Checkpoints Description

CP1: Import the game API, change the 2-D coordinates of the sprite.

Specifications

The requirements for this assignment are to:

1. Place Persephone somewhere in the bottom right quadrant of the screen
2. Draw a text string consisting of your name below Persephone in a custom color of your choosing

CP2: Create a vector-type object and use java queues to move the position of the sprite across the screen

Specifications

The requirements for this assignment are to:

1. Load your custom sprite (128 x 128) into the game engine using Art.txt
2. Make your sprite appear when running the program
3. Write the Vector2D class using the template
4. Load your frames into the first Queue
5. Using a timer, two Vector2D queues, and your custom engine, make the image move from the left side of the screen to the right. Once the image reaches the end of the animation, it should reset by transferring the second Queue back to the first

CP3: Using a single ArrayList to hold sprite animation frames and positional coordinates

The requirements for this assignment are to:

1. Load your custom sprites (128 x 128) into the game engine using Art.txt
2. Write the spriteInfo class using the template
3. Load your frames (and coordinates) into the ArrayList in the start method
4. Create a cursor variable that holds the index of your current displayed frame (relative to your ArrayList). This makes for easy updating of the next frame once a timer is up.
5. Using a timer, your ArrayList, and your custom engine, make the images move from the left side of the screen to the right. Once the image reaches the end of the animation, it should reset by making the cursor index 0.

CP4: Use a hashmap to store character dialogue lines; draw a new dialogue line every 5 seconds using a random key from the hashmap.

The requirements for this assignment are to:

1. Name your animated character from Checkpoint #3
2. Write five (5) lines of dialog for your character
3. Save dialog into a text file using Key*Value pairs as described in this module
4. Load text file into game engine in setup method. Load ALL dialog into a Java HashMap using a StringTokenizer to parse the lines into Key, Value pairs
5. Retrieve a single line of text from your HashMap using the "get" method from Java's HashMap class and display it using the Gaming API "drawString" method (at location 100x, 250y).

CP5: Trigger dialog by processing key input.

The requirements for this assignment are to:

1. Program triggers for the following keys: w, a, s, d, and spacebar
2. Each trigger should say "(keyname) has been triggered" (similar to the tutorial video on the KeyProcessor)