

# What is the Bash file extension?

Asked 4 years, 9 months ago   Active 9 months ago   Viewed 61k times



65



15

I have written a bash script in a text editor, what extension do I save my script as so it can run as a bash script? I've created a script that should in theory start an ssh server. I am wondering how to make the script execute once I click on it. I am running OS X 10.9.5.

bash

shell

edited Jan 19 at 1:55



codeforester

21k 8 48 77

asked Jan 7 '15 at 6:23



Amedeo

454 1 5 9

- 3 Shell script doesn't require any particular extension. Just execute it as `bash myscript` – [anubhava](#) Jan 7 '15 at 6:24
- 5 Its usually `.sh` , but the extension isn't required to exist at all. Linux is not Windows. The program that shall interpret your script is determined in its first line, that should be `#!/bin/bash` . It may even include parameters. – [Havenard](#) Jan 7 '15 at 6:25
- 1 @anubhava How would i get my script to execute if i were just to double click on it and not actually type " `bash myscript`" – [Amedeo](#) Jan 7 '15 at 6:27
- 2 @Amedeo that would be heading your file with the line `#!/bin/bash` . – [Havenard](#) Jan 7 '15 at 6:28
- 3 No extensions: [Commandname extensions considered harmful](#). – [gniourf\\_gniourf](#) Jan 9 '15 at 16:17

## 5 Answers



77



Disagreeing with the other answers, there's a common convention to use a `.sh` extension for shell scripts -- but it's not a useful convention. It's better not to use an extension at all. The advantage of being able tell that `foo.sh` is a shell script because of its name is minimal, and you pay for it with a loss of flexibility.

To make a bash script executable, it needs to have a [shebang](#) line at the top:



`#!/bin/bash`

and use the `chmod +x` command so that the system recognizes it as an executable file. It then needs to be installed in one of the directories listed in your `$PATH`. If the script is called `foo`, you can then execute it from a shell prompt by typing `foo`. Or if it's in the current directory (common for temporary scripts), you can type `./foo`.

Neither the shell nor the operating system pays any attention to the extension part of the file name. It's just part of the name. And by *not* giving it a special extension, you ensure that anyone (either a user or another script) that uses it doesn't have to care how it was implemented, whether it's a shell script (sh, bash, csh, or whatever), a Perl, Python, or Awk script, or a binary executable. The system is specifically designed so that either an interpreted script or a binary executable can be invoked without knowing or caring how it's implemented.

UNIX-like systems started out with a purely textual command-line interface. GUIs like KDE and Gnome were added later. In a GUI desktop system, you can typically run a program (again, whether it's a script or a binary executable) by, for example, double-clicking on an icon that refers to it. Typically this discards any output the program might print and doesn't let you pass command-line arguments; it's much less flexible than running it from a shell prompt. But for some programs (mostly GUI clients) it can be more convenient.

Shell scripting is best learned from the command line, not from a GUI.

(Some tools *do* pay attention to file extensions. For example, compilers typically use the extension to determine the language the code is written in: `.c` for C, `.cpp` for C++, etc. This convention doesn't apply to executable files.)

Keep in mind that UNIX (and UNIX-like systems) are not Windows. MS Windows generally uses a file's extension to determine how to open/execute it. Binary executables need to have a `.exe` extension. If you have a UNIX-like shell installed under Windows, you can configure Windows to recognize a `.sh` extension as a shell script, and use the shell to open it; Windows doesn't have the `#!` convention.

edited Oct 17 '18 at 15:59

answered Jan 7 '15 at 16:41



[Keith Thompson](#)

**204k** 28 315 510

- 
- 2 What I'm aiming for is what you mentioned in the fourth paragraph. Running my script when clicked on an icon that refers to it. – [Amedeo](#) Jan 7 '15 at 23:34
- 
- 2 @Amedeo: Then it depends on your desktop environment. In the one I use (Cinnamon under Ubuntu), double-clicking on the icon for an executable script prompts me to run it in a terminal, display it in an editor, or run it without a terminal. It does this regardless of the file extension. – [Keith Thompson](#) Jan 24 '17 at 22:10
- 
- 3 (Necro) If you omit the extension, you cannot then have a same-name folder. So, for instance, you have `deploy.sh` (or `deploy.bash`), and a folder, `deploy`, with additional deployment logic. If you rename the script to simply `deploy` it'll cause a name conflict. Naming either file or folder differently hurts manageability and possibly file sorting ( `ls`, in editors, etc.). Of course, the shebang is - in the end - the determinant. But file extensions do have their rightful place. – [Kafoso](#) Oct 31 '17 at 11:55
- 
- 2 @Kafoso: I don't think I've ever felt a need to have a script and a directory with the same name. If I did, I might call the directory `Deploy`, though that

can cause problems when copying to a case-insensitive file system. – [Keith Thompson](#) Oct 31 '17 at 16:29

- 2 On a Mac, double clicking files *always* opens them in the default app. Thus it's helpful to have a `.sh` extension so that the script opens in your desired editor. If you want to run a script when double clicked, give it the `.command` extension and it'll run in Terminal when double clicked. – [BallpointBen](#) Aug 20 '18 at 3:04

You don't need any extension (or you could choose an arbitrary one, but `.sh` is a useful convention).

15

You should start your script with `#!/bin/bash` (that first line is understood by [execve\(2\)](#) syscall), and you should make your file executable by `chmod u+x .` so if your script is in some file `$HOME/somedir/somescriptname.sh` you need to type once

```
chmod u+x $HOME/somedir/somescriptname.sh
```

in a terminal. See [chmod\(1\)](#) for the command and [chmod\(2\)](#) for the syscall.

Unless you are typing the whole file path, you should put that file in some directory mentioned in your `PATH` (see [environ\(7\)](#) & [execvp\(3\)](#)), which you might set permanently in your `~/.bashrc` if your login shell is `bash` )

BTW, you could write your script in some other language, e.g. in Python by starting it with `#!/usr/bin/python` , or in Ocaml by starting it with `#!/usr/bin/ocaml` ...

Executing your script by double-clicking (on what? you did not say!) is a [desktop environment](#) issue and could be desktop specific (might be different with Kde, Mate, Gnome, .... or IceWM or RatPoison). Perhaps reading [EWMH](#) spec might help you getting a better picture.

Perhaps making your script executable with `chmod` might make it clickable on your desktop (apparently, [Quartz](#) on MacOSX). But then you probably should make it give some visual feedback.

And several computers don't have any desktop, including your own when you access it remotely with [ssh](#).

I don't believe it is a good idea to run your shell script by clicking. You probably want to be able to give arguments to your shell script (and how would you do that by clicking?), and you should care about its output. If you are able to write a shell script, you are able to use an interactive shell in a terminal. That it the best and most natural way to use a script. Good interactive shells (e.g. [zsh](#) or [fish](#) or perhaps a recent `bash` ) have delicious and configurable [autocompletion](#) facilities and you won't have to type a lot (learn to use the `tab` key of your keyboard). Also, scripts and programs are often parts of composite commands (pipelines, etc...).

PS. I'm using Unix since 1986, and Linux since 1993. I never started my own programs or scripts by clicking. Why should I?

edited Jan 19 at 2:00

[codeforester](#)

answered Jan 7 '15 at 6:27

[Basile Starynkevitch](#)



21k 8 48 77



186k 15 192 405

- 
- 3 Okay, so I have created my script in a text editor. I save the file to my desktop. When i click on my script thats saved on my desktop I want it to actually run upon me clicking on it. – [Amedeo](#) Jan 7 '15 at 6:35
- 
- 6 I don't know what is your desktop (KDE, Gnome, MATE, ...). I strongly invite you to use the command line in a terminal, especially to run your scripts (you probably want to give them some arguments; how would you do that on the desktop?). If you are able to code a shell script you should be able to use the shell interactively in a terminal – [Basile Starynkevitch](#) Jan 7 '15 at 6:37
- 
- 2 My point is that if you are coding a shell script you should take the habit of using a command line in a terminal. – [Basile Starynkevitch](#) Jan 7 '15 at 6:43
- 
- 2 I understand that, its for a project I'm working. I want the script to execute once clicked on. Im trying to avoid using terminal to run the script. – [Amedeo](#) Jan 7 '15 at 6:52
- 
- 3 I'd use `chmod +x` rather than `chmod u+x` , unless there's a specific reason to restrict executability to the owner. – [Keith Thompson](#) Jan 7 '15 at 15:38
- 

just `.sh` .

2

Run the script like this:

`./script.sh`

EDIT: Like anubhava said, the extension doesn't really matter. But for organisational reasons, it is still recommended to use extensions.

answered Jan 7 '15 at 6:25

[Marc Anton Dahmen](#)

963 5 6

- 
- 6 If you're running a script, you generally have no reason to care what it's written in. A bash script and a binary executable are executed the same way. Adding a `.sh` suffix to an executable script is generally useless clutter. – [Keith Thompson](#) Jan 7 '15 at 8:21
- 
- 1 yes, but as i said in my edit - it is a convention to organize scripts - not more?! – [Marc Anton Dahmen](#) Jan 7 '15 at 16:05
- 
- 4 I do see a lot of scripts with a `.sh` extension (and fewer with a `.bash` extension), but I'm not convinced it's at all useful. If I name a script `foo.sh` and I later decide to reimplement it in Perl, I can either change the name (and edit everything that uses it) or leave it with a misleading extension. If I name it `foo` , I don't have that problem. – [Keith Thompson](#) Jan 7 '15 at 16:20
-



I know this is quite old now but I feel like this adds to what the question was asking for.

2



If your on a mac and you want to be able to run a script by double clicking it you need to use the `.command` extension. Also same as before make file executable with `chmod -x .`

As was noted before, this isn't really that useful tbh.

answered Apr 4 '18 at 14:45



w\_jay

141 4 8



0



**TL;DR -- If the user (not necessarily the developer) of the script is using a GUI interface, it depends on what file browser they are using. MacOS's Finder will require the `.sh` extension in order to execute the script. Gnome Nautilus, however, recognizes properly shebanged scripts with or without the `.sh` extension.**

I know it's already been said multiple times the reasons for and against using an extension on bash scripts, but not as much why or why not to use extensions, but I have what I consider to be a good rule of thumb.

If you're the type who hops in and out of bash and using the terminal in general or are developing a tool for someone else who does not use the terminal, put a `.sh` extension on your bash scripts. That way, users of that script have the option of double-clicking on that file in a GUI file browser to run the script.

If you're the type who primarily does all or most of your work in the terminal, don't bother putting any extension on your bash scripts. They would serve no purpose in the terminal, assuming that you've already set up your `~/.bashrc` file to visually differentiate scripts from directories.

Edit:

In the Gnome Nautilus file browser with 4 test files (each with permissions given for the file to be executed) with stupidly simple bash command to open a terminal window ( `gnome-terminal` ):

1. A file with NO extension with `#!/bin/bash` on the first line.

It worked by double-clicking on the file.

2. A file with a `.sh` extension with `#!/bin/bash` on the first line.

It worked by double-clicking on the file.

3. A file with NO extension with NO `#!/bin/bash` on the first line.

It worked by double-clicking on the file...technically, but the GUI gave no indication that it was a shell script. It said it was just a plain text file.

4. A file with a `.sh` extension with NO `#!/bin/bash` on the first line.

It worked by double-clicking on the file.

However, as Keith Thompson, in the comments of this answer, wisely pointed out, relying on the using the `.sh` extension instead of the bash shebang on the first line of the file ( `#!/bin/bash` ) it could cause problems.

Another however, I recall when I was previously using MacOS, that even properly shebanged (is that a word?) bash scripts without a `.sh` extension could not be run from the GUI on MacOS. I would love for someone to correct me on that in the comments though. If this is true, it would prove that there is at least one file browser out there where the `.sh` extension matters.

edited Jan 19 at 1:40

answered Dec 20 '18 at 13:51




Ryan Hart

31 4

---

Which GUI file browser do you use (or, more relevantly, what do your users use)? Does it use the extension to decide how to run a script? – [Keith Thompson](#) Jan 15 at 22:42

---

Well, the platform that I have the most experience on is MacOS. In Finder, the user decides which apps to associate with what extension. Now, I'm on Linux using Gnome, but I haven't got around to experimenting how Gnome's Nautilus file browser interacts with files with no extension yet. – [Ryan Hart](#) Jan 17 at 21:00 

---

If you have an executable script with a `.sh` suffix and `#!/bin/bash` on the first line, is a GUI file browser going to use `sh` to invoke it (ignoring the shebang)? If so, that could cause some problems. (The answer could be different for different browsers.) – [Keith Thompson](#) Jan 17 at 23:07

---

Good points there. I've updated my answer above with some actual testing I did just now. – [Ryan Hart](#) Jan 19 at 1:41

---

Interesting, but ... You said it "worked". For each of the 4 test cases, it would be good to know whether it invoked it with `/bin/bash` or `/bin/sh` (the latter is the default for a script with no shebang). If `/bin/sh` is a symlink to `/bin/bash` (which is fairly common) then you can tell by looking at the value of `$0`. (Also, if bash is invoked as `sh` it sets `POSIXLY_CORRECT=y`.) – [Keith Thompson](#) Jan 19 at 3:14

---

protected by [codeforester](#) Jan 19 at 1:54

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 [reputation](#) on this site (the [association bonus does not count](#)).

Would you like to answer one of these [unanswered questions](#) instead?