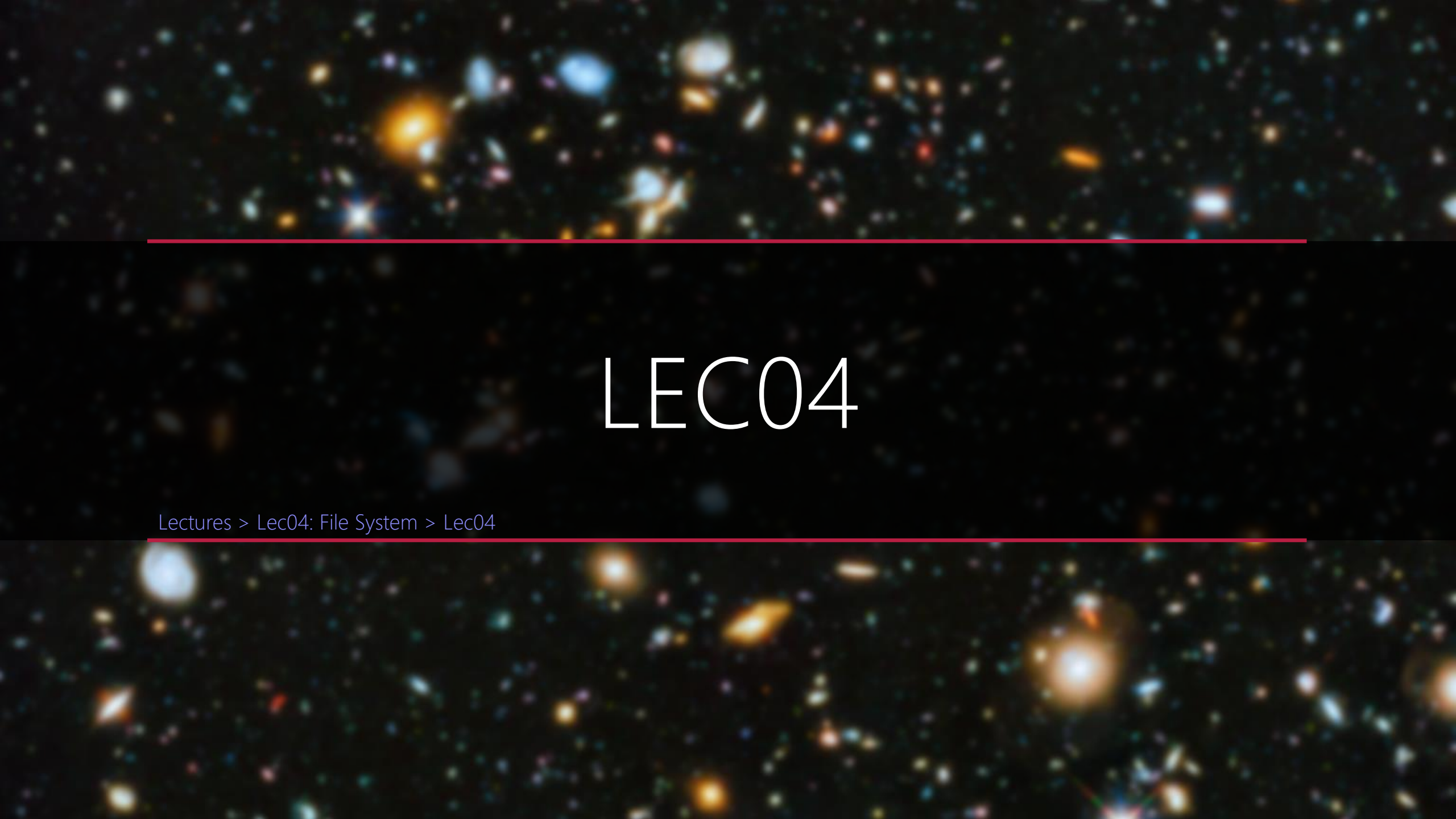
LAB04

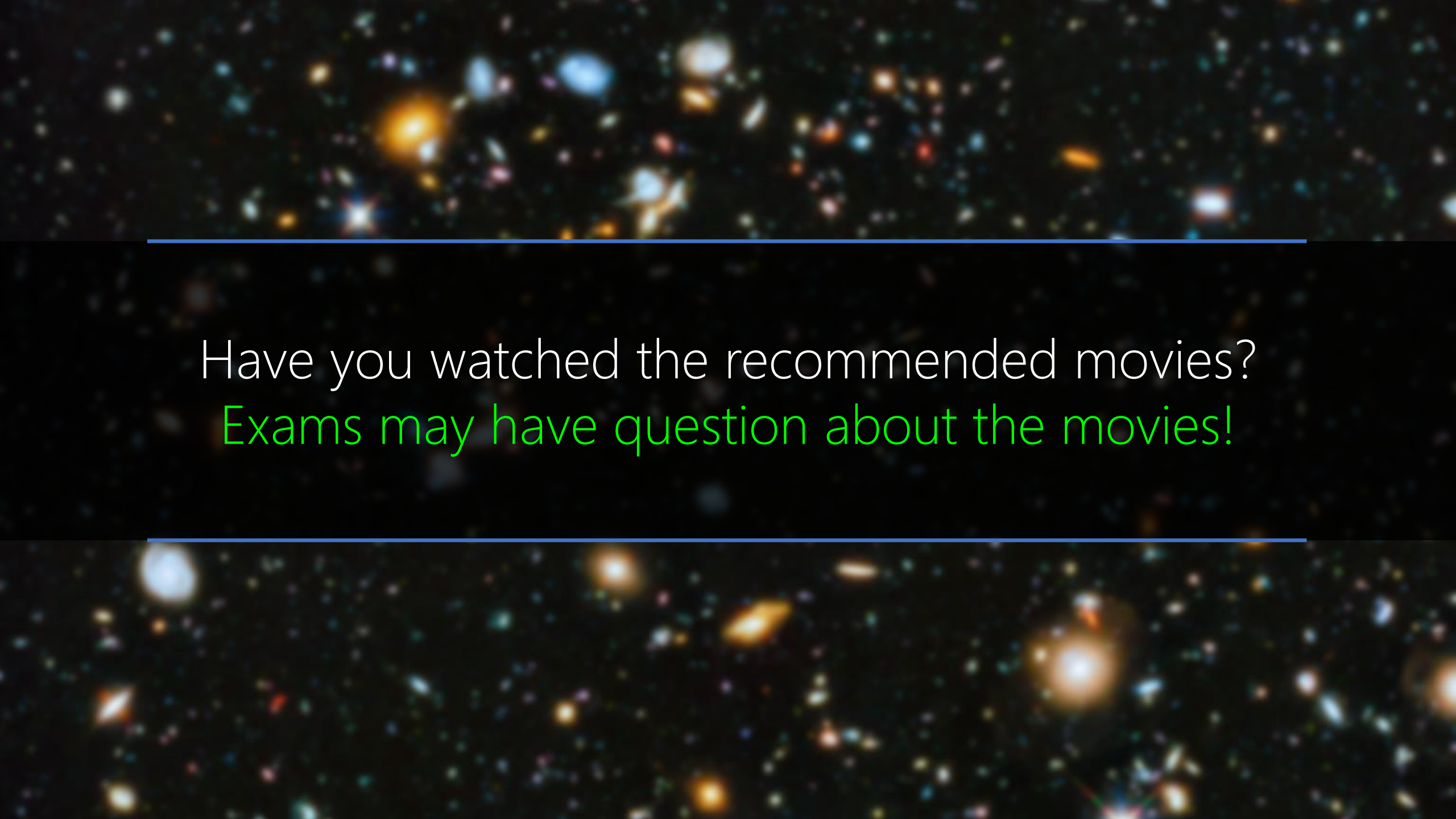
Labs > Lab04: Bash Script for Test Cases > Lab04

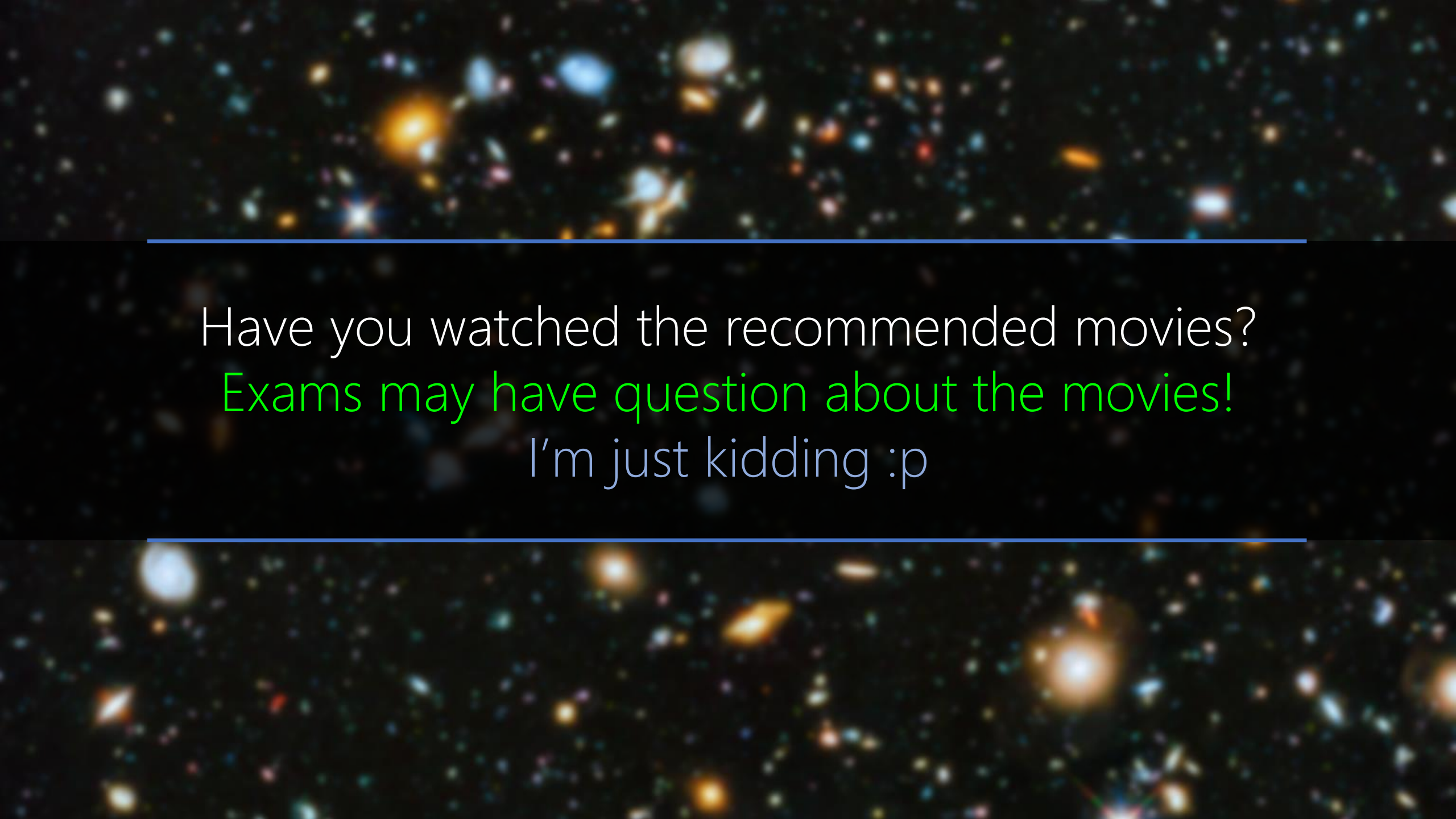


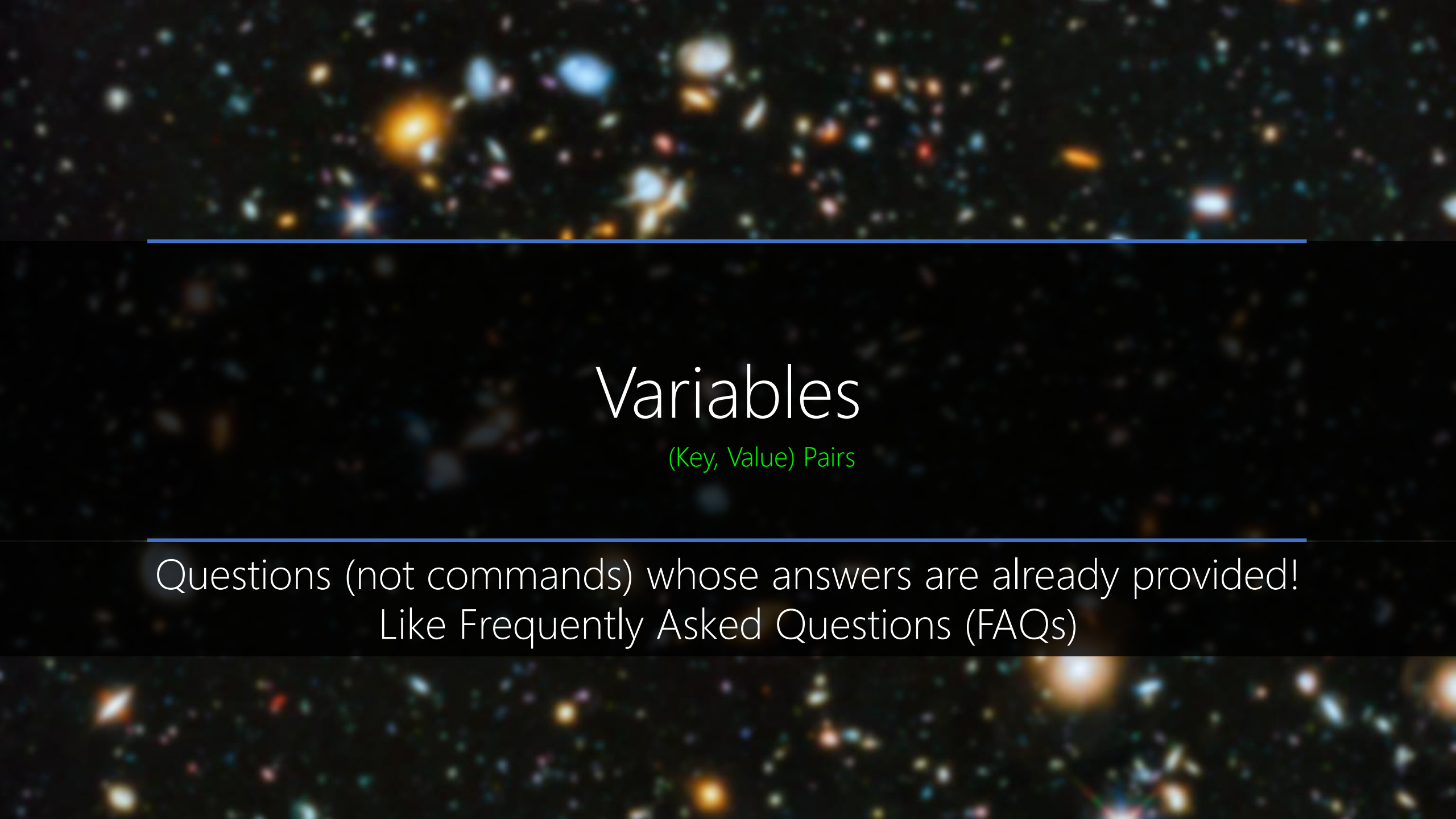
LEC04

Lectures > Lec04: File System > Lec04


MATT DAMON
THE
BOURNE ULTIMATUM


Have you watched the recommended movies?
Exams may have question about the movies!


Have you watched the recommended movies?
Exams may have question about the movies!
I'm just kidding :p



Variables

(Key, Value) Pairs

Questions (not commands) whose answers are already provided!
Like Frequently Asked Questions (FAQs)

System Variables

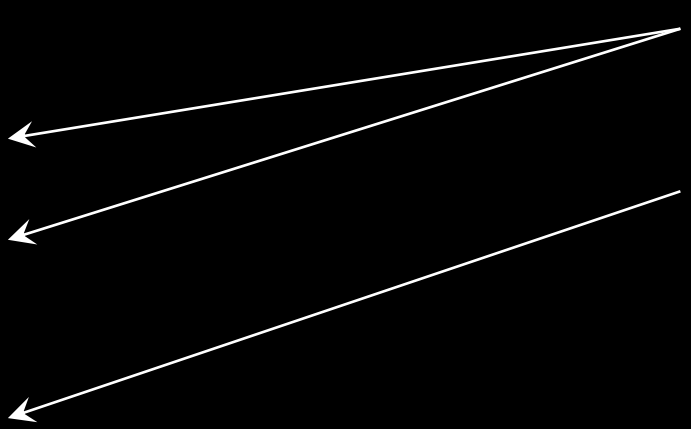
aka. Environment Variables, Global Variables, Unix Variables

Hossein: Kernel Variables

By convention, keys are UPPERCASE

To see the value, `echo ${KEY}`

```
hfani@alpha:~$ echo $OSTYPE
linux-gnu
hfani@alpha:~$ echo $USER
hfani
hfani@alpha:~$ echo $LOGNAME
hfani
hfani@alpha:~$ echo $HOME
/home/hfani
hfani@alpha:~$ echo $HOST

hfani@alpha:~$ echo $DISPLAY

hfani@alpha:~$ echo $EDITOR

hfani@alpha:~$ echo $SHELL
/bin/bash
hfani@alpha:~$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games:/opt/maple2021/bin:/opt/netlogo:/
ome/hfani/.dotnet/tools
hfani@alpha:~$
```



Not Set! Unset.

An important one! Very important actually.

PATH

Colon(:)-delimited list of directories

Tells the shell where to look/search when you request a particular program

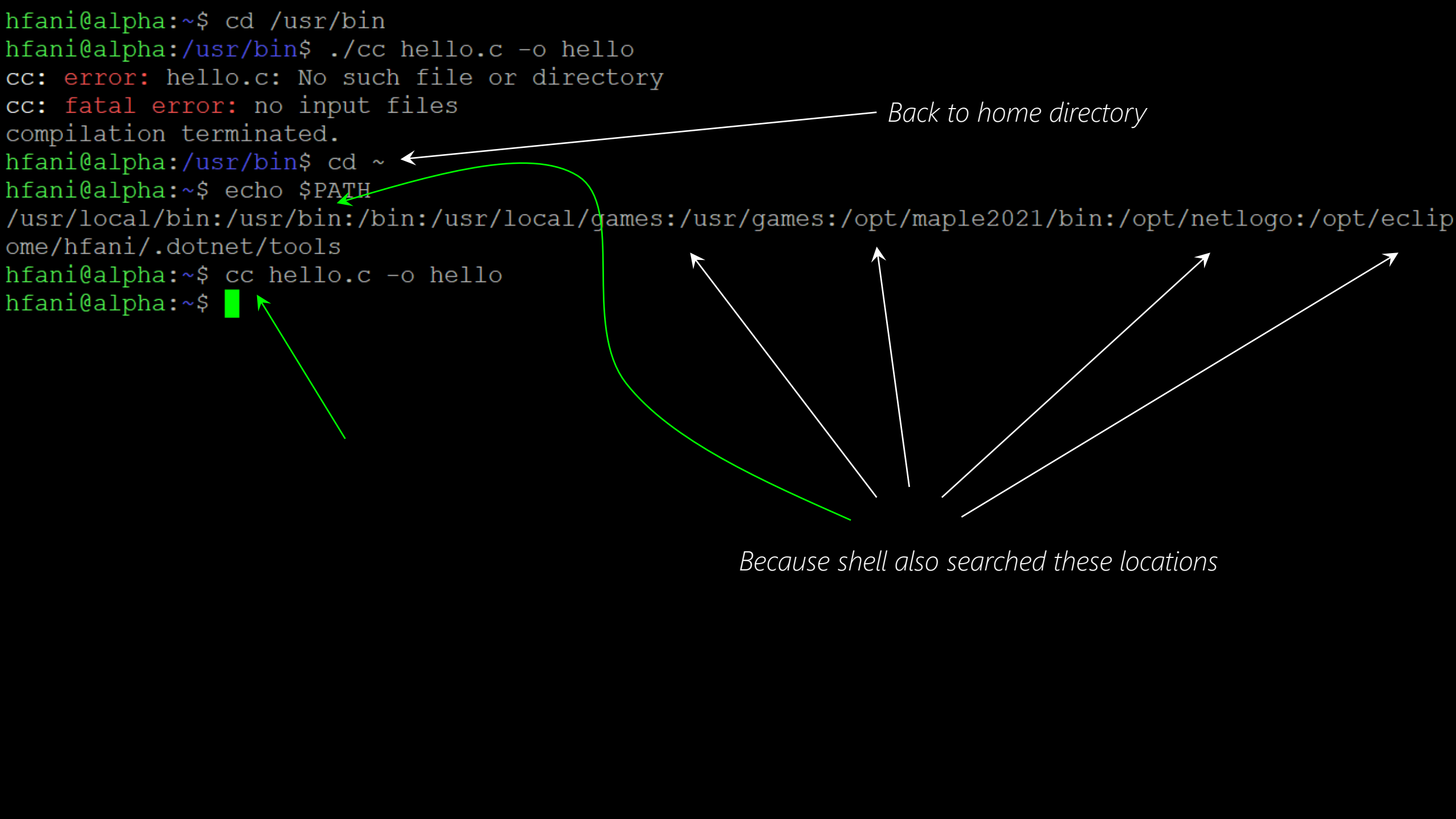
```
hfani@alpha:~$ cd /usr/bin
hfani@alpha:/usr/bin$ ./cc hello.c -o hello
cc: error: hello.c: No such file or directory
cc: fatal error: no input files
compilation terminated.
hfani@alpha:/usr/bin$
```

The actual location of program file for C compiler

Either you have to copy your files to /usr/bin
Or copy cc to your directory

Both are impossible due to lack of administrative privileges

```
hfani@alpha:~$ cd /usr/bin
hfani@alpha:/usr/bin$ ./cc hello.c -o hello
cc: error: hello.c: No such file or directory
cc: fatal error: no input files
compilation terminated.
hfani@alpha:/usr/bin$ cd ~
hfani@alpha:~$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games:/opt/maple2021/bin:/opt/netlogo:/opt/eclip
ome/hfani/.dotnet/tools
hfani@alpha:~$ cc hello.c -o hello
hfani@alpha:~$
```

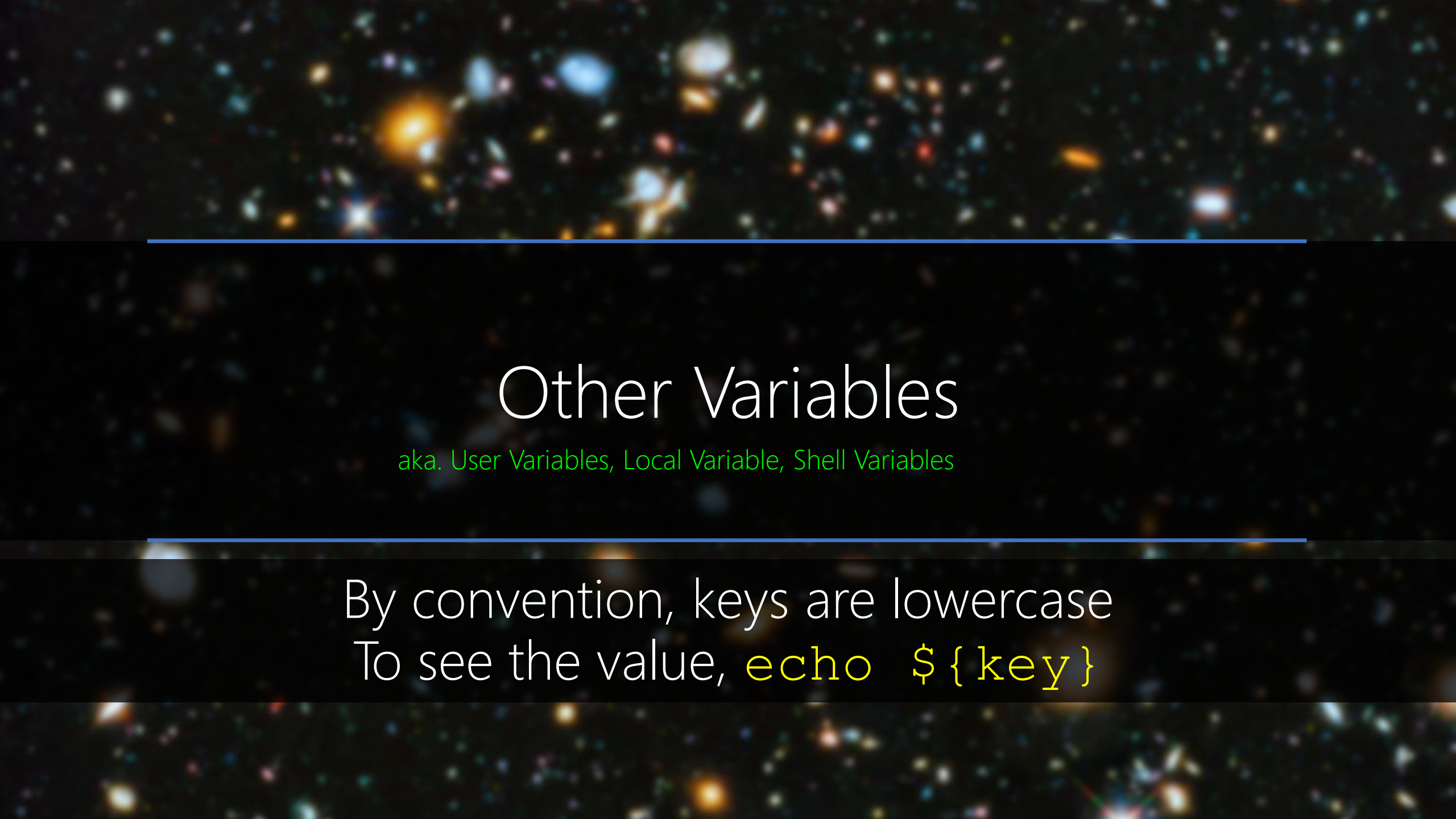


System Variables

aka. Environment Variables, Global Variables, Unix Variables

Hossein: Kernel Variables

Is it able to modify the KEY's value? Yes.
Is it able to unset the KEY's value? Yes.
Is it able to add a new KEY=Value pair? Yes.
Is it able to persist the change? Yes.
How? It depends on the shell ☹



Other Variables

aka. User Variables, Local Variable, Shell Variables

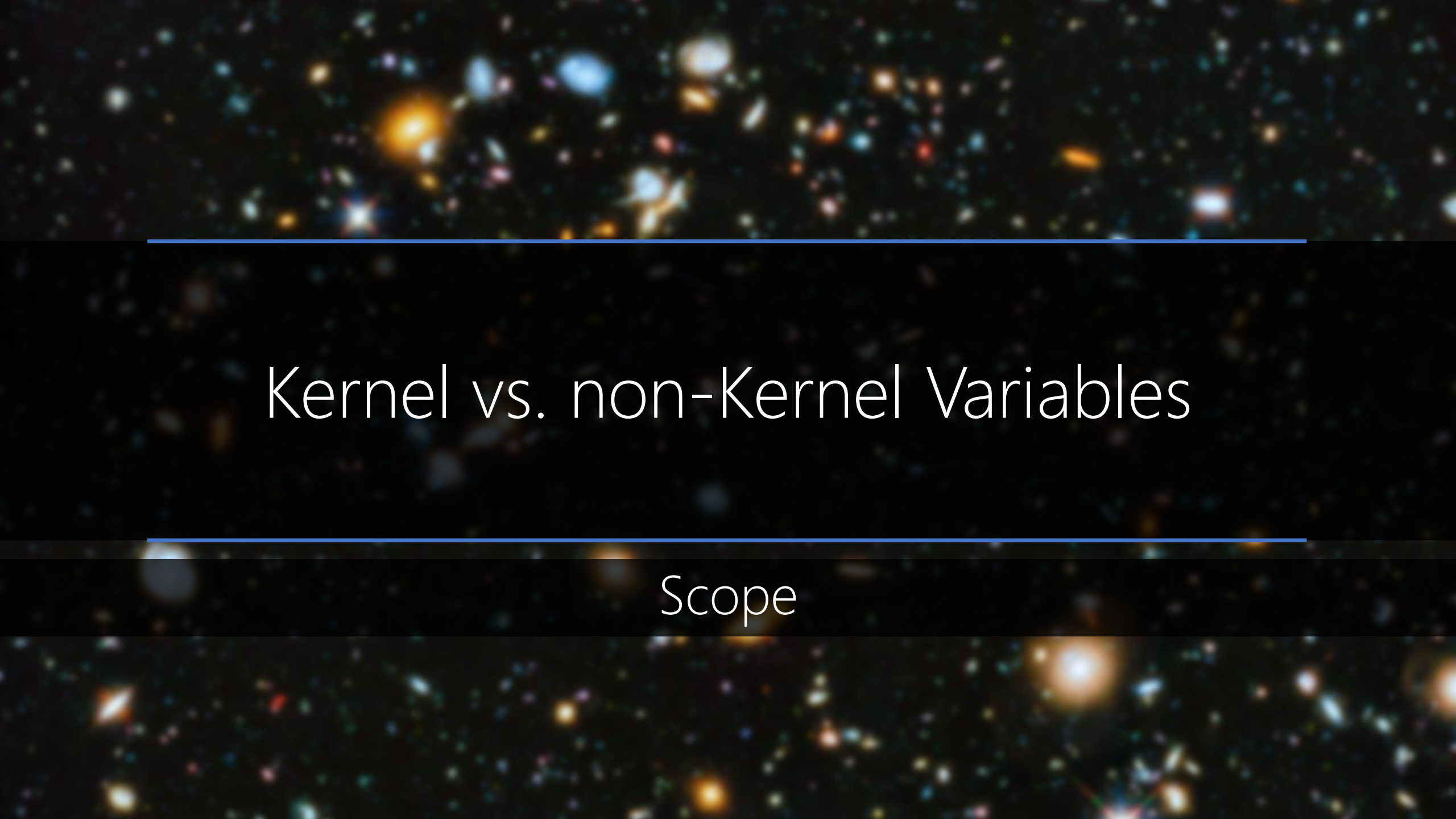
By convention, keys are lowercase

To see the value, `echo ${key}`

Other Variables

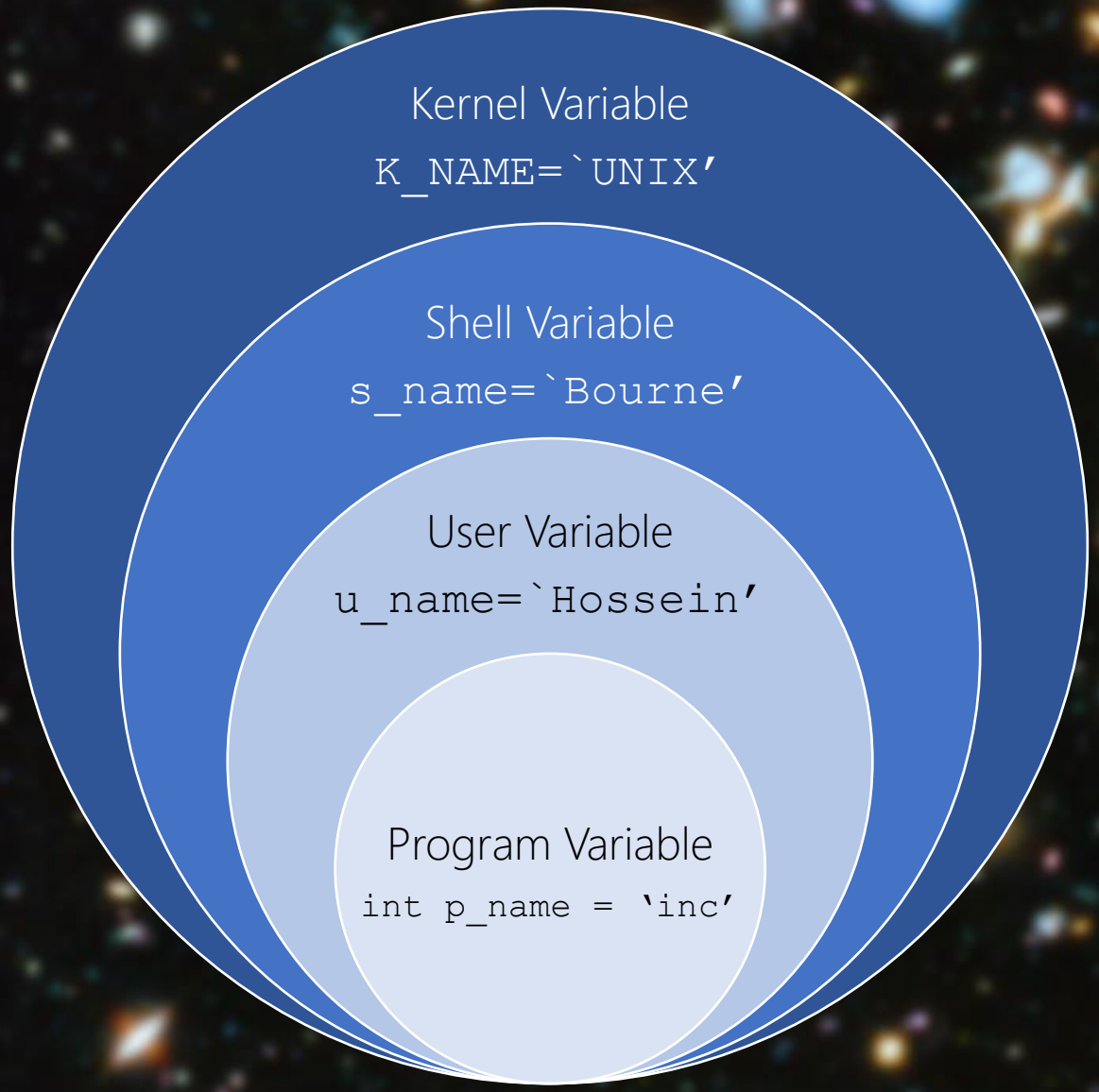
aka. User Variables, Local Variable, Shell Variables

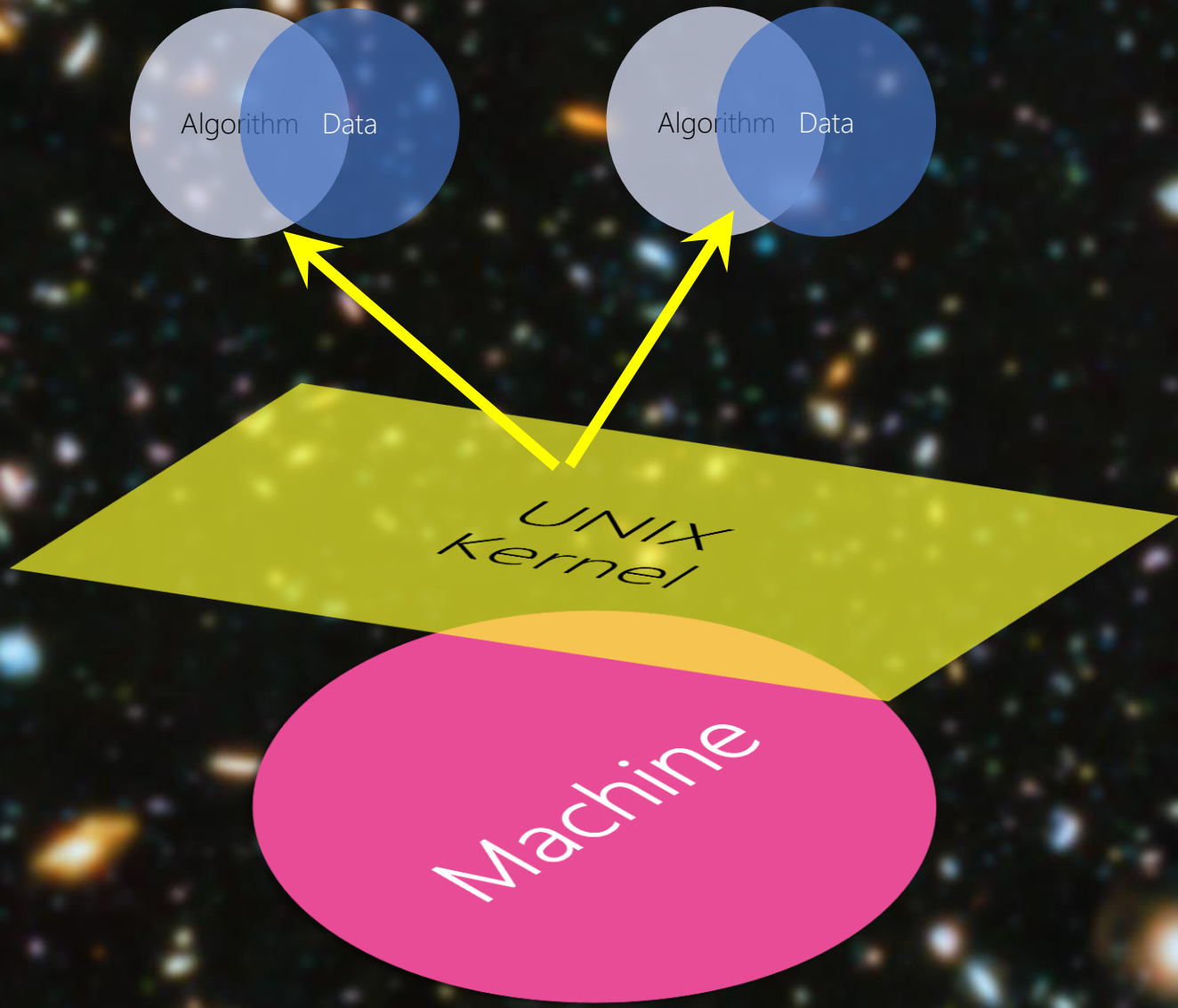
Is it able to modify the key's value? Yes.
Is it able to unset the key's value? Yes.
Is it able to add a new key=value pair? Yes.
Is it able to persist the change? Yes.
How? It depends on the shell ☹

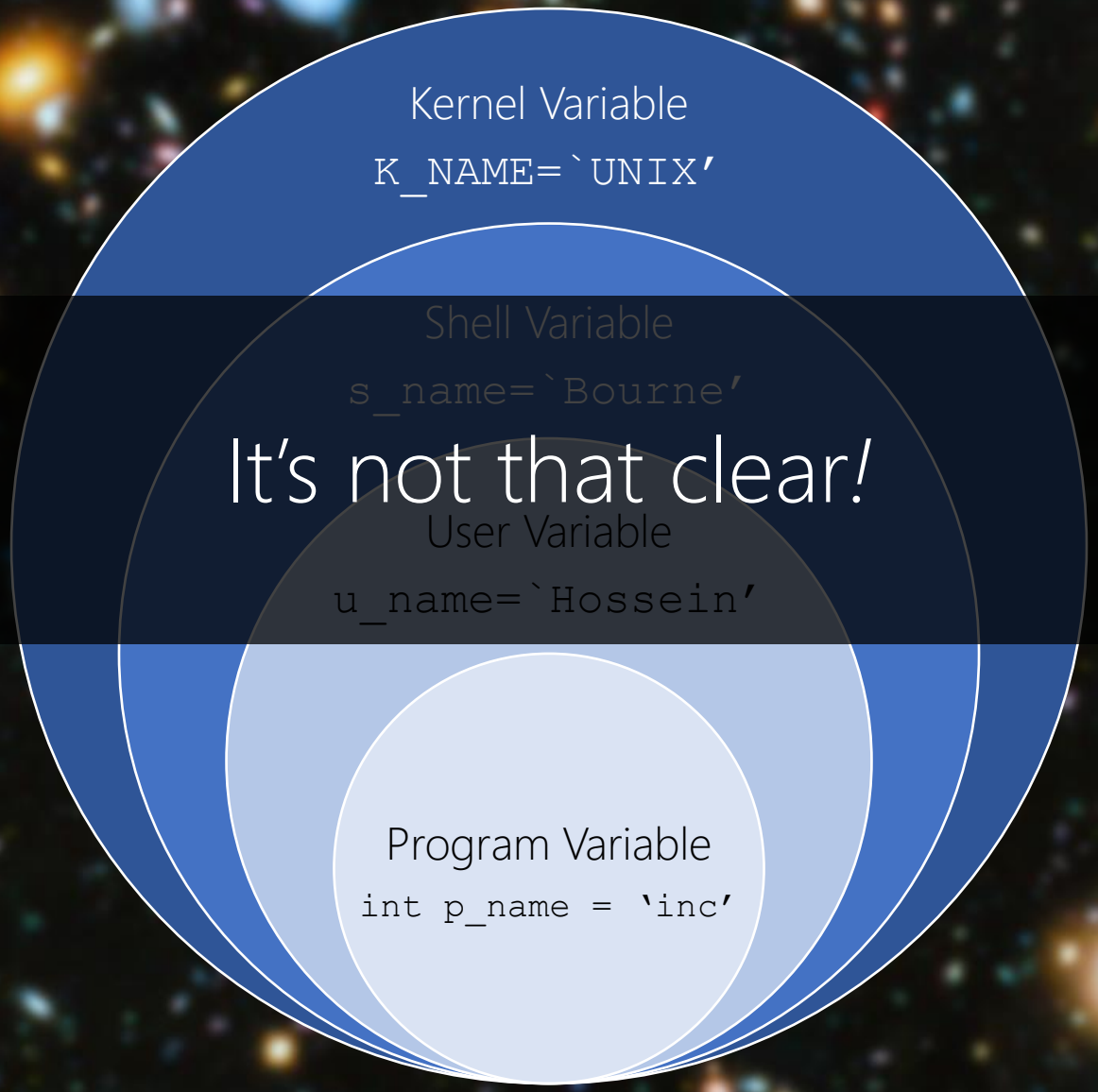


Kernel vs. non-Kernel Variables

Scope


Kernel Variable
K_NAME=`UNIX'
Shell Variable
s_name=`Bourne'
User Variable
u_name=`Hossein'
Program Variable
int p_name = 'inc'

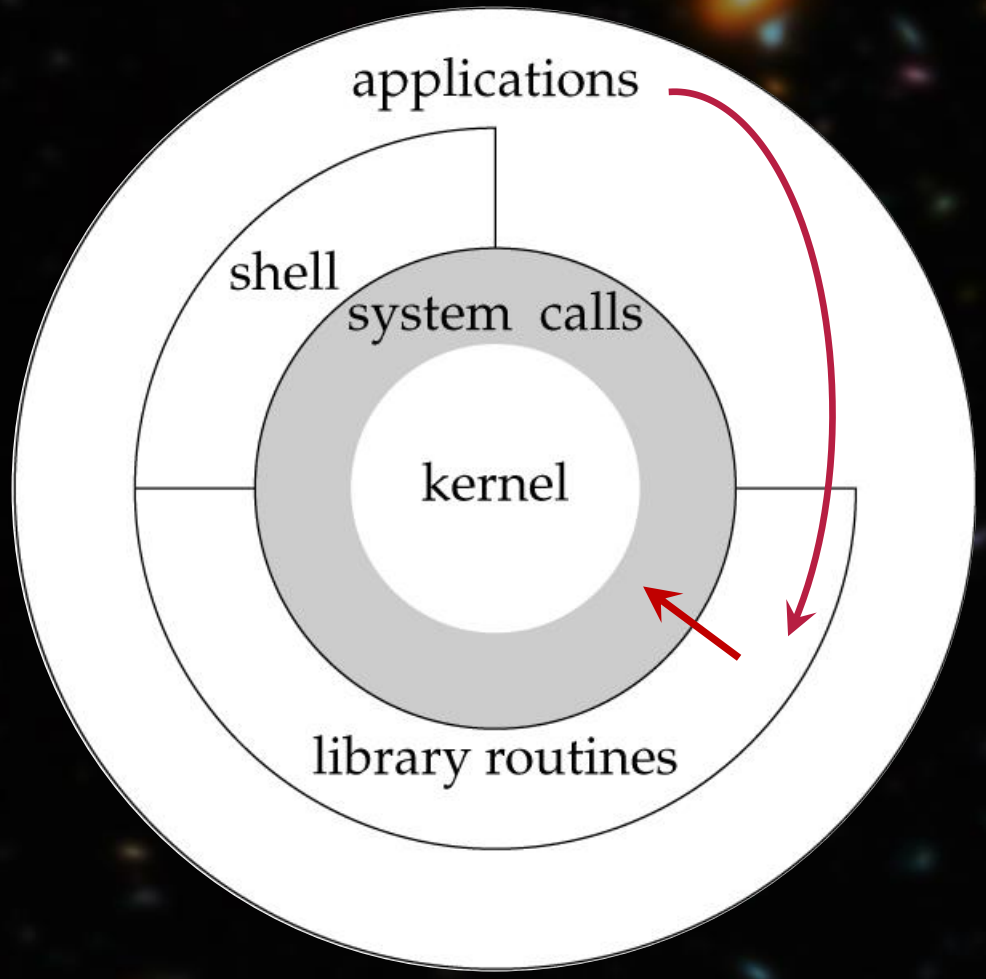

Algorithm
Data
Algorithm
Data
UNIX
kernel
Machine


Kernel Variable
K_NAME=`UNIX'
Shell Variable
s_name=`Bourne'
It's not that clear!
User Variable
u_name=`Hossein'
Program Variable
int p_name = 'inc'

Access Kernel Variables by Call to Library Routine

Lab02

```
#include <stdlib.h>;
char *getenv(const char *KEY)
```



Header	FreeBSD 8.0	Linux 3.2.0	Mac OS X 10.6.8	Solaris 10	Description
<assert.h>	•	•	•	•	verify program assertion
<complex.h>	•	•	•	•	complex arithmetic support
<ctype.h>	•	•	•	•	character classification and mapping support
<errno.h>	•	•	•	•	error codes (Section 1.7)
<fenv.h>	•	•	•	•	floating-point environment
<float.h>	•	•	•	•	floating-point constants and characteristics
<inttypes.h>	•	•	•	•	integer type format conversion
<iso646.h>	•	•	•	•	macros for assignment, relational, and unary operators
<limits.h>	•	•	•	•	implementation constants (Section 2.5)
<locale.h>	•	•	•	•	locale categories and related definitions
<math.h>	•	•	•	•	mathematical function and type declarations and constants
<setjmp.h>	•	•	•	•	nonlocal goto (Section 7.10)
<signal.h>	•	•	•	•	signals (Chapter 10)
<stdarg.h>	•	•	•	•	variable argument lists
<stdbool.h>	•	•	•	•	Boolean type and values
<stddef.h>	•	•	•	•	standard definitions
<stdint.h>	•	•	•	•	integer types
<stdio.h>	•	•	•	•	standard I/O library (Chapter 5)
<stdlib.h>	•	•	•	•	utility functions
<string.h>	•	•	•	•	string operations
<tgmath.h>	•	•	•	•	type-generic math macros
<time.h>	•	•	•	•	time and date (Section 6.10)
<wchar.h>	•	•	•	•	extended multibyte and wide character support
<wctype.h>	•	•	•	•	wide character classification and mapping support

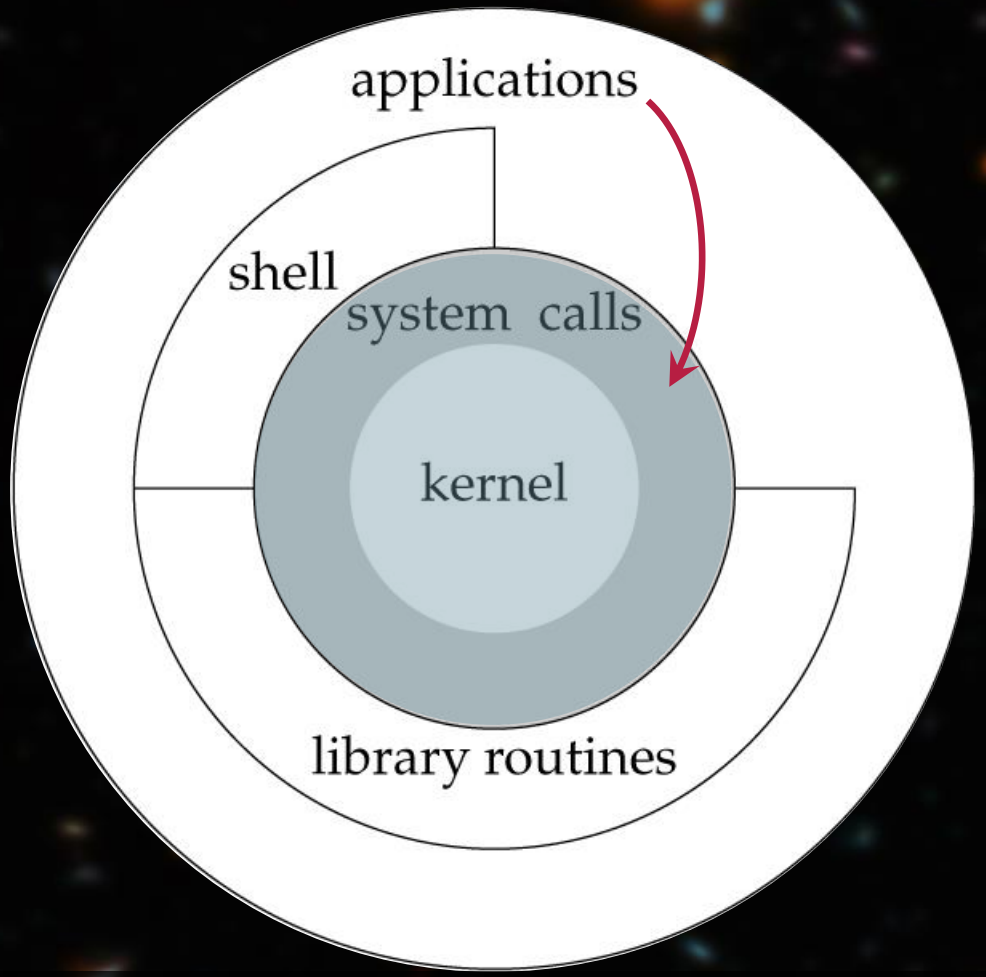
```c
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
int main()
{
  printf("Hello World! This is hfani@uwindsor.ca, StudentID: 123456789\n");
  time_t t = time(NULL);
  struct tm tm = *localtime(&t);
  printf("now: %d-%02d-%02d %02d:%02d:%02d\n", tm.tm_year + 1900, tm.tm_mon + 1, tm.tm_m
  printf("%s@shell:%s$\n", getenv("USER"),getenv("PWD"));

}
```

The library routine that does system call to Kernel
Your program statically linked this library!

Getting the values of USER and PWD (path of working directory)

Access Kernel Variables by System Call

```
#include <unistd.h>;
extern char **environ
```

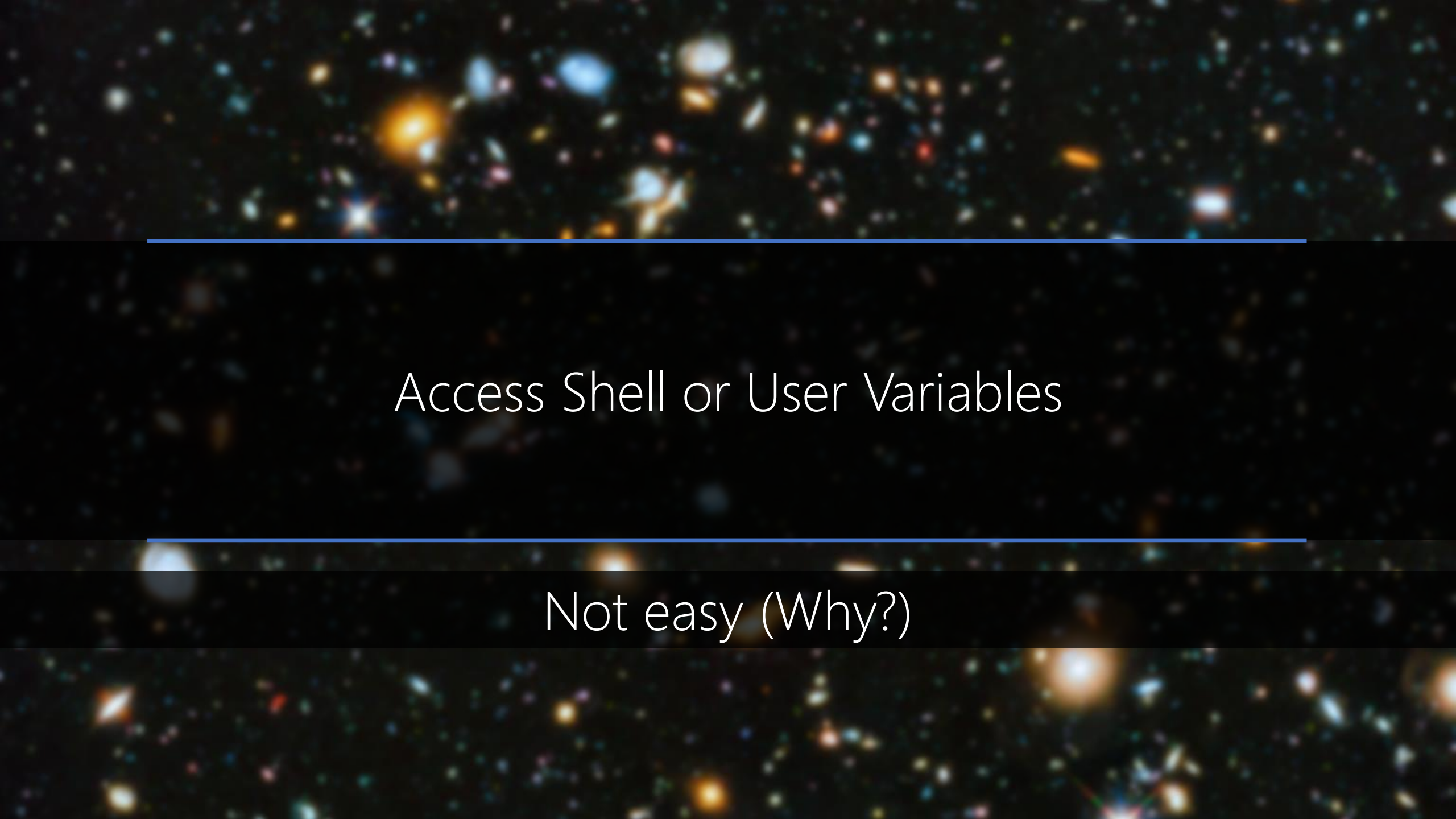


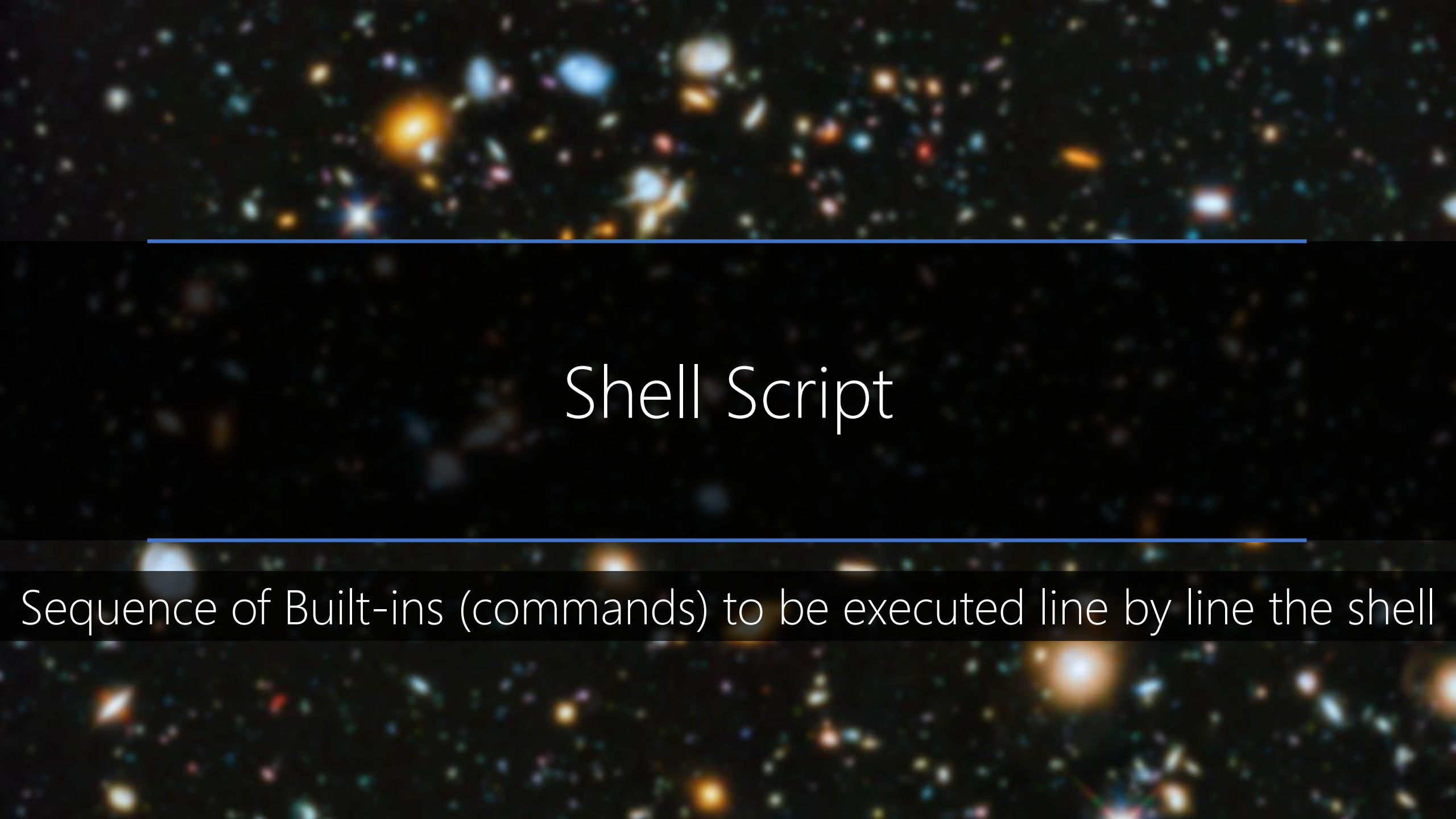
Header	FreeBSD 8.0	Linux 3.2.0	Mac OS X 10.6.8	Solaris 10	Description
<aio.h>	•	•	•	•	asynchronous I/O
<cpio.h>	•	•	•	•	cpio archive values
<dirent.h>	•	•	•	•	directory entries (Section 4.22)
<dlfcn.h>	•	•	•	•	dynamic linking
<fcntl.h>	•	•	•	•	file control (Section 3.14)
<fnmatch.h>	•	•	•	•	filename-matching types
<glob.h>	•	•	•	•	pathname pattern-matching and generation
<grp.h>	•	•	•	•	group file (Section 6.4)
<iconv.h>	•	•	•	•	codeset conversion utility
<langinfo.h>	•	•	•	•	language information constants
<monetary.h>	•	•	•	•	monetary types and functions
<netdb.h>	•	•	•	•	network database operations
<nl_types.h>	•	•	•	•	message catalogs
<poll.h>	•	•	•	•	poll function (Section 14.4.2)
<pthread.h>	•	•	•	•	threads (Chapters 11 and 12)
<pwd.h>	•	•	•	•	password file (Section 6.2)
<regex.h>	•	•	•	•	regular expressions
<sched.h>	•	•	•	•	execution scheduling
<semaphore.h>	•	•	•	•	semaphores
<strings.h>	•	•	•	•	string operations
<tar.h>	•	•	•	•	tar archive values
<termios.h>	•	•	•	•	terminal I/O (Chapter 18)
<unistd.h>	•	•	•	•	symbolic constants
<wordexp.h>	•	•	•	•	word-expansion definitions
<arpa/inet.h>	•	•	•	•	Internet definitions (Chapter 16)
<net/if.h>	•	•	•	•	socket local interfaces (Chapter 16)
<netinet/in.h>	•	•	•	•	Internet address family (Section 16.3)
<netinet/tcp.h>	•	•	•	•	Transmission Control Protocol definitions
<sys/mman.h>	•	•	•	•	memory management declarations
<sys/select.h>	•	•	•	•	select function (Section 14.4.1)
<sys/socket.h>	•	•	•	•	sockets interface (Chapter 16)
<sys/stat.h>	•	•	•	•	file status (Chapter 4)
<sys/statvfs.h>	•	•	•	•	file system information
<sys/times.h>	•	•	•	•	process times (Section 8.17)
<sys/types.h>	•	•	•	•	primitive system data types (Section 2.8)
<sys/un.h>	•	•	•	•	UNIX domain socket definitions (Section 17.2)
<sys/utsname.h>	•	•	•	•	system name (Section 6.9)
<sys/wait.h>	•	•	•	•	process control (Section 8.6)

```c
#include <stdio.h>
#include <unistd.h>

extern char **environ;
int main(int argc, char *argv[])
{
    int index = 0;
    printf("Environment variables:\n");
    index = 0;
    while (environ[index])
    {
        printf("envp[%d]: %s\n", index, environ[index]);
        ++index;
    }
    return 0;
}
```

*System call to Kernel! (**uni**x **st**andar**d** library)*
Either statically or dynamically linked.

```
Environment variables:
envp[0]: SHELL=/bin/bash
envp[1]: LANGUAGE=en_CA:en
envp[2]: NO_AT_BRIDGE=1
envp[3]: TWO_TASK=cs01
envp[4]: PWD=/home/hfani
envp[5]: LOGNAME=hfani
envp[6]: XDG_SESSION_TYPE=tty
envp[7]: PRINTER=cs_commons
envp[8]: MOTD_SHOWN=pam
envp[9]: VIRTUALENVWRAPPER_SCRIPT=/usr/share/virtualenvwrapper/virtualenvwrapper.sh
envp[10]: HOME=/home/hfani
envp[11]: LANG=en_CA.UTF-8
envp[12]: LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00:su=37;41:
:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;
=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.l
*.lzo=01;31:*.xz=01;31:*.zst=01;31:*.tzst=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01
r=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:
;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35
1;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:
;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.webp=01;35:*.ogm=01;35:*.mp4=01;35:
;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.f
:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00
=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36
0;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
envp[13]: _VIRTUALENVWRAPPER_API= mkvirtualenv rmvirtualenv lsvirtualenv showvirtualenv workon add2virtualenv cdsitepackages cd
nv lssitepackages toggleglobalsitepackages cpvirtualenv setvirtualenvproject mkproject cdproject mktmpenv wipeenv allvirtualenv
alenv rmvirtualenv lsvirtualenv showvirtualenv workon add2virtualenv cdsitepackages cdvirtualenv lssitepackages toggleglobalsit
s cpvirtualenv setvirtualenvproject mkproject cdproject mktmpenv wipeenv allvirtualenv
envp[14]: ORACLE_HOME=/usr/lib/oracle/12.1/client64
envp[15]: SSH_CONNECTION=137.207.140.134 63217 137.207.82.51 22
envp[16]: WINEDLLOVERRIDES=winemenubuilder.exe=d
envp[17]: LESSCLOSE=/usr/bin/lesspipe %s %s
envp[18]: XDG_SESSION_CLASS=user
envp[19]: TERM=xterm
envp[20]: LESSOPEN=| /usr/bin/lesspipe %s
envp[21]: USER=hfani
```


Access Shell or User Variables
Not easy (Why?)



Shell Script

Sequence of Built-ins (commands) to be executed line by line the shell

```
hfani@alpha:~$ vi myscript.sh
```

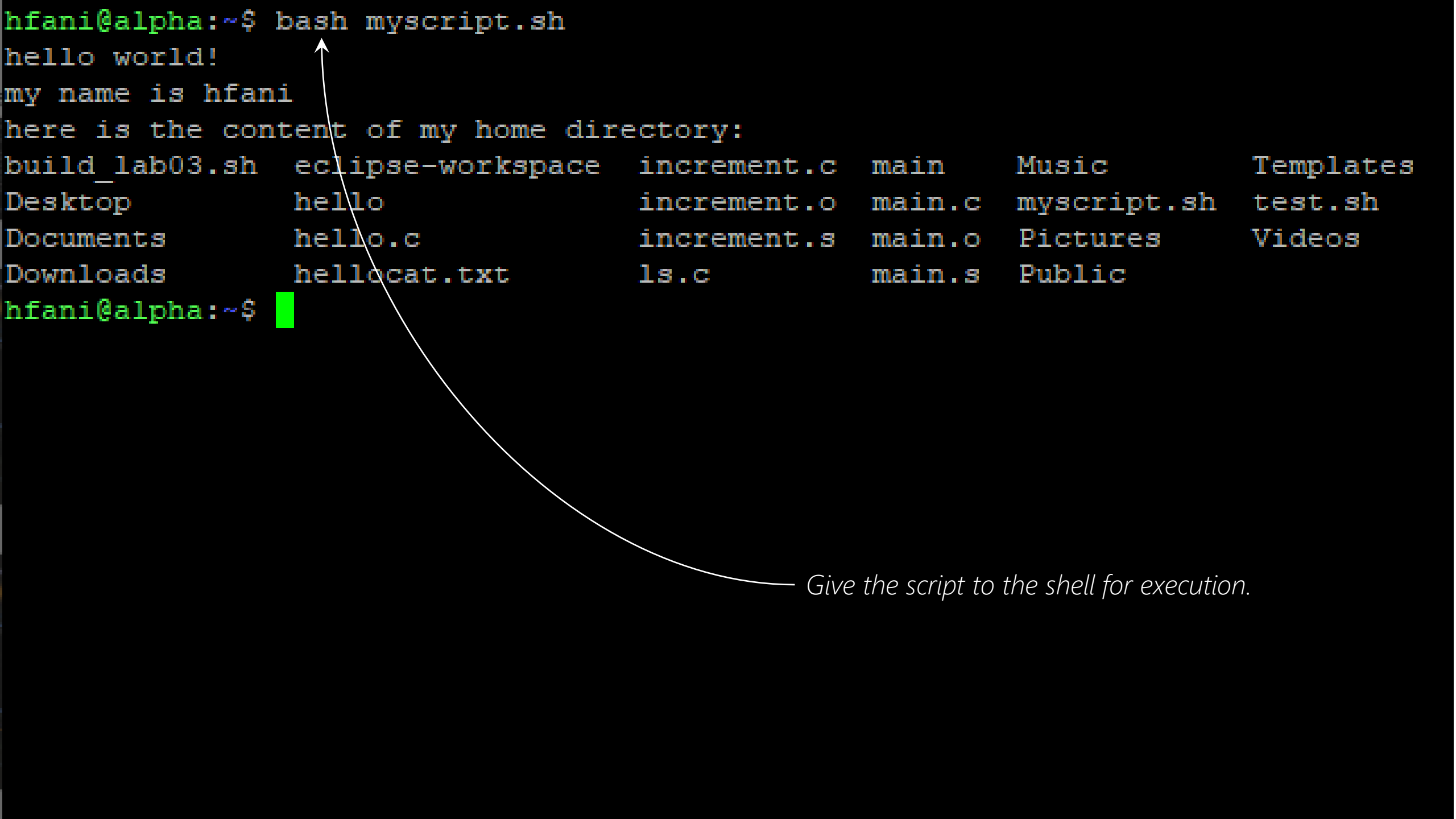
By convention, the extension is .sh

```
#!/bin/bash
```

Important: under what shell?

```
echo "hello world!"
echo "my name is $USER"
echo "here is the content of my home directory:"
ls ~/
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

```
hfani@alpha:~$ bash myscript.sh
hello world!
my name is hfani
here is the content of my home directory:
build_lab03.sh  eclipse-workspace  increment.c  main    Music       Templates
Desktop         hello              increment.o  main.c  myscript.sh  test.sh
Documents       hello.c            increment.s  main.o  Pictures    Videos
Downloads       hellocat.txt       ls.c         main.s  Public
hfani@alpha:~$
```



```
hfani@alpha:~$ chmod +x myscript.sh
hfani@alpha:~$ ./myscript.sh
hello world!
my name is hfani
here is the content of my home directory:
Desktop    Downloads          hello    hellocat.txt  increment.o  ls.c    main.o
Documents  eclipse-workspace  hello.c  increment.c   increment.s  main.c  main.s
hfani@alpha:~$
```

Important: make it executable

Shell Script

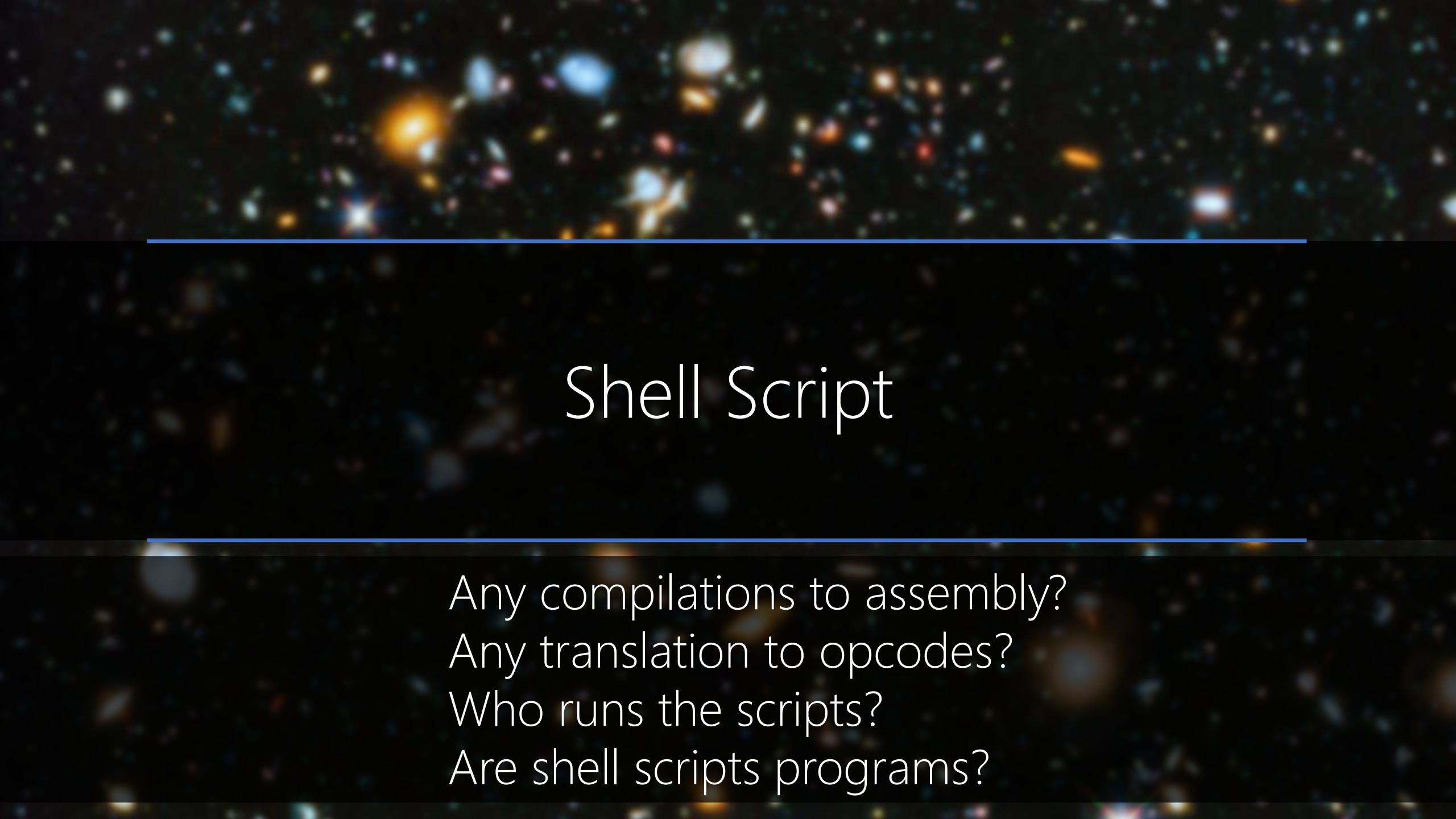
Lab03

```
hfani@alpha:~$ vi build_lab03.sh
```

```
#!/bin/bash
echo "start building lab03 program:"
echo "compiling to assembly lines ..."
cc main.c -S
cc increment.c -S
echo "translating to opcodes ..."
cc main.s -c
cc increment.s -c
echo "statically linking all required opcodes ..."
cc main.o increment.o -o main
echo "build successfully done!"
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

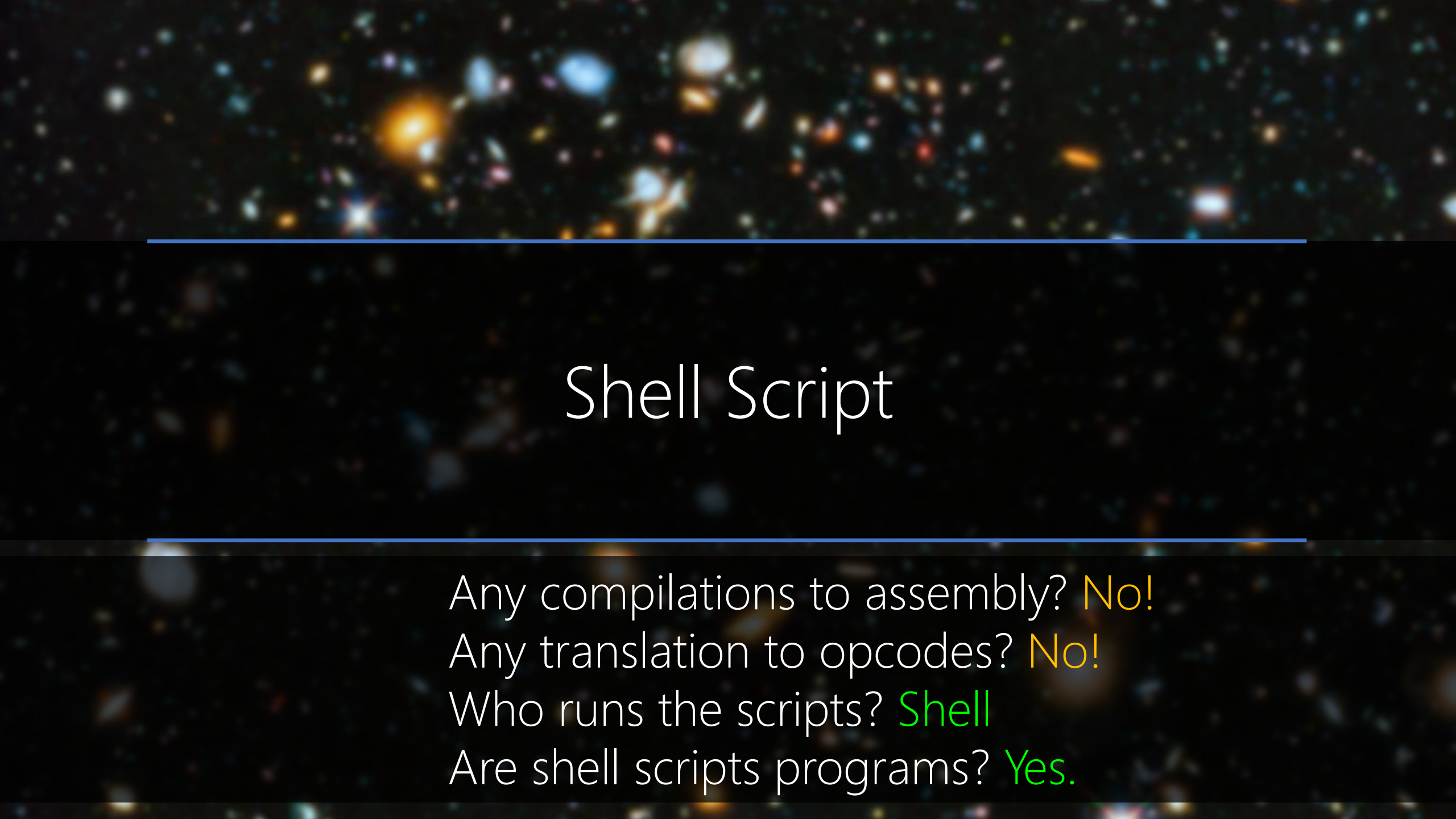
```
hfani@alpha:~$ chmod +x build_lab03.sh
hfani@alpha:~$ ./build_lab03.sh
start building lab03 program:
compiling to assembly lines ...
main.c: In function ‘main’:
main.c:5:6: warning: implicit declaration of function ‘increment’ [-Wimplicit-function-de
    5 |  a = increment(a);
      |      ^~~~~~~~~
translating to opcodes ...
statically linking all required opcodes ...
build successfully done!
hfani@alpha:~$
```

Important: make it executable



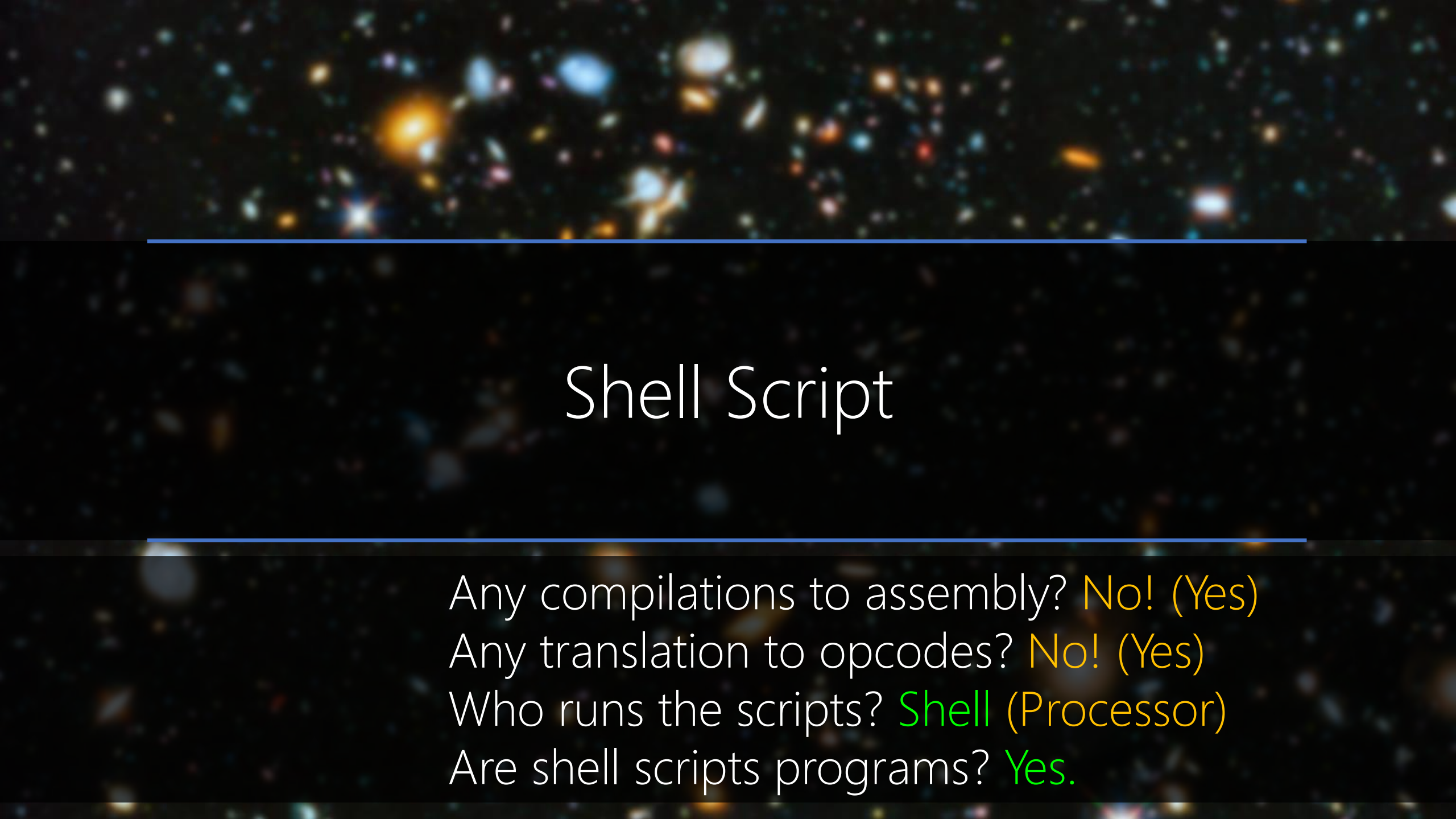
Shell Script

Any compilations to assembly?
Any translation to opcodes?
Who runs the scripts?
Are shell scripts programs?



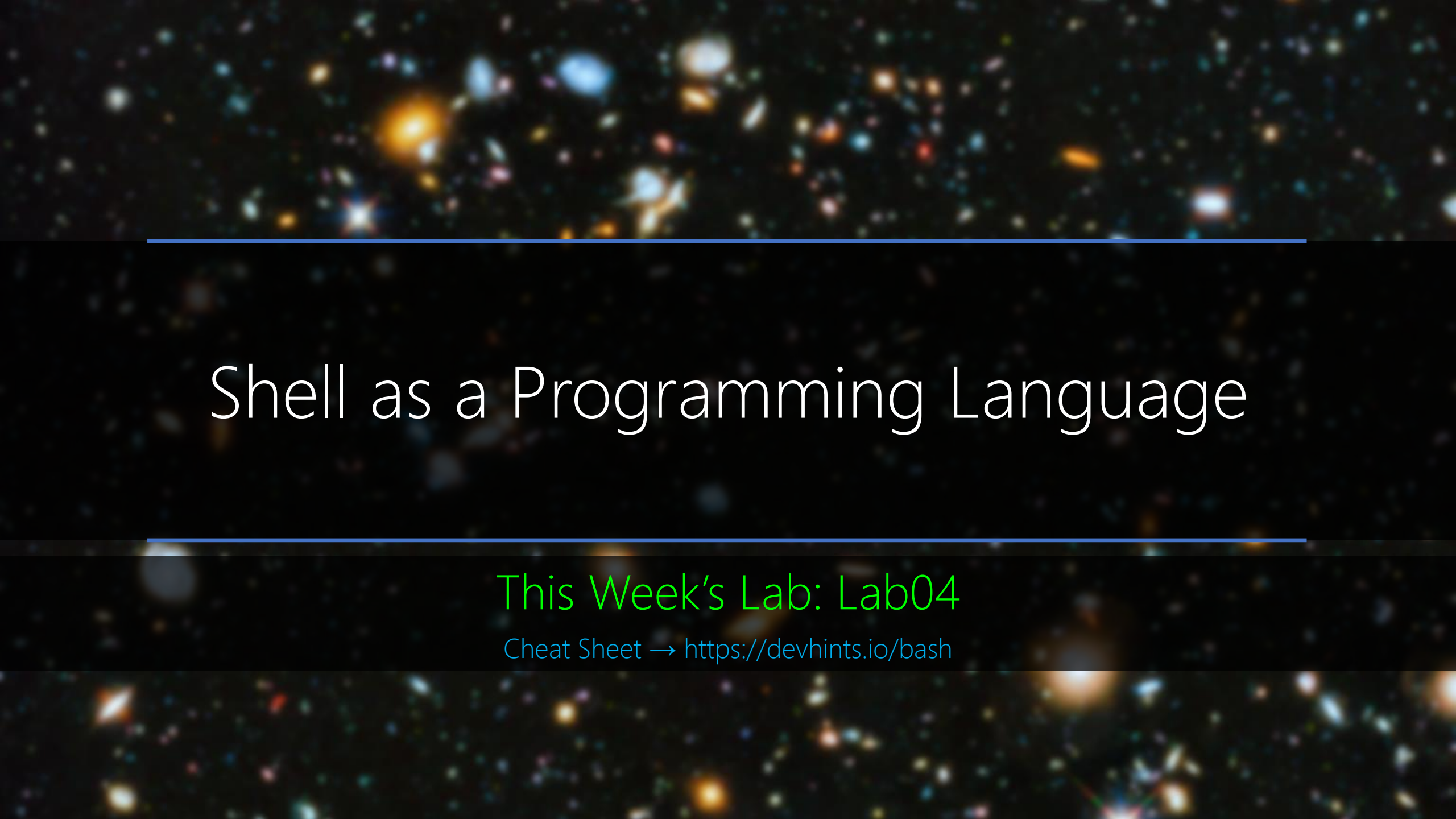
Shell Script

Any compilations to assembly? No!
Any translation to opcodes? No!
Who runs the scripts? Shell
Are shell scripts programs? Yes.



Shell Script

Any compilations to assembly? No! (Yes)
Any translation to opcodes? No! (Yes)
Who runs the scripts? Shell (Processor)
Are shell scripts programs? Yes.

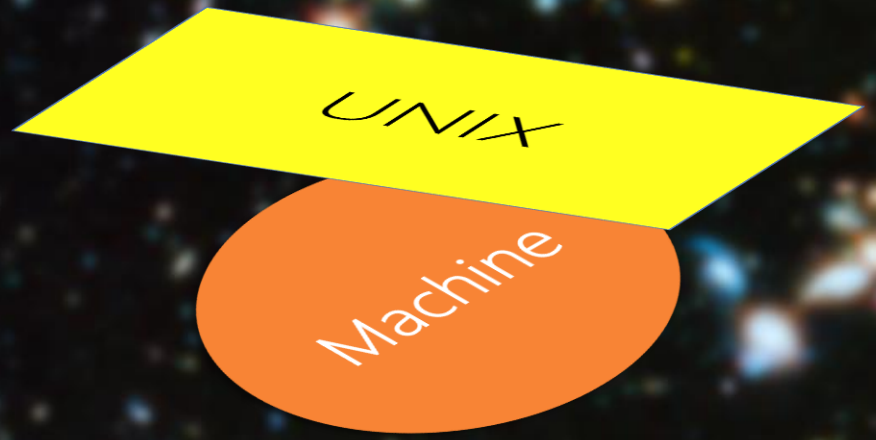


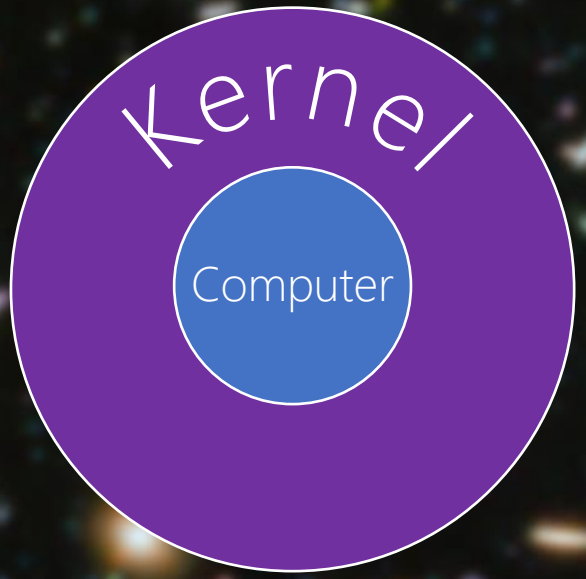
Shell as a Programming Language

This Week's Lab: Lab04

Cheat Sheet → https://devhints.io/bash


The Only Way In Or Out Is Through Phone Lines
- The Matrix (1999), Lana & Lilly Wachowski





Computer

Memory

Kernel: Device Manager

Kernel: Memory Manager

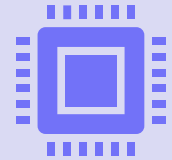
Kernel: File Manager

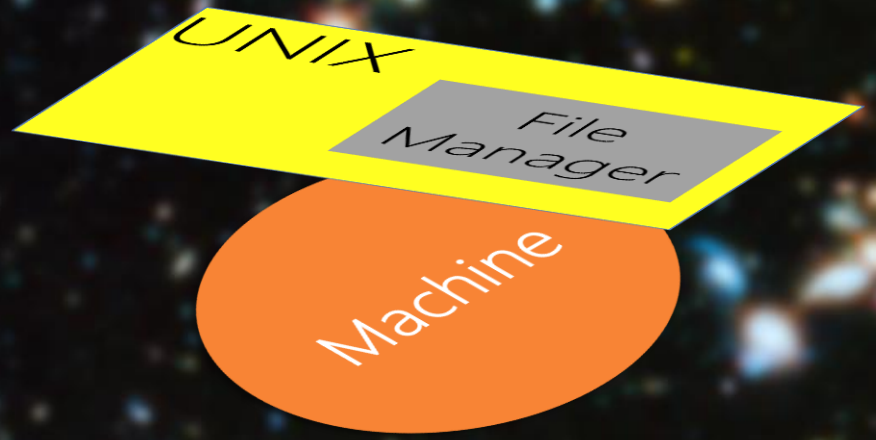
Kernel: Network Manager

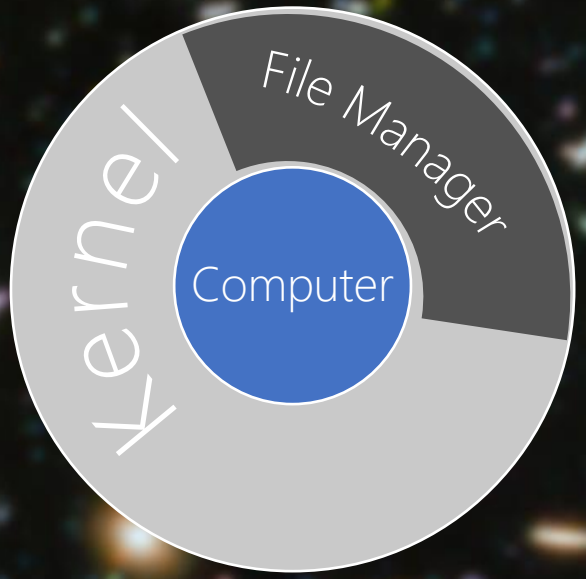
Kernel: Process Manager

Bus

Processor







Computer

Memory

Kernel: Device Manager

Kernel: Memory Manager

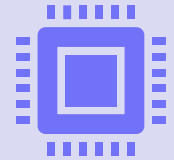
Kernel: File Manager

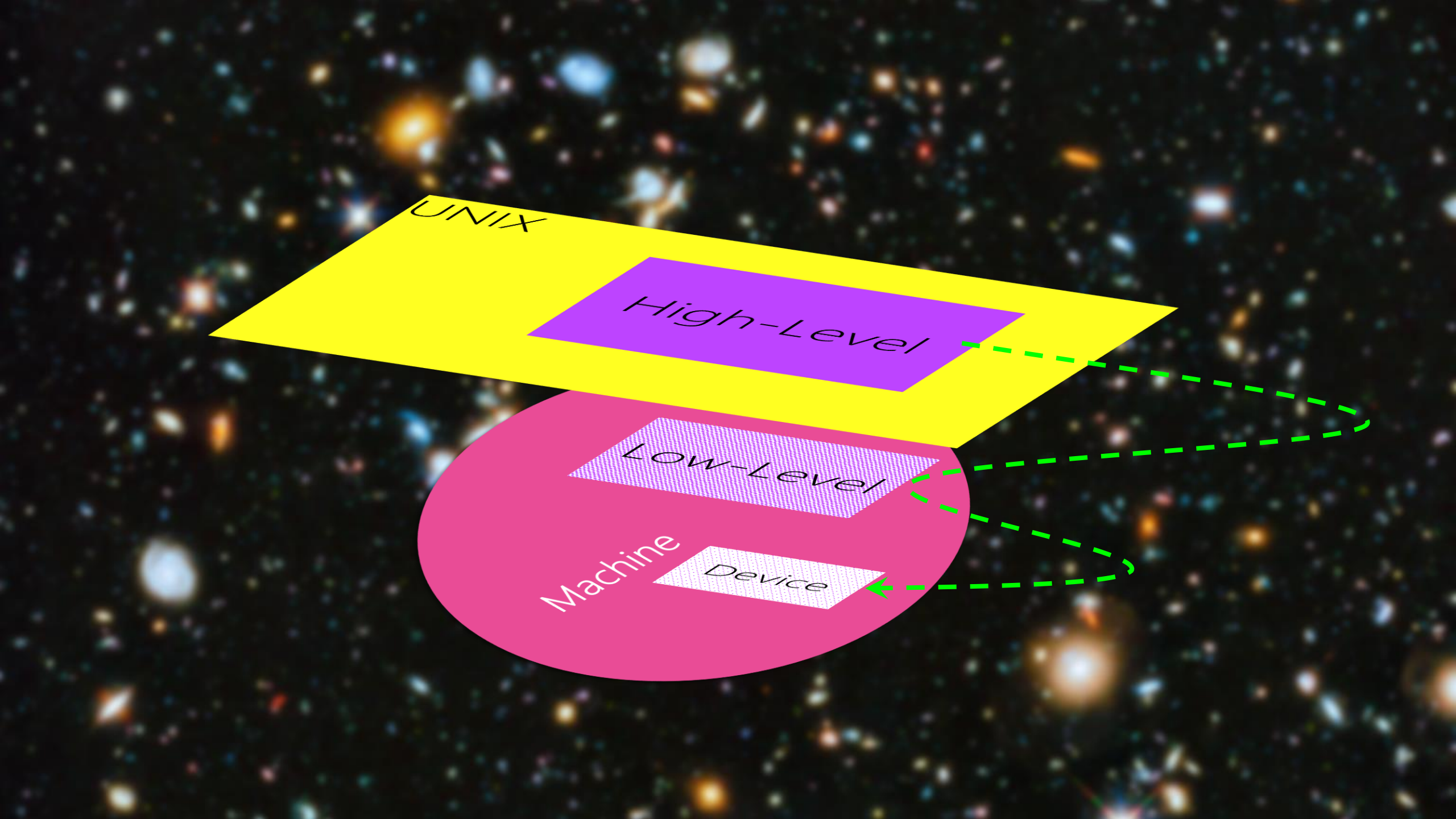
Kernel: Network Manager

Kernel: Process Manager

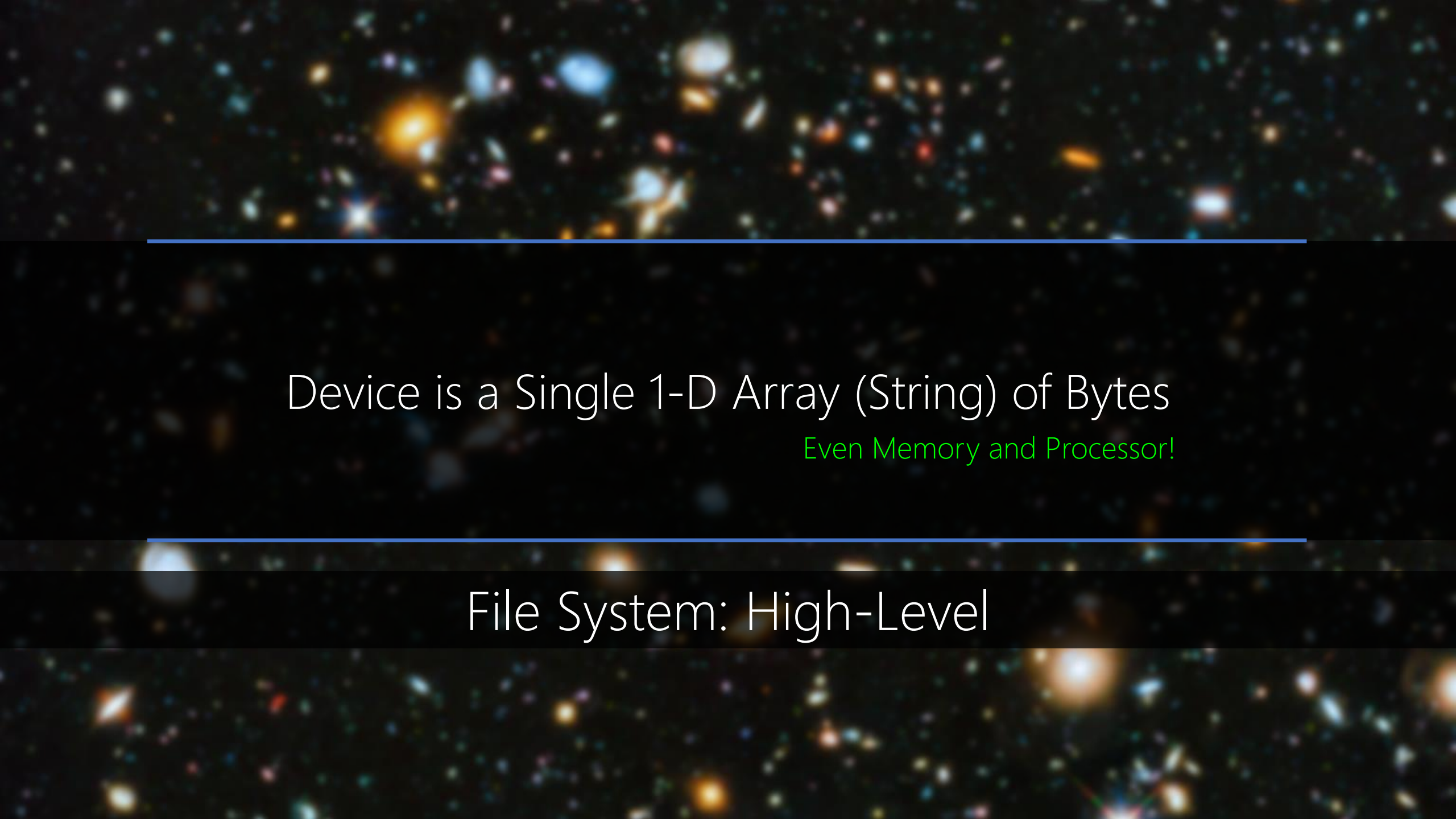
Bus

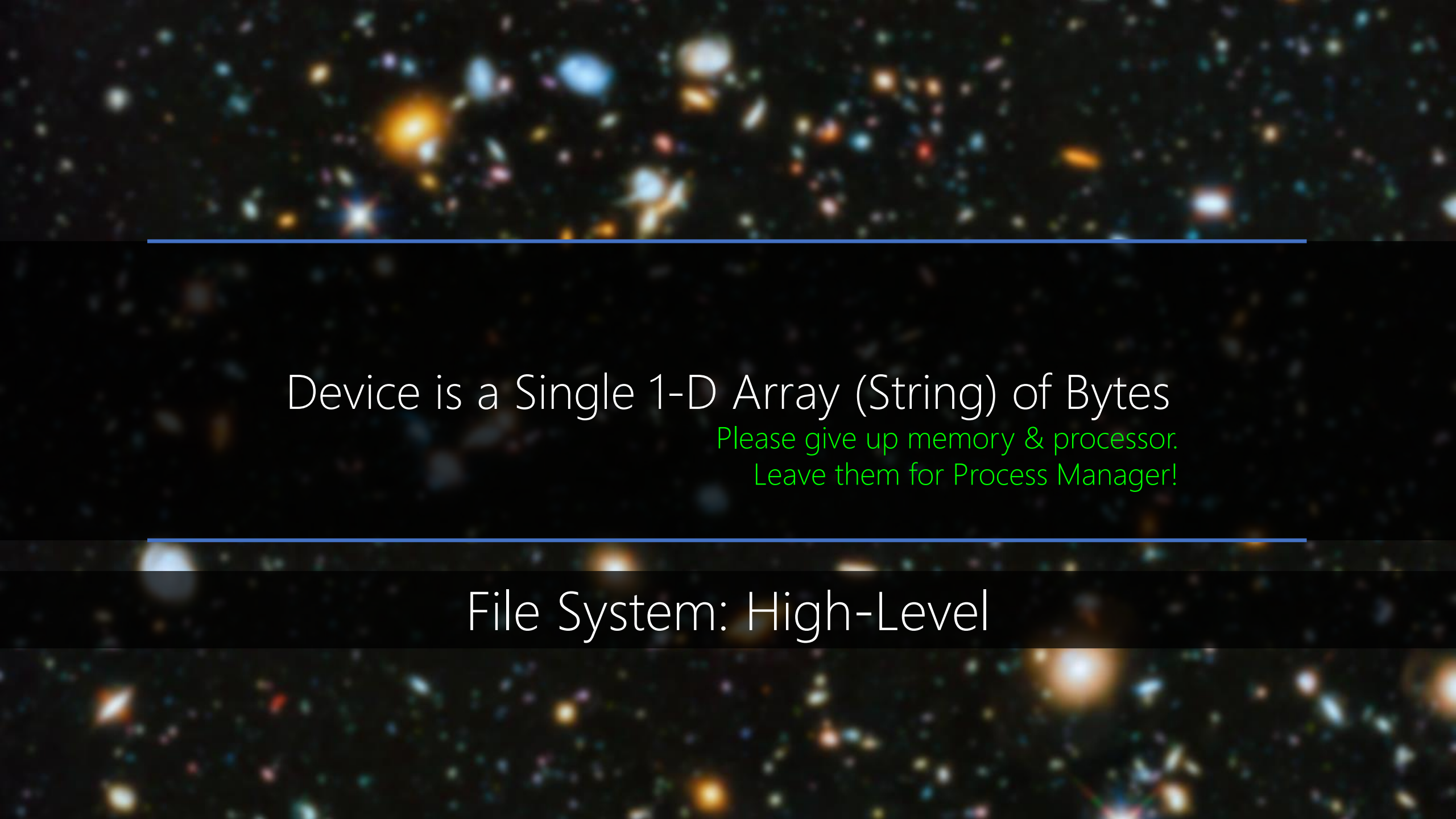
Processor

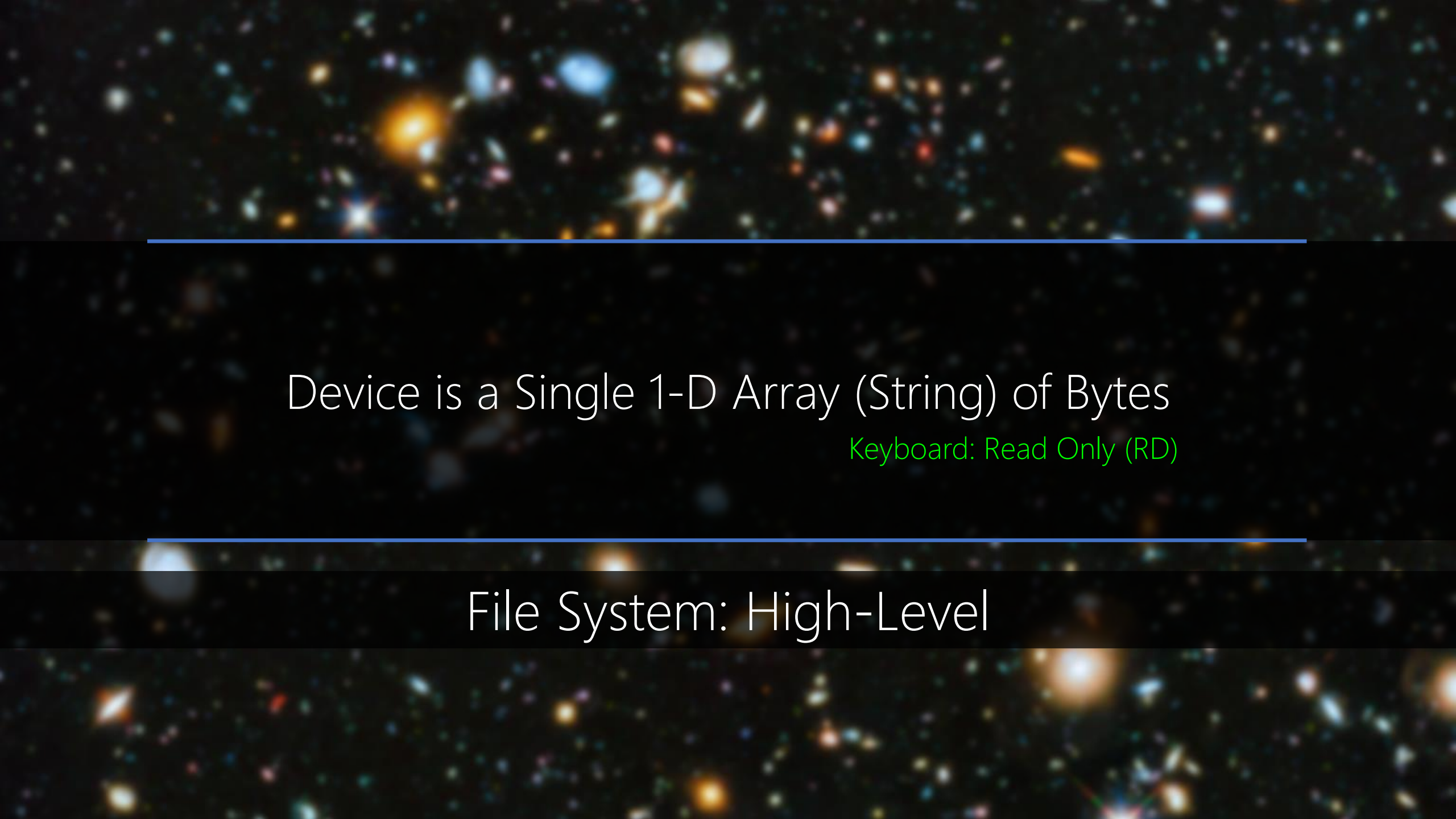


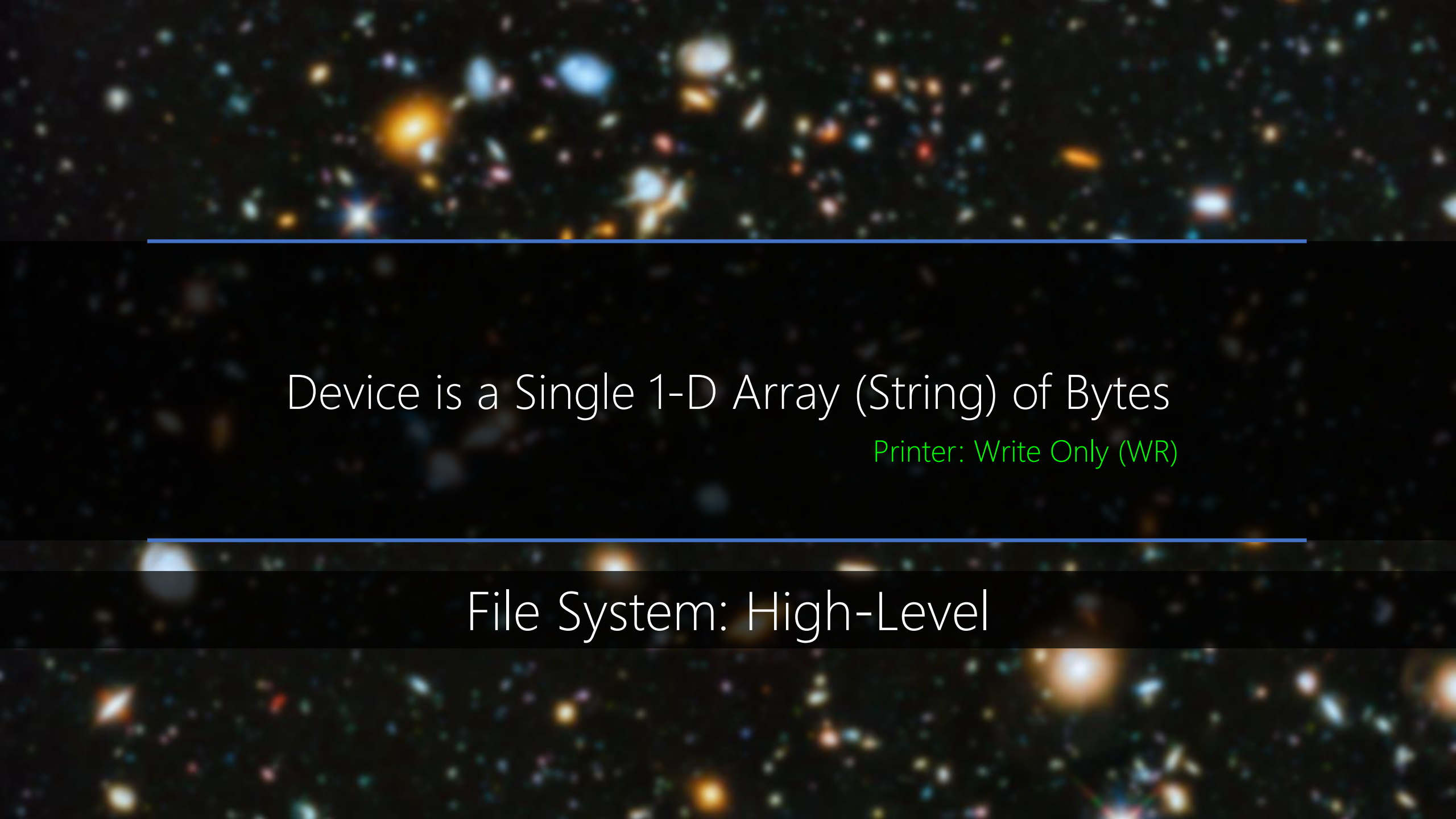

UNIX
High-Level
Low-Level
Machine
Device

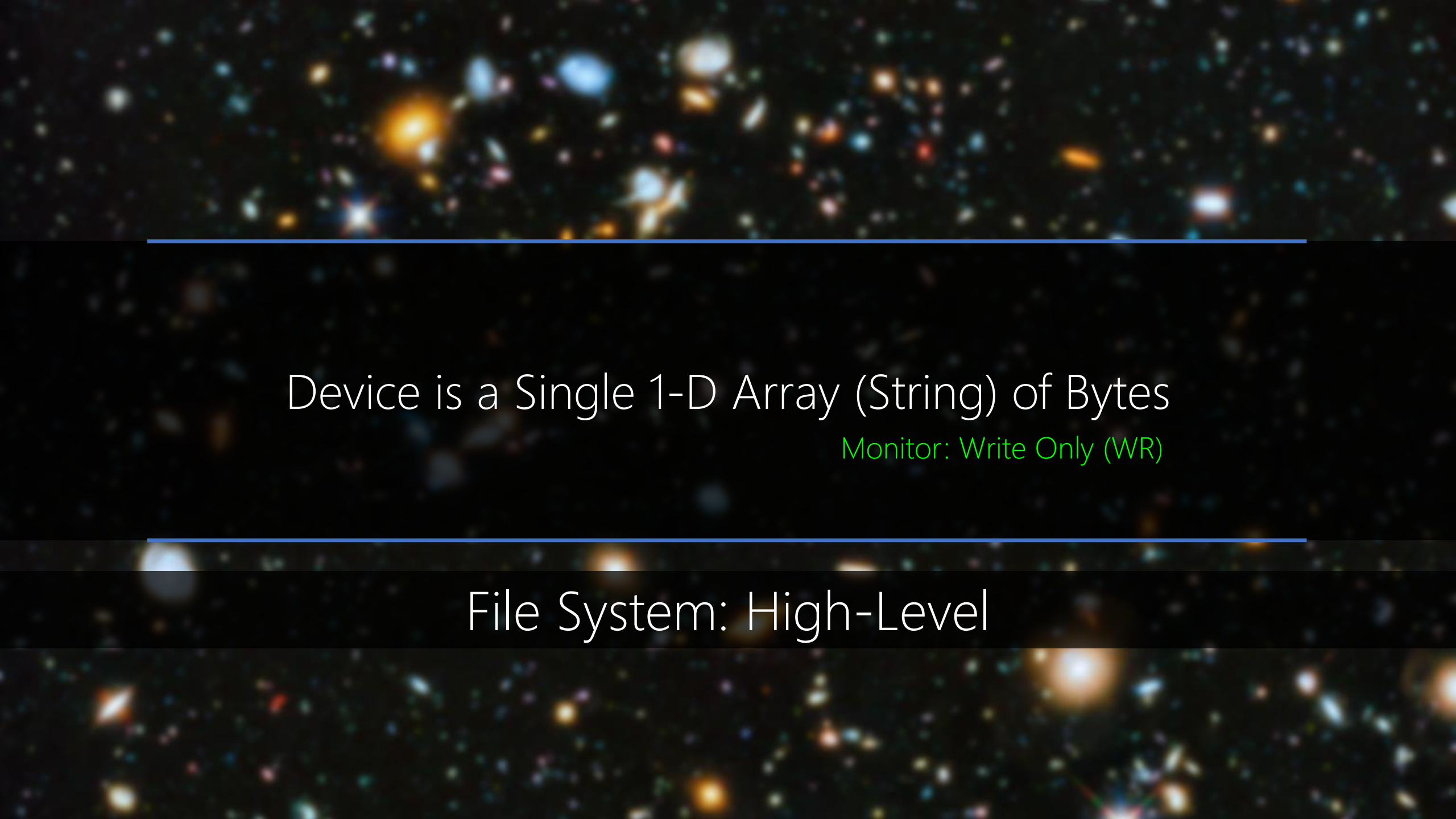

File Manager
widely known as File System
High-Level

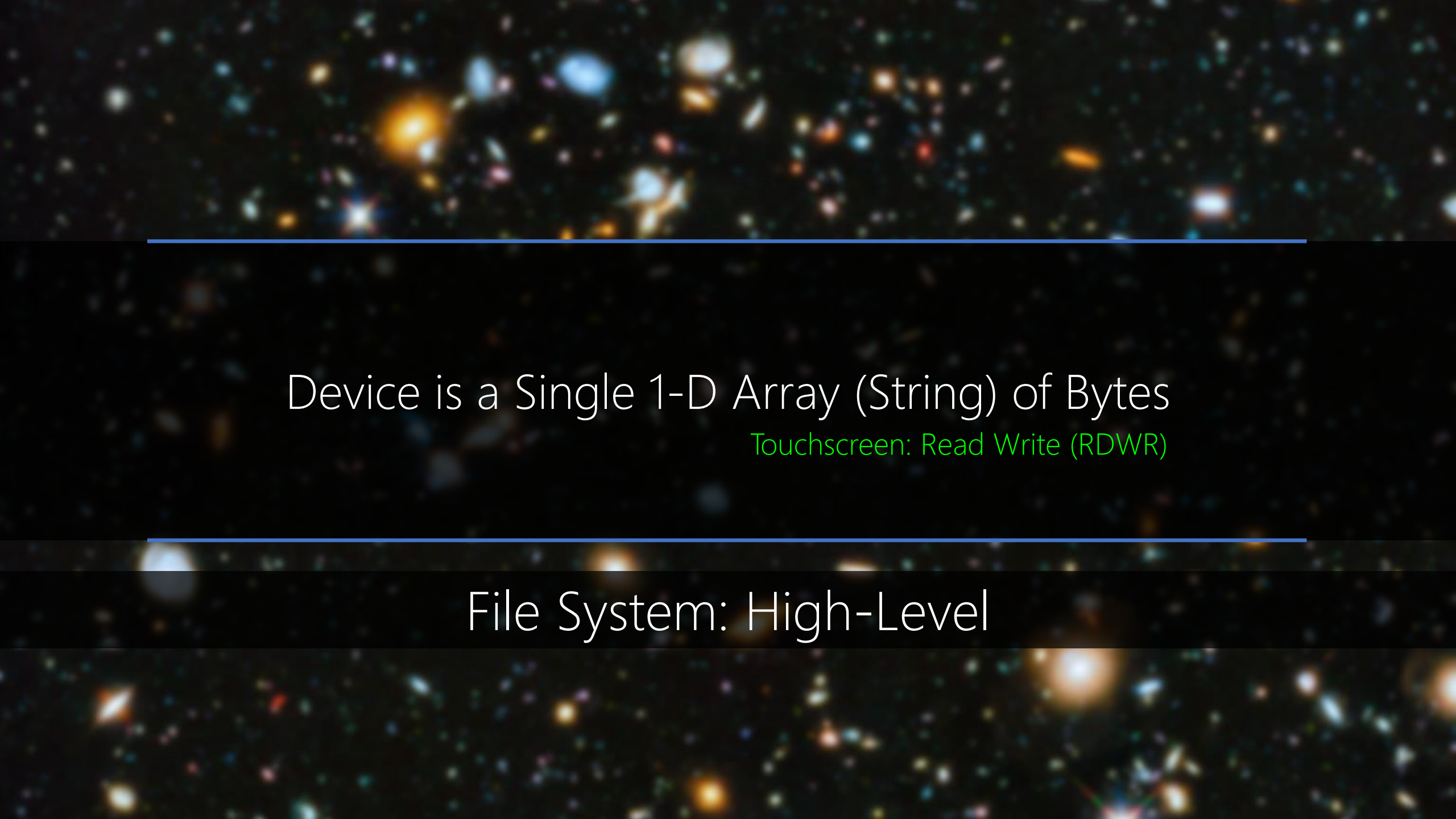

Device is a Single 1-D Array (String) of Bytes
Even Memory and Processor!
File System: High-Level


Device is a Single 1-D Array (String) of Bytes
Please give up memory & processor.
Leave them for Process Manager!
File System: High-Level

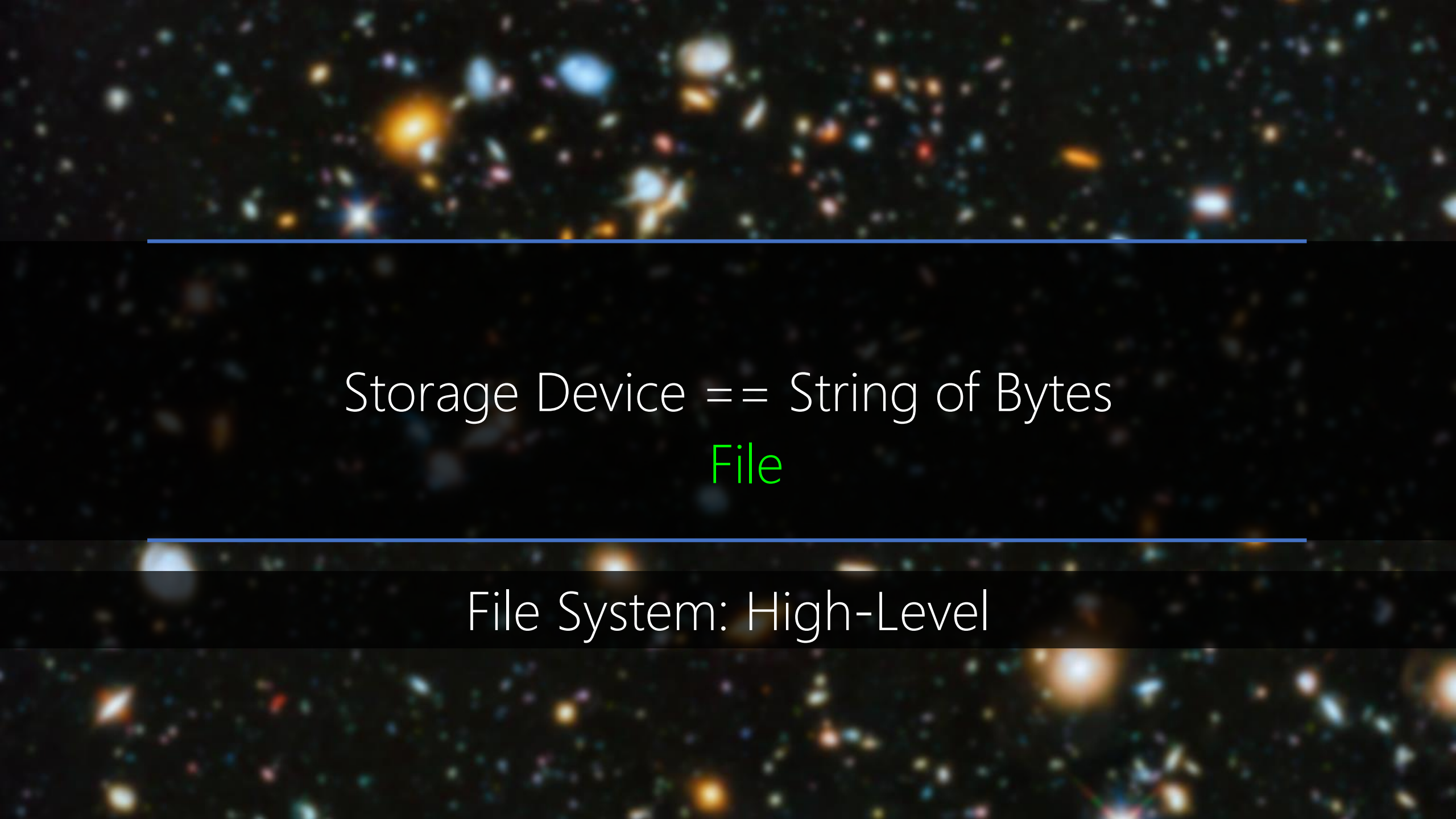

Device is a Single 1-D Array (String) of Bytes
Keyboard: Read Only (RD)
File System: High-Level

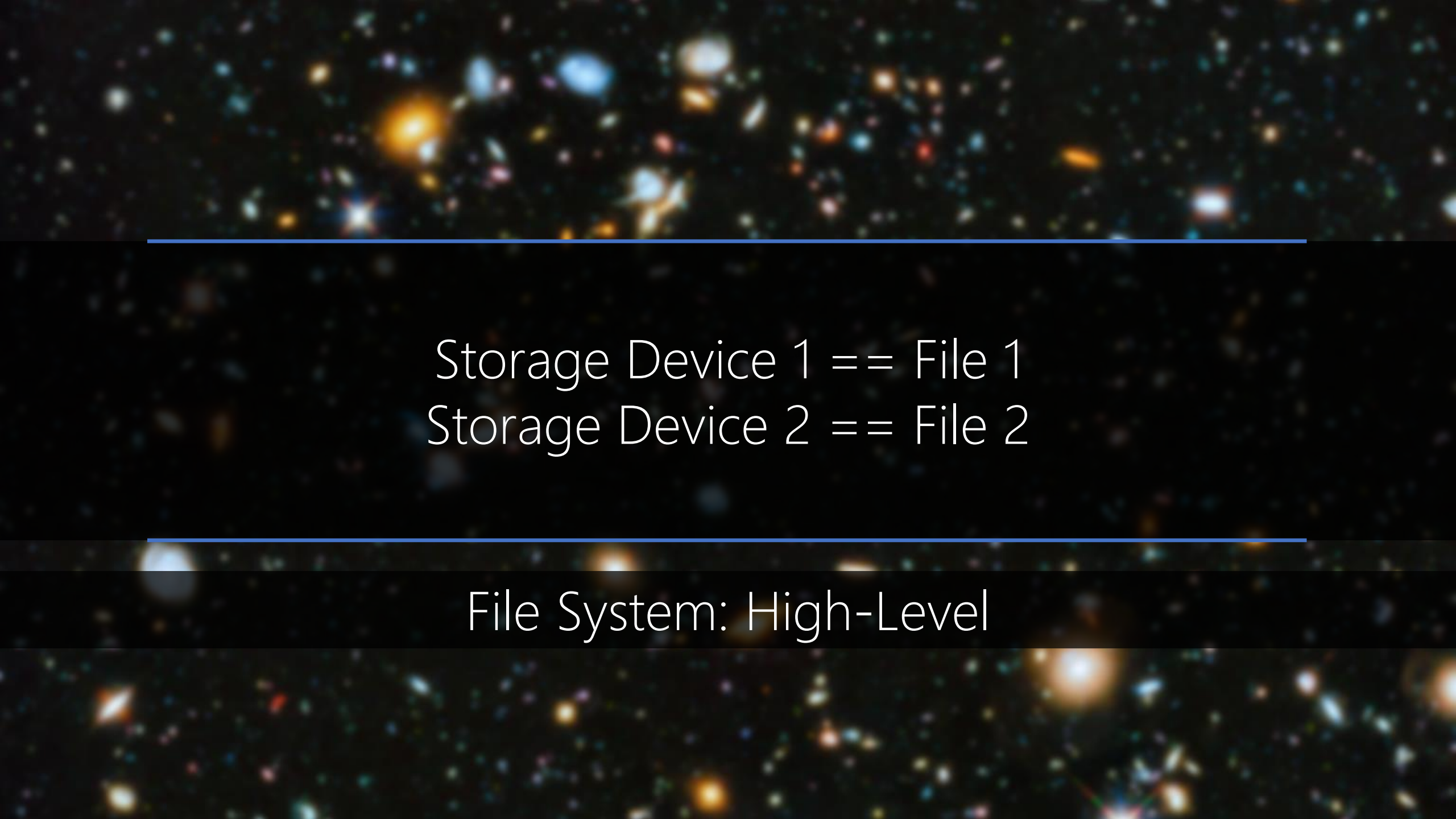

Device is a Single 1-D Array (String) of Bytes
Printer: Write Only (WR)
File System: High-Level


Device is a Single 1-D Array (String) of Bytes
Monitor: Write Only (WR)
File System: High-Level

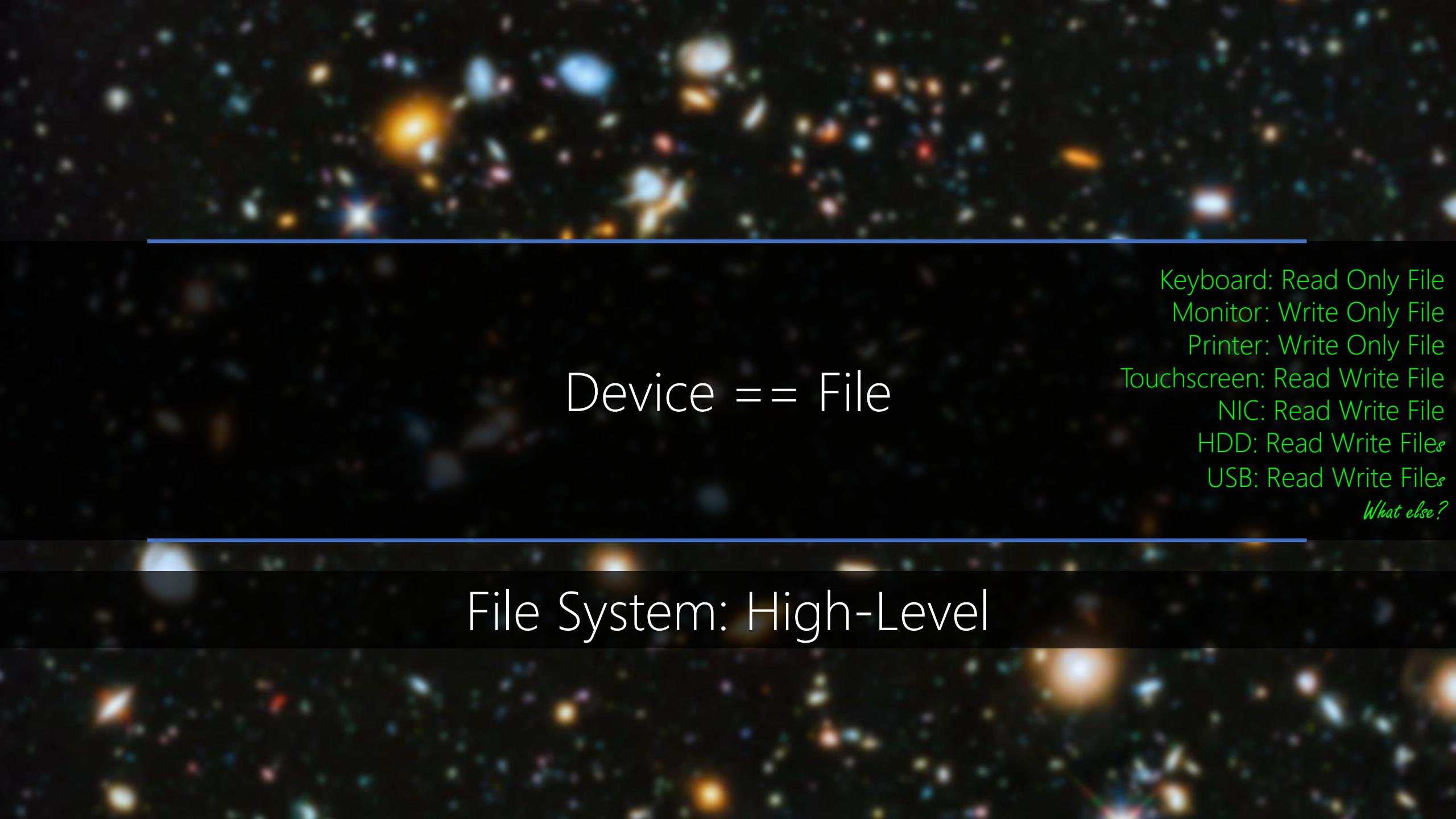

Device is a Single 1-D Array (String) of Bytes
Touchscreen: Read Write (RDWR)
File System: High-Level


Device is a Single 1-D Array (String) of Bytes
Storage: Read Write (RDWR)
HDD, USB, SSD, NVMe, CD-RW, DVD-RW
File System: High-Level


Storage Device == String of Bytes
File
File System: High-Level


Storage Device 1 == File 1
Storage Device 2 == File 2
File System: High-Level


Large Storage Device == Set of Files
set of sub-devices
File System: High-Level



Device == File

Keyboard: Read Only File
Monitor: Write Only File
Printer: Write Only File
Touchscreen: Read Write File
NIC: Read Write File
HDD: Read Write Files
USB: Read Write Files
What else?

File System: High-Level