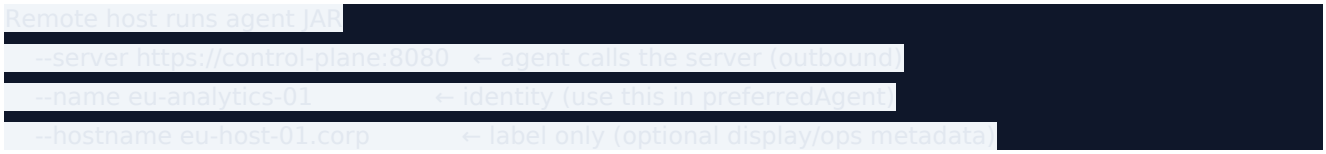
Just Run It — Agents Guide

Agents: registration, name, and hostname

Out-of-process workers claim and run jobs on the host where they run. This document explains **how agents identify themselves**, what **`hostname` means**, and how that differs from **preferred agent** on jobs.

Mental model



Field	Role
name	Unique agent identity. Used for register, heartbeat, claim, delete, and job **preferred agent**.
hostname	Optional label: “where this process says it lives.” **Not** used for networking or routing.
capacity	Server-side concurrent-claim limit for this agent name. Stock agent still runs **one job at a time** in-process.

Pull model: the control plane never dials the agent hostname. The agent process opens HTTP to the server (`AUTOMATION_SERVER` / `--server`).

How hostname is set

Agent process (`automation-agent` JAR)

Source: `agent/src/main/java/com/enterprise/automation/agent/AgentMain.java`

Default order for hostname:

1. `--hostname` CLI flag
2. else environment variable `HOSTNAME` (common on Linux/containers)
3. else `"localhost"`

On start, the agent registers with JSON:

```
{"name":"<name>","hostname":"<hostname>","capacity":5}
```

Example:

```
java -jar agent/target/automation-agent-0.1.0-SNAPSHOT.jar \
  --server https://automation.example.com \
  --name eu-analytics-01 \
  --hostname eu-app-01.corp.local \
  --user operator --password <secret>
```

Env equivalents: `AUTOMATION_SERVER`, `AUTOMATION_AGENT_NAME`, `HOSTNAME` (or `--hostname`), `AUTOMATION_USER`, `AUTOMATION_PASSWORD`, `AUTOMATION_CAPACITY`, `AUTOMATION_POLL_MS`.

Server API

`POST /api/v1/agents/register` (OPERATOR+)

- Body: `name` (required), `hostname` (optional; default `"unknown"`), `capacity` (optional; default 5)
- Implementation: `AgentRestController` → `AgentService.register(name, hostname, capacity)`

UI

Agents page form field **hostname** (default `localhost`) posts to the same register path via `UiController` → `AgentService.register`.

What the server stores

`Agent` entity (`agents` table):

Column	Notes
`name`	Unique, required
`hostname`	Optional string (up to 256 chars)
`status`	ONLINE / OFFLINE (heartbeat / stale)
`capacity`	Claim concurrency limit (server-side)
`runningJobs`	Counter

Column	Notes
`lastHeartbeat`	Updated on register and heartbeat

Register is **upsert by name**: same name reconnects (audit action `RECONNECT`), updates hostname, capacity, status ONLINE, and heartbeat.

Source: `AgentService.register`.

How hostname is used

Use	Does hostname participate?
Shown in Agents UI list	**Yes** (display only)
Register / reconnect payload	**Yes** (stored)
Heartbeat path	**No** — `POST /api/v1/agents/{name}/heartbeat`
Claim work	**No** — `POST /api/v1/agent/{name}/claim`
Preferred agent on a job	**No** — matches agent **name**
`AgentService.selectAgent`	**No** — ONLINE + capacity + **name**
Control plane → agent TCP/SSH	**No** — not implemented; agent pulls

There is **no** DNS lookup, reverse tunnel, or outbound connection from the server to `hostname`. A remote hostname value is only a human-readable label (e.g. `eu-app-01.corp.local`).

Preferred agent vs hostname

Jobs may set **Preferred agent** (`JobDefinition.preferredAgent`) to an agent **name**.

- Correct: preferred agent = `eu-analytics-01` (the agent's `--name`)
- Incorrect: preferred agent = hostname string (unless you also named the agent that way)

If preferred agent is empty, any ONLINE agent with free capacity may claim (least `runningJobs`).

In **agent execution mode**, the worker with that **name** must be running and claiming; hostname does not route the job to a machine.

Pre-register in UI, then start the agent

1. UI (or API) register creates/updates a row by **name**, stores hostname + capacity, marks ONLINE.
2. When the real agent starts with the **same name**, it registers again → **RECONNECT**: updates hostname/capacity/heartbeat on the same row.
3. If the process uses a **different name**, you get a **second** agent; the first goes OFFLINE after heartbeat timeout and is not the claim identity for the new process.

Mismatch trap: register UI as `agent-prod` with hostname `serverA`, but start:

```
--name agent-docker-1
```

→ control plane treats them as two agents. Prefer aligning **name** everywhere.

Lifecycle (relevant endpoints)

Action	Endpoint / path	Keyed by
Register	`POST /api/v1/agents/register`	`name`
Heartbeat	`POST /api/v1/agents/{name}/heartbeat`	`name`
Claim	`POST /api/v1/agent/{name}/claim`	`name`
Complete	`POST /api/v1/agent/{name}/complete/...` (see API)	`name`
Delete	UI / `DELETE` agents API	id or name
List	UI Agents, `GET /api/v1/agents`	—

Stale heartbeat → agent marked OFFLINE; mid-run work may become **ORPHANED** (see orphan simulation guide).

Related docs

- Security — agent Basic auth; agents are trusted compute
- Capabilities — agent assignment, capacity
- Orphan simulation guide — offline mid-run
- README **Standalone agent** section — quick start commands

Source map

Area	Path
Agent CLI / register body	`agent/.../AgentMain.java`
REST register	`server/.../api/AgentRestController.java`
Domain + service	`server/.../domain/Agent.java`, `service/AgentService.java`
UI list / form	`server/.../templates/agents/list.html`, `UiController`
Claim	`server/.../service/AgentWorkService.java`, `api/AgentWorkRestController.java`
Preferred agent	`JobDefinition.preferredAgent`, `AgentService.selectAgent`