Final Project: Musical Frogger

The Assignment

In your project you should design and implement an interactive graphics-based game with musical effects. While the design of the game is similar to what you have done before, adding the musical effects adds a new dimension to the project.

We want you to explore the possibilities of the sound library, design the musical components cleanly, defining classes you may need to generate the tunes to be played on next tick or next key event.

For example, if you play a continuous tune in the background, you may compose a canon by selecting two tunes from the sequence, with a fixed offset from the first to the second.

Document clearly this part of your project.

The Frogger World

Frogger game involves a frog trying to cross a road and a river to reach the other side. There are cars on the road that can run the frog over. There are lily pods on the river on which the frog can float across. There may be dangerous logs that make the frog fall into the river and drown.

All of these objects move - typically one lane in one direction, another one in the opposite.

The game may give the frog several lives before the game ends.

The implementation of this part of the game follows the pattern we have used when designing other games. especially the Mario game.

Musical Effects

The web site for the *isdraw* library contains documentation and samples of how the library can be used.

You want to add musical effects on each tick and the logs and lily pads move, effects on key events when the payer moves the frog forward, and some final drumroll when the frog dies or reaches the other side.

Think of how you can compose more complex melodies from some simple ones. You should probably design a separate class that will deal just with producing the next collection of sounds to be played.

I will be looking for good design of this part of the program, as well as imagination and ingenuity.

Using Libraries

The *isdraw* library extends the *idraw* by giving the programmer the opportunity to play music on each tick or in response to a key event. The `World` contains two containers for tunes, `TuneBucket`s. All tunes that are added to the `tickTunes` `TuneBucket` are played on the next tick and continue playing until the next tick. All tunes that are added to the `keyTunes` `TuneBucket` are played in response to the key evens and play for the duration of four ticks.

A tune consists of an instrument that plays the note and the note to be played.

If you are interested in how the music is played, feel free to explore the library, and to explore further the support for MIDI music synthesis provided in Java libraries.

Furthermore, throughout the project you are encouraged to leverage as much as possible from the existing Java libraries (both the Java Collections Framework, and the JPT libraries).