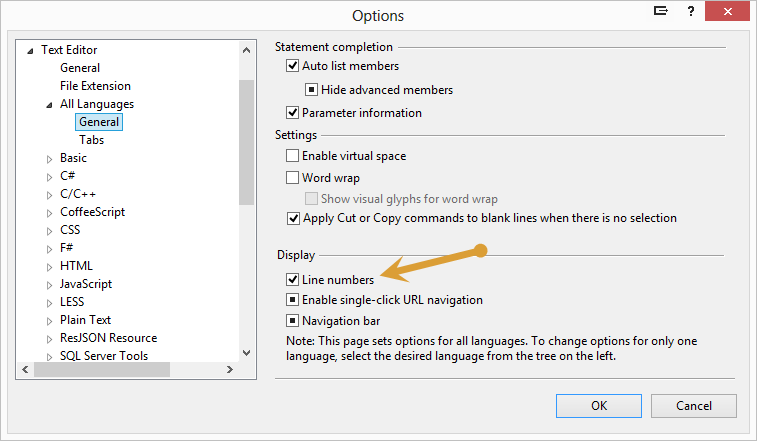


Figure . All Languages | General

### Text Editor | All Languages | Tabs

This tab is very important that all programmers in your group set consistently. If you don't and someone checks out a file and reformats the code according to their settings, then upon check in, the history of that file will show that all lines changed. This makes it very difficult to track what changed from one version of to another.

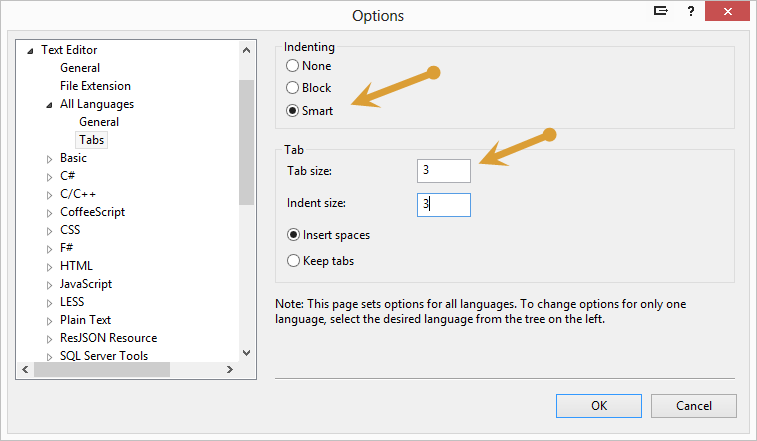


Figure . All Languages | Tabs

### Text Editor | HTML | Validation

Most browsers support HTML 5 today, so it is best if you set the Visual Studio editor to validate based on HTML 5 rules.

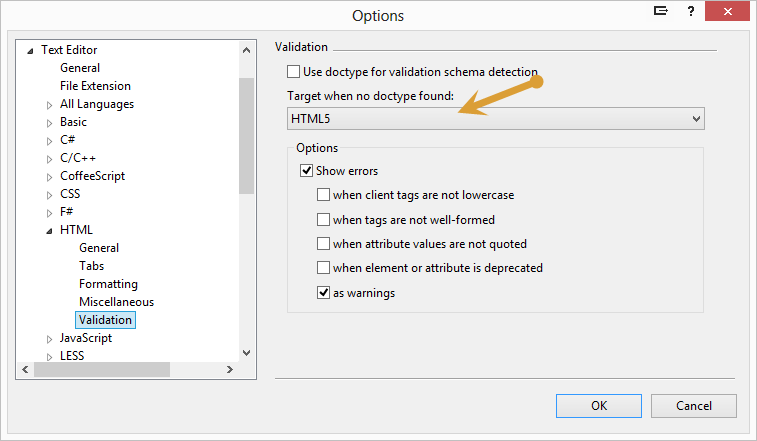


Figure . HTML | Validation

### Text Editor | XAML | Formatting | Spacing

Make your XAML more readable by positioning each attribute on a separate line and remove empty lines in your content.

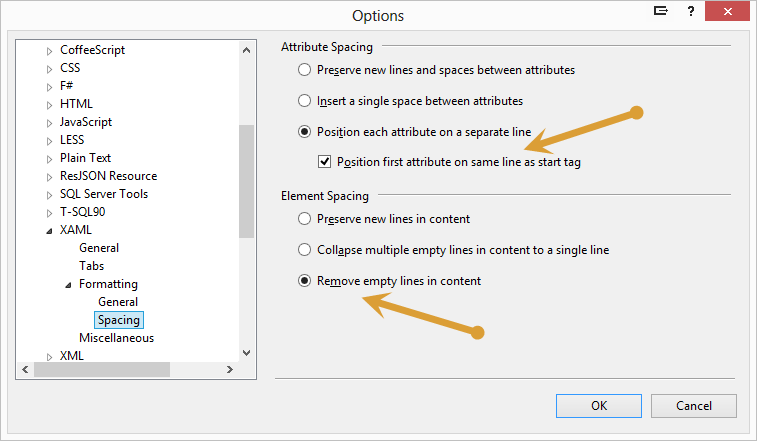


Figure . XAML | Formatting | Spacing

### Text Editor | XML | Formatting

Make your XML more readable by positioning each attribute on a separate line and remove empty lines in your content.

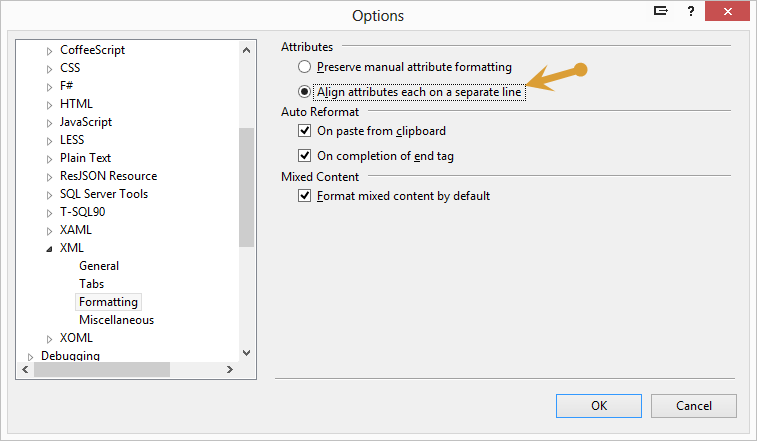


Figure . XML | Formatting

## Structuring your Projects

There are generally four types of classes for any given application you create.

* Classes that can be re-used in any application. We call these Generic Classes.
* Classes that are used in only one type of application; ie. Web Forms, MVC, WPF, Silverlight, Windows Forms, etc. We call these UI Specific Classes.
* Classes that are re-used only within one application. We call these Application Specific Classes.
* Classes that are only used once within an application. We call these Application Classes.

As such, you need to design a folder structure on your hard drive that reflects this type of application framework.

### Generic Components

You should always have your own application framework (like our PDSA .NET Productivity Framework) where you keep all of your generic components that you use for all of your applications. All of these components/classes should be grouped logically into DLLs according to their function. For example, all classes that work with files you might put into a File DLL and a File Folder. All classes that are used for WPF, you might put into a WPFLibrary DLL and folder. Figure 11 shows an example of a generic set of class library projects. These are from the PDSA .NET Productivity Framework. This is only a small portion of the various libraries that we have that we can use across many applications.

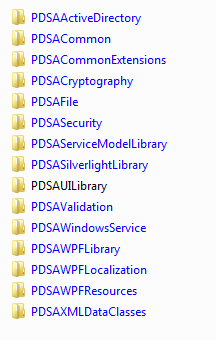


Figure : Example of a Generic Framework Structure

Summary

This chapter introduced you to a standard configuration for your development computers as well as some basic information about Solution and VB or C# Project files. Setting up your environment is very important if you want to get off to a good start developing a .NET application. In this chapter you learned about the various settings that should be consistent among the developers in your shop.