

Chapter 1

Getting Started with VSTO

In This Chapter

- ▶ Figuring out what you need to get started
 - ▶ Picking a version
 - ▶ Installing VSTO
 - ▶ Building your first Office applications
-

Visual Studio Tools for Office is exactly what it sounds like it is — a set of tools that is part of Visual Studio and designed to enhance Office. In less politically correct terms, VSTO is Microsoft's answer to those who want to use the more robust .NET to get VBA-like functionality in their Office development.

VSTO has a lot of parts, does a lot of loosely related things, and means different things to different people. For that reason, we talk about what VSTO can do for you before we get into a little code in Chapters 2 and 3. In this chapter, we look at logistics with a tour through versions, options, and languages.

Harnessing the Power of VSTO

Visual Studio Tools for Office isn't a replacement technology for anything currently in the Microsoft pantheon of applications. VSTO doesn't replace Visual Basic for Applications, it doesn't replace scripting Office applications, and it doesn't replace Windows or Web forms.

VSTO is a set of tools that you can use with Visual Studio to supercharge Office. Put simply, VSTO provides tools to build add-ins and Customized Documents for Office applications like Word and Excel. That simplicity hides a wealth of power and functionality.

Talking about add-ins and Customized Documents

Add-ins, like the one in Figure 1-1, are little bits of programs that are managed by Office. You probably use add-ins all the time without even knowing it. If you have Adobe Acrobat installed and have the Save To PDF button in your toolbar, you've seen an add-in. OneNote puts an add-in in Office to allow you to move documents between Word and a Notebook, for example.

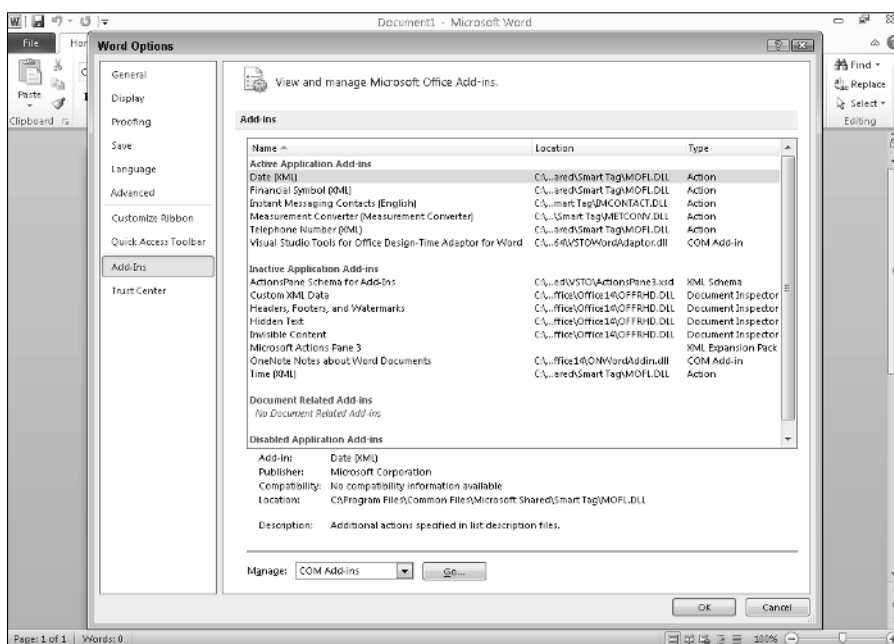


Figure 1-1:
Add-ins in
Word.

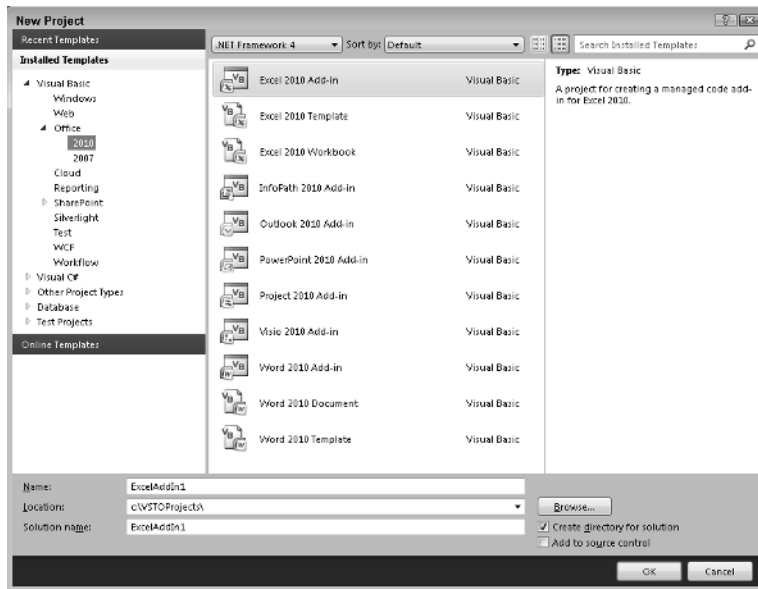
From the programmer's point of view, add-ins are just Windows applications that have a special home — an Office application. You can use VSTO to build add-ins, just as you could use Visual Basic 6 and other programming applications before VSTO. Add-ins have been around a long time. VSTO allows you to use managed code (for example, the .NET Framework) for the first time.

Customized Documents are new and take more explanation (see Figure 1-2). Customized Documents are regular Office documents (like an Excel spreadsheet) that have a .NET DLL associated with them that give them special powers. You can use code in the back-end of the document to fill in fields from a database, validate data, or respond to certain user requests.



VSTO is very different from what you're used to. It's not script code in a macro or in a Visual Basic for Applications project. After you make a document a Customized Document, it's a compiled project, just like a Windows application. VBA is also part of the document, while the code from a VSTO project is in a DLL linked to the document.

Figure 1-2:
Making
a new
Customized
Document
in Visual
Studio.



Customized Documents are cool but hard to use in the wild. Peter shows you an example in Chapter 3, and we also build more complex Customized Documents in Chapters 6 and 8.

At their core, Customized Documents are document-level only, while add-ins are application-level and can apply to all documents used by that installation of Office. They both use more or less the same path to get where they're going: Visual Studio Tools for Office.

Solving your problems with four VSTO features

Four important features are available in VSTO. By *features*, we're talking about the things that you need to remember when you're considering using

this toolkit to solve your problems. VSTO allows you to do these four things to solve your problems:

- ✓ Add your custom functionality to existing Office applications, via add-ins.
- ✓ Leverage all the existing Office functionality in your applications, like Word document layout and Excel formulas, through communicating with those add-ins.
- ✓ Make documents that integrate fully with existing applications and databases.
- ✓ Use Microsoft SharePoint to increase communications between knowledge workers with add-ins and documents that communicate.

The following sections dig into each feature in more detail.

Add functionality to Office

PowerPoint does a lot when it comes to presentations. If you want to automatically generate sample images from the live photo repository in your company, though, you're in a tough spot. The good news is that VSTO add-ins can fix that issue.

Using an add-in, you can get the latest images from the project repository and make them available to a presentation author in real time. This add-in can provide them with the latest images for status reports and sales presentations. Without an add-in, you're back to searching through the Q drive, or whatever, and different people solving the problem different ways.

When you're building a solution for a company, automating common tasks is among the most common requests. Add-ins allow you to use Windows form elements, the Action or Task pane, and Smart Tags to do things specific to your company. What's more, these tools are available to every document that is created by the computer with the add-in installed.

Leverage Office functionality

Another consideration is the ability to leverage Office as an extension to your development platform. For example, while you can build pivot tables into your ASP.NET based reporting system, doing so takes a long time or requires the use of expensive third-party products. Or, you can use VSTO.

How, you ask? Just have your Reports button launch Excel with your custom add-in. Then users can get the data they want and then work with it using the tools they already know. Your Excel add-in provides a connection to the database, and the built-in security model ensures that your confidential data is safe.

Making good software quickly is the expectation of independent developers and IT departments alike these days. Leveraging functionality already purchased, and knowledge already gained, makes that goal easier to reach.

This ability sort of crosses over the two project types (documents and add-ins) in that both of those types can leverage different functionality. Documents leverage document level functions, and add-ins leverage the application level tools.

Make documents that think

Applications aren't the only piece of the puzzle. A Word letter template that knows how to check the Exchange server for names and addresses will make communication that much easier. And VSTO has just the ticket.

Customized Documents allow you to do just about anything in an Office document that you can do in a Windows application or in an old VBA project. Accessing databases, making calculations, and checking Web Services are all possible. Fields in a Word document can easily be bound to a data source using ADO.NET data binding.

Thinking out of the box is a core skill in information technology, and Customized Documents are right in there. While getting the ideas under your belt takes a while, you'll be surprised what you can do after you get a model for their use. Then, after you get the tool the way you want, it's mobile — transferable from installation to installation (at least in theory). Customized Documents are just that — documents. The logic follows the document itself from computer to computer under controlled circumstances.

Make connections between existing applications and Office

"Sure, we can get their sales data. Some of it is in this Excel spreadsheet, some in Outlook Business Contact Manager, and the rest in the Oracle ERP database that we access with this InfoPath form." VSTO to the rescue. Tying together Office applications has never been easier.

It doesn't matter how you look at it or what models you use, VSTO can make it so that your Office applications sing in unison through SharePoint. A Customized InfoPath form that knows where to go to get to the Excel data and the Outlook add-in will do the trick here. No matter what the combination, it's better than writing reams of VB Script.

Convergence is where it is at, too. Getting the disparate applications of the contemporary IT department to not only play nice but to be accessible by the knowledge workers is something special, and VSTO can make it oh-so-much smoother.

Exploring the Different Versions of VSTO

VSTO runs in a cycle that is between Office releases and Visual Studio releases, so sometimes it's tough to tell what version does what. We clear up any confusion in this section.



In this book, we cover only the version of VSTO that is part of Visual Studio 2010 and works with Office 2010: VSTO 4.0. Much of what we do in this book applies to many other platforms, except deployment, which is quite different. Due to the deployment story, I encourage you to use Visual Studio and Office 2010 whenever possible.

Visual Studio 2003 and 2005

Visual Studio Tools for Office weren't part of the default Visual Studio releases — instead, a separate product called Visual Studio Tools for Office was available. This product contained tools used in developing for Microsoft Office platform and was a complete edition (called SKU) of Visual Studio.

VSTO 2003 supported only Office 2003 and document-level customizations for Word and Excel. The next version, called VSTO 2005, included support for Outlook add-ins.

VSTO 2005 SE

Moving forward to 2005 when a new version — the so-called second edition of VSTO — was released. VSTO became part of Visual Studio 2005 Professional and higher SKUs, and the separate SKU for VSTO was removed from the lineup.

At this time, Microsoft released Office 2007, which included support for developing add-ins and document-level customizations. VSTO 2005 SE for Office 2003 also added support for application-level add-ins for Office products other than Outlook: Word, Excel, PowerPoint, and Visio.

Support for Office 2007 started with this version of VSTO as well, and you could create document-level customizations for InfoPath and add-ins for Word, Excel, Outlook, PowerPoint, Visio, and InfoPath. Note that in VSTO 2005 SE, you couldn't create Word or Excel document-level customizations targeting Office 2007 — that came in the next version.

VSTO 3.0

With VSTO 3.0, nothing changed if you were developing for Office 2003. VSTO 3.0 is built into Visual Studio 2008 Professional or higher SKUs. If you developed for Office 2007, you now had support for Word and Excel document-level customizations. In application-level or add-in world, VSTO 3.0 included support for Office Project. Apart from focusing only on Office client applications, Microsoft decided to add a project template to target Office Server System. Project templates for SharePoint 2007 workflow projects were added in this version in order to support Microsoft Office SharePoint Server 3.0.

Visual Studio 2010 and VSTO 4.0

The biggest feature for VSTO 4.0 in Visual Studio 2010 is support for 64-bit machines. Compared to previous versions, VSTO doesn't include any new project templates. If you think about it, there's not much the programmers could add to VSTO. With Visual Studio 2010, the focus of VSTO switched from supporting Office Client applications to SharePoint support.

Visual Studio 2010 contains a large set of project templates for developing against SharePoint 2010 (see Part III). Apart from SharePoint workflows, Visual Studio gained support for the following projects:

- ✓ Workflows
- ✓ List Definition
- ✓ List Instance
- ✓ Site Definition
- ✓ Content Type
- ✓ Module
- ✓ Empty SharePoint Project

Office 2003

Using the Shared Add-In project type, you can create Office 2003 add-ins with just a default install of Visual Studio of any version. This project type isn't VSTO; none of the libraries that I talk about here are available. The Shared Add-In project type uses the default COM Standard Add-In hooks and everything else is up to you.



COM is the Component Object Model that Office still uses. It is the predecessor of .NET, and still in use in a lot of systems, actually. C++ is the usual programming language of choice. We don't have to use C++, though, to use the add-in hooks provided by VSTO because all the classes are provided to us! Yay!

If you want to produce application-specific add-ins and Customized Documents for Office 2003, then you have to have a version of VSTO. How much power you get depends on what version you run:

- ✓ VSTO Version 2003 includes customized document projects for Word and Excel.
- ✓ VSTO Version 2005 adds add-in projects for Outlook, Word, and Excel.

- ✓ VSTO Version 2005 SE (found in Visual Studio 2008) further supports add-in projects for PowerPoint and Visio.
- ✓ VSTO Version 4 (found in Visual Studio 2010) adds features for Office development.

If you have a contemporary MSDN Professional subscription, the sum total accessible to you when working in Office 2003 includes

- ✓ Customized Documents:
 - Word 2003
 - Excel 2003
- ✓ Add-ins
 - Word
 - Excel
 - Outlook
 - PowerPoint
 - Visio

Office 2007

Office 2007 has a new user interface and a new file format for all office documents. VSTO 2003 and VSTO 2005 don't support Office 2007 at all. VSTO 2005 SE adds add-in support for InfoPath, as well as the following:

- ✓ Word
- ✓ Excel
- ✓ Outlook
- ✓ PowerPoint
- ✓ Visio
- ✓ InfoPath

The document support for Office 2007 is found in Visual Studio 2008 and Visual Studio 2010 and includes Word and Excel.

What there is in Office 2007 — if you're running VSTO 2005 SE or Visual Studio 2008 — is support for some of the cool new Office user interface features:

- ✓ Custom task panes
- ✓ The Ribbon, in the form of Ribbon Extensions
- ✓ Outlook Form Regions

Earlier versions

Running Office XP or earlier? Upgrade. Office XP isn't supported, though interop assemblies are available for non-VSTO solutions using Visual Studio. You can also use a shared add-in because they speak COM. Nonetheless, VSTO add-ins don't run in Office XP.

Additionally, Office 2003 officially ended support when Office 2007 came out. That said, there is still VSTO support for 2003, but you can't expect it to last long. Microsoft provides support for current and previous versions only, and you can't assume that it will just keep supporting 2003 to be nice.

Installing VSTO

This book is about the Visual Studio 2010 implementation of VSTO targeting Office 2010. Throughout the book, we mention if functionality is different from Office 2007. Installing Visual Studio itself is actually pretty simple. You can, though, just get and install VSTO if all you're building is Office applications. Because you may be working in an earlier version and reading this book, I cover some other possibilities.

Requirements

According to Microsoft, hardware and software requirements are as follows:

✓ Hardware

- Computer with a 1.6 GHz or faster processor
- 1GB (32 bit) or 2GB (64 bit) RAM (Add 512MB if running in a virtual machine)
- 3GB of available hard disk space
- 5400 RPM hard disk drive
- DirectX 9 capable video card running at 1024 x 768 or higher-resolution display
- DVD-ROM drive

✓ Operating system (any one)

- Windows Vista (x86 or x64), all editions except Starter Edition
- Windows XP (x86 or x64), Service Pack 2 or later, all editions except Starter Edition
- Windows Server 2003 (x86 or x64), Service Pack 1 or later, all editions

- Windows Server 2003 R2 or later (x86 or x64), all editions
- Windows Server 2008 (x86 and x64) or later (all editions)
- Windows Server 2008 R2 (x64) Enterprise Edition
- Windows 7 (x86 and x64) Ultimate Edition

✓ Office system software (any one)

- Microsoft Office 2010, Professional Edition
- Microsoft Office 2010, Professional Edition with InfoPath
- Microsoft Word 2010
- Microsoft Excel 2010
- Microsoft InfoPath 2010
- Microsoft Outlook 2010
- Microsoft Office SharePoint Server 2010(for SharePoint Workflows)
- Microsoft SharePoint 2010 (for SharePoint specific projects)



In order to use SharePoint specific project templates, you need to run a server operating system. The system requirements are different for those OSs. Developing for SharePoint 2007 requires Microsoft Office SharePoint Services installation on a 32-bit machine. If you want to use SharePoint 2010 projects, you need to install either a server OS or Vista or Windows 7 OS. Note that SharePoint 2010 is only supported on 64-bit machines.

Supporting software



In order to develop for Office, you must have Office installed (see Figure 1-3). If you have Office 2007 installed, you can build Office 2007 add-ins and Office 2007 documents. If you have Office 2010 installed, you can build Office 2010 add-ins. If you have any combination of Office installations, you get everything. If you have nothing, VSTO will install, but will complain a lot when you try to use it. In short: you won't be able to do much with VSTO if Office isn't installed.



Please run Windows Update a few times if you install VSTO. All this supporting software has security patches available. Unprotected machines on the Internet have a 12-hour lifespan. Don't be a victim.

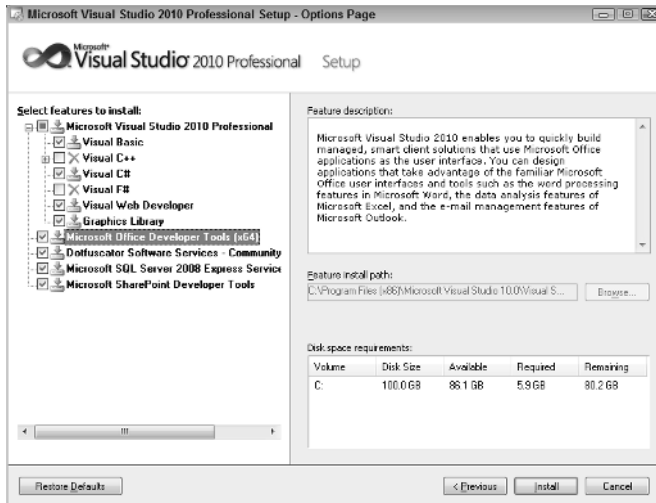


Figure 1-3:
Installing
VSTO.

Using VSTO in Visual Studio

VSTO uses Visual Studio as its Integrated Development Environment. A bunch of features come baked in to the collaboration, including

- ✓ Drag and drop user controls (some specific to Office)
- ✓ IntelliSense (a sort of dynamic, built-in documentation) for functions and properties
- ✓ Debugging
- ✓ Deployment

Because of the collaboration between VSTO and Visual Studio, some of this book is about using Visual Studio. If you are an experienced user of Visual Studio, you may find the coverage repetitive, but, in true *For Dummies* style, you're welcome to read only the information you need. Bear with us, though, because a surprising amount of the information is new.

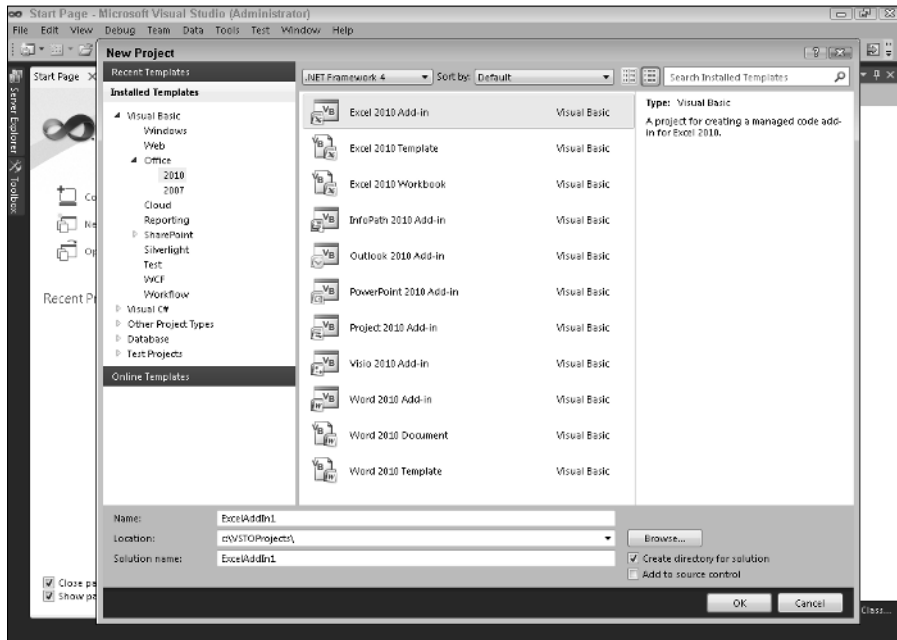
Using VSTO projects

From the starting user's perspective, a VSTO add-in or Customized Document is implemented as a Project Type in the New Project dialog box. To see what we mean, follow these steps:

1. Launch Visual Studio 2010 by choosing Start→All Programs.
2. Choose File→New Project.
3. In the Project Types panel, open the Visual Basic tree (if it isn't already open) and click Office.

The New Project dialog box, shown in Figure 1-4, appears.

Figure 1-4:
The New
Project dia-
log box.



The contents of Figure 1-4's Installed Templates change based on your installed base. For more on templates, review the section “Exploring the Different Versions of VSTO.” No matter what is in the templates list, you find subtle categories — add-ins and Customized Documents.

Working toward a finished product

Add-ins are application extensions that run within a host office program. Office applications are designed to run add-ins, as are many other Microsoft applications such as Internet Explorer and Visual Studio.

Add-ins add custom functionality to a host application. If you want to format an Excel cell a certain way, you'd just use a style or a custom number format. If you want to provide an Excel user with a way to use the latest ratios to convert all the dollars in a workbook to euros, you'd use an add-in. Basically, the functionality in an add-in is available to all documents.

Documents, which I usually call Customized Documents (to differentiate them from default documents, I guess), are actually files made to be opened by Office applications that have code you produce attached to them. Documents have a subcategory of their own — *templates* — which produces the template sort of document (like a .DOTX file, or a .XLTX file) that makes other documents.

Normal Customized Documents are just Office files with an attitude. They have a custom code library associated with them that gives special super powers.

For example, if you want to write a product catalog in Word, you'd just open Word and start typing. If you want that product catalog to get the list of products from a database, then you use a Customized Document. The functionality that you write will be available in just this one document, not every document the user opens.

Customized templates are interesting. You've probably used templates to make a custom work order for a company. You can include the logo and address of the company, and every time you need a new work order, the contents of the template are copied into a new spreadsheet. To teach that new work order how to save its contents to your accounting system, you use a customized template.

Starting with the end in mind

If you start with a project, you have to end with a result, right? You bet. You're not building scripts. You're building compiled applications, with executables and dynamic link libraries and configuration files. They just may not take the form you're expecting.

An installer

If you build an add-in project for Office 2003, you end up with an installer. The installer will be an .MSI file (for Microsoft Installer) that you can run on another Windows computer with the right version of Office already installed. The installer makes the add-in you wrote available to Office users on that computer. The VSTO installer is really the same as most other Visual Studio installers, except that it includes special Office registry keys.

After you've built this MSI, it's distributable, with certain security restrictions, to anyone. You can sell it, or give it away, or make it a download on your Web site or your corporate intranet. That installer will install a fully functioning application — it just requires the right version of Office to sit in, provided the security is set up.

A document file

Ever compiled something to a .DOC or .DOCX file? Neither had I until I used VSTO. It's a pretty cool feeling. When you make a document or document template project in Visual Studio, that's what you get — a file that you can open right up in an Office application.

The distribution of Customized Documents has a lot more restrictions, though. Theoretically, anyone can use them, but realistically, the end user (or their technical support engineer) has to do major security maneuvering to get the file to run. After those restrictions are met, though, you have a .DOCX or .XLSX file that is ready to do some of the user's work for them.