



# UI Test Automation of Windows with Appium WinAppDriver

## Chapter 1

**Naeem Akram Malik**

[www.testautomationtv.com](http://www.testautomationtv.com)

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

# Table of Contents

Section	Title	Page No
	<u>Intended Audience</u>	1
	<u>What You'll Need for Practice</u>	1
	<u>Abbreviations Used in the Book</u>	2
	<u>Support &amp; Example Code</u>	3
	<u>About This Book</u>	4
	<u>About the Author</u>	5
1	<u>Appium Architecture</u>	6
2	<u>Getting Ready</u>	8
3	<u>Enable Windows 10 Developer Mode</u>	9
4	<u>Installing Visual Studio Community Edition</u>	11
5	<u>Installing Node.Js and Appium</u>	14
6	<u>Installing WinAppDriver</u>	16
7	<u>How &amp; Why to Run WinAppDriver</u>	19
8	<u>Launch Notepad with Appium and WinAppDriver in C#</u>	22
9	<u>Project Setup [Visual Studio] and Launching an Application with WinAppDriver and Appium</u>	23
10	<u>Launching Legacy Application Notepad using AppiumOptions</u>	28
11	<u>Using WindowsDriver for Accessing Application Window</u>	32
12	<u>Demonstration</u>	35
13	<u>How to use AppiumService to Run Test Automation</u>	42

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

# Intended Audience

This book is intended for automation testers and developers who want to enhance their skills in web-based automation using Appium. It is assumed that you have basic knowledge of Windows application testing, Appium, Windows Application Driver (WinAppDriver), and programming in C# .Net.

## What You'll Need for Practice

You will need the software listed below to get the most out of this book. Section 1 will show you how to download & install all of these.

- 1) Windows 10 (**with Developer Mode enabled**)
- 2) Microsoft Visual Studio- Community Edition or higher with the following features:
  - .Net desktop development
  - Nuget package manager
  - Testing tools
- 3) Windows Application Driver
- 4) Node.JS
- 5) Appium

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

# Abbreviations Used in the Book

In this book, you will find a few styles of text that distinguish between different types of information provided.

Technical terms in text are shown as follows:

“After installation, run the command **WinAppDriver** to ensure that we are ready with Windows Application Driver.” A block of code is set as follows:

```
namespace RunNotePad
{
    class Program
    {
        static void Main(string[] args)
        {
            WindowsDriver<WindowsElement> notePadSession;
            AppiumOptions desiredCapabilities = new AppiumOptions();
            desiredCapabilities.AddAdditionalCapability("app",
                @"C:\Windows\System32\notepad.exe");

            notePadSession = new WindowsDriver<WindowsElement>(
                new Uri("http://127.0.0.1:4723"), desiredCapabilities);

            if (notePadSession == null)
            {
                Console.WriteLine("App not started.");
                return;
            }
        }
    }
}
```

Any command-line input or output is written as follows:

**android create avd -n <name of the AVD> -t <targetID>**

**New terms** and **important words** are shown in bold.

Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: “Click on **Advanced, system settings.**”

## Notes, Tips & Tricks:

Warnings or important notes appear in a box.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

# Support & the Example Code

You can download the example code files from the URL below  
<http://www.testautomationtv.com>

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

# About This Book

This book(and the [course](#)) is about UI automation and automated functional testing using WinAppDriver and C#.

Prior to WinAppDriver (Windows Application Driver), Microsoft Coded UI was the main UI automation tool sold by Microsoft. The Coded UI is now deprecated since 2019 and Microsoft is pushing ahead with WinAppDriver which is a free tool. WinAppDriver is clearly the future of UI automation on Microsoft platform.

In this book(and the [course](#)) you will learn how to perform automated testing of all sorts of Windows applications(supported by WinAppDriver i.e. WinForms, WPF, Win32, UWP).

Since WinAppDriver isn't very well known yet, I'm going to show you how to get started with it in a step by step fashion.

That's why I'll show you what tools do you need to install for test automation of Windows applications, everything from Visual Studio to WinAppDriver.

Right after installations, we'll get down to scripting. You'll learn how to get the title of an application, how to maximize a window, how to take screenshots and how to close an application.

You will be playing with UI automation scripts before the end of this book. It will be really quick, trust me.

My book (and the [course](#)) is going to be helpful for both experienced professionals and newbies.

Regards,  
**Naeem Akram Malik**

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

# About the Author

Naeem Akram Malik has been working as an Automation Engineer since 2009.

He has used many automated testing tools during the past decade. This list includes WinAppDriver, CodedUI, Selenium, Postman, and many others.

He is a person with many talents and interests, ranging from gardening to programming and poetry to cooking.

Right now he is working as Sr. Software Test Engineer. On his job, he uses Microsoft Coded UI, Selenium, and WinAppDriver for creating test automation scripts.

It is also a part of his job to maintain these automation scripts and run them via CI/CD using TFS. Apart from teaching, he is also teaching actively on Udemy, his core interests being, test automaton.

He has been writing computer software code (C++, C# .Net, Java, Dephi) since 2006, working on test automation since 2009, creating Android apps since 2012, and publishing online courses since 2014.

He has extensively worked on network communications, computer telephony integration(CTI) using Microsoft TAPI 2.x and 3.x, LDAP programming (Active Directory), and test automation.

He has been sharing his knowledge through his blogs too. He is confident that it's time to share more things that he has experienced with the world in even more formats including videos and e-books.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

## Section 1

# Appium Architecture

**Appium based Windows Application Driver (WinAppDriver)** is the newest automation testing tool for **UI Automation** and Software Functional Testing for testing desktop applications by **Microsoft**.

Microsoft Coded UI, the UI test automation licensed tool sold by Microsoft in the past, is now deprecated (since 2019). **Windows Application Driver(WinAppDriver)** is a free tool that provides APIs for many programming languages, including **C#. Net, Java, and Python**. This book shows you how to use it with C# .Net.

The **WinAppDriver** is based on Appium, which is based on Selenium, hence a community driver industry standard automation testing tool. **Appium** based **WinAppDriver** is entirely compliant with the WebDriver specifications.

### Notes & Tips:

Appium is based on WebDriver which is also the basis of Selenium, this way if you start from Appium/WinAppDriver you can easily move forward to Selenium based UI testing with little effort. Although, you will need to understand how the DOM based HTML websites work.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>



**WinAppDriver** provides the following things to facilitate **Automated Testing**:

1. **WinAppDriver.exe**: UI remote control, the executable which is capable of receiving Appium commands and performing actions on the UI of a given Windows PC for automation testing.
2. **WinAppDriver UI Recorder**: a lightweight UI inspection tool that allows you to find various properties of Windows UI elements. It can also generate XPath for various application UI elements.
3. API support for various programming languages, including C#, Java, Python, and many more through client APIs.

Appium is very popular in the automated tester community for mobile Automation. Appium for iOS and Android is already out there, Microsoft also joined the bandwagon by providing the necessary tooling to bring Windows application testing into the Appium world.

## Section 2

# Getting Ready

In this section, I am going to provide you **an overview of the prerequisites** of using Windows Application Driver.

We will enable the **Windows 10 developer mode** first.

**Windows 10 is a mandatory** requirement for running WinAppDriver.

WinAppDriver is **based on Appium** which is based on **Node.js**, we'll install both **Node.js** and **Appium**.

Windows Application Driver conforms to the standard **WebDriver Specification** and implements only the functionality specified in **WebDriver** specifications.

For writing C# .Net scripts, we will **install Visual Studio Community Edition**. It is a free programming IDE.

Behind the scenes, your UI automation scripts will use **http** to **communicate** to an instance of a WinAppDriver. You may do it yourself, but it is recommended to use the **Appium Nuget** package.

### Notes & Tips:

Windows Application Driver is also known as **WinAppDriver**.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

## Section 3

# Enable Windows 10 Developer Mode

WinAppDriver can be run in **Windows 10 developer** mode only. Let's **enable** it before proceeding.

I'll click the **Start** button and click the **Settings** icon

In the search textbox I'll type "**Developer**" and hit **enter**

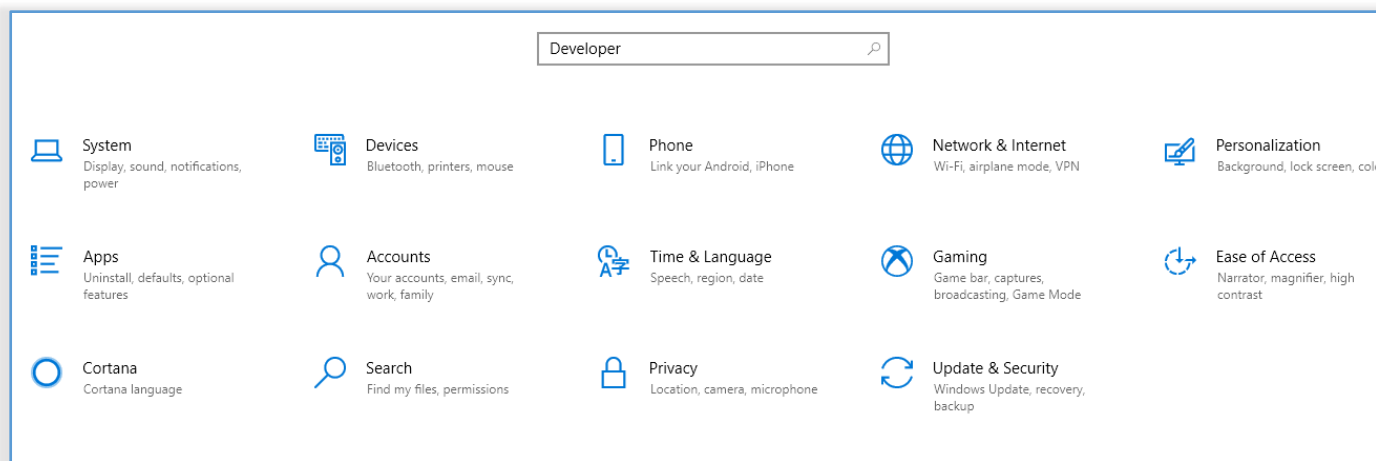


Figure 1

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

On the next screen, select the radio button, “**Developer Settings**”, select “**Developer Mode**” and click the “**Yes**” button on the next pop-up.

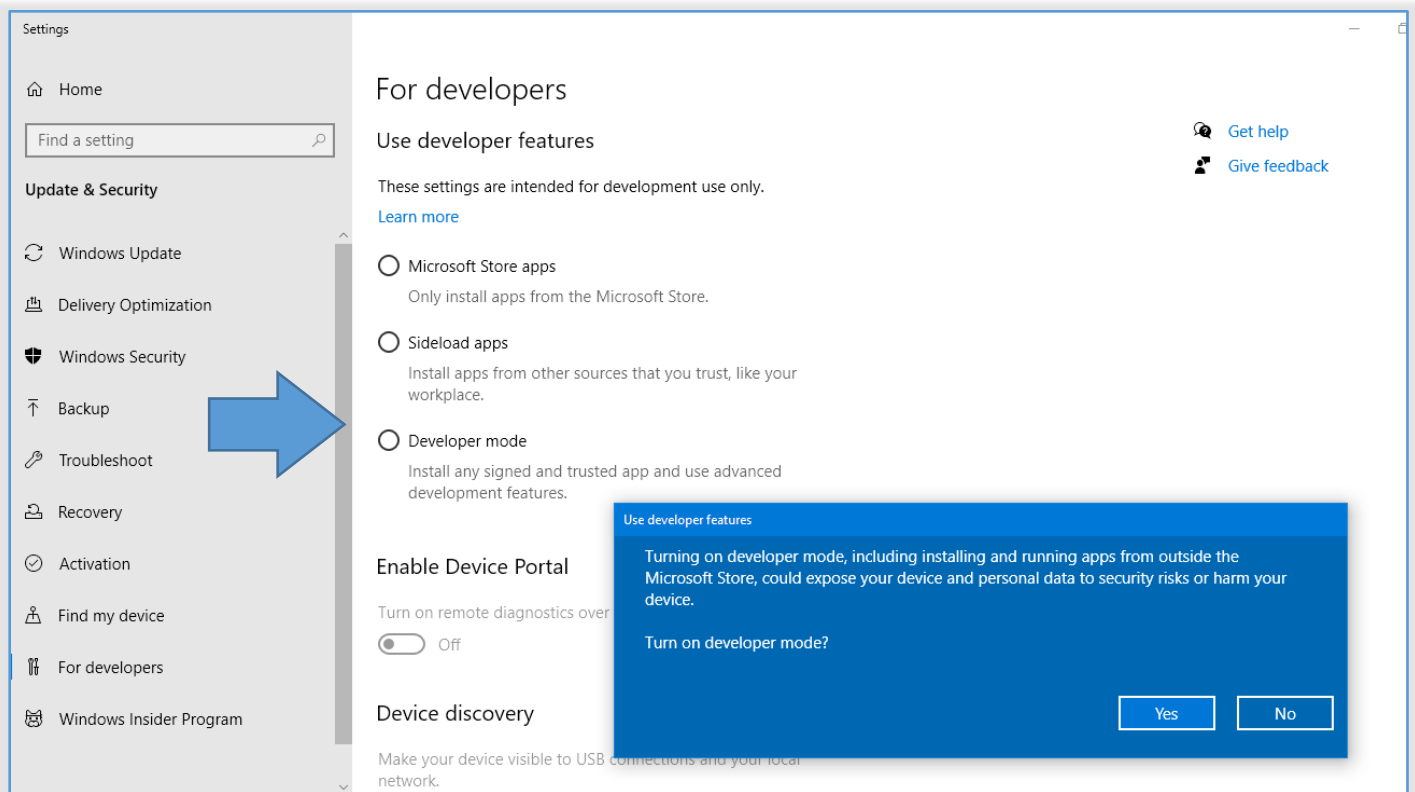


Figure 2

That's it, in the next section, we'll install Visual Studio .Net and other tools such as Node.js.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

## Section 4

# Installing Visual Studio Community Edition

The first thing you need to install is a version of Microsoft Visual Studio IDE. We will need the Visual Studio to write scripts in C#.Net programming language which we're going to use in this book.

### Notes & Tips:

IDE means Integrated Development Environment.

Microsoft VS Community Edition is a free set of software development tools. You may use a higher version of Visual Studio if you have the license. Otherwise, Visual Studio community edition is sufficient for this course.

Visual Studio community edition can be downloaded from the following URL:

<https://visualstudio.microsoft.com/vs/community/>

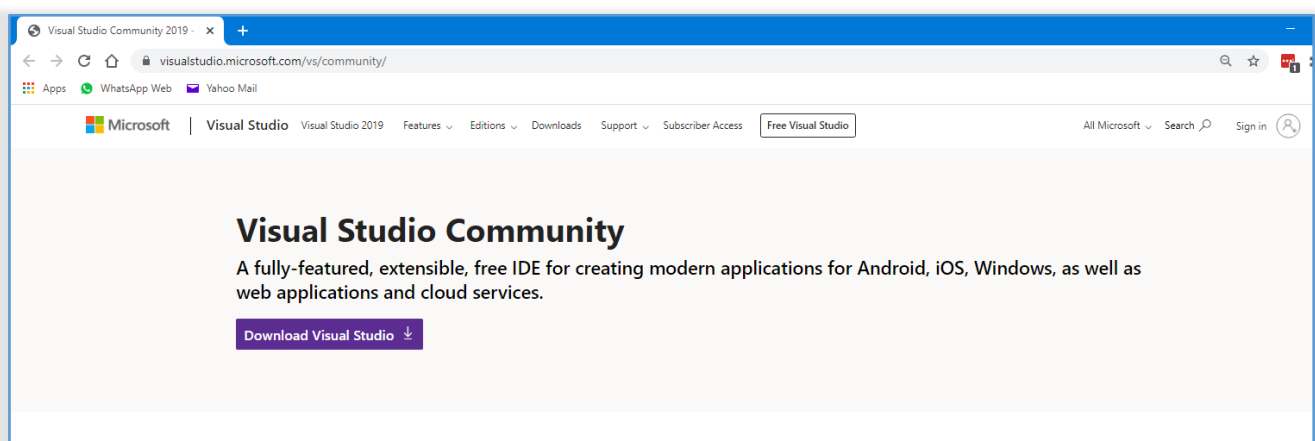


Figure 3

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

Once you've opened the URL, click the **Download Visual Studio** button. After downloading the VS Community Edition Setup, run it and make sure you select the following options in the setup.

- ✓ .Net desktop development
- ✓ Nuget package manager
- ✓ Testing tools

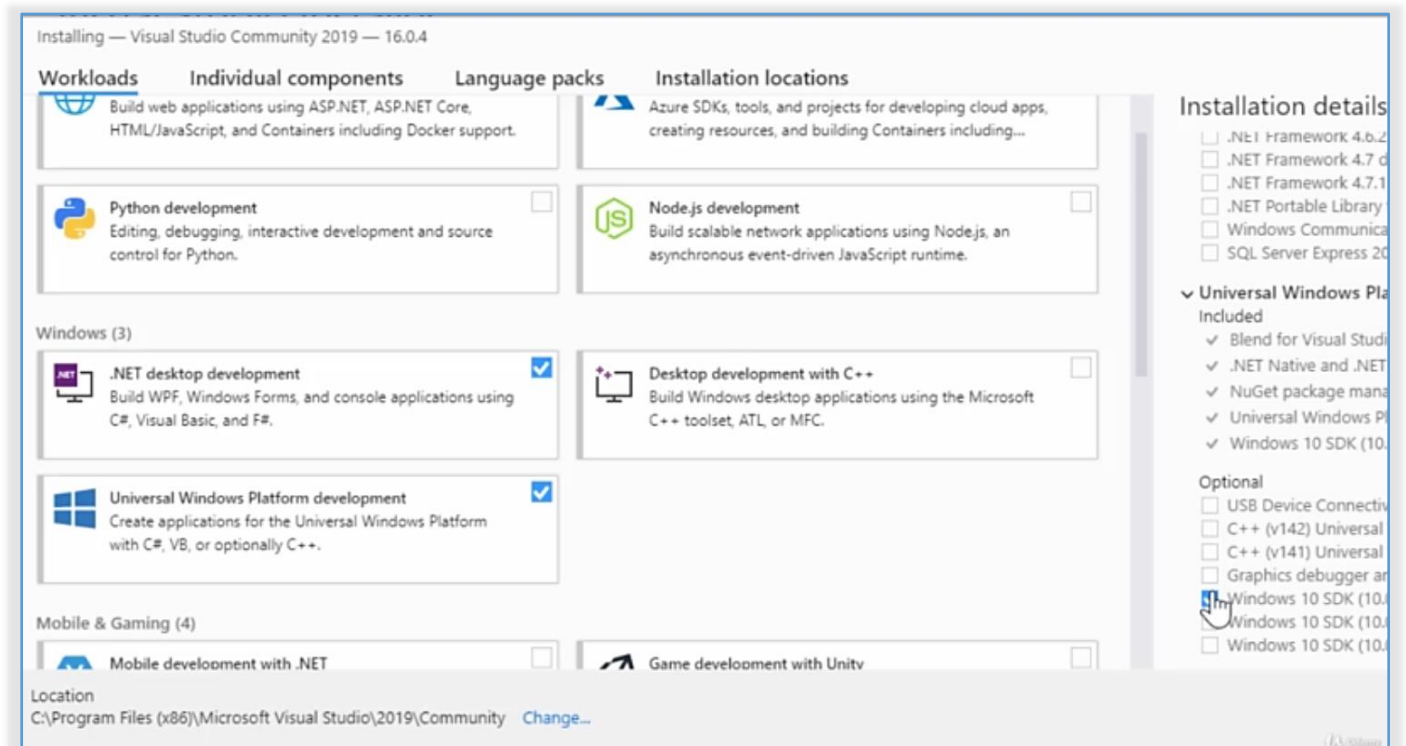


Figure 4

Click the **Install** button finally and let the download and installation complete.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

## Section 5

# Installing Node.js and Appium

Now we will install Node.js and after that we will install Appium.

Node.js can be downloaded from the following URL:

<https://nodejs.org/en/download/>

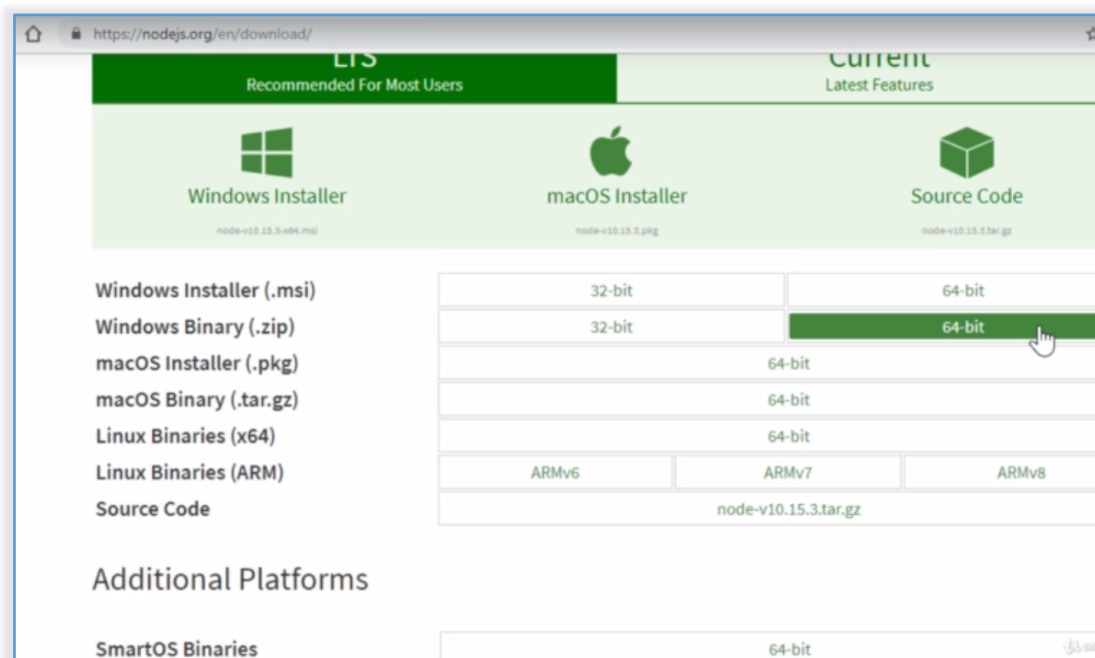


Figure 5

Minimum required version of Node is version 6 or higher for running WinAppDriver and Appium. Download and install the version suitable for your machine and install it.

The second last step will be installing Appium on your machine through Node Package Manager also known as **NPM**.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

In order to install Appium, open the command prompt on your PC. Press **Windows + R** and type **cmd** and press Enter key to launch command prompt window.

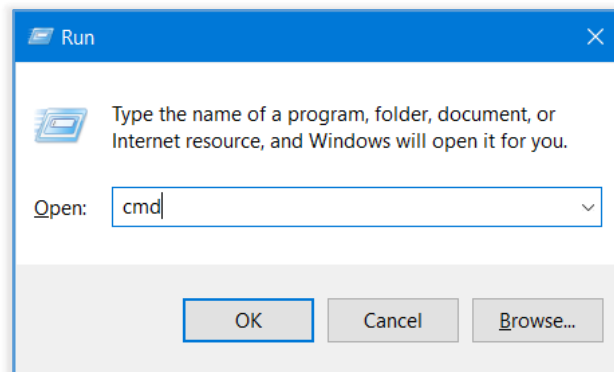


Figure 6

In the command prompt window, type the following command and hit enter key:

**NPM install -g appium**

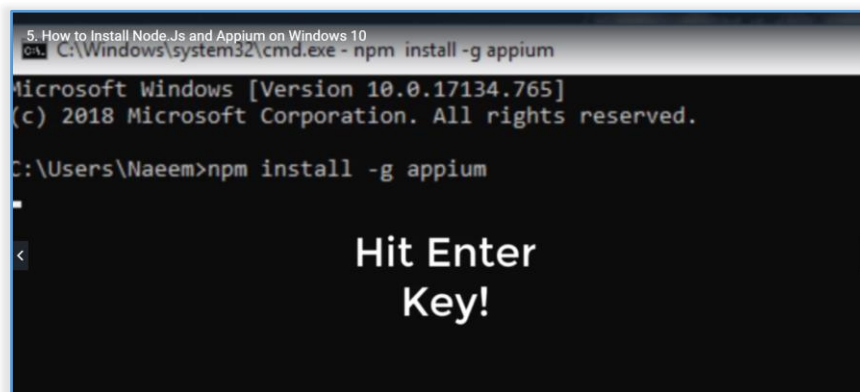


Figure 7

Installing Appium will take a while; let it happen.

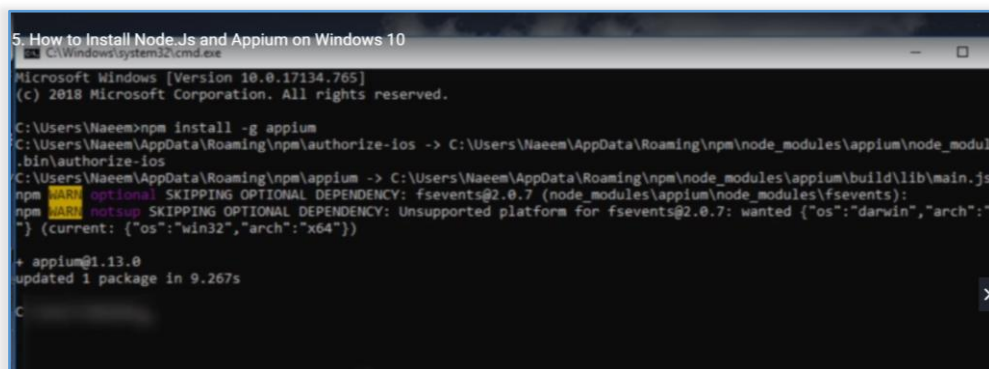


Figure 8

In the next section, we'll download the WinAppDriver.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>



## Section 6

# Installing WinAppDriver

The last but most important software which we need to install is WinAppDriver.

Let's Go to the WinAppDriver github page, the URL is given below <https://github.com/Microsoft/WinAppDriver>

Click on “**Releases**” link. At the time of this book being compiled, version v1.2.1 is the latest stable release and we are going to download & install it.

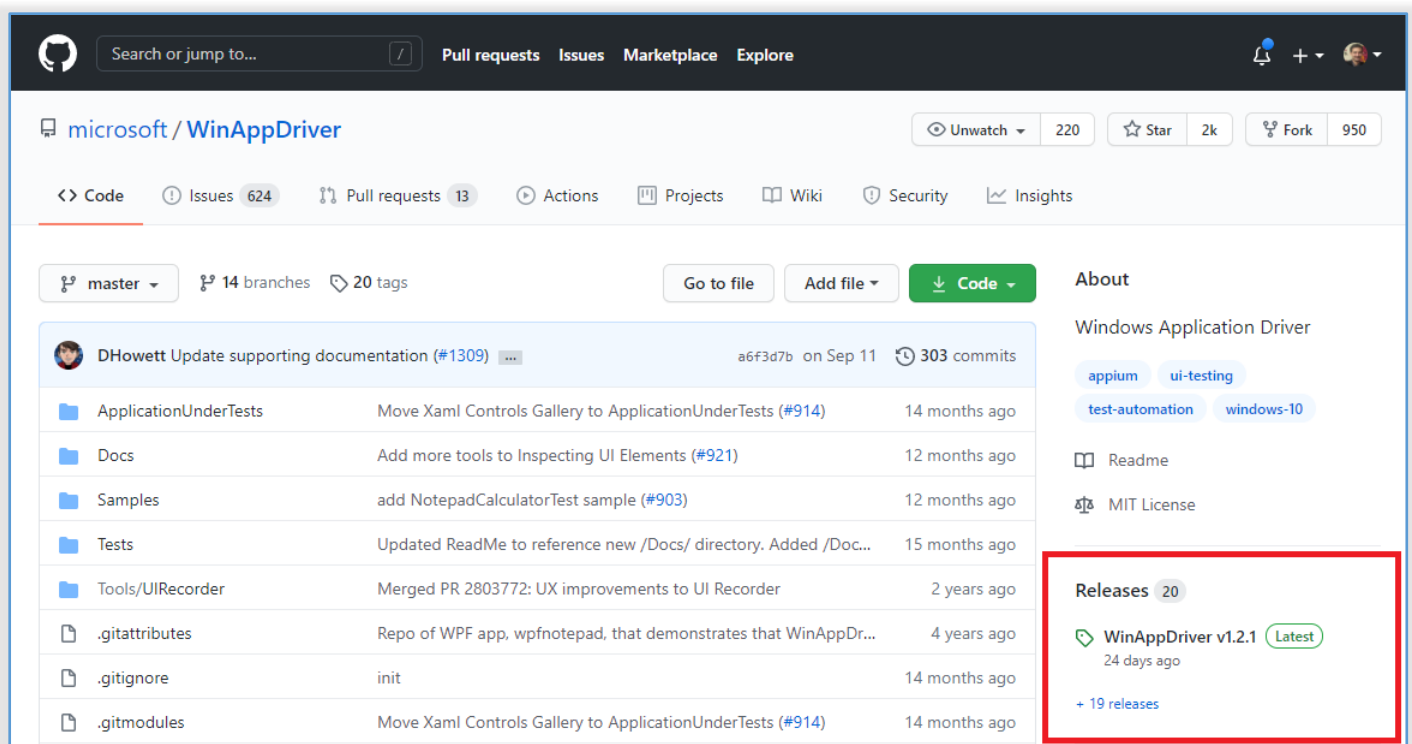


Figure 9

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

It has a green box on left side which says **Latest release**.

Click on the “**Assets**” arrow to expand it.

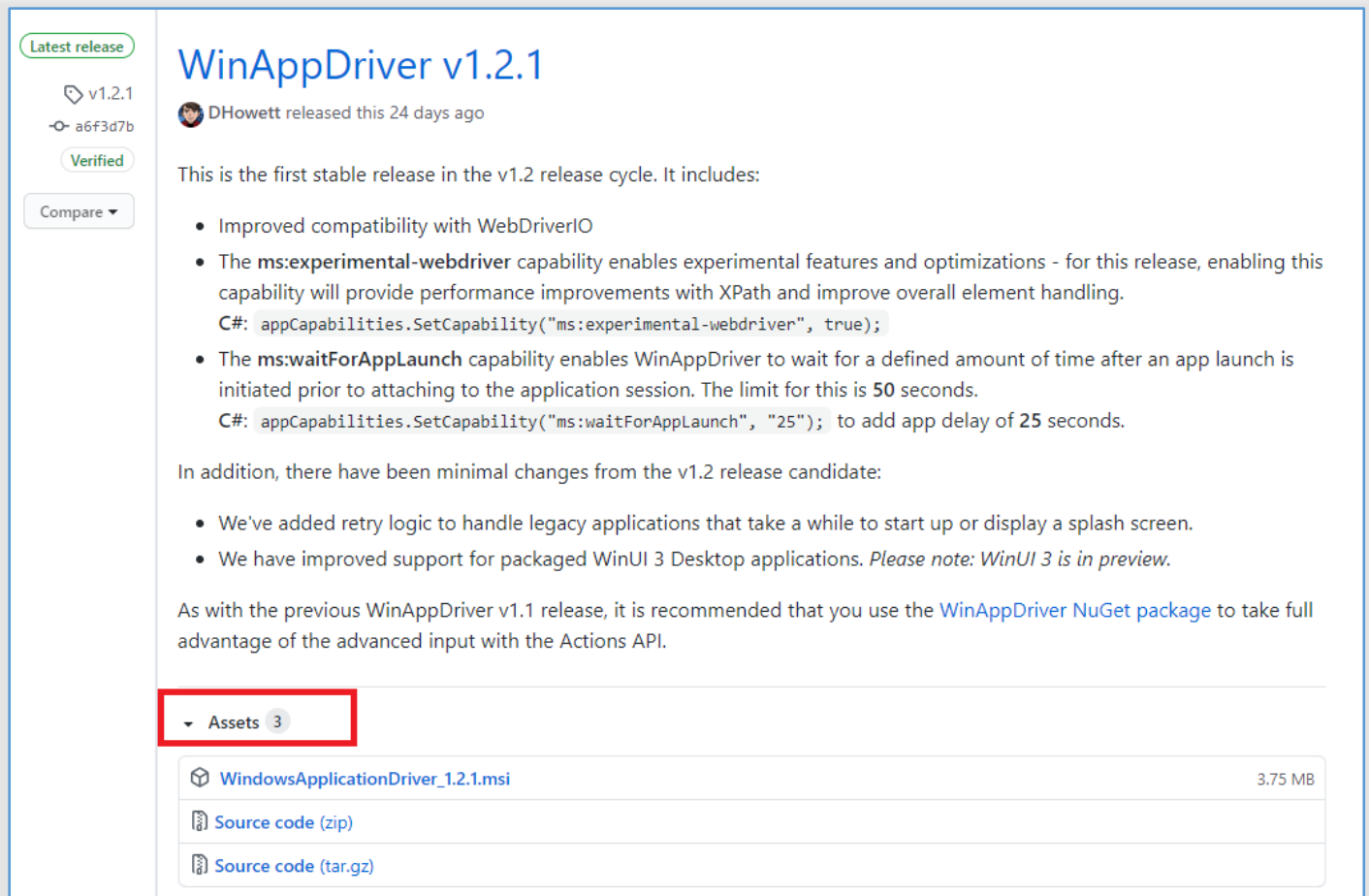


Figure 10

Click **WindowsApplicationDriver.msi** to download it.

Once it is downloaded, install it.

That's it; we're done with the installations.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

I highly recommend you to download the full source code of WinAppDriver repository and look into the samples later on.

If you're totally new to test automation, you may check the repository source code after you get comfortable with the basics of test automation.

Once you get comfortable with the basics, the samples will be very helpful in developing further understanding. The Issues section of this repository is also very active and visiting it can be helpful from a learning perspective.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

## Section 7

# How & Why to Run WinAppDriver

In the previous section, we installed WinAppDriver on our PC. Let's go to the installation folder and see what is in there?

You'll also learn what is the importance of WinAppDriver.exe in UI automation based on WinAppDriver.

I'll open file explorer and type path of WinAppDriver installation folder.

On a 64-bit machine, WinAppDriver is installed at the path "**C:\Program Files (x86)\Windows Application Driver**".

On a 32-bit machine, WinAppDriver is installed at the path "**C:\Program Files\Windows Application Driver**".

Inside the folder, you will have the **WinAppDriver.exe** file.

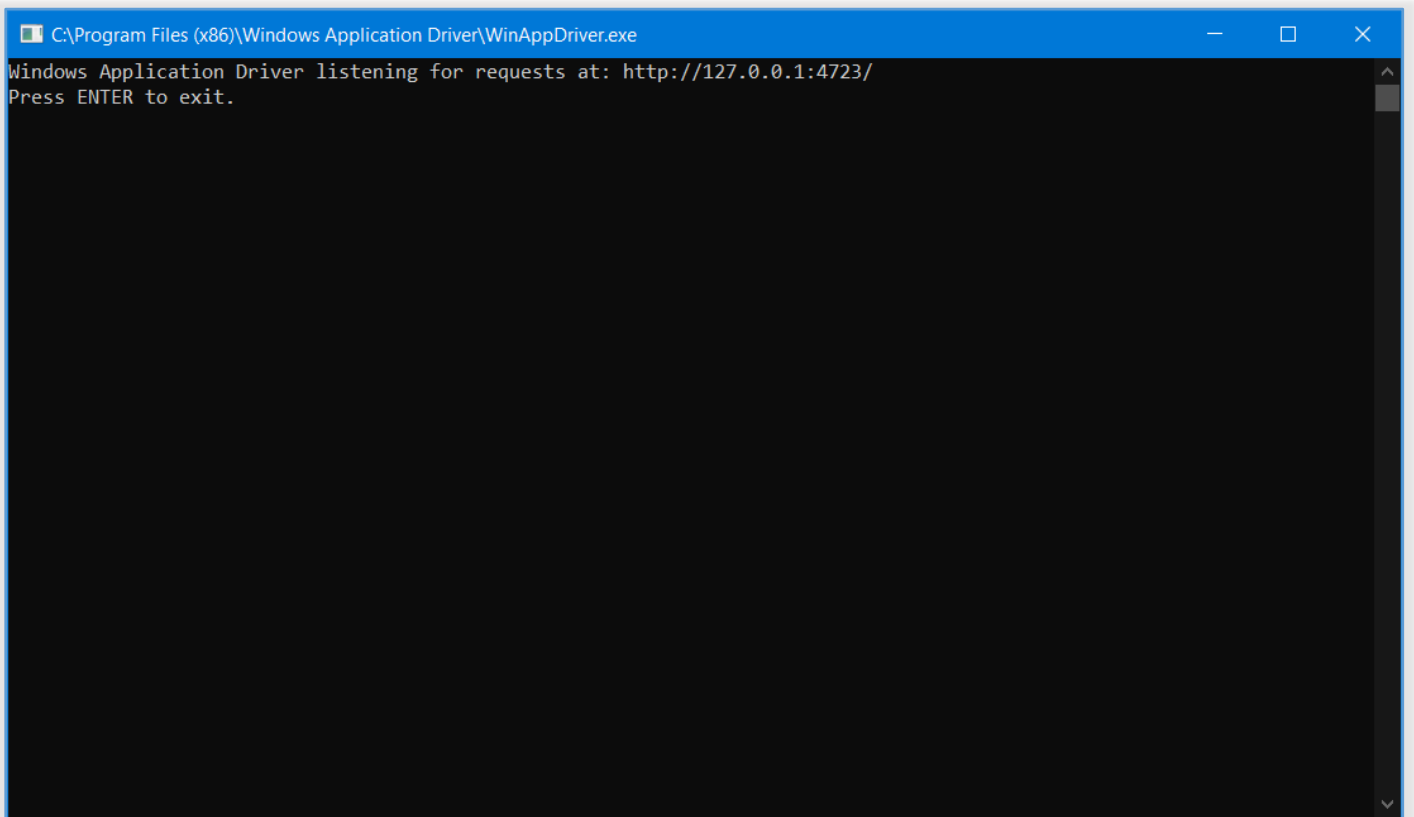
### Notes & Tips:

Your WinAppDriver based UI automation will fail if the file WinAppDriver.exe is not up & running.

The WinAppDriver.exe is the heart of UI automation on the WinAppDriver platform.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

Let's start it. It's a console application and does not have a UI.



*Figure 11*

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

The WinAppDriver.exe is listening for incoming automation commands using **http** protocol. It is listening for commands on **port number 4723**.

Your script tells the WinAppdriver what automation operation has to be performed. Actual automation on the ground is performed by WinAppDriver.exe.

The script and the driver can be either on the same machine or on different machines.

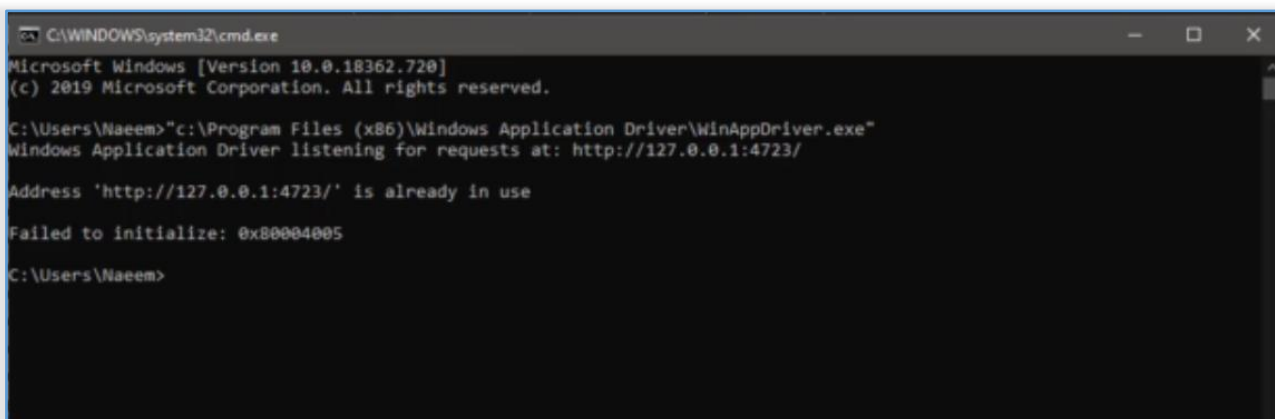
#### Notes & Tips:

Your script will fail if WinAppDriver is not reachable through the port number, you specified in your script.

Windows Firewall can cause serious pain in this regard sometimes.

If let's suppose, the port number is already in use by another program, you can specify different one through the command line.

This is what happens if the port is already in use.

A screenshot of a Windows Command Prompt window. The title bar reads 'C:\WINDOWS\system32\cmd.exe'. The window content shows the following text:

```
Microsoft Windows [Version 10.0.18362.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Waeem>"c:\Program Files (x86)\Windows Application Driver\WinAppDriver.exe"
Windows Application Driver listening for requests at: http://127.0.0.1:4723/

Address 'http://127.0.0.1:4723/' is already in use

Failed to initialize: 0x80004005

C:\Users\Waeem>
```

Figure 12

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

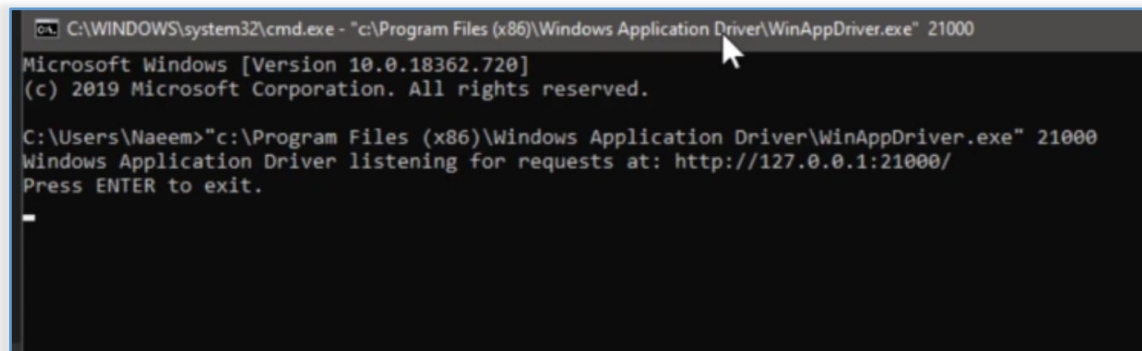
To understand it further, open the command prompt.

Type the full path of WinAppDriver and press enter.

Here, I can simply type

**WinAppDriver.exe [the port number]**

Let's suppose I'll type 21000 and hit **Enter** key.



```
C:\WINDOWS\system32\cmd.exe - "c:\Program Files (x86)\Windows Application Driver\WinAppDriver.exe" 21000
Microsoft Windows [Version 10.0.18362.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Naeem>"c:\Program Files (x86)\Windows Application Driver\WinAppDriver.exe" 21000
Windows Application Driver listening for requests at: http://127.0.0.1:21000/
Press ENTER to exit.
```

Figure 13

The server will start running on this port now.

#### Notes & Tips:

Please note that the port number must be greater than 1024 and less than 65535. First 1024 are reserved for the system.

You may create a shortcut to the WinAppDriver.exe on your desktop as well to simplify opening it every time.

It's better to let the WinAppDriver run on the default port, 4723.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

## Section 8

# Launch Notepad with Appium and WinAppDriver in C#

This section shows how to use Appium and WinAppDriver to launch the built in Windows 10 Notepad.exe application.

In the coming section, you will learn how to create an object of **AppiumOptions** and specify the app key. You'll also learn how to create an instance of a **WindowsDriver<WindowsElement>** to do multiple operations with the application UI.

For example, retrieve the application title with automation, take a screenshot for later uses, close the application with automation etc.



## Section 9

# Project Setup [Visual Studio] and Launching an Application with WinAppDriver, Appium Client

In this section, we are going to create a new solution in Visual Studio. It will be our “hello world” of test automation. We will add a reference to the **Appium API** client libraries. I am going to show you how to launch the legacy application Notepad using **WinAppDriver**.

So, let's get to work and open **Visual Studio**.

Click "**Create a new Project**"

I will Select “**Console App (.Net Framework)**” from the new project dialog. You may use .Net Core, it is equally good.

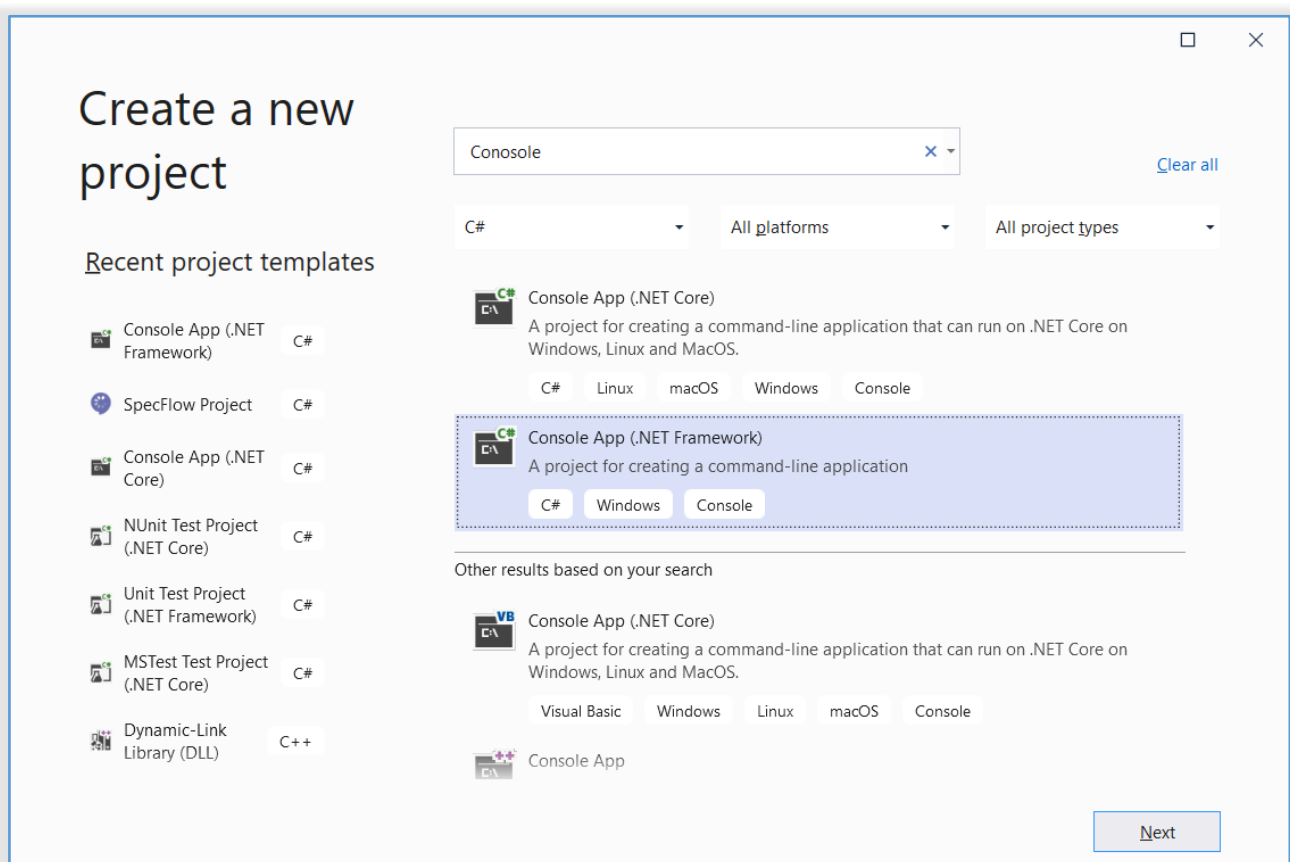


Figure 14

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

I'll click the **Next** button.

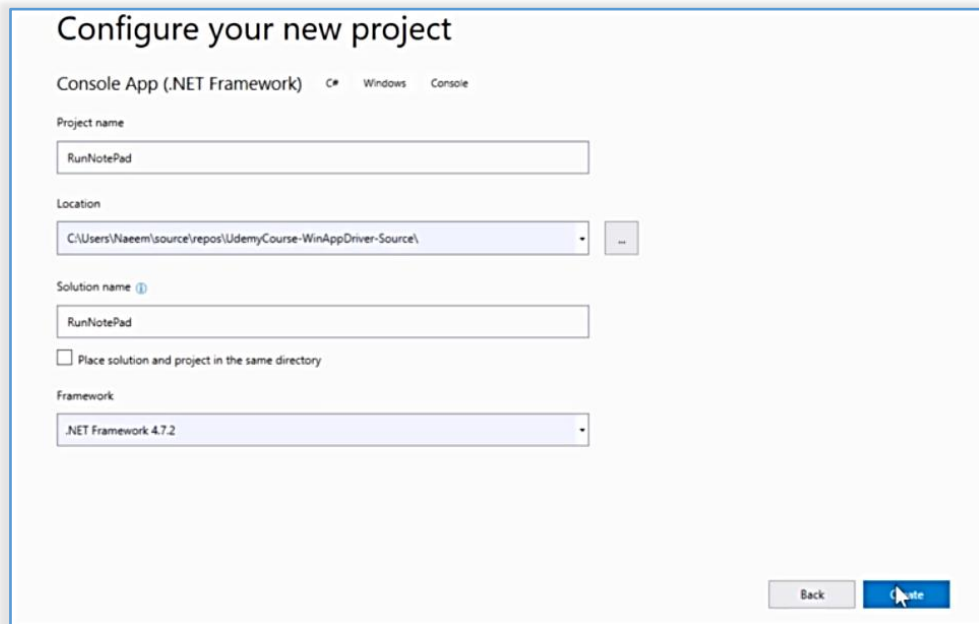


Figure 15

I'll change the name to "**RunNotepad**", leave everything else as it is and click the "**Create**" button.

Once the new project is created, expand the **Solution Explorer**.

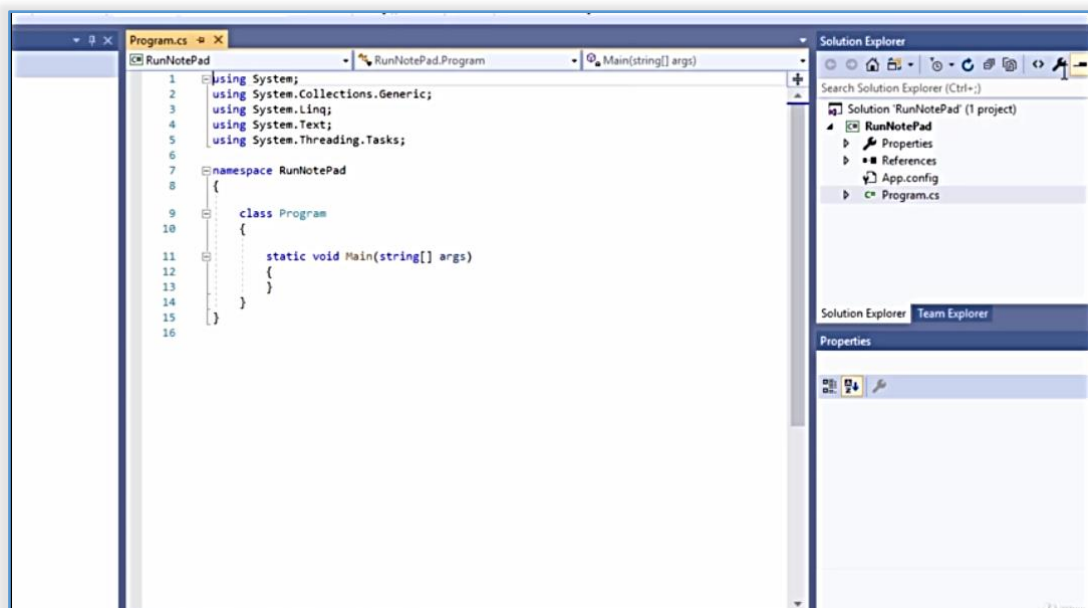


Figure 16

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

Right click the **References** and select “**Manage Nuget Packages**”.

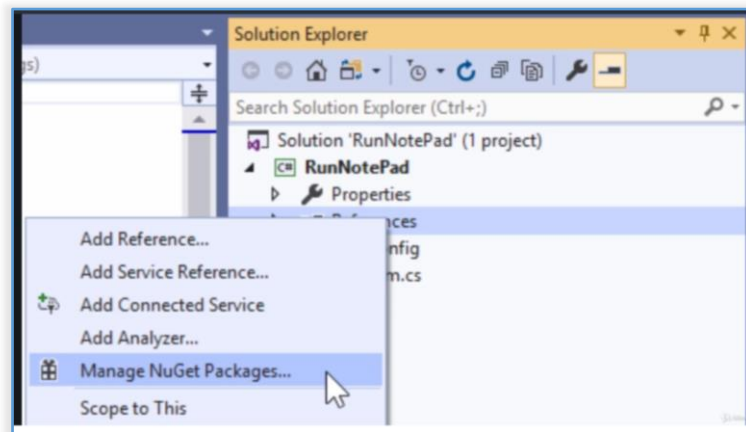


Figure 17

Click the “**Browse**” tab and search for **Appium.WebDriver**

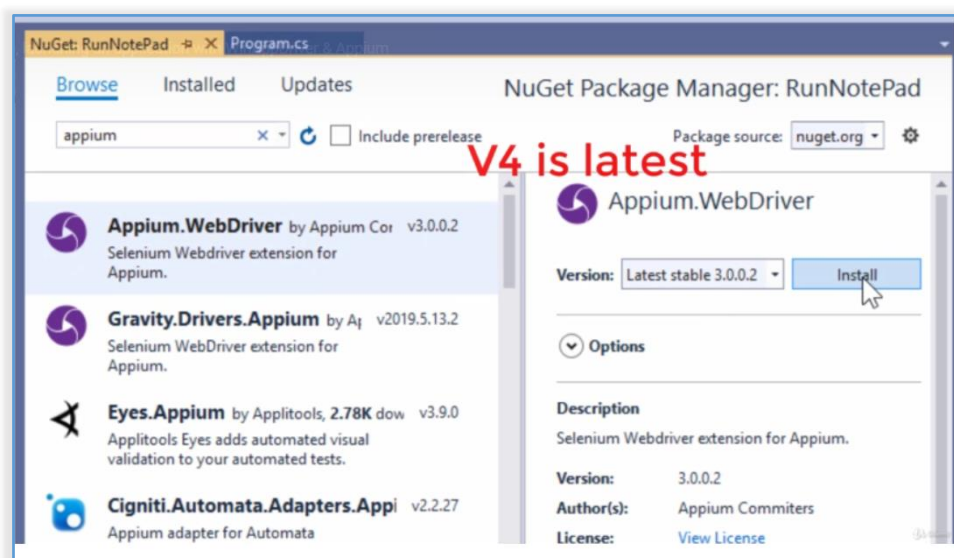


Figure 18

Select the package in the list.  
We're going to install it by clicking “**Install**” button.

Click **OK** on preview changes dialog.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

When the installation is done, click the file, **Program.cs** in **Solution Explorer**.

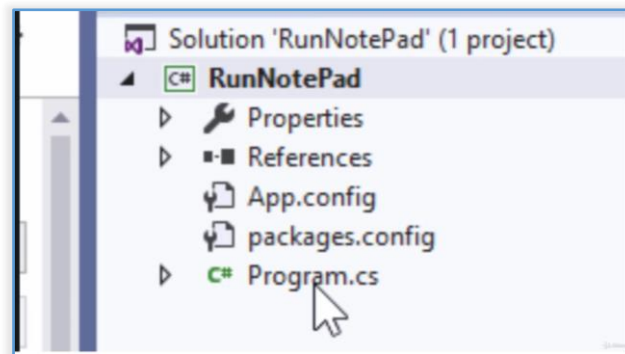


Figure 19

We'll go to the main method in this file.

We are going to create an object of **WindowsDriver** with generic type **WindowsElement** as shown below.

```
WindowsDriver<WindowsElement> notePadSession;
```

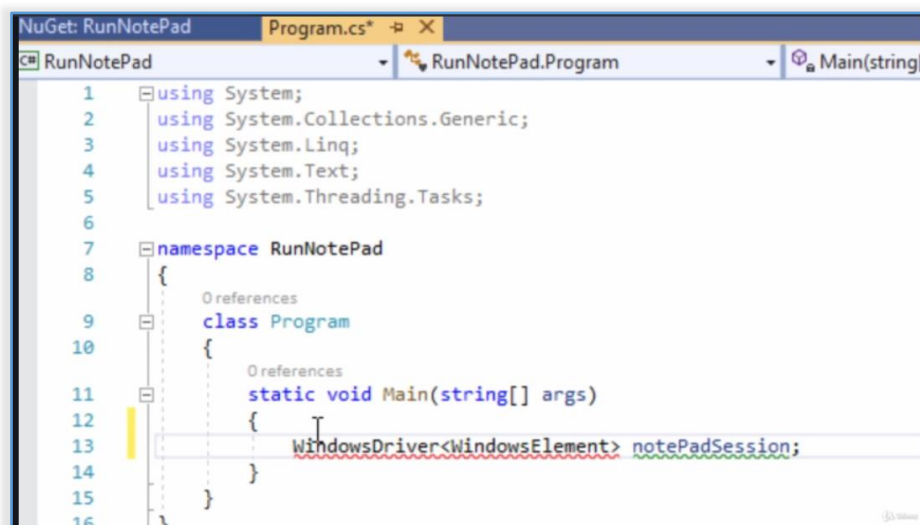


Figure 20

### Notes & Tips:

Appium client API is used to communicate with Appium server in a proper way.

The variable is named `notePadSession`. I'm not going to assign it a value yet.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

Since we haven't included the **namespaces**, let me click the word **WindowsDriver** behind **notepadSession** object.

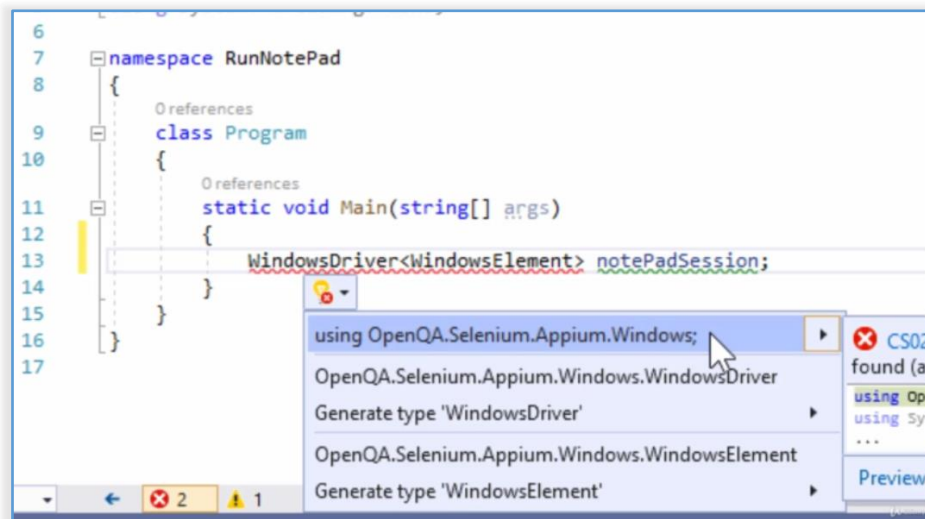


Figure 21

When a bulb will show up, click it and select import **namespace**. In order to run a legacy application, we must tell the **WindowsDriver** the full path of the **.exe** file.

That is done using an object of **AppiumOptions** class.

**AppiumOptions** class is used for specifying the options we desire to be present in the application which we want to test with **WinAppDriver**.

Let's define and instantiate that object.

```
AppiumOptions desiredCapabilities = new AppiumOptions();
```

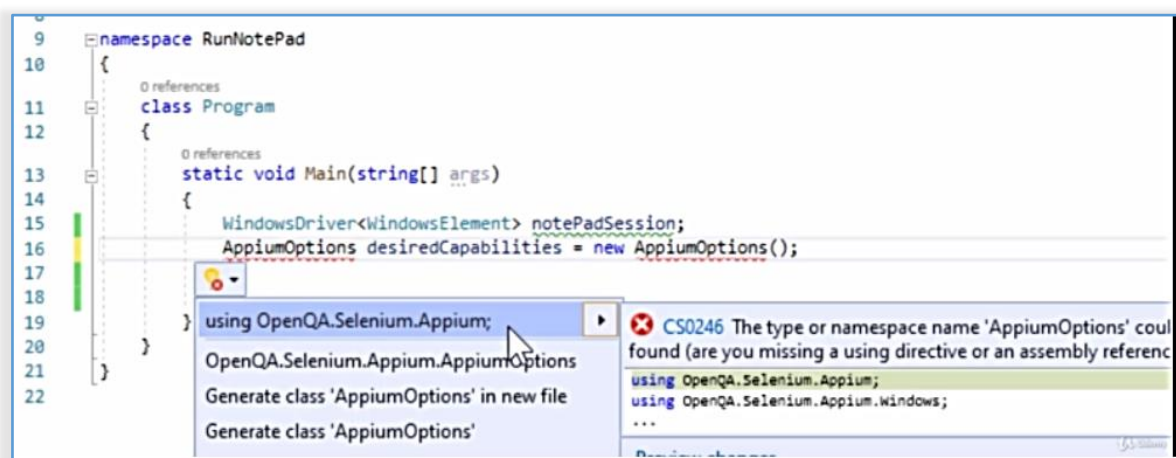


Figure 22

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

Bring in the namespace, and click the save all button. Now build the solution. It should build without any error.

## Section 10

# Launching a Legacy Application Notepad using AppiumOptions

We have already created an object of **WindowsDriver** and an object of **AppiumOptions**. These belong to the **Appium** library.

In this section, we're going to launch the Windows Notepad application using these objects.

Since notepad is a legacy application, we will need to tell Appium the path of **Notepad.exe** file.

The path of the application is specified using the object of the class **AppiumOptions** which we created earlier.

We will call the method **AddAdditionalCapability** on this object to specify various applications capabilities such as the path of the application under test.

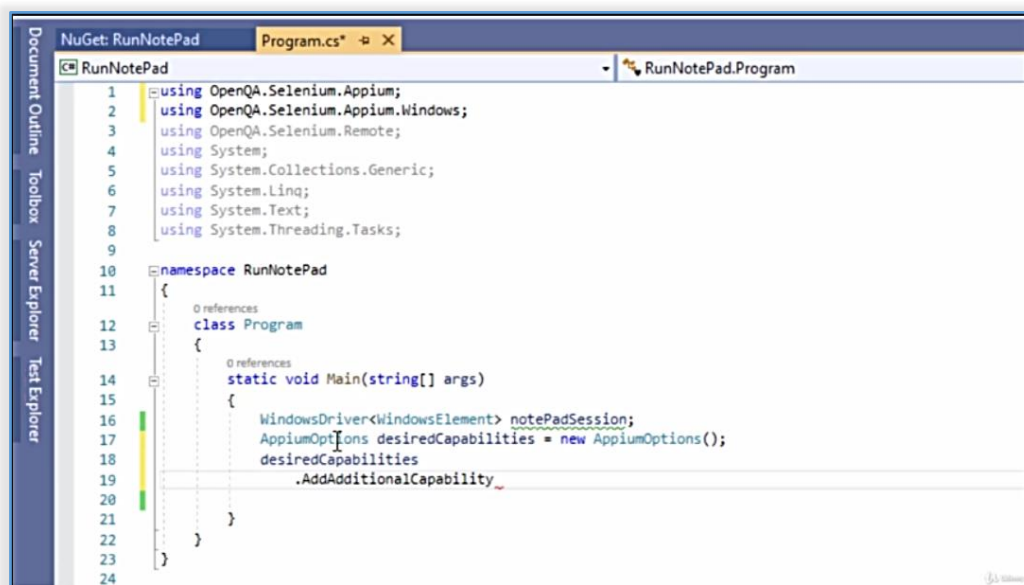


Figure 23

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>





This method takes input in name, value pair format.

We'll pass the string value **"app"** to the first parameter.

If you're wondering where is this magic string coming from, stop worrying and learn to live with magic strings. The key "app" is also defined in the MobileCapability namespace (OpenQA.Selenium.Appium.Enums) and it can be accessed as following:

### Notes & Tips:

All possible valid values of this first parameter are defined on the WinAppDriver GitHub page [here](#).

MobileCapabilityType.App

Capabilities	Descriptions	Example
app	Application identifier or executable full path	Microsoft.MicrosoftEdge_8wekyb3d8bbwe!MicrosoftEdge
appArguments	Application launch arguments	<a href="https://github.com/Microsoft/WinAppDriver">https://github.com/Microsoft/WinAppDriver</a>
appTopLevelWindow	Existing application top level window to attach to	0xB822E2
appWorkingDir	Application working directory (Classic apps only)	C:\Temp
platformName	Target platform name	Windows
platformVersion	Target platform version	1.0

Figure 24

The second parameter value is the path of the notepad application.

```
desiredCapabilities.AddAdditionalCapability(
    "app", @"C:\Windows\System32\notepad.exe");
```



Figure 25

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

I'll close the method call bracket and place a semicolon.

Now let's assign a value to **notePadSession**.

```
notePadSession =
    new WindowsDriver<WindowsElement>(
        new Uri("http://127.0.0.1:4723"),
        desiredCapabilities);
```

The first parameter will be a new object of Uri. I will pass it the parameter **"127.0.0.1:4723"**. This will work when the **WinAppDriver.exe** is running on the same machine as your code. It is by default listening on port number **4723**.

The second parameter will be the **AppiumOptions** object which we created earlier.

By doing this, we're informing the driver object about the application path we want it to launch.

If the application is launched successfully, the return value will be not null.

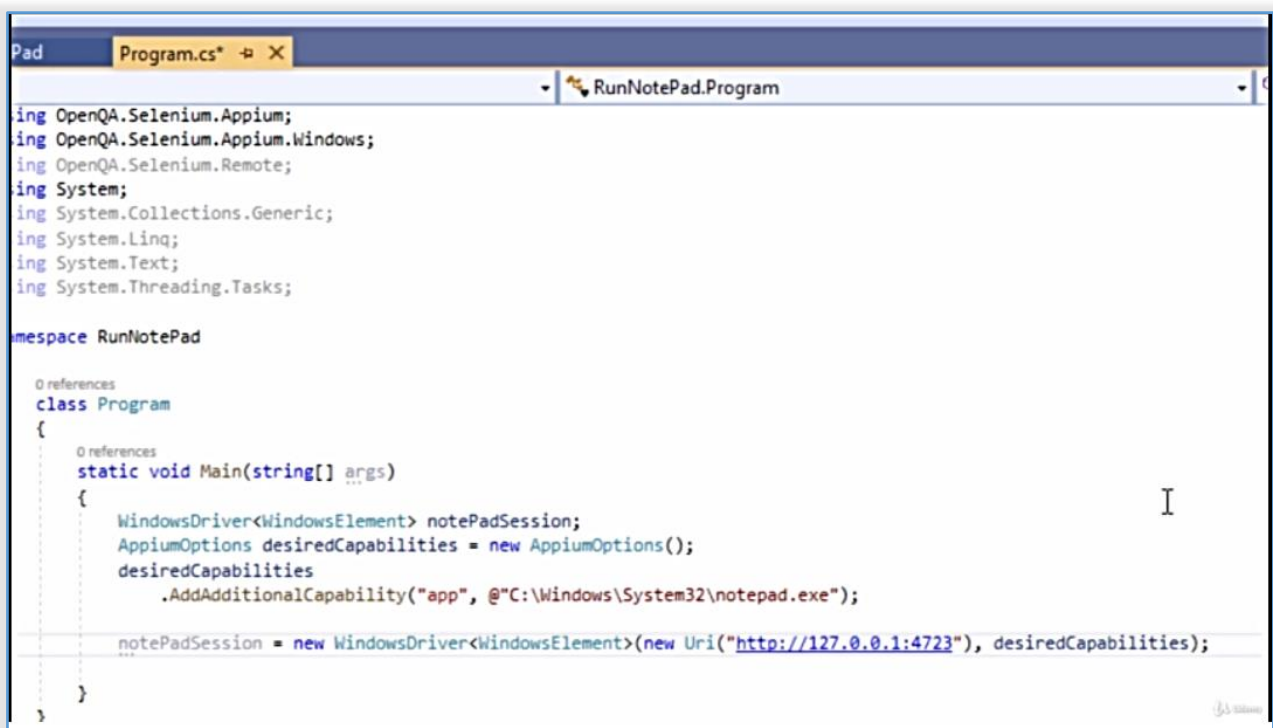


Figure 26

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

Let's test it in an if condition.

```
if (notePadSession == null)
{
    Console.WriteLine("App not started.");
    return;
}
```

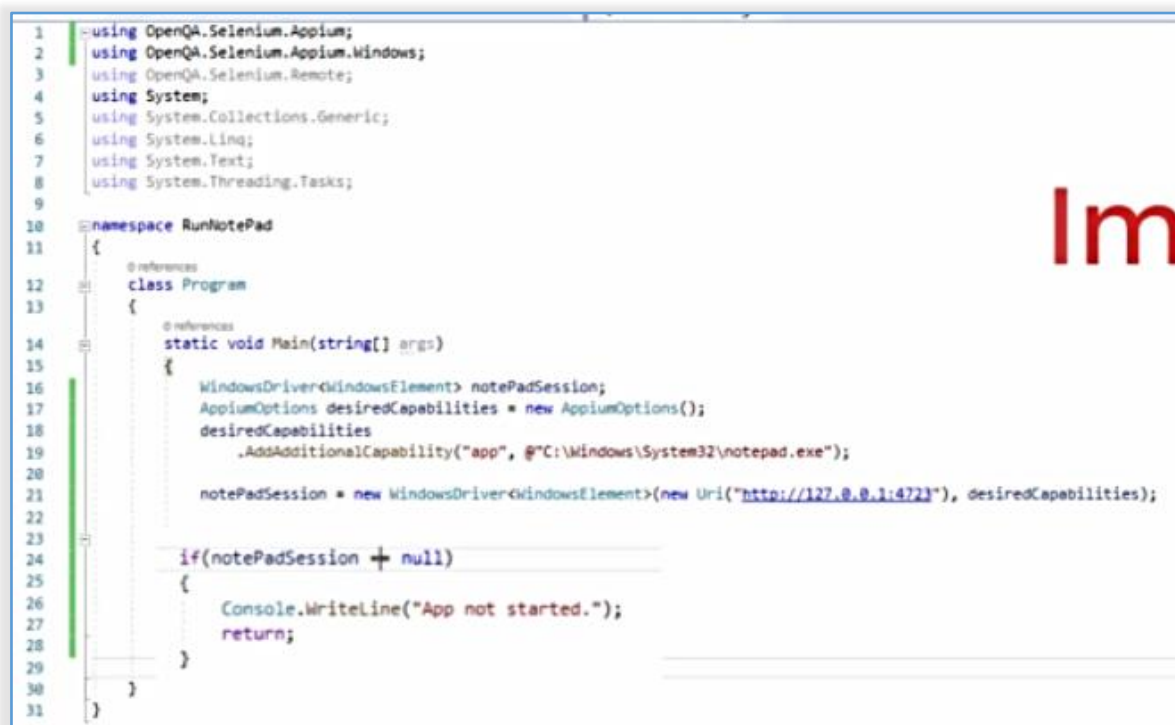


Figure 27

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

## Section 11

# Using WinAppDriver for Accessing Application Window

## 1. How to Get Application Title

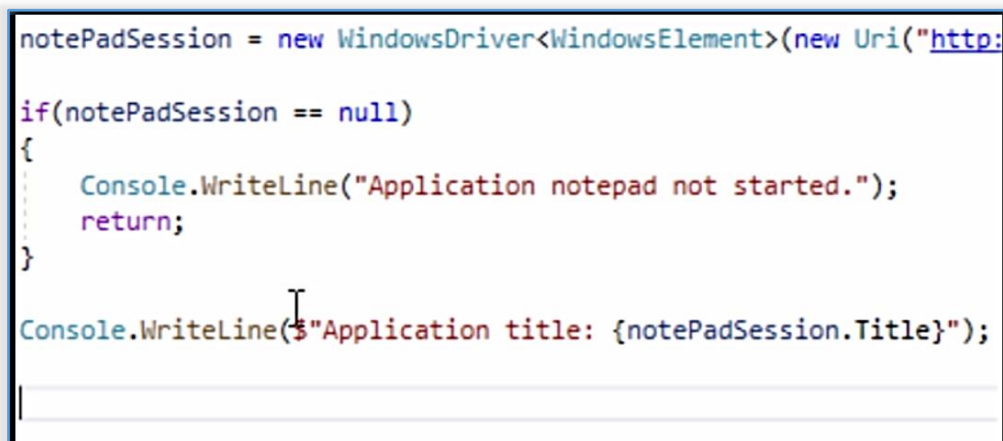
The **WindowsDriver** object shown here is a very important object. It can be used to retrieve various properties of the application under test. For example, the **title of the application**.

Sometimes we can use the title to make sure that the application has loaded successfully before proceeding.

The title is exposed through a property of the `WindowsDriver<WindowsElement>` object in use, the name of the property is `Title`.

Let's print the title of Notepad to the console.

```
Console.WriteLine($"Application title: {notePadSession.Title}");
```



```
notePadSession = new WindowsDriver<WindowsElement>(new Uri("http:
if(notePadSession == null)
{
    Console.WriteLine("Application notepad not started.");
    return;
}
Console.WriteLine($"Application title: {notePadSession.Title}");
```

Figure 28

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

## 2. How to Maximize an Application Window

We can also maximize the main window of an application under test. It is done through the following line of code:

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

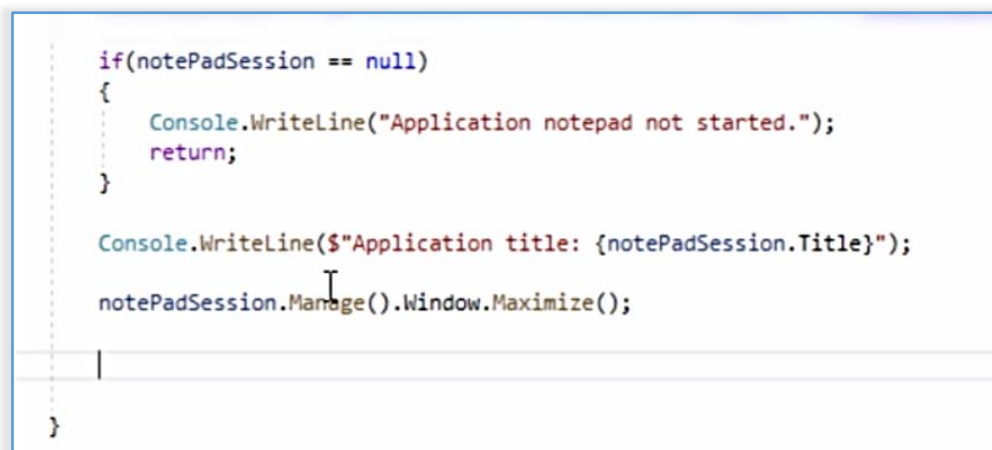


Figure 29

## 3. How to Take a Screenshot of the AUT

You can also retrieve a screenshot of the application under test(AUT) through the relevant WindowsDriver object. The name of the method used for this purpose is GetScreenShot. For example:

```
var screenShot = notePadSession.GetScreenshot();
```

This screenshot can be saved to the disc, by calling the method SaveAsFile on the object returned. Please see the line of code below.

```

screenShot.SaveAsFile(
    $"\\Screenshot{DateTime.Now.ToString("ddMMyyyyhhmmss")}.png",
    OpenQA.Selenium.ScreenshotImageFormat.Png);

```

The first parameter takes the file path to where we want the screenshot to be saved.

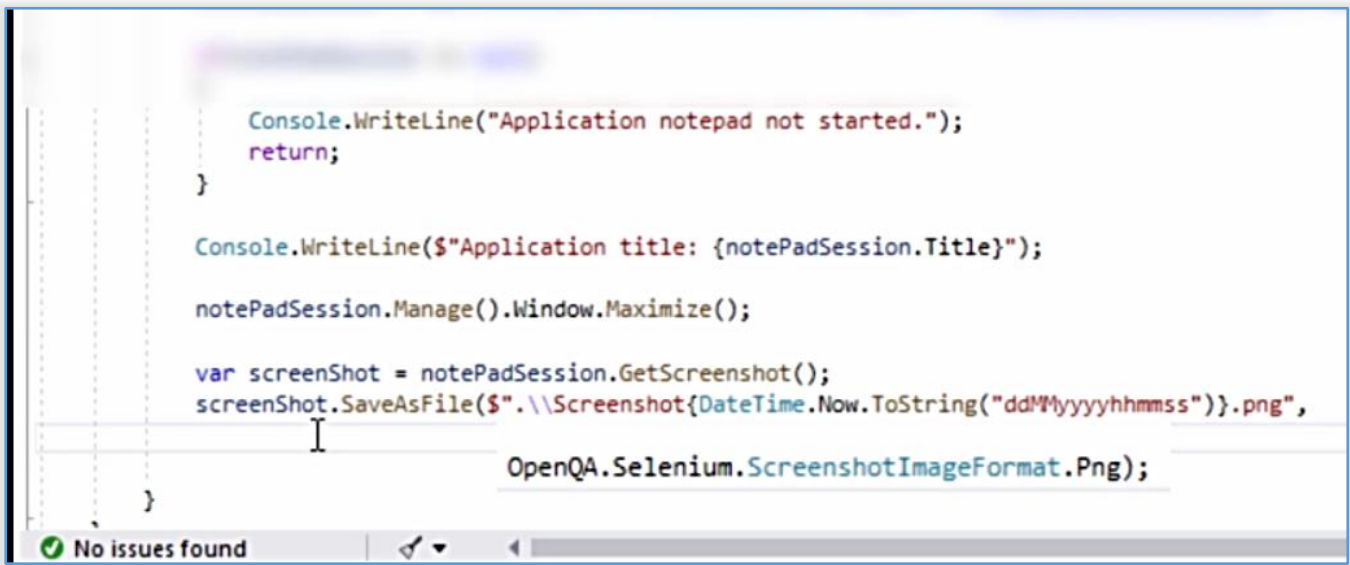
I've used C# 6 style string interpolation to generate a unique image name. Otherwise, it could cause problems.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

Don't include any characters in the path which are not allowed, for example the backslash character. That would result in a script failure.

The screenshot will be saved in the present working directory of the .exe file of your hello world program.

The second parameter is the image format. I've specified .png here.



## **4. How To Close the Application**

If you want to quit the application under test, just call the quit method on the relevant `WindowsDriver<WindowsElement>` object as shown below.

```
notePadSession.Quit();
```

## Section 12

# Demonstration

We will now run the application.

But before that, we must make sure that the WinAppDriver must be running.

Start the WinAppDriver.exe before your proceed.

From the Visual Studio, start the application by pressing **Ctrl + F5**.

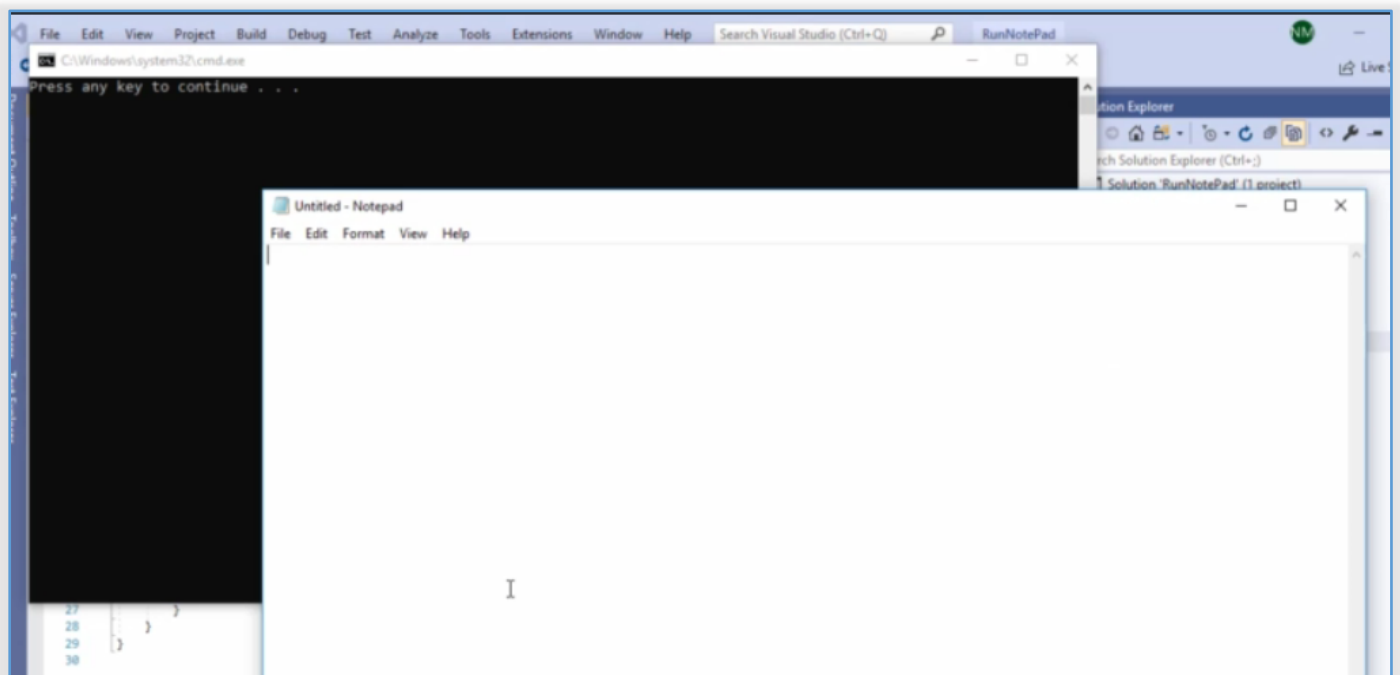
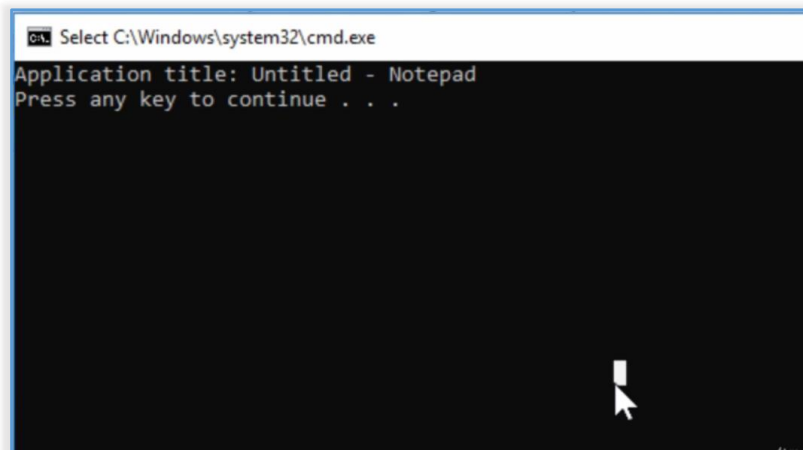


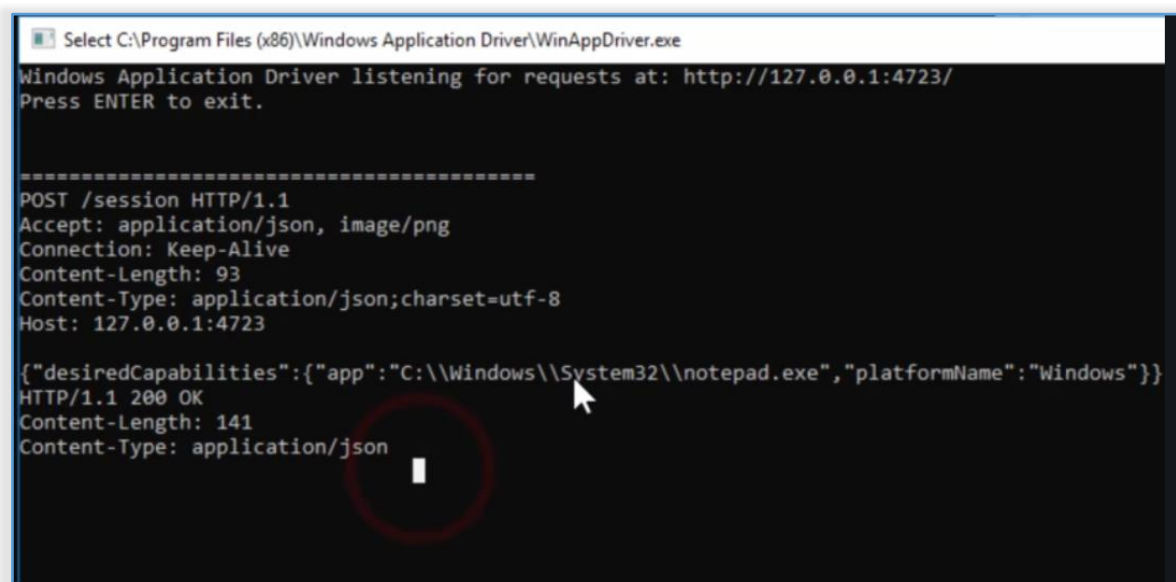
Figure 31

First of all, you'll see that the application title is printed on the console.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

*Figure 32*

The WinAppDriver window will also print console log about the operations performed by your script.

*Figure 33*

Secondly, if we open the application directory, we will be able to see a screenshot created by our script.

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>



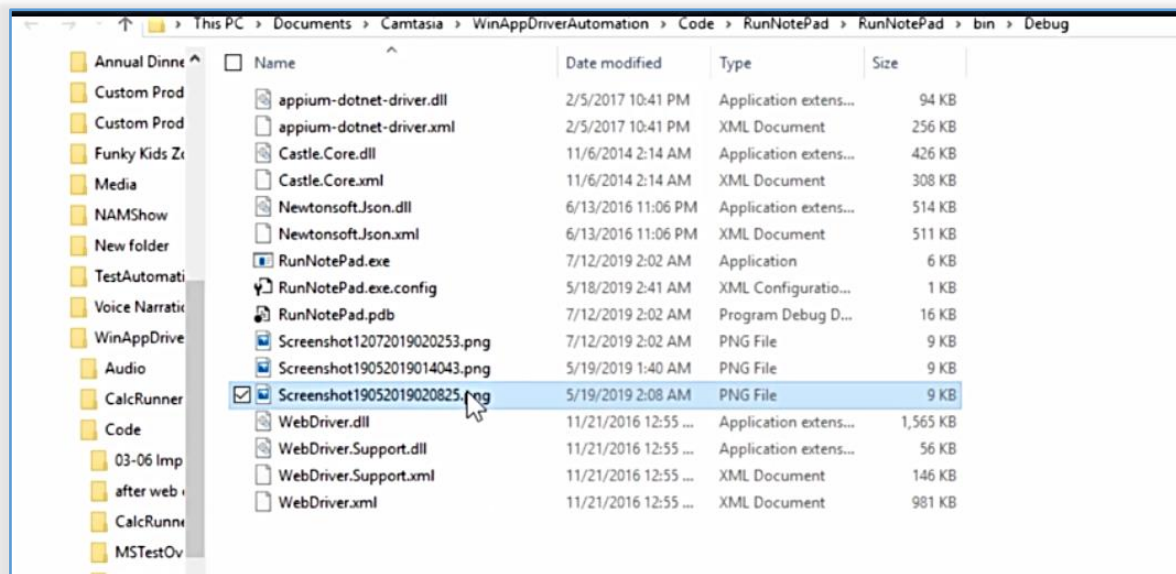


Figure 34

**Complete source code:**

```

static void Main(string[] args)
{
    WindowsDriver<WindowsElement> notePadSession;
    AppiumOptions desiredCapabilities = new AppiumOptions();
    desiredCapabilities.AddAdditionalCapability(
        "app", @"C:\Windows\System32\notepad.exe");

    notePadSession =
        new WindowsDriver<WindowsElement>(
            new Uri("http://127.0.0.1:4723"), desiredCapabilities);

    if (notePadSession == null)
    {
        Console.WriteLine("App not started.");
        return;
    }

    Console.WriteLine($"Application title: {notePadSession.Title}");

    notePadSession.Manage().Window.Maximize();

    var screenShot = notePadSession.GetScreenshot();
    screenShot.SaveAsFile(
        $"{DateTime.Now.ToString("ddMMyyyyhhmmss")}.png",
        OpenQA.Selenium.ScreenshotImageFormat.Png);

    notePadSession.Quit();
}

```



## Section 13

# How to Use AppiumService to Run Test Automation

Until so far, we've been starting WinAppDriver.exe explicitly before running our test automation scripts. It is possible to get rid of this step. You must have Node.js and Appium installed on the PC running your application under test.

The class used for this purpose is **AppiumServiceBuilder**. We can create an instance of this class and call various methods on it to specify the properties of our instance of WinAppDriver.

After specifying the properties of the service we'll call the Build method.

For example, you can modify the script we already wrote for launching Notepad application.

```
var appiumLocalService = new AppiumServiceBuilder().UsingPort(4723).Build();
```

All methods on the right-hand side of the above statement return an instance of **AppiumServiceBuilder** class. Making the method call chaining possible.

We will create and setup an object of **AppiumOptions** class just like we did it before. It is shown below for reference.

```
AppiumOptions desiredCapabilities = new AppiumOptions();
desiredCapabilities.AddAdditionalCapability(
    "app", @"C:\Windows\System32\notepad.exe");
```

In order start your automation scripts, you'll utilize the instance of **AppiumServiceBuilder** setup in the above lines.

The main difference here is that in the first parameter in the constructor of **WindowsDriver<WindowElement>** we will provide an instance of

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

**AppiumLocalService**, instead of an object of Uri pointing to the WinAppDriver server.

```
WindowsDriver<WindowsElement> notePadSession =
    new WindowsDriver<WindowsElement>(
        appiumLocalService, desiredCapabilities);
```

As a result of executing this code, WinAppDriver will launch the Notepad application on your PC. You will be able to perform all the operations

### Writing WinAppDriver log to a file

If you want to write the log of WinAppDriver to a file, you can call the method call **WithLogFile** before calling the **Build** method.

This method takes as parameter an object of **System.IO.FileInfo** class.

It is your responsibility to make sure that the path you specify is valid.

The code will look as following after the change:

```
var appiumLocalService = new AppiumServiceBuilder().
    UsingPort(4723).
    WithLogFile(
        new System.IO.FileInfo(
            @"C:\Users\Naeem Malik\Documents\Logs\TestLog.txt")).
    Build();
```

## Closing Remarks

This is the end of this book.

This book is based on my Udemy course “Appium WinAppDriver UI Automation Testing Windows Apps in C#”.

Since you're in a flow, you can continue your test automation journey immediately by visiting the link below. Complete working source code is also available in the video course. I am also available to answer student questions as well.

The link of my course is given below:

<https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

I also teach how to setup Azure DevOps build pipelines for running WinAppDriver based test automation. You may check it out by visiting the link below.

<https://www.udemy.com/course/windows-ui-automation-on-azure-devops-build-pipelines/?referralCode=31F4FCC272434D3B1C3C>

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

# Cheat sheet

What is the name of the Nuget package which we add to use Appium in our solution?

Appium.WebDriver

What is the latest major version of Appium?

Version 4 is the latest version

Which class is used in principal for creating Windows UI automation?

WindowsDriver<WindowsElement>

What is the namespace of Windows UI automation?

OpenQA.Selenium.Appium.Windows

Which class is used to specify the path of application under test and other options?

AppiumOptions

How to create an instance of AppiumOptions?

```
AppiumOptions desiredCapabilities = new AppiumOptions();
```

What is the namespace of AppiumOptions?

OpenQA.Selenium.Appium

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>

Which method is used to specify an additional capability?

AddAdditionalCapability

How do we specify the path of an application under test?

```
AppiumOptions desiredCapabilities = new AppiumOptions();
desiredCapabilities.AddAdditionalCapability("app", @"C:\Windows\System32\notepad.exe");
```

What's the class which defines most keys used by AddAdditionalCapability?

MobileCapability

E.g.

MobileCapabilityType.App

MobileCapabilityType.PlatformName

MobileCapabilityType.DeviceName

What is the namespace of MobileCapability?

OpenQA.Selenium.Appium.Enums

Where can I find a list of all possible capability keys supported by WinAppDriver?

<https://github.com/microsoft/WinAppDriver/blob/master/Docs/AuthoringTestScripts.md>

How do we create an object of WindowsDriver<WindowsElement>?

```
WindowsDriver<WindowsElement> app = new WindowsDriver<WindowsElement>(new
Uri("http://127.0.0.1:4723"), desiredCapabilities);
```

How to retrieve the title of an application:

notePadSession.Title

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>



How to maximize the application window:

```
notepadSession.manage().Window.Maximize();
```

How to save screenshot of an application under test:

```
notepadSession.GetScreenshot().SaveAsFile(".\\Screenshot{DateTime.Now.ToString("ddMMyyyyhhmmss")}.png",  
OpenQA.Selenium.ScreenshotImageFormat.Png);
```

How to quit an application:

```
notepadSession.Quit();
```

What are the prerequisites for running AppiumLocalService?

```
Node.js  
Appium on Node.js  
WinAppDriver
```

How can we create an instance of AppiumLocalService:

```
var appiumLocalService = new AppiumServiceBuilder().UsingPort(4723).Build();  
appiumLocalService.Start();
```

How do we create an instance of AppiumLocalService so that log gets written to a specific file?

```
var appiumLocalService = new AppiumServiceBuilder().UsingPort(4723).WithLogFile(new FileInfo(@"C:\Users\My  
Name\Documents\Logs\TestLog.txt")).Build();
```

Udemy course: <https://www.udemy.com/course/appium-winappdriver-automation-testing/?referralCode=ED22C3A4CE5BB5E22E53>