Paloma Resendiz

Dr. Lehr

CS 3398

February 21, 2024

Unity Movement Research

1. Event listeners: write code to detect when certain keys are pressed on the keyboard (arrow keys)
2. Write code to update position of the object’s x and y coordinates by a certain amount when the arrow keys are pressed
3. document.addEventListener('keydown', function(event) {  
   if (event.code === 'ArrowUp') {  
   player.y -= player.speed;  
   }  
   else if (event.code === 'ArrowDown') {  
   player.y += player.speed;  
   }  
   else if (event.code === 'ArrowLeft') {  
   player.x -= player.speed;  
   }  
   else if (event.code === 'ArrowRight') {  
   player.x += player.speed;  
   }  
   });
4. Can use (Input.GetKey(KeyCode.UpArrow)) as a second option for movement
5. Later add animations for character walking left or right
6. anim = GetComponent<Animator>();

if (Input.GetKey(KeyCode.RightArrow))

{

anim.Play("Walking");

transform.position += Vector3.right \* speed \* Time.deltaTime;  
}

if (Input.GetKey(KeyCode.LeftArrow)

{

anim.Play("Walking");

transform.position += Vector3.left \* speed \* Time.deltaTime;

}

1. Using Unity’s physics is the easier approach
2. Can get basic movement up and running in a couple of lines of code
   1. Drawback is the approach won’t feel great, mostly for prototyping
3. Custom physics is harder but gives more control over movement and feels more polished and professional
   1. Drawback is need to come up with tricky edge cases
4. Add Rigidbody 2D and Box Collider 2D components to player object
5. Adjust RigidBody 2D gravity to 0
6. You can add new scripts by clicking add component
   1. Float inputX = Input.GetAxis(“Horizontal”);

Float inputY = intput.GetAxis(“Vertical”);

Vector3 movement = new Vector3(speed.x \* inputX, speed.y \* inputY, 0);

Movement \*= Time.deltaTime;

Transform.Translate(movement);

1. Movement scripts go in void Update()
2. movementDirection = new Vector2(Input.GetAxis(“Horizontal”), Input.GetAxis(“Vertical”));