Jason Komoda

jason.komoda@gmail.com | 808.389.3619 | *jasonkomoda.github.io*

Education

University of Denver – Denver, CO
Bachelor of Science in Computer Science (Game Development) **Graduated: June 2018**
Minor in Mathematics and Emergent Digital Practices

- Major GPA: 3.8

Mahidol University International College – Salaya, Thailand **September 2016 – December 2016**

- Study Abroad

Technical Skills

Languages/Frameworks: Java, C#, C++, HTML, CSS, Javascript, Typescript, Angular, Node.js, Ionic
Tools/DB: Git, WebGL, UNIX shell, Vim, Jasmine/Karma, SQL, MongoDB, Apache Cassandra

Work Experience

Everi – Las Vegas, NV **July 2018 - Current**
Software Developer I

- Converted the API for a legacy .NET application to use a NO-SQL database (Cassandra) instead of MSSQL and re-wrote the entire application using Ionic 4.
- Back end development using Koa.js/Node.js and Apache Cassandra DB.
- Front end development using Angular 6 and Ionic Framework.

Live in the Game LLC – Denver, CO **May 2017 – September 2017**
Software Development Intern

- Worked on two indie games for this startup using Unity2D/3D and C#.
- Worked with artists and project leads to implement new game features and story lines.

Projects

Ski-Royale [Unity2D, C#] *jasonkomoda.github.io/Ski-Royale*

- Created split screen multiplayer using multiple cameras, layer masks, and other UI elements.
- Developed an item system that uses RNG and current player state to determine the item on pickup.
- Implemented a combat system consisting of projectiles, collisions, health, and physics2D.

CollectNCapture [UE4, C++, Blueprints] *github.com/jasonkomoda/CollectNCapture*

- Programmed enemy AI movement and shooting using a behavior tree.
- Implemented a game mode and game state system, 3D animations, custom materials, terrain, particle systems, sound effects and HUD using widgets.

Spidey-Sense [Unity 2D, C#] *jasonkomoda.github.io/Spidey-Sense*

- Designed and implemented enemy movement, particle systems, sprite animations, coroutines, sound effects, level progression, and saving/loading using a singleton game model.