

Selenium con configuración de C#

Cómo configurar Visual Studio con Selenium controlador web:

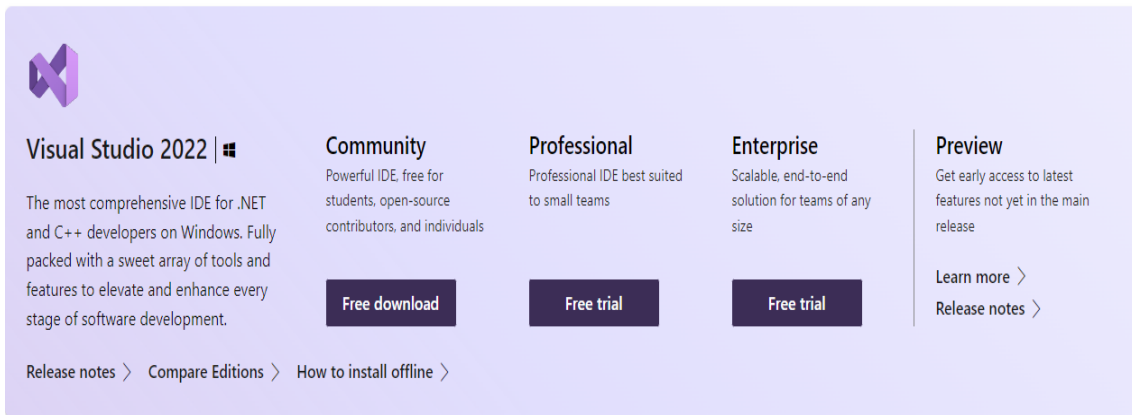
Ahora en este Selenium Tutorial de C#, aprendamos el proceso paso a paso para configurar Visual Studio con Selenium controlador web:

Visual Studio es un entorno de desarrollo integrado (IDE) que utilizan los desarrolladores para crear aplicaciones en múltiples plataformas como Windows, Android, iOS y aplicaciones basadas en la nube.


Paso 1) Navegar a la URL

<https://visualstudio.microsoft.com/downloads/> y haga clic en el '**Descarga gratis**' Botón que se muestra en la pestaña Comunidad de Visual Studio

Downloads



The screenshot shows the 'Downloads' section of the Visual Studio 2022 website. It features the Visual Studio logo and a list of download options for different editions: Community, Professional, Enterprise, and Preview. Each option includes a brief description and a button to download or try. The Community edition is highlighted as 'Free download', while Professional and Enterprise are 'Free trial'. The Preview section offers early access to latest features. At the bottom, there are links for 'Release notes', 'Compare Editions', and 'How to install offline'.

Visual Studio 2022 | 

The most comprehensive IDE for .NET and C++ developers on Windows. Fully packed with a sweet array of tools and features to elevate and enhance every stage of software development.

Community
Powerful IDE, free for students, open-source contributors, and individuals
[Free download](#)

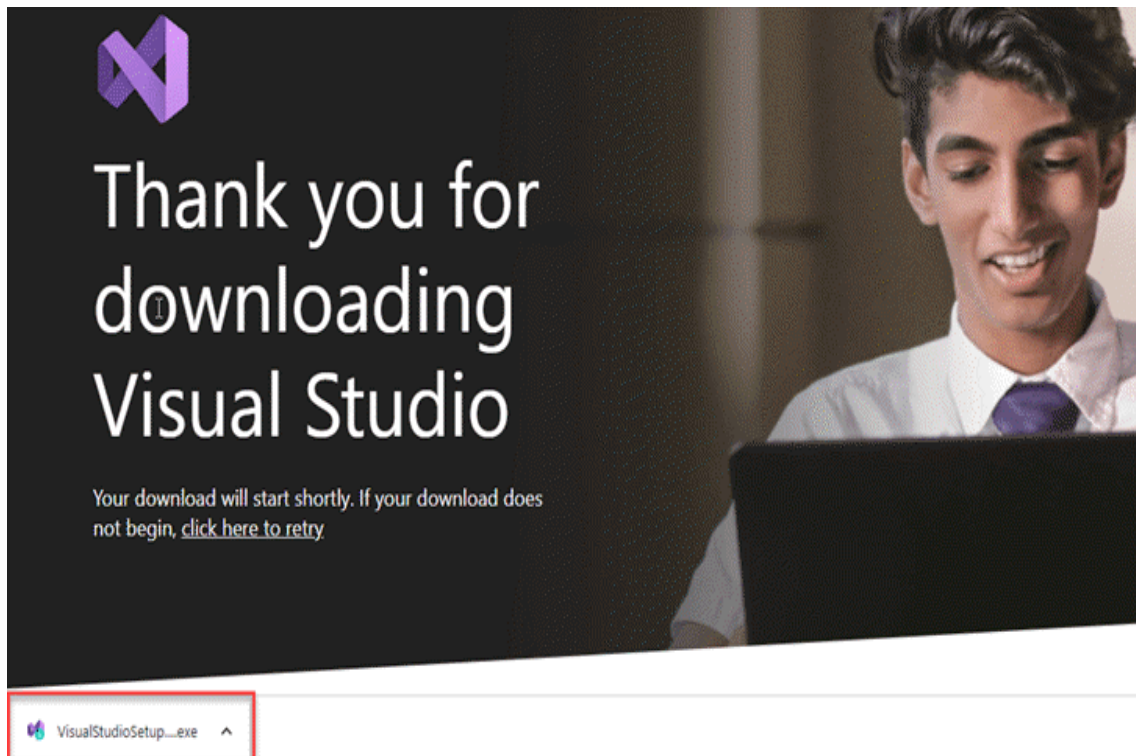
Professional
Professional IDE best suited to small teams
[Free trial](#)

Enterprise
Scalable, end-to-end solution for teams of any size
[Free trial](#)

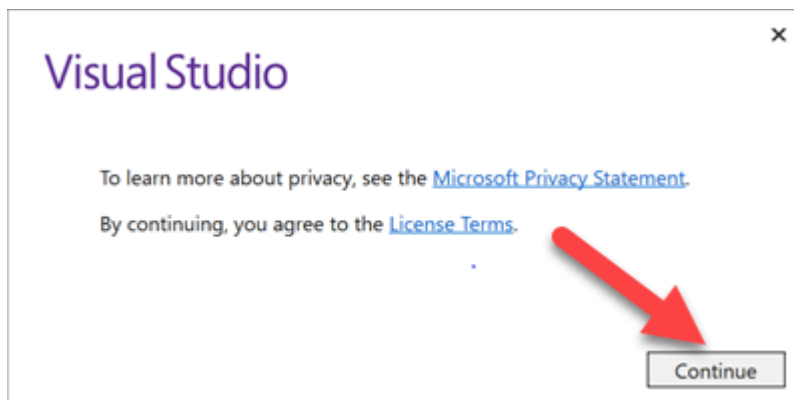
Preview
Get early access to latest features not yet in the main release
[Learn more >](#)
[Release notes >](#)

[Release notes >](#) [Compare Editions >](#) [How to install offline >](#)

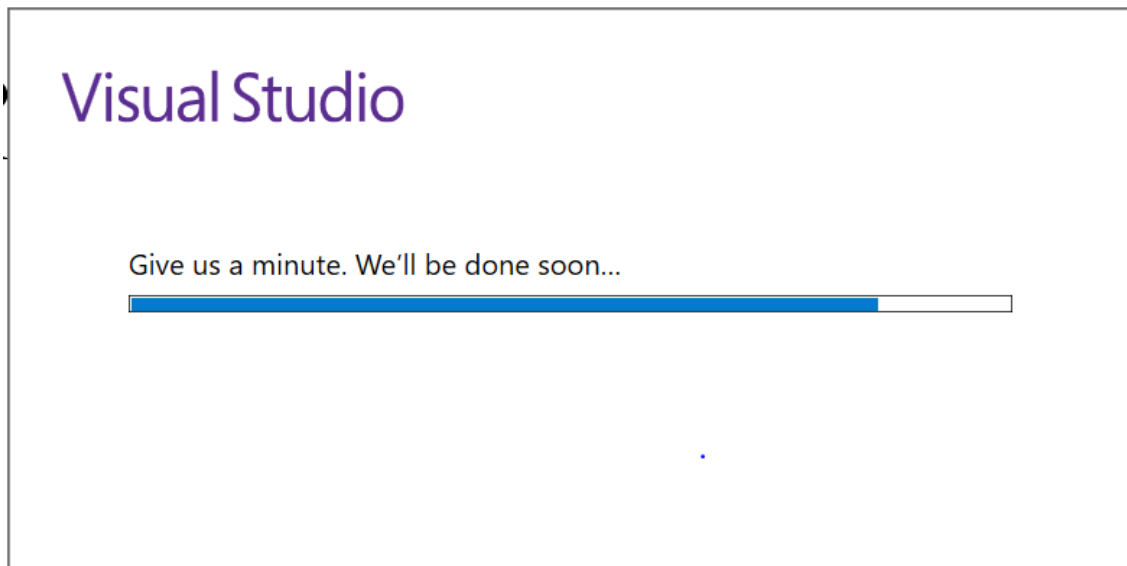
Paso 2) Abra el exe descargado. Haga clic en "Sí" si se le solicitan derechos de administrador.



Paso 3) Aparecerá la siguiente ventana emergente. Haga clic en el botón 'Continuar'.



Los archivos se descargarán como se muestra en la ventana emergente a continuación.

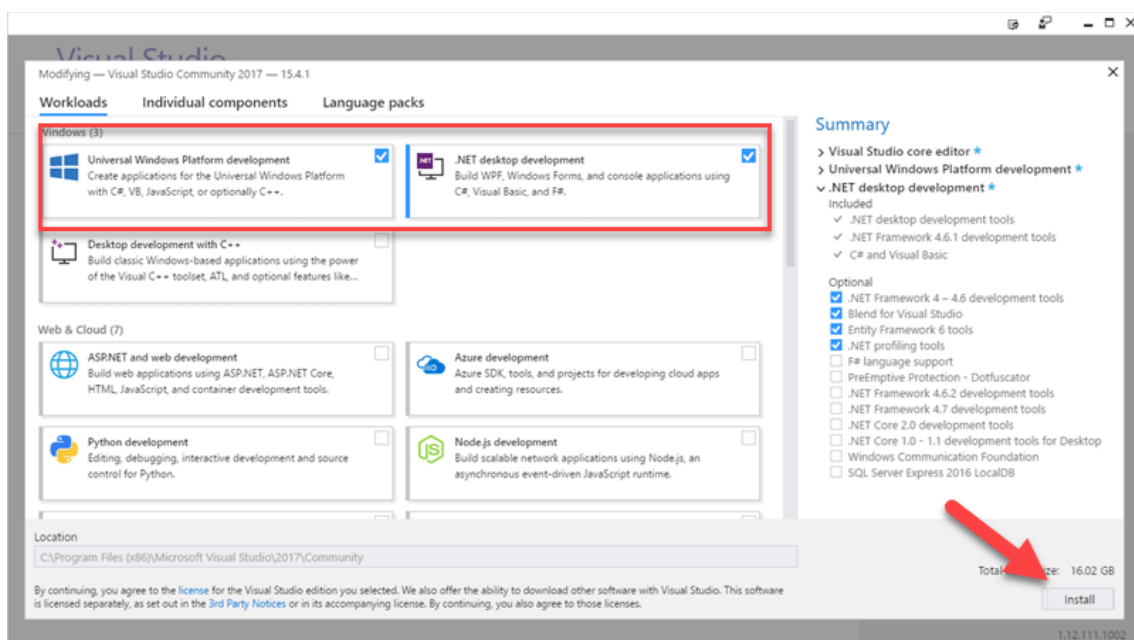


Paso 4) En la siguiente pantalla,

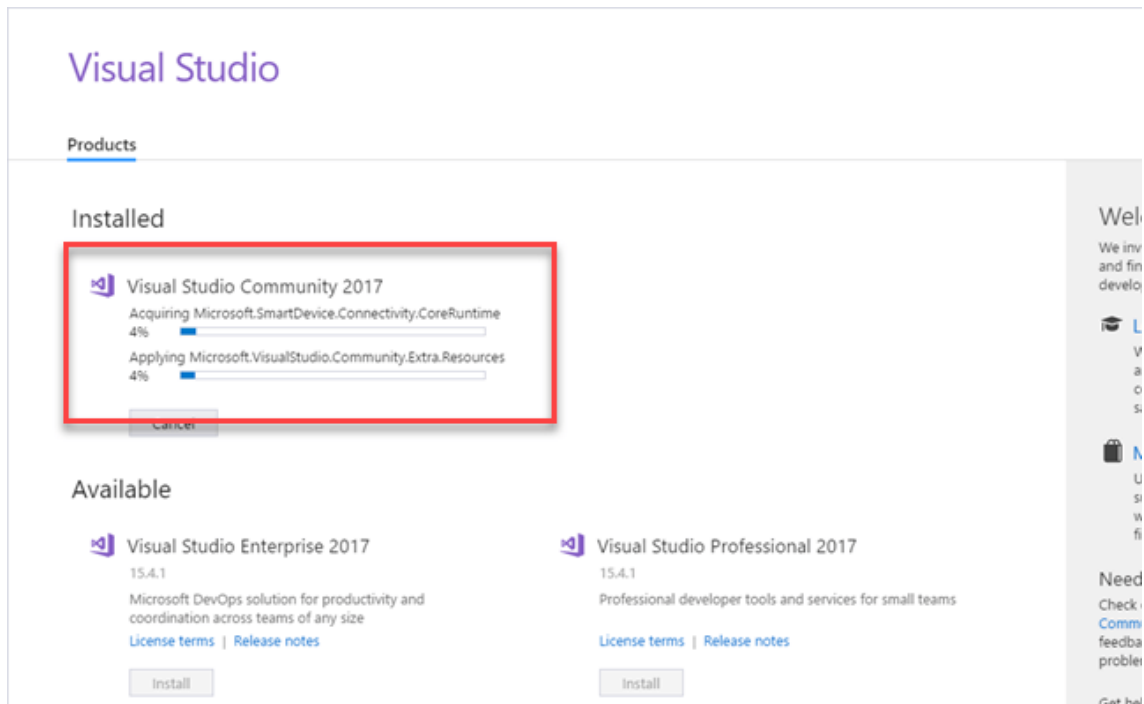
Seleccione las casillas de verificación para

- Universal Windows Desarrollo de plataforma
- Desarrollo de escritorio .Net

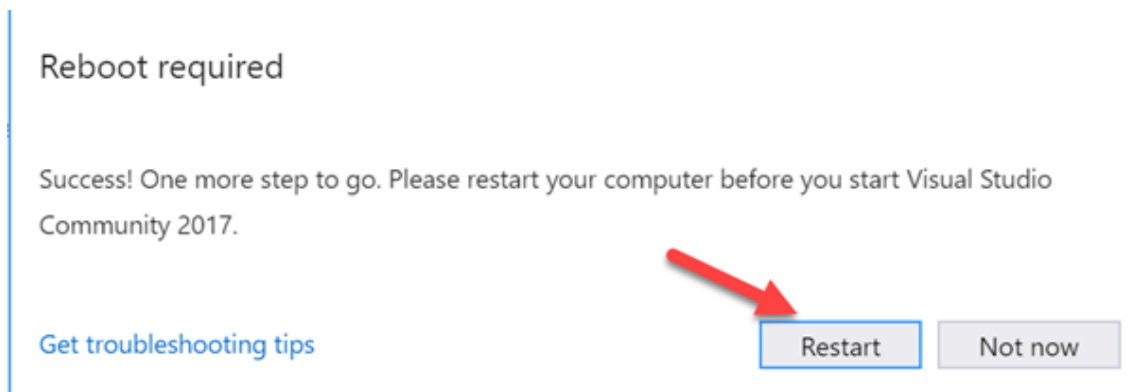
Haga clic en 'Instalar'.



Espera a que se complete la instalación de cada componente. Los archivos tienen un tamaño de 16 GB y llevarán tiempo.



Paso 5) Aparecerá la ventana emergente que aparece a continuación. Haga clic en el botón "Reiniciar".



Paso 6) Una vez que se reinicie la máquina, busque “Visual Studio” en el menú de inicio y haga clic en el resultado de la búsqueda. Aparecerá la siguiente ventana emergente. Haga clic en el enlace “Ahora no, tal vez más tarde” si no tiene una cuenta existente.

Visual Studio

Welcome!

Connect to all your developer services.

Sign in to start using your Azure credits, publish code to a private Git repository, sync your settings, and unlock the IDE.

[Learn more](#)

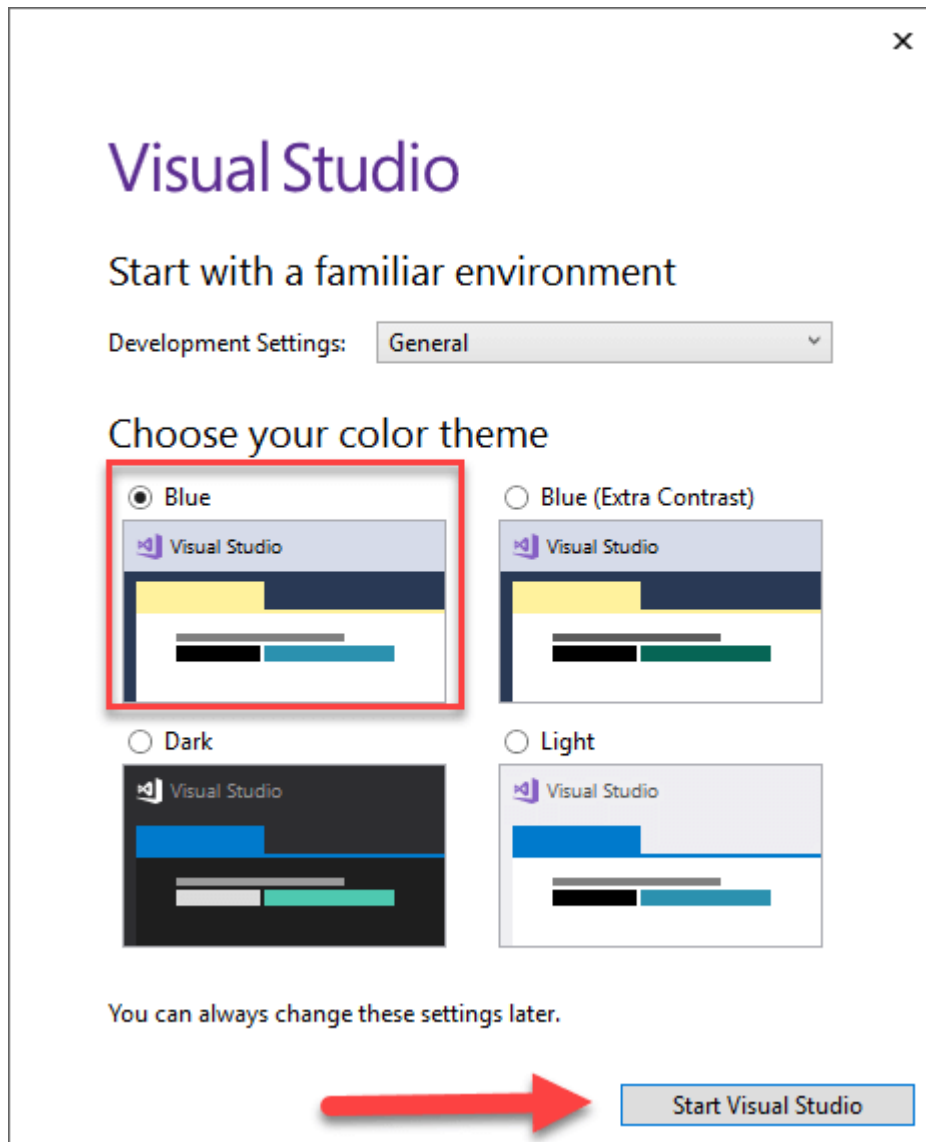
Sign in

Don't have an account? [Sign up](#)

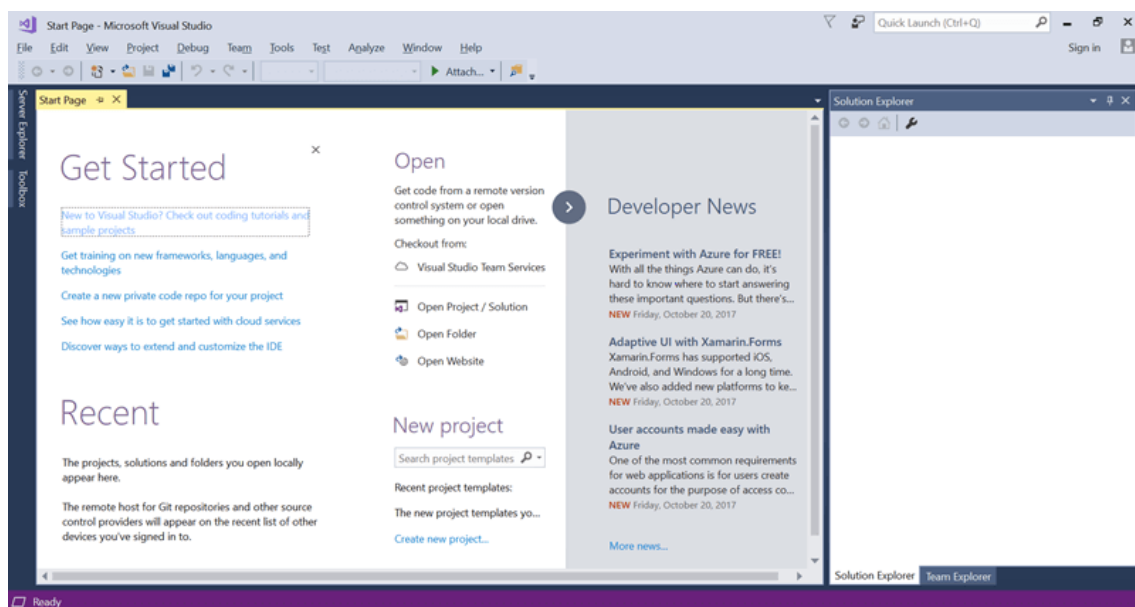
[Not now, maybe later.](#)

Paso 7) En la siguiente pantalla,

- Seleccione el tema de color de su agrado
- Haga clic en el botón “Iniciar Visual Studio”

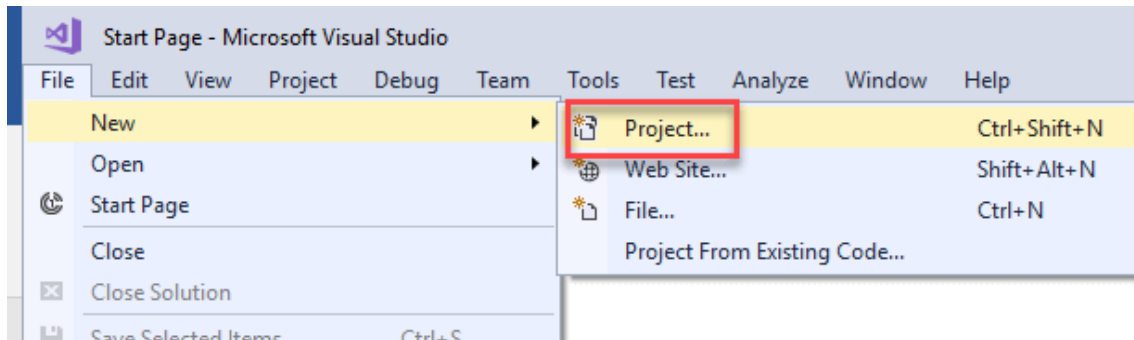


Paso 8) Aparecerá la pantalla 'Comenzar' de Visual Studio.



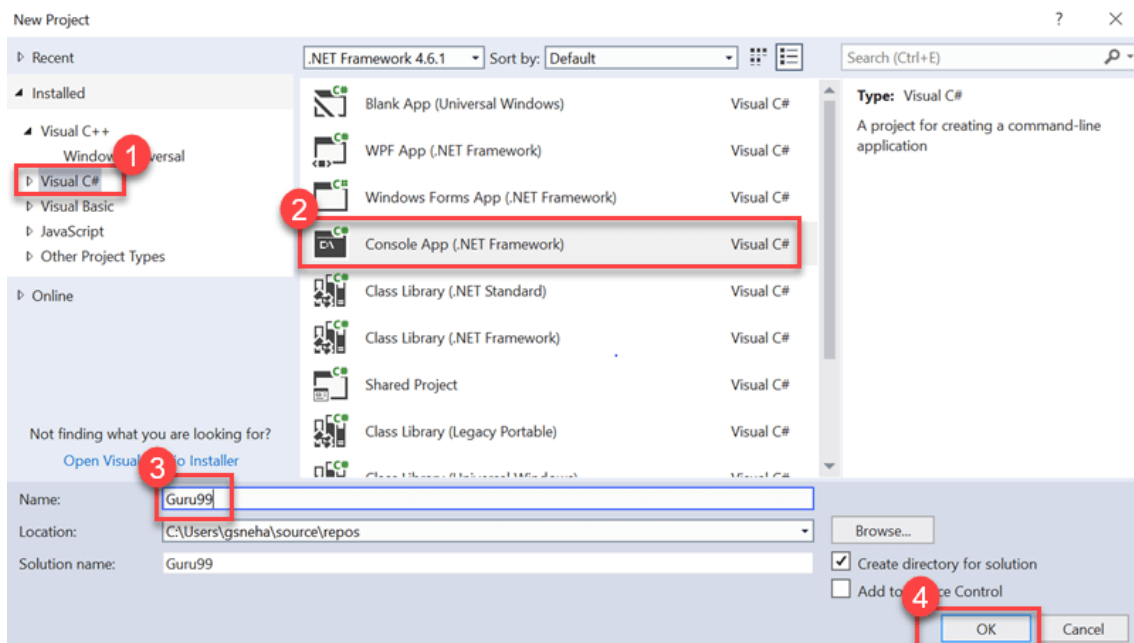
Cree un nuevo proyecto en Visual Studio:

Paso 1) En el menú Archivo, haga clic en Nuevo > Proyecto

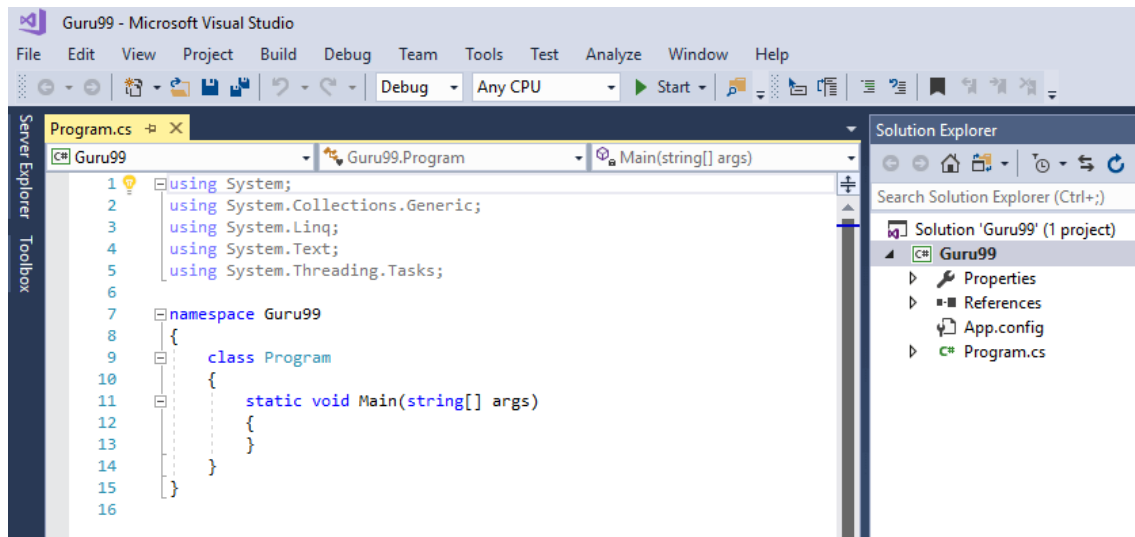


Paso 2) En la siguiente pantalla,

1. Seleccione la opción 'Visual C#'
2. Haga clic en Aplicación de consola (.NET Framework)
3. Ingrese el nombre como "Guru99"
4. haga clic en Aceptar

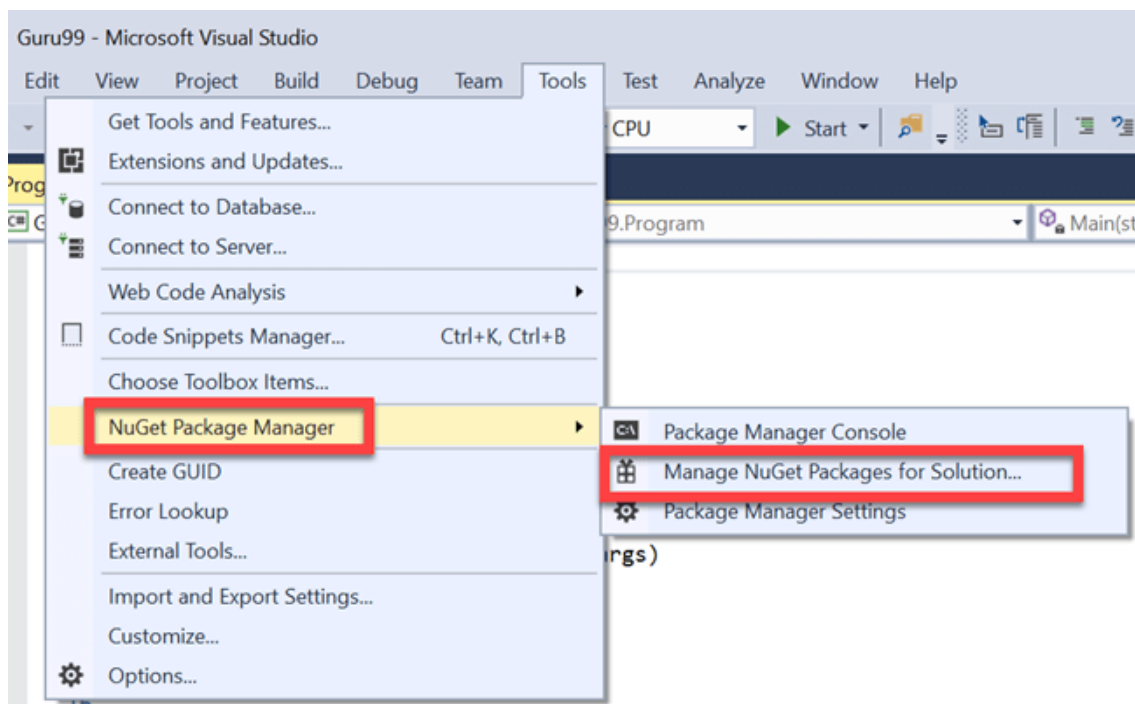


Paso 3) La siguiente pantalla se mostrará una vez que el proyecto se haya creado correctamente.



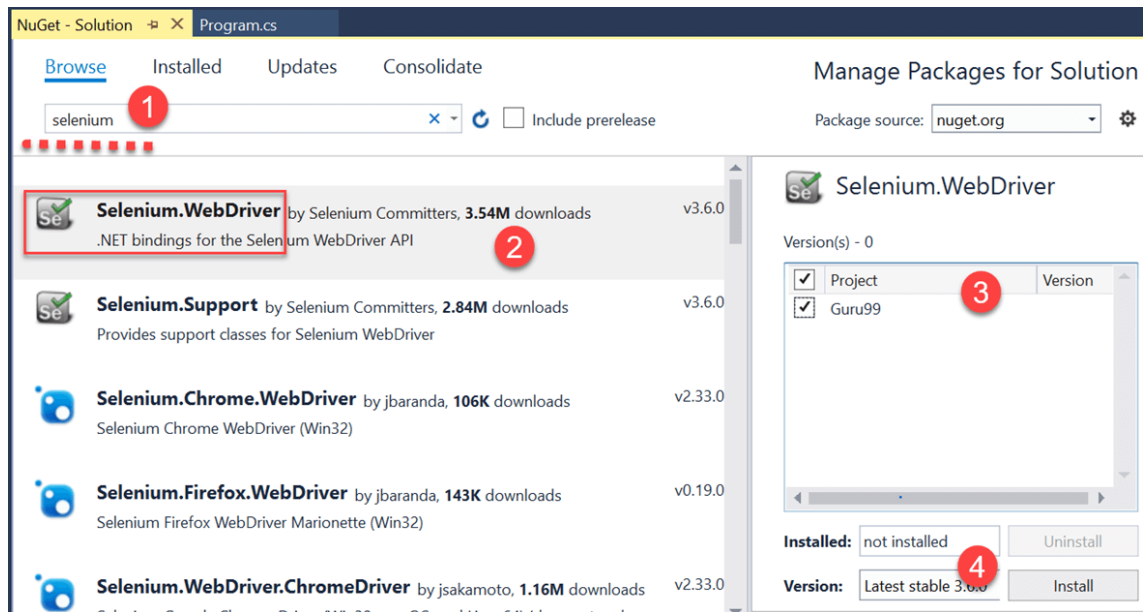
Configurar Visual Studio con Selenium controlador web:

Paso 1) Vaya a Herramientas -> NuGet Administrador de paquetes -> Administrar NuGet Paquetes de solución

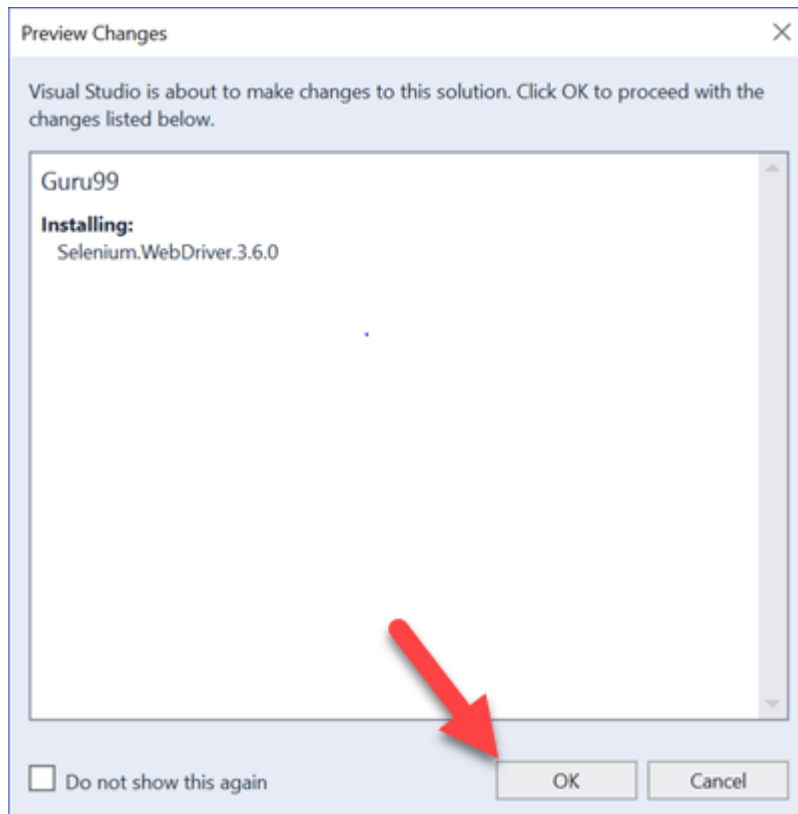


Paso 2) En la siguiente pantalla

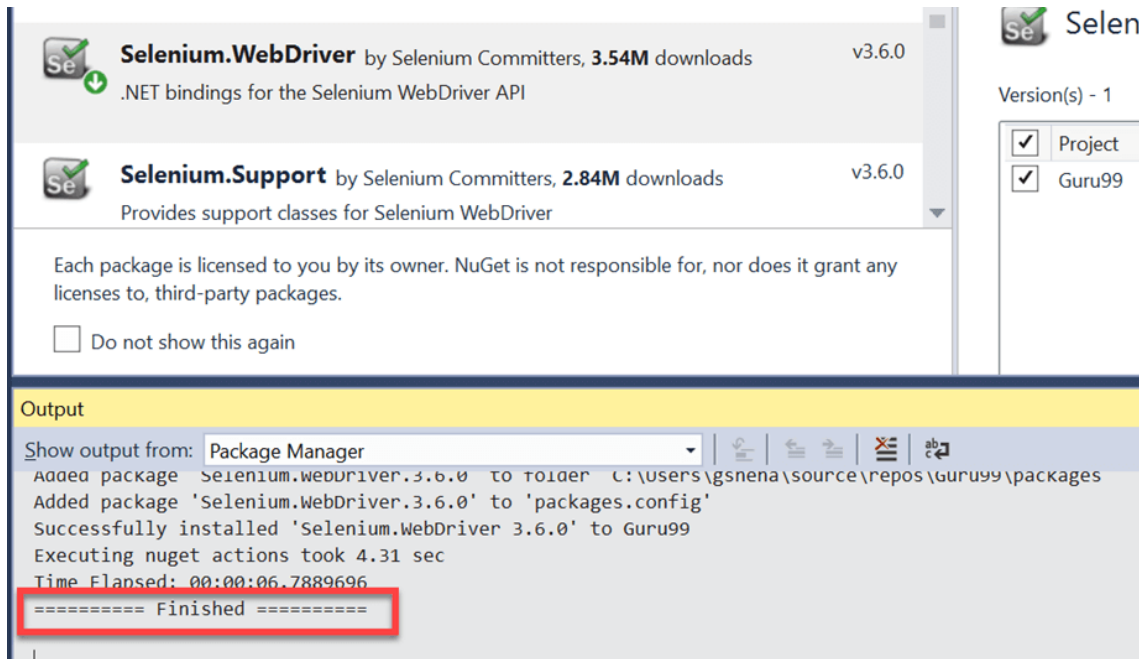
1. Busque Selenium en la pantalla resultante
2. Seleccione el primer resultado de búsqueda
3. Marque la casilla de verificación del proyecto
4. Haga clic en 'Instalar'



Paso 3) Haga clic en el botón "Aceptar" en la pantalla emergente.



Paso 4) El siguiente mensaje se mostrará una vez que el paquete se haya instalado correctamente.



Marco NUnit: descripción general

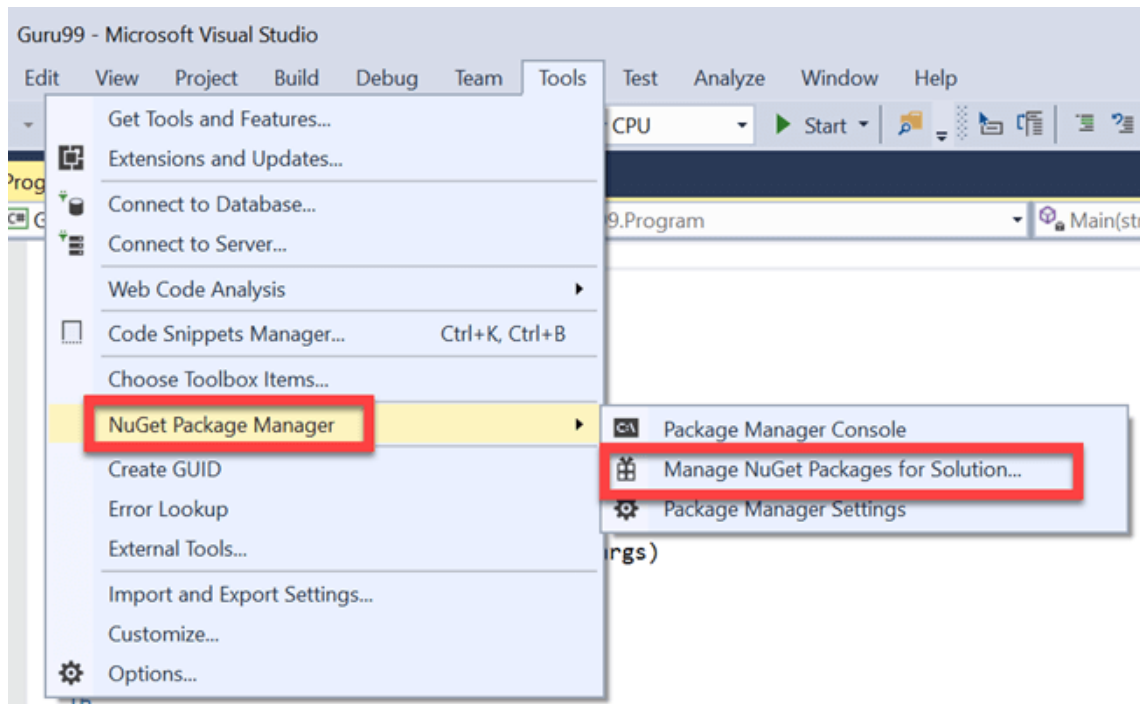
NUnit es un marco compatible con Visual Studio y Selenium Controlador web. NUnit es el marco de pruebas unitarias más utilizado para aplicaciones .Net. NUnit presenta los resultados de las pruebas en un formato legible y permite al evaluador depurar las pruebas automatizadas.

Necesitamos instalar NUnit Framework y NUnit Test Adapter en Visual Studio para poder usarlo.

Pasos para instalar NUnit Framework

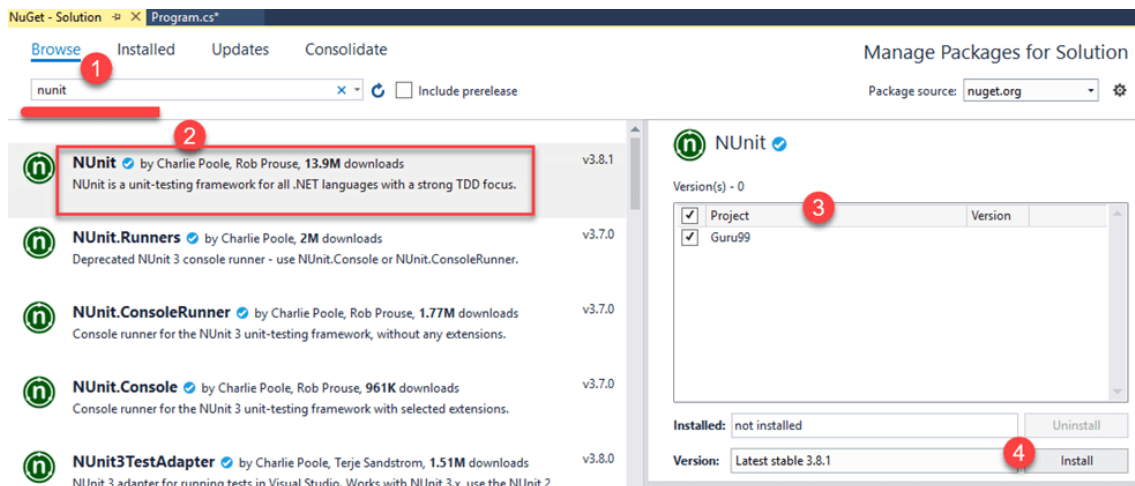
Ahora en este Selenium Con el tutorial de C#, aprendamos el proceso para instalar el marco NUnit:

Paso 1) Vaya a Herramientas -> NuGet Administrador de paquetes -> Administrar NuGet Paquetes de solución

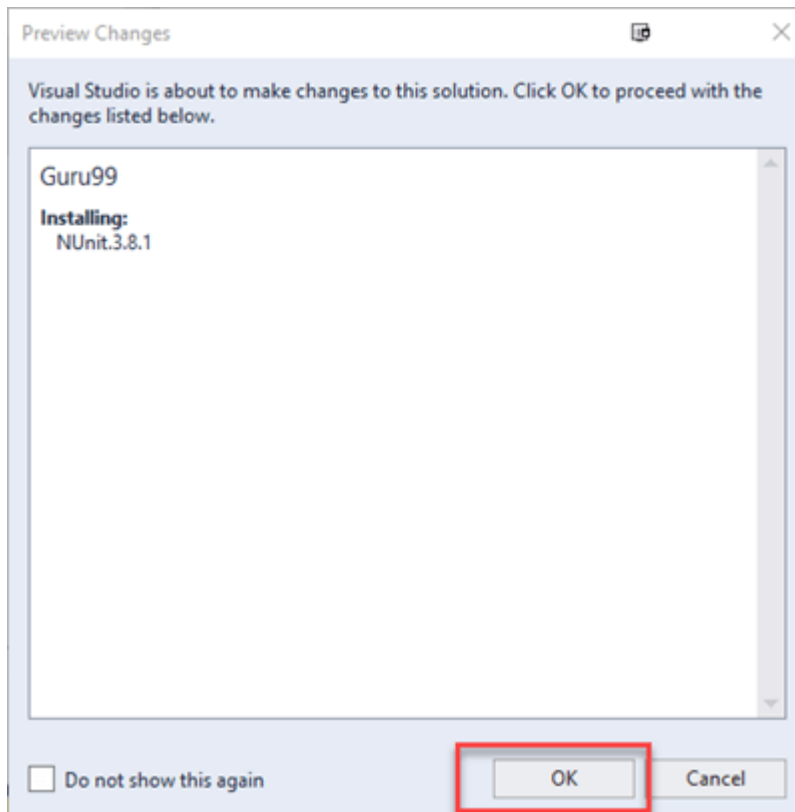


Paso 2) En la siguiente ventana

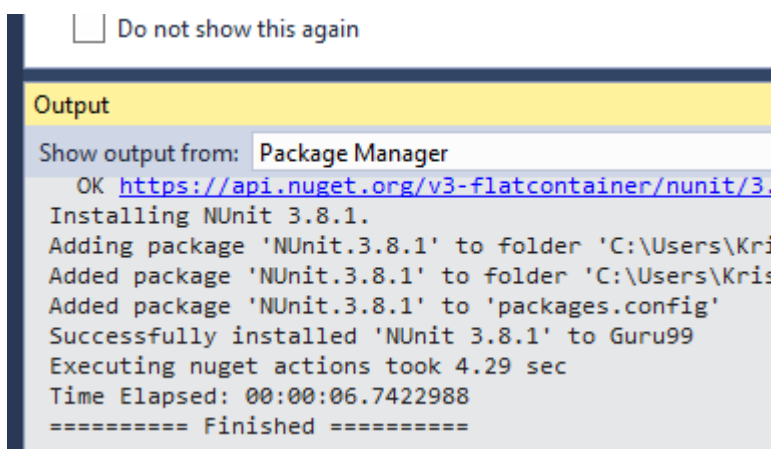
1. Buscar NUnit
2. Seleccione el resultado de la búsqueda
3. Seleccionar proyecto
4. Haga clic en Instalar



Paso 3) Aparecerá la siguiente ventana emergente. Haga clic en el botón "Aceptar".



Paso 4) El siguiente mensaje aparecerá una vez que se complete la instalación.



Pasos para descargar el adaptador de prueba NUnit

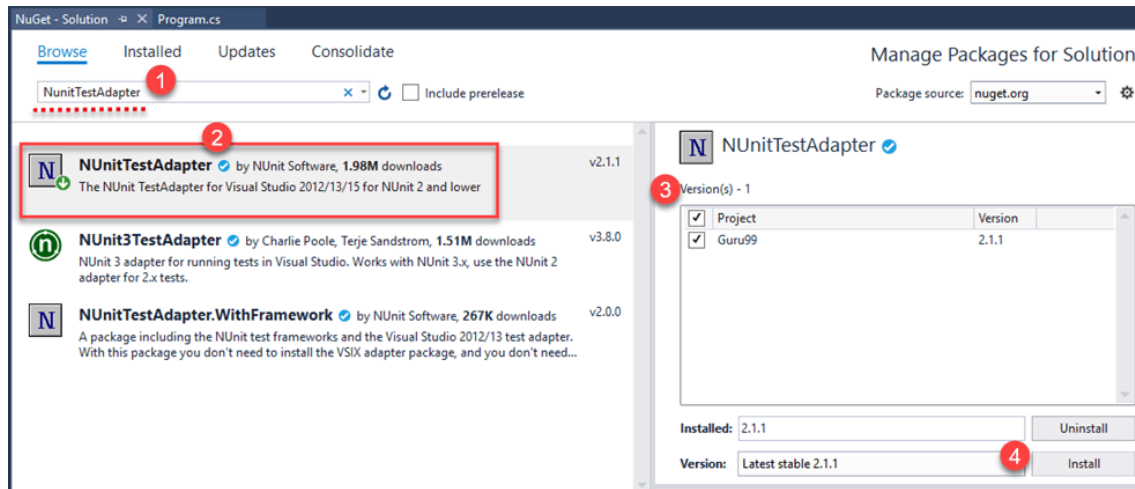
Ahora, en este tutorial de NUnit, cubriremos los pasos para descargar el Adaptador de prueba NUnit:

Tenga en cuenta que los pasos siguientes funcionan solo en equipos de 32 bits. En el caso de equipos de 64 bits, debe descargar el "Adaptador de prueba NUnit3" siguiendo el mismo proceso que se menciona a continuación.

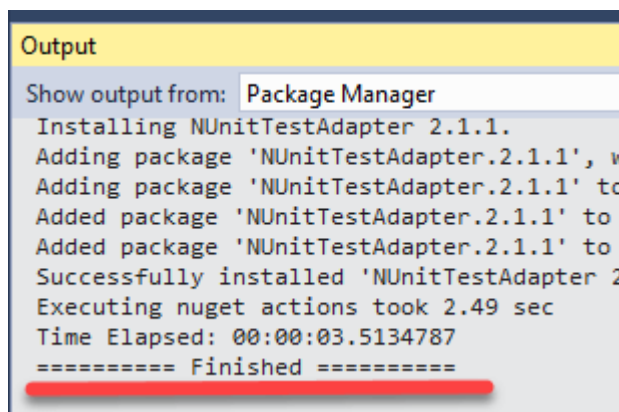
Paso 1) Vaya a Herramientas -> NuGet Administrador de paquetes -> Administrar NuGet Paquetes para solución. En esa pantalla,

1. Buscar NUnitTestAdapter
2. Haga clic en Resultado de búsqueda

3. Seleccionar proyecto
4. Haga clic en Instalar



Paso 2) Haga clic en Aceptar en la ventana emergente de confirmación. Una vez finalizada la instalación, verá el siguiente mensaje:



Selenium y marco NUnit

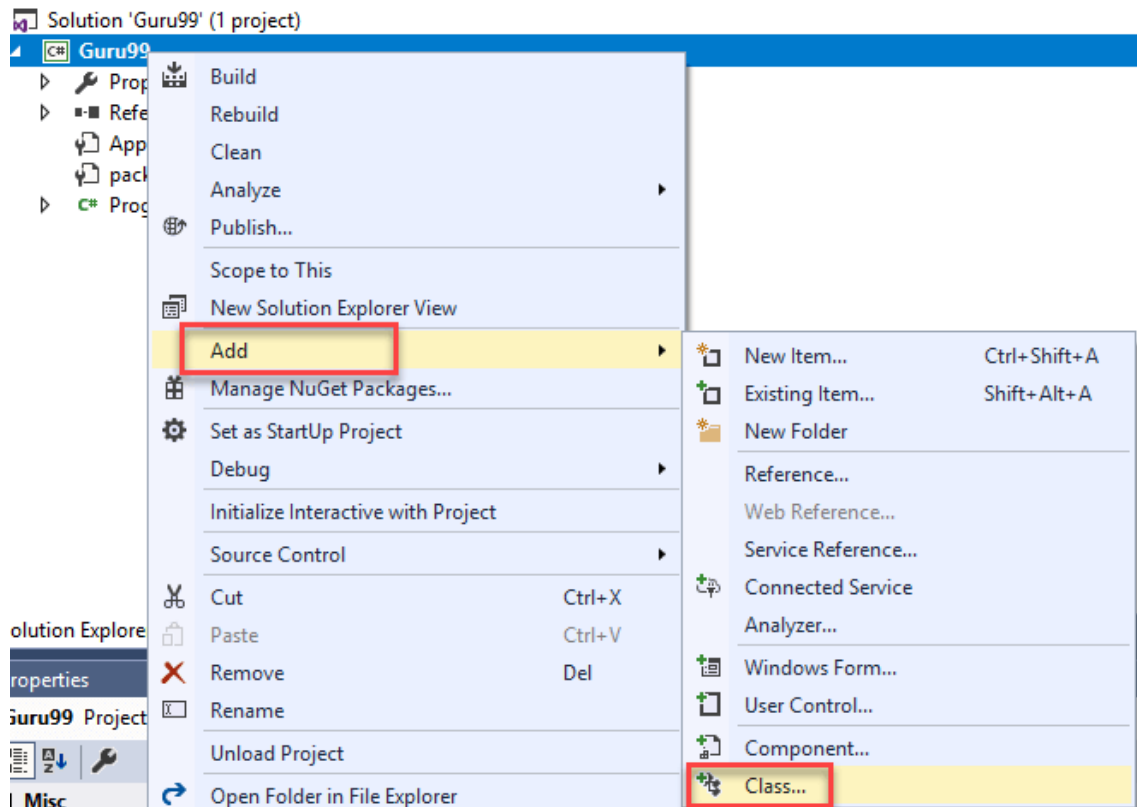
La integración de Selenium con el marco de trabajo NUnit permite que el evaluador pueda diferenciar entre varias clases de prueba. NUnit también permite que los evaluadores utilicen anotaciones como `SetUp`, `Test` y `TearDown` para realizar acciones antes y después de ejecutar la prueba.

El marco NUnit se puede integrar con Selenium creando una clase de prueba NUnit y ejecutando la clase de prueba usando el marco NUnit.

A continuación los pasos en este Selenium Se necesita un tutorial del marco C# para crear y ejecutar una clase de prueba utilizando el marco NUnit.

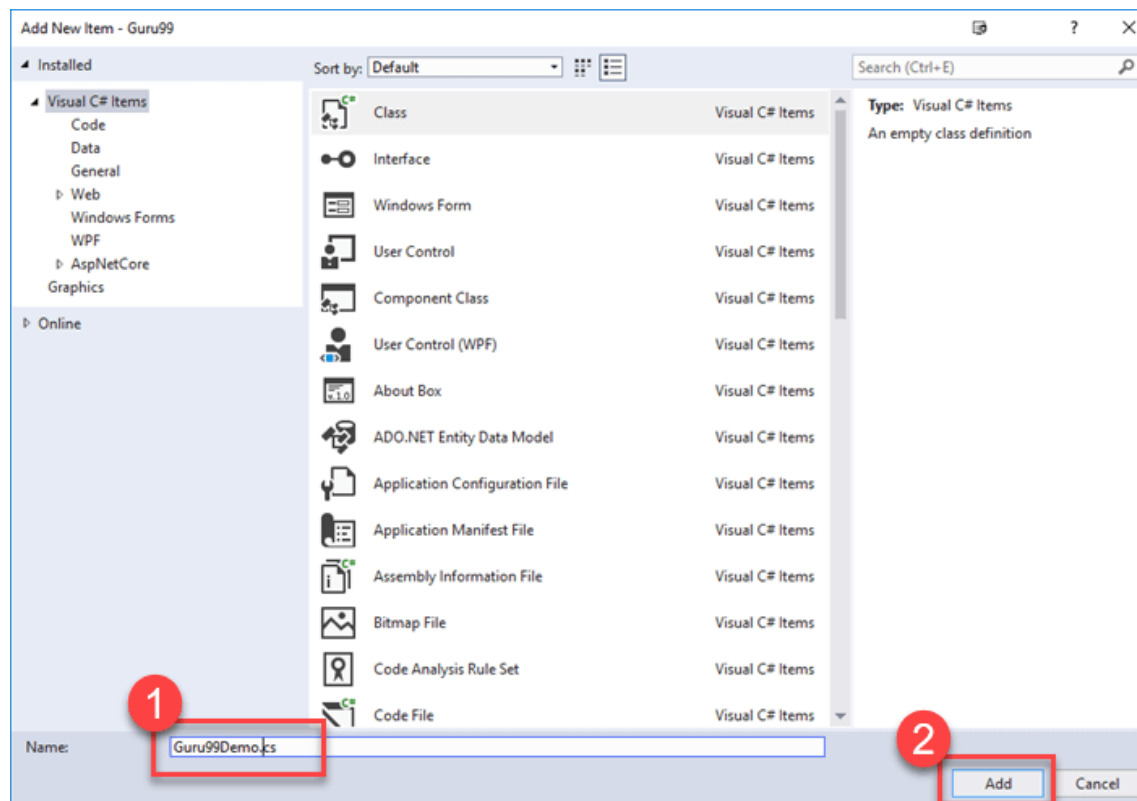
Pasos para crear una clase de prueba NUnit en Selenium:

Paso 1) En el Explorador de soluciones, haga clic derecho en proyecto > Agregar > Clase

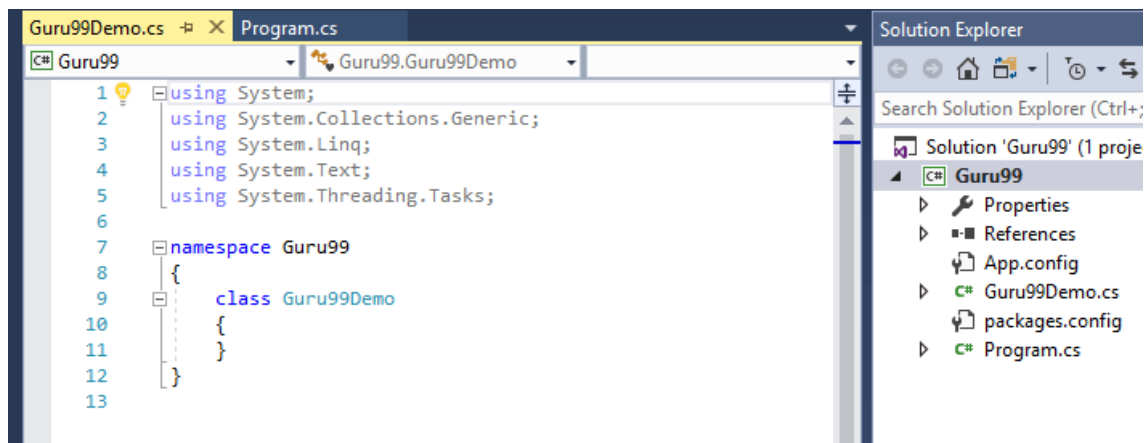


Paso 2) Aparecerá la ventana de creación de clases.

1. Proporcionar un nombre a la clase.
2. Haga clic en el botón Agregar



Paso 3) Aparecerá la siguiente pantalla.



Paso 4) Agregue el siguiente código a la clase creada. Tenga en cuenta que debe especificar la ubicación del archivo 'chromedriver.exe' durante la inicialización del controlador de Chrome.

```
using NUnit.Framework;
```

```
using OpenQA.Selenium;
```

```
using OpenQA.Selenium.Chrome;
```

```
using OpenQA.Selenium.Firefox;
```

```
using System;
```

```
using System.Collections.Generic;
```

```
using System.Linq;
```

```
using System.Text;
```

```
using System.Threading.Tasks;
```

```
namespace Guru99Demo
```

```
{
```

```
    class Guru99Demo
```

```
    {
```

```
        IWebDriver driver;
```

```
        [SetUp]
```

```
            public void startBrowser()
```

```
        {
```

```
            driver = new ChromeDriver("D:\\3rdparty\\chrome");
```

```

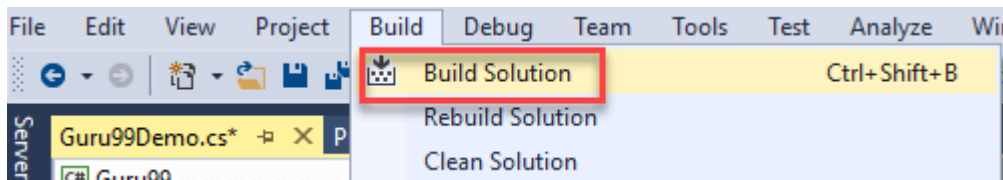
    }

[Test]
    public void test()
    {
        driver.Url = "http://www.google.co.in";
    }

[TearDown]
    public void closeBrowser()
    {
        driver.Close();
    }
}
}

```

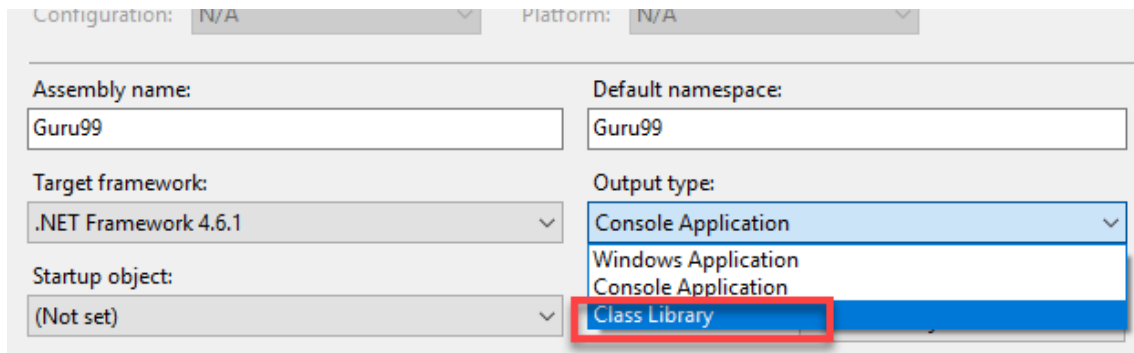
Paso 4) Haga clic en 'Crear' -> 'Crear solución'



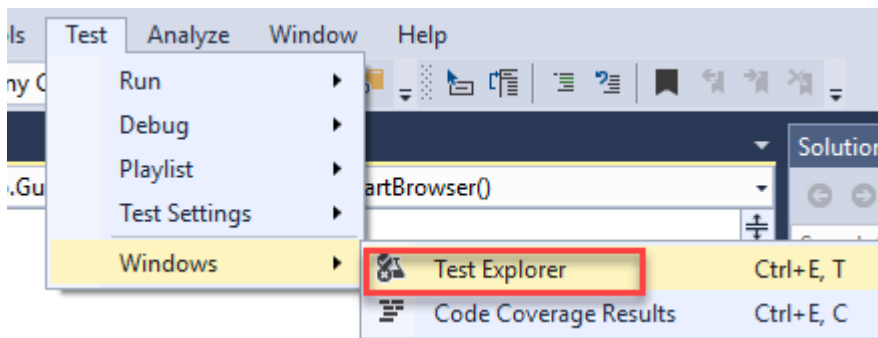
NOTA: Es posible que reciba un error como "No contiene un método 'principal' estático adecuado para un punto de entrada" cuando compila

| | Code | Description | Project |
|---|--------|---|---------|
| | | Program does not contain a static 'Main' method suitable for an entry point | Guru99 |
| ✖ | CS5001 | | |

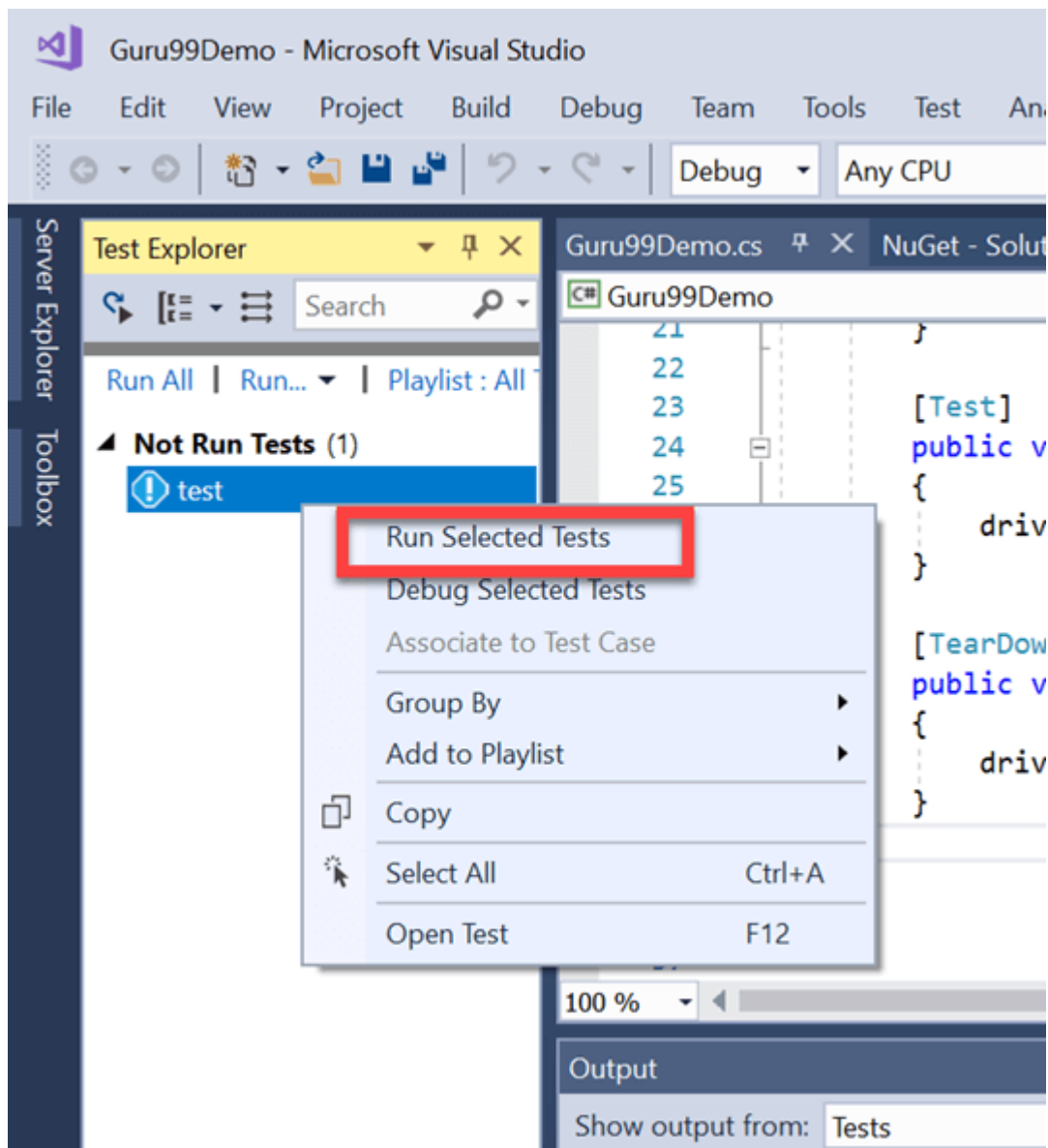
Para resolver esto, vaya a Proyecto > Propiedades y cambie el Tipo de salida a "Biblioteca de clases". El valor predeterminado es "Aplicación de consola".



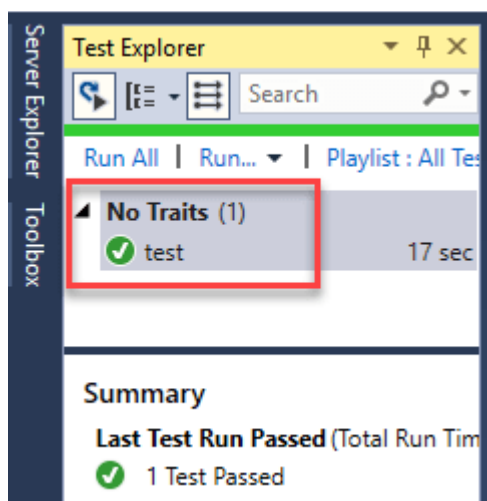
Paso 5) Una vez que la compilación sea exitosa, debemos abrir la ventana del Explorador de pruebas. Haga clic en Prueba -> Windows -> Explorador de pruebas



Paso 6) Se abre la ventana del Explorador de pruebas con la lista de pruebas disponibles. Haga clic derecho en Test Explorer y seleccione Ejecutar pruebas seleccionadas



Paso 7) Selenium debe abrir el navegador con la URL especificada y cerrarlo. El estado del caso de prueba se cambiará a "Aprobado" en la ventana del Explorador de pruebas.



AD

Selenium Comandos de WebDriver en C#

C# utiliza la interfaz 'IWebDriver' para las interacciones con el navegador. Las siguientes son las categorías de comandos disponibles con Selenium C^a#.

1. Comandos del navegador
2. Comandos de elementos web
3. Comandos desplegables

Ahora en el Selenium Tutorial de C Sharp, estudiemos el Selenium Comandos de WebDriver uno por uno

Comandos del navegador:

A continuación se muestra una lista de comandos del navegador disponibles en C# con Selenium.

| Nombre del comando | Descripción | Sintaxis |
|------------------------------------|--|--|
| Comando URL | Este comando se utiliza para abrir una URL específica en el navegador. | <code>driver.Url = "https://www.guru99.com"</code> |
| Comando de título | Este comando se utiliza para recuperar el título de la página web que está actualmente abierta | <code>String title = driver.Title</code> |
| Comando de origen de página | Este comando se utiliza para recuperar el código fuente de la página web que está actualmente abierta. | <code>String pageSource = driver.PageSource</code> |
| Cerrar comando | Este comando se utiliza para cerrar la instancia del navegador abierta recientemente. | <code>driver.Close();</code> |
| Salir del comando | Este comando se utiliza para cerrar todas las instancias abiertas del navegador. | <code>driver.Quit();</code> |
| Comando Atrás | Este comando se utiliza para navegar a la página anterior del historial del navegador. | <code>driver.Navigate().Back();</code> |
| Comando de avance | Este comando se utiliza para navegar a la siguiente página del historial del navegador. | <code>driver.Navigate().Forward()</code> |
| Actualizar comando | Este comando se utiliza para realizar la actualización del navegador. | <code>driver.Navigate().Refresh()</code> |

Comandos de elementos web

Un WebElement representa todos los elementos de una página web. Se representan mediante etiquetas HTML. Todos los botones, cuadros de texto, enlaces, imágenes, tablas y marcos se incluyen en Webelements. OperaLas interacciones con elementos web se pueden activar mediante la interfaz IWebElement. Para interactuar con un elemento web, necesitamos encontrar el elemento en la página web y luego realizar operaciones en él.

Se pueden utilizar herramientas como Firebug y Firepath para identificar el elemento. [Xpath](#) de Webelement.

La siguiente es la lista de comandos de Webelement disponibles en C#.

| Nombre del comando | Descripción | Sintaxis |
|--------------------------------|---|--|
| Haga clic en el comando | Este comando se utiliza para hacer clic en un elemento web. Para que se pueda hacer clic en el elemento, este debe estar visible en la página web. Este comando también se utiliza para operaciones con casillas de verificación y botones de opción. | <pre>IWebelement element = driver.FindElement(By.xpath("xpath of Webelement")); element.Click();</pre> |
| Borrar comando | Este comando se utiliza específicamente para borrar el contenido existente de los cuadros de texto. | <pre>IWebelement element = driver.FindElement(By.xpath("xpath of Webelement")); element.Clear();</pre> |
| Comando enviar claves | Este comando se utiliza para ingresar un valor en cuadros de texto. El valor que se ingresará debe pasarse como parámetro a | <pre>IWebelement element = driver.FindElement(By.xpath("xpath of Webelement")); element.SendKeys("guru99");</pre> |
| Comando mostrado | Este comando se utiliza para identificar si un elemento específico se muestra en la página web. Este comando devuelve un valor booleano; verdadero o falso según la visibilidad del elemento web. | <pre>IWebelement element = driver.FindElement(By.xpath("xpath of Webelement")); Boolean status = element.Displayed;</pre> |
| Comando habilitado | Este comando se utiliza para identificar si un elemento web en particular está habilitado en la página web. Este comando devuelve un valor booleano; verdadero o falso como resultado. | <pre>IWebelement element = driver.FindElement(By.xpath("xpath of Webelement")); Boolean status = element.Enabled;</pre> |

| Nombre del comando | Descripción | Sintaxis |
|-------------------------------------|--|--|
| Comando seleccionado | Este comando se utiliza para identificar si un elemento web en particular está seleccionado. Este comando se utiliza para casillas de verificación, botones de opción y operaciones de selección. | <pre>IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); Boolean status = element.Selected;</pre> |
| Enviar comando: | Este comando es similar al comando de clic. La diferencia radica en si el formulario HTML tiene un botón del tipo Enviar. Mientras que el comando de clic hace clic en cualquier botón, el comando de envío hace clic en los únicos botones con tipo enviar. | <pre>IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); element.submit();</pre> |
| Comando de texto | Este comando devuelve el texto interno de un elemento web. Este comando devuelve un valor de cadena como resultado. | <pre>IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); String text=element.Text;</pre> |
| Comando nombre de etiqueta | Este comando devuelve la etiqueta HTML de un elemento web. Devuelve un valor de cadena como resultado. | <pre>IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); String tagName = element.TagName;</pre> |
| Comando ObtenerCSSValue: | Este método se utiliza para devolver el color de un elemento web en forma de cadena rgba (rojo, verde, azul y alfa). | <pre>IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); String color = element.getCSSValue;</pre> <p>Salida– Si el color del elemento es rojo, la salida sería rgba(255,0,0,1)</p> |

Comandos desplegables:

Las operaciones desplegables en C# se pueden lograr utilizando la clase SelectElement.

Las siguientes son las distintas operaciones desplegables disponibles en C#.

| Nombre del comando | Descripción | Sintaxis |
|---|--|---|
| Comando SeleccionarPorTexto | Este comando selecciona una opción de un menú desplegable según el texto de la opción. | <pre> IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); SelectElement select = new SelectElement(element); select.SelectByText("Guru99"); </pre> |
| Comando SeleccionarPorÍndice | Este comando se utiliza para seleccionar una opción según su índice. El índice del menú desplegable comienza en 0. | <pre> IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); SelectElement select = new SelectElement(element); select.SelectByIndex("4"); </pre> |
| Comando SeleccionarPorValor | Este comando se utiliza para seleccionar una opción según su valor de opción. | <pre> IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); SelectElement select = new SelectElement(element); select.SelectByValue("Guru99"); </pre> |
| Comando de opciones | Este comando se utiliza para recuperar la lista de opciones que se muestran en un menú desplegable. | <pre> IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); SelectElement select = new SelectElement(element); List<IWebElement> options = select.Options; int size = options.Count; for(int i=0;i<options.size();i++) { String value = size.elementAt(i).Text; Console.WriteLine(value); } </pre> |

| Nombre del comando | Descripción | Sintaxis |
|-----------------------------------|--|--|
| | | El código anterior imprime todas las opciones en la consola dentro de un menú desplegable. |
| Comando es múltiple | Este comando se utiliza para identificar si un menú desplegable es un menú desplegable de selección múltiple; Un menú desplegable de selección múltiple permite al usuario seleccionar más de una opción en un menú desplegable a la vez. Este comando devuelve un valor booleano. | <pre> IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); SelectElement select = new SelectElement(element); Boolean status = select.IsMultiple(); </pre> |
| Comando DeSeleccionar Todo | Este comando se utiliza en menús desplegables de selección múltiple. Borra las opciones que ya han sido seleccionadas. | <pre> IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); SelectElement select = new SelectElement(element); select.DeSelectAll(); </pre> |
| Comando DeSelectByIndex | Este comando deselectiona un valor ya seleccionado usando su índice. | <pre> IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); SelectElement select = new SelectElement(element); select.DeSelectByIndex("4"); </pre> |
| Comando DeSelectByValue | Este comando deselectiona un valor ya seleccionado usando su valor. | <pre> IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); SelectElement select = new SelectElement(element); select.DeSelectByValue("Guru99"); </pre> |
| Comando DeSelectByText | Este comando deselectiona un valor ya seleccionado usando su texto. | <pre> IWebElement element = driver.FindElement(By.xpath("xpath of Webelement")); </pre> |

| Nombre del comando | Descripción | Sintaxis |
|--------------------|-------------|---|
| | | SelectElement select = new SelectElement(element); select.DeSelectByText("Guru99"); |

Ejemplos de código

Ejemplo 1: haga clic en un enlace usando XPATH Locator:

Escenario de prueba:

1. Vaya a la página web Demo Guru99 – <https://demo.guru99.com/test/guru99home/>
2. Maximiza la ventana
3. Haga clic en el menú 'Pruebas'
4. Cerrar el navegador

```
using NUnit.Framework;
```

```
using OpenQA.Selenium;
```

```
using OpenQA.Selenium.Chrome;
```

```
using System;
```

```
using System.Collections.Generic;
```

```
using System.Linq;
```

```
using System.Text;
```

```
using System.Threading.Tasks;
```

```
namespace Guru99Demo
```

```
{
```

```
    class CSS
```

```
    {
```

```
        IWebDriver m_driver;
```

```
        [Test]
```

```
            public void cssDemo()
```

```
            {
```

```
                m_driver = new ChromeDriver("D:\\3rdparty\\chrome");
```

```
                m_driver.Url = "https://demo.guru99.com/test/guru99home/";
```



```

        m_driver.Manage().Window.Maximize();

        IWebElement link = m_driver.FindElement(By.XPath("//*[@id='rt-
header']/div[2]/div/ul/li[2]/a"));

        link.Click();

        m_driver.Close();

    }

}

}

```

Ejemplo 2: ingresar datos en textBox y haga clic en un botón usando el localizador XPATH:

Escenario de prueba:

1. Navegue a la página de demostración de Guru 99
– <https://demo.guru99.com/test/guru99home/>
2. Introduzca datos en el cuadro de texto del correo electrónico
3. Haga clic en el botón de registro

```

using NUnit.Framework;

using OpenQA.Selenium;

using OpenQA.Selenium.Chrome;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Guru99Demo
{
    class CSS
    {
        IWebDriver m_driver;

        [Test]

        public void cssDemo()
    }
}

```

```

{
    m_driver = new ChromeDriver("G:\\");
    m_driver.Url = "https://demo.guru99.com/test/guru99home/";
    m_driver.Manage().Window.Maximize();

    // Store locator values of email text box and sign up button

    IWebElement emailTextBox =
m_driver.FindElement(By.XPath(".*[@id='philadelphia-field-email']"));

    IWebElement signUpButton =
m_driver.FindElement(By.XPath(".*[@id='philadelphia-field-submit']"));

    emailTextBox.SendKeys("test123@gmail.com");
    signUpButton.Click();

}
}
}

```

Ejemplo 3: ingresar datos en textBox y haga clic en un botón usando el localizador CSS:

Escenario de prueba:

1. Navegue a la página de demostración de Guru 99
– <https://demo.guru99.com/test/guru99home/>
2. Introduzca datos en el cuadro de texto del correo electrónico
3. Haga clic en el botón de registro

```

using NUnit.Framework;
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

```

```
using System.Threading.Tasks;
```

```
namespace Guru99Demo
```

```
{
```

```
    class CSS
```

```
    {
```

```
        IWebDriver m_driver;
```

```
        [Test]
```

```
            public void cssDemo()
```

```
            {
```

```
                m_driver = new ChromeDriver("G:\\");
```

```
                m_driver.Url = "https://demo.guru99.com/test/guru99home/";
```

```
                m_driver.Manage().Window.Maximize();
```

```
                // Store locator values of email text box and sign up button
```

```
                IWebElement emailTextBox =  
m_driver.FindElement(By.CssSelector("input[id=philadelphia-field-email]));
```

```
                IWebElement signUpButton =  
m_driver.FindElement(By.CssSelector("input[id=philadelphia-field-submit]));
```

```
                emailTextBox.SendKeys("test123@gmail.com");
```

```
                signUpButton.Click();
```

```
            }
```

```
        }
```

```
    }
```

Ejemplo 4: seleccione un valor en el menú desplegable:

Escenario de prueba:

1. Navegue a la página de demostración de Guru 99
– <https://demo.guru99.com/test/guru99home/>
2. Haga clic en SAP enlace
3. Introduzca datos en los cuadros de texto de nombre y correo electrónico
4. Seleccione un valor del menú desplegable Curso
5. Cerrar el navegador

```
using NUnit.Framework;
```

```
using OpenQA.Selenium;
```

```
using OpenQA.Selenium.Chrome;
```

```
using OpenQA.Selenium.Support.UI;
```

```
namespace Guru99Demo
```

```
{
```

```
    class TestSelect
```

```
    {
```

```
        IWebDriver m_driver;
```

```
        [Test]
```

```
            public void selectDemo()
```

```
            {
```

```
                m_driver = new ChromeDriver("G:\\");
```

```
                m_driver.Url = "https://demo.guru99.com/test/guru99home/";
```

```
                m_driver.Manage().Window.Maximize();
```

```
                IWebElement course = m_driver.FindElement(By.XPath(".*[@id='awf_field-91977689']"));
```

```
                var selectTest = new SelectElement(course);
```

```
                // Select a value from the dropdown
```

```
selectTest.SelectByValue("sap-abap");

    }

}

}
```

Resumen

- Para usar Selenium Para utilizar WebDriver con C# es necesario instalar Visual Studio.
- NUnit es el marco de pruebas unitarias compatible con Visual Studio y [Selenium controlador web](#)
- Necesitamos instalar NUnitFramework y NUnit Test Adapter en Visual Studio para poder usarlo.
- El marco NUnit se puede integrar con Selenium creando una clase de prueba NUnit y ejecutando la clase de prueba usando el marco NUnit.
- NUnit también permite a los evaluadores utilizar anotaciones como SetUp, Test y TearDown para realizar acciones antes y después de ejecutar la prueba.
- Selenium Los comandos de WebDriver se pueden clasificar en comandos del navegador, comandos de elementos web y comandos desplegables.