How to determine if .NET Core is installed

206

53

I know that for older versions of .NET, you can determine if a given version is installed by following

```
https://support.microsoft.com/en-us/kb/318785
```

Is there an official method of determining if .NET **Core** is installed?

(And I don't mean the SDK, I want to check a server without the SDK, to determine if it has DotNetCore.1.0.0-WindowsHosting.exe installed on it)

I can see

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\.NET Cross-Platform Runtime Environment\.NET
Framework 4.6\Win\v1-rc1
```

with Version# of 1.0.11123.0 on my windows 7 machine, but I don't see the same stuff on my Windows 10 machine.

c# asp.net-core .net-core

edited Jan 31 '18 at 13:24
Pac0
8,656 3 30 49

asked Jul 25 '16 at 12:07
weloytty
2,022 5 22 29

1 Good question. Anyone following .NET Core knows that the Runtime and SDK versioning is a very confusing topic. – Sean Feb 18 '17 at 0:10

@Chiramisu, All of the checked ones below worked for me, but because of some irrelevant implementation details, I went with Desired State Configuration, and used that to ensure that dnc windows server hosting is installed. (I.e I have Ensure=Absent on DotNetCore.1.0.0-WindowsServerHosting.exe and Ensure=Present on DotnetCore.2.0.5-WindowsServerHosting.exe) (or any other filename you can find want). DSC handles all of the mess involved with checking to make sure the appropriate package is installed/uninstalled. – weloytty Mar 23 '18 at 11:23

2 dotnet --list-sdks and dotnet --list-runtimes are available on my host with 2.1.300-preview1-008174 as the active version – jumpercake May 10 '18 at 19:06

Run This below command in powershell dotnet --info Source – manikanta kumar Apr 26 at 6:17

12 Answers

123

Using Powershell:

Runtimes:

```
(dir (Get-Command dotnet).Path.Replace('dotnet.exe',
'shared\Microsoft.NETCore.App')).Name
```

SDK's:

```
(dir (Get-Command dotnet).Path.Replace('dotnet.exe', 'sdk')).Name
```

edited Jan 31 '18 at 13:59

answered Jan 31 '18 at 13:14
Andriy Tolstoy
3,242 1 17 22

1 none worked in win 10 – Marcelo Filho Mar 1 '18 at 19:17

1 @MarceloFilho Which version do you have? You can get it using `[System.Environment]::OSVersion` . I tested above-mentioned commands using Windows 10 Version 10.0.15063.0. It works fine. – Andriy Tolstoy Mar 2 '18 at 8:06

4 I can confirm this works perfectly on Windows server 2016 and windows 10. – user5389726598465 May 1 '18 at 9:58

5 That's actually cool. But we should simply use Chiramisu's answer instead. – rsenna Jun 27 '18 at 12:20

2 Works in PowerShell on Windows 10 but I prefer `dotnet --info` suggested in other answers. – Manfred Oct 24 '18 at 2:20

252

Great question, and ~~the answer is not a simple one. There is no "show me all .net core versions" command, but there's hope.~~

EDIT:

I'm not sure when it was added, but the info command now includes this information in its output. It will print out the installed runtimes and SDKs, as well as some other info:

```
dotnet --info
```

If you only want to see the SDKs: `dotnet --list-sdks`

If you only want to see installed runtimes: `dotnet --list-runtimes`

I'm on Windows, but I'd guess that would work on Mac or Linux as well with a current version.

Also, you can reference the [.NET Core Download Archive]() to help you decipher the SDK versions.

OLDER INFORMATION: Everything below this point is old information, which is less relevant, but may still be useful.

See installed **Runtimes**:

Open `C:\Program Files\dotnet\shared\Microsoft.NETCore.App` in Windows Explorer

See installed **SDK's**:

Open `C:\Program Files\dotnet\sdk` in Windows Explorer

(Source for the locations: [A developer's blog]())

In addition, you can see the **latest** Runtime and SDK versions installed by issuing these commands at the command prompt:

`dotnet` *Latest Runtime version is the first thing listed.* **DISCLAIMER:** This no longer works, but may work for older versions.

`dotnet --version` *Latest SDK version* **DISCLAIMER:** Apparently the result of this may be affected by any global.json config files.

On macOS you could check .net core version by using below command.

```
ls /usr/local/share/dotnet/shared/Microsoft.NETCore.App/
```

On Ubuntu or Alpine:

```
ls /usr/share/dotnet/shared/Microsoft.NETCore.App/
```

It will list down the folder with installed version name.

edited Nov 14 '18 at 3:17

answered Feb 18 '17 at 0:08

Sean

4,904 3 24 28

1 @SergiiVolchkov thanks for comment. Can you please tell me how to uninstall dotnet

122 The correct answer for runtime-only environments without the SDK, such as a server with the Windows Hosting package installed, is to run PowerShell with the following command:

```
dotnet --info
```

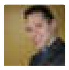
Per the official documentation:

- The `--version` option "Prints out the version of the .NET Core SDK in use." and therefore doesn't work if the SDK is not installed. Whereas...
- The `--info` option "Prints out detailed information about the CLI tooling and the environment, such as the current operating system, commit SHA for the version, and other information."

Here's another official article explaining how .NET Core versioning works. :)

edited Oct 20 '17 at 22:53

answered Oct 20 '17 at 22:44



Chiramisu
3,160 6 38 70

8 What's incredible is this is the actual answer. And it's buried underneath a stack of answers from people that didn't even bother to read the question correctly. – Jammer Apr 12 '18 at 8:53

`--info` doesn't work on my server, whereas `--version` does work. The info option gives me: `Did you mean to run dotnet SDK commands? Please install dotnet SDK from: http://go.microsoft.com/fwlink/?LinkID=798306&clcid=0x409` – ArieKanarie Jun 4 '18 at 13:29

@ArieKanarie You might need to repair using the Microsoft .NET Framework Repair Tool. – Chiramisu Jun 4 '18 at 18:44

4 This is way better than the hacky accepted answer :(– Mariusz Jamro Jun 9 '18 at 19:09

73 You can check if dotnet.exe is available:

```
where dotnet
```

You can then check the version:

```
dotnet --version
```

UPDATE: There is now a better way of doing this, which is well explained in many other answers:

```
dotnet --info
```

edited May 30 at 16:22

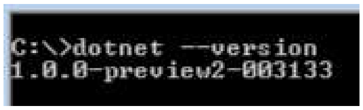
answered Jul 25 '16 at 14:08

Robert Paulsen
2,683 1 14 24

46

One of the dummies ways to determine if `.NET Core` is installed on Windows is:

- Press Windows + R
- Type `cmd`
- On the command prompt, type `dotnet --version`



```
C:\>dotnet --version
1.0.0-preview2-003133
```

If the `.NET Core` is installed, we should not get any error in the above steps.

answered Oct 18 '16 at 5:47
student
12.2k 10 79 137

5 See comment above to the same answer: It outpus dotnet CLI version, not runtime.. It's two different things. Having CLI installed, doesn't mean runtime is installed and if it's same version – Michael Freidgeim Sep 13 '17 at 3:58

Nitpcking, but you can't have a CLI that works without a runtime installed. So if you have a CLI, you will have *some* runtime, it's just that it may be a completely different version. – Omair Majid Dec 13 '17 at 23:06

@omajid Not sure what CLI you're talking about. The CLI I'm referring to in my answer is the default command prompt CLI on Windows 10 x64 which is installed/available without special installations – student Dec 14 '17 at 5:58

1 This is correct & should be accepted answer – Irfan Ashraf Feb 14 '18 at 11:23

@IrfanAshraf thanks for your kind words! Anyways, only the asker have the ability to accept it as an answer – student Feb 17 '18 at 10:39

17

(1) If you are on the Window system.

Open the command prompt.

```
 dotnet --version
```

(2) Run the below command If you are on Linux system.

```
dotnet --version

dotnet --info
```

edited Apr 2 '18 at 9:24
JohnLBevan
14.8k 1 49 115

answered Dec 2 '17 at 5:37
Hiren Parghi
742 8 22

13

I work primarily with Windows development machines and servers.

I just wanted to point out (at least for NET.Core 2.0 and above) the only thing needed is to execute `dotnet --info` in a command prompt to get information about the **latest** version installed. If .NET Core is installed you will get some response.

On my development machine (Windows 10) the result is as follows. SDK is 2.1.2 and runtime is 2.0.3.

```
.NET Command Line Tools (2.1.2)

Product Information:
 Version:            2.1.2
 Commit SHA-1 hash:  5695315371

Runtime Environment:
 OS Name:     Windows
 OS Version:  10.0.15063
 OS Platform: Windows
 RID:         win10-x64
 Base Path:   C:\Program Files\dotnet\sdk\2.1.2\

Microsoft .NET Core Shared Framework Host

  Version  : 2.0.3
  Build    : a9190d4a75f4a982ae4b4fa8d1a24526566c69df
```

On one of my servers running Windows Server 2016 with Windows Server Hosting pack (no SDK) result is as follows. No SDK, runtime is 2.0.3.

```
Microsoft .NET Core Shared Framework Host

Version  : 2.0.3
Build    : a9190d4a75f4a982ae4b4fa8d1a24526566c69df
```

Cheers !

answered Jan 5 '18 at 19:30
Hakan Çelik MK1
317 4 7

The following commands are available with .NET Core SDK 2.1 (v2.1.300):

8

To list all installed .NET Core SDKs use: `dotnet --list-sdks`

To list all installed .NET Core runtimes use `dotnet --list-runtimes`

(tested on Windows as of writing, 03 Jun 2018, and again on 23 Aug 2018)

Update as of 24 Oct 2018: Better option is probably now `dotnet --info` in a terminal or PowerShell window as already mentioned in other answers.

edited Oct 24 '18 at 2:18

answered Jun 2 '18 at 22:26
Manfred
4,046 2 19 24

I didn't downvote, but I speculate it may be because you mentioned commands "available with .NET Core SDK", whereas question states "I want to check a server without the SDK". Your answer would be improved if you determine which of the above commands work when *only runtime* is installed. – ToolmakerSteve Mar 14 at 13:48

@ToolmakerSteve Yes, that could be the point. I don't have an environment, though, without the SDK and wouldn't want to go to that length removing it.... Thank you for your comment, though. Very much appreciated. – Manfred Mar 17 at 21:58

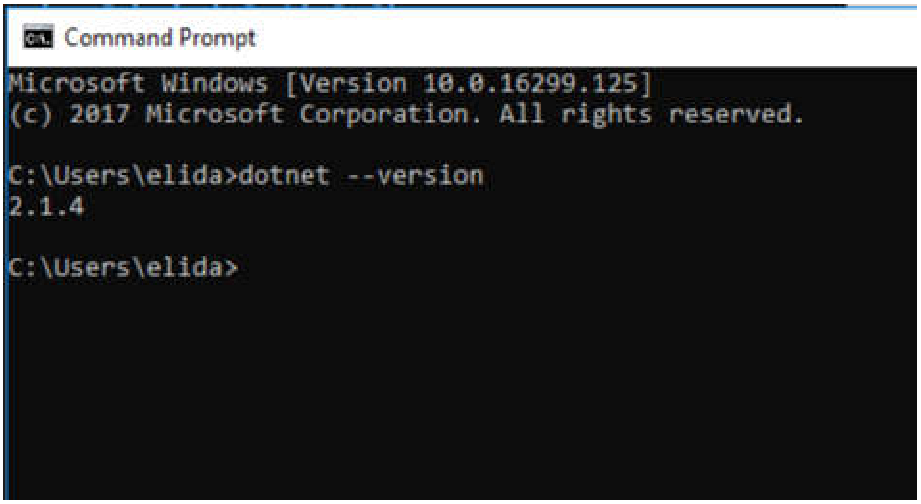
5

On windows, You only need to open the command prompt and type:

```
dotnet --version
```

If the .net core framework installed you will get current installed version

see screenshot:



```
Microsoft Windows [Version 10.0.16299.125]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\elida>dotnet --version
2.1.4

C:\Users\elida>
```

edited May 30 '18 at 19:40

answered May 30 '18 at 19:35
Eli Dagan
525 3 13

2 This gives you the SDK version, not the runtime version – silkfire Oct 6 '18 at 10:29

0

Look in `C:\Program Files\dotnet\shared\Microsoft.NETCore.App` to see which versions of the runtime have directories there. Source.

A lot of the answers here confuse the SDK with the Runtime, which are different.

answered Oct 23 '18 at 16:40
jaycer
1,525 2 16 29

0

After all the other answers, this might prove useful.

Open your application in Visual Studio. In Solutions Explorer, right click your project. Click Properties. Click Application. Under "Target Framework" click the dropdown button and there you are, all of the installed frameworks.

BTW - you may now choose which framework you want.

answered May 15 at 14:17
Baruch Atta
320 2 10

I used Visual Studio 2017. YMMV. – Baruch Atta May 15 at 14:37

-6

It doesn't need a installation process.

I have pinned "VSCore" on my taskbar (win10), so open it, and open a task manager choose "Visual Studio Core" process expand left arrow and over any of them child process right button over it and click in "Open File Location" menu.

If you don't remember where is installed search "Code.exe" file in all your hard drives.

answered Apr 24 '18 at 14:06
JoseRamos
1

1 The question is about .NET CORE not the ide VS CODE. – user5389726598465 May 1 '18 at 9:54
