🔷 Example with fg and bg

1. Start a process in the foreground

sleep 100

- This runs for 100 seconds.
- Terminal is **blocked** until it finishes or you stop it.

---

2. Suspend the process (pause it)

While sleep 100 is running, press:

Ctrl+Z

- This sends a **STOP signal** to the process.
- You'll see something like:
- [1]+ Stopped sleep 100

---

3. Send the job to the background (bg)

bg %1

- %1 = job number 1.
- The process resumes but keeps running in the **background**.
- Your terminal is now free again.

---

4. Check jobs

jobs

- Output:
- [1]+ Running sleep 100 &

---

5. Bring it back to foreground (fg)

fg %1

- The process comes back to the **foreground**, so the terminal is blocked until it finishes (or until you stop it again).

---

6. Kill the process if needed

Ctrl+C

- Terminates the foreground process.

🔷 **Quick Analogy**

- Ctrl+Z = "Pause the movie"
- bg = "Keep playing the movie in another room while I do other things"
- fg = "Bring the movie back to the main screen in front of me"

👉 **Ctrl+Z does *not* kill the process.**
It sends a **SIGTSTP** (terminal stop) signal, which means *"pause/suspend this process"*, not terminate it.

- A **stopped process** is still *alive* in memory, just not executing.
- You can see it with jobs or ps (status = T for "stopped").
- Then you can:
    - Resume it in the **foreground** with:
    - fg %1
    - Resume it in the **background** with:
    - bg %1

👉 By contrast:

- **Ctrl+C** sends **SIGINT** (interrupt) → process is *terminated* (killed).
- **kill -9 <PID>** sends **SIGKILL** → process is *forcefully killed* (cannot be caught or resumed).

So:

- Ctrl+Z = "Pause (you can resume later)"
- Ctrl+C = "Stop forever (killed)"