Nancy Wang
Sina MohammadiNiyaki
Derek Huang
Duy Nguyen

Project Implementation Report

Overview of the Implementation Approach

Our initial goal was ambitious, targeting the implementation of player movement and map drawing as key features in our game. We envisioned two major milestones: the first being the development of a navigable world with player movement, and the second, the completion of a fully implemented game. However, as the project progressed, we encountered various challenges and time constraints that necessitated a shift in our approach.

To navigate these challenges, we adopted a scrum-like methodology, particularly in the latter half of the project's second phase. This approach, characterized by daily meetings, was instrumental in maintaining a steady development pace and enhancing communication within our small team. The use of a backlog for task distribution was a crucial element in effectively managing our workload and ensuring that each team member was clear on their responsibilities.

In terms of technical development, we started with basic game development tutorials on YouTube, which laid the groundwork for our project. Once we had a handle on the game engine, collision detection, and movement mechanics, we expanded our focus to more complex aspects such as the user interface, object interaction, and enemy behavior. Continuous testing was a critical part of our process, allowing us to verify the functionality of each feature as it was developed. This rigorous approach to testing and development helped us maintain steady progress despite initial setbacks.

Adjustments and Modifications

Our initial design, as outlined in the class diagrams and use cases from Phase 1, underwent several modifications. The fast-paced nature of the project and the evolving understanding of our capabilities led us to prioritize core functionalities over additional features. For example, The decision to focus more on essential elements like player movement and basic map functionalities was a strategic shift to ensure project feasibility within the given timeframe.

In the course of our project, we made several key adjustments to the initial design, as depicted in our phase 1 class diagrams and use cases. One of the most significant changes was the simplification of our class structure. We moved away from using separate inanimate and animate object classes. Instead, we streamlined our approach by introducing a player class and a zombie class, both categorized as entities, and created distinct object classes for items that could be picked up. This modification made our code simpler and more intuitive.

Another major adjustment was the introduction of a 'gamepanel' class, which became a central hub for managing all aspects of the game. This approach allowed for a more consolidated and efficient management system. For the game's map design, we opted out of creating a specific barrier class. Rather, we used tile classes, each with its own properties, managed by a 'tilemanager'. This approach provided us with more versatility in map creation and a cleaner, more modular codebase.

Nancy Wang
Sina MohammadiNiyaki
Derek Huang
Duy Nguyen

Additionally, we tackled the challenge of character animation. Initially unsure of the best method, we eventually found a way to animate characters directly within their existing class, eliminating the need for additional classes solely for animation purposes. This resulted in a more integrated and efficient approach to character design.

Management Process and Division of Roles

During this phase of our project, we employed a highly collaborative and flexible management approach. Instead of assigning rigid roles, we collectively engaged with a shared backlog from which each team member could choose tasks. This method was grounded in the principle of contributing where each individual felt most skilled and confident. Such a strategy enabled us to quickly adapt to evolving project needs and effectively tackle any challenges encountered along the way. By emphasizing flexibility and leveraging the diverse strengths of our team, we were able to navigate the complexities of development with agility and cooperation.

External Libraries Used

We relied primarily on standard Java libraries, with a specific emphasis on Java Swing and Java AWT for GUI development. These libraries were chosen for their robustness, compatibility, and the extensive documentation available, which was crucial for troubleshooting and efficient development.

Code Quality Enhancement Measures

To ensure high code quality in our game development project, we adopted several key practices. We engaged in external testing, inviting friends and family to test the game, which provided us with diverse feedback and insights. Internally, we focused on thorough documentation, utilizing Javadoc comments. This not only improved understanding and collaboration within our team but also established a foundation for the future scalability of our project. While we had ambitions to organize our code into packages for enhanced modularity, time constraints prevented us from doing so. In the future, we hope to be able to refactor our project to organize all the files into proper packages. However, we emphasized clear, descriptive naming conventions and the use of helper functions to maintain code clarity and efficiency. These combined efforts were instrumental in elevating the overall quality and maintainability of our code.

Challenges Faced

In this phase, our team encountered several significant challenges. One of the most daunting was dealing with merge conflicts, particularly when multiple team members committed work simultaneously. This experience highlighted the need for more effective use of branches in GitHub to prevent such issues. Additionally, as many of us were using GitHub and its commands in a team setting for the first time, there was a slight learning curve, especially in terms of bug identification and resolution. We realised how powerful Github was for version control and team projects.

The project's scale initially posed a challenge; determining where to start was difficult, but we found that once coding began, progress accelerated. Time constraints also presented a significant hurdle, restricting

our ability to implement desired features such as varying difficulty levels, attacking zombies, and extensive code refactoring to enhance quality. This is something that we hope to be able to implement in the future.

From a technical standpoint, developing a sophisticated AI for the zombie character proved challenging. While we managed to achieve decent capabilities, the AI's interaction with obstacles like walls still needs improvement, which we aim to address in future iterations. Additionally, creating a custom map with a diverse range of tiles was a complex task, requiring significant effort and creativity.

Lessons Learned

Through this project, our team gained valuable lessons that extended beyond the technical aspects of game development. A critical takeaway was the importance of gameplay balancing. We learned to fine-tune elements like zombie speed and placement, which are crucial for creating an engaging and challenging player experience. Our approach to teamwork evolved significantly; we discovered the effectiveness of dividing work based on individual skills and strengths, leading to more efficient and quality outputs.

Regular meetings emerged as a vital component of our project management. They not only facilitated better communication and problem-solving but also helped in keeping the team aligned with project goals and deadlines. Additionally, we learned the importance of flexibility in project management. Adapting to changing requirements and unforeseen challenges was crucial for keeping the project on track.

We also realized the value of iterative development and continuous feedback. Implementing features in small increments and seeking regular feedback, both internally and from external testers, greatly enhanced the quality and relevance of our game. Another lesson was the significance of version control proficiency, particularly in a collaborative environment. Understanding tools like Git not only reduced technical hurdles but also streamlined our workflow.

Finally, this project taught us the necessity of balancing ambition with realistic goal-setting. While it's important to aim high, acknowledging and working within time and resource constraints is essential for successful project completion.

Conclusion

This phase of the project, though challenging, was a significant learning curve for the team. It not only enhanced our technical skills but also our ability to work collaboratively under tight deadlines. We look forward to applying these learnings in future phases of the project and further refining our game.