Guardian Nimbus: Technical Document

Prepared by: Evan Schipellite

Required Components

- A player system containing information about the flower's health, the current time, the current status of cool-downs, and the mouse's current location.
    - This class is used to monitor and alter most of the information on the screen.
    - This class should have a percentage amount for each of the abilities depending on how soon the player can utilize them again.
- A flower class that controls the bounding box of the flower and its animations.
    - Upon collision with the enemy it will call the player system to decrement the flower's health.
- An enemy class that controls the spawn location, direction, velocity, and animation of the enemy.
    - Most of the spawning mechanics will likely be random, and therefore most of the calculations will be done within the class as long as the constructor takes in the screen size and platform locations.
    - This class will handle any 'falling' situations and also animate and remove itself upon collision with one of the ability objects.
    - The enemy should be able to halt movement, fade, and delete itself when hit with lightining.
    - The enemy should be able to be sent flying in a random direction while rotating quickly, and delete itself off-screen when hit with a tornado.
- A lightning class that spawns a lightning object that will end on the player's mouse position at activation.
    - This class will create a particle effect and a bounding box for a second or two at the player's mouse location.
    - The enemy class will respond if collision occurs, and then lightning object will remove itself and the particle effect at the end of its lifespan.
- A tornado class that spawns a tornado object with an attached particle effect and bounding box at a given location.
    - The tornado will check for collisions with enemies during its lifespan.
    - At the end of its lifespan it will remove itself and its particle effect.
- Rain and Snowstorm class and effect
    - This differs from the previous abilities as a tangible object is not actually created. Instead, the class will inherit the status of 'rain' and create an ongoing rain particle effect until otherwise altered by the gameplay.
    - The player class will call the rain and snowstorm class to change when the player uses the snowstorm ability. At this point the class should switch its particle effect from rain to snow and call a function in the enemy class that slows and animates enemies properly.
    - After several seconds, the rain and snowstorm class should return to its rain status and call a function in the enemy class in order to return the enemies to their normal stance.
- World class
    - This class should merely serve to construct a texture to display the immediate terrain on the screen.
        - It will also set the bounding boxes depending on the sprites utilized to create the word.
- Background class

    - This class is merely utilized to create a scrolling background with depth, darkening, and parallax on plains, mountains, and clouds in the distance. Gameplay should not have to influence this class.
- An input manager
    - A basic input manager for registering keys and mouse clicks. (Works with Xbox too)
- A menu manager
    - A basic menu manager for creating menu screens and a pause screen.
        - The menus should allow for wrapped / scrolling text.
        - The menus should include 'buttons' instead of key presses.
        - Menus should allow for animations to be displayed.
- Particle System
    - A revision to the particle system will be required to allow it to be more flexible when creating particles for specific sizes and lifespans.
- Particle Effects
    - A rain particle effect must be created.
        - Rain should cover the entire screen and adhere to a specific depth.
    - A tornado particle effect must be created.
        - The tornado should have a wind-up and wind-down visual and effect must be able to accept a size and lifespan.
    - A lightning particle effect must be created.
        - The lightning should be able to stretch between two positions and accept a size and lifespan.
    - A snow particle effect must be created.
        - Snow should cover the entire screen and adhere to a specific depth.
- Pixel Shaders
    - Various shaders will be utilized in association with animations
    - Enemies struck by lightning should be affected by a 'shock' shader.
        - The shock shader may invert the enemy sprite or give it a blue/white allure.
    - Enemies struck by lightning should also be affected by a 'fade' shader that will decrease there alpha as they disappear from the game.
    - Enemies in the snowstorm should be affected by the 'snow' shader
        - These enemies should turn slightly blue while they are moving slower.
    - The death screen should be accompanied by a 'grayscale' shader.
    - While the screen itself will be a fully lit, a 'light' shader will be centered on the flower.
        - This will increase the brightness of the middle of the screen
        - The screen will become slightly more dim upon approaching the extremes.

Risk Assessments

- Creating the particle effects for lightning and tornado may prove difficult and will require more debugging and research in order to complete.
- Avoiding lag due to particle effects shouldn't be an issue, but it something to be aware of when managing the creation/deletion of effects.
- Creating the visuals for the cool-downs may require a new form of login to display the percentages in a way that shows the player the progress of the abilities return after usage.

- Drawing, not technical, but still important. Drawing for animations will take some time. Therefore images should be extremely simple and menus should be just as simple.
- Creating new pixel shaders and applying them to specific objects may take some time and revising of code, but the logic behind the new pixel shaders should be simple.

Milestones and Schedule

Week 1

- Create GD and TD
    - Finalize GD as much as possible
    - Ensure that the goals are very simple while leaving room for extra features if possible
    - Make sure the TD is detailed with everything needed to be built and a balanced schedule.
- Create a prototype
    - Utilize level editor to construct the 'basic' game level.
    - Enlist 'enemies' that run toward the middle area.
    - Allow the player to click on the enemies to destroy them.
    - Game won after time, Game lost after too many enemies reach the center.
    - Keep everything as simple as can be, just show the core gameplay.
        - This should show the game direction, while leaving the advanced abilities and enemy ai for later development.

Week 2

- Draw the cloud, plant, and enemy sprites and animations.
    - These sprite sheets should at least have the **essential** images for the cloud, plant, and enemies.
        - This means idle animations for the cloud and plant.
        - This means idle, movement, and death animation for the enemies.
- Create the player class.
    - A simple player system that records the basic gameplay information.
    - Cool-down display and information **does not** have to be done yet.
    - The player should be able to use the mechanical aspect of lightning.
        - Upon clicking on an enemy it should delete itself.
- Create the **foundation** of the lightning class
    - Creates a bounding box for a brief time period.
        - Upon colliding with an enemy bounding box the enemy should be removed.
        - Removes itself after a short time period.
- Create the world class.
    - Takes in a sprite sheet and text file.
    - Creates the basic terrain set-up for the level.
    - Places the proper bounding boxes on the platform pieces.
- Create the **foundation** of the enemy class
    - This should allow for animate enemies to spawn at the sides of the screen.
    - Enemies should be deleted when colliding with lightning.

Week 3

- Finish the lightning class
    - Create a lightning particle effect for the game.
    - The lightning class should now spawn a lightning particle effect when created.
    - The lightning particle effect should start from the cloud area at the top of the screen and end where the mouse was at activation.
    - It should also delete the particle effect upon completion.
- Enemy class should be complete except for tornado and snow interaction.
    - Enemies struck by lightning should now be influenced by a 'shock' shader, fade, and then be deleted.
    - A 'shock' shader will be created, and will likely work as an inverted pixel shader with slight alterations.
    - Enemies should properly fall from the upper platforms, land on the ground, and should continue moving toward the flower in the center.
- The player class will be altered in order to display the Lightning symbol and cool-down display on the screen.
- The menu manager should be started in order to set-up the basics for the menu, instructions, story, options, and game start. Basic texts and buttons should be displayed.

Week 4

- Create the tornado class.
    - Basic implementation should create a collision box in an area for a set time.
    - Enemies that collide with this box should begin rotating and should be sent flying off-screen in a random direction.
    - Further implementation should call for creating a tornado particle effect that spawns when the tornado object is created and dies when it is deleted.
- The player class will be altered in order to display the Tornado symbol and cool-down display on the screen.
- A precipitation class should be created that, for the moment, has one status that allows it to create a rain particle effect that causes rain to appear on the screen.
    - A rain particle effect must be created.
    - The rain should be behind all of the objects on the screen through depth.
- At this point lightning and tornado abilities, as well as enemy reaction should be working properly.
- The background class, simply showing mountains and other dark clouds affected by parallax, should be created. This should be left as very simple with few details to avoid distracting the player.

Week 5

- Add the snow status to the precipitation class.
    - When called by the player system, this class should switch the rain particle effect to a snow particle effect.
    - This player system should also call the enemy class to begin slowing the enemies.
    - After a certain amount of time, the snow status should change back to rain.
- The player system should now also display the cool-down visual for the snowstorm ability.
- The enemy class should also be able to react to the snow particle effect change when called.

    - Enemies should be slowed and a ‘snow’ shader should be impressed on them in order to turn them slightly more blue.
- The flower class should animate properly when enemies deduct health and alter depending on how much health it has.
- The menus in place should now be polished.

Week 6

- Add Pause and Credits
- The four languages should now be part of the system and changeable through the options.
- Set Xbox Controls
- Polish everything.
- Playtest the game.
- ~~Add sound effects if time allots.~~