What is Bash Scripting?

Bash scripting means writing a series of commands in a **Bash (Bourne Again Shell)** file so the computer can execute them **automatically**, one after another.

Think of it like this □
□ *Instead of typing the same Linux commands again and again, you write them once in a file and run that file.*

What is Bash?

- **Bash** is a **command-line shell** used in Linux and Unix systems.
- It understands commands like:
- `ls`
- `cp`
- `mv`
- `rm`
- `pwd`

When you put these commands into a file, it becomes a **bash script**.

What is a Bash Script?

A **bash script** is:

- A **text file**
- Contains **Linux commands + logic**
- Usually ends with `.sh`
- Runs as a **program**

Example file: `backup.sh`

Simple Example (Very Easy)

Script file: `hello.sh`

```
#!/bin/bash
echo "Hello, World!"
```

Run it:

```
chmod +x hello.sh
./hello.sh
```

Output:

```
Hello, World!
```

Why Do We Use Bash Scripting?

Bash scripting helps to:

✔ Automate repetitive tasks
✔ Save time and reduce mistakes
✔ Manage servers
✔ Write DevOps automation
✔ Schedule jobs (cron)
✔ Monitor systems

Real-life example:
Instead of manually:

- Taking backup
- Checking disk space
- Restarting services

□ One script can do all of this automatically.

What Can You Do With Bash Scripts?

You can use:

- **Variables**
- **If–else conditions**
- **Loops**
- **Functions**
- **User input**
- **Files & logs**

Summary:

Bash scripting is writing Linux commands in a file to **automate tasks** instead of typing them manually.

- Uses **Bash (Bourne Again Shell)**
- Script files usually end with `.sh`
- Can run commands, conditions, loops, and functions
- Saves time and reduces errors
- Widely used in **Linux, servers, and DevOps**

In short: Bash scripting helps automate repetitive work in Linux.