ENMITY TECHNICAL DESIGN

Programming and Design
by John McCoubrie

OVERVIEW

- **Gameplay Overview**
  - Thought process of development, desired endstate
- **Lesson Learned**
  - Programming and design issues during development
  - Solutions to encountered issues
- **AI System**
  - Enemy, Boss 1 and Boss 2 AI programming and mechanics
- **Movement System**
  - WASD movement, mouse rotation, dash mechanic
- **Tools Overview**
  - Consumables, player manager, enemy and boss managers
- **Gameplay Programming**
  - Ammo boxes, health boxes, special ammo, timers
- **UI Manager**
  - UI programming, features, and explanation
- **Links**
  - Source code, playable game, portfolio link, assets used, and contact info

GAMEPLAY OVERVIEW

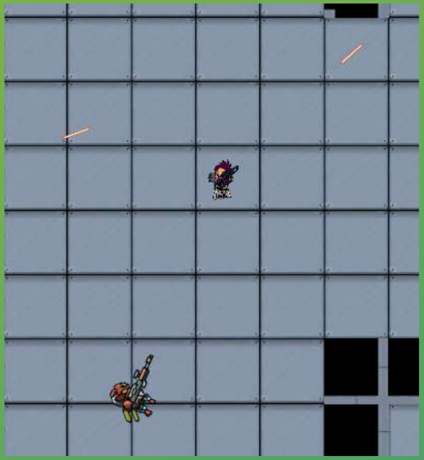
- Enmity is a 2D dungeon crawler shooter with 2 playable levels and 2 boss encounters. The player has a limited amount of ammunition and must utilize the dash mechanic, cover, health boxes, and ammo boxes in order to survive.
- Boss 1, named "The Dome" is a floating head that drops exploding fireballs as he moves around the level randomly. The player must dodge the fireballs while attempting to secure "special ammo." Special ammo will destroy the shield surrounding The Dome allowing him to take damage from regular ammo. Once The Dome takes 2 damage, the shield will respawn, and the player will need to obtain more special ammo. If the player touches The Dome, the player will instantly die. The player must kill The Dome before the level becomes overrun with fireballs.
- Boss 2, named "The Hound" is a floating dog head that charges the player every 2 seconds. The player must dodge The Hound while attempting to secure special ammo. Special ammo boxes must be hovered over for 2 seconds in order to obtain the ammo. The player must play a cat and mouse game with The Hound in order to obtain the special ammo. Similar to The Dome, the special ammo will destroy the shield surrounding The Hound and allow the player to damage him with regular ammo. The Hound's speed will increase as his health becomes lower.

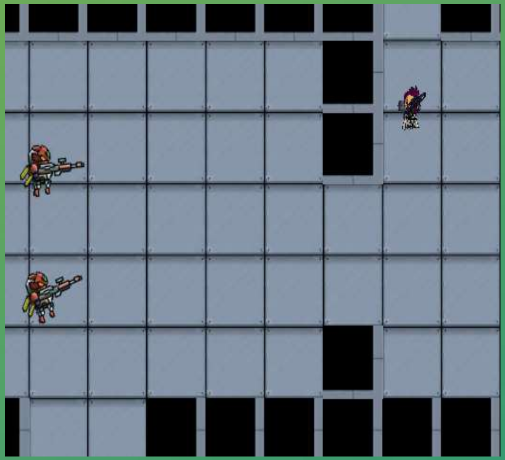
LESSONS LEARNED

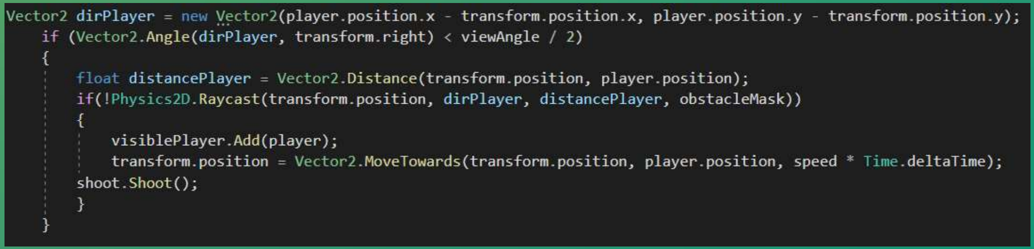
Issue	Solution
Boss 1 continuously got stuck in the corner from the random movement generator.	If boss 1 touches the wall a new movement vector will now generate in the direction of the player.
Too many timers (health, ammo, boss, enemy tracking, cooldowns) lowered performance.	Created listeners on each item to only fire when absolutely needed (i.e when the player is in range).
Bullets were affected by the weapons rotation and physics properties.	Handle physics properties individually (i.e in the mover script for the bullet).
Do not include heavy performance items such as math functions in the update function of a class.	Create separate classes to calculate math functions and logic, only call them when necessary based on a condition.
The AI systems needed to be separated.	Originally had one AI system with multiple methods, became cluttered and difficult to maintain. The different AI systems would interact with each other. Each enemy now has their own AI system.
Input controls not responsive.	Programed all physics settings to have low mass and low drag in order to favor responsive controls while still allowing push-pull feeling of a physics system.
UI size was too small.	Update to a dynamic canvas and position to account for different screen sizes.

AI SYSTEM - ENEMIES

- Enemies track player movement through walls
- Using a rayast, the enemy will fire on the player when he is in line of sight with 0 delay
- When the player is in line of sight, the enemy will slowly move towards the player, when he is not in line of sight the enemy will only track and aim towards the player
- Enemy scripts separated into bullet, look mechanic, shoot mechanic, and fields of view
- Desired feel: the player must quickly round walls and use the dash mechanic to avoid the tracking system and quick shots



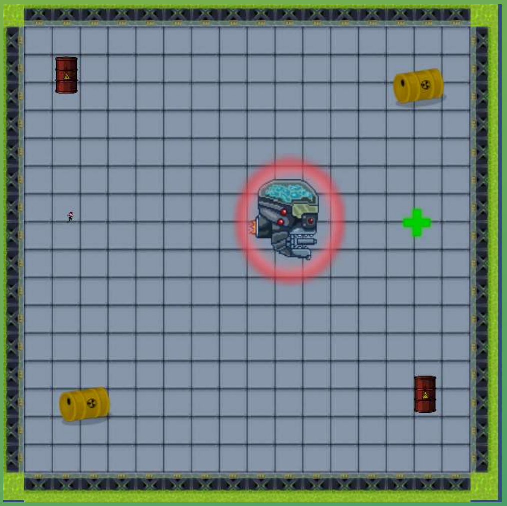


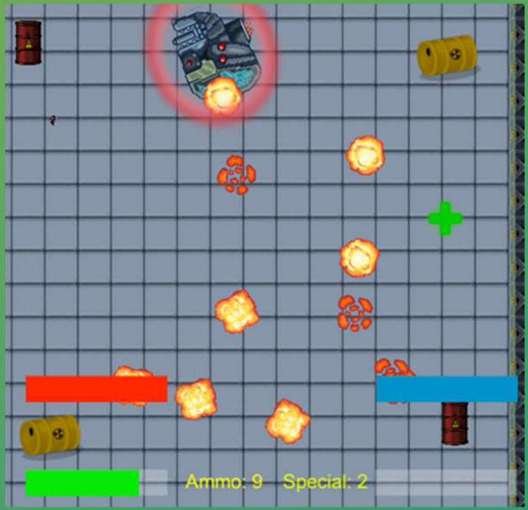


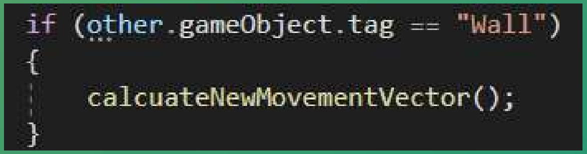
```
Vector2 dirPlayer = new Vector2(player.position.x - transform.position.x, player.position.y - transform.position.y);
    if (Vector2.Angle(dirPlayer, transform.right) < viewAngle / 2)
    {
        float distancePlayer = Vector2.Distance(transform.position, player.position);
        if(!Physics2D.Raycast(transform.position, dirPlayer, distancePlayer, obstacleMask))
        {
            visiblePlayer.Add(player);
            transform.position = Vector2.MoveTowards(transform.position, player.position, speed * Time.deltaTime);
        shoot.Shoot();
        }
    }
```

AI SYSTEM – BOSS 1

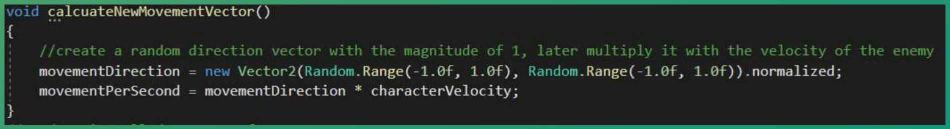
- Boss one randomly generates physics based movement
- Clamped in -1 and 1, the boss will be pushed in a random direction while firing in the players direction
- The bosses shield can only be broken by “special ammo” – red ammo boxes, the shield will regenerate after the boss takes 2 damage (cannot take damage while the shield is active)
- Issue: the boss would get stuck in the corner
- Solution: created a new tag to calculate a movement vector when it ran into a wall and send it in the general direction of the player







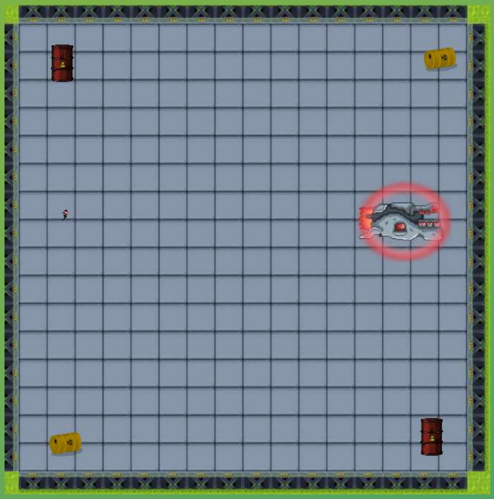
```
if (other.gameObject.tag == "Wall")
{
    calcuateNewMovementVector();
}
```

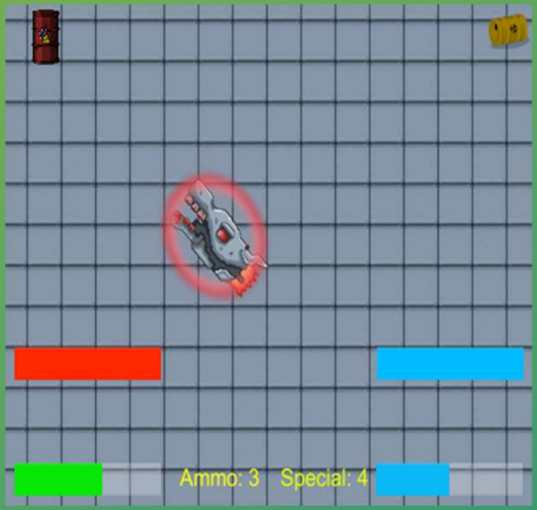


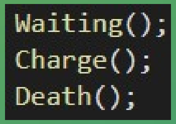
```
void calcuateNewMovementVector()
{
    //create a random direction vector with the magnitude of 1, later multiply it with the velocity of the enemy
    movementDirection = new Vector2(Random.Range(-1.0f, 1.0f), Random.Range(-1.0f, 1.0f)).normalized;
    movementPerSecond = movementDirection * characterVelocity;
}
```

AI SYSTEM – BOSS 2

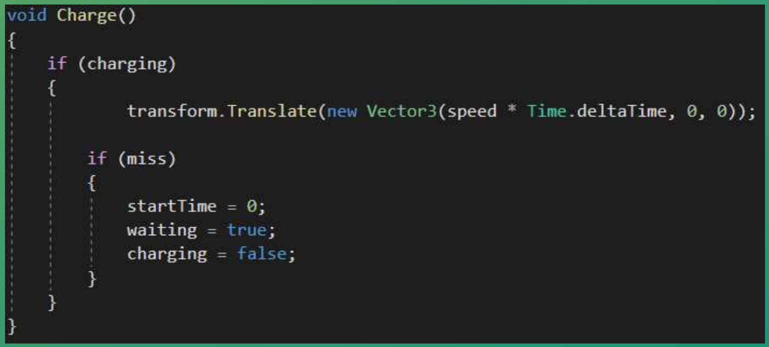
- Boss 2 will charge at the player after a set waiting period (editable)
- Boss 2’s speed will increase as his health gets lower
- The player must stay on a red ammo box for 2 seconds to get special ammo to destroy the bosses shield
- Simple flow of methods: waiting, charge, and death
- Use booleans, timers, and public (exposed to editor) variables to allow ease of use for designers







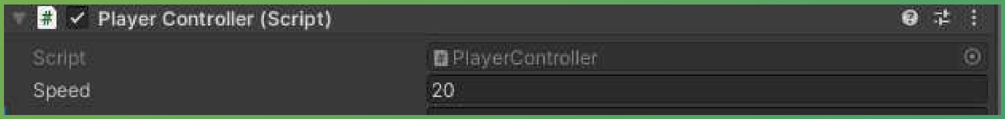
```
Waiting();
Charge();
Death();
```



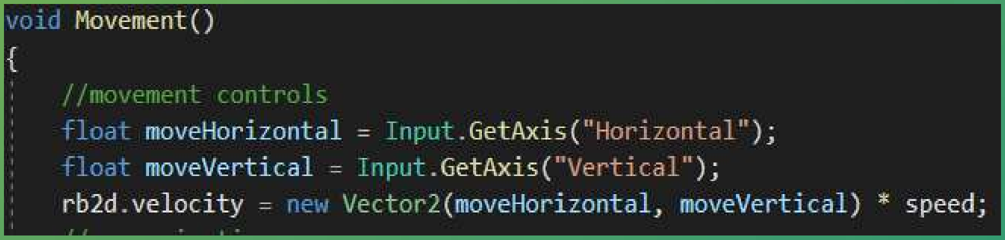
```
void Charge()
{
    if (charging)
    {
            transform.Translate(new Vector3(speed * Time.deltaTime, 0, 0));

        if (miss)
        {
            startTime = 0;
            waiting = true;
            charging = false;
        }
    }
}
```

MOVEMENT SYSTEM

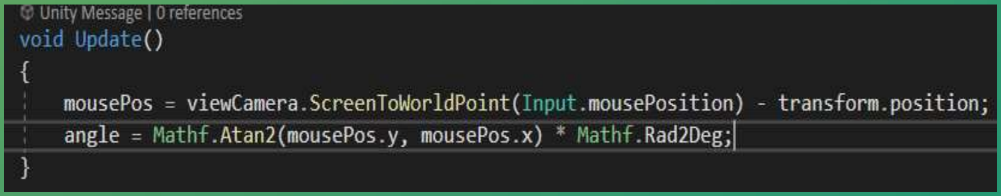
- Physics based movement system
  - Low mass and drag to give player a twitchy feeling
- WASD keys for player movement
  - Only speed exposed in the editor for design simplicity
- ScreenToWorldPoint in “RotateMouse” class point weapon
- Desired look and feel of Hotline Miami



```
Player Controller (Script)
Script    PlayerController
Speed     20
```



```
void Movement()
{
    //movement controls
    float moveHorizontal = Input.GetAxis("Horizontal");
    float moveVertical = Input.GetAxis("Vertical");
    rb2d.velocity = new Vector2(moveHorizontal, moveVertical) * speed;
```

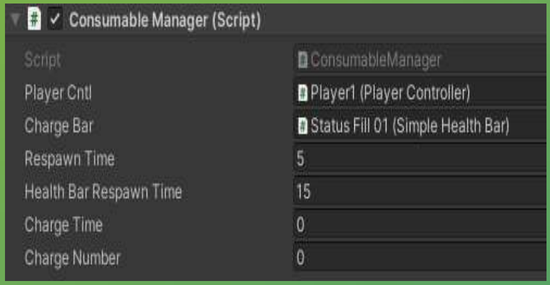


```
Unity Message | 0 references
void Update()
{
    mousePos = viewCamera.ScreenToWorldPoint(Input.mousePosition) - transform.position;
    angle = Mathf.Atan2(mousePos.y, mousePos.x) * Mathf.Rad2Deg;
}
```

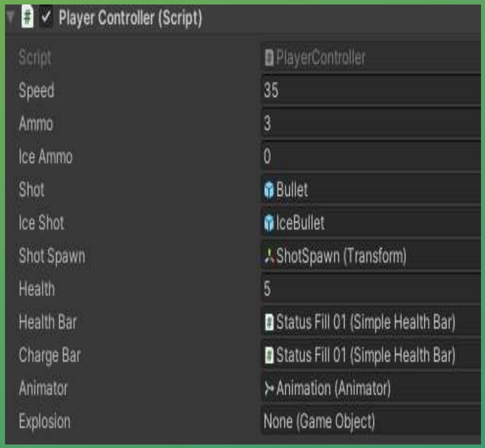
TOOLS OVERVIEW

Below lists the features that were programmed to use in the editor, some with limits, to give a designer realistic flexibility

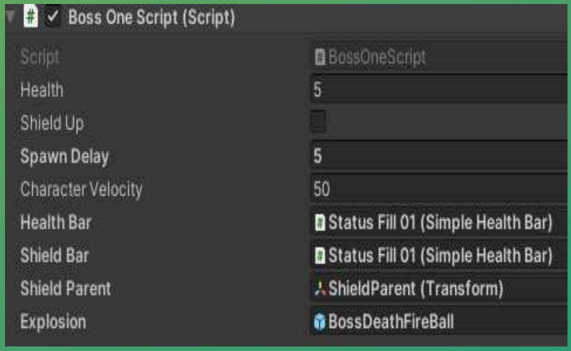
- Consumables
  - Type, duration of respawn, charge time
- Player Controller
  - Speed, fire rate, ammo counters
- Boss 1 Controller
  - Health, shield status, speed, boss spawn delay
- Boss 2 Controller
  - Movement change timer, speed, movement multiplier



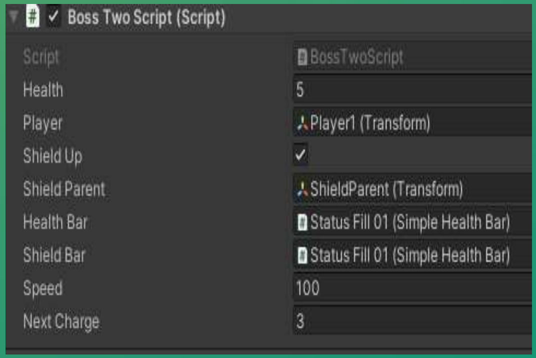
# ✓ Consumable Manager (Script)	
Script	ConsumableManager
Player Cntl	Player1 (Player Controller)
Charge Bar	Status Fill 01 (Simple Health Bar)
Respawn Time	5
Health Bar Respawn Time	15
Charge Time	0
Charge Number	0



# ✓ Player Controller (Script)	
Script	PlayerController
Speed	35
Ammo	3
Ice Ammo	0
Shot	Bullet
Ice Shot	IceBullet
Shot Spawn	ShotSpawn (Transform)
Health	5
Health Bar	Status Fill 01 (Simple Health Bar)
Charge Bar	Status Fill 01 (Simple Health Bar)
Animator	Animation (Animator)
Explosion	None (Game Object)



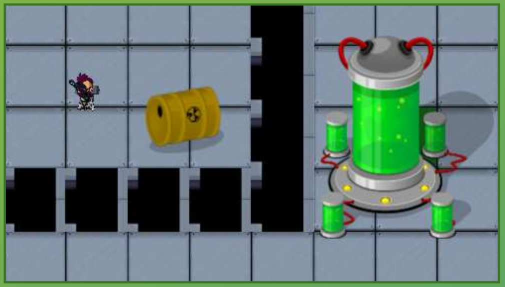
# ✓ Boss One Script (Script)	
Script	BossOneScript
Health	5
Shield Up	☐
Spawn Delay	5
Character Velocity	50
Health Bar	Status Fill 01 (Simple Health Bar)
Shield Bar	Status Fill 01 (Simple Health Bar)
Shield Parent	ShieldParent (Transform)
Explosion	BossDeathFireBall

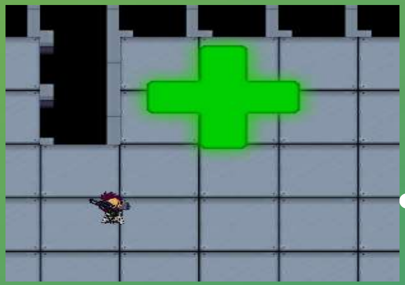


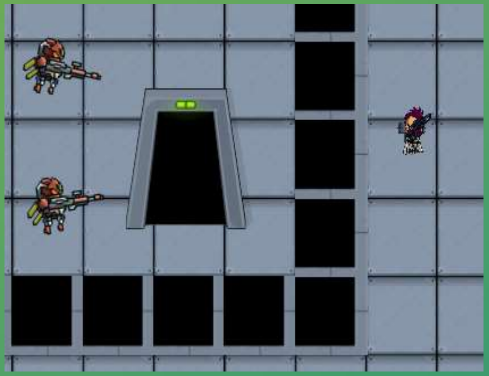
# ✓ Boss Two Script (Script)	
Script	BossTwoScript
Health	5
Player	Player1 (Transform)
Shield Up	✓
Shield Parent	ShieldParent (Transform)
Health Bar	Status Fill 01 (Simple Health Bar)
Shield Bar	Status Fill 01 (Simple Health Bar)
Speed	100
Next Charge	3

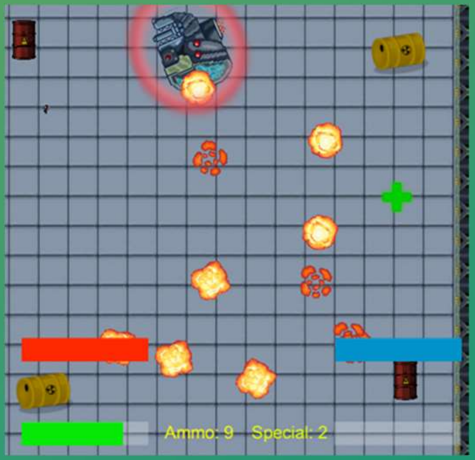
GAMEPLAY PROGRAMMING

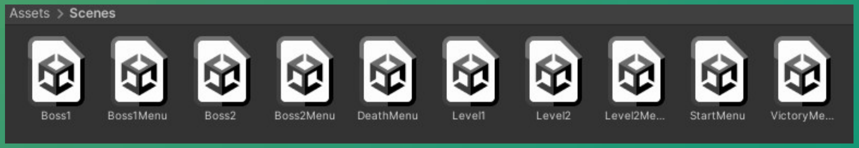
- Coroutines used for all respawnable items (ammo, health, special)
- Listeners on each script for the player to interact – allowed logic to exist on multiple compartmentalized scripts, helped performance
- Battle tracked all health/ammo of player and enemies on the GameController and broadcasted to the UIController





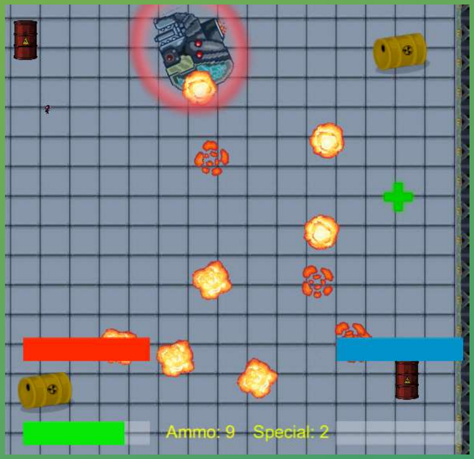


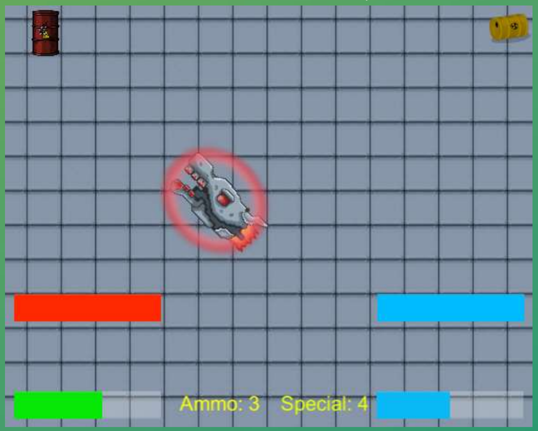


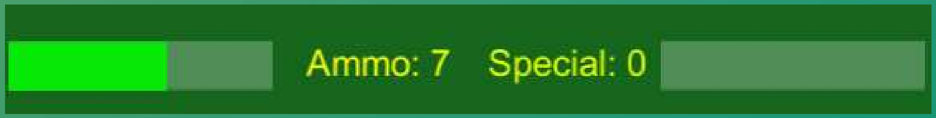


UI MANAGER

- Clamped to the bottom of the screen
- Delegates attached to each UI component to update whenever the game is updated
- Fill lines for health, shield, and special ammo all dynamically scale to the values associated
- All UI components are generally in the players FOV so they can focus on game mechanics







CONTACT INFORMATION

- Github source code
  - https://github.com/john-mccoubrie
- Portfolio Link
  - Check out my other projects here
  - https://john-mccoubrie.github.io/
- Itch.IO
  - Playable game
  - https://johnmccoubrie.itch.io/
- Reach out
  - https://www.linkedin.com/in/john-mccoubrie/
  - Email: Jmccoubrie4@gmail.com
  - Phone: (610) 908-6115
- Assets used
  - Character editor: https://assetstore.unity.com/packages/2d/characters/character-editor-megapack-116375
  - 2D asset pack: https://assetstore.unity.com/packages/2d/gui/icons/2d-pixel-item-asset-pack-99645
  - Enemies: https://assetstore.unity.com/packages/2d/environments/robot-shooting-game-sprite-free-93902
  - Sci-fi: https://assetstore.unity.com/?q=2d%20sci-fi&orderBy=1