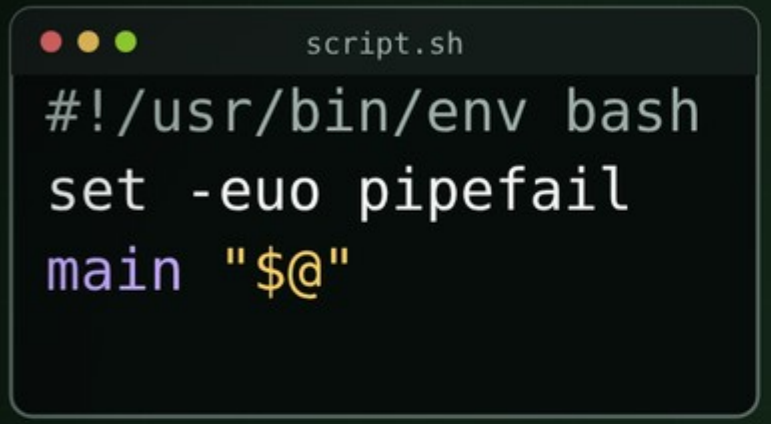
LESSON 12



$ _

Best Practices



A FREE MYWEBUNIVERSITY COURSE

BASH & ZSH SCRIPTING

LESSON 12

Best Practices and Next Steps

Portability, style, security, when not to use shell, and projects

A Professional Script Template

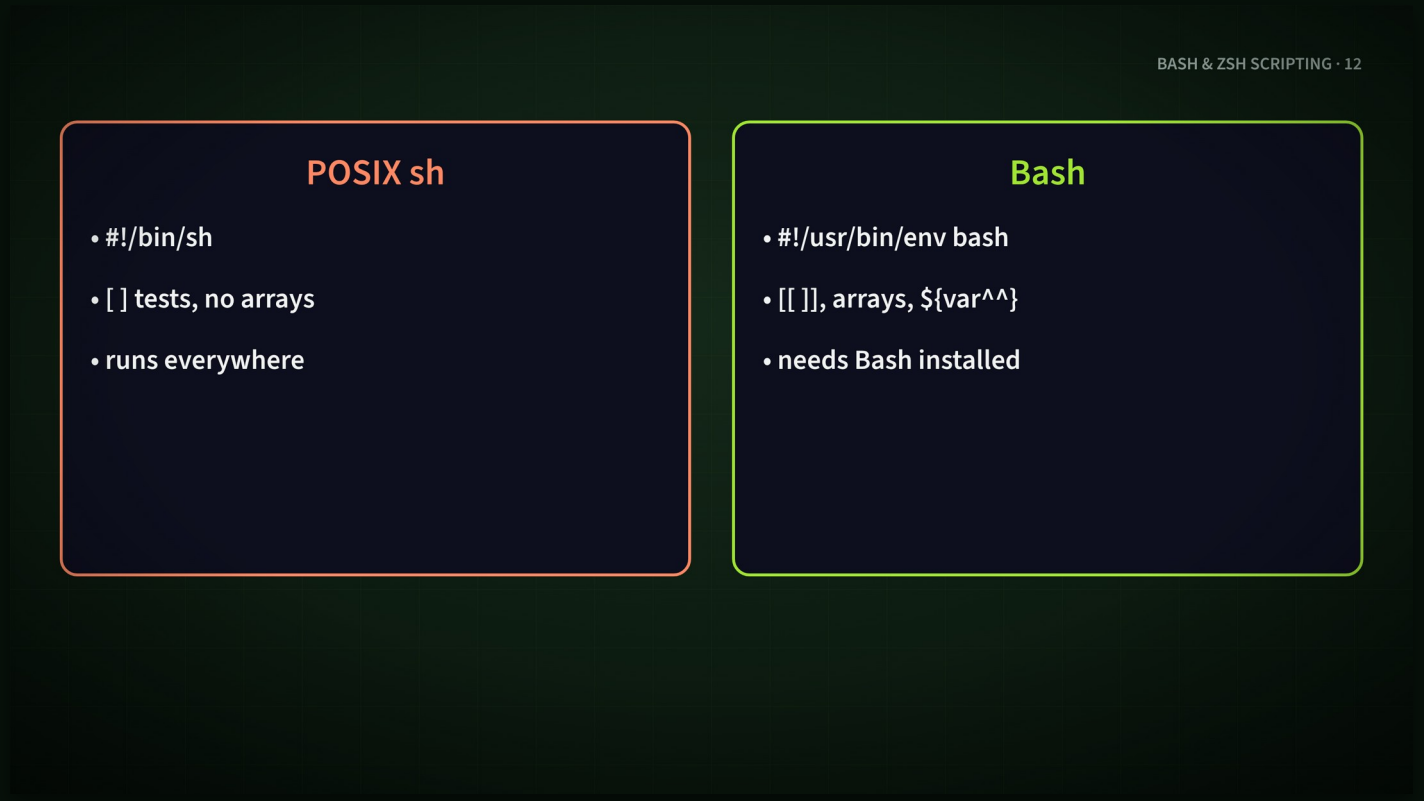
template.sh

```bash
#!/usr/bin/env bash
set -euo pipefail
usage() { echo "usage: ${0##*/} [-h] ARG"; }
die()   { echo "ERROR: $*" >&2; exit 1; }
main() {
    [[ $# -ge 1 ]] || { usage; exit 1; }
    echo "working on $1"
}
main "$@"
```

- Start every script from the same template: shebang, strict mode, a usage function, a die …
- A consistent shape makes every script in your team easy to read and change.

01

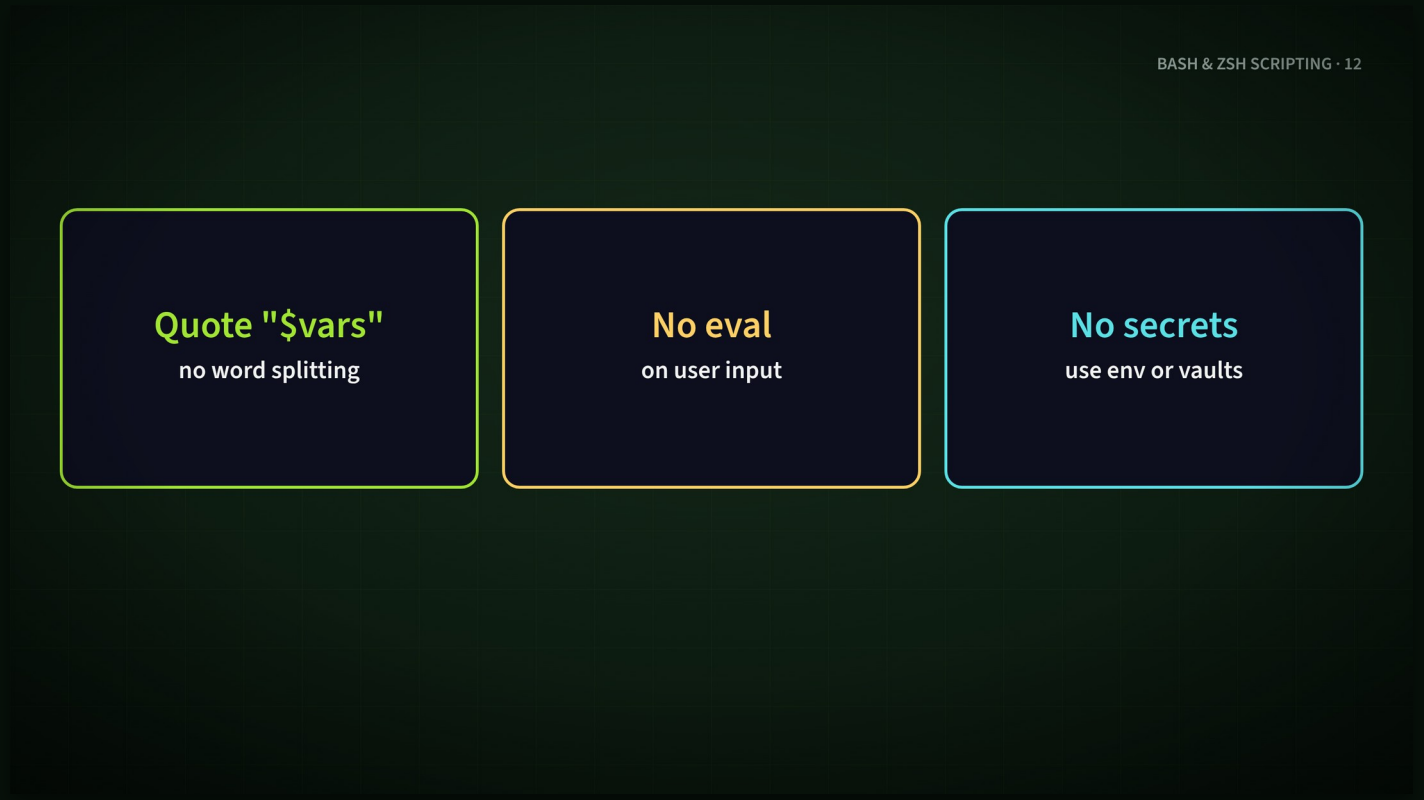
Portability



- If a script must run everywhere, including tiny containers, write POSIX sh: no arrays, no …
- If Bash features are fine, say so honestly with a Bash shebang, and use slash usr slash bin …

02

Security Rules



- Quote "$vars" — no word splitting
- No eval — on user input
- No secrets — use env or vaults

03

When Not to Use the Shell

Task	Best tool
run commands, move files	Bash / Zsh
columns, counting, reports	AWK
JSON, APIs, big logic	Python
Windows administration	PowerShell

- Shell is perfect glue, but past a few hundred lines, or when you need complex data, math, …
- A good rule: if you start fighting the shell, it is time for a real programming language. …

04

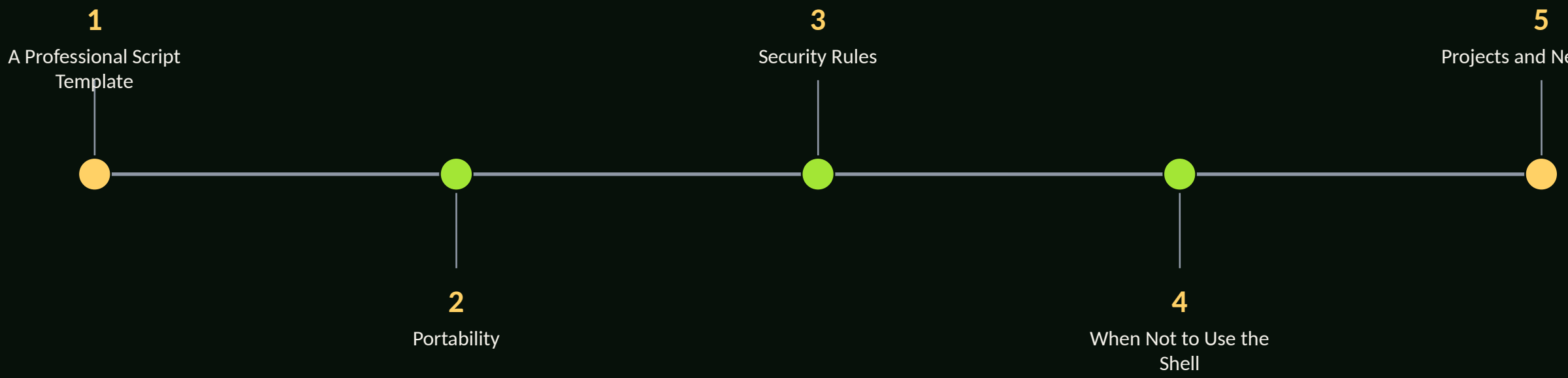
Projects and Next Steps



- 1 — Dotfiles installer
- 2 — Nightly backup
- 3 — Health report

05

Lesson Map


1
A Professional Script Template
2
Portability
3
Security Rules
4
When Not to Use the Shell
5
Projects and Next Steps

Sources and Further Reading





Thank you for watching

MyWebUniversity.com

Free technical education since 2000 — English and Dari

@MyWebUniversity · @MyWebUniversity-Dari