Session: 19

Debugging C# in Visual Studio 2017

Objectives

- Describe the concept of debugging in Visual Studio 2017
- Identify the different ways of debugging C# applications
- Explain the process of installing, updating, and removing NuGet packages

Introduction to Debugging

Debugging is a process that tracks an application by going through its each line of code.

Visual Studio 2017

- Allows debugging of C# applications.
- Indicates status of running code.
- Provides wide set of debugging tools.
- Allows changing values to see the possible outcome while the application is in its running state.

Debugging Tools in Visual Studio 2017 1-2

Default Build Configurations

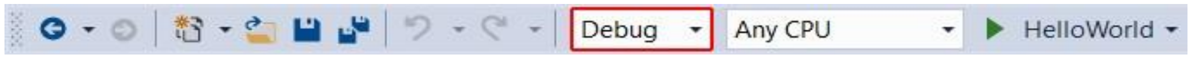
Release

- **Release configuration is for distributing the final version.**

Debug

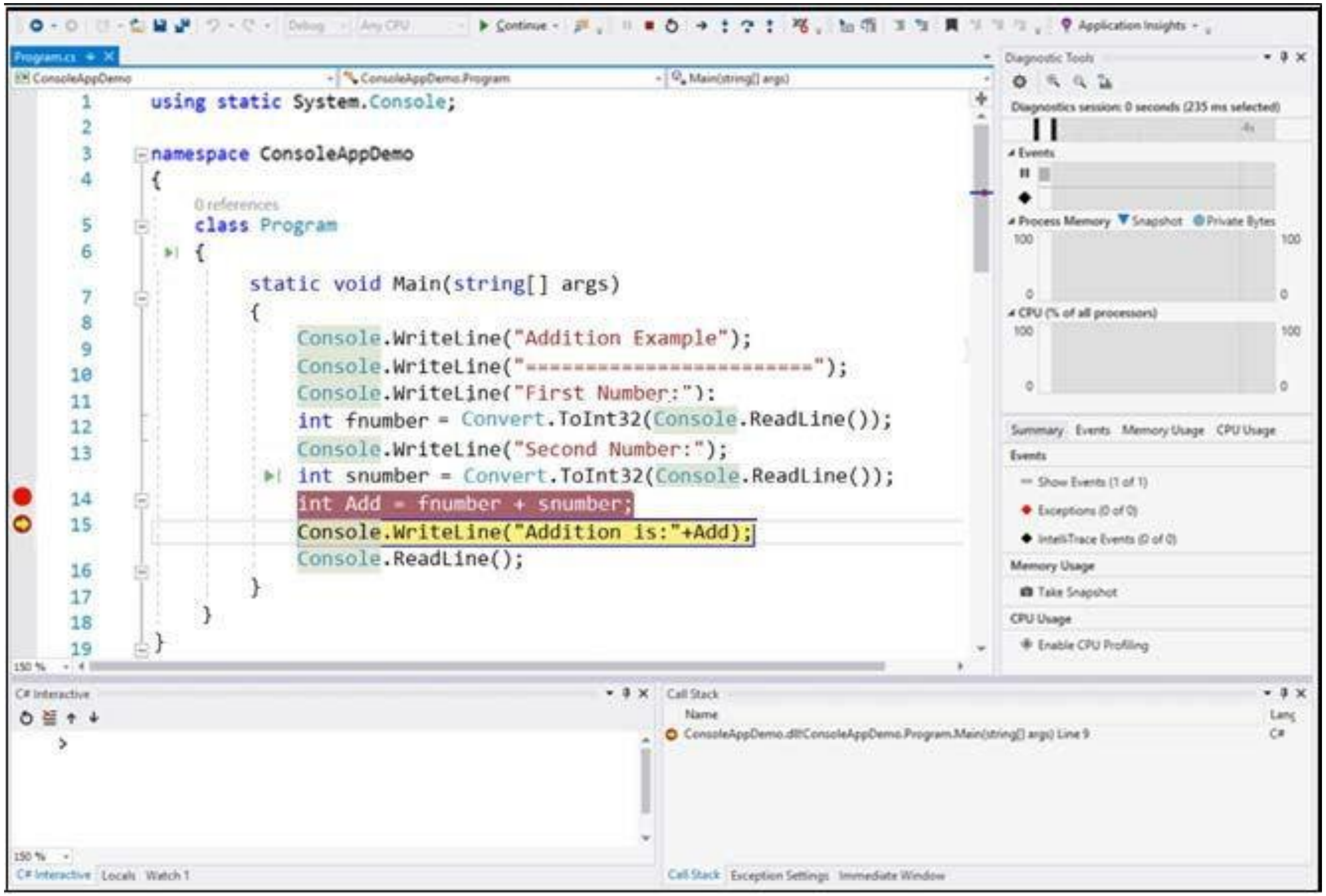
- **Debug configuration is for debugging the application code.**

Figure indicates that Visual Studio is set to compile the opened application in the Debug mode.



Debug Mode

Debugging Tools in Visual Studio 2017 2-2



```
using static System.Console;

namespace ConsoleAppDemo
{
    0 references
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Addition Example");
            Console.WriteLine("========================");
            Console.WriteLine("First Number:");
            int fnumber = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Second Number:");
            int snumber = Convert.ToInt32(Console.ReadLine());
            int Add = fnumber + snumber;
            Console.WriteLine("Addition is:"+Add);
            Console.ReadLine();
        }
    }
}
```

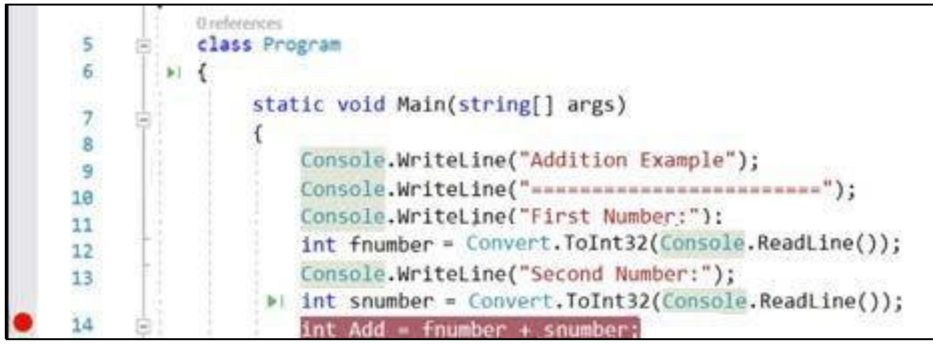
```
ConsoleAppDemo.dll!ConsoleAppDemo.Program.Main(string[] args) Line 9    C#
```

Debugging Environment in Visual Studio 2017

Using Breakpoints 1-2

Breakpoint:

- Refers to a signal that stops the application's execution at the already set points in the editor.
- Pauses execution prior to running the line of code having the breakpoint.
- Waits for the next instruction or command to proceed.
- Allows breaking the execution at any line of code with the help of debugger.



```
0 references
class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Addition Example");
        Console.WriteLine("=========================");
        Console.WriteLine("First Number:"):
        int fnumber = Convert.ToInt32(Console.ReadLine());
        Console.WriteLine("Second Number:");
        int snumber = Convert.ToInt32(Console.ReadLine());
        int Add = fnumber + snumber;
```

Breakpoint

Using Breakpoints 2-2

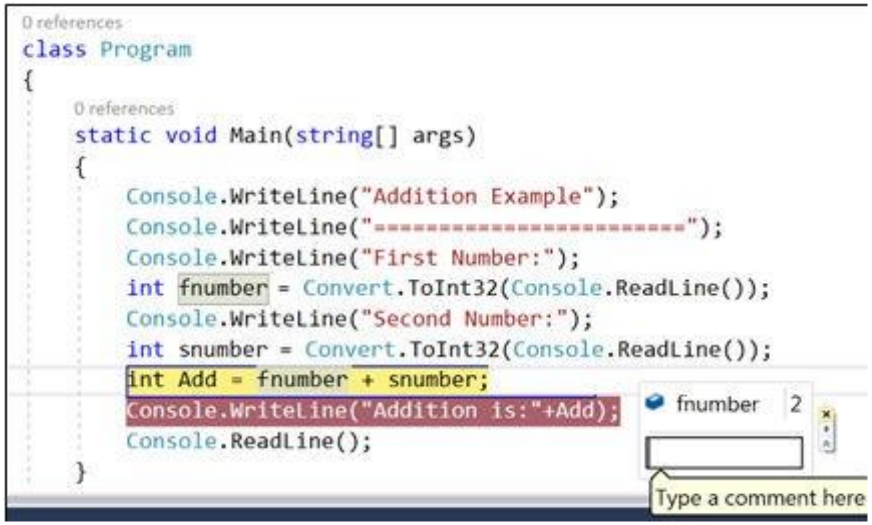
Ways to run debugger:

- Press F5
- Click green-play Icon on the toolbar
- Select Debug→ Start Debugging

Using Data Tips

Data Tip:

- Informative tooltips available only in break mode.
- Reveals more information about a selected variable or an object in the present execution scope.
- Can be pinned and can be commented.



```
0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        Console.WriteLine("Addition Example");
        Console.WriteLine("======================");
        Console.WriteLine("First Number:");
        int fnumber = Convert.ToInt32(Console.ReadLine());
        Console.WriteLine("Second Number:");
        int snumber = Convert.ToInt32(Console.ReadLine());
        int Add = fnumber + snumber;
        Console.WriteLine("Addition is:"+Add);
        Console.ReadLine();
    }
```

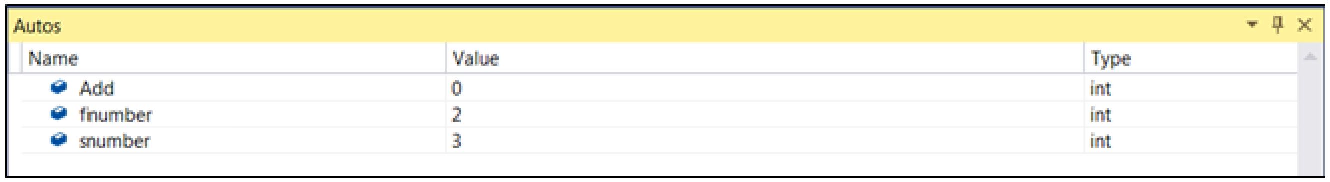
Data Tip

Using Windows and Conditional Breakpoints 1-3

If a developer sets a breakpoint on the line that prints user input, the Autos, Locals, Immediate and Watch windows are quite useful to observe.

Autos

A window that displays the information of all variables and objects such as type and value that are in the present or the previous line.



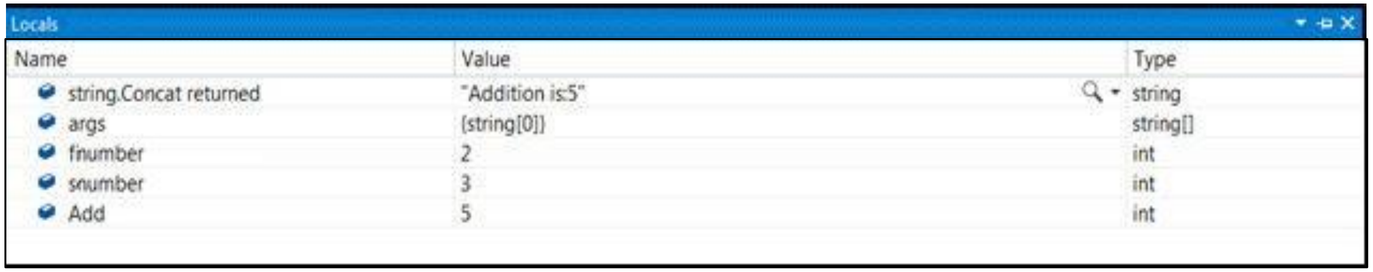
Autos

Name	Value	Type
Add	0	int
fnumber	2	int
snumber	3	int

Autos Window

Locals

Shows the values of local variables and objects that are within the scope of current execution.



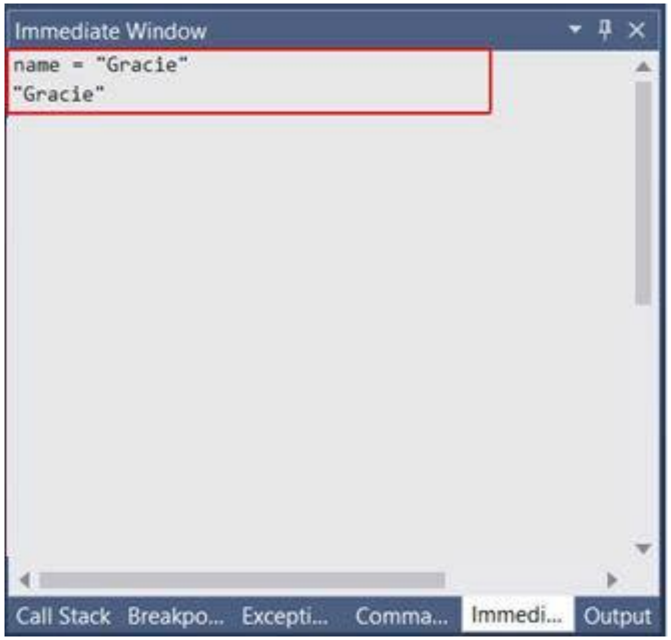
Locals

Name	Value	Type
string.Concat returned	"Addition is:5"	string
args	{string[0]}	string[]
fnumber	2	int
snumber	3	int
Add	5	int

Locals Window

Using Windows and Conditional Breakpoints 2-3

Immediate: A window that allows changing the values.

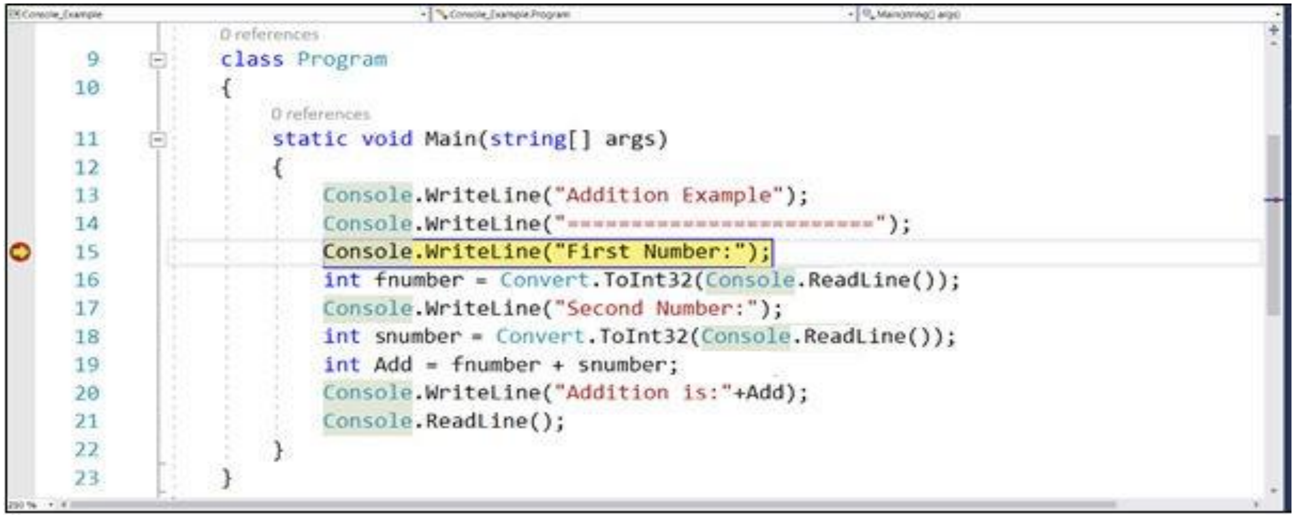


```
name = "Gracie"
"Gracie"
```

Immediate Window

Watch: A window that display information about variables and objects depending on the changes made to them.

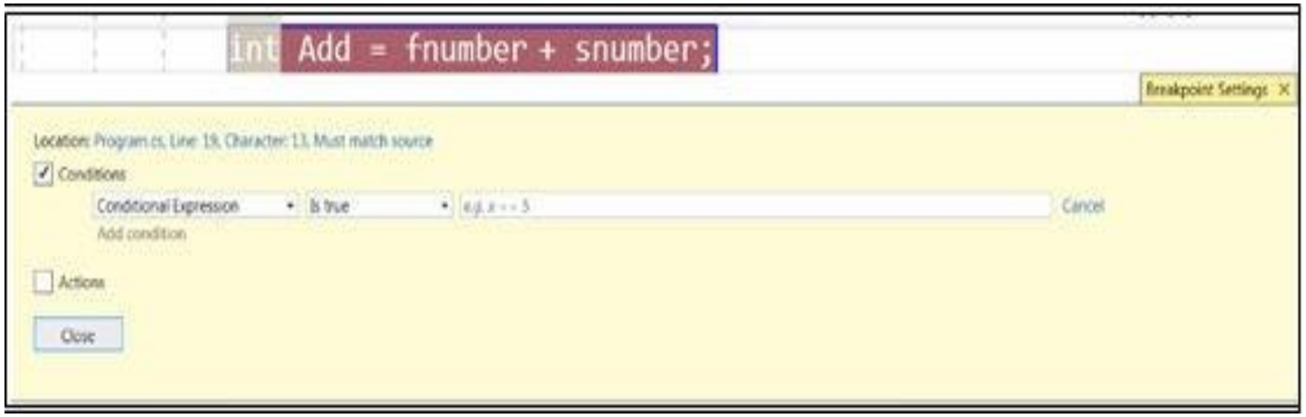
Using Windows and Conditional Breakpoints 3-3



```
Console_Example | Console_Example.Program | Main(string[] args)

0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        Console.WriteLine("Addition Example");
        Console.WriteLine("========================");
        Console.WriteLine("First Number:");
        int fnumber = Convert.ToInt32(Console.ReadLine());
        Console.WriteLine("Second Number:");
        int snumber = Convert.ToInt32(Console.ReadLine());
        int Add = fnumber + snumber;
        Console.WriteLine("Addition is:"+Add);
        Console.ReadLine();
    }
}
```

Code in Debug Mode



```
int Add = fnumber + snumber;
```

Breakpoint Settings

Location: Program.cs, Line 19, Character 13, Must match source

[✓] Conditions

Conditional Expression | Is true | e.g. x == 5 | Cancel

Add condition

[ ] Actions

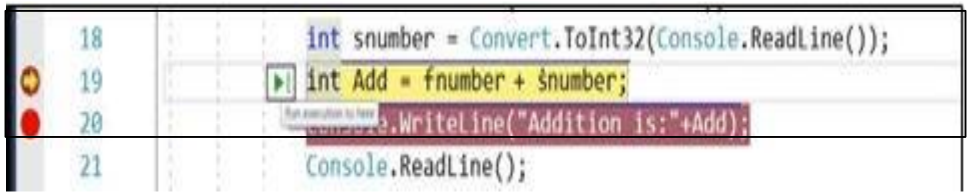
Close

Breakpoint Settings Dialogue Box

Using Run to Click

Run to click

- Aims to simplify the debugging steps.
- Informs the debugger to execute until it arrives at this line.
- Useful for debugging from one line to another in a block by simply clicking the icon.
- At this line, the debugger breaks and waits for the next command.



```
int snumber = Convert.ToInt32(Console.ReadLine());
int Add = fnumber + snumber;
        e.WriteLine("Addition is:"+Add);
Console.ReadLine();
```

Run to Click Feature

NuGet and Advantages

A free extension for Visual Studio that makes it efficient to look for all the open-source packages and use them in projects.

Makes it simple to add, update, and delete libraries in projects that rely on the .NET Framework.

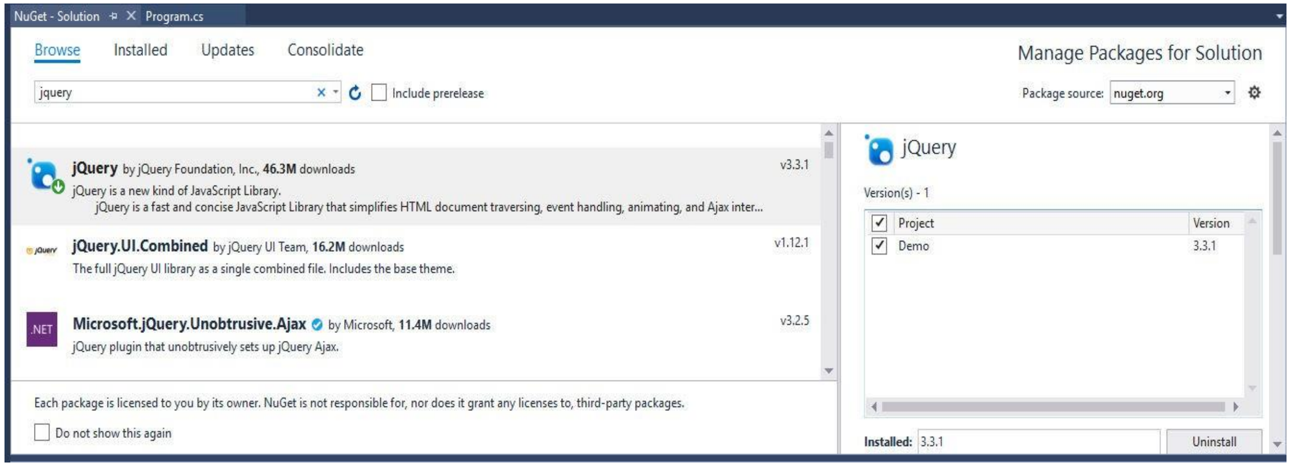
Is a free package management system, which allows sharing code.

Allow developers to reuse their own code across their own projects.

Automates the tasks of installing, setting up, and updating packages or components in a Visual Studio project.

Using NuGet to Install a Package 1-3

1. Go to **Solution Explorer**.
2. Right-click a project solution.
3. Select **Manage NuGet Packages for Solution**. The NuGet – Solution window is displayed. This is the NuGet Package Manager.



NuGet - Solution   Program.cs

Browse   Installed   Updates   Consolidate

Manage Packages for Solution

jquery   Include prerelease

Package source: nuget.org

jQuery by jQuery Foundation, Inc., 46.3M downloads   v3.3.1
jQuery is a new kind of JavaScript Library.
jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax inter...

jQuery.UI.Combined by jQuery UI Team, 16.2M downloads   v1.12.1
The full jQuery UI library as a single combined file. Includes the base theme.

Microsoft.jQuery.Unobtrusive.Ajax by Microsoft, 11.4M downloads   v3.2.5
jQuery plugin that unobtrusively sets up jQuery Ajax.

Each package is licensed to you by its owner. NuGet is not responsible for, nor does it grant any licenses to, third-party packages.

Do not show this again

jQuery

Version(s) - 1

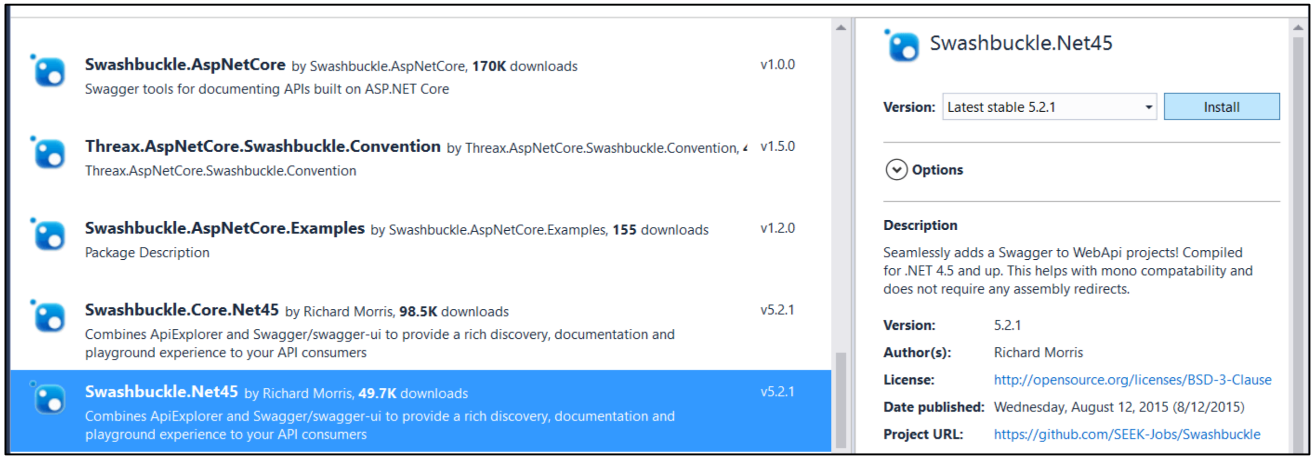
	Project	Version
✓	Demo	3.3.1

Installed: 3.3.1   Uninstall

NuGet – Solution Window

4. Enter the name of the desired package in the search field on the top.
5. Select the package from the list box and its corresponding details such as description, version, author, and license.
6. Select a suitable version from the Version drop-down list.

Using NuGet to Install a Package 2-3



Swashbuckle.AspNetCore by Swashbuckle.AspNetCore, **170K** downloads v1.0.0
Swagger tools for documenting APIs built on ASP.NET Core

Threax.AspNetCore.Swashbuckle.Convention by Threax.AspNetCore.Swashbuckle.Convention, v1.5.0
Threax.AspNetCore.Swashbuckle.Convention

Swashbuckle.AspNetCore.Examples by Swashbuckle.AspNetCore.Examples, **155** downloads v1.2.0
Package Description

Swashbuckle.Core.Net45 by Richard Morris, **98.5K** downloads v5.2.1
Combines ApiExplorer and Swagger/swagger-ui to provide a rich discovery, documentation and playground experience to your API consumers

Swashbuckle.Net45 by Richard Morris, **49.7K** downloads v5.2.1
Combines ApiExplorer and Swagger/swagger-ui to provide a rich discovery, documentation and playground experience to your API consumers

Swashbuckle.Net45

Version: Latest stable 5.2.1 Install

Options

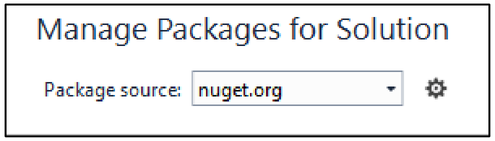
Description

Seamlessly adds a Swagger to WebApi projects! Compiled for .NET 4.5 and up. This helps with mono compatability and does not require any assembly redirects.

Version:	5.2.1
Author(s):	Richard Morris
License:	http://opensource.org/licenses/BSD-3-Clause
Date published:	Wednesday, August 12, 2015 (8/12/2015)
Project URL:	https://github.com/SEEK-Jobs/Swashbuckle

Package Details

7. Click **Install** to add the package.
8. Accept all prompts regarding licensing. The **Choose NuGet Package Manager Format** dialog box will be displayed.
9. Click **OK**. The Output window at the bottom of Visual Studio IDE displays a message that the package is installed successfully.
10. For installing a package, it is essential to specify the right source for it.



Manage Packages for Solution

Package source: nuget.org

Package Source

Using NuGet to Install a Package 3-3

Following are some ways to install a NuGet package:

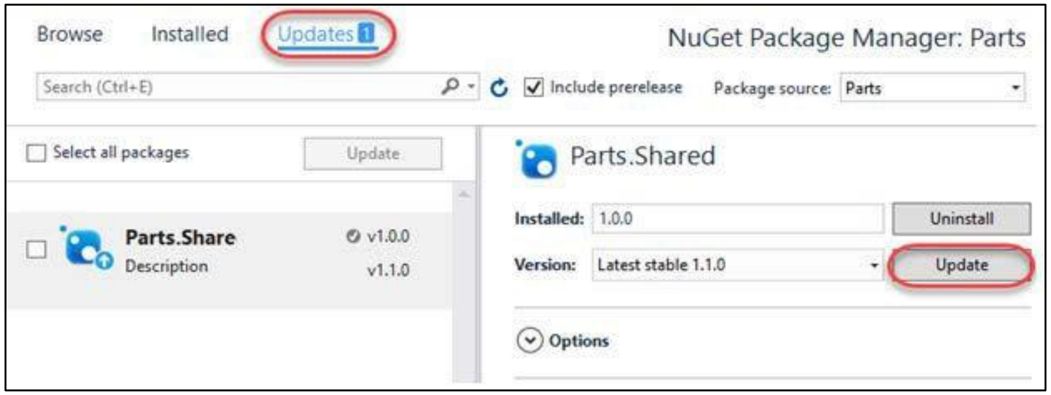
By using command window in the project folder and **dotnet.exe** command as: dotnet add package <package_name> which obtains the package as per its name, puts its contents into the current directory, and refers to it

By using the Package Manager Console (**Tools → NuGet Package Manager → Package Manager Console**) and entering the install package <package_name> command. The command obtains the package and its dependencies from a chosen source and installs it into the project

By using **nuget.exe** and entering the nuget package <package_name> command

Using NuGet to Update a Package

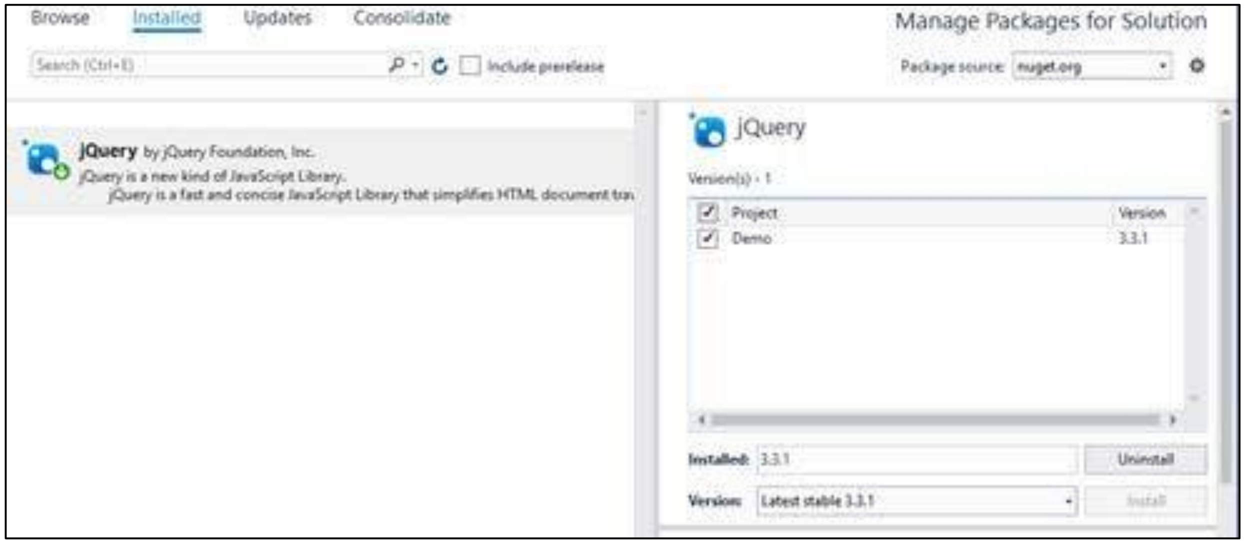
The **Updates** tab of the **NuGet – Solution** window allows updating the installed packages. It shows the packages whose updates are available from the chosen package sources.



Updates Tab

Using NuGet to Uninstall a Package

The **Installed** tab of the **NuGet – Solution** window allows updating or uninstalling the installed packages.



jQuery by jQuery Foundation, Inc.
jQuery is a new kind of JavaScript Library.
jQuery is a fast and concise JavaScript Library that simplifies HTML document tra

Version(s) - 1

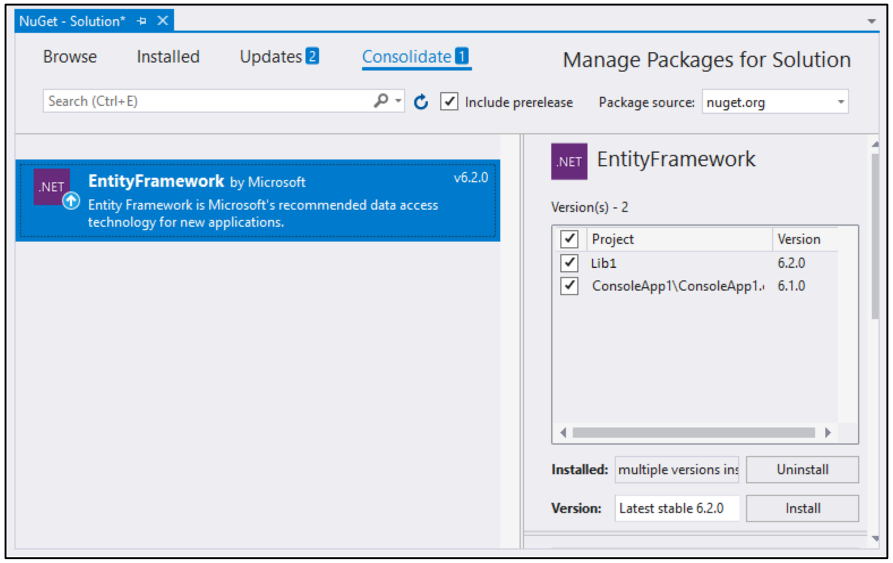
	Project	Version
✓	Demo	3.3.1

Installed: 3.3.1
Version: Latest stable 3.3.1

Installed Tab

Using NuGet to Monitor Packages

If different versions of NuGet package are in use, it is possible to manage or merge them using the **Consolidate** tab of the **NuGet – Solution** window.


NuGet - Solution*
Browse
Installed
Updates 2
Consolidate 1
Manage Packages for Solution
Search (Ctrl+E)
Include prerelease
Package source: nuget.org
.NET
EntityFramework by Microsoft
v6.2.0
Entity Framework is Microsoft's recommended data access technology for new applications.
EntityFramework
Version(s) - 2
Uninstall
Install

Project	Version
✓ Lib1	6.2.0
✓ ConsoleApp1\ConsoleApp1.	6.1.0

Installed: multiple versions ins

Version: Latest stable 6.2.0

Consolidate Tab

Summary 1-2

- Developers should start testing an application in the debug mode, as it provides comprehensive details while building the application.
- The Visual Studio debugger allows inspecting what the targeted program does while it is executing.
- In the debug mode, breakpoints as special indicators inform the debugger to stop the program's execution when it arrives at a targeted line of code.
- Visual Studio allows executing the current line and then move to the next one using the **Debug** → **Step Over** option.
- The **Debug** → **Step Into** option allows executing each line of code within a method or functionality for troubleshooting it.
- The **Autos**, **Locals**, and **Immediate** windows allow viewing the status and values of variables and objects while debugging.

Summary 2-2

- The Run to Click feature is useful for debugging from one line to another in a block by simply clicking the icon rather than inserting breakpoints.
- NuGet is a free extension in Visual Studio 2017 that allows integrating the third-party packages in one to two clicks in a project.
- A NuGet package is a ZIP file containing all compiled code, a manifest file, and some more files associated with the code.
- In Visual Studio 2017, the integrated NuGet Package Manager installs, updates, and removes packages.