Evergrow

AI Presentation

Team 12

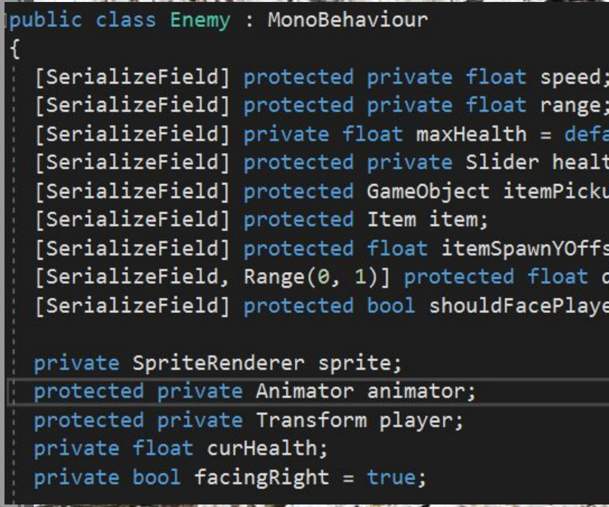
Game Description

- Play as the protagonist who falls down a rabbit hole into a scary underground world
- Earn the favor of the local deity, embodied as a dying tree, by restoring fertility to the underground land to return home
- Fight through hordes of enemies to find randomly dropped materials and seeds to grow mysterious plants
- Three areas to fight through, with four unique types of monsters and plants to grow in each area

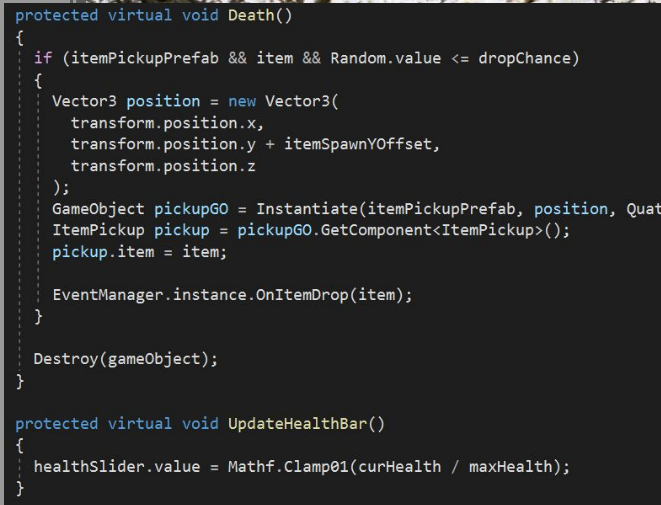
- Gameplay-wise: 2D side scrolling platformer

AI Design

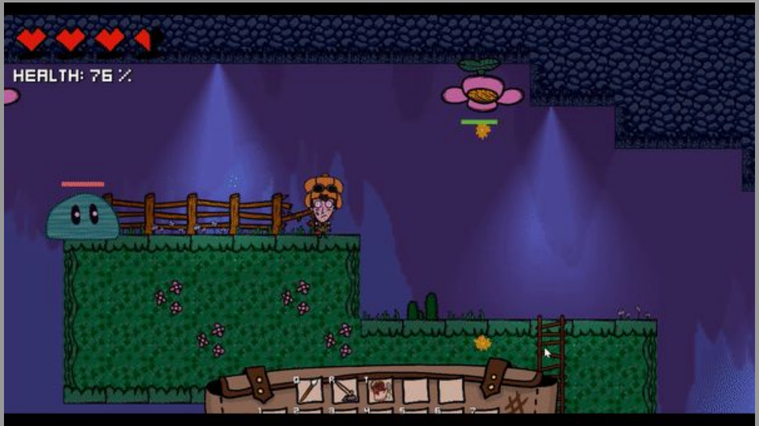
- For our monsters:
  - All monster exist in a separate layer
  - Have a probability to drop a seed once killed
  - Will only move and/or attack player if they are in a certain range
  - Have health bar to indicate the current health of the monster



```
public class Enemy : MonoBehaviour
{
    [SerializeField] protected private float speed;
    [SerializeField] protected private float range;
    [SerializeField] private float maxHealth = defa
    [SerializeField] protected private Slider healt
    [SerializeField] protected GameObject itemPicku
    [SerializeField] protected Item item;
    [SerializeField] protected float itemSpawnYOffs
    [SerializeField, Range(0, 1)] protected float d
    [SerializeField] protected bool shouldFacePlaye

    private SpriteRenderer sprite;
    protected private Animator animator;
    protected private Transform player;
    private float curHealth;
    private bool facingRight = true;
```

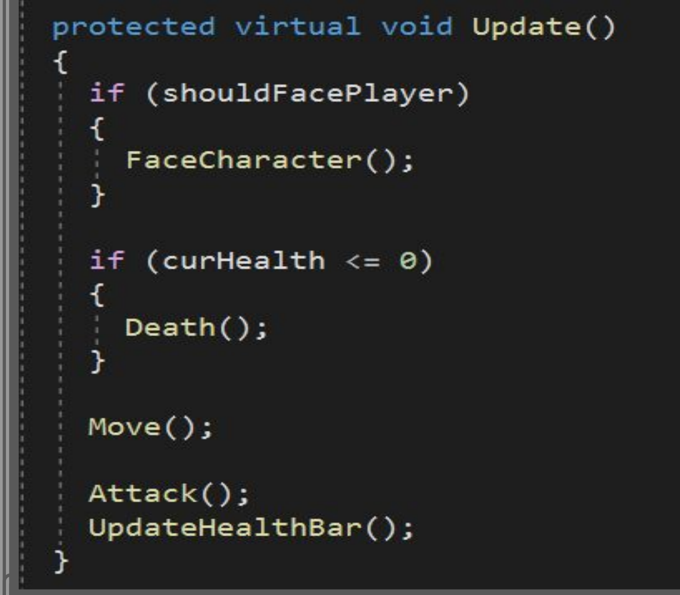


```
protected virtual void Death()
{
  if (itemPickupPrefab && item && Random.value <= dropChance)
  {
    Vector3 position = new Vector3(
      transform.position.x,
      transform.position.y + itemSpawnYOffset,
      transform.position.z
    );
    GameObject pickupGO = Instantiate(itemPickupPrefab, position, Quat
    ItemPickup pickup = pickupGO.GetComponent<ItemPickup>();
    pickup.item = item;

    EventManager.instance.OnItemDrop(item);
  }

  Destroy(gameObject);
}

protected virtual void UpdateHealthBar()
{
  healthSlider.value = Mathf.Clamp01(curHealth / maxHealth);
}
```

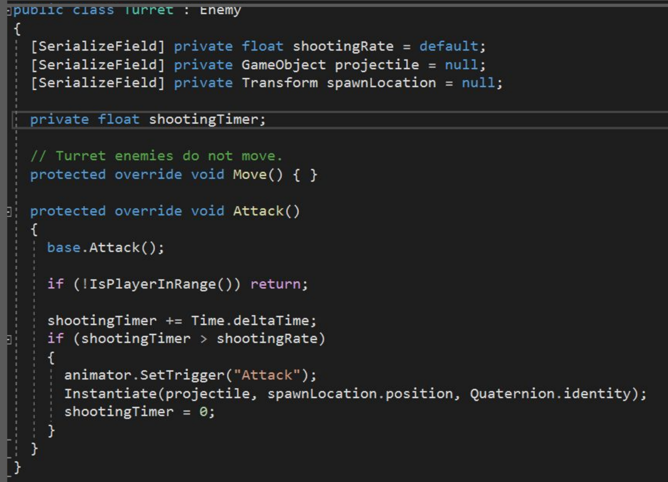


AI Design

- For our monsters:
  - All monsters inherit a base Enemy class
  - Each monster has their own behaviors that overrides
  - certain functions in the generic Enemy class
  - Use Behavioral Tree



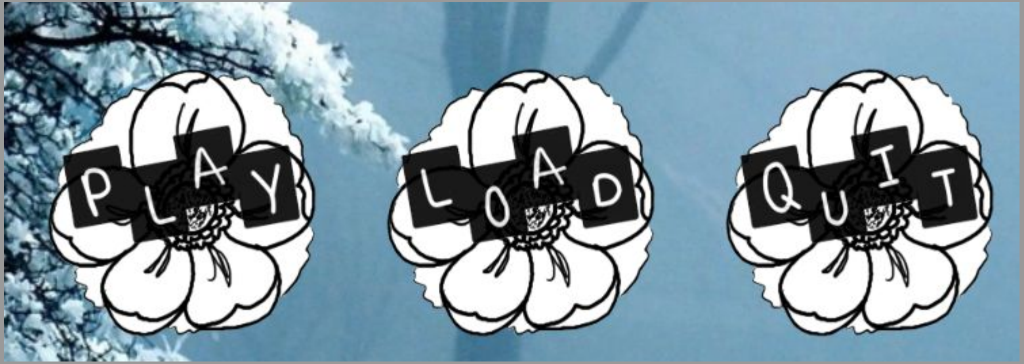
```
protected virtual void Update()
{
    if (shouldFacePlayer)
    {
        FaceCharacter();
    }

    if (curHealth <= 0)
    {
        Death();
    }

    Move();

    Attack();
    UpdateHealthBar();
}
```



```
public class Turret : Enemy
{
    [SerializeField] private float shootingRate = default;
    [SerializeField] private GameObject projectile = null;
    [SerializeField] private Transform spawnLocation = null;

    private float shootingTimer;

    // Turret enemies do not move.
    protected override void Move() { }

    protected override void Attack()
    {
        base.Attack();

        if (!IsPlayerInRange()) return;

        shootingTimer += Time.deltaTime;
        if (shootingTimer > shootingRate)
        {
            animator.SetTrigger("Attack");
            Instantiate(projectile, spawnLocation.position, Quaternion.identity);
            shootingTimer = 0;
        }
    }
}
```

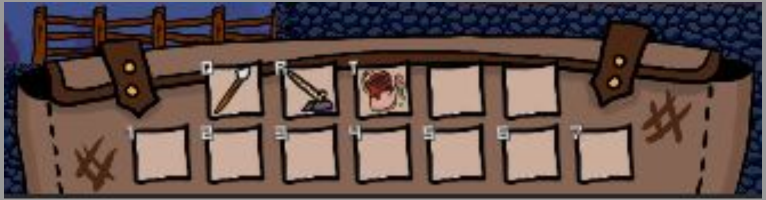
UI Design

- Menus & Buttons:
  - Simple menus; game start, pause, game over which includes a save and load system
  - Flower menu buttons based on the game's initial white-flowered tree design



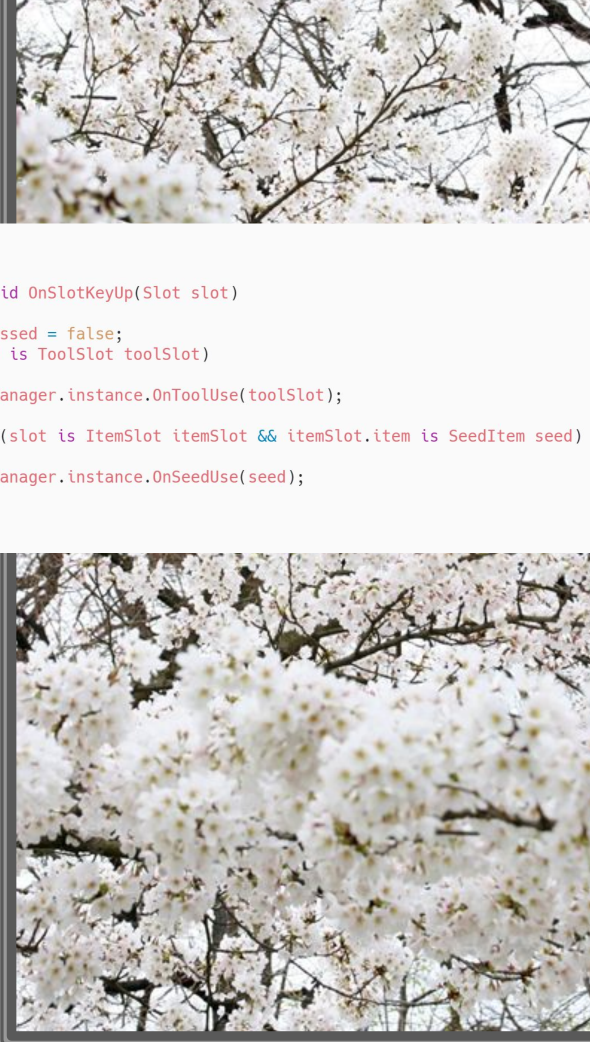
UI Design

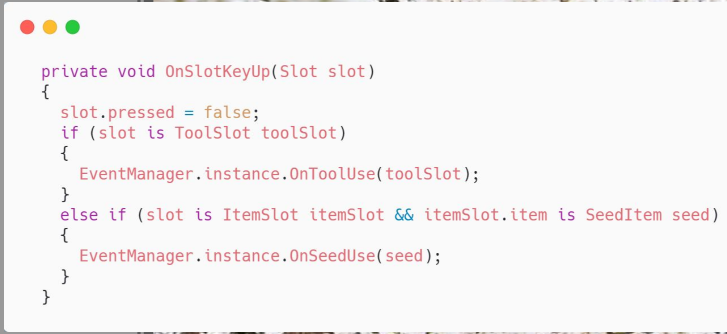
- Dialogue:
  - Simple text overlay above the inventory
  - Originally text boxes, but had issues getting them to follow and flip with the character
- Inventory:
  - Backpack hotbar-like window that always shows what items & tools you have accessible
  - Select plots by clicking on their block
  - Press a key on the keyboard to use tools and use seeds when selecting valid (unplanted) plots



Data Driven Programming: Inventory

- Item
  - `ScriptableObject` with name and sprite icon
  - Diverges into SeedItem and ToolItem
- Slot
  - Storing anything inheriting an item
  - Keybind assigned to the slot
  - Publishes drag, drop, right click, et al with item payload
- BackpackManager
  - Singleton powered by a pub/sub event system
  - Dynamically generates ToolSlot and ItemSlot instances
  - Subscribes to slot events to drop, rearrange, or receive items
  - Publishes to CharacterManager or PlantManager depending on if a ToolSlot or an ItemSlot (with SeedItem) was used





```
private void OnSlotKeyUp(Slot slot)
{
  slot.pressed = false;
  if (slot is ToolSlot toolSlot)
  {
    EventManager.instance.OnToolUse(toolSlot);
  }
  else if (slot is ItemSlot itemSlot && itemSlot.item is SeedItem seed)
  {
    EventManager.instance.OnSeedUse(seed);
  }
}
```

Data Driven Programming: Farming

- FarmManager
  - Viable farm plots exist only on a specific tilemap
  - Virtually generates plots in a Dictionary<Vector3, Plot>
  - Relates screen coordinates to tilemap cells and plots
  - Updates growth stages for each plot
  - Subscribes to ToolItem usage for tilling and watering
  - Publishes to TreeManager changes in plant growth
- Plot
  - SeedItem determines plant sprites and growth levels
  - Finite state machine `tilling -> watering -> growing`
- SeedItem
  - `ScriptableObject` - Plant reference
- Plant
  - `ScriptableObject` - array of sprites, growth time, and the growth to the tree deity upon completed plant growth

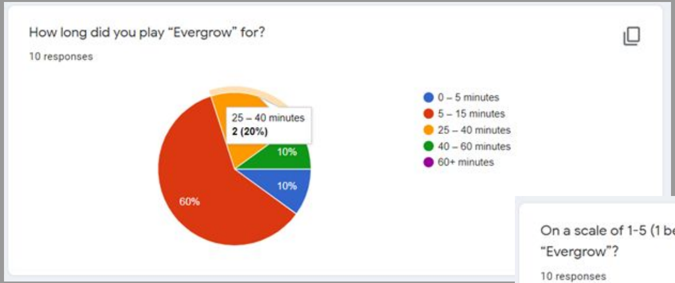
Data Driven Programming: Dialogue

- DialogManager
  - Dumps Dialogue entity sentences onto a queue
  - Advances to the next sentence on a timer or on user input
  - Subscribes to an event accepting a Dialogue
- Dialogue
  - System.Serializable containing an array of sentences.
  - Attached to objects with DialogueTrigger
- DialogueTrigger
  - Customizable script that conditionally starts a Dialogue
  - Triggers include character range, or by event
  - Optionally destructable

Software Design Decisions

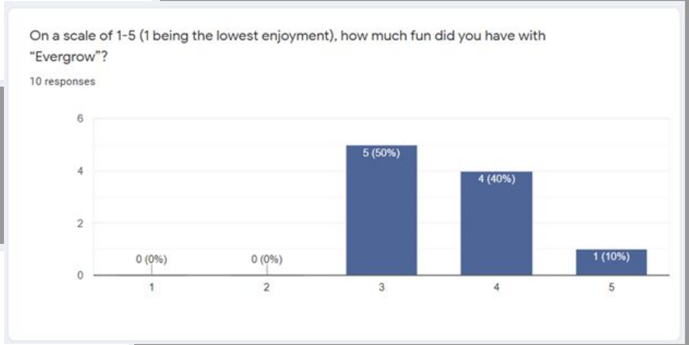
- The switch away from isomorphic ate into most early design space
- Order in which we developed the systems was likely incorrect
  - Inventory & Items -> Plants -> Tools & Weapons -> Enemies
- Completed farming and realized the game is lacking interest factor
- Unclear on the ideal way of correlating tilemap tiles to gameobjects
  - Our first example plant was made by changing tiles above the farm plot where now we calculate the center of the tile and instantiate a prefab
- Early investigation into liberally using `ScriptableObject` and writing extendable classes has rarely rarely required us to redo code

Survey results



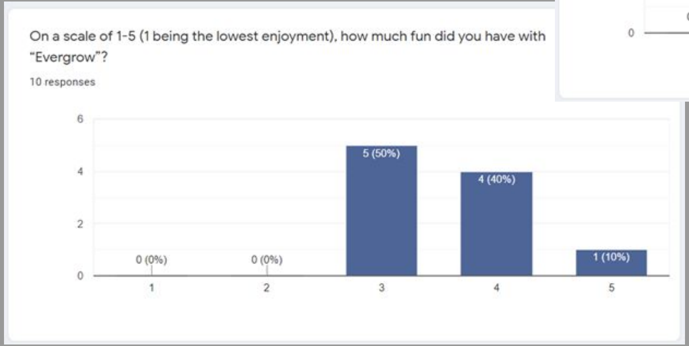
How long did you play "Evergrow" for?

10 responses



On a scale of 1-5 (1 being the lowest enjoyment), how much fun did you have with "Evergrow"?

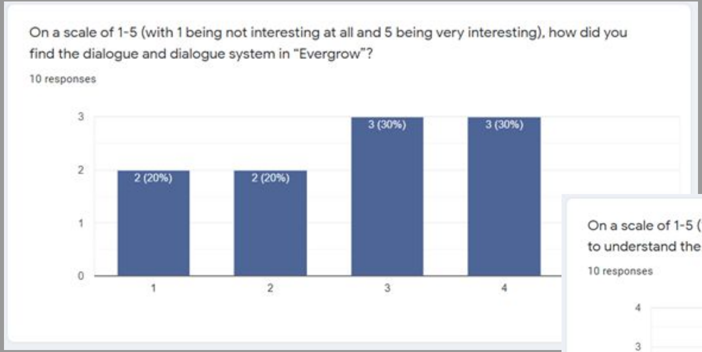
10 responses



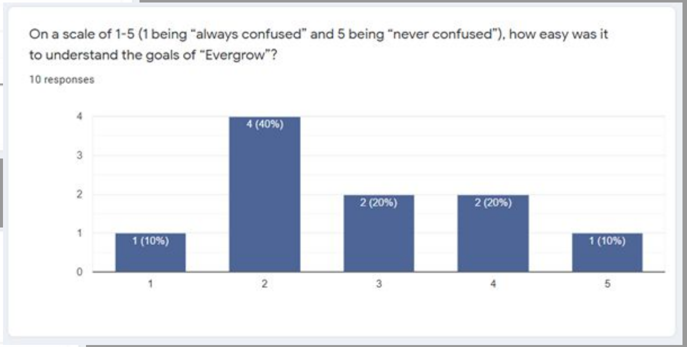
On a scale of 1-5 (1 being the lowest enjoyment), how much fun did you have with "Evergrow"?

10 responses

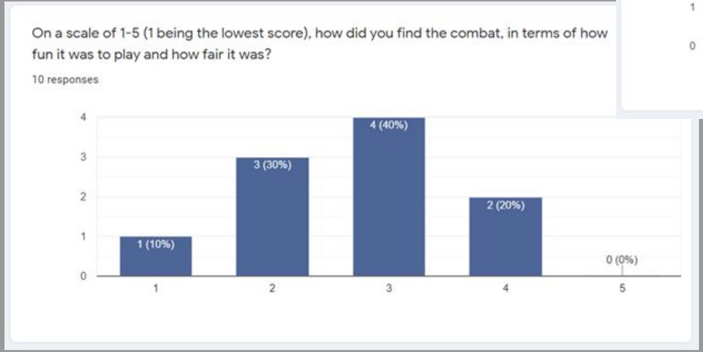
Survey results


On a scale of 1-5 (with 1 being not interesting at all and 5 being very interesting), how did you find the dialogue and dialogue system in "Evergrow"?

10 responses

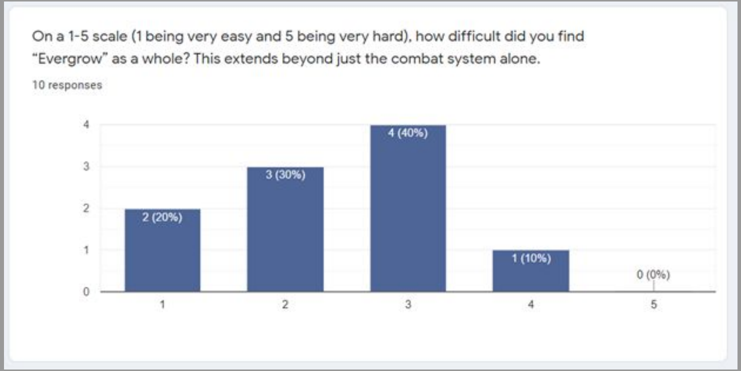

On a scale of 1-5 (1 being "always confused" and 5 being "never confused"), how easy was it to understand the goals of "Evergrow"?

10 responses

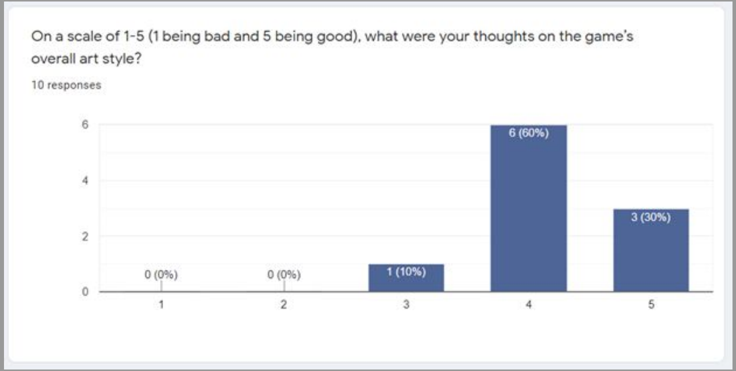

On a scale of 1-5 (1 being the lowest score), how did you find the combat, in terms of how fun it was to play and how fair it was?

10 responses

Survey results


On a 1-5 scale (1 being very easy and 5 being very hard), how difficult did you find "Evergrow" as a whole? This extends beyond just the combat system alone.

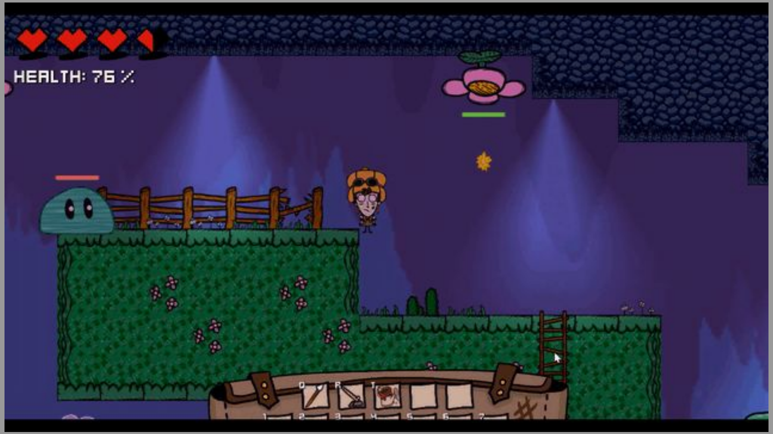
10 responses


On a scale of 1-5 (1 being bad and 5 being good), what were your thoughts on the game's overall art style?

10 responses

Playtesting Iteration

- Issues found by our users:
  - Tutorial dialogue was unclear
  - Users had trouble climbing two ladders on the map
  - Some ceiling-mounted enemies were difficult to hit considering the user only has one melee-type weapon
  - Background music was too loud for some
  - Attack animations weren't very fluid



Playtesting Iteration

- Our solution:
  - Plans to add more tutorial and story dialogue
  - Fix bugs that players found (ladders, falling into ground)
  - Plan to add some variety of projectile weapon (bow & arrow, magic) to allow more flexibility in gameplay
  - Addition of a volume slider (or at least lower the default background music volume)
  - Improvement to attack animations

Game Development Issues

- We originally made chat bubbles for dialogue, but we couldn't get the chat bubbles to properly follow the character that was speaking
- Ladders had to be set up very specifically or the user would run into problems climbing or getting stuck
- Sometimes the character collides with the ground in a way that causes them to get stuck in the tiles
- Some confusion on which layer (foreground, background, farmland, etc.) to add certain tiles on the tilemap which led to some portions of the map having to be redone when tiles were accidentally placed on the incorrect layer

Questions?