Filesystem Hierarchy Standard (FHS)

FHS Specification Series[1]

- based on two independent distinctions:
  - shareable vs. unshareable
  - variable vs. static
  - example:
    - *sharable*: user home dirs are shareable, device lock files not
    - *static*: binaries, libraries; files that do not change without system admin invtervention

root filesystem

dir	description
bin	Essential command binaries
boot	Static files of the boot loader
dev	Device files
etc	Host-specific system configuration
lib	Essential shared libraries and kernel modules (in subdir/link `moduels`)
media	Mount point for removable media
mnt	Mount point for mounting a filesystem temporarily
opt	Add-on application software packages
run	Data relevant to running processes
sbin	Essential (vital) system binaries and root-only commands
srv	Data for services provided by this system
tmp	Temporary files
usr	Secondary hierarchy
var	Variable data

- Optional:
  - `/home` user home dirs
  - `/root` home dir for root user
  - `/lib<qual>` alternate format essential libraries, e.g.

```
$ ls /lib
lib/   lib64/
```

- **/sys** *not in table above*
  - the location where information about devices, drivers, and some kernel features is exposed.

  sysfs is a ram-based filesystem [...]. It provides a means to export kernel data structures, their attributes, ...

- **/proc** *not in table above*
  - referred to as *process information pseudo-file system*
  - regarded as '*control and information center for the kernel*'
    - many sys utils just use the files in proc
    - `lsmod` is the same as `cat /proc/modules`

[1] https://refspecs.linuxfoundation.org/fhs.shtml

  - contains runtime system information e.g. mounted devices, hardware, configuration
  - read/change kernel parameters by using files in proc (`sysctl`)
  - *Note:* all files in '/proc' have a file size of '0' (with the exception of kcore, mtrr and self)
  - detail description in man page `man 5 proc`
  - see also
    * tldp: 1.14. /proc[2]
    * proc man page[3]
    * kernel.org doc on 'proc'[4]

- **/run**

  The purposes of this directory were once served by `/var/run`. In general, programs may continue to use `/var/run` to fulfill the requirements set out for /run for the purposes of backwards compatibility. E.g. on both Ubuntu and ArchLinux

```
$ ls -ld /var/run
lrwxrwxrwx 6 root 19 Jan 02:32 /var/run -> ../run
```

- Examples entries in 'run':

```
/run/sshd.pid
$ ls /srv/
ftp  http
```

- **/tmp**

  Programs must not assume that any files or directories in `/tmp` are preserved between invocations of the program.

/usr/*

dir	description
bin	Most user commands
lib	Libraries
local	Local hierarchy (empty after main installation)
sbin	Non-vital system binaries
share	Architecture-independent data

- Optional:
  - `games` games and educational binaries
  - `include` header files included by C programs
  - `libexec` binaries run by other programs
  - `lib<qual>` alternate Format Libraries
  - `src` source code
- **/urs/bin**

  primary directory of executable commands on the system. E.g. python, perl, etc
  /bin contains *essential* user command binaries such as `mount`, `rm`, `ls` etc

```
[archlinux@archlinux ~]$ ls -ld /bin
lrwxrwxrwx 7 root 19 Jan 02:32 /bin -> usr/bin
```

[2] https://tldp.org/LDP/Linux-Filesystem-Hierarchy/html/proc.html
[3] https://man7.org/linux/man-pages/man5/proc.5.html
[4] https://www.kernel.org/doc/html/latest/filesystems/proc.html

```
ubuntu@ubuntu20:~$ ls -ld /bin
lrwxrwxrwx 1 root root 7 Feb  1 17:20 /bin -> usr/bin
```

- **/usr/local**

  The /usr/local hierarchy is for use by the system administrator when installing software locally. It needs to be safe from being overwritten when the system software is updated.

- Requires the following sub-dirs (exerpt): `bin`, `etc`, `include`, `share`, etc

- **/usr/sbin**

  > . . . non-essential binaries used exclusively by the system administrator. Note: System admin programs required for system repair, system recovery, mounting /usr, or other essential functions must be placed in /sbin instead. No subdirectories allowed.

  ```
  [archlinux@archlinux ~]$ ls -ld /usr/sbin
  lrwxrwxrwx 3 root 19 Jan 02:32 /usr/sbin -> bin
  [archlinux@archlinux ~]$ ls -ld /sbin
  lrwxrwxrwx 7 root 19 Jan 02:32 /sbin -> usr/bin

  ubuntu@ubuntu20:/usr$ ls -ld /usr/sbin/
  drwxr-xr-x 2 root root 16384 May 23 09:58 /usr/sbin/
  ubuntu@ubuntu20:/usr$ ls -l /sbin
  lrwxrwxrwx 1 root root 8 Feb  1 17:20 /sbin -> usr/sbin
  ```

- **/usr/share**

  all read-only architecture independent (i386, Alpha, etc) data files. E.g. the following directories (or symlinks) must be in `/usr/share`

dir	description
man	man pages
misc	Misc arch-independent data

/var/*

- Variable data files e.g.
  - spool directories
  - administrative data
  - logging data
  - temp and transient files
- contains both
  - shareable portions (e.g. `/var/mail`, `/var/cache/fonts`)
  - non-shareable portions (e.g. `/var/lock`, `/var/log`)

dir	description
cache	Application cache data
lib	Variable state information
local	Variable data for /usr/local
lock	Lock files
log	Log files and directories
opt	Variable data for /opt
run	Data relevant to running processes
spool	Application spool data

dir	description
tmp	Temporary files preserved between system reboots

- **/var/lib**

  This hierarchy holds state information pertaining to an application or the system. State information is data that programs modify while they run, and that pertains to one specific host. Examples: `/var/lib/pacman`, `/var/lib/apt`, `/var/lib/man-db`

- **/var/opt**

  Variable data of the packages in /opt must be installed in /var/opt/<subdir>, where <subdir> is the name of the subtree in /opt where the static data

- **/var/spool**
- data which is awaiting some kind of later processing, e.g.
  - `lpd` - printer spool dir
  - `mqueue` - outgoing mail queue
- **/var/tmp**

  The /var/tmp directory is made available for programs that require temporary files or directories that are preserved between system reboots. Therefore, data stored in /var/tmp is more persistent than data in /tmp.

Links, references etc

- `/usr/local` vs `/opt`
  - Linux Journal: Point/Counterpoint - /opt vs. /usr/local[5]
  - Stackexchange: What is the difference between /opt and /usr/local?[6]

[5] https://www.linuxjournal.com/magazine/pointcounterpoint-opt-vs-usrlocal
[6] https://unix.stackexchange.com/questions/11544/what-is-the-difference-between-opt-and-usr-local