VSTO add-in developer's guide to Office Web Add-ins

06/25/2025

So, you've made some VSTO add-ins for Office applications that run on Windows and now you're exploring the new way of extending Office that will run on Windows, Mac, and the web browser version of the Office suite: Office Web Add-ins.

> **Important**
>
> COM and VSTO add-ins aren't supported in the new Outlook on Windows. These add-ins are still supported in the classic Outlook on Windows desktop client. To learn more, see Develop Outlook add-ins for new Outlook on Windows.

Your understanding of the object models for the Excel, Word, and the other Office applications will be a huge help because the object models in Office Web Add-ins follow similar patterns. But there are going to be some challenges:

- You'll be working with a different language (either JavaScript or TypeScript) instead of C# or Visual Basic .NET. (There is also a way, described later, to reuse some of your existing code in a web add-in.)
- Office Web Add-ins are deployed differently from VSTO add-ins.
- Office Web Add-ins are web applications that run in a simplified webview control that is embedded in the Office application, so you need to gain a basic understanding of web applications and how they're hosted on web servers or cloud accounts.

For these reasons, much of this article duplicates our Beginner's guide to Office extensions. What we've added are some learning resources to help VSTO add-in developers leverage their experience, and also help them reuse their existing code.

Step 0: Prerequisites

- Office Web Add-ins (also referred to as Office Add-ins) are essentially web applications embedded in Office. So, you should first have a basic understanding of web applications and how they're hosted on the web. There's an enormous amount of information about this on the Internet, in books, and in online courses. A good way to start if you have no prior knowledge of web applications at all is to search for "What is a web app?" in your search engine.

- The primary programming language you'll use to create Office Add-ins is JavaScript or TypeScript. If you're not familiar with either of these languages, but you have experience with VBA, VB.NET, C#, you'll probably find TypeScript easier to learn. Again, there's a wealth of information about these languages on the Internet, in books, and in online courses.

Step 1: Begin with fundamentals

We know you're eager to start coding, but there are some things about Office Add-ins that you should read before you open your IDE or code editor.

- Office Add-ins Platform Overview: Find out what Office Web Add-ins are and how they differ from older ways of extending Office, such as VSTO add-ins.
- Develop Office Add-ins: Get an overview of Office Add-in development and lifecycle including tooling, creating an add-in UI, and using the JavaScript APIs to interact with the Office document.

There are a lot of links in those articles, but if you're transitioning to Office Web Add-ins, we recommend that you come back here when you've read them and continue with the next section.

Step 2: Install tools and create your first add-in

You've got the big picture now, so dive in with one of our quick starts. For purposes of learning the platform, we recommend the PowerPoint quick start for Visual Studio.

Step 3: Code

You can't learn to drive by reading the owner's manual, so start coding with this PowerPoint tutorial. You'll be using the Office JavaScript library and some XML in the add-in's manifest. There's no need to memorize anything, because you'll be getting more background about both in a later step.

Step 4: Understand the JavaScript library

Get the big picture of the Office JavaScript library with the article Develop Office Add-ins.

Then, explore the Office JavaScript APIs with the Script Lab tool -- a sandbox for running and exploring the APIs.

Special resource for VSTO add-in developers

This would be a good place to take a look at the sample add-in, Excel Add-in JavaScript SalesTracker. It was created to highlight the similarities and differences between VSTO add-ins and Office Web Add-ins, and the readme of the sample calls out the important points of comparison.

Step 5: Understand the manifest

Get an understanding of the purposes of the web add-in manifest and an introduction to its XML markup or JSON in Office Add-ins manifest.

Step 6 (for VSTO developers only): Reuse your VSTO code

You can reuse some of your VSTO add-in code in an Office web add-in by moving it to your web application's back end on the server and making it available to your JavaScript or TypeScript as a web API. For guidance, see Tutorial: Share code between both a VSTO Add-in and an Office Add-in by using a shared code library.

Step 7: Create a Partner Center account

If you plan to publish your add-in to AppSource, create a Partner Center account. This could take some time. Get this process going as soon as possible to avoid release delays.

Next Steps

Congratulations on finishing the VSTO add-in developer's learning path for Office Web Add-ins! Here are some suggestions for further exploration of our documentation:

- Tutorials or quick starts for other Office applications:
  - Excel tutorial
  - OneNote quick start
  - Outlook tutorial
  - Project quick start
  - Word tutorial

> **Note**
>
> These tutorials use Yo Office as their tooling infrastructure, instead of Visual Studio and the Office development templates.

- Other important subjects:
  - Develop Office Add-ins
  - Best practices for developing Office Add-ins
  - Design Office Add-ins
  - Test and debug Office Add-ins
  - Deploy and publish Office Add-ins
  - Resources
  - Learn about the Microsoft 365 Developer Program