Web Based Tactical Role Playing Game in JavaScript

June 2014

David Cilva
Paul Doyle
Stephen Hill
Rachel Judish

Abstract

The goal of this senior project was to develop a web-based game that emphasized narrative through gameplay within the context of a tactical RPG gameplay style. As the project developed, emphasis shifted from narrative to refining gameplay and making decisions within gameplay meaningful and satisfying. The project elaborates on the standard tactical RPG combat system, incorporating mechanics and ideas from various other games. This was done in order to make the play feel challenging and rewarding as well as to avoid some of the pitfalls that bog down existing games in this genre. Since work was concentrated on building the core gameplay, the final product was a functional game with a focus on a strong strategic core instead of progression and narrative content.

Introduction

Tactical RPGs are a subcategory of the role-playing game genre. They take a more strategic approach to the combat systems that are typical of turn-based RPGs. Many franchises have seen success and critical acclaim using this style of combat, including the Fire Emblem series and the Disgaea series. The goal of this project was to take cues from these games as well as others and implement a version that was easily accessible on the web, fun for new players, and capable of providing players with unique choices and interesting mechanics that made gameplay engaging and challenging.

The most important aspect of a game centered on strategy is choice. Providing players with a wealth of strategic options facilitates interesting gameplay. Game designers seek to limit complexity but maximize depth. The *complexity* of a game is simply the quantity and intricacy of the rules governing the game's systems. In order to play the game successfully, the player needs to understand all of these rules; thus the complexity of a game can create a more interesting experience but also creates a barrier to entry and forces players to spend more time learning rules rather than enjoying play. The *depth* of the game is the wealth of choices that emerge from this rule system. Games that are deep give players plenty to experiment with and plenty of emergent strategies to discover. Limiting complexity while maximizing depth is the cornerstone of building a game that embodies the "minute to learn, lifetime to master" ideal.

This project combined features found in other tactical RPGs as well as more novel elaborations in order to introduce as much depth as possible while still conforming to a style of game that's familiar to many players (effectively reducing its practical complexity). It was this goal that drove the choices made regarding which features to implement and how to structure systems like character attributes, weapons, and items.

Background

There are two aspects that define a tactical RPG. The first is the traditional role playing element, seen through character development through the users actions and choices, gaining experience and changing over time, and player customization. The second is a strong emphasis on strategy. Tactical RPGs focus more on creating a favorable situation through careful placement and planning then they do timing or reflexes. Tactical RPGs usually revolve around grid based movement and have relatively simple combat systems so the player can accurately plan situations to their advantage. Lastly, this genre of game tends to focus on single player linear gameplay. The game attempts to craft a strong narrative for the player to follow instead of having the player choosing his or her own path. While this lowers the player's sense of agency, it allows for a more finely crafted story and consistently smoother gameplay experience.

Technologies

The technologies used for this project can be broken into back-end and front-end technologies. The front-end includes the game logic (written in JavaScript, based on the enchant.js framework) and the web page on which the game is hosted (written in basic HTML and CSS). Communicating with the back-end is

accomplished through ajax (using JQuery functionality and sending data in JSON format). The back-end is built on the PHP framework Codeigniter.

Enchant.js

Enchant.js is a JavaScript framework for building games. It provides a system of default objects to abstract away the HTML5 canvas calls and facilitate easier management of game objects. It is organized around a Scene object, onto which a tree of child nodes is added. Nodes include Sprites and Labels, which can be extended with more functionality.

JQuery

JQuery is a JavaScript library whose purpose is to simplify JavaScript features such as Ajax and event handling. We decided to use JQuery because the library had very simple functions to help handle our JSON and HTTP requests.

JSON

JSON is a format for storing data. We decided to use JSON because it is easily parsed into JavaScript Objects, and it is a standard format for sending and receiving data using HTTP requests.

PHP

PHP is a server-side language that runs the back-end of many websites. We chose it as the language to run on our server because it was the language we were most familiar with. PHP easily handles GET and POST requests and works well with a mySQL database.

Codeigniter

Codeigniter is a PHP framework that follows a Model-View-Controller architecture. Codeigniter also includes many useful PHP libraries for assisting users with tasks such as sending emails and working with forms. Codeigniter provided a variety of utilities that simplified our project. It overwrites the standard “query-string” approach to URLs with a segment-based approach. This means that Codeigniter breaks up a URL and uses the segments as parameters. This made it possible to build a simple REST API to handle POST and GET requests easily. Additionally, Codeigniter has a database library that simplifies communication with a MySQL database. Finally, Codeigniter includes a library for creating user sessions. This way we could have a user be logged in and our system would create a session for them, effectively letting the server know which game information to send to the user.

mySQL

MySQL is a database management system written in C and C++. MySQL is controlled using the SQL language. We chose to use MySQL because the server we were using had MySQL already built in. Also, we are familiar with MySQL databases and SQL.

Tactical RPG Game

Basic Gameplay

The combat system follows the style of a tactical RPG (like Fire Emblem or Final Fantasy Tactics). Players control a team of units on a square grid-based map with the goal of eliminating the enemy units. As the player progresses, he or she will gradually gain access to more units. The player will then be forced to choose which units to deploy for each battle. Each unit has its own unique offensive and defensive capabilities as defined by attributes like attack, defense, magic, and agility.

The goal of this combat system is to begin with a turn-based system typical of classic RPGs, in which player and enemy units deal damage based on their particular abilities (expressed through numerical

attributes), and impart to it a strategic positioning element. This additional layer of decision-making provides players with a more dynamic battle that encourages careful management of units.

Game Design Decisions

Every game has trade-offs that are made during development. The choices that were made for this game were, for the most part, added in order to add depth to the game and shy away from the traditional grinding nature of role playing games.

- **Skills over attributes:** In traditional role playing games every character has numerous attributes that have to be micromanaged every time they level up. This means you have more control over your characters' development, but for the most part does not add to an engaging gameplay experience. Instead, skills were chosen as a reward for leveling up. Each skill is unique and adds an extra layer of complexity to every battle. By having the player chose each character's skill every battle, they are forced to begin planning ahead before the battle even starts, adding to the tactical nature of the game.
- **Command points over traditional movement:** In traditional tactical RPG's, every character can move and attack once. This implies that every action made is of the same importance, and leads to the awkward situation where the player feels obligated to move every character, even if it's not needed. Instead, a command point system was implemented. In this system, every action is given a cost, and the player is given a set amount of points and can decide how to spend them. The amount of points is always less than the maximum that could be spent, making the player choose how best to use his team. It also lets the developer put a higher cost on some actions than others.
- **Basic weapons for less complexity:** In traditional role playing games the player is constantly presented with new weapons that have slightly better stats than the one's they already have. While this gives the player a feeling of accomplishment, it does not add to gameplay and leads to micro-management. Because the weapons are given to the player at set times, it is no different than simply increasing their character's base attack. For this game, stat increases were chosen instead of creating hundreds of unique weapons, both to facilitate rapid development and to cut down on the inherent grindy nature of role playing games.
- **Mechanics over art:** Over the course of development, the focus of the game was on creating a strong foundation and solid gameplay elements. At every point a choice had to be made on whether to add another gameplay element or spend time creating a more aesthetically pleasing environment. Both because the development team lacked artistic backgrounds and because fun gameplay was generally valued over aesthetics, art was put in the backseat in favor of mechanics.

Meaningful Choices

The essential focus of this style of RPG combat is to provide players with a wealth of meaningful choices that make the game engaging and challenging. This means building a variety of units with unique capabilities. Some units deal physical damage while others deal magic damage; players who use damage types carefully and exploit enemy weaknesses will be more successful. Some units have more attack range than other units; players who plan attack ranges carefully can avoid being counterattacked. Each unit has an inventory and can be given healing items; players who choose carefully when to advance, when to retreat, when to attack, and when to heal may survive more challenging battles.

The combat system also adds a resource management element to moving units. Each turn, the player has a certain number of command points to spend on moving units. Moving and attacking each spend one point. In order to be successful, each turn the player must choose which units to move and which actions to take. Sometimes it may be more beneficial to move a badly-hurt unit out of harm's way than it is to advance and attack with a healthy unit.

The final decision-making component added to the game was a system for each unit to have a unique ability called a skill. Skills can heal allied units, damage enemy units from afar, or even allow units to move through walls or modify terrain. This gives players unique strategic opportunities and provides the

potential for a more rich and varied experience.

Making Units Unique

Each unit is provided with a set of values that define its particular strengths and weaknesses. These values are broken into six attributes:

- **Attack** - Determines the power of the unit's attack by applying a multiplier to the base damage of the equipped weapon.
- **Defense** - Applies a percent damage reduction to all incoming physical damage (based on attacking unit's Attack stat).
- **Magic** - Determines the power of a unit's attack when attacking with a weapon that deals magic damage. This functions similarly to the attack stat.
- **Resistance** - Applies a percent damage reduction to all incoming magic damage. Functions similarly to Defense.
- **Agility** - Provides a unit with a chance to dodge incoming attacks and a chance to strike with a critical hit, which deals double damage. Units with high Agility will dodge and deal critical hits more often.
- **Focus** - Decreases the chances of enemy units dodging or dealing critical hits. Focus functions as a counter to agility.

Each unit has a particular affinity for each attribute. Some units may have high Defense and Resistance values but relatively weak Attack values. Others may have moderate Attack values and high Agility values, making them difficult targets to hit. By providing a variety of combinations, developers can populate the game with a diverse cast of units, each of which is particularly effective in certain situations. Making units feel unique on the battlefield may additionally give them more strongly defined identities in the game's narrative as well.

Skills

After each unit moves, it has the option to perform a skill. Each skill has different effects, ranging from damaging enemy units to manipulating unit placement on the map. Skills augment each character's specific strengths and help to offset the units' weaknesses. For example, Michael has a relatively low defense stat. As discussed above, this means that he will take more damage per hit than the other characters. However, he can use the Stealth skill to avoid getting hit at all. Each unit has a unique skill:

- **Heal** - Heal a friendly unit by 50% of its max health. Used by Paul.
- **Smite** - Reduce the health of a targeted enemy by 50% of its current health. Used by David.
- **Swap** - Trade places with any unit within range. Used by Rachel.
- **Teleport** - Move to any spot on the map within range. Used by Stephen.
- **Stealth** - Become invisible until the next turn. Used by Michael.
- **Healing Pact** - Sacrifice oneself to fully heal a friendly unit. Used by Mandy.

Skills add more choices and therefore more strategy to the game. Each skill costs a different amount of command points, so the player may choose to use a few skills at the cost of not being able to move some units. A few well-used skills have the potential to turn the tide of the game to the player's favor. In addition, the uniqueness of the skills increases the uniqueness of the characters, which makes them less expendable and increases the potential for building narration around the characters. For example, Michael, with high attack, low defense, and the Stealth skill, could easily be a ninja or some other stealthy character in a narrative.

Saving Game State

This framework provides the ability to save a game's state onto a server that can be accessed from any computer with an internet connection to allow a user to play their game from the same state that they left the game at previously. The first step involves allowing users to have an account. A user is able to register their email and a password to create an account specifically for the game. The system also allows a means of login so that the system is aware of which user's data should be used. The next step involves allowing a logged-in user to transfer the state of their game to the server where the data will be stored under their email. This is done through means of "saving the game". The last step in this process involves a logged-in user to load their game's state from the database to their game where they can play their game from where they left off.

Design and Implementation

Design choices

The first major design choice was the platform to build the game on. We opted for a browser-based javascript game because it allows access from most computers without requiring any sort of installation. It also synergizes better with web features like game state saving than formats like flash.

The next design choice was selecting a game design library to use. We chose enchant.js because it provided us with many valuable tools for abstracting classes and creating the game structure without restricting us. If there were any aspects that we found lacking or unacceptable we could simply add onto them or ignore them completely and use our own implementation. In addition, enchant.js was adaptable enough to fit the design we had for the game, unlike several other Javascript game engines that were designed for use in a specific type of game, such as voxel.js, or required webgl to use, like three.js. Enchant was also familiar to a few professors who recommended it.

PHP was our choice for the backend language because it is simple to set up and use for a project of this scale. Also, we installed the codeigniter framework to make REST API calls simple to create and use. The design of the actual game was chosen before any code was written simply because it seemed like the most efficient way to encapsulate the different aspects of the game while keeping their interactions meaningful and logical (See figure 1).

Due to time constraints and the amount of work involved in making a good game, we had to prioritize our goals for the game and determine which aspects were the most important. We decided that the game's combat system was the most important to making the game fun and that narrative elements like story, sound effects, and graphics were less important. Based on this decision, we chose to use free, easily available graphics instead of making our own, opting to spend the bulk of the development time on combat, characters, and maps.

Software Diagrams

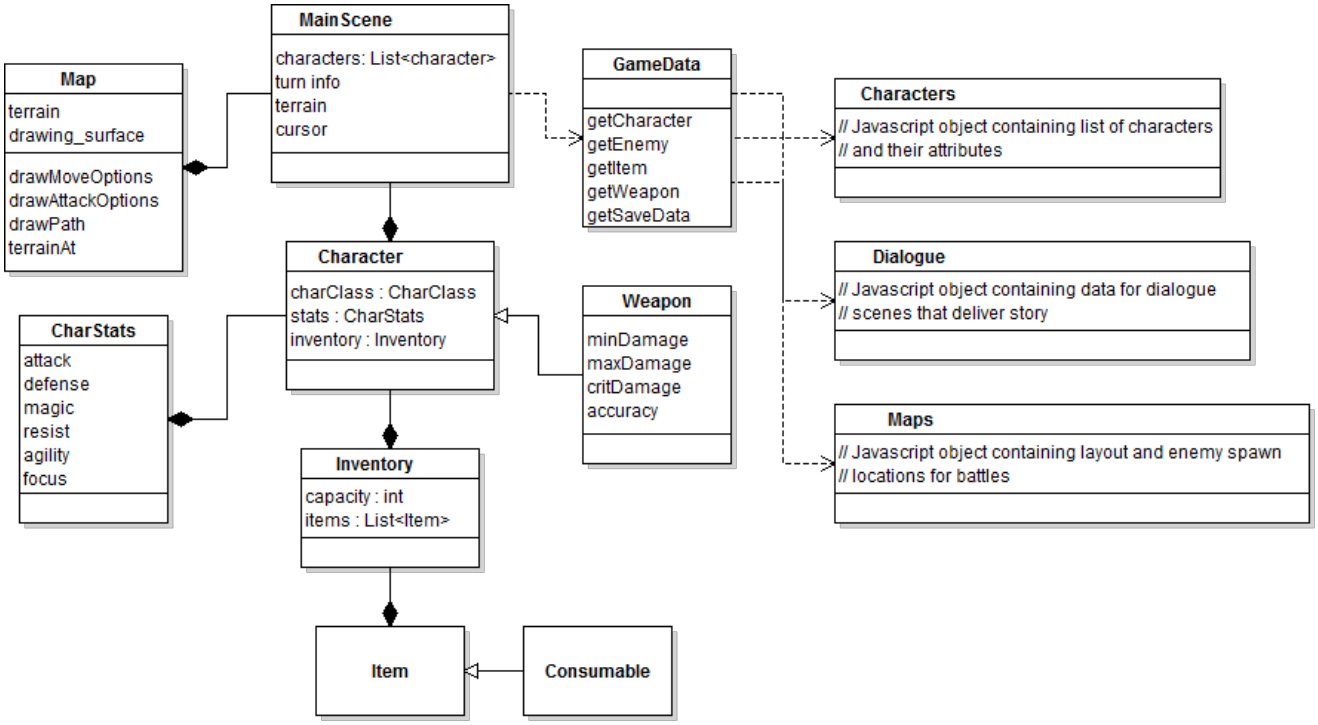


Figure 1: Game design UML diagram

Assessment

Playtesting

In order to assess the game's balance and depth, three prototypical battles were developed and a cast of six characters was included. Each of these characters had a unique skill with various strategic implications.

Player Feedback

Assessing the game's success at engaging player's and creating meaningful decisions required surveying those who played the game. Players were asked to fill out the following survey:

Multiple Choice (strongly disagree 1 - 2 - 3 - 4 - 5 - strongly agree)

1. I am familiar with tactical RPGs
2. I found the gameplay engaging.
3. I thought the levels were balanced (not too easy or too hard).
4. I understand how the stats and command points work.
5. I would willingly play this again.

Short answers

6. What did you like about the combat system?
7. What didn't you like about the combat system?
8. What would you want to see more of in the game?
9. Were there any strategies that worked every time?
10. What role did luck play in your interactions with enemy units? How important were critical hits and

evasion?

11. How did the different units' unique skills affect your strategy?
12. Did you encounter any problems? If so, please describe them in detail.

Out of 11 playtesters who provided specific feedback, an average score of 3.8 out of 5 was given for whether players agreed that the gameplay was engaging. An average score of 2.7 was given for whether playtesters thought that the levels were balanced. Players were divided in regards to the balance and difficulty. Players who felt it was too hard found that their characters died quickly and did not see obvious ways to account for being outnumbered by enemies; those who found it too easy had discovered certain strategies that were too universally successful and often exploited the limitations of the enemy AI. Players that had not played a tactical RPG before tended to think that the levels were unbalanced. One way to remedy this would be to include a sort of tutorial in order to better inform players of the options and strategies at their disposal. This would be useful since many playtesters did not have a clear understanding of how character attributes affected damage dealt in battle and the command points system worked. Even with some of these difficulties, an average score of 3.4 was given for whether players would play the game again. We also saw an average score of 3.8 for whether playtesters were familiar with tactical RPGs.

Several playtesters commented that the use of ranged characters, ones that could attack from 2 grid squares away, were too powerful. Many playtesters employed similar strategies in which they would “hide” their units behind an obstacle and deal damage using their ranged characters. This was especially the case with the “smite” skill which had even greater range of attack. Since a game such as this should come down to choice, we as the developers should work against one tactic being more effective than possible others. One way that could alleviate this issue would be to create enemy units that could pass over obstacles, preventing the player from staying in one spot and attacking from a distance.

Playtesters also commented on certain skills. One character had a skill that made him invisible to enemies. A few players discovered that the AI still treated the square he stood in as occupied (even if they wouldn’t attack him or advance on him) and used the character as a movable block to keep enemies from getting through small spaces. Maps that have choke points only one square wide could be indefinitely blocked, which makes the skill too powerful or means that map design needs to change. On the other hand, one character had a skill that would heal another character at the cost of all her own health. This skill was disliked and rarely used by most of the playtesters simply because it was not worth sacrificing a unit just to heal another. With this information, a next step would be to balance out the skills so that there is no clear “best skill”. This would allow players to develop varying strategies and styles while playing our game.

Conclusion

Over the course of development, the direction of the project shifted from creating an entire game to creating a more concentrated experience. Ultimately, however, the completed project managed to provide interesting play and unique choices to players both familiar and unfamiliar with the genre. Given that playtesting only saw one iteration, the feedback provided reveals a lot of potential improvements, modifications, and balance tweaks that could make the game significantly better. Regardless, the initial goal of providing players with unique strategies showed in the way that each of the players who provided feedback seemed to be employing different tactics.

Future Work

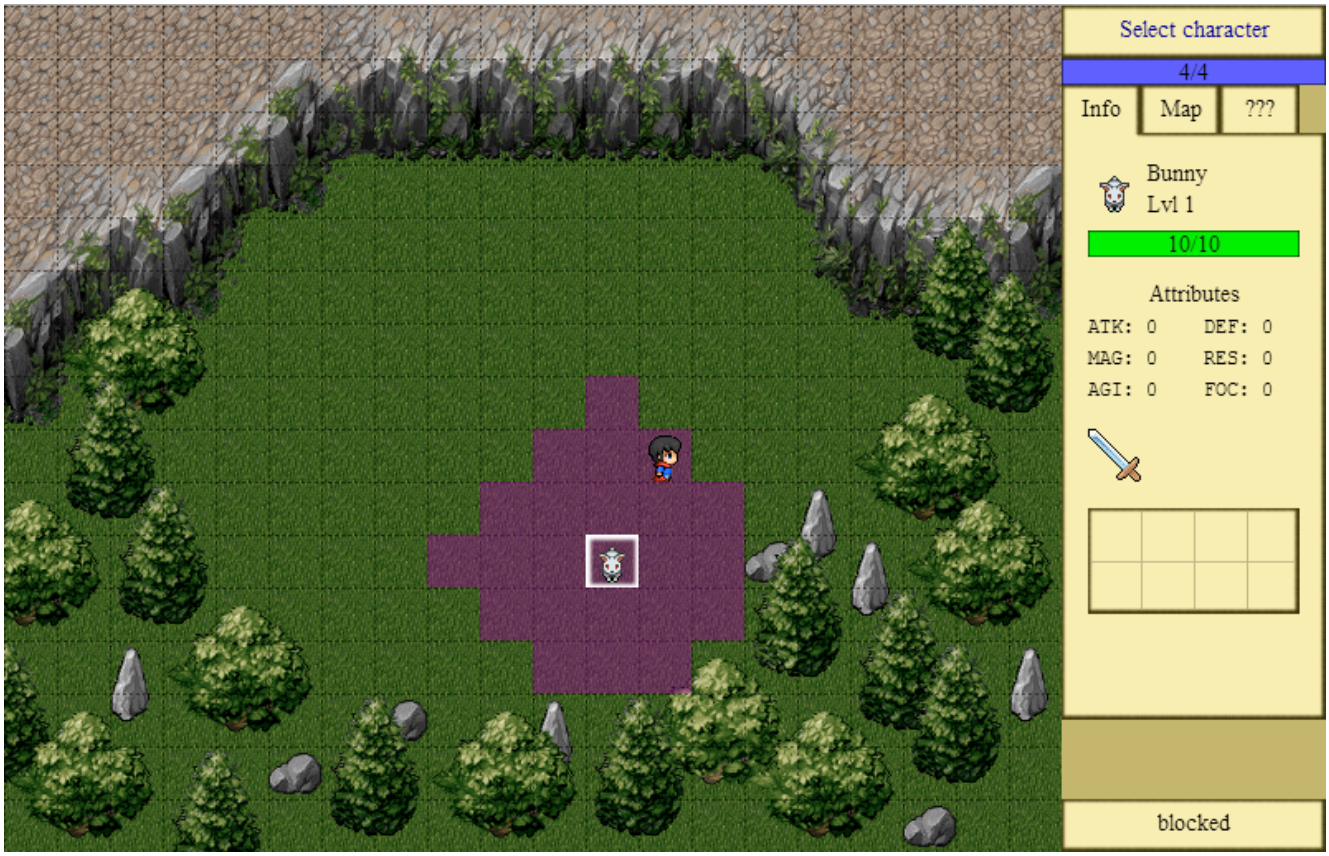
The player feedback received indicates several areas where balance could be improved. Of the characters provided, certain characters’ skills saw much more use and appeared to have much more strategic value than others. This could be improved by changing the ranges and effects of these skills to better align with the command point values assigned to them.

Additionally, the game is extremely short and concentrated with no narrative content whatsoever. Adding story, weapons, maps, enemies, and an even larger cast of characters would give this game the potential to capture and hold the attention of players across play sessions, and even entice them to return after they've closed the game. The best games weave a tapestry of plot, characterization, gameplay, and experiential novelty in order to fully captivate players and keep them returning; while the fighting system requires further iteration to fine-tune and balance, there is potential for future work in the vein of content creation. A compelling context to the game and a larger and more fleshed out pool of game content could help make the game significantly more entertaining.

Ultimately, players seemed divided on the issue of whether or not the character attributes and command point system were easily understood. Since all of the game's tutorials were provided simply by descriptions on the website, players found difficulty forcing themselves to read paragraphs about the game before playing. As game developers know, the best games have tutorials that are helpful and informative without being tedious and requiring players to read paragraphs of text. Good tutorials *show* players how the game works and give them tools to experiment safely while they learn the system.

References

- w3school.com - JavaScript programming documentation and tutorials
- StackOverflow.com - JavaScript programming advice
- enchantjs.com - JavaScript game development framework
- Foaad Kohsmood - Advice on working with enchant.js
- HelioHost - Hosting and database setup tools
- LostGarden - Free-to-use game graphics


Select character
4/4
Info
Map
???
Bunny
Lvl 1
10/10
Attributes
ATK: 0
DEF: 0
MAG: 0
RES: 0
AGI: 0
FOC: 0
blocked


Skip >>
Father
Well, suit yourself, for now. If the wind doesn't return, we won't even leave the docks today anyways.