【GitLab】gitlab-runner 執行環境種類

下面整理成表格給你（主流可用的 executor）：

Executor	運作方式	優點	缺點	適用情境	設定/註冊要點
shell	直接在 Runner 主機的 shell 執行（bash / PowerShell）	最快、零容器開銷；可直接用主機已安裝工具	隔離最弱、容易「弄髒」主機；多專案易衝突	單機/內網、快速 smoke test、固定工具鏈	`gitlab-runner register --executor shell`；確保 `gitlab-runner` 使用者權限（如加入 `docker` 群組）與 `builds_dir` 可寫
docker	每個 job 以容器執行；`image:` 可指定環境	環境可重現、隔離佳、跨專案穩定	需管理映像；使用 Docker 需選擇掛 `docker.sock` 或 DinD	大多數 CI/CD 場景（前後端 build、容器化）	`--executor docker --docker-image <base>`；若 job 需 `docker build`：① 掛 `/var/run/docker.sock`，或 ② DinD：`privileged=true` + `services: docker:dind`
kubernetes	每個 job 建一個 Pod 來跑	彈性擴縮、資源/隔離/配額完善	需 K8s 叢集與維運成本	中大型團隊、尖峰負載、雲原生	`--executor kubernetes`；於 `config.toml` 設 namespace、serviceAccount、pull secrets 等
ssh	Runner 透過 SSH 登入遠端主機執行	可用現成遠端環境、無需在目標機器裝 Runner	隔離弱、擴展性差、權限管控要嚴格	特定機器/設備、臨時需求	`--executor ssh`；於 `config.toml` 填 host/user/key、路徑與環境
custom	以自訂腳本（prepare/run/cleanup）對接任意執行環境	彈性最高、可接非官方平台	需自寫與維護整套腳本，診斷較複雜	特殊平台、內部排程器	`--executor custom`；提供自訂 hook 腳本並測好錯誤處理

修訂版本 #1
★由 treeman 建立於 14 K 2025 10:09:18
由 treeman 更新於 14 K 2025 10:13:29