KEVIN 'TYLER' COX

Unreal Engine 5 C++ Gameplay Systems Developer

(843) 718-4024 kevincox103@gmail.com Charleston, SC

github.com/Ty-lerCox linkedin.com/in/tyler-cox-1715065a/ ty-lercox.github.io/portfolio/

Summary

UE5/C++ systems designer/developer focused on systemic, player-facing loops—progression (quests/codex), economy (inventory/shop), and needs-based population/crowd flow (traffic-like) simulation. Favor maintainable C++ for core logic with targeted Blueprints for UI/prototyping where they make sense. Author of clear specs, data schemas, and tuning frameworks; telemetry-savvy (Grafana stack), comfortable with live debugging and fast iteration.

Education

Computer Science
Trident Technical College

Technical Skills

Unreal Engine 5, C++ (Gameplay Framework, UObjects/Actors/Components), Blueprints (prototyping/UI glue), UMG/HUD, Dialogue & Quest Systems, Inventory/Economy Loops, Population/Crowd Simulation (needs-based, traffic-like), Redux-style State (actions/effects/state), Data-Driven Design (Data Assets/Data Tables), Tooling & Debugging, Profiling & Optimization, Git

Soft Skills

Systems Thinking, Problem Solving, Communication, Cross-Discipline Collaboration (remote/time zones), Constructive Reviews; Open to Mentorship, Iteration & Rapid Prototyping

Additional Skills

Design Specs, Data Schemas & Tuning Frameworks, Telemetry-Informed Balancing (Grafana/Prometheus/Loki/Tempo experience), Cinematic UI/UX, Testability

Work Experience

Independent / Self-Employed Dec 2024 — Present

UE5/C++ Game Systems Developer

Design and implementation of core gameplay systems and tools in Unreal Engine 5 with emphasis on maintainable, data-driven C++.

- Built branching Dialogue System with multi-option choices and conditional availability based on quest state (current/completed/required).
- Engineered Quest System supporting multi-objective tasks, required counts, dependencies, and side effects (trigger cinematics, spawn NPCs, state changes).
- Established Redux-style architecture in C++ (actions, effects, state) to isolate subsystems and improve testability and maintainability.
- Created AI movement and a needs model; authored advanced spawner logic for unpredictable NPC distribution with region rules, cooldowns, and variance; implemented needs-based population/foot-traffic simulation (crowd flow).
- Implemented story-management and transition volumes to control floor/zone visibility and scene flow.
- Developed cinematic HUD widgets and web-inspired UI components to accelerate iteration and improve UX.
- Delivered inventory and shop subsystems integrated with global game state; built a codex/collection system to track collectibles and progression.
- Integrated an in-game guide chatbot (LLM) with command interface; used context caching to reduce token usage and latency.
- Authored design specs, data schemas, and a tuning framework (Data Assets/Data Tables) for consistency and readability across systems.
- Used Blueprints selectively for UI wiring and quick prototypes while keeping long-term maintainable logic in C++.
- Provided constructive feedback in reviews; open to offering/receiving mentorship.
- Practiced data-driven configuration, profiling, and optimization across systems; designs are telemetry-ready and informed by prior Grafana stack experience.
- Additionally shipped TypeScript-based game mods (feature ownership and UX), separate from C++ work.

Expeditors International (Expeditors) Aug 2014 — Present

Angular Application Developer (Additional Professional Experience)

Primary employer; application development, DevOps, and observability (transferable engineering practices).

- Led state management patterns (NgRx, Angular Signals) and modular architecture across multiple internal applications.

& Maintainability, Documentation, Kanban & Delivery Flow

Languages

English

- Stood up observability with Grafana stack (Prometheus metrics, Loki logs, Tempo traces) and alerting; emphasized instrumentation and telemetry—skills directly applicable to gameplay event logging and telemetry-informed balancing.
- Automated CI/CD with GitHub Actions/GitLab Runners and Ansible; supported Kubernetes deployments; improved iteration speed and reliability.
- Implemented SSO (Keycloak OIDC/SAML), Kerberos integrations, and PKI/CA for TLS/mTLS; enforced secure defaults.
- Modernized data flows from bespoke Kafka producers to database-level CDC → Kafka with medallion layers (bronze/silver/gold).
- Used Python for rapid prototypes; production services in Java Spring Boot and C#/.NET; practiced feature flags, DORA metrics, and Kanban for flow.

Projects

Dialogue System (UE5/C++) Apr 2025 — May 2025

Branching, conditional dialogue with quest-aware availability and consequence tracking.

- Multi-option choices; gating by current/completed/required quests; integrates with Redux-style game state; documented usage and tuning parameters.

Quest System (UE5/C++) Apr 2025 — May 2025

Objectives with counts, dependencies, and side effects wired to gameplay events.

- Triggers cinematics; spawns NPCs; updates global state via actions/effects; specs, data schemas, and tuning framework included.

AI Needs & Advanced Spawning May 2025 — Jun 2025

AI behavior model with needs (Sims-like) and region-based spawner rules.

- Needs-based routing and foot-traffic (crowd/traffic-like) simulation; unpredictable yet controlled distribution; cooldowns and variance; tunable via data assets.

Story/Level Flow Controls May 2025 — Jun 2025

Volume-based story transitions with floor/zone visibility controls.

- Performance-aware visibility; clean narrative gating; improved player readability; Blueprint glue for UI interactions when appropriate.

Inventory & Shop Subsystems Jun 2025 — Jul 2025

Item catalogs, purchasing, and persistence integrated with global game state.

- Economy loop foundation; predictable side effects; data-driven tuning; testable modules; readable UI prompts.

Codex/Collection System Jun 2025 — Jul 2025

Collectible tracking and progression feedback loops.

- Player progress surfaces; unlock conditions; consistent data schema; tuning exposed via tables.

Guide Chatbot Integration (LLM) Jul 2025 — Aug 2025

In-game user-guide chat agent using language models.

- Command interface; context caching to lower inference cost and latency; developer notes for configuration; clear UX for player guidance.

Redux-Style Game State Library Jul 2025 — Aug 2025

Shared actions/effects/state managers used across gameplay subsystems.

- Separation of concerns; maintainable C++ modules; easier testing and debugging; versioned APIs and integration notes.

TypeScript Game Mods (Shipped)

Shipped game mods written in TypeScript; owned feature design and UX within mod constraints.

- Shipped mods; documented design decisions and tuning notes; collaborated with player feedback loops.