Qu-Dash
John Lee, Marcus Chiu, Andrew Halcomb

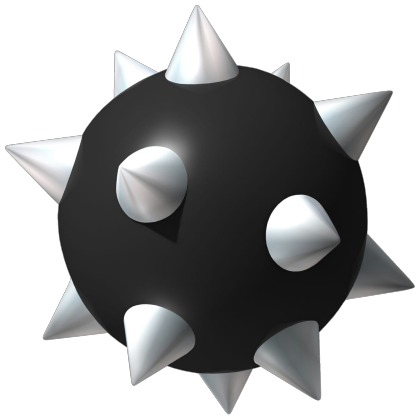
Milestones/Schedule

16- October: Gather building blocks and main sprites
23- October: Implement movements and time slow downs.
30- October: Working player (combine art, code, animation)
6- November: Implement enemies, AI
13- November: Finish level 1
20- November: Working Prototype- All basic coding, 1 Level completed
27- November: Level 2 and 3 completed
30- November: Boss level/Full prototype
Due Date: polish levels, sprites, etc…

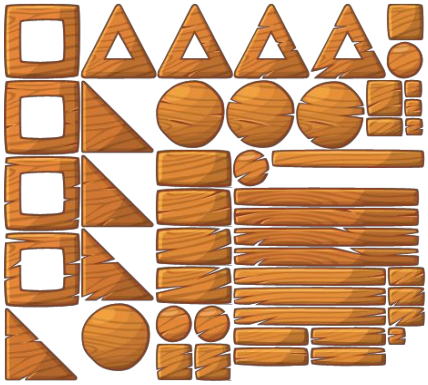
We're on track for the most part, however we fell behind on creating enemies due to mid-terms. We plan on catching up over Thanksgiving break. We did however add hazards and various obstacles- lava pits, spikes.

Work Performed

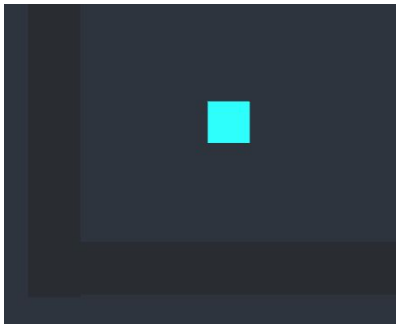
- x Gather building blocks and hazard sprites- Marcus- 2 days
- x Main character and background- John- 1 days
- x Implement movements and time slow downs- Andrew- 4 days
- x Working player- John- 3 days
- x Create obstacles and hazards- Marcus- 3 days
- x Moving platforms- Andrew 1 days



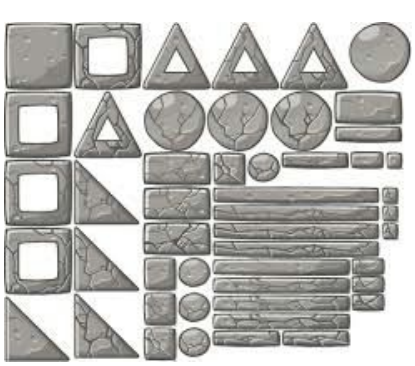
Spikes



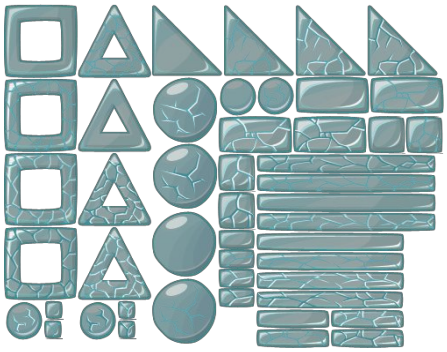
Wood building block



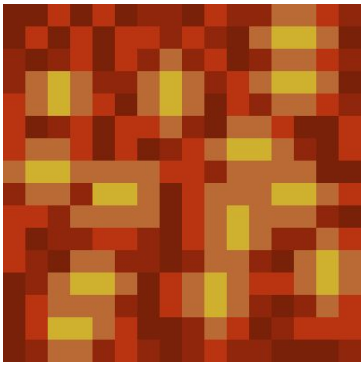
Working Player



Stone building blocks



Ice building blocks



Lava texture

```
using UnityEngine;
using System.Collections;

public class MoveScript : MonoBehaviour {

    // Use this for initialization
    void Start () {
        waiting = 0.0f;
        needToWait = false;
    }

    public Vector2 point1;
    public Vector2 point2;
    public GameObject obj;
    public float waitTime;
    private float waiting;
    private float acceptDistance = .01f;
    private bool needToWait;


    // Update is called once per frame
    void Update () {
        if (waiting <= 0.0f) {
            obj.gameObject.transform.position = Vector2.Lerp (point1, point2, Mathf.PingPong (Time.time * speed, 1.0f));
            needToWait = true;
        }
        else {
            waiting-= Time.deltaTime;
        }
        if ((Vector2.Distance (obj.gameObject.transform.position, point1) <= acceptDistance ||
            Vector2.Distance (obj.gameObject.transform.position, point2) <= acceptDistance) && needToWait)
        {
            needToWait = false;
            waiting = waitTime;
        }
    }
}

```

Code for moving platforms

```
using UnityEngine;
using System.Collections;

public class GameController : MonoBehaviour {

    // Use this for initialization
    void Start () {

    }

    // Update is called once per frame

    public boolean timeSlow; // This is changed by external methods
    void Update () {

        if (timeSlow) { // this can also be if (Input.GetButtonDown(/*name of button*/){}
                        // if it's all internal
            Time.timeScale = .2f;
        }
        else {
            Time.timeScale = 1.0f; // Restore speed if time is not slowed
        }

    }
}

```

Time slow code

```
using UnityEngine;
using System.Collections;

public class SmoothCamera : MonoBehaviour {

    public float dampTime = .15f;
    private Vector3 velocity = Vector3.zero;
    public Transform target;

    // Update is called once per frame
    void Update ()
    {
        if (target)
        {
            Vector3 point = GetComponent<Camera>().WorldToViewportPoint(target.position);
            Vector3 delta = target.position - GetComponent<Camera>().ViewportToWorldPoint(new Vector3(0.5f, 0.5f, point.z)); //(new Vector3(0.5, 0.5, point.z));
            Vector3 destination = transform.position + delta;
            transform.position = Vector3.SmoothDamp(transform.position, destination, ref velocity, dampTime);
        }

    }
}
```

Smooth Camera follow code

```
using UnityEngine;
using System.Collections;

public class Cube : MonoBehaviour {

    private float speed = 8f;            // The speed that the player will move at.
    Vector3 movement;                   // The vector to store the direction of the player's movement.
    Rigidbody playerRigidbody;          // Reference to the player's rigidbody.
    bool dash = false;

    void Awake ()
    {
        // Set up references.
        playerRigidbody = GetComponent <Rigidbody> ();
    }


    void FixedUpdate ()
    {
        // Store the input axes.
        float h = Input.GetAxisRaw ("Horizontal");
        float v = Input.GetAxisRaw ("Vertical");

        if (Input.GetKeyDown("space"))
        {
            dash = !dash;
        }

        // Move the player around the scene.
        Move (h, v, dash);
    }

    void Move (float h, float v, bool m)
    {
        if (!m)
        {
            // Set the movement vector based on the axis input.
            movement.Set(h, v, 0f);

            // Normalise the movement vector and make it proportional to the speed per second.
            movement = movement.normalized * speed * Time.deltaTime;

            // Move the player to it's current position plus the movement.
            playerRigidbody.MovePosition(transform.position + movement);
        }
        else
        {
            movement.Set(h, v, 0f);

            // Normalise the movement vector and make it proportional to the speed per second.
            movement = movement.normalized * 15 * Time.deltaTime;

            // Move the player to it's current position plus the movement.
            playerRigidbody.MovePosition(transform.position + movement);
        }
    }
}
```

Player movement (including dash) code

Work Still Required

Enemy Art- Andrew- 4 days

Enemy AI- Marcus- 5 days

- Base AI- 1 days
- Time Immune- 2 days
- Obstacle Immune- 2 days

Element Interaction- John & Marcus- 2 days

Game State Machine- Andrew & John- 3 days

Level Design- John- 3 days

- 1 days per level

Game sound- TBD- time permitting

Boss - TBD- time permitting

Changes

- Decreased enemy priority in favor of natural hazards- lava pits, spikes
- Boss moved to Backlog- will do if time permitting
- Checkpoint moved to Backlog- will do if time permitting

Overview

We encountered several setbacks with school, but to compensate we've lowered the priority on extra, time consuming tasks such as a boss level and checkpoints. We are slightly behind when it comes to level design, enemy AI, and the game state machine, but we should be able to catch up over Thanksgiving break.