

What is the difference between Shell, Kernel and API

More jobs
means more choice


Get started


13 I want to understand how this applies to an operating system and also to those things that are not infact operating systems. I can't understand the difference between the three and their essence. API is the functions we can call but what is Shell? If we have an API than what exactly is the Kernel of the operating system? I understand the an operating system has a Core that is not going to change and this core does the fundamental Job of a typical OS while we may have different user interfaces like GUI or command line with the same Kernel. So the problem is I am confused how these things are different. Aaaaaaarhg!

5 Can the functions like printf and fopen in C be called API calls?

[api](#) [shell](#) [operating-system](#) [kernel](#)

[share](#) [edit](#)

asked Aug 26 '12 at 17:19



[quantum231](#)
560 2 7 25

[add a comment](#)

4 Answers

[active](#) [oldest](#) [votes](#)

- 17
- A **shell** is a command interpreter, i.e. the program that either process the command you enter in your terminal emulator (interactive mode) or process shell scripts (text files containing commands) (batch mode). In early Unix times, it used to be the unique way for users to interact with their machines. Nowadays, graphical environments are replacing the shell for most casual users.
 - A **kernel** is a low level program interfacing with the hardware (CPU, RAM, disks, network, ...) on top of which applications are running. It is the lowest level program running on computers although with virtualization you can have multiple kernels running on top of virtual machines which themselves run on top of another operating system.
 - An **API** is a generic term defining the interface developers have to use when writing code using libraries and a programming language. **Kernels have no APIs** as they are not libraries. They do have an **ABI**, which, beyond other things, define how do applications interact with them through system calls. Unix application developers use the standard C library (eg: `libc` , `glibc`) to build ABI compliant binaries. `printf(3)` and `fopen(3)` are not wrappers to system calls but `(g)libc` standard facilities. The low level system calls they eventually use are `write(2)` and `open(2)` and possibly others like `brk` , `mmap` . The number in parentheses is a convention to tell in what manual the command is to be found.

The first volume of the Unix manual pages contains the **shell** commands.

The second one contains the system call **wrappers** like `write` and `open` . They form the interface to the **kernel**.

The third one contains the standard library (including the Unix standard **API**) functions (excluding system calls) like `fopen` and `printf` . These are **not** wrappers to specific system calls but just code using system calls when required.

[share](#) [edit](#)

edited Aug 1 '13 at 14:13

answered Aug 26 '12 at 22:29

[jlliagre](#)



16.5k 2 26 44

Than I wish to know one more thing. Is there any such thing as Windows Shell? What about situations where we have a GUI instead of a command line prompt? This is the only last thing I do not yet understand. –

[quantum231](#) Aug 31 '12 at 23:42

shell is a term initially and still mostly used only in an Unix context. Anyway, the fact there is a GUI doesn't make a difference. Whether you use windows, Gnome or KDE, you can run a shell or equivalent in a command line window (terminal emulator). – [jlliagre](#) Sep 1 '12 at 20:54

IMHO, "system call" is OS's API for implementing certain programming language's cross-platform standard library. Actually, fopen & printf are indeed kind of "wrapper" to system calls. some references: stackoverflow.com/questions/13179102 – [Bossliaw](#) Aug 1 '13 at 5:54

- 1 [@Bossliaw](#) fopen and printf have a lot of specific code and share tables, buffers and context with each other. For that reason, I don't call them wrappers to system calls, especially as for example you can call printf without any system call being involved. They just sometimes happen to eventually use system calls but this is true for every library function. The link you posted as reference which I totally agree with doesn't say anything that contradict that point. – [jlliagre](#) Aug 1 '13 at 9:06

[add a comment](#)

Get personalized
job matches now



 **stackoverflow**
JOBS

[Get started](#)

6

The Shell is the way to communicate with the OS and kernel by command line. The Shell does this by also calling the API. The kernel is indeed the core of the OS and does memory management, task scheduling, handles with filesystems, I/O handling,... All the things that the kernel does, can in some way be invoked by the API the OS provides.

printf and fopen are wraps around the system calls (API) provided by the OS and kernel

[share](#) [edit](#)

answered Aug 26 '12 at 17:27



[hamon](#)

648 3 7

[add a comment](#)

4

Shell: It is like a command line interface to your operating system. Commands like ls, ps, kill and many more can be used to request to complete the specific operation to the OS. It is like "cmd" on windows.

Kernel: It is the main code of any operating system. Any request you give on shell or through GUI (like memory allocation, opening a file etc) are finally fulfilled by Kernel.

And yes, the calls you mentioned are regarded as API calls. The request to these calls are also handled by Kernel. Please go the below link to find API calls in unix.http://www.mksoftware.com/docs/api_index.asp

This is the overall picture in a unix os:

Applications => (shell+library routines) => system calls => kernel

look the final request handler is the Kernel. Thx!

[share](#) [edit](#)

answered Aug 26 '12 at 17:29



[rahul](#)

3,077 1 14 30

Ahah, so Shell is merely a command line. So the cmd in windows is the Shell for windows or DOS? – [quantum231 Aug 26 '12 at 18:09](#)

Neither, because windows (and other OS) do not need to have a shell. Maybe shell should be renamed as default interface to the OS services? – [Christopher Aug 26 '12 at 18:17](#)

So our windows GUI environment with all the icons is not Shell or is it? – [quantum231 Aug 26 '12 at 19:52](#)

Is the Dos prompt Dos shell? Also, in order to have Linux Shell we shall have to run the Shell on the Windows OS right? – [quantum231 Aug 26 '12 at 19:54](#)

SO we communicate with the Kernel using API and these API calls can be made through a Shell when we are making direct requests to the OS e.g Open file, change directory e.t.c. Thus the DIR shall be an API call in DOS to its Kernel? – [quantum231 Aug 26 '12 at 19:55](#)

[add a comment](#)

1 Consider an example, You are watching the movie is on **shell** and actually process done over hardware is the **kernel**. **shell** is approximately work as that of os for user and software interface and **kernel** work as that of os for software and hardware.

[share](#) [edit](#)

[edited Jul 10 '13 at 9:07](#)



[Uwe Plonus](#)

6,807 1 16 32

[answered Jul 10 '13 at 8:44](#)



[Sanket](#)

11 1

And what is API in this example? – [Artemix Jul 10 '13 at 9:17](#)
