Marc Kendal

Research Engineer — Python · Real-Time Systems · Research Infrastructure London, UK
marckendal@gmail.com · +44 7939 130 443 marckendal.com · github.com/MKToronto

---

Summary

Python engineer drawn to the work where the design doesn't exist yet — figuring out what the thing should be before building it. Five years owning the Python image-processing layer inside Fortress Technology's flagship ICON X-ray inspection machine — **real-time systems research engineering on Raspberry Pi edge hardware** (SysV shared memory + numba JIT + atomic file publishing + 9-lock async coordination; deployed at scale across Fortress's installed base). Since May 2025, solo-building a production-grade algorithmic trading platform (~200 Python / asyncio modules, live across four broker APIs), alongside **two Claude Code plugins** — one published on the Claude Code marketplace + Codex CLI, one private — that **operationalise expert methodology into installable research tooling** (López de Prado's three quant-ML books; Arjan Codes's clean-architecture teaching). Works spec-first with AI coding agents — plans before prompts, reviews every change before commit. **Every piece of code shipped at Fortress went out with zero post-ship bugs and no customer complaints.**

Experience

Founding Engineer — Automated Trading Platform (independent R&D) · May 2025 – Present

Sole architect and engineer of a production-grade algorithmic trading platform in Python / asyncio — ~200 modules covering execution across four broker APIs, risk management, and strategy iteration. Live against real capital; ML-ops layer filtering signals before they become orders under active development. *(Following paternity leave, Sep 2024 – April 2025.)*

- Owned **execution** end-to-end: order lifecycle management, position reconciliation against broker reality (three-step resolver: retry-with-backoff, query the original order's fill state, post corrective `PositionAdjustment`), corrective adjustment when local records diverge from live fills.
- Architected a single exchange-abstraction protocol behind which Interactive Brokers, IG, OANDA, Alpha Vantage, and a mock venue all implement; identical strategy code runs against live, paper, or mocked venues through one switch.
- Built the **async event-driven core** — candle aggregator with atomic-pointer-swap ring buffer (CPython GIL), event bus with mixed sync/async dispatch via `inspect.iscoroutinefunction` + `asyncio.to_thread`, timezone-aware overnight guard, shadow / paper-trade mode, adaptive rate limiting, startup guardian.
- Iterated the strategy framework through five generations against real trade logs — registry-pattern signal composition, partial-application signal functions, per-strategy balance tracking across candle, momentum, mean-reversion, and spread families.
- Built a separate FastAPI monitor sub-app with live PnL / Sharpe / drawdown dashboards, WebSocket updates, Pushover critical-alert integration, and runtime tuning endpoints exposing `tracemalloc` memory diagnostics.
- **Architected and shipped the published `python-clean-architecture`** Claude Code plugin — 12 commands (scaffold-api, refactor-legacy, extract-god-class, decouple, make-pythonic, review-architecture, etc.) + skill pack (7 design principles with refactoring recipes, 22 code-quality rules, 25 Pythonic pattern references) + a working FastAPI hotel-booking example. **Dual-distributed on the Claude Code marketplace and Codex CLI.**

- **Architected and directed the private `quant-finance` plugin** — 17 skills + 5 commands + a dedicated `quant-expert` subagent operationalising Marcos López de Prado's trilogy (*Advances in Financial ML* | *ML for Asset Managers* | *Causal Factor Investing*). Covers triple-barrier labelling, fractional differentiation (expanding-window → FFD with ADF stationarity testing), covariance denoising (Marcenko-Pastur bounds), causal-factor validation with three Monte Carlo experiments (confounder / collider / mediation), seven-sins backtest validation (CPCV + embargo + deflated Sharpe + PBO). Used inside the trading research. Claude-assisted authorship; architecture, skill-boundary curation, and validation owned directly — the AI-assisted-authorship-of-shippable-expert-tooling skill named honestly.

Research Engineer — Fortress Technology (Toronto) · Oct 2019 – Aug 2024

Owned the Python computer-vision and data-ingestion layers across Fortress's industrial-inspection product line. Flagship ICON X-ray inspection machine deployed at scale across Fortress's installed base (many hundreds of production facilities). **Zero post-ship bugs and no customer complaints across shipped work.**

- **Owned the real-time image-processing pipeline** on Raspberry Pi edge hardware, alongside legacy C / OpenGL / Perl: C detector code writing into SysV shared memory; Python reading via `sysv_ipc` with byte offsets computed programmatically from a numpy dtype mirror of the C struct (`extract_constants.py` parses the C header and caches to JSON, keeping the C ↔ Python contract from drifting); **numba-JIT** hot paths (`@jit(nopython=True, parallel=True, fastmath=True)` with `prange`) including adaptive-geometry rendering with **sqrt-scaled brush radius** under performance pressure and recursive hole-filling with configurable stop thresholds; atomic `hardlink_to` → `.replace()` file publishing; FastAPI / WebSocket layer coordinating nine distinct asyncio locks over disjoint state regions, with a ThreadPoolExecutor / asyncio bridge bringing SHM polling into the async server. **Telemetry-by-correlation** (`stats_on_production.py`) uses Pearson coefficients between render time and mask characteristics to find real bottlenecks empirically rather than guessing. *(product demo)*
- **Led the on-device contaminant-detection modelling work** (bone and metal classification), including a coordinate-auto-generation function that made object-detection training viable on production-shaped data. Directly influenced hardware design of the next generation of machines.
- **Made the strategic call to invest in labelled-data acquisition** at R&D-leadership level, then designed and built the `image_dev_tool_fastapi` tool — FastAPI + Svelte + Redis-queue pipeline that pulled time-windowed image batches from fielded machines over SSH and gave operators a UI to categorise contaminants. That tool built the dataset every downstream anomaly-detection model relied on.
- **Built the SvelteKit operator UI** running on every shipped ICON X-ray (`xray_streaming`) — kiosk-mode Chromium served via `sirv` on Pi, with a full offline-deployable installation pipeline. Iterated through seven streaming architectures (aiortc / WebRTC / HLS / gstreamer / various queued-Python designs) before landing on the kiosk design — path-finding through bad designs as part of the research. *(Fortress ICON product page)*
- **Founding engineer on the Knapp pill-dispensing product** — Flask control system shipped through external-customer handoff on a first Fortress project with zero post-ship bugs.
- **Owned the HMI translations pipeline (`Translations_v3`)** — bidirectional JSON ↔ Excel tool synchronising 40+ languages across every Fortress machine.
- **Mentored junior engineers** on the React + Python auth stack through pair programming and code review; 150+ bugs closed together with measurable skill growth.

Software Engineer — British Airways (London, UK) · 2014 – 2016

- Built live-data interception code for ba.com in Python (aspect-oriented programming / AOP) — captured real production data streams so tests ran against realistic live inputs instead of synthetic stubs.
- Built Selenium / Cucumber / Concordion BDD suites for desktop and mobile releases.
- Co-authored a "date-rolling" Python service that kept stubbed test data valid as calendar dates advanced — adopted across all teams.

Selected Projects

- `python-clean-architecture` — Published Claude Code marketplace plugin (dual-distributed to Codex CLI). 12 commands + skill pack (7 design principles, 22 code-quality rules, 25 Pythonic patterns, ~50 reference knowledge files) + working FastAPI example. Honest attribution to Arjan Codes.
- **Automated trading platform** — ~200 Python / asyncio modules; four-broker execution abstraction; custom risk-management + reconciliation layer; five strategy generations; live.
- `quant-finance` — Private Claude plugin operationalising López de Prado's trilogy into 17 skills + dedicated `quant-expert` subagent. Personal-use in trading research.
- `xray_graphics` **production pipeline** — SysV shared memory + numba JIT + adaptive geometry + 9-lock async coordination on Raspberry Pi. Deployed inside Fortress's flagship ICON X-ray machine.
- `localytics` — Published self-hosted alternative to GitHub Insights with **function-level cyclomatic complexity** (via Radon), weekly / monthly / yearly activity heatmaps, and per-file churn. Two FastAPI apps — local scanner + Render-hosted dashboard (FastAPI + Redis) — exchanging only aggregated numbers; source code never leaves the machine. Built to track the trading platform's development; MIT-licensed. Live demo.
- `docugit` — Published CLI extracting detailed git code-changes over configurable time-ranges for R&D tax-credit report generation via LLMs.
- **Shiurim transcription pipeline** — Whisper + SpeechBrain speaker ID + Claude Code CLI classification over a large Dropbox audio corpus; ChromaDB-backed natural-language query plugin (`triebitz-halacha`).

Tech

Primary: Python · C (ctypes + shared memory) · JavaScript / Svelte / SvelteKit **Scientific / Research Python:** NumPy · SciPy · Pandas · numba · scikit-learn · FastAI · PyTorch · OpenCV **Systems / Infra:** FastAPI · asyncio · Docker · SQLite · Redis / RQ · WebSockets · systemd · mamba / conda · Raspberry Pi deployment · paramiko / SCP · memray profiling · SysV shared memory **AI tooling:** Claude Code plugin architecture · Anthropic SDK · Codex CLI distribution · ChromaDB · Whisper · SpeechBrain **Working practice:** Agile · CI/CD · code review · unit testing · refactoring · spec-driven, agent-orchestrated development · review every change before commit · Claude Code + Codex CLI day-to-day

---

Education

MSc Data Science · City, University of London · 2016 – 2019 Dissertation (Computer Vision): *Improving the Optical Character Recognition of Middle Eastern Languages* — PyTorch CNNs, convolution analysis and heat-maps, classifier interpretation.

MSc Philosophy of the Social Sciences · London School of Economics Certificate of attendance. Left to pursue formal philosophy studies in Jerusalem (see below).

PGDip Philosophy · The Heiden Institute

BSc (Hons) Economics, 2:1 · University of Nottingham · 2005 – 2008