Hit a Moving Target - Diary

- Started a 2D Unity project
- Created a square sprite in unity
- Placed and scaled square sprites down on the scene view to use as a play area background, to use as play zone barriers attaching colliders, and obstacles connecting colliders, colouring them different colours to stand out from each other
- Grabbed a free tank sprite asset pack online via craftpix.net
- Placed down the tank sprites, on for the player and one for the enemy, and set their pixel per unit (128 PPU) half of their resolution (256x256), adding colliders and a Rigidbody2D component to the player tank with 0 gravity to prevent it falling for a top-down view, and freezing its rotation
- Attached empty game objects to the tanks and placed them in the centre of their gun ball joints to use as an anchor point for the guns
- Attached the gun sprites to the gun anchor points, centring their rotation points with the gun anchor
- Created prefabs out of the tank game objects I created

Player movement set up

- Created a script 'PlayerMovement' to control the player tank behaviour
- Added two float variables 'moveSpeed' and 'rotateSpeed' as Serializable Fields to make them editable in the inspector properties, a Rigidbody2D variable 'rb' to reference the Rigidbody, and two additional float variables 'horiInput' and 'vertInput' to hold input values
- Referenced the player tank object's Rigidbody in the start method set to 'rb'
- Started a new function 'GetInput()' to handle the input, setting 'horiInput' to the raw horizontal axis input, and 'vertInput' to the raw vertical axis, calling this function in the Update method
- Started another function 'MovePlayer()' to handle the player tank's movement, using 'rb' I set the Rigidbody velocity to move forward using 'transform.up' times by the 'vertInput' value times by the 'moveSpeed' value
- Formed another function 'RotatePlayer()' to handle the player tank's rotation, first I declared a float 'rotation' setting this to the 'horiInput' value times 'rotateSpeed', then using the 'transform.Rotate' method to apply the rotation with 'Vector3.forward' times 'rotation'
- Inside the 'PlayerMovement' script's inspector properties I set the 'moveSpeed' and 'rotateSpeed' values to 5
- Testing the player tank moves forward and back, but the rotation is flipped, and the tanks speeds seem a little too fast
- Inside the 'RotatePlayer()' function I set the 'horiInput' value being set in 'rotation' to a negative to flip the rotation
- Set the 'rotateSpeed' and 'moveSpeed' to 3 in the 'PlayerMovement' script's inspector properties
- Testing the player tank now rotates in the correct rotation and the speeds feel better

Enemy Tank aim set up

- Started a new script 'EnemyAI', declaring a GameObject variable 'player' to hold the player GameObject reference, a GameObject 'gunAnchor' as a Serialize Field to reference the enemy tank's gun anchor, and a Vector2 'playerDir' to hold the player's direction
- In the start method I reference the player tank by finding it by its 'Player' tag, and in the update method I set the 'playerDir' to the player's transform position minus the enemy tank's position to calculate the direction to the player tank from the enemy tank
- Started a new function 'AimAtTarget()' to handle the enemy tank's aim behaviour, setting the gun anchor's 'transform.up' attribute to 'playerDir', calling this function in the update method
- Testing the tank aims at the player as expected, but also through obstacles
- Started a bool function 'CanSeePlayer()' to provide a true/false value if the player can be seen by the enemy tank without obstacles in the way, started with a RaycastHit2D variable 'hit' setting this to

Raycast from the enemy position in the player direction, then using an if to check if the Raycast has hit the player tank's collider, if true returning false from the function with an else returning false

- Back in the 'AimAtTarget()' function I surrounded its current code with an if checking the 'CanSeePlayer()' function is true
- Testing the enemy tank is not aiming at the player at all
- Back in the 'CanSeePlayer()' function I use a debug draw line to see if the Raycast is aiming in the correct direction
- Testing the debug line show the Raycast is aiming in the correct direction
- Back in 'CanSeePlayer()' I use a debug log to output the Raycast hit tag value to see what the Raycast is hitting
- Testing it seem the Raycast is constantly hitting and being blocked by the enemy tank's collider
- I set the enemy tank's Layer to 'Ignore Raycast' based off advice I found online
- Testing the Raycast is no longer being blocked by the enemy tank collider, and the tank is correctly aiming at the player if it is not blocked by an obstacle collider, but its gun aims suddenly without smoothing as I would like
- I declare a float variable 'gunTurnSpeed' to the 'EnemyAI' script to control the gun's turn speed
- Inside the 'AimAtTarget()' function, within its if I declare a float 'step' to regulate the gun's turn speed based on performance, I set this to 'gunTurnSpeed' times 'Time.deltaTime', then I replace and set the gun anchors 'transform.up' property to a 'Vector2.MoveTowards' using the gun anchors current up position, the 'playerDir' as the target, with its speed set to 'step'
- Inside the inspector settings I set the 'gunTurnSpeed' to 10
- Testing the enemy tank's gun now aims smoothly

<u>Bullet prefab setup</u>

- I place a bullet sprite and set its Pixel Per Unit to correctly size it on the scene, attaching a collider and a Rigidbody 2D with 0 gravity
- I create and attach a script 'BulletBehaviour' to the bullet game object, declaring a public bool 'exploded' with a false default value, and an Animator 'anim' to reference its animator component
- In the start method I reference the animator in 'anim' using 'GetComponent()'
- Next, I drag a range of explosion sprites onto the scene to create an explosion animation, setting the sprites Pixel Per Unit to be correctly sized on the scene, and turning off the animation loop
- Then, I create a new animation with the bullet selected, leaving the animation blank for an idle state of the bullet, and renaming its animation controller
- Inside the bullet's animation controller, I create a new trigger parameter 'Hit', drag in the explosion animation to the controller and make a transition between the bullet's idle state and the explosion state with a condition checking for the 'Hit' trigger
- Back on the bullet's animation pane with the explosion animation selected I add a property setting the 'exploded' bool of 'BulletBehaviour' to true when animation had finished
- Within the 'BulletBehaviour' script I use an 'OnCollisionEnter2D' method to call the animation's 'Hit' trigger when it collides with anything, and inside the update method I use an if to check 'exploded' is true to destroy the bullet's game object
- Finally, I turn the bullet into a prefab

<u>Enemy tank shooting</u>

- I attach an empty game object to the tank prefabs guns 'ShootPos' to use as a position to shoot from, placing this in front of the gun barrels
- Back inside the 'EnemyAI' script I declare a float 'bulletSpeed' as a SerializeField to control the bullet speed, a GameObject variable 'bullet' to reference the bullet prefab as a SerializeField, a GameObject 'shootPos' to reference the 'ShootPos' game object as a SerializeField, a float 'startTimeBtwShots' to manage the fire speed of the enemy tank as a SerializeField, and a float 'timeBtwShots' to use as the fire speed timer

- Within the start method I set the 'timeBtwShots' value to 'startTimeBtwShots' to set the timer value it should count down from
- I start a new function 'Shoot()' starting with an if checking 'timeBtwShots' is lower or equal to 0, if true instantiating a bullet at the 'shootPos' position declaring the spawned bullet under an new GameObject variable 'newBullet'; using the reference to set it's 'transform.up' position to the gun anchors 'transform.up' position, setting the bullet's Rigidbody velocity to the gun anchor's 'transform.up' times by the 'bulletSpeed' value, lastly resetting the 'timeBtwShots' countdown timer to 'startTimeBtwShots'. Then using an else to countdown the timer by taking away 'Time.deltaTime' from 'timeBtwShots'
- Back in the 'AimAtTarget()' function I call the 'Shoot()' function inside its if checking 'CanSeePlayer()' is true
- Testing the enemy tank shoots, but the bullet explodes straight away after being spawned
- Inside the 'BulletBehaviour' script I declare a public string 'goIgnore' to take the name of the game object it should ignore when triggering its explosion, surrounding an if around the code of 'OnCollisionEnter2D' to check if the collision game object is not 'goIgnore'
- In the 'EnemyAI' script, after instantiating a bullet I set the bullet's 'goIgnore' value to the enemy tank's game object name using 'gameObject.name'
- Testing the bullet now shoots and explodes when hitting colliders, but the tank aims and shoots directly in the player tank's direction even when it moves, it needs to predict the players future position whilst moving to successfully hit the player

<u>Enemy Tank aim prediction</u>

- Searching online I find a code snippet of a function that predicts a position based on velocity and projectile speed
- I paste the function 'predictedPosition' into the 'EnemyAI' script converting its code from Vector3 to Vector2 values to be compatible with my code
- Inside the 'AimAtTarget()' function I replace the 'Vector2.MoveTowards' target argument to use the 'predictedPosition' function with it required arguments; minus the enemy tank's position to convert it to a direction
- Testing the tank now shoots in front of the player whilst it is moving successfully scoring a hit

<u>Player Tank Aim & Shoot</u>

- Started a new script 'AimShoot' script, started by declaring two Vector2 variables, 'direction' holding a direction, and 'mousePos' holding the world coordinates the mouse cursor is positioned at, and further Serializable Fields to be alterable from the inspector including a GameObject variable 'bullet' to reference the projectile prefab, a float 'bulletSpeed' to control the projectile speed, a GameObject 'player' to reference the player game object, a GameObject 'gunAnchor' to reference the gun anchor game object to manipulate, and a Transform 'shootPos' to reference a game object's transform for a shoot position
- In the update method I set the 'mousePos' to get a world point from the camera view using the mouse position, then setting 'direction' to 'mousePos' minus the gun Anchor's transform position to calculate a direction vector
- Formed a new function 'FaceMouse()' to control the aim direction of the player gun, inside setting the gun anchor's 'transform.right' to the 'direction' vector, finally calling this function in the update method
- Testing the player gun now aims in the mouse cursors direction
- Added a new function 'Shoot()' in the 'AimShoot' script, recycling some code from the 'EnemyAI' script to instantiate the projectile, applying the collider to ignore, direction and velocity
- In the update function I used an if to detect the left mouse being clicked, if so calling the 'Shoot()' function to fire a projectile
- Testing the player tank now shoots a projectile correctly

<u>Health System</u>

- I placed and scaled a square sprite as a bar, setting its colour to grey with half opacity for a health bar background, then duplicated this on top of it setting its colour to red with half opacity and made it a child of the first bar for a current health
- Scaling the current health bar, I found it scales itself from the middle
- Instead, I add an empty game object 'HealthScaler' to health bar background, placing this at the left edge of the bar, then I made the current red health bar a child of the empty game object placed on the edge of the health bar background, using the empty game object scale to scale the current health bar from right to left instead of from the middle previously
- Applied the health bar to both tank prefabs
- I start a new script 'Health', declaring an int 'maxHealth' with a default value of 100 as a Serializable field, an int 'health' to hold the health value, a private int 'lastHealth' to hold the last recorded health value, a Transform 'healthBar' as a SerializeField to reference the game object responsible for scaling the health bar, and a public bool alive with a private set modifier
- In the start method I set 'health' to 'maxHealth' to provide the health value with its maximum set health, set 'lastHealth' to health, 'alive' to true
- Created a new function 'HealthSys()' to handle the health system, starting with an if checking 'health' is lower than 0; if so setting 'health' back to 0 to prevent lower values and the health bar scaling in it opposite direction, another if checking 'health' is bigger than 'maxHealth'; if so setting 'health' to 'maxHealth' to prevent greater values than its maximum health, and a last if checking 'health' is not equal to 'lastHealth' meaning there was a change in health, if true updating 'lastHealth' to the new 'health' value, calling this function in the update method
- Formed another function 'UpdateHealthBar()' to handle the health bar scale, starting by declaring a float 'healthScale' and setting it to 'health', then dividing the 'healthScale' by 100, lastly applying the 'healthScale' to the local scale x of 'healthBar' to scale the health bar to the current health, finally calling this function in the last if of 'HealthSys()' checking for a health change
- Started another public function 'Damage()' taking an int argument 'damageVal', inside taking away the 'damageVal' from 'health'
- Added another public function 'AddHealth()' taking an in argument 'healVal', inside adding 'healVal' to 'health'
- Created another function 'Death()', inside setting 'alive' to false, calling this function in the first if of 'healthSys()' checking 'health' is lower than 0
- Attached the 'Health' script to both tank prefabs, referencing the game objects scaling the health bar 'HealthScaler'
- Inside the 'BulletBehaviour' script I declared ints 'hitDamageMin' and 'hitDamageMax' as SerializeFields, and an int 'hitDamage', then in the start method I set a random value between 'hitDamageMin' and 'hitDamageMax', next I added an if inside the 'OnCollisionEnter2D' method to check in the collided object has a 'Health' script attached, if so calling the 'Health' script's 'damage' function using 'hitDamage' as the argument
- Inside the 'bullet' prefab I set some damage values for the bullet to randomise between
- Testing the bullets apply damage to the tanks health and correctly scale the health bar relative to the current health value

<u>Health Bonus consumable</u>

- Placed a sprite from the free asset pack onto my scene, attaching a box collider 2D as a trigger, scaling it slightly bigger than the sprite, creating a prefab out of it 'HP_Bonus'
- In the animation pane I create an animation to give the 'HP_Bonus' a pulsing effect, and another fading out its sprite for it has been collected, adding a trigger ' Vanish' to its animator controller, making a transition between the pulsing and vanish animation; using the 'Vanish' trigger as its condition

- Created and attached a new script ‘HPBonus’ to the ‘HP_Bonus’ prefab, declaring two Serializable Field ints ‘healAmountMin’ and ‘healAmountMax’, an int ‘healAmount’, a bool ‘isVanished’ with a false default value as a SerializeField, and an Animator variable ‘anim’ to reference an animator
- In the start method is set the ‘healAmount’ to a random int between the ‘healAmountMin’ and ‘healAmountMax’ values, and referenced the animator component under ‘anim’
- Using an ‘OnTriggerEnter2D’ method, I used an if to check the object entering the trigger had a ‘Health’ script attached, if so calling the ‘AddHealth()’ function of the ‘Health’ script, using ‘healAmount’ as its argument, then calling the animator trigger ‘Vanish’ to transition to the vanish animation state
- Added a private function ‘Vanish()’, inside checking if ‘isVanished’ is true, if so destroying the game object, if not recalling the function until the statement is true
- In the vanish animation of the ‘HP_Bonus’ I added a property enabling the ‘isVanished’ bool at the end of the vanish animation to destroy the game object

Scene spruce up

- I added tracks to the tank prefabs, using a script on the player tank prefab ‘TankAnim’ to animate movement on the track if it was moving
- I imported another free asset pack from craftpix.net to add appealing tile maps and props to scene
- I added a bool prop to the scene with a circle collider and Rigidbody set to 0 gravity, including a physics material to make it bouncy, creating a prefab out of it