Convention — projects/<client>/<skill>/<run_id>/

````
## §3 Proposed Convention

### 3.1 Root location

All skill outputs, evidence artifacts, and forge deliverables write under:

```
~/projects////
```

Where:
- `` — client slug (e.g., `bilko`, `drop`, `intesa`) or `alai` for internal ALAI work.
- `` — one of: skill name (e.g., `task-postflight`), `mc-` (for MC-scoped work), or `forge` (for
- `` — `YYYY-MM-DD--` (sortable by date, traceable to MC, unique per run).

### 3.2 Path table

| Output type | Path pattern | Concrete example |
|---|---|---|
| Skill output (generic) | `~/projects////` | `~/projects/alai/task-postflight/2026-05-05-99292-
| Builder evidence | `~/projects//mc-/evidence/` | `~/projects/bilko/mc-10611/evidence/` |
| Forge artifacts | `~/projects//mc-/forge/` | `~/projects/alai/mc-99102/forge/` |
| Brief (input) | `/brief.md` | `~/projects/alai/task-postflight/2026-05-05-99292-a1b2c3/brief.m
| Outcome (output) | `/outcome.md` | `~/projects/alai/task-postflight/2026-05-05-99292-a1b2c3/ou
| Supplemental evidence | `/evidence/` | `~/projects/alai/task-postflight/2026-05-05-99292-a1b2c
| Provenance record | `/.provenance.json` | `~/projects/alai/task-postflight/2026-05-05-99292-a1

### 3.3 run_id format

```
YYYY-MM-DD--
```

Example: `2026-05-05-99292-a1b2c3`

- **YYYY-MM-DD** — ISO date of run start. Enables `ls -t` chronological sort.
- **** — Mission Control task id. Enables direct lookup via `mc.js show`.
- **** — 6-character random hex. Prevents collision when the same MC triggers multiple runs in c

For runs not tied to a specific MC (e.g., background daemon output), use `mc-0` as the mcid segm

### 3.4 `.provenance.json` schema
````

```json
{
  "mc_id": "99292",
  "agent": "codecraft",
  "model": "claude-sonnet-4-6",
  "timestamp": "2026-05-05T21:26:00Z",
  "skill": "task-postflight",
  "client": "alai",
  "run_id": "2026-05-05-99292-a1b2c3"
}
```

All fields are string. `timestamp` is ISO 8601 UTC. Written at run start (not completion) so par

3.5 brief.md

The brief is the input contract for the run. It is written by the invoking agent before executio

Brief is written first, outcome last. If a run crashes, the brief survives and the run is recove

---

Revision #2
Created 2026-05-07 10:24:47 UTC by John
Updated 2026-06-07 20:01:30 UTC by John