Braille Learning Tool

Authoring Requirements Document

Team 4
Allauddin, Qassim
Li, Derek
Mohamed, Yassin
Solovey, Artem

EECS 2311 - Software Development Project (Winter 2017)
Instructor: Bil Tzerpos

Table of Contents

1 INTRODUCTION 3
1.1 PURPOSE 3
1.2 CUSTOMER'S NEEDS 3
2 AUTHORING FEATURES 4
2.1 NEW SCENARIO AND FILE EXPLORING 4
2.2 ADDING NODES FOR FUNCTIONALITY 5
3 ACCEPTANCE CASES 6

1 Introduction

This section of the document gives an overview for the authoring app of the project. It describes the needs of the customer and their required features.

1.1 Purpose

The purpose of this project is to design a learning tool for children to read braille. The player is a piece of the software that allows the user of the braille learning tool to play through story scenarios. This document is meant to provide a detailed description of the features of the authoring and is for people who intend to create a similar braille learning tool that implements our API.

1.2 Customer's Needs

The Authoring App is a program that will help educators develop story scenarios for the API. Since the user of this program likely has no background knowledge on programming, the final product will be used with a graphical user interface and is going to hide the technical and coding aspects of the Player App and Simulator App.

The Authoring App provides a graphical interface that allows users to build story scenarios. It will also have the functionality to save, edit, and load story scenarios. Previously, the scenario would be created in a text file that is manually created by the user. The text file would contain tags that controls the Player's functionality. The Player would then read the text file and play the scenario accordingly.

The goal of the Authoring App is, not only to hide the technical aspects, to abstract the use of tags when creating the scenario. This will make the software more convenient to use for the end user because it removes the complexity and error of having tags inputted manually.

It was also required that Java is used as the programming language of the Authoring App.

2 Authoring Features

The Authoring App is a very intuitive software for building scenario files for the Player App. The graphical user interface shows the branches to the story in the form of boxes and arrows. The boxes represent an action (i.e. speech, user inputs) and the arrows represent the story flow.

2.1 New Scenario and File Exploring

This section will provide a description of the functionality of the buttons that are used for creating, saving, and loading new scenarios.

Table 2.1 – Description of the functionality of the buttons that manage saving and loading scenarios.

Button	Function
New Scenario	Creates a new scenario with an initial root node. The user must input the number of braille cells and buttons.
Save Scenario	Converts scenario displayed on the graphical user interface into a .txt file and write it to the disk.
Browse Scenario	Converts a .txt file into a scenario and is displayed on the graphical user interface.

2.2 Adding Nodes for Functionality

This section will provide a description of the functionality of the buttons that are used for adding story elements to the scenario.

Table 2.2 - Description of the functionality of the buttons that manage editing scenarios.

Button	Function
Add Pause Node	Forces the story to pause for a certain number of seconds. The user must input the number of seconds in the dialogue window.
Add Set-Voice Node	Changes the sound of the voice to one in the built-in voice library. The user inputs an integer between 1-4 to choose between the voices. The GUI window will give more information.
Add Display-String Node	Changes the braille cells to an inputted string. The user must input the string to be displayed.
Add Two Branches	Creates a branching story. The user can branch the story into two paths. Eventually, they will be back on the main path.
Add Play-Audio Node	Opens and plays the selected audio file. The user must browse for and select the audio file.
Add Clear-All Node	Clears all braille cells.
Add Clear-Cell Node	Clears a single braille cell. The user must input the index of the braille cell that is going to be cleared.
Add Set-Pins Node	Sets certain pins to be raised on a braille cell. The user must provide the binary string to read from.
Add Display-Character Node	Sets a selected braille cell to a certain character. The user must provide the index of the braille cell and the character.
Add Text-To-Speech Node	Allows user to enter text to be read by the text-to-speech engine.

3 Acceptance Cases

This section gives an example of how each feature should be used. In the following table only one example is given for each feature. The reason this is sufficient is because this section is meant to provide one instance of how the feature can be used. For more extensive cases, the reader should see the Testing Document.

Table 3.1 – Acceptance cases for each feature of the simulator.

Feature	Input	Output
New Scenario	`2 2`	[in scenario file] Cells 2 Button 2
Save Scenario	`Scenario as Test1.txt.`	File named Test1.txt.
Browse Scenario	`Test2.txt`	Scenario in Test2.txt.
Pause	`4`	[in scenario file] /~pause:4
Display String	`snake`	[in scenario file] /~disp-string:snake
Two Branches	No input required.	[in scenario file] /~skip-button:0 Branch1 /~skip-button:1 Branch2 /~user-input /~Branch1 … /~skip:mainBranch /~Branch2 … /~skip:mainBranch /~mainBranch
Play Audio	`Test3.mp3`	[in scenario file] /~sound:Test3.mp3
Clear All	No input required.	[in scenario file] /~disp-clearAll
Clear Cell	`1`	[in scenario file] /~disp-clear-cell:1

Set Pins	`11001011`	[in scenario file] /~disp-cell-pins:11001010
Display Character	`1 p`	[in scenario file] /~disp-cell-char:1 p
Text-To-Speech	`Hello`	“Hello” outputted as sound.