Dangerous Kave

Basic Box Mechanics

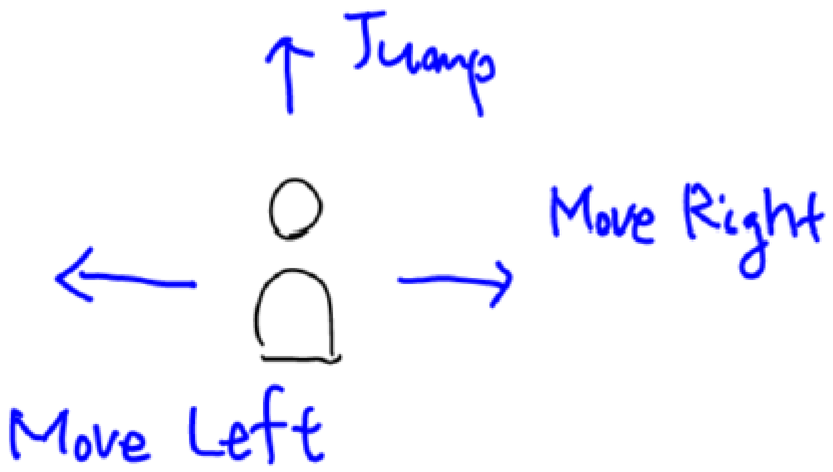
jintaeks@dongseo.ac.kr

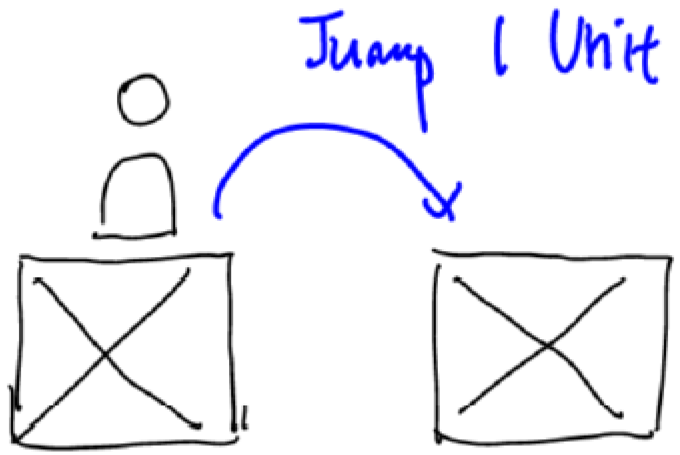
March 2020

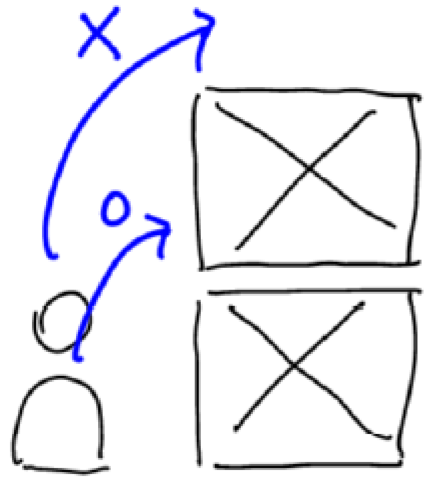
DSU Dongseo University
동서대학교

Game Mechanics: Character

✓ Character

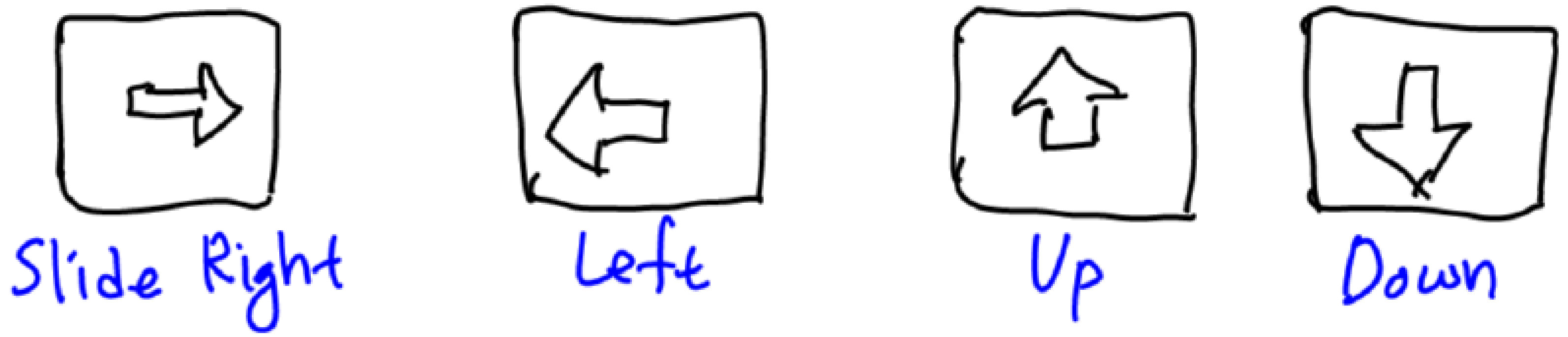




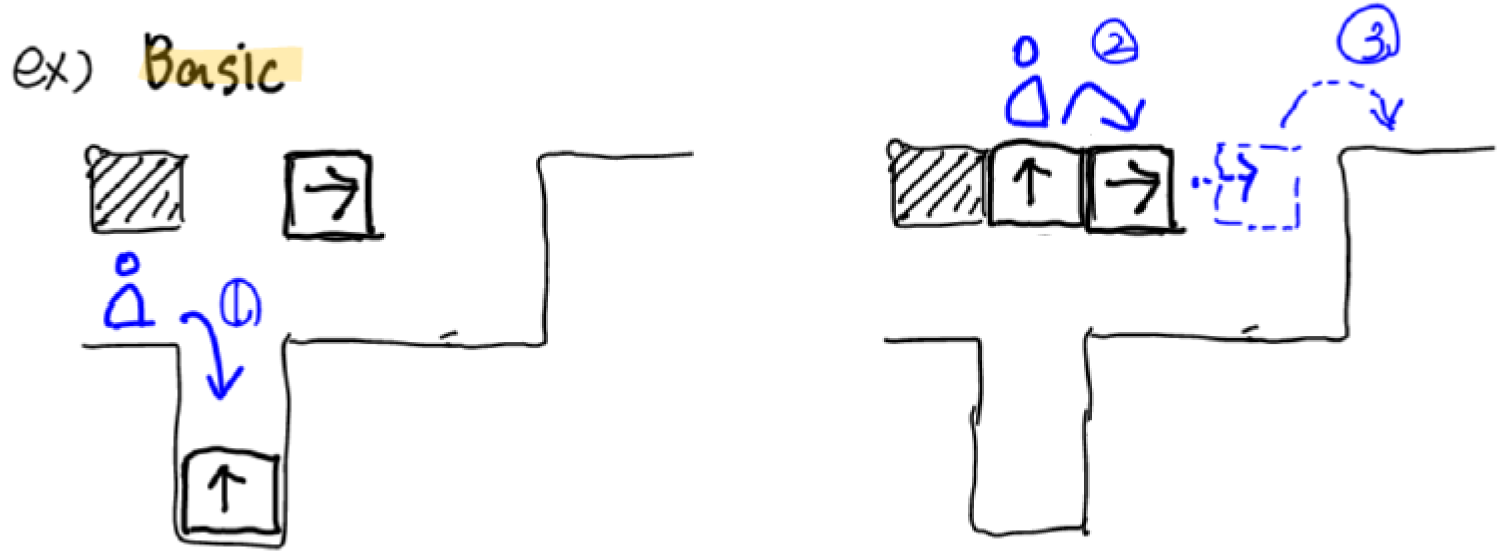


Game Mechanics: Box

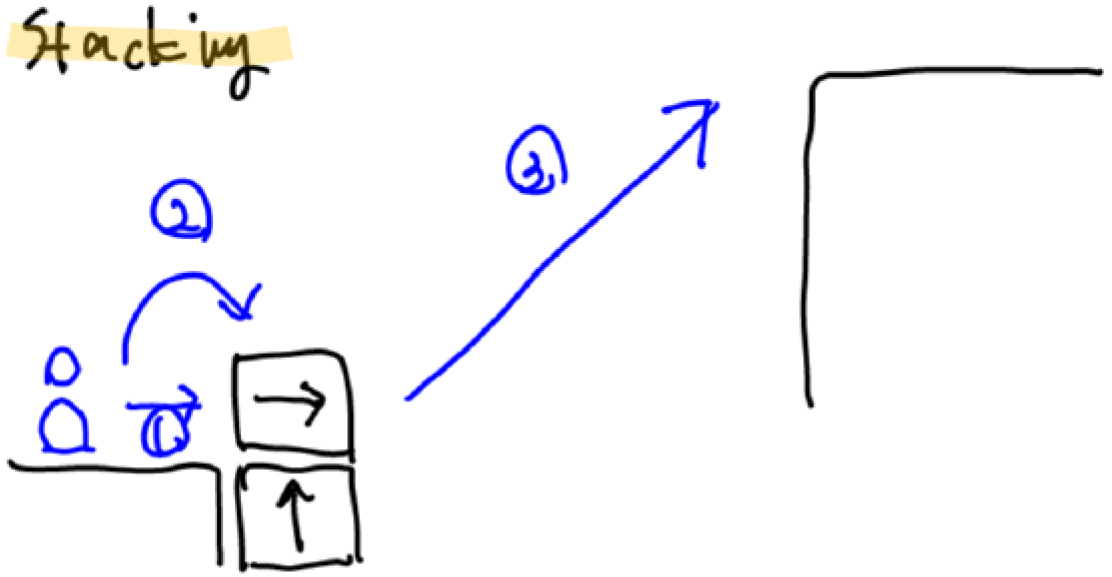
- ✓ Box



✓ Box→Basic



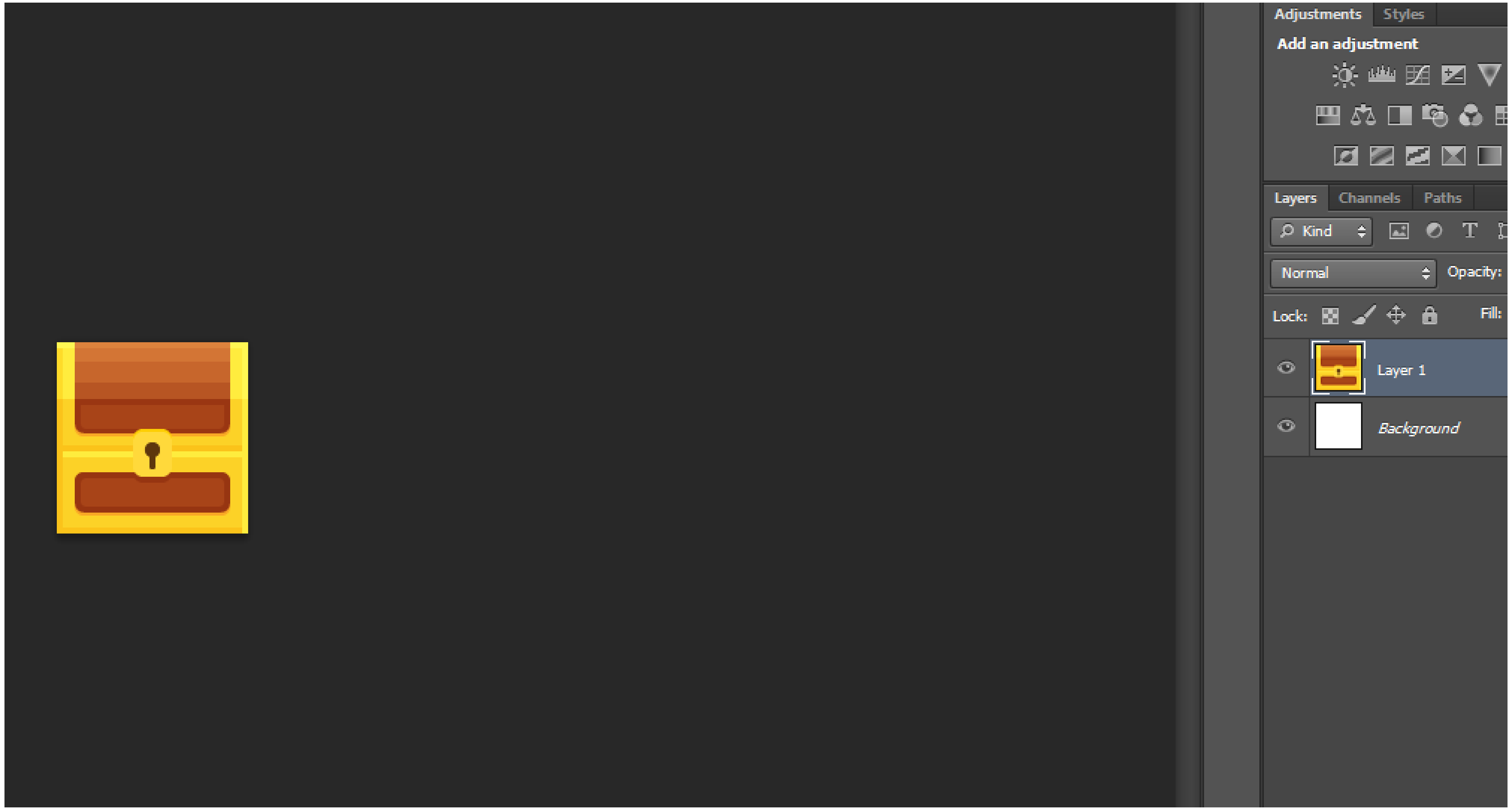
✓ Box->Stacking

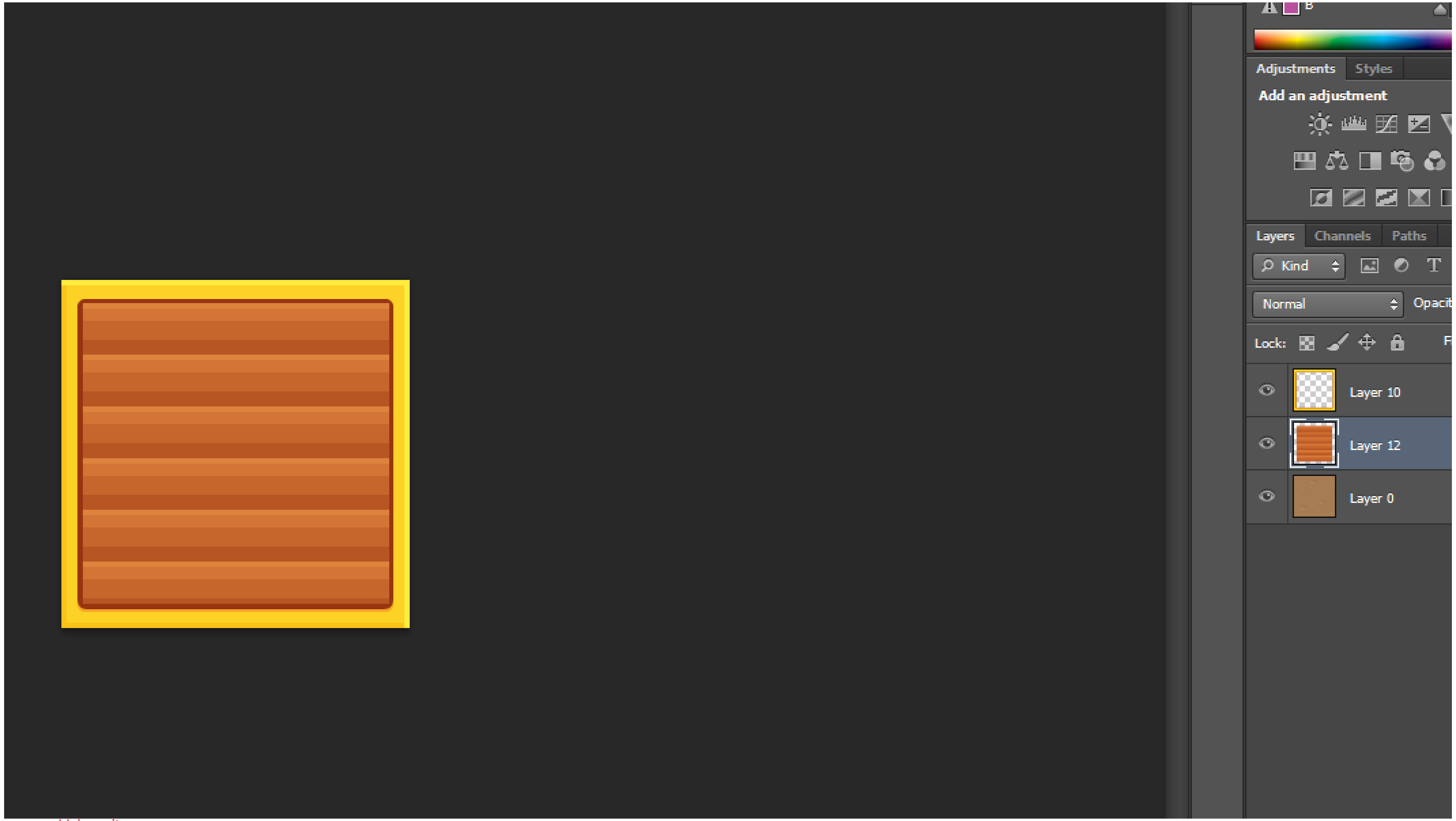


Divide and Conquer

✓ Box

- State
- Velocity(direction and speed)


Adjustments
Styles
Add an adjustment
Layers
Channels
Paths
Kind
Normal
Opacity:
Lock:
Fill:
Layer 1
Background


Adjustments
Styles
Add an adjustment
Layers
Channels
Paths
Kind
Normal
Opacit
Lock:
Layer 10
Layer 12
Layer 0

Add the following line of code in the Update method.

```
transform.Translate(velocity * Time.deltaTime);
```

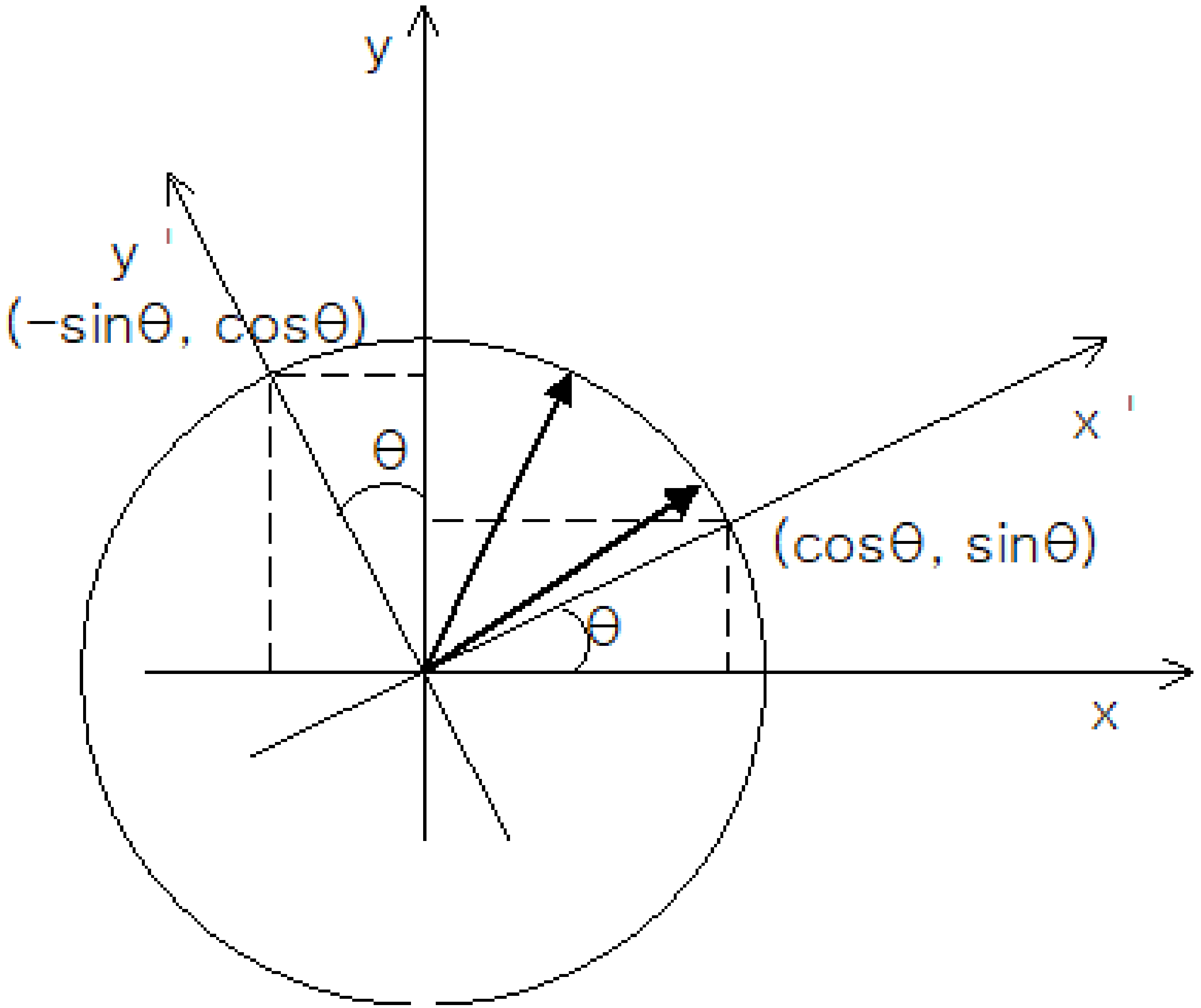
Our velocity isn't being modified yet, so our controller won't move. Let's change that by adding some horizontal velocity when the left or right keys are pressed.

```
float moveInput = Input.GetAxisRaw("Horizontal");
velocity.x = Mathf.MoveTowards(velocity.x, speed * moveInput,
walkAcceleration * Time.deltaTime);
```

Mathf.MoveTowards is being used to move our current x velocity value to its target, our controller's speed (in the direction of our sampled input).

$$\mathbf{s}(t) = \mathbf{s}_0 + \mathbf{v}_0 t + \frac{1}{2}\mathbf{a}t^2$$

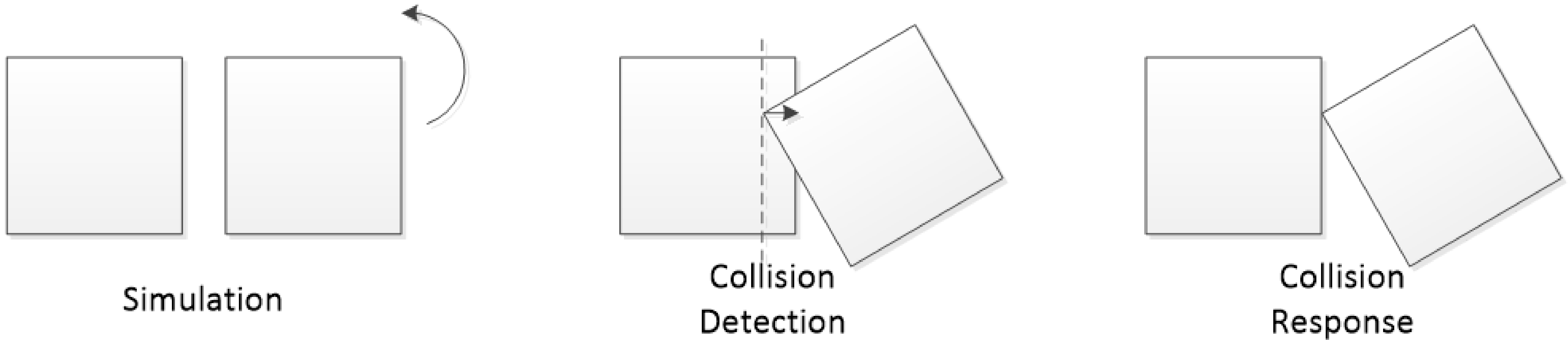
$$\mathbf{v}(t) = \mathbf{v}_0 + \mathbf{a}t$$

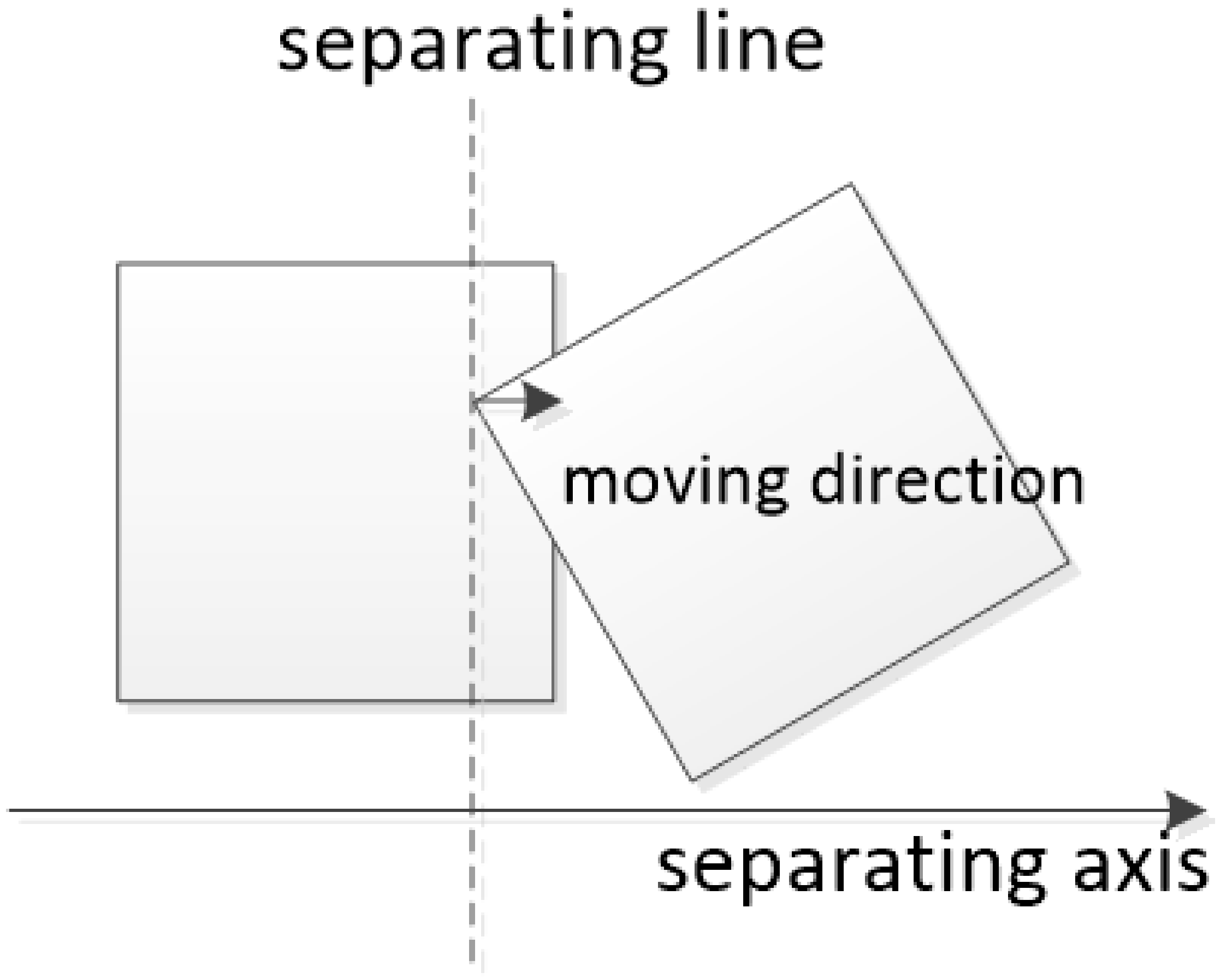

y
y '
(−sinθ, cosθ)
θ
x '
(cosθ, sinθ)
θ
x

$$x' = (\cos\theta, \sin\theta),\ \ y' = (-\sin\theta, \cos\theta)$$

$$Q' = \begin{bmatrix} a' \\ b' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$$

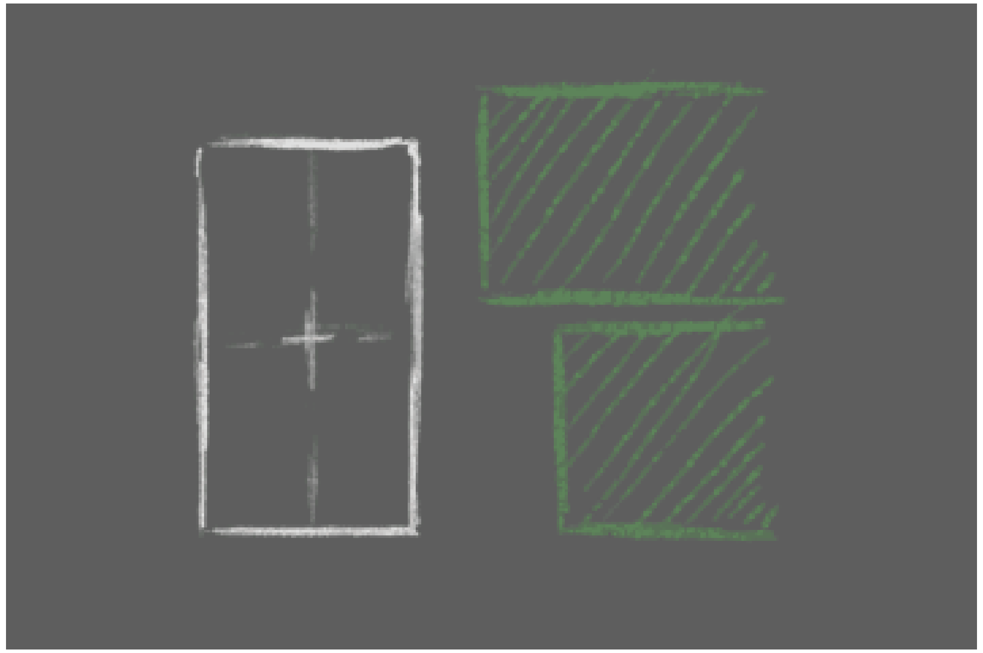
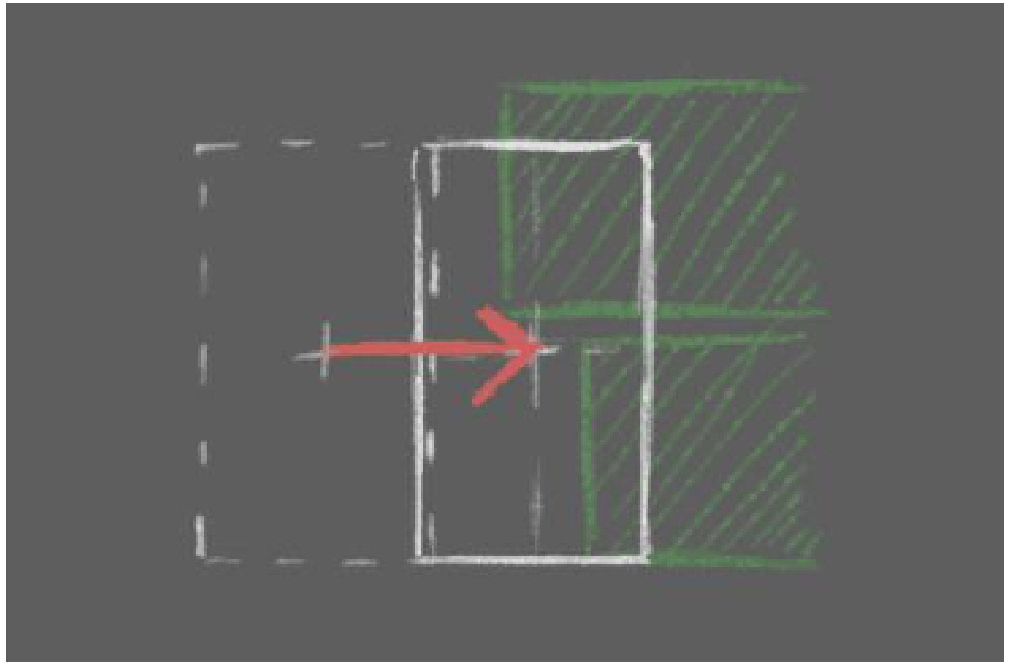
Coding Style and Method Layout

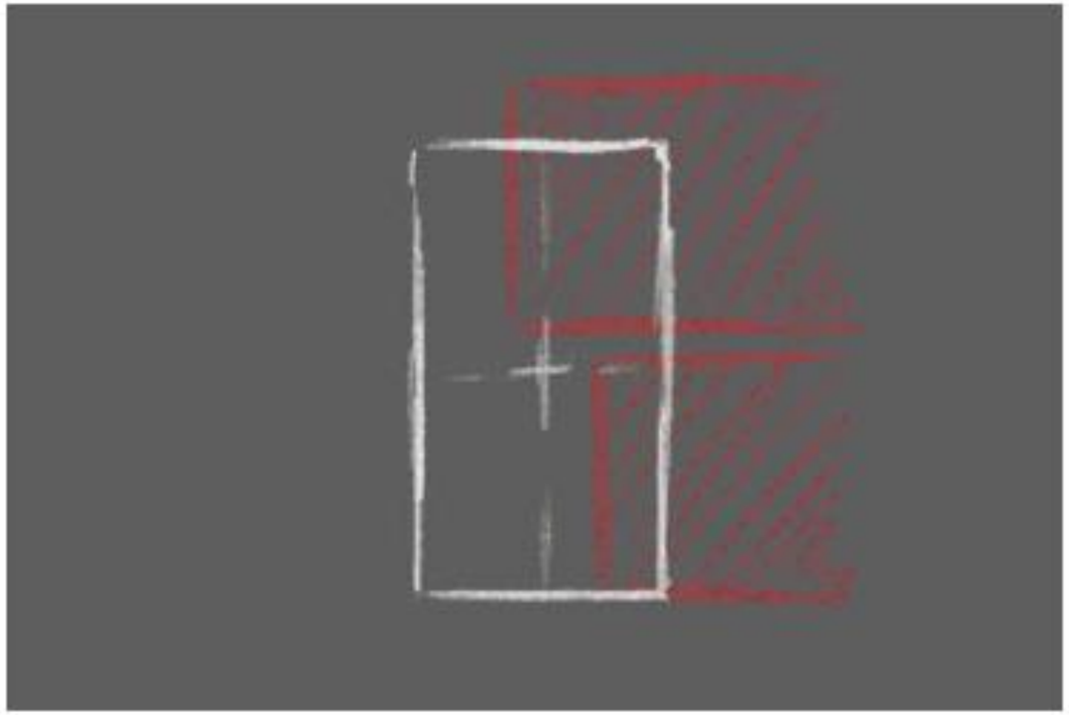
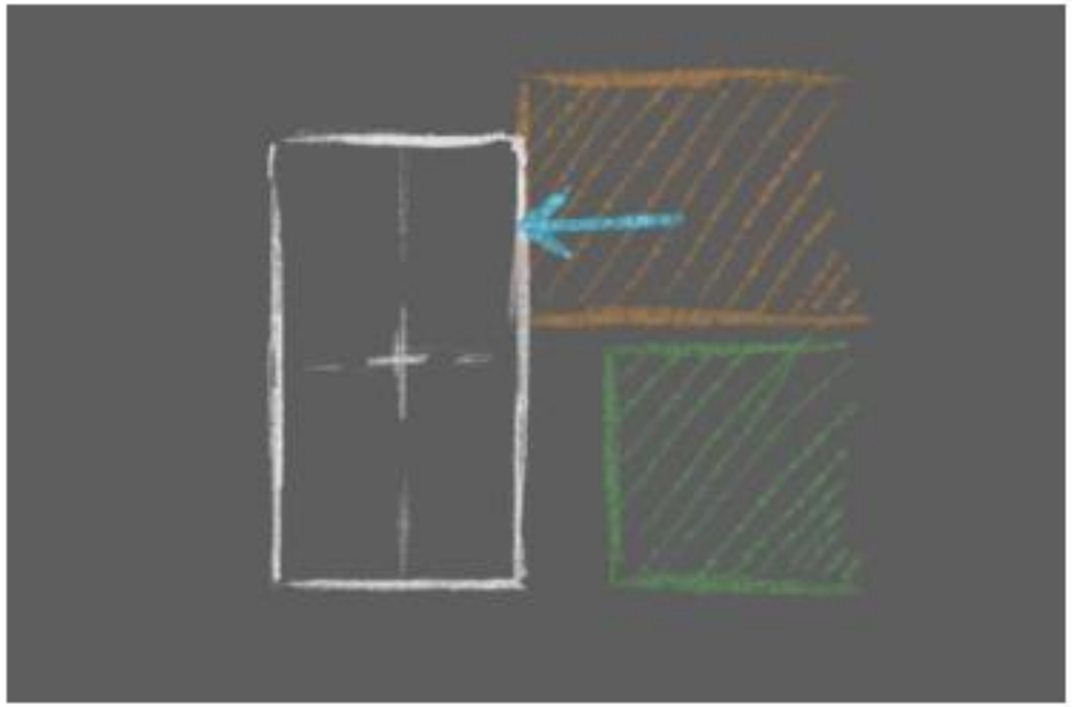

Simulation
Collision Detection
Collision Response


separating line
moving direction
separating axis

```
  Collider2D[] hits = Physics2D.OverlapBoxAll(transform.position,
boxCollider.size, 0);
```

This will give us an array of all colliders that are intersected by the box we defined, which is the same size as our `BoxCollider` and at the same position. Note that because of this, the array will also contain our own `BoxCollider`.

The main problem is to decide which direction, and how far, we need to translate our controller to depenetrate from each collider. Ideally, we should move it *the minimum distance required to be no longer touching the other collider*. Unity provides a method to find that distance for us, Collider2D.Distance.

```
  foreach (Collider2D hit in hits)
  {
      if (hit == boxCollider)
          continue;

      ColliderDistance2D colliderDistance = hit.Distance(boxCollider);

      if (colliderDistance.isOverlapped)
      {
          transform.Translate(colliderDistance.pointA - 
colliderDistance.pointB);
      }
  }
```

As noted above, the array will contain our own `BoxCollider`—we skip it during our foreach loop.

`ColliderDistance2D.isOverlapped`, tells us if the two colliders are touching. Once we have ensured they are, we take the `Vector2` from `pointA` to `pointB`. This is the shortest vector that will push our collider out of the other, resolving the collision.

Unity Builtin Attribute

✓ [Range(0,100)]

Draw Gizmos in Scene View

✓ OnDrawGizmos()

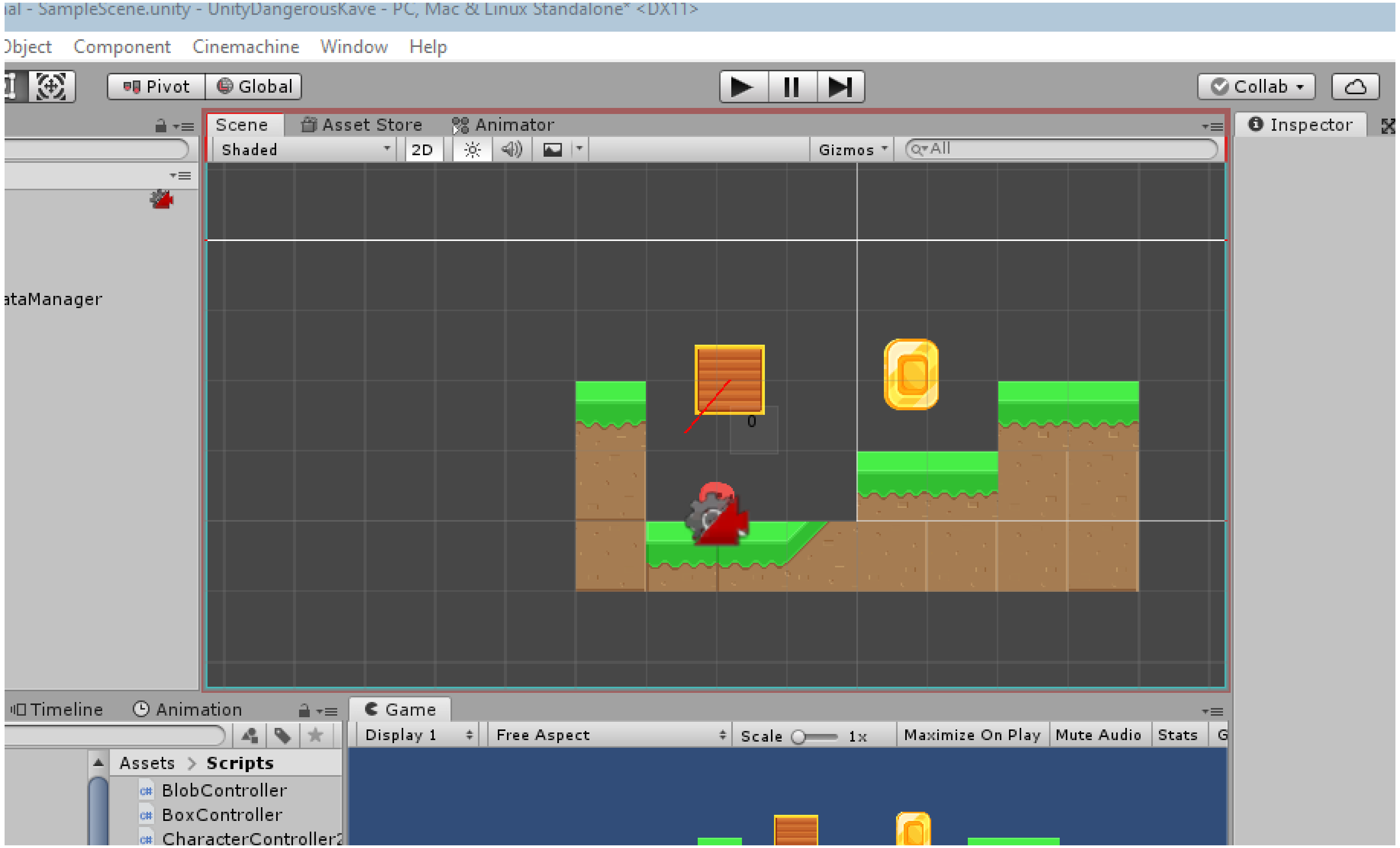
Dongseo University
동서대학교

To add State to the Box Mechanic

✓ DrawTextInSceneView()

- Exact position problem in Scene View

How I solved this problem.


ial - SampleScene.unity - UnityDangerousKave - PC, Mac & Linux Standalone* <DX11>
)bject Component Cinemachine Window Help
Pivot Global
Collab
Scene Asset Store Animator
Shaded 2D Gizmos All
Inspector
ataManager
0
Timeline Animation
Game
Display 1 Free Aspect Scale 1x Maximize On Play Mute Audio Stats G
Assets > Scripts
BlobController
BoxController
CharacterController2

Next

- ✓ How can we add states?
- ✓ A character on the box must move together, how can we impl ement it?

MY **BRIGHT** FUTURE

DSU Dongseo University
동서대학교