

Creating Custom Tasks using the Visual Studio Templates

Visual Studio templates provide a "quick start" method for creating new development projects that have a specific purpose or focus. The templates provided here can help to provide a quick start when creating a new custom task project. They help to automate the process of creating a simple project with a single custom task that has a simple user interface.

The templates are designed to work with:

- Microsoft Visual Studio 2008 Professional Edition (or Enterprise and Team System editions)
- Microsoft Visual Basic 2008 Express Edition
- Microsoft Visual C# 2008 Express Edition

The template definitions reside in these two files:

- CS2008SasTemplate.zip (for C# projects)
- VB2008SasTemplate.zip (for Visual Basic .NET projects)

To install the templates for use in Visual Studio:

1. Navigate to this folder on the PC (must be the machine where Visual Studio is installed):
%userprofile%\Documents\Visual Studio 2008\Templates\Project Templates
Note: *%userprofile%* is a Windows environment variable that resolves to your personal folders area.
2. Create a new folder named "SAS Custom Tasks" under the Project Templates area.
3. Copy the two .zip files (CS2008SasTemplate.zip and VB2008SasTemplate.zip) to the new folder. **Do not** extract the files; the templates remain as ZIP archive files for use by Visual Studio.

The remaining sections describe how to use the templates to create development projects for your custom tasks.

Creating a Custom Task Project using Microsoft Visual Basic 2008 Express Edition

1. Open Microsoft Visual Basic 2008 Express Edition.
2. Choose File->New Project. The New Project window appears, as shown Figure 1.

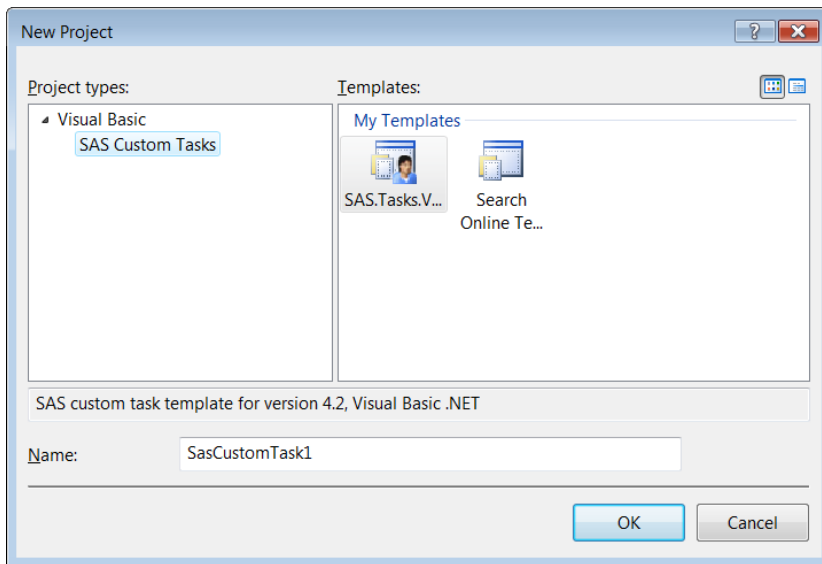


Figure 1. SAS Custom Task template within Visual Basic

The custom task template appears in the SAS Custom Tasks category.

3. Select the SAS.Tasks.VBTemplate42 template.

4. Enter a name for your project.

Note: A one-level name that contains no "dot" characters will work best. For example, use "SasStatTask" instead of "SAS.Stat.Task". Additional dots within the task name can cause namespace problems when the task is built later.

Click OK when complete. Microsoft Visual Basic creates a new project with the custom task classes, as shown in Figure 2.

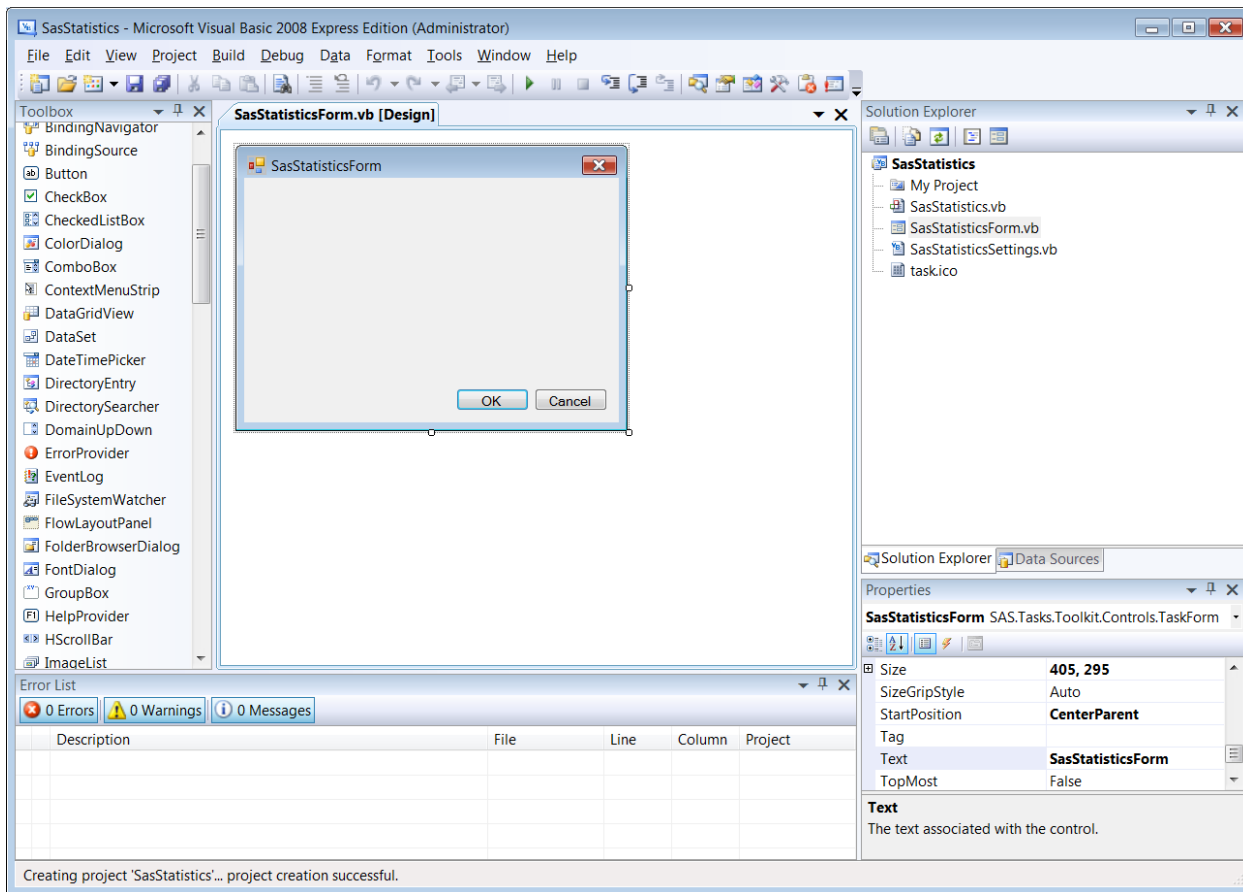


Figure 2. A new Visual Basic project named SasStatistics, ready to build

5. Choose File->Save All to save the new project. The Save Project window allows you to specify a directory location for the new project files. When you compile and build the project, the output files will appear in the directory location that you specify in this step.

The new project contains:

- A Visual Basic project file with references to the SAS Enterprise Guide DLLs that are needed to build the task. The references will work if SAS Enterprise Guide 4.2 is installed in its default location ("C:\Program Files\SAS\EnterpriseGuide\4.2"). If your installation is in a different location, you might need to add a reference path to your project to include the path for SAS Enterprise Guide or SAS Add-In for Microsoft Office.
- A class file (*projectName.vb*) that implements the SAS custom task APIs. This implementation provides the task description information (such as name and category), as well as default implementations for saving the task state, launching the task user interface, and generating a SAS program when the task is run. This class inherits from `SAS.Tasks.Toolkit.SasTask`, which implements most of the mechanics needed for a custom task to appear within a SAS application.
- A Windows form class (*projectNameForm.vb*) that provides a simple Windows Form as a user interface to the task.
- A task settings class (*projectNameSettings.vb*) that provides a convenient place for you to record and track properties and settings that are used within the task.

To build the task (and thus create a task DLL), select the Build *projectName* item from the Build menu. The project should build without errors.

Creating a Custom Task Project using Microsoft Visual C# Express Edition

1. Open Microsoft Visual C# Express Edition.
2. Choose File->New Project. The New Project window appears, as shown in Figure 3.

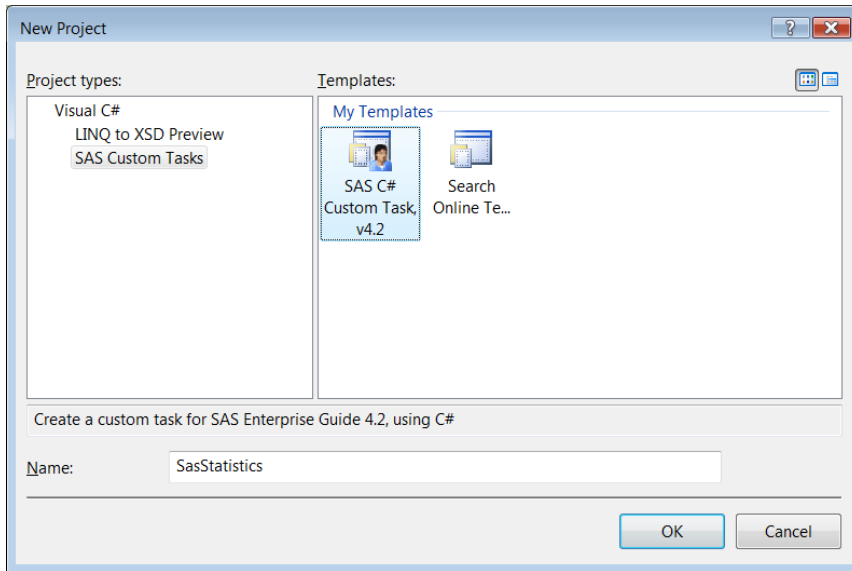


Figure 3. The New Project window in Visual C#

3. Select the SAS C# Custom Task v4.2 template.
4. Enter a name for your new project and click OK.

Note: Unlike Visual Basic, the C# environment can work well with multilevel namespaces. If you want to create a project name with a root namespace like "SAS.Statistics", you can use that as the name of your project, including the "dot" character.

Microsoft Visual C# creates a new project with the custom task classes, as shown in Figure 4.

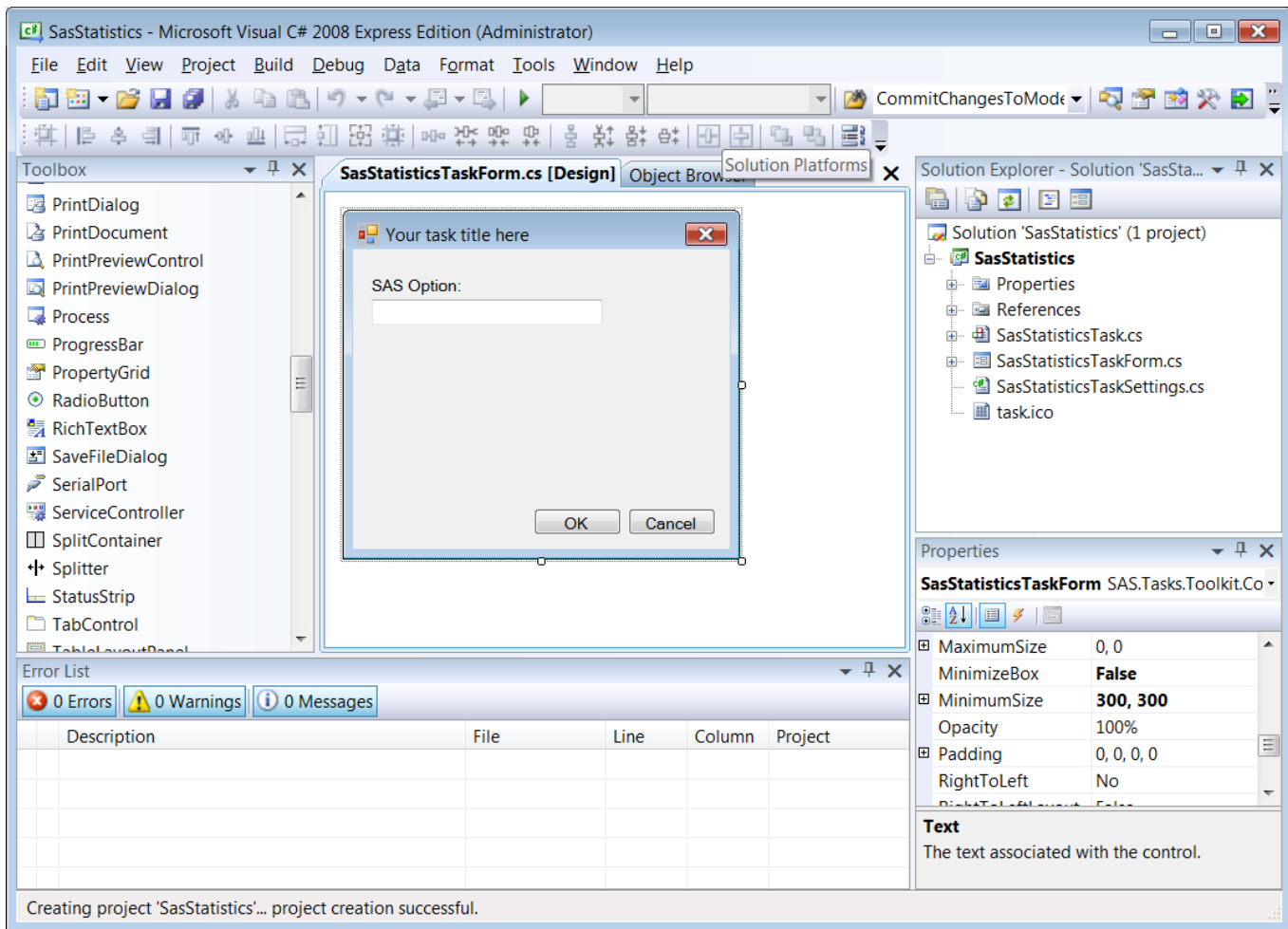


Figure 5. A new C# project named SasStatistics, ready to build

5. Choose File->Save All to save the new project. The Save Project window allows you to specify a directory location for the new project files. When you compile and build the project, the output files will appear in the directory location that you specify in this step.

The new project contains:

- A Visual C# project file with references to the SAS Enterprise Guide DLLs that are needed to build the task. The references will work if SAS Enterprise Guide 4.2 is installed in its default location ("C:\Program Files\SAS\EnterpriseGuide\4.2"). If your installation is in a different location, you might need to add a reference path to your project to include the path for SAS Enterprise Guide or SAS Add-In for Microsoft Office.
- A class file (*projectNameTask.cs*) that implements the SAS custom task APIs. This implementation provides the task description information (such as name and category), as well as default implementations for saving the task state, launching the task user interface, and generating a SAS program when the task is run. This class inherits from `SAS.Tasks.Toolkit.SasTask`, which implements most of the mechanics needed for a custom task to appear within a SAS application.
- A Windows form class (*projectNameTaskForm.cs*) that provides a simple Windows Form as a user interface to the task.

- A task settings class (*projectNameTaskSettings.cs*) that provides a convenient place for you to record and track properties and settings that are used within the task.

Creating a Custom Task Project using Microsoft Visual Studio Professional Edition

You can use these templates within Microsoft Visual Studio Professional Edition (or Enterprise or Team Developer editions). These higher end editions of Microsoft Visual Studio support many more project types; and you can use your preferred development language (C# or Visual Basic) from within this environment.

1. Open Microsoft Visual Studio.
2. Choose File->New->Project. The New Project window appears, as shown in Figure 6.

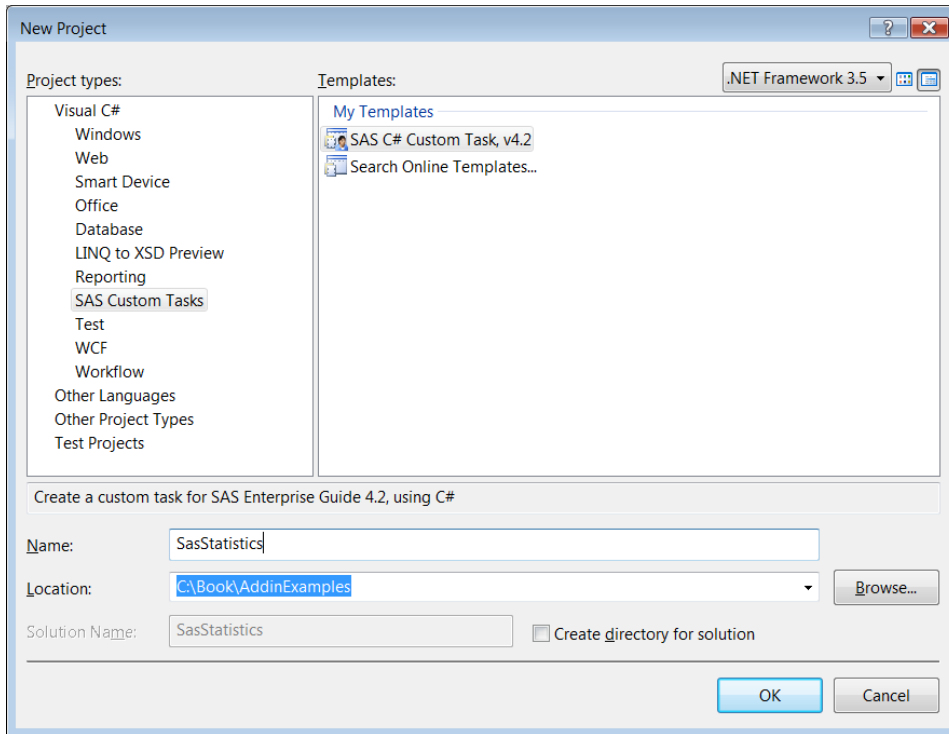


Figure 6. The New Project window within Microsoft Visual Studio

3. Select the SAS Custom Tasks category in the Project Types list. It appears under both the Visual C# list as well as the Other Languages->Visual Basic list.
4. Within the SAS Custom Tasks category, select the type of custom task (C# or Visual Basic) to create.
5. Specify a name for the project, and a directory location to store the project files and build output; then click OK.

The project is created and opened within Microsoft Visual Studio.

When working with your new project, here are some important notes to keep in mind:

- The project contains references to SAS Enterprise Guide assemblies (DLLs) for the custom task APIs and the SAS.Tasks.Toolkit implementation. These references point to the default location for SAS Enterprise Guide 4.2, which is in "C:\Program Files\SAS\EnterpriseGuide\4.2". If your installation of SAS Enterprise Guide is in a different location, you might need to change these references or add a new reference path. Likewise, if you

don't have SAS Enterprise Guide installed, but instead are using SAS Add-In for Microsoft Office, you will need to add a reference path to your project for that location (by default, "C:\Program Files\SAS\Add-InForMicrosoftOffice\4.2").

- The project contains a class file that implements the SAS custom task APIs. This implementation provides the task description information (such as name and category), as well as default implementations for saving the task state, launching the task user interface, and generating a SAS program when the task is run. This class inherits from SAS.Tasks.Toolkit.SasTask, which implements most of the mechanics needed for a custom task to appear within a SAS application.
- The project contains a Windows form class that provides a simple Windows Form as a user interface to the task.
- A task settings class that provides a convenient place for you to record and track properties and settings that are used within the task. The use of a settings class is optional, but it provides a useful separation between the task content and the mechanics of registering/displaying a task within the SAS products.

Deploying Custom Tasks

A custom add-in task "ships" within a .NET assembly (or DLL file). You may have multiple tasks within a single .NET assembly or simply organize them as one custom task per file, depending on your preferences. It is convenient to package several related tasks together into a single assembly, as it makes it easier to share code and implementations among the tasks.

Once the .NET assembly has been built, deploying and registering to target client machines is simple, using one of two methods. Once added, the add-in tasks will be available in from the Add-In menu in SAS Enterprise Guide, as well as in the task list. In the SAS Add-in for Microsoft Office, you will find the new entries when you select Manage SAS Favorites.

Method 1: "Drop-in" Deployment

You do not need to perform a separate registration step in order for SAS Enterprise Guide or SAS Add-In for Microsoft Office to recognize your custom task. Instead, you can simply copy the assembly (DLL file) to a special folder and the application will recognize it automatically the next time that you start.

With SAS Enterprise Guide 4.2 and SAS Add-In for Microsoft Office 4.2, there are multiple special folders that the application will scan for custom tasks. For SAS Enterprise Guide, the folders are:

- %appdata%\SAS\EnterpriseGuide\4.2\Custom
- C:\Program Files\SAS\EnterpriseGuide\4.2\Custom

For SAS Add-In for Microsoft Office, the folders are:

- %appdata%\SAS\AddInForMicrosoftOffice\4.2\Custom
- C:\Program Files\SAS\AddInForMicrosoftOffice\4.2\Custom

And finally, if you want to deploy a single copy of a custom task in both applications, you can copy it to this one special folder:

%appdata%\sas\SharedSettings\4.2\Custom

The "%appdata%" notation is a Microsoft Windows environment variable that maps to your personal profile area, which is specific to your user account on the machine. This allows you to add custom tasks to your installation without affecting any other users who might share your machine.

Copyright © 2010 SAS Institute Inc.

Note: if your custom task depends on other assemblies that ship with SAS Enterprise Guide or SAS Add-In for Microsoft Office, you do not need to copy those files to the Custom directory. However, if your custom task depends on assemblies that are not supplied by SAS (either 3rd party or that you developed), you *do* need to copy those to the Custom directory along with the task.

Method 2: Using the Add-In Manager

The Add-In Manager window lets you select a custom task from any location and register it for use in your SAS applications. To get started:

1. Copy the .NET assembly (and any dependent assemblies that your implementation might reference, excluding those provided with SAS Enterprise Guide or the SAS Add-in for Microsoft Office) to a location on the target machine. If you plan to deploy more than one assembly with add-in tasks, you might want to designate a single directory to group them together.
2. In SAS Enterprise Guide 4.2, select Tools->Add-In->Add-In Manager...

In SAS Add-In for Microsoft Office 4.2, run "C:\Program Files\SAS\AddInForMicrosoftOffice\4.2\RegAddin.exe" (the actual location of this program might vary depending on your configuration).

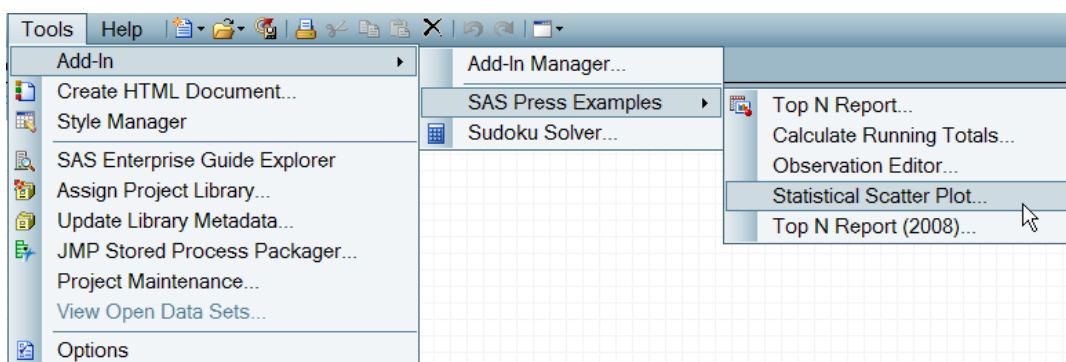
The Add-in Manager dialog allows you to register new add-in tasks by simply browsing to the .NET assembly.

1. Click the Browse... button to navigate to where your add-in assembly is located and then click Open. The Add-in Manager shows the available tasks within the assembly that you selected.
2. Click OK to accept the add-ins.

Accessing Custom Tasks from the Menu

After your task is registered, either with the "drop-in" method or the add-in manager method, you can launch the task from either the menu or the task list.

In SAS Enterprise Guide 4.2, the Add-In menu located under the top-level Tools menu. This figure shows the Add-In menu within SAS Enterprise Guide 4.2:



You can also find the custom tasks listed within the task list (accessed with View->Task List). In the task list, the tasks are organized by category and the custom tasks are intermixed with the built-in tasks. In the menus, the custom tasks are separated into their own "Add-In" menu structure.