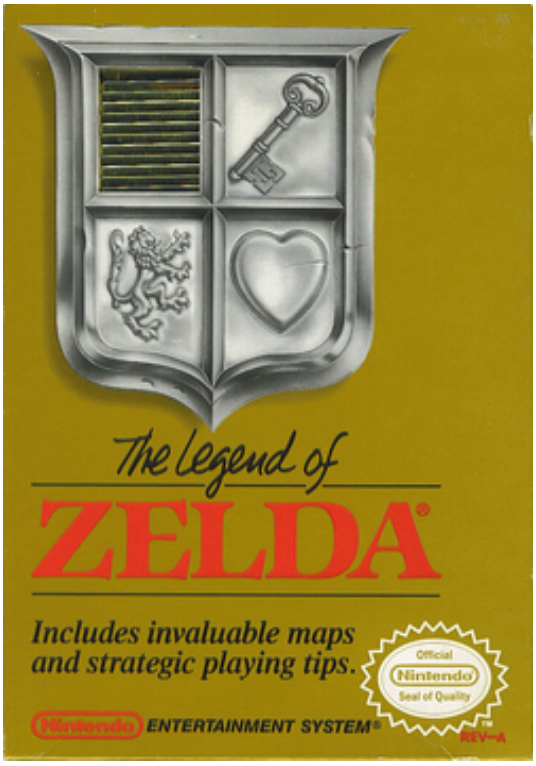
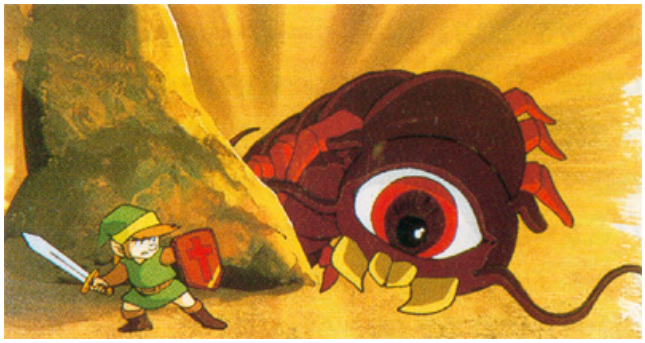
Technical Design Document (TDD)

The Legend of Zelda: Classic Edition





1. Introduction

1.1. Scope of the Document

This document outlines the technical approach to creating a faithful, identical recreation of the original "The Legend of Zelda" game released on the NES in 1987 for the Nintendo Switch.

2. Core Sections

2.1. Features from the GDD

- Faithful recreation of the original "The Legend of Zelda" game
- Identical gameplay, graphics, and audio
- Support for Nintendo Switch input methods

2.2. Technical Goals

- Efficient, modular, and extensible architecture
- High performance on the Nintendo Switch
- Accurate representation of original game assets
- Seamless integration with the Nintendo Switch ecosystem

2.3. Technical Risks

- Accurate emulation of original game mechanics
- Performance optimization on Nintendo Switch hardware
- Ensuring compatibility with Nintendo Switch input methods

3. Game Engine and Middleware

3.1. Choice of Game Engine

- Unity, for its robust feature set, extensive platform support, and ease of use
- Unity's 2D capabilities, sprite handling, and tilemap system will facilitate a faithful recreation of the original game's visuals and gameplay

3.2. Tools and Middleware

- Version control systems (e.g., Git) and project management tools (e.g., Jira) to streamline development and collaboration
- Unity's PlayerPrefs system and the Nintendo Switch's built-in save data management for handling save data and progress

4. Core Systems

4.1. Rendering

- Unity's built-in 2D rendering system, ensuring accurate representation of the original game's graphics

4.2. Physics

- Unity's 2D physics engine for movement and collision mechanics

4.3. Audio

- Unity's audio system for playback of original music and sound effects

4.4. Input

- Support for the Nintendo Switch's various input methods, including Joy-Con controllers and the Pro Controller

5. Data Management

- Game assets, such as sprites and tilemaps, sourced from the original game and converted into appropriate formats for use in Unity
- Configuration files for storing game settings and parameters, allowing for easy tweaking of values during development

6. Platform-Specific Considerations

- Optimization for performance on the Nintendo Switch
- Support for features unique to the Nintendo Switch, such as HD Rumble and touch screen controls

7. Performance and Optimization

- Techniques such as sprite atlasing, object pooling, and efficient data structures
- Regular profiling and testing for optimal performance on the Nintendo Switch hardware
- Maintain a consistent frame rate of 60 FPS
- Limit memory usage to fit within Nintendo Switch specifications
- Optimize asset loading times

8. Testing and Debugging

- Unit tests for critical game systems and mechanics
- Comprehensive test plan for thorough testing of all game features and functionality

9. Deployment and Distribution

- Unity's built-in support for Nintendo Switch deployment to create a game package that meets Nintendo's requirements
- Submission to Nintendo for approval and distribution through the Nintendo eShop
- Compliance with Nintendo's submission and approval process

10. Code Style Guidelines and Organization

10.1. Code Style Guidelines

- Consistent naming conventions for variables, functions, and classes
- Use of comments to explain complex code segments
- Proper indentation and formatting for readability
- Adherence to Unity's best practices for scripting

10.2. Code Organization Overview (UML)

- UML diagram for visual representation of the project's structure and organization

10.3. Branching Policy

- Master branch for stable builds
- Develop branch for ongoing development
- Feature branches for specific tasks and bug fixes

11. Build Delivery Method and Version Management

11.1. Build Delivery Method

- Unity Cloud Build for automated builds and distribution
- Local builds for testing and debugging

11.2. Version List

- Version numbers following the semantic versioning format (major.minor.patch)

12. Delivery Platform and Requirements

- Delivery to the Nintendo Switch via the Nintendo eShop
- Compliance with Nintendo's submission and approval process

13. Extra Sections

13.1. Art and Audio Tools

- Adobe Photoshop for sprite and texture editing
- Audacity for audio editing and conversion

13.2. 3D Objects, Terrain, and Scene Management

- Not applicable, as the game is a 2D recreation

13.3. Use of Physics Engine

- Unity's 2D physics engine for movement and collision mechanics

13.4. Artificial Intelligence

- AI behavior scripts for enemy characters based on original game logic

13.5. Networking and Multiplayer

- Not applicable, as the game is single-player

14. Conclusion

This TDD has outlined the technical approach to creating a faithful, identical recreation of the original "The Legend of Zelda" game released on the NES in 1987 for the Nintendo Switch. By following the guidelines and recommendations in this document, the development team can create a high-quality game that will be enjoyed by players of all ages.

Document by: Ivan Peric, 2023