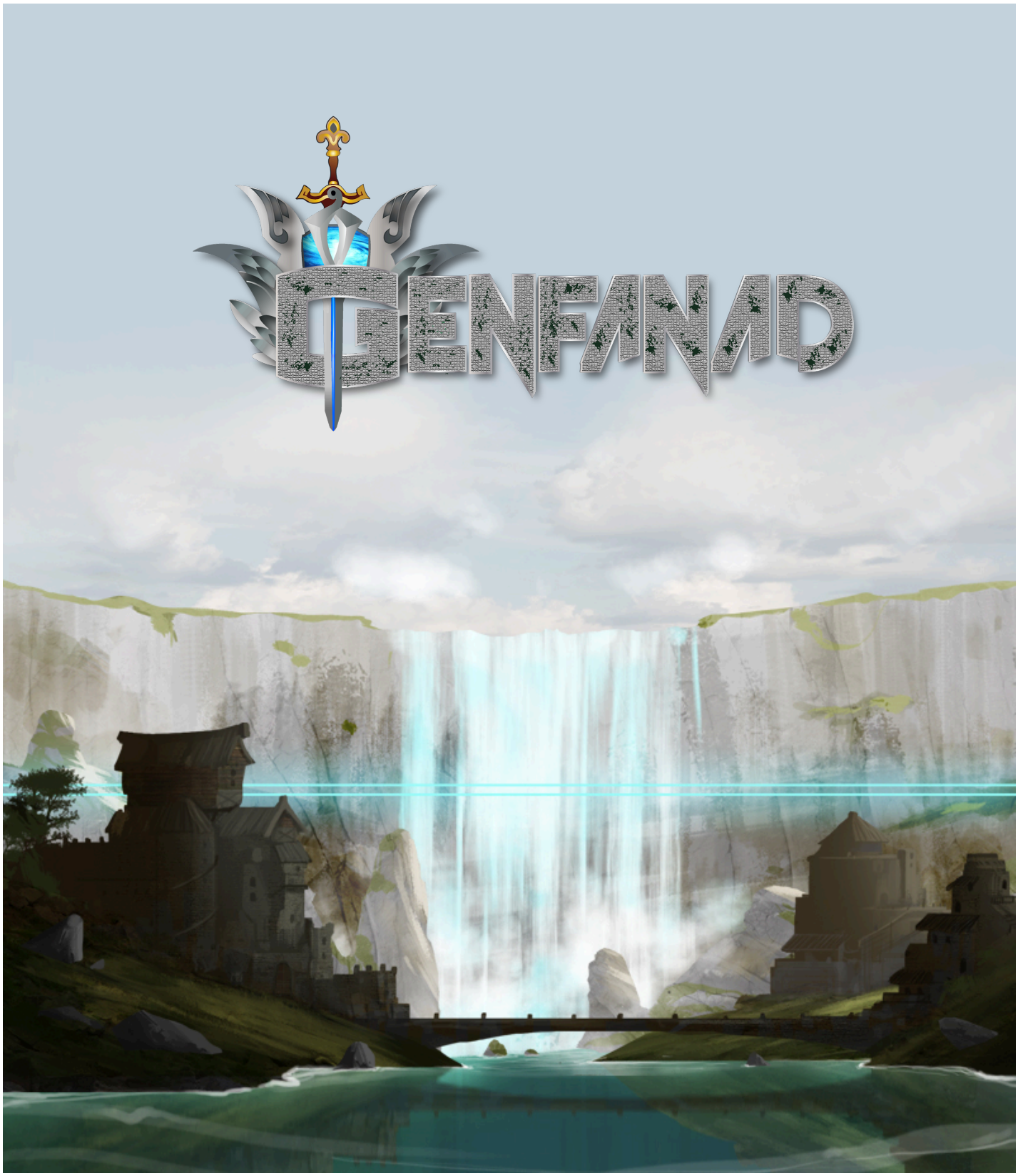

GENFANMD

Genfanad

Game Design Document

Discord https://discord.gg/44C6wmM

Overview

Genfanad is a small-scale multiplayer online RPG, with gameplay and art as a throwback to early 2000-era MMOs.

The two main concepts behind Genfanad are ‘Nostalgia’ and ‘Exploration’. The design is built around building new entertaining content with a low overhead cost.

Players create a character and level up a character’s skills, fight monsters, gather items, and explore a persistent game world. Progression can be observed by the player whenever they gain XP or level up, when they receive a new drop, when they defeat a monster they could not tackle before, or when they get access to new areas.

The setting is an anachronistic fantasy world, with the usual fantasy tropes of wizards, swordfighters, archers, et al. The setting is primarily a backdrop for holding the game mechanics together and does not take itself seriously, the writing primarily focused on humour through lampshading its own tropes or making references to other media.

Core Gameplay

The core gameplay of Genfanad is an RPG. Players control a character with a set of skills and an inventory.

Skills in Genfanad can only go up, and are used as both a gate, a scale, and an intrinsic reward. Certain content may have a skill restriction (you need to be a good lumberjack to cut this rare tree). Some content will scale (the better lumberjack you are the faster you can chop a tree). Skills are also how players can compete against each other (the top 10 lumberjacks in the game can be seen in the high scores). The XP requirement is exponential, so players will be able to get levels quickly early on but will require more investment to reach higher levels.

Combat is an activity that accounts for a subset of skills and allows players to fight NPCs or other players in a turn-based system. When combat is initiated, the participants are considered ‘locked’ and cannot enter combat with another participant. Every ‘round’, one participant is the attacker and one is the defender. Based on the relative stats of the participants, either a hit, miss, or critical hit are generated, which will reduce the defender’s HP. Once HP goes to 0, the participant is killed - for an NPC this will generate items from a drop table, for a player that will drop most of their items on the ground and give them the option to respawn.

Combat skills affect how players can fight. Support skills either don’t directly affect the combat calculations or are primarily used for non-combat means.

Gathering skills are where players interact with scenery and a given tool in order to spawn items for their use, trade, or sale. Mining is a gathering skill where players interact with rocks and a pickaxe to get ore. Logging is a gathering skill where players use a hatchet and a tree to get logs. Et cetera.

Processing skills take items that players have and transform them into new forms. Forging is a processing skill that turns ores into bars and eventually into smithed equipment. Cooking is a processing skill that takes raw food and turns it into edible food.

Content

One of the main core aspects of Genfanad is ‘exploration’. As such, there is an expectation of a large amount of content.

One consideration is to procedurally generate content. This can work wonderfully for exploration (such as in Minecraft) but procedurally generated dungeons and worlds tend to become very similar. Once a player understands the algorithm that generates the world, they will rarely have the element of surprise.

For Genfanad, the choice of building hand-crafted content but minimizing the cost (time and budget) of generating that content is key. One of my favorite memories of Runescape Classic growing up was the excitement of ‘update monday’ - each would add a new quest, location, or even potentially a new skill. That era of content expansion is the goal.

All content in Genfanad is implemented using standard JSON notations. No content should have custom code - neither server nor clientside - in order to reduce the risk of content becoming more difficult to develop over time.

All new features should be controlled by configuration rather than by code - so there shouldn’t be code that says ‘when you kill a cow increment the bovine menace flag’, but rather a ‘monster kill trigger’ feature that has a configuration that allows you to specify which monster type it applies to, which flag it modifies, and when the rule is eligible.

Locations

The Genfanad map is a set of interrelated layers, with two to start. The ‘world’ layer represents the world map, and the ‘dungeon’ layer is for exploring underground. Eventually there will be layers for higher floors and deeper dungeons.

Each layer consists of 128x128 square tiles. Each map tile connects seamlessly to each of four directions - when a player gets close enough to the boundary of their current tile, it will load the next one in that direction.

Tiles consist of terrain (color, elevation), scenery (up to one object per square), buildings (floor, walls, roof). Additionally, tiles can contain spawns for items, NPCs (monsters and characters), or unique models (a large model that represents something large in the area such as a bridge or a ship).

Creating a new piece of terrain is pretty straightforward and can be done with any existing tooling that exports a height and color map.

Buildings are defined using a modular ‘wall + roof’ system. New building styles just require creating a set of textures for the roof and individual wall types. Each square can have one of four wall types (plus-x, plus-y, and the two diagonals) and can optionally have a roof. This is repeated for up to 4 levels to make multi-story buildings.

Scenery is defined as standard OBJ or FBX models with a set of metadata that explains how it should be loaded in game - name, examine text, texture information, scale, position, and rotation parameters. When placed, it can be customized to a minor extent - such as a chair being rotated in the square that it is placed).

Playerkit

The ability of players to customize their own character - appearance and stats wise - is a significant appeal of the genre.

Players (and human NPCs) are assembled out of a 2D doll system with 43 possible layers in 5 directions (front, front-right, right, back-right, and back). The game will display the appropriate direction based on where the camera is and where that character is facing.

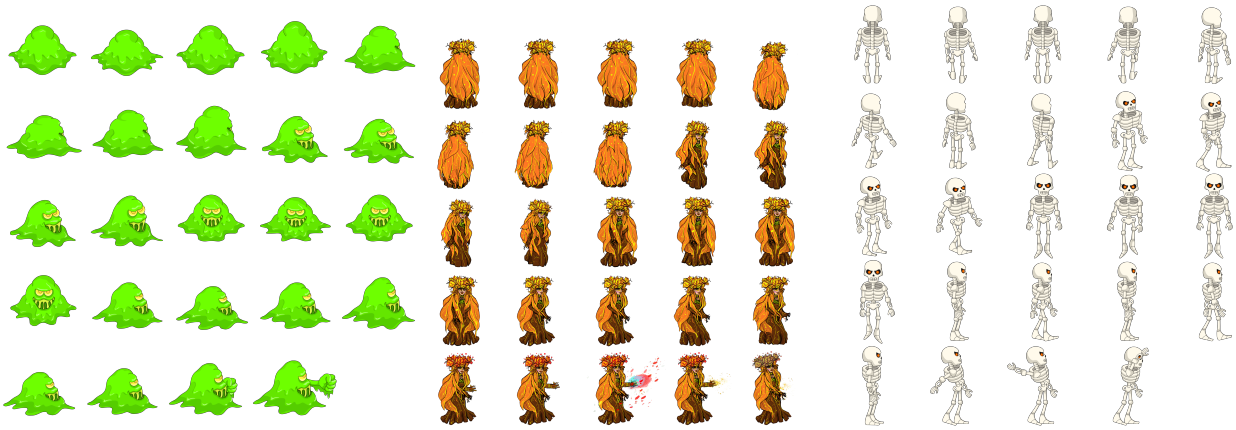
With this doll system, each layer in a given direction is reserved for a specific item, which will hopefully prevent item ordering or z-fighting issues. If a new helmet needs to be created for example, the layers for ‘head’ are all that will need to be added.

More information about the player kit (which now uses Spine animation) can be found here.

Any assets that are created for the playerkit are usable for all NPCs and players, and can be recolored dynamically as well. For example, the ‘short sword’ asset is reused for all the various metals that are created by tinting that layer. The default clothes and hairstyles are recolored for players as well.

Monsters

Monsters have the exact same 3D appearance as players but have dedicated spritesheets with fewer frames. The art style should be cartoonish, but a fully uniform style is not required - having unique-looking monsters is part of the appeal.



Once a spritesheet is created for a monster, its stats and droplist need to be decided.

Quests

This is not completed yet, but players will have a set of variables that can be used for various things. Quests are going to be driven through a conversation system with NPCs that work on sets of variables and flags.