title: "Deploy a project with Transact-SQL (VS Code) | Microsoft Docs"  
ms.date: "08/21/2017"  
ms.topic: "article"  
ms.prod: "sql-server-2017"  
ms.technology:

* "integration-services"  
  author: "douglaslMS"  
  ms.author: "douglasl"  
  manager: "craigg"  
  ---

# Deploy an SSIS project from Visual Studio Code with Transact-SQL

* This quick start demonstrates how to use Visual Studio Code to connect to the SSIS Catalog database on an Azure SQL database server, and then use Transact-SQL statements to deploy an SSIS project to the SSIS Catalog.

Visual Studio Code is a code editor for Windows, macOS, and Linux that supports extensions, including the mssql extension for connecting to Microsoft SQL Server, Azure SQL Database, or Azure SQL Data Warehouse. For more info about VS Code, see [Visual Studio Cod](https://code.visualstudio.com/).

[!NOTE] Only the project deployment model is supported. For more info about SSIS deployment, and about converting a project to the project deployment model, see [Deploy Integration Services (SSIS) Projects and Packages](https://docs.microsoft.com/en-us/sql/integration-services/packages/deploy-integration-services-ssis-projects-and-packages.md).

## Prerequisites

Before you start, make sure you have installed the latest version of Visual Studio Code and loaded the mssql extension. To download these tools, see the following pages:

* [Download Visual Studio Code](https://code.visualstudio.com/Download)
* [mssql extension](https://marketplace.visualstudio.com/items?itemName=ms-mssql.mssql)

## Set language mode to SQL in VS Code

Set the language mode is set to **SQL** in Visual Studio Code to enable mssql commands and T-SQL IntelliSense.

1. Open Visual Studio Code and then open a new window.
2. Click **Plain Text** in the lower right-hand corner of the status bar.
3. In the **Select language mode** drop-down menu that opens, select or enter **SQL**, and then press **ENTER** to set the language mode to SQL.

## Connect to the SSIS Catalog database

Use Visual Studio Code to establish a connection to the SSIS Catalog on your Azure SQL Database server..

[!IMPORTANT]  
Before continuing, make sure that you have your server, database, and login information ready. Once you begin entering the connection profile information, if you change your focus from Visual Studio Code, you have to restart creating the connection profile.

1. In VS Code, press **CTRL+SHIFT+P** (or **F1**) to open the Command Palette.
2. Type **sqlcon** and press **ENTER**.
3. Press **ENTER** to select **Create Connection Profile**. This creates a connection profile for your SQL Server instance.
4. Follow the prompts to specify the connection properties for the new connection profile. After specifying each value, press **ENTER** to continue.

|  |  |  |
| --- | --- | --- |
| Setting | Suggested value | Description |
| **Server name** | The fully qualified server name | The name should be something like this: **mysqldbserver.database.windows.net**. |
| **Database name** | **SSISDB** | The name of the database to which to connect. |
| **Authentication** | SQL Login | This quickstart uses SQL authentication. |
| **User name** | The server admin account | This is the account that you specified when you created the server. |
| **Password (SQL Login)** | The password for your server admin account | This is the password that you specified when you created the server. |
| **Save Password?** | Yes or No | Select Yes if you do not want to enter the password each time. |
| **Enter a name for this profile** | A profile name, such as **mySSISServer** | A saved profile name speeds your connection on subsequent logins. |

1. Press the **ESC** key to close the info message that informs you that the profile is created and connected.
2. Verify your connection in the status bar.

## Run the T-SQL code

Run the following Transact-SQL code to deploy an SSIS project.

1. In the **Editor** window, enter the following query in the empty query window.
2. Update the parameter values in the catalog.deploy\_project stored procedure for your system.
3. Press **CTRL+SHIFT+E** to run the code and deploy the project.

DECLARE @ProjectBinary AS varbinary(max)  
DECLARE @operation\_id AS bigint  
SET @ProjectBinary = (SELECT \* FROM OPENROWSET(BULK '<project\_file\_path>.ispac', SINGLE\_BLOB) AS BinaryData)  
  
EXEC catalog.deploy\_project @folder\_name = '<target\_folder>', @project\_name = '<project\_name',  
 @Project\_Stream = @ProjectBinary, @operation\_id = @operation\_id out

## Next steps

* Run a package. To run a package, you can choose from several tools and languages. For more info, see the following articles:
  + [Run from SSMS](ssis-everest-quickstart-run-ssms.md)
  + [Run with T-SQL from SSMS](ssis-everest-quickstart-run-tsql-ssms.md)
  + [Run with T-SQL from VS Code](ssis-everest-quickstart-run-tsql-vscode.md)
  + [Run from command prompt](ssis-everest-quickstart-run-cmdline.md)
  + [Run from PowerShell](ssis-everest-quickstart-run-powershell.md)
  + [Run from C# app](ssis-everest-quickstart-run-dotnet.md)
* Schedule a package. For more info, see [Schedule page](ssis-everest-howto-schedule-package.md)