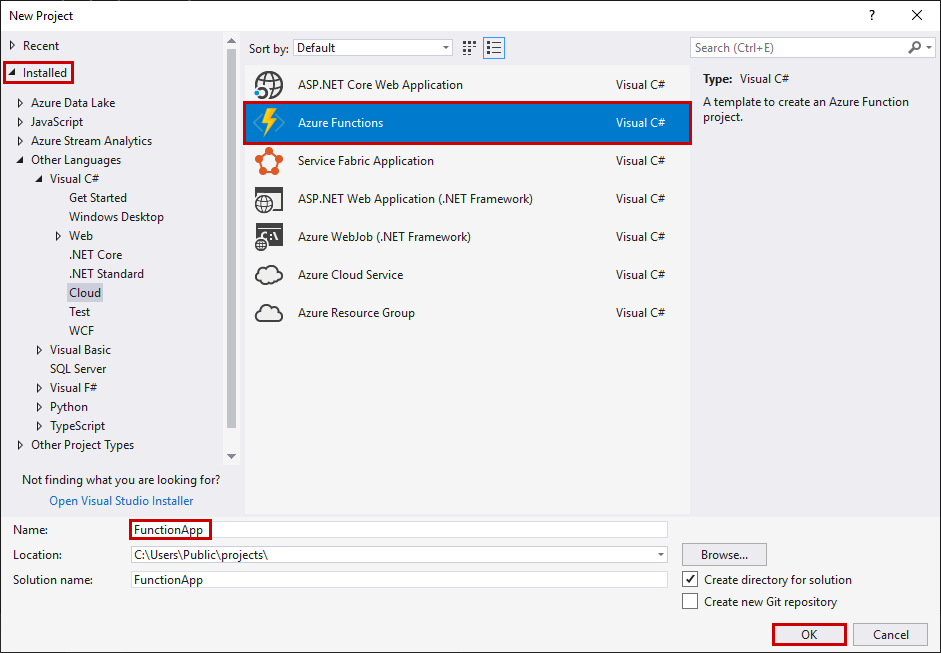
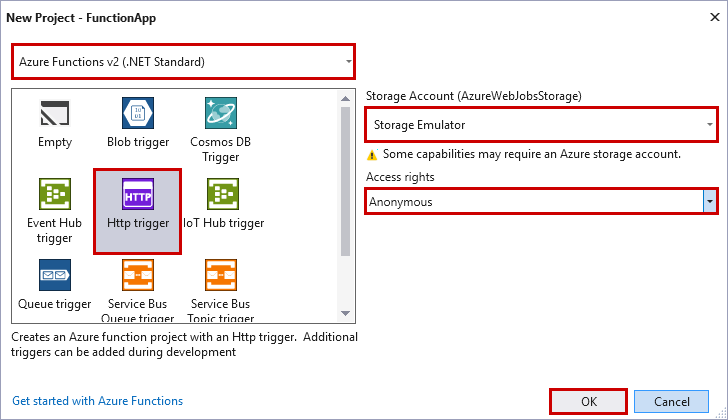
**Create a function app project**

The Azure Functions project template in Visual Studio creates a project that can be published to a function app in Azure. A function app lets you group functions as a logical unit for management, deployment, and sharing of resources.

1. In Visual Studio, select **New** > **Project** from the **File** menu.
2. In the **New Project** dialog, select **Installed**, expand **Visual C#** > **Cloud**, select **Azure Functions**, type a **Name** for your project, and click **OK**. The function app name must be valid as a C# namespace, so don't use underscores, hyphens, or any other nonalphanumeric characters.



1. Use the settings specified in the table that follows the image.



| **Setting** | **Suggested value** | **Description** |
| --- | --- | --- |
| **Version** | Azure Functions 2.x  (.NET Core) | This creates a function project that uses the version 2.x runtime of Azure Functions which supports .NET Core. Azure Functions 1.x supports the .NET Framework. For more information, see [How to target Azure Functions runtime version](https://docs.microsoft.com/en-us/azure/azure-functions/functions-versions). |
| **Template** | HTTP trigger | This creates a function triggered by an HTTP request. |
| **Storage account** | Storage Emulator | An HTTP trigger doesn't use the Storage account connection. All other trigger types require a valid Storage account connection string. |
| **Access rights** | Anonymous | The created function can be triggered by any client without providing a key. This authorization setting makes it easy to test your new function. For more information about keys and authorization, see [Authorization keys](https://docs.microsoft.com/en-us/azure/azure-functions/functions-bindings-http-webhook#authorization-keys) in the [HTTP and webhook bindings](https://docs.microsoft.com/en-us/azure/azure-functions/functions-bindings-http-webhook). |

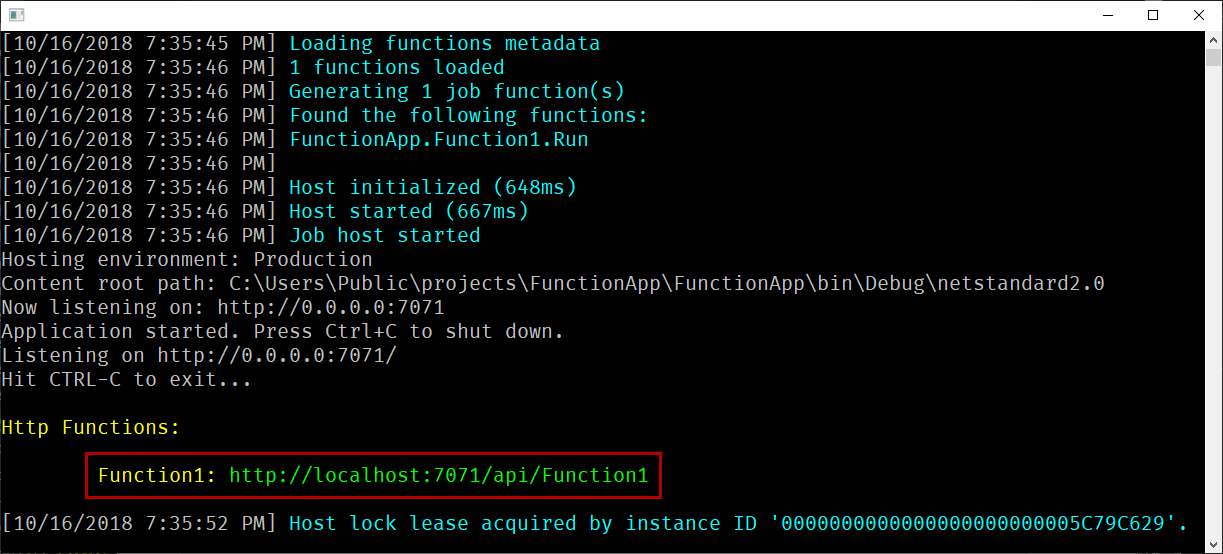
1. Click **OK** to create the function project and HTTP triggered function.

Visual Studio creates a project and in it a class that contains boilerplate code for the chosen function type. The **FunctionName** attribute on the method sets the name of the function. The **HttpTrigger** attribute specifies that the function is triggered by an HTTP request. The boilerplate code sends an HTTP response that includes a value from the request body or query string. You can add input and output bindings to a function by applying the appropriate attributes to the method. For more information, Now that you've created your function project and an HTTP-triggered function, you can test it on your local computer.

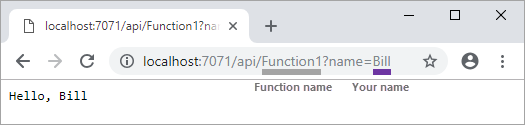
**Test the function locally**

Azure Functions Core Tools lets you run an Azure Functions project on your local development computer. You are prompted to install these tools the first time you start a function from Visual Studio.

1. To test your function, press F5. If prompted, accept the request from Visual Studio to download and install Azure Functions Core (CLI) tools. You may also need to enable a firewall exception so that the tools can handle HTTP requests.
2. Copy the URL of your function from the Azure Functions runtime output.



1. Paste the URL for the HTTP request into your browser's address bar. Append the query string ?name=<YOUR\_NAME> to this URL and execute the request. The following shows the response in the browser to the local GET request returned by the function:



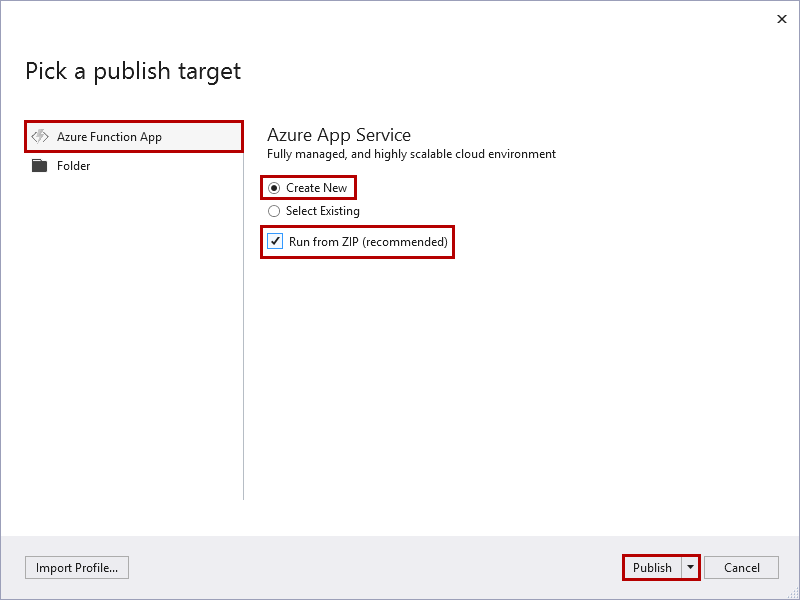
1. To stop debugging, press **Shift + F5**.

After you have verified that the function runs correctly on your local computer, it's time to publish the project to Azure.

**Publish the project to Azure**

You must have a function app in your Azure subscription before you can publish your project. You can create a function app right from Visual Studio.

1. In **Solution Explorer**, right-click the project and select **Publish**.
2. Select **Azure Function App**, choose **Create New**, and then select **Publish**.

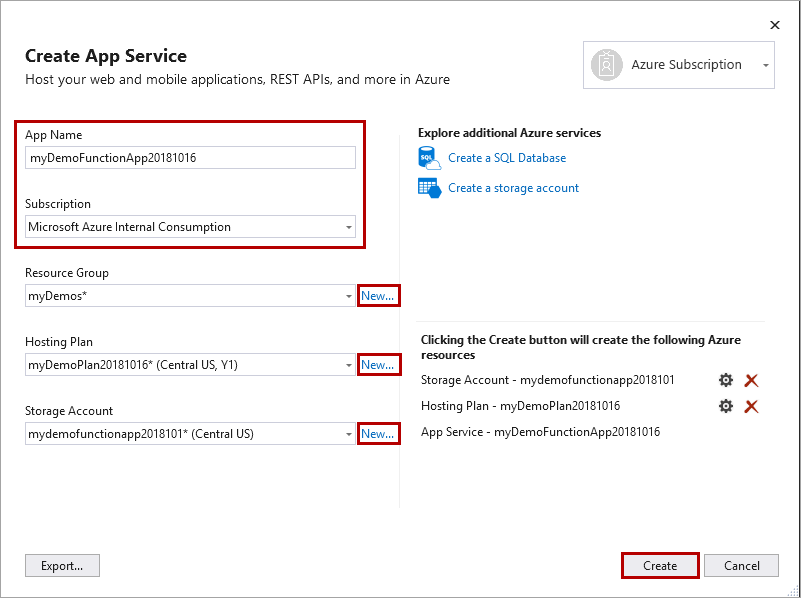


When you enable **Run from Zip**, your function app in Azure goes into read-only mode and is run directly from the deployment package. For more information, see [Run your Azure Functions from a package file](https://docs.microsoft.com/en-us/azure/azure-functions/run-functions-from-deployment-package).

Caution

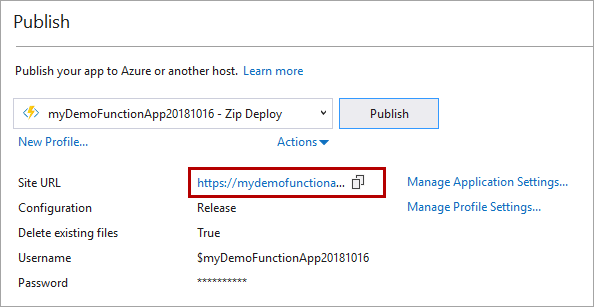
When you choose **Select Existing**, all files in the existing function app in Azure are overwritten by files from the local project. Only use this option when republishing updates to an existing function app.

1. If you haven't already connected Visual Studio to your Azure account, select **Add an account...**.
2. In the **Create App Service** dialog, use the **Hosting** settings as specified in the table below the image:



| **Setting** | **Suggested value** | **Description** |
| --- | --- | --- |
| **App Name** | Globally unique name | Name that uniquely identifies your new function app. |
| **Subscription** | Choose your subscription | The Azure subscription to use. |
| [**Resource Group**](https://docs.microsoft.com/en-us/azure/azure-resource-manager/resource-group-overview) | myResourceGroup | Name of the resource group in which to create your function app. Choose **New** to create a new resource group. |
| [**App Service Plan**](https://docs.microsoft.com/en-us/azure/azure-functions/functions-scale) | Consumption plan | Make sure to choose the **Consumption** under **Size** after you click **New** to create a serverless plan. Also, choose a **Location** in a [region](https://azure.microsoft.com/regions/) near you or near other services your functions access. When you run in a plan other than **Consumption**, you must manage the [scaling of your function app](https://docs.microsoft.com/en-us/azure/azure-functions/functions-scale). |
| [**Storage Account**](https://docs.microsoft.com/en-us/azure/storage/common/storage-quickstart-create-account) | General purpose storage account | An Azure storage account is required by the Functions runtime. Click **New** to create a general purpose storage account. You can also use an existing account that meets the [storage account requirements](https://docs.microsoft.com/en-us/azure/azure-functions/functions-scale#storage-account-requirements). |

1. Click **Create** to create a function app and related resources in Azure with these settings and deploy your function project code.
2. After the deployment is complete, make a note of the **Site URL** value, which is the address of your function app in Azure.



**Test your function in Azure**

1. Copy the base URL of the function app from the Publish profile page. Replace the localhost:port portion of the URL you used when testing the function locally with the new base URL. As before, make sure to append the query string ?name=<YOUR\_NAME> to this URL and execute the request.

The URL that calls your HTTP triggered function should be in the following format:

Copy

http://<APP\_NAME>.azurewebsites.net/api/<FUNCTION\_NAME>?name=<YOUR\_NAME>

1. Paste this new URL for the HTTP request into your browser's address bar. The following shows the response in the browser to the remote GET request returned by the function:

