FRAUDS! are…

David Allen (dpallen@wpi.edu)

Henry Wheeler-Mackta (hjwheelermackta@wpi.edu)

Working Game Title:

"Perl-Jam"

Figure 1: Game Logo

Logline:

"Jam on!"

Summary:

Perl Jam is an interactive music-creation tool designed both for novices with zero composition experience and musically-learned individuals who just want to make a quick tune. It features an intuitive one-screen interface that features a section for melody, rhythm, and tempo,

all of which being manipulated via mouse interaction. Clicking on a position in either of the first two sections will create a note that will automatically be played in a rhythmic manner. The more notes that are added, the more complex the repeated song! The tempo of the musical loop can further be altered via an easy-to-use slider on the bottom of the application – allowing both for upbeat and slower tunes to be made.

- Click on the Perl Jam grid to toggle different notes on or off.
- Your composition will automatically play as soon as the first note is added.
- Add notes to either the melody or rhythm area to create the perfect musical loop.
- All notes are on the pentatonic scale, ensuring that any composition will sound good.
- An integrated tempo-slider allows for easy tempo-shifting.
- Pressing spacebar pauses the playback in realtime.

Experience Goals:

The goal of Perl Jam is to allow for musical creativity to flow unhindered. With its simple interface and seamless transition from composition to playback, the program will allow for players to develop songs in real time; adjusting note location and pitch without having to stop the music. In comparison to the traditional digital audio workstation, which often require the user to stop audio playback when manipulating a given track, Perl Jam allows for easy editing on-the-fly, with the resulting output being promptly played. This allows for players of all ages to enjoy the satisfaction that comes from creating music in an immediate way.

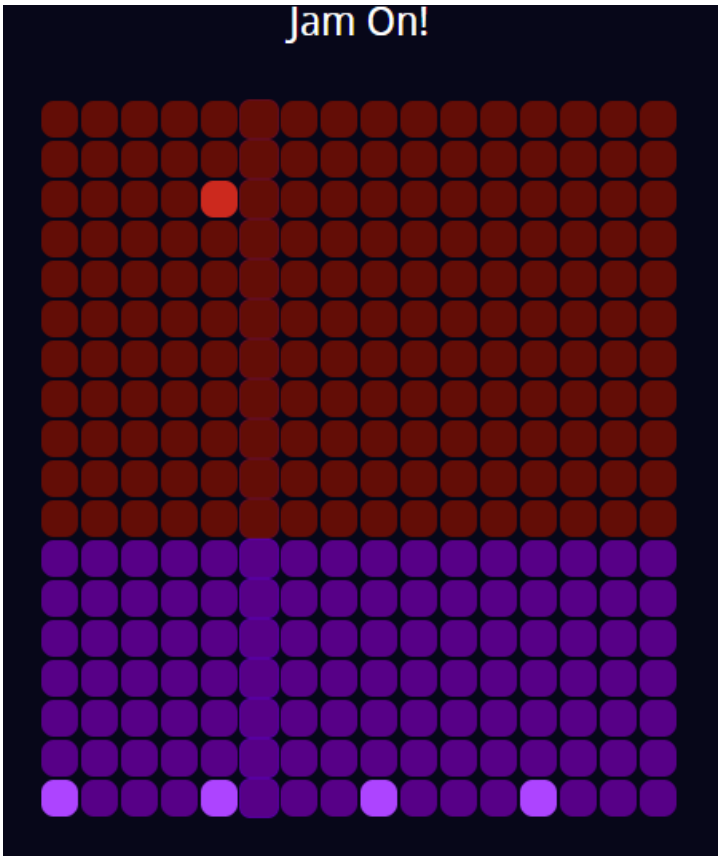

Figure 2: Entire Grid

Gameplay Description:

The Three Grids

The Perl Jam interface is made up of three primary grids: The melody grid, the rhythm grid,

and the tempo grid. The first two are closely related, whereas the third impacts the playback of the others. All grids can be interacted-with by the user, with every bead being clickable and thus manipulating the overall song. It will be immediately visible to the user which beads are in which interacted state.

The Melody Grid

The first grid is initially empty, with each cell set to an "off" state. When any of its cells are clicked on, they will toggle to "on." This will be represented via a change in color. The x-axis of the grid represents pitch, yet will be unlabeled to prevent users from being overwhelmed whilst also maintaining a minimalist aesthetic. Thus, each row of the grid represents a different note or pitch. The y-axis of the grid will represent time, with the progression of notes flowing from left to right. Much like a real instrument, multiple notes can be toggled within the same column, thus allowing for simultaneous pitches to be played.

The *Melody Grid*'s outputted sound will be a harmonious xylophone.

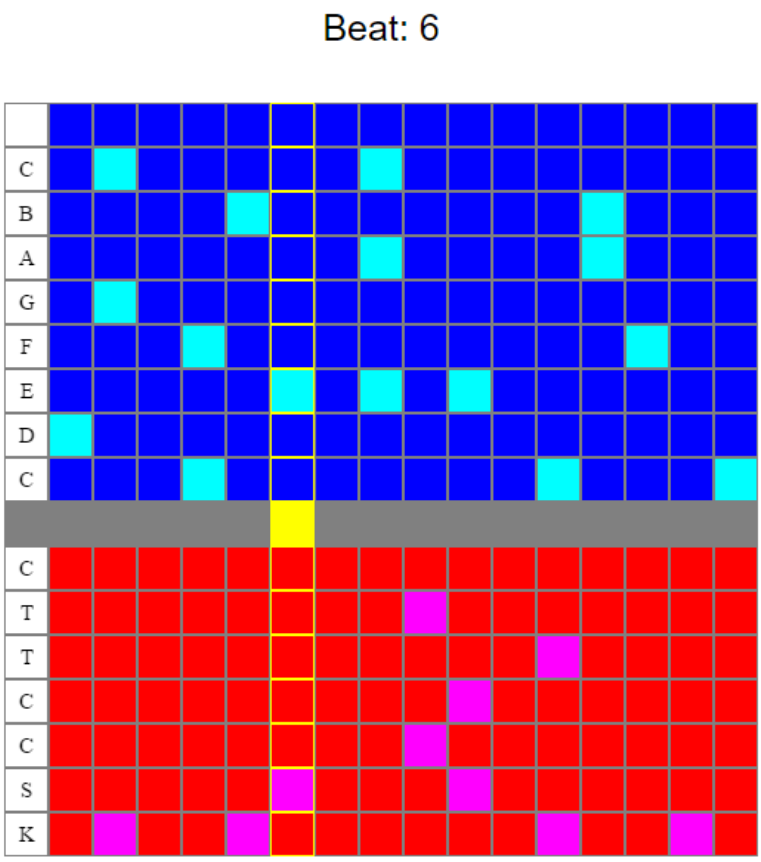

Figure 3: Mockup of Melody and Rhythm Grid with Placed Notes and Progressing Metronome

The Rhythm Grid

Aesthetically and functionally identical to the *Melody Grid*, the *Rhythm Grid* essentially represents a drum-track. Because of the lower components of a drum when compared to other instruments, this grid will ultimately contain less rows.

The *Rhythm Grid*'s outputted sound will be an analog drum kit.

The Tempo Grid

Unlike the other two grid components of Perl Jam, the *Tempo Grid* will function as a slider rather than a cell-based

board. The slider will feature numerous positions that spread from left to right, with the leftmost representing the slowest tempo and the rightmost representing the fastest tempo. This will likewise be colored such to represent the distinction. The color of the slider will be a gradient of one or two distinctive colors that will easily convey to the player that one side results in a slower playback while the other results in a faster playback. The movable portion of the slider – this being the component that determines what the playback tempo is – will be represented by a simple border that allows for its position to still be easily visible.

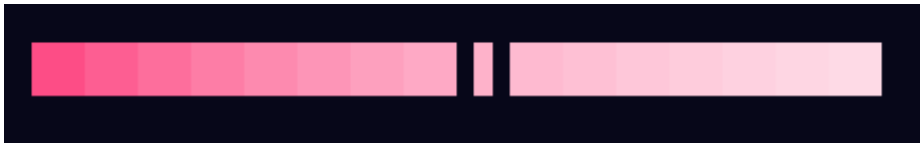

Figure 4: Tempo Slider

Musical Playback

Perl Jam will feature automated musical playback that is initialized as soon as the application is opened. This is represented via a slim line that repeatedly runs from left to right across both the *Melody Grid* and *Rhythm Grid*. This line is handled via an internal metronome – the tempo of which being adjusted via the *Tempo Grid*. When the line approaches a note, that note is played. The note played will be taken from a library of pre-recorded synth sounds that span the pitch range of the given grid.

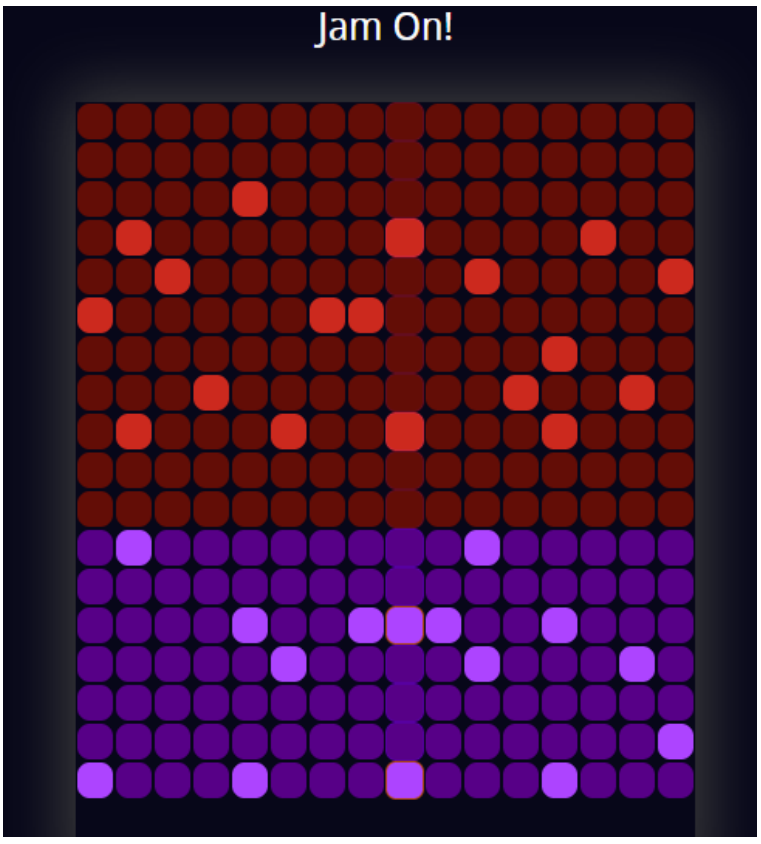

Figure 5: The Complete Grid Populated with Notes

Lack of Hotkeys

To maintain Perl Jam's intuitive and compatible-with-all-ages interface, there will be no required hotkeys for tool usage. Likewise, pausing the loop will be impossible such to maintain the program's identity as a music-editing toy rather than an actual digital audio workstation. The ability to pause or otherwise alter playback

would theoretically ruin the tool's inherent flow of music, thus they will not be present in any sense.

Initial Reveal

Perl Jam is intended to be an easy-to-pickup experience that is as intuitive as it is fun. In addition to allowing the user to create an engaging musical beat, it is also intended to help the user in learning how to build such a beat. Thus, when first initialized, Perl Jam will only permit that the user interacts with the major beat columns. Upon clicking on one of these, the rest of the grid will open up and playback will begin. This gives the user a musically-sound jumping off point, and also provides for a cinematic introduction to Perl Jam.

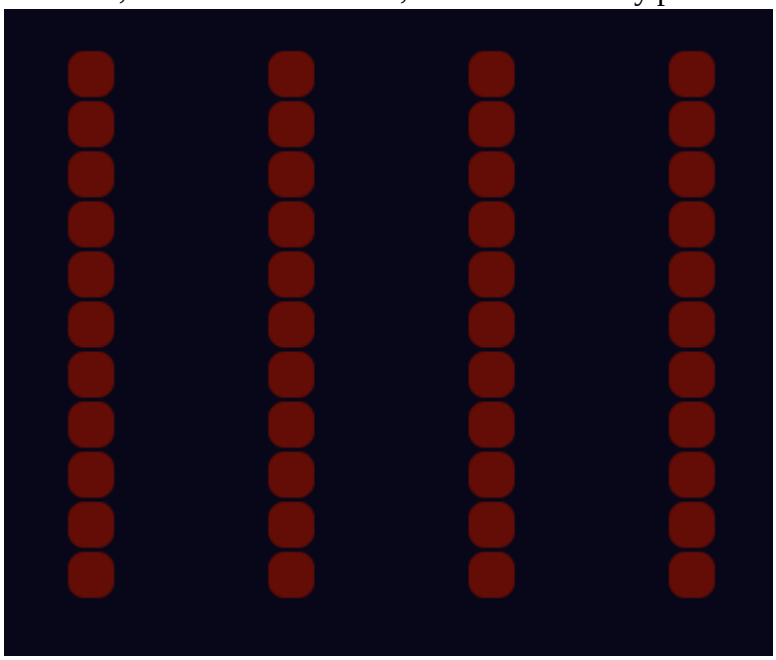

Figure 6: Initial State

Product Details:

Member Roles

David Allen - programmer, sound designer, mockups

Henry Wheeler-Mackta - programmer, writer, artist, producer

Schedule

Sunday, March 20th:

Toy treatment	–	Henry Wheeler-Mackta
Screen mockups	–	David Allen
Toy prototype	–	David Allen

Monday, March 21st:

Determine color pallet	–	Henry Wheeler-Mackta/David Allen
Collect musical sounds from professional library	–	David Allen

Prototype of overall UI	–	Henry Wheeler-Mackta/David Allen
Add toy treatment to FRAUDS! website	–	Henry Wheeler-Mackta

Tuesday, March 22nd:

Present Perl Jam	-	Henry Wheeler-Mackta/David Allen
Implement metronome component	-	Henry Wheeler-Mackta
Implement sounds for playback	-	David Allen

Wednesday, March 23rd:

Testing	-	Henry Wheeler-Mackta/David Allen
Aesthetic refinement	-	Henry Wheeler-Mackta/David Allen

Thursday, March 24th:

Testing	-	Henry Wheeler-Mackta/David Allen
Aesthetic refinement	-	Henry Wheeler-Mackta/David Allen

Friday, March 25th:

Present Prototype	-	Henry Wheeler-Mackta

Saturday, March 26th:

User Testing w/ Non-Wpi-Residents	-	Henry Wheeler-Mackta/David Allen