Music Times: A Music Learning Game

Emily Woo

Computer Science Department
California Polytechnic State University, San Luis Obispo

Project Advisor: Zoë Wood

March 2017
Abstract

*Music Times* is a two dimensional educational video game with the purpose of gamifying learning of musical concepts. It has elements of adventure and visual novel games, and incentivizes the player to learn music to explore new levels. It is developed in the Unity game engine, scripted in C#, and targeted for mobile devices. It has six working levels: three lesson levels and three corresponding challenge levels. Each level contains slight differences in visual and aural feel.

Introduction

Learning to read and understand music is an undertaking that can be intimidating to those outside the music world. Several methods of learning exist: taking in-person classes, self-teaching with educational materials, and consuming music education media such as television or games. On the media side, plenty of music learning games exist, but many are limited in the gameplay experience. *Music Times* aims to gamify the music learning experience, and create an adventure that unfolds as the player gains musical knowledge.

Previous Work

Music education games currently exist on many platforms and devices. The games vary greatly in style and material. Some representative examples are Flowkey[1], Solfa[2], and Melody Cats[3]. Flowkey is a comprehensive web and mobile application aimed to teach users the basics of piano playing. It uses videos and interactive exercises that require the user to play on a keyboard. This limits music learning to piano concepts, and requires the user to acquire a keyboard of some kind. Solfa, available on many platforms, works like a flashcard quiz, by showing pictures of a note and having the user press the correct key on a keyboard on the screen. There are no lessons, so it requires prior learning. Melody Cats is a series of mobile applications that teaches music reading through games where the user must “collect” the correct note out of a series of notes.

These existing games have several elements such as teaching of music comprehension, repeated exercises, and fun gameplay. However, few of them have all of these elements. Some,
like Flowkey and Solfa, are exercise-heavy with few game elements. Melody Cats comes closest to providing a game-like experience, but in doing so, it provides very minimal teaching and no explanation of the music concepts its games are based on.

**Music Times** takes elements of these solutions, and combines them into an engaging, guided, visual novel style game. It teaches music concepts, uses exercises to test these concepts, and then rewards the player with fun graphics and advancement to the next level.

**Algorithm**

**Overview of Solution**

*Music Times* is a fun, playcentric take on the music education game. It is a text-heavy game that teaches music concepts, tests these concepts in the forms of “challenges,” and rewards the player with a fun game experience. The game is developed in the Unity 5.4 game engine, and is touch-compatible. It can port to multiple platforms, but was developed with Android devices in mind, as it was tested on an Android tablet.

**System Design**

Because *Music Times* is developed in Unity, a lot of its functionality is handled in the form of scripts. Scripts were created to handle different pieces of functionality. These different areas of functionality include touch controls, character movement, textbox display, loading of the next level, image display, audio play, level fadeout, and “challenge” management. Each of these pieces of functionality have their own scripts, which can then be added to GameObjects placed within each Unity level.

One example of script usage is the movement of the player on screen. The sprite of the main character, placed within the level, has a CharacterControls script component, which controls the character’s movement. The level also contains an arrow pad with left and right arrows, each controlled by a Touch script component. The Touch script will communicate touches to the left and right arrows to the character sprite’s CharacterControls script, which will then translate them into moving the character left or right within the level. The
CharacterControls component also determines whether to play an idle or walking animation for the character.

“Lessons” and “Challenges”

Music Times is split into a series of levels. There are two types of levels: “lessons” and “challenges.” “Lessons” are levels in which the user must walk to a piano, which will trigger a music lesson that appears within a textbox along the bottom of the screen. The text is accompanied by graphics that appear in the sky above the textbox, and audio that demonstrates concepts. “Challenges” are levels in which the user must also walk to a piano, for consistency’s sake, which will trigger a short quiz. The dialogue of the quiz appears within the textbox. Four images appear in the sky above the textbox, representing multiple choices the player may choose to answer each question. If an incorrect choice is chosen, an error sound effect plays. If a correct choice is chosen, a confetti-like particle effect plays, a success sound effect plays, and the next question is displayed. Once the quiz is completely finished, all the images dissolve into particle effects.

Each “lesson” precedes a corresponding “challenge” on one music concept. The player can travel between levels by passing through a door placed in each level. However, the level’s door does not trigger the next level if the quiz or lesson has not been completed.

Graphics

Graphics assets were all created in the illustration software Paint Tool SAI with a Wacom Bamboo tablet. Assets include level backgrounds, object sprites, and character sprites. The poses for the main character sprite’s walk animation were made by heavily referencing a walk sequence generated on a sprite sheet generator[7]. Several UI images such as a textbox background and left and right arrows are taken from tutorials[4][5].

Particle Effects

Particle effects are used to help guide the player through the gameplay. A soft glowing particle system behind the doors or pianos indicates where the player should move towards. A
confetti-like system indicates when a quiz question has been correctly answered. A colorful dissolving rainbow particle effect indicates a successfully answered quiz level. These all provide some feedback for the player as they navigate through the world. These are built off of concepts and resourced found in tutorials[6].

Sound

Sound assets include sound effects and background music. These assets were obtained from audio databases with Creative Commons licenses. Sound effects are used to give the user feedback during “challenges” on whether or not the questions are answered correctly, and during “lessons” to accompany and demonstrate lesson material. Different background tracks play depending on whether the current level is a “challenge” or a “lesson” to distinguish the experience between the two.

Results

Summary

Music Times is a two dimensional educational mobile video game developed in the Unity game engine, scripted in C#, and illustrated using Paint Tool SAI. It focuses on teaching music by using elements of adventure and visual novel games and incentivizing the player to learn music to explore new levels. The player moves through the game as a young human with a dog companion. There are six working levels, composed of three “lessons” and three “challenges”, and corresponding differences in visual feel. Each level teaches a different music concept. As seen in Figures 1-6 below, lesson 1 teaches the player about music notes, lesson 2 teaches rests, and lesson 3 introduces the staff and clefs.
Figure 1: Lesson 1 and challenge 1 on music notes.
Figure 2: Lesson 2 and challenge 2 on rests.
Figure 3: Lesson 3 and challenge 3 on the staff and clefs.

Movements are controlled by arrows arranged as an arrow pad with touch access. Reaching the piano will trigger each “lesson” or “challenge” and after completing the material, the door leads to the next level.
Development Iterations

Throughout development of *Music Times*, each major iteration was playtested on fellow Cal Poly students. The playtesters had varying degree of knowledge of music.

The first version playtested had a simple setup of each “lesson” and “challenge.” From this round of playtesting, one bug was overlooked, which was that players could pass through doors and go up levels without reading a music lesson or taking a “challenge.” Playtesters were confused about where to go, and how to move the main character, as the arrows were placed far apart, on the upper left and right corners. They also commented on the lack of interesting graphics.

From this feedback, a second version was made with the door and piano both visible on the screen from the start, showing the player the two places they are allowed to go. Additional backgrounds and graphics were also added in, and particle effects were added to the “challenges.” Playtesting on this second version found that it was easier to figure out where to go, but still initially hard to figure out how to move the main character. Additions to particle effects and graphics, however, were welcomed.

Final Results

The final version presented here is built from their feedback on the game mechanics, the flow of the game, and the challenge of the material. This version has 6 total levels: 3 “lessons,” and 3 “challenges.” The following are screenshots showing various areas of the final version.

![Figure 4: Idle character in game, with glowing piano indicating “lesson” or “challenge” to be completed.](image)

Final Results

The final version presented here is built from their feedback on the game mechanics, the flow of the game, and the challenge of the material. This version has 6 total levels: 3 “lessons,” and 3 “challenges.” The following are screenshots showing various areas of the final version.

![Figure 4: Idle character in game, with glowing piano indicating “lesson” or “challenge” to be completed.](image)
Figure 5: Character walking towards glowing door indicating next level.

Figure 6: “Lesson” in progress.

Figure 7: “Challenge” in progress.
Future Work

Improved “Challenge” Mechanics

The current “challenge” setup has no randomization built in. Each question in the “challenge” is sequentially read in from a text file, and each question has a hardcoded set of four possible image answers. The images are never randomly placed or spawned, so they will always appear in the same spot. To better quiz and challenge players, introducing randomization to the order of questions and answers would be needed.

Additional Levels

The final version of Music Times created in the time span of this senior project contains 6 levels, with 3 music concepts taught. In the future, more levels can be added to teach a wider scope of music concepts.

Story Elements

The underlying goal of Music Times is to completely gamify the music learning experience. The original vision was to have a side-scroller visual novel type game, where learning “lessons” and completing “challenges” would unlock chapters of a story. Because of time constraints, these were not included in the final product presented here. Story elements would have added to the player’s game experience and incentivized them into playing on to discover more of the story. In the future, additional “story” levels can be added between each
“lesson”/“challenge” pair of levels. This could involve different types of sprites placed in the level to trigger storytelling.

Reflections

Developing Music Times was a great self-educating opportunity. It was an introduction to the game development world. I was able to get an introduction to Unity through walking through helpful introduction tutorials for building simple games. From there, I learned the fundamental basics of developing and scripting in Unity and was able to begin developing Music Times. For each piece I wanted to build, I consulted documentation, the Unity question forum, and video tutorials to figure out how to achieve each piece. I was able to pull from all these sources and create the final version of Music Times. I learned how to write scripts with C#, create levels within Unity’s level editor, and search online sources and documentation for pieces I needed. I learned about the game development process as a whole, including the extras needed to add to the play experience, the different roles involved in the process (artist, sound director, programmer, writer, etc.), and the iterative playtesting process.

As I developed Music Times, I also ran into limitations. Attempting to learn the to develop with Unity while implementing the mechanics of the game in parallel led to a somewhat inflexible and convoluted implementation. Thus, improvements such as randomization of question generation are left for future work. In the future, I would rework the functionality into more modular scripts.

Ultimately, I was given a chance to explore game development on my own and to create a game from everything I learned.
Resources

Referenced Games

[2] https://sites.google.com/site/learnsightreadingmusicnotes/home

Tutorials

[5] https://www.youtube.com/watch?v=ehmBIP5sj0M (Dialogue box tutorial)
[6] https://www.youtube.com/watch?v=rR_bm8f8rVE (Particle effects tutorial and resources, under Creative Commons Attribution 4.0 license)

Graphics


Sound

[11] https://www.freesound.org/people/jobro/packs/2489/ (Under Creative Commons Attribution 3.0 license. Some modifications were made.)
[12] https://www.freesound.org/people/Heshl/sounds/269160/ (Under Creative Commons Attribution 3.0 license)
[13] https://www.freesound.org/people/shinephoenixstormcrow/sounds/337049/ (Under Creative Commons Attribution 3.0 license)