Building a webbased text RPG for prospective students

Isabella Hofstede & Margriet van der Molen

2024-01-20

Isabella Hofstede
Margriet van der Molen
BFV3
2024-01-20
Marcel Kempenaar

Building a webbased text RPG for prospective students

Building a webbased text RPG, using Java, Javascript and HTMl.

Student: Isabella Hofstede
Student: Margriet van der Molen
Bio-informatica
Life science en technology
Lecturer: Marcel Kempenaar
Date: 2024-01-20

Contents

Summary

Teaching, seen by many as a challenging endeavor, becomes even more challenging when attempting to convey a big portion of academic content within a limited timeframe. Such as during an Open Day, where students pick their next field of study based on mere minutes of interaction. This documentation outlines the process involved in the creation of a web-based text Role-Playing Game (RPG) intended for prospective students during the "open day" event. Made to be accessible to participants with varying levels of expertise, the game circumvents the necessity for coding knowledge while assuming a basic understanding of biology.The use of a text-based gaming format was based on the premise that user engagement is higher when interacting with a responsive environment rather than passively consuming information like in a textbook. The design prioritizes a seamless user experience, negating the need for users to consult manuals or engage with a command line. To maintain accessibility, the coding aspect of the workshop has been represented as 'graphical coding,' meaning users do not need to generate their own code.

Although the project exhibits certain imperfections in its final version, the majority of its design components have proven successful. A narrative was written, which was tied to three mini-games. The text-based game guides users through the storyline, transitioning between mini-games and text interactions. These mini-games contain content that's centered around biology and coding. A notable area for improvement lies in improving the visual elements related to user-input responses within the game. This is among one of the many development aspects that can be improved to add to user engagement. Furthermore, transmitting information from the text-based game to the mini-games relies on a GET-request, allowing users to edit actual scores. Addressing this "looping" issue should be prioritized in future iterations of the project.

In summary, this project has delivered a gaming experience functioning as an instructive workshop to introduce new students to the realm of bioinformatics. While achieving its primary objectives, there remains room for improvement. This is particularly in refining visual elements and optimizing data transfer mechanisms between game segments.

Material and Methods

Tools and Software

All code and information pertaining to this project can be found within the github repository for this paper. For this project there was no pre-existing data to be used and thus everything was made from scratch. There are however multiple pre-exiting tools and frameworks that were used for this project. These tools consisted of: Thymeleaf, to create HTML templates for the user interface [1]. JavaScript, for the frontend logic which is responsible for managing game states, handling user choices, and dynamically updating the content of the webpage. Additionally, multiple javascript files were created to handle the logic of the mini-games, these will be discussed in further detail down the line.

This project was made in IntelliJ IDEA 2023.2. IDE [3]. Java served as the primary backend programming language for this project [2]. The web configuration and web requests were handled through java servlets. The project was managed through Gradle v8.5 [4]. Lastly there is the use of Apache Tomcat v9.0.85, the webserver used to deploy the java web applications, handle HTTP requests and hosting server-side logic [5]. All versions of the tools and software are available in the recourses section (link to resources). (Table with versies)

Project Structure

The struture of the project is as follows: The folder config contains configuration files related to the project, settings and configurations for the backend. The folder for servlets houses Java servlets, which handle HTTP requests and responses. These servlets are the backend controllers responsible for managing the game's logic and interactions. The images folder stores image files used in the project and contains graphics or icons related to the game. The js folder contains JavaScript files, for frontend logic. These files are responsible for handling user interactions, game states, and other frontend functionalities. Lastly, the folder templates holds HTML templates, implemented using Thymeleaf. These templates define the structure and layout of the web pages that make up the user interface. For the hirarchial structure, refer to the README file linked in the github.

Frontend Development

Web-Based Text RPG

The frontend development of the text-based RPG was implemented using HTML, javascript and Bootstrap. These maintain the core logic for managing game states and displaying text nodes. The player is guided through a text-based adventure game and solves minigames along the way. At the end of the game the amount of errors they made determines whether the player gets a good or bad ending.

Text Node Logic

The game's logic is determined by the textgame.js file. The next text node is based on the player's choices and current gamestate. If for example a player obtained a an item, it will be set as a state. Based on that state, the text nodes in the sequence will guide the player towards a minigame that will let them play the minigame associated with that specific item. These nodes can be expanded upon and additional minigames can be added seperately.

Game Integration

Each text node has an ID and a sequence in which those ID's are linked. Whenever a user picks a node that relates to the next game, the user get redirected to a page that hosts the minigame. Whenever the user finished the game, they get redirected to the next text node in the sequence. When a user makes a certain amount of errors, the game can show a fail-state. When this fail-state is achieved, the user can try again. The games that were included in the current version are memorygame.js, which involves players matching images of body cells; drag_and_drop_event.js, which is a drag and drop game which has users provide the right

answer without having to type anything; fill_in_blanks.js, which has users fill in textboxes with answers to questions.

HTML and bootstrap

All of the code that is related to formatting is present in the HTML template files. For the purpose of streamlining the code, bootstap was used as a framwork for all the HTML pages.

Backend Development

Servlets

The backend consists of several java servlets that present the webpages to the user in order of which they should appear. These servlets consist of: CoverServlet, which shows the welcome page; DragDropServlet, which shows the drag-and-drop minigame; FillinServlet, which shows the fill-in-the-blanks minigame; MemoryGameServlet, which shows the memory game and TextGameServlet, which shows the text node system.

References

[1] Thymeleaf: *Download Thymeleaf*, Download Thymeleaf, Retrieved from https://www.thymeleaf.org/download.html on 20-01-2024

[2] Java, Oracle: *Download Java*, Download Java, Retrieved from https://www.java.com/nl/download/ on 20-01-2024

[3] IntelliJ IDEA: *Download IntelliJ*, Download IntelliJ IDEA, Retrieved from https://www.jetbrains.com/idea/download/ on 20-01-2024

[4] Gradle: *Download Gradle*, Download Gradle, Retrieved from https://gradle.org/releases/ on 20-01-2024

[5] Tomcat: *Download Tomcat*, Download Tomcat, Retrieved from https://tomcat.apache.org/ on 20-01-2024