HW1 ReadMe File

Lawson Busch

Roland Munsil

February 6th, 2017

ReadMe

* To keep track of the characters, we made a small class name fallingChars that keeps track of the character, position, and velocity of a text object on the string.
* We decided to have our characters picked from the *Itsy Bitsy Spider* children’s rhyme in order to better fit the text rain theme. We wanted to have a reasonable chance for complete words from this rhyme to appear and not just characters from the string, so we used a random number generator, and if the number generated is less than 1/20.0, we pick a word from an Array of words from the *Itsy Bitsy Spider* and add all of the characters at once. These characters also have a smaller velocity range than normal characters, so that the word falls at approximately the same rate. Otherwise, if random less than 1/2, we add a single character from the *Itsy Bitsy Spider* to our array of fallingChars.
* In order to prevent our characters from falling through thin black pixels, we made our testing algorithm more robust by testing multiple points on each fallingChar, roughly testing the bottom left corner, bottom middle, bottom right corner, top right corner, and top left corner.
* In order to avoid screen clutter, we implemented a LinkedList that stored all of our fallingChar objects. We used a LinkedList so that as the screen began to clutter (we determined that “cluttered was around 100 fallingChars”) we could remove the oldest fallingChars (char 0) from the screen in an O(1) operation.
* For our Wizards attempt we drew a simply image of a spider, and animated it to descend from the top of the screen if the word spider was selected by our text rain program. To easily see this affect, you can comment lines 128, 129 and uncomment line 130.