

# Python 3 Installation on Windows

## Step 1: Select Version of Python to Install

The installation procedure involves downloading the official Python .exe installer and running it on your system.

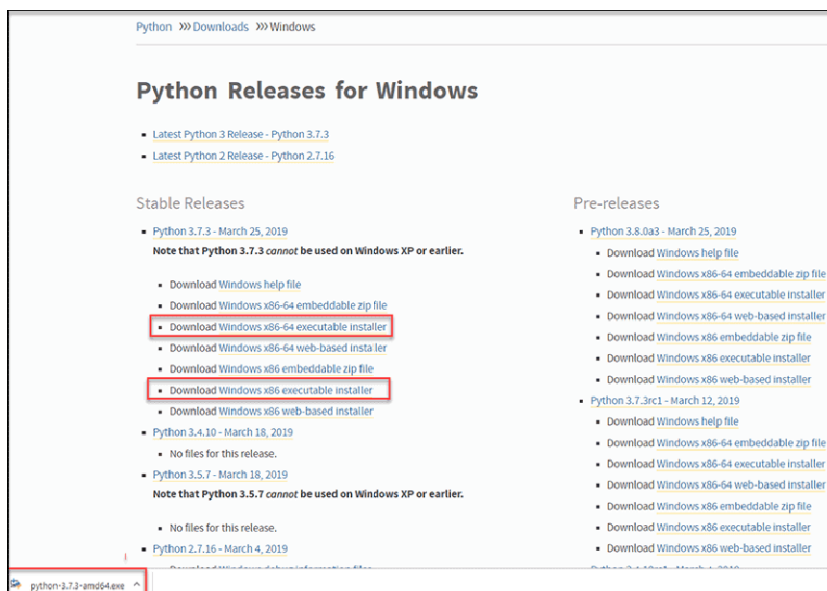
The version you need depends on what you want to do in Python. For example, if you are working on a project coded in Python version 2.6, you probably need that version. If you are starting a project from scratch, you have the freedom to choose.

If you are learning to code in Python, we recommend you **download both the latest version of Python 2 and 3**. Working with Python 2 enables you to work on older projects or test new projects for backward compatibility.

**Note:** If you are installing Python on a remote Windows server, log in via [Remote Desktop Protocol \(RDP\)](#). Once you log in, the installation procedure is the same as for a local Windows machine.

## Step 2: Download Python Executable Installer

1. Open your web browser and navigate to the [Downloads for Windows](#) section of the [official Python website](#).
2. Search for your desired version of Python. At the time of publishing this article, the latest Python 3 release is version 3.7.3, while the latest Python 2 release is version 2.7.16.
3. Select a link to download either the **Windows x86-64 executable installer** or **Windows x86 executable installer**. The download is approximately 25MB.



**Note:** If your Windows installation is a 32-bit system, you need the **Windows x86 executable installer**. If your Windows is a 64-bit version, you need to download the **Windows x86-64 executable installer**. There is nothing to worry about if you install the “wrong” version. You can uninstall one version of Python and install another.

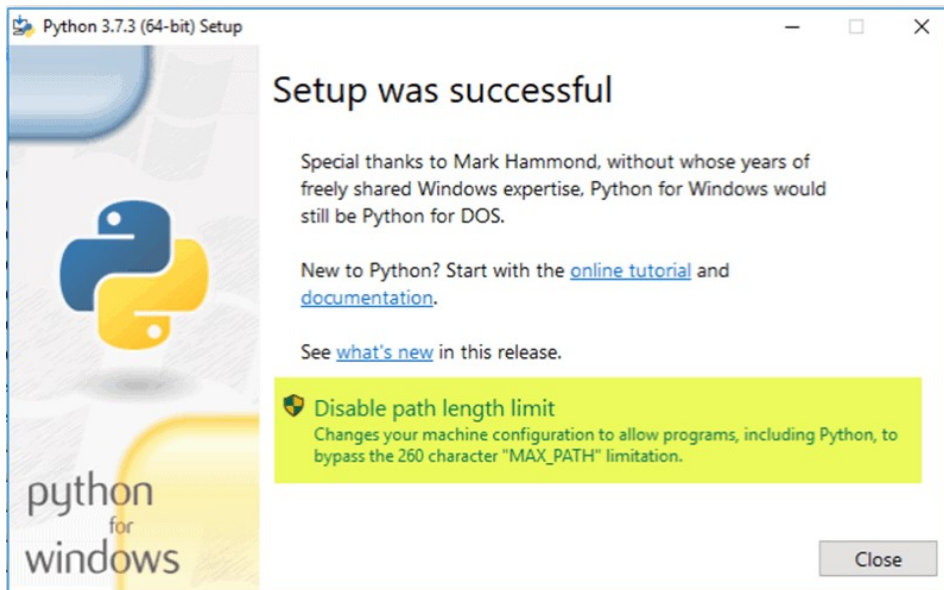
## Step 3: Run Executable Installer

1. Run the **Python Installer** once downloaded. (In this example, we have downloaded Python 3.7.3.)
2. Make sure you select the **Install launcher for all users** and **Add Python 3.7 to PATH** checkboxes. The latter places the interpreter in the execution path. For older versions of Python that do not support the **Add Python to Path** checkbox, see [Step 6](#).
3. Select **Install Now** – the recommended installation options.



For all recent versions of Python, the recommended installation options include **Pip** and **IDLE**. Older versions might not include such additional features.

4. The next dialog will prompt you to select whether to **Disable path length limit**. Choosing this option will allow Python to bypass the 260-character MAX\_PATH limit. Effectively, it will enable Python to use long path names.



The **Disable path length limit** option will not affect any other system settings. Turning it on will resolve potential name length issues that may arise with Python projects developed in Linux.

## Step 4: Verify Python Was Installed On Windows

1. Navigate to the directory in which Python was installed on the system. In our case, it is `C:\Users\Username\AppData\Local\Programs\Python\Python37` since we have installed the latest version.
2. Double-click `python.exe`.
3. The output should be similar to what you can see below:

The image shows a Command Prompt window titled 'Command Prompt - python'. The text inside the window is as follows:

```
Microsoft Windows [Version 10.0.17134.648]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Dejan>python
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
```

**Note:** You can also check whether the installation was successful by typing `python -v` in **Command Prompt**. The output should display your installed version of Python. In our case, it is "Python 3.7.3."

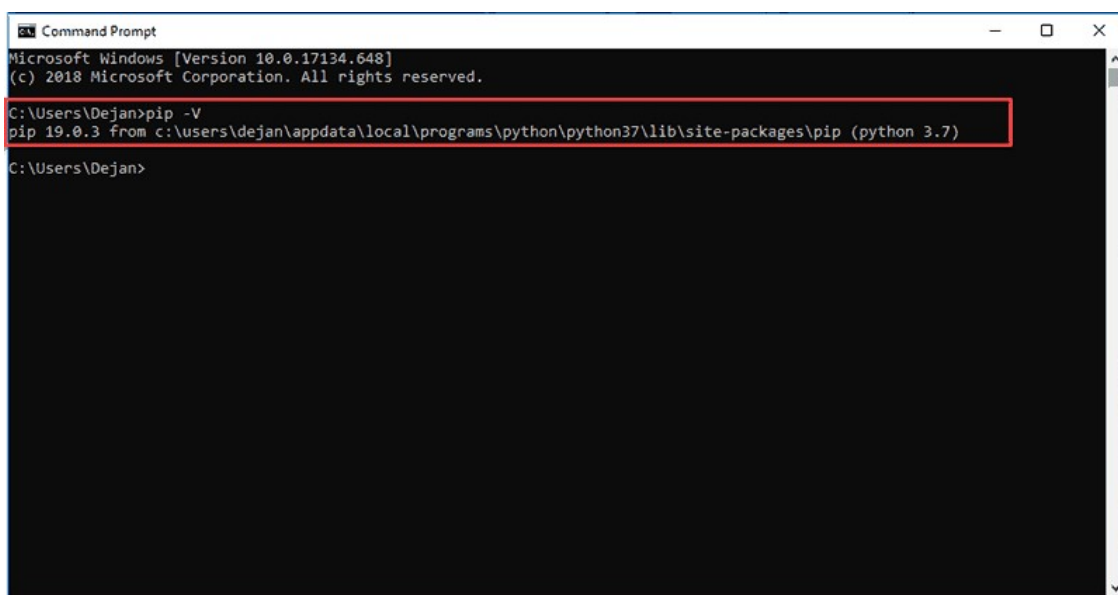
## Step 5: Verify Pip Was Installed

If you opted to install an older version of Python, it is possible that it did not come with Pip preinstalled. Pip is a powerful package management system for Python software packages. Thus, make sure that you have it installed.

We recommend using Pip for most Python packages, especially when working in virtual environments.

To verify whether Pip was installed:

1. Open the **Start** menu and type "**cmd.**"
2. Select the **Command Prompt** application.
3. Enter **pip -v** in the console. If Pip was installed successfully, you should see the following output:



```
Command Prompt
Microsoft Windows [Version 10.0.17134.648]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Dejan>pip -v
pip 19.0.3 from c:\users\dejan\appdata\local\programs\python\python37\lib\site-packages\pip (python 3.7)

C:\Users\Dejan>
```

Pip has not been installed yet if you get the following output:

'pip' is not recognized as an internal or external command,  
Operable program or batch file.

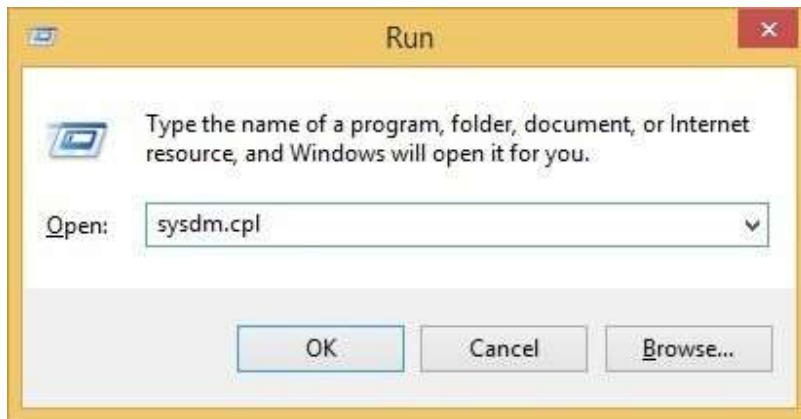
If your version of Python is missing Pip, see our article [How to Install Pip to Manage Python Packages on Windows](#).

## Step 6: Add Python Path to Environment Variables (Optional)

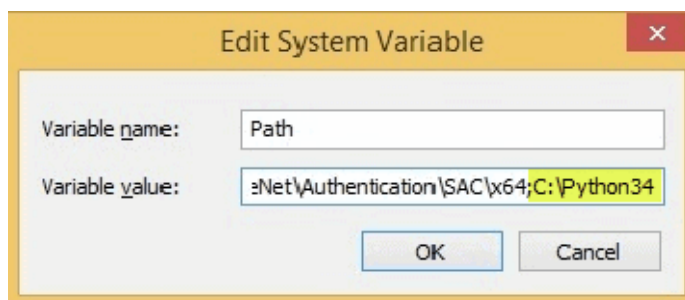
We recommend you go through this step if your version of the Python installer does not include the **Add Python to PATH** checkbox or if you have not selected that option.

Setting up the Python path to system variables alleviates the need for using full paths. It instructs Windows to look through all the PATH folders for “python” and find the install folder that contains the python.exe file.

1. Open the **Start** menu and start the **Run** app.



2. Type **sysdm.cpl** and click **OK**. This opens the **System Properties** window.
3. Navigate to the **Advanced** tab and select **Environment Variables**.
4. Under **System Variables**, find and select the **Path** variable.
5. Click **Edit**.
6. Select the **Variable value** field. Add the path to the **python.exe** file preceded with a **semicolon (;)**. For example, in the image below, we have added **";C:\Python34."**



7. Click **OK** and close all windows.

By setting this up, you can execute Python scripts like this: **Python script.py**

Instead of this: **C:/Python34/Python script.py**

As you can see, it is cleaner and more manageable.

## Step 7: Install virtualenv (Optional)

You have Python, and you have Pip to manage packages. Now, you need one last software package - **virtualenv**. Virtualenv enables you to create isolated local virtual environments for your Python projects.

**Why use virtualenv?**

Python software packages are installed system-wide by default. Consequently, whenever a single project-specific package is changed, it changes for all your Python projects. You would want to avoid this, and having separate virtual environments for each project is the easiest solution.

To install virtualenv:

1. Open the **Start** menu and type "**cmd**."
2. Select the **Command Prompt** application.
3. Type the following **pip** command in the console:

```
C:\Users\Username> pip install virtualenv
```

Upon completion, virtualenv is installed on your system.