Judd Cohen

Seattle, WA
Phone: 206-552-0118
Email: jcohen@juddnet.com
Last updated: *May 2023*

TECHNICAL SKILLS

Keystone Skills	**Unreal Engine**: 10 years of gameplay programming experience **C++**: 10 years of experience **Python**: 15 years of experience
Tools	Unreal Engine 5, Unreal Engine 4, Unity3D, Blender, Photoshop
Programming	C++, Python, C#, Rust, Java, Javascript, Typescript, Dear ImGui, Git

EMPLOYMENT

2019 — 2023

City State Entertainment
Gameplay Programmer - *Seattle, WA*

Worked on:
Camelot Unchained

Final Stand: Ragnarok (2021)

- Built gameplay features in custom massively multiplayer game engine across a wide array of systems (designer gameplay data, server logic and networking, client processing, and user interface)
- Supported live multiplayer game - regularly went on-call to monitor servers
- Overhauled user interface by writing a custom tool to parse the entire codebase, automatically make needed API changes, and extract data
- Built custom server data management tool for designers and other gameplay programmers (TypeScript and React)
- Built custom artist tool for generating textures (C++ and Dear ImGui)
- Regularly reviewed and approved pull requests from other programmers

2017 — 2019

Studio Wildcard
Gameplay Programmer - *Seattle, WA*

Worked on:
ARK: Survival Evolved (2017)

ARK: Aberration (2017)

ARK: Extinction (2018)

ARK: Genesis (2019)

- Collaborated with character artists, animators, and VFX/tech artists to build rideable dinosaurs, weapons, placeable structures, and items
- Built a mission system from scratch as a new low-level core gameplay system with an API that is easy for designers to use
- Assisted with planning, time estimates, and scheduling; mentored less experienced gameplay programmers
- Built complete playable multiplayer prototype of an unannounced game including an inventory system, structure building system, weapon system, and resource harvesting
- Supported live multiplayer game; added numerous features and fixed bugs in existing gameplay systems without breaking old content

2013 — 2017

Zen Relay Games
Lead Developer, Lead Designer, Owner - *Seattle, WA*

> *Worked on:*
> **Geoid** (2017) `https://store.steampowered.com/app/636910/Geoid`

- Designed, developed, and shipped **Geoid**, a 3d platformer game
- Managed the full game development process, including design docs, prototyping, implementation, testing, polish, and release
- Built using Unreal Engine 4 with game logic in C++ and art assets created in Blender and Photoshop
- Streamlined modeling workflow by writing custom Blender add-ons in Python

2015 — 2016

Tricky Fast Studios
Game Developer (Contractor) - *Leominster, MA*

- Developed mobile games with Unity3D
- Integrated 3rd party C# asynchronous libraries into an existing framework and codebase
- Worked with producers, artists, and lead server architect to integrate features

EDUCATION

Graduated 2015

University of Puget Sound
Tacoma, WA

- Bachelor of Science in Computer Science
- Studio Art Minor, specializing in interactive wood sculptures using Arduinos

PERSONAL PROJECTS

2023

JXC
Language Designer, Lead Developer

> *Worked on:*
> **JXC Structured Data Language** (2023) `https://github.com/juddc/jxc`

- A language similar to JSON but designed to be pleasant to edit for game designers, and easy to extend for gameplay programmers
- Designed language syntax, wrote the reference implementation in C++, wrote extensive unit tests and comprehensive technical documentation
- The language syntax supports key/value maps, arrays, type hinting, numbers with units, timestamps, comments, and custom lexical expressions (for embedding math expressions or small snippets of game logic into configuration files)
- Wrote syntax highlighting extensions for Sublime Text and Visual Studio Code

2015

Hoshiko
University of Puget Sound Computer Science Capstone Project

- A game using Unreal Engine 4 that taught basic programming concepts in a visual way by having the player enter code into in-game computer terminals
- Each in-game computer terminal had an embedded Python subinterpreter thread with APIs for controlling game mechanics (for example, typing door.open() would open a door connected to that terminal)
- The in-game code editing UI was powered by Chromium Embedded Framework