Instructions

Close

Overview

1. Make a copy of your code from part 3b

2. Make an array of tree positions

- Declare a variable called `trees_x`
- In `setup`, initialise it with an array of numbers.
- Each number should represent the x-position at which a tree will be drawn on the canvas.

3. Draw the trees

- In the `draw` function create a for loop to traverse the `trees_x` array.
    - HINT: you need to use `trees_x.length` to make sure you loop over every item in the array.
- Copy your tree drawing code from part 2b into the body of the for loop
- Now modify your code so that each tree is drawn using the corresponding x position from `trees_x`.
    - HINT: If your for loop uses a variable called `i` you can get the x position by using `trees_x[i]` - You should end up with lots of trees in different positions.

4. Make an array of clouds

- In `setup`, declare and initialise a `clouds` variable with an array containing some cloud objects (e.g. at least 3).
    - HINT: you can copy the one from part 2b but vary the x and y positions of each object.

5. Draw the clouds

- In the `draw` function create a for loop to traverse the `clouds` array.
- Copy your cloud drawing code from part 2b into the body of the for loop.
- Now modify your code so that each cloud is drawn with the position and size determined by the corresponding object in the array

6. And now for the mountains

- Repeat stages 4 and 5 for the mountains

7. Implement scrolling

- To create an expansive game world we need to make a virtual camera which follows the game character as they move about the game world
- We're going to do this by making the background scenery scroll in the opposite direction when the game character moves left or right.
- We can achieve this by using p5's [`translate`](https://p5js.org/reference/#/p5/translate) function in combination with [`push`](https://p5js.org/reference/#/p5/push) and [`pop`](https://p5js.org/reference/#/p5/pop)
- Make sure you've read about how these work before attempting the following steps
    - Declare a variable called `cameraPosX` and initialise it to 0
    - Make sure that all of code which draws all of your game scenery appears consecutively within your draw loop directly BEFORE the code which draws your game character.
    - Just *After* the code which draws the ground and just *Before* your first item of game scenery, add the command `push()` followed by the command `translate(-cameraPosX, 0)`
    - Just *After* the code which draws the game character add the command `pop()`
    - Now `cameraPosX` controls the left most position of where the camera is within the game world.
    - At the start of the draw loop, write a line of code to continually change the value of cameraPosX so that the game character always appears in the center of the screen but the background moves behind them.
    - Test that your canyon and collectable still interact as expected

8. Code Presentation

Make sure you produce readable code:

- Use correct indentation

- Remove unnecessary whitespace
- Remove any unused commented-out code
- Remove old and redundant variable declarations
- Make sure all variables are declared
- Name your variables carefully
- Include brief, descriptive comments for each section

9. Submission format

- Before submitting make sure that your code runs and that all the necessary files are included in the sketch folder
- Zip the sketch folder. Make sure that it is only zipped at one level and that the file extension is a .zip

Review Criteria

- Character Interaction & Rendering (5)
- Collectable Interaction (5)
- Canyon Interaction (5)
- Anchoring of background objects (5)
- Initialisation of data structures (5)
- Traversal using for loops (5)
- Scrolling Implementation (3)
- Code Quality (6)
- Submission organisation (1)