UNIT3

GAMEOBJECTS INTERACTIONS

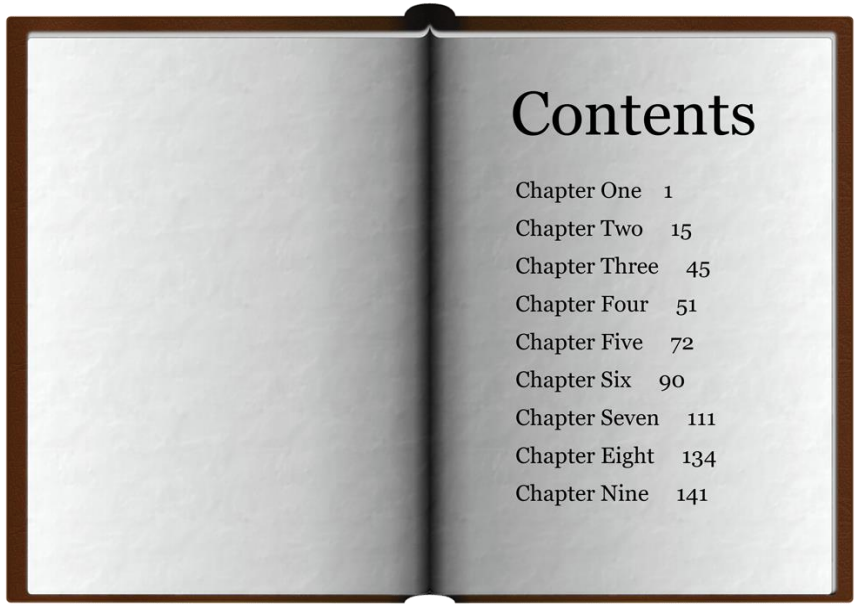
PMDM - 2DAM

Àngel Olmos (a.olmosginer@edu.gva.es)

Jose Pascual Rocher (jp.rochercamps@edu.gva.es)

Content

1. Introduction
2. Build the scene
3. Keep the player inbounds: IF statement
4. Prefabs
5. Launch projectiles: Instantiate()
6. Destroy Game Objects
7. Spawn manager: Arrays
8. Randomize spawning
9. Timed intervals: InvokeRepeating()
10. Collider and trigger components
11. Destroy on collision: OnTriggerEnter()
12. ACTIVITY



Introduction

In this lesson we will make the GameObjects interact between them

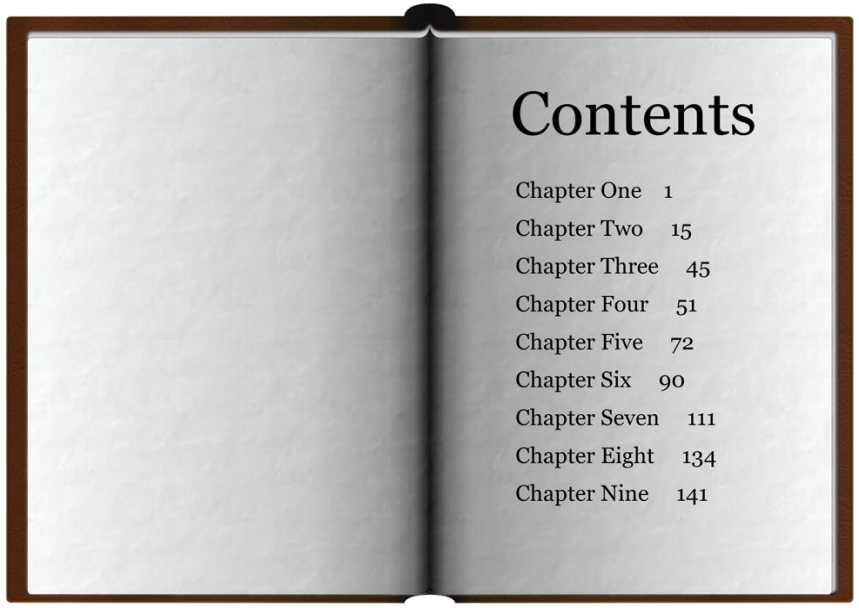
- Among other things, the following characteristics will be introduced:
  - Use of prefabs
  - If statement
  - Instanciate objects
  - Use of Arrays
  - Spawning on time intervals
  - Collision trigger

Introduction

Video

Content

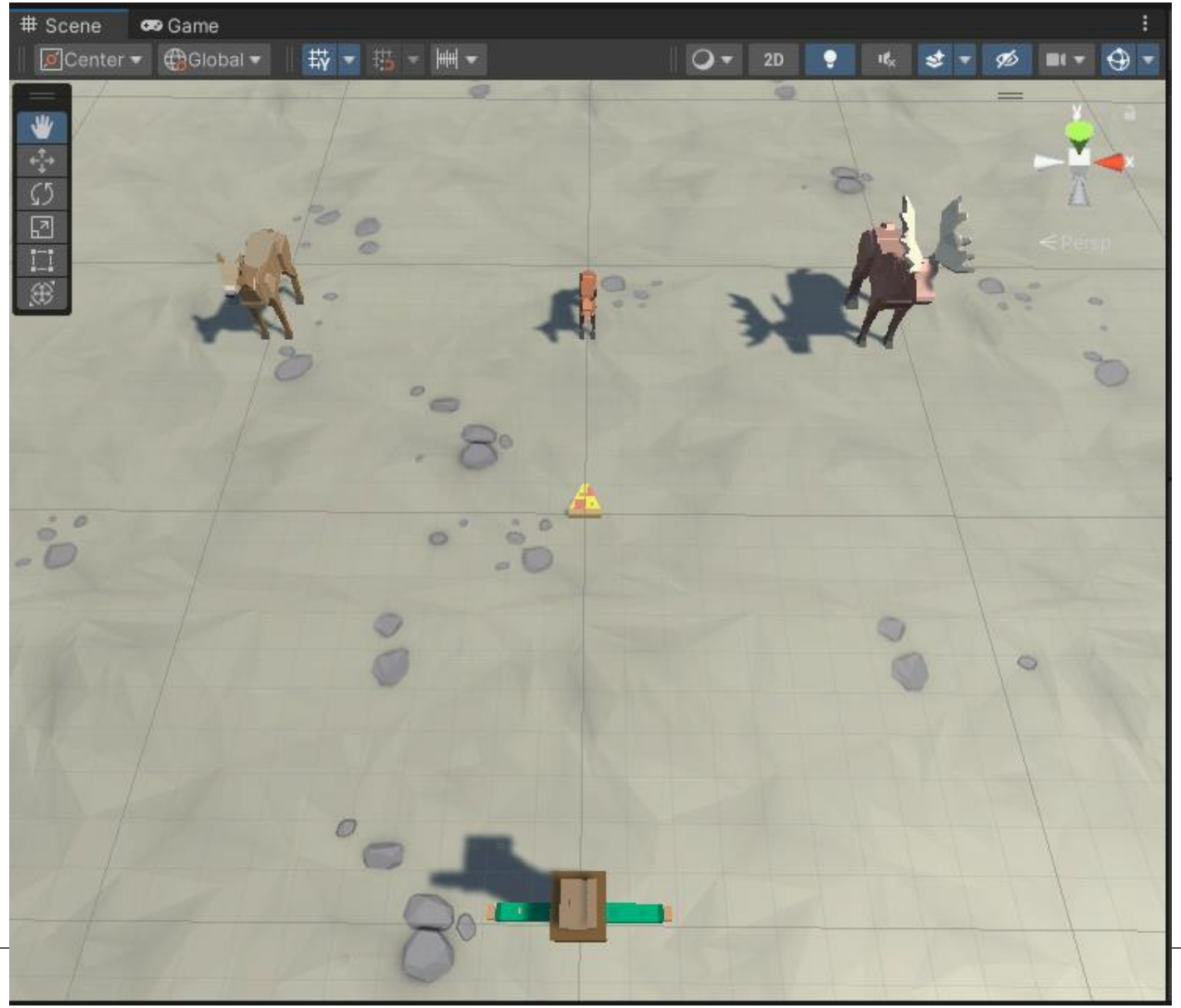
1. Introduction
2. Build the scene
3. Keep the player inbounds: IF statement
4. Prefabs
5. Launch projectiles: Instantiate()
6. Destroy Game Objects
7. Spawn manager: Arrays
8. Randomize spawning
9. Timed intervals: InvokeRepeating()
10. Collider and trigger components
11. Destroy on collision: OnTriggerEnter()
12. ACTIVITY



Build the scene

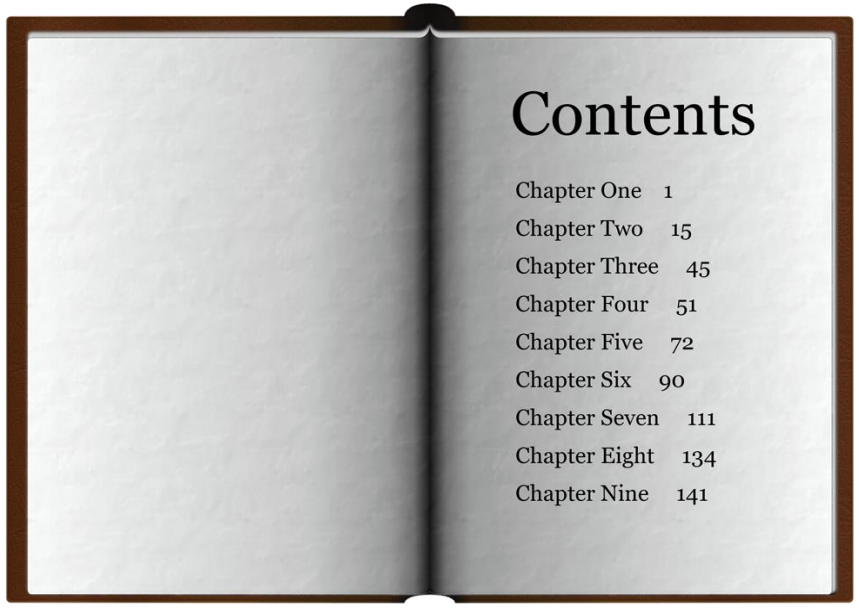
1) Import the assets *Animals.unitypackage* to a new empty project
2) If you want, drag a different material onto the Ground (Course Library > Materials)
3) Drag **1 Human, 3 Animals, and 1 Food** object
4) Rename the character to **"Player"**
5) Adjust the **XYZ scale of the food** so you can easily see it from above

Build the scene



Content

1. Introduction
2. Build the scene
3. Keep the player inbounds: IF statement
4. Prefabs
5. Launch projectiles: Instantiate()
6. Destroy Game Objects
7. Spawn manager: Arrays
8. Randomize spawning
9. Timed intervals: InvokeRepeating()
10. Collider and trigger components
11. Destroy on collision: OnTriggerEnter()
12. ACTIVITY



Keep the player inbounds: IF statement

1) Control the **player horizontal movement**
   a) Create the script "PlayerController"
   b) Get the horizontal input
   c) Move the player left/right at a given speed when arrows are pressed

```
void Update()
{
    horizontalInput = Input.GetAxis("Horizontal");
    transform.Translate(Vector3.right * horizontalInput * Time.deltaTime * speed);
}
```

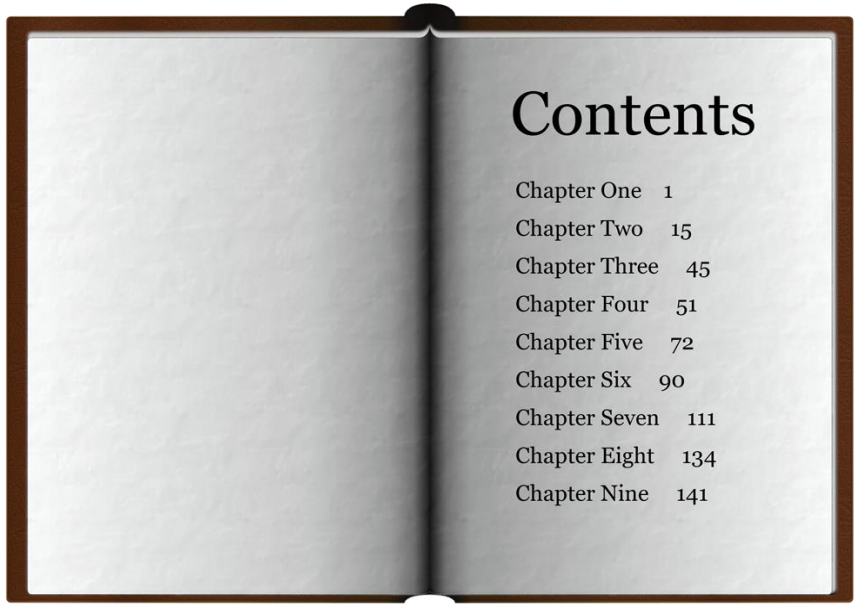
Keep the player inbounds: IF statement

1. Prevent the player from going off the side of the screen with an **if statement**
   a) What condition will you check in the if() ?
   b) How will you write such a condition sentence?
   c) How will you limit the player position?
2. Hint1: transform.position.x // .position.y // .position.z
3. Hint2: instanciate a new Vector3 to limit the positions

```
if (transform.position.x > xrange){
    transform.position = new Vector3(xrange, transform.position.y, transform.position.z);
}
if (transform.position.x < -xrange){
    transform.position = new Vector3(-xrange, transform.position.y, transform.position.z);
}
```

Content

1. Introduction
2. Build the scene
3. Keep the player inbounds: IF statement
4. Prefabs
5. Launch projectiles: Instantiate()
6. Destroy Game Objects
7. Spawn manager: Arrays
8. Randomize spawning
9. Timed intervals: InvokeRepeating()
10. Collider and trigger components
11. Destroy on collision: OnTriggerEnter()
12. ACTIVITY

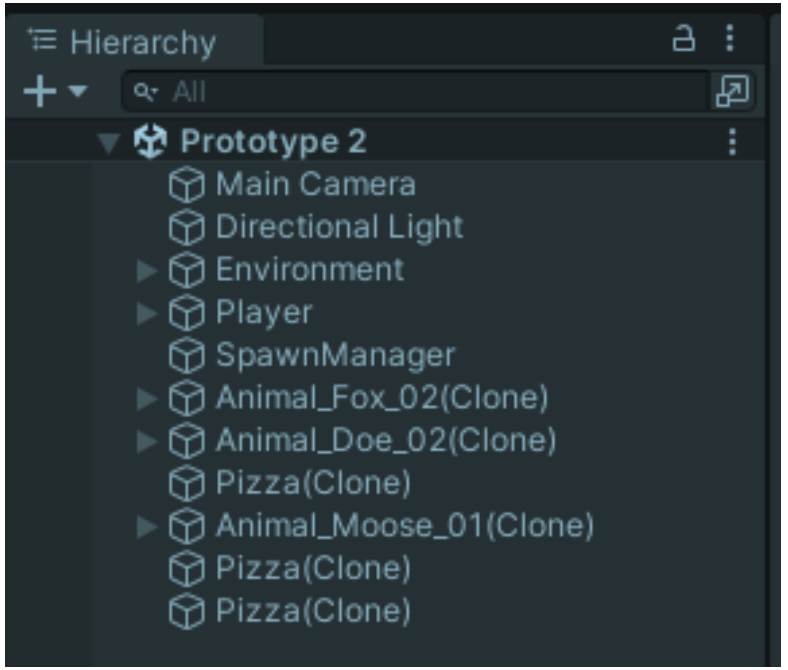


Prefabs

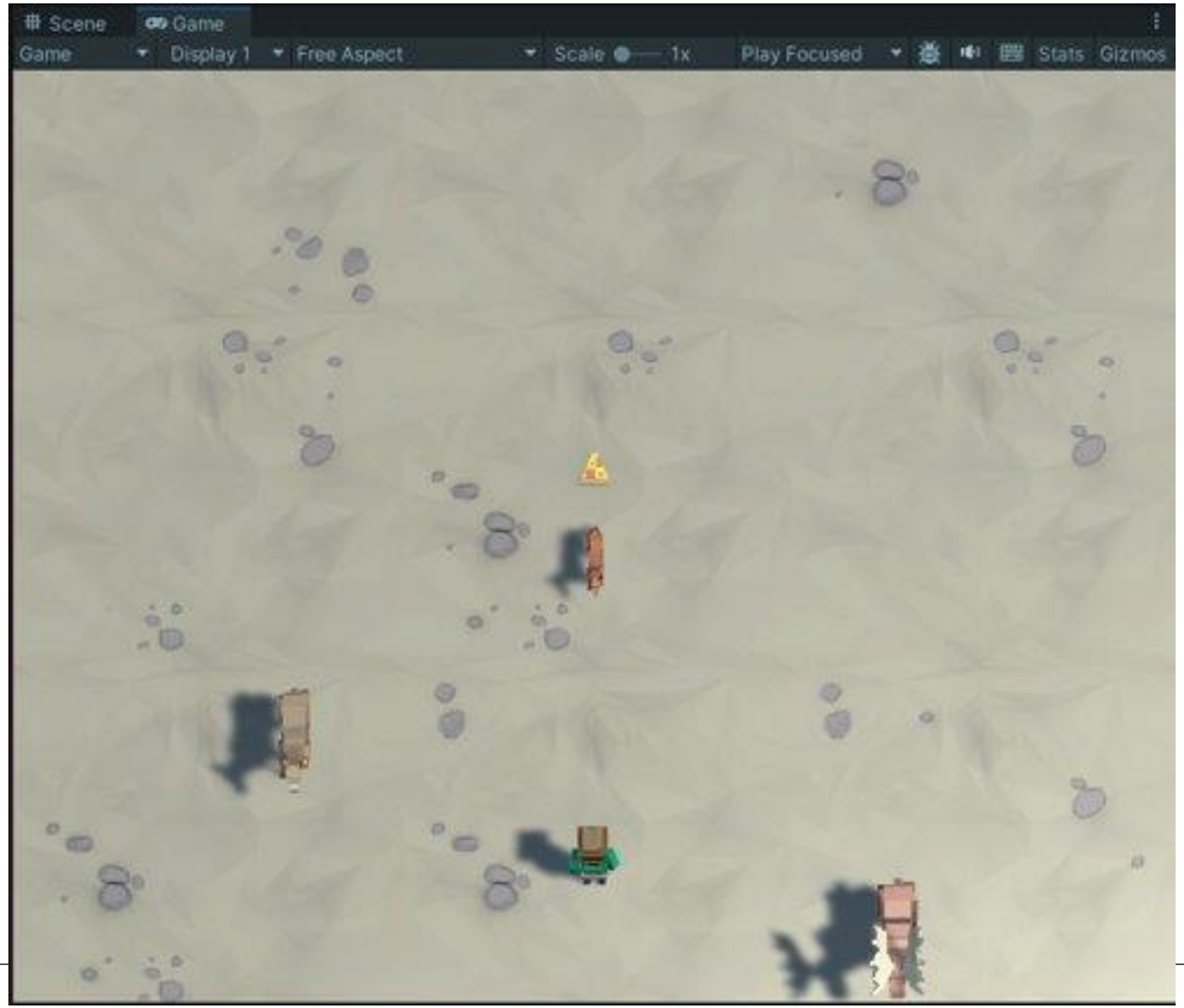
1. Make the food projectile **fly forwards** until the infinite
   a) Create a new script called "MoveForward" to program it
   b) Use a public variable to set its speed --> Then play and test
2. Make the **projectile into a prefab**
   a) In *PlayerController.cs*, declare a new GameObject variable to instanciate projectiles
   b) Drag the projectile prefab to the new Player variable box --> Why the prefab and not the original projectile?
3. Delete the projectile from the scene and run. Drag new projectiles from Prefabs at runtime to check that it works

Prefabs

1. Move the **animals forward**
   a) Rotate them 180° to face down
   b) Add the *MoveForward* script to them.
      Select all three animals in the hierarchy and
      Add Component > type Move Forward
   c) Set different speeds to the animals and test
2. Once working as expected, **make also the animals as prefabs**
3. Test by dragging prefabs during gameplay

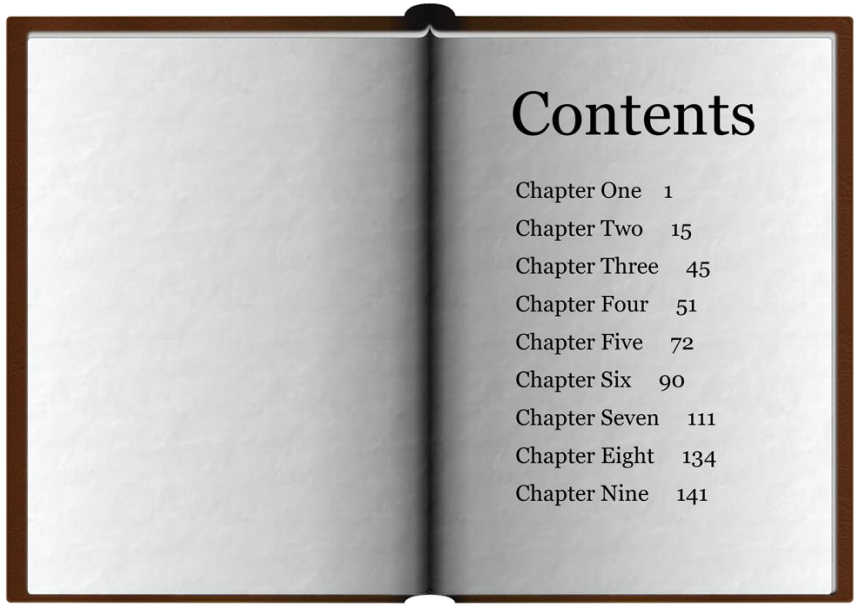


Prefabs


Scene
Game
Game
Display 1
Free Aspect
Scale
1x
Play Focused
Stats
Gizmos

Content

1. Introduction
2. Build the scene
3. Keep the player inbounds: IF statement
4. Prefabs
5. Launch projectiles: Instantiate()
6. Destroy Game Objects
7. Spawn manager: Arrays
8. Randomize spawning
9. Timed intervals: InvokeRepeating()
10. Collider and trigger components
11. Destroy on collision: OnTriggerEnter()
12. ACTIVITY



Launch projectiles: Instanciate()

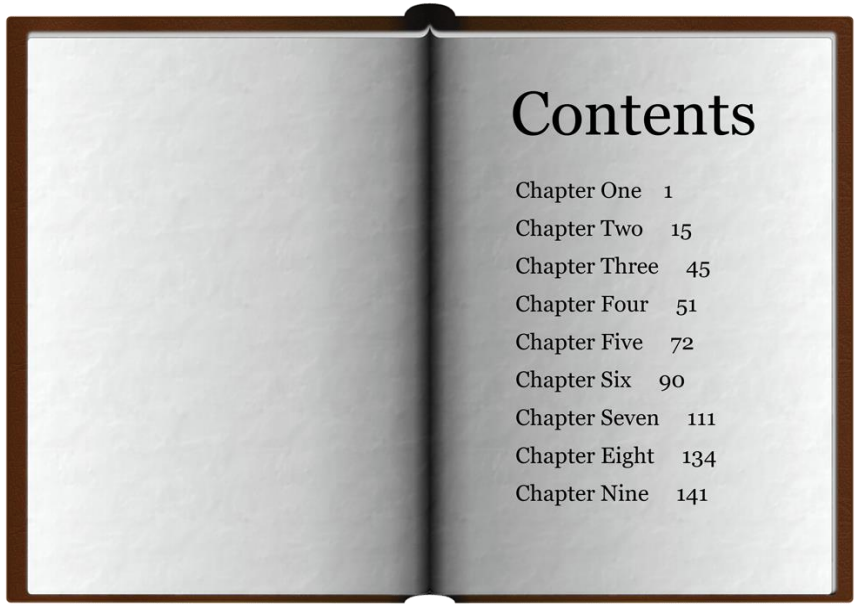
1. Check **if the spacebar has been pressed** in *PlayerController.Update()*
   a) The method *Input.GetKeyDown()* returns *true* if a given key is pressed
   b) Use *KeyCode* class as the input of the method
2. **Launch projectile on spacebar** press
   a) Use the *Instantiate()* method to spawn a projectile
   b) Do it at the player's position with the prefab's rotation

```
if (Input.GetKeyDown(KeyCode.Space))
{
    Instantiate(projectilePrefab, transform.position, projectilePrefab.transform.rotation);
}
```

Launch projectiles: Instanciate()

Content

1. Introduction
2. Build the scene
3. Keep the player inbounds: IF statement
4. Prefabs
5. Launch projectiles: Instantiate()
6. Destroy Game Objects
7. Spawn manager: Arrays
8. Randomize spawning
9. Timed intervals: InvokeRepeating()
10. Collider and trigger components
11. Destroy on collision: OnTriggerEnter()
12. ACTIVITY



Destroy GameObjects

- Whenever we spawn a projectile or animal, it drifts past the play area into eternity
- **In order to improve game performance**, we need to destroy them when they go out of bounds

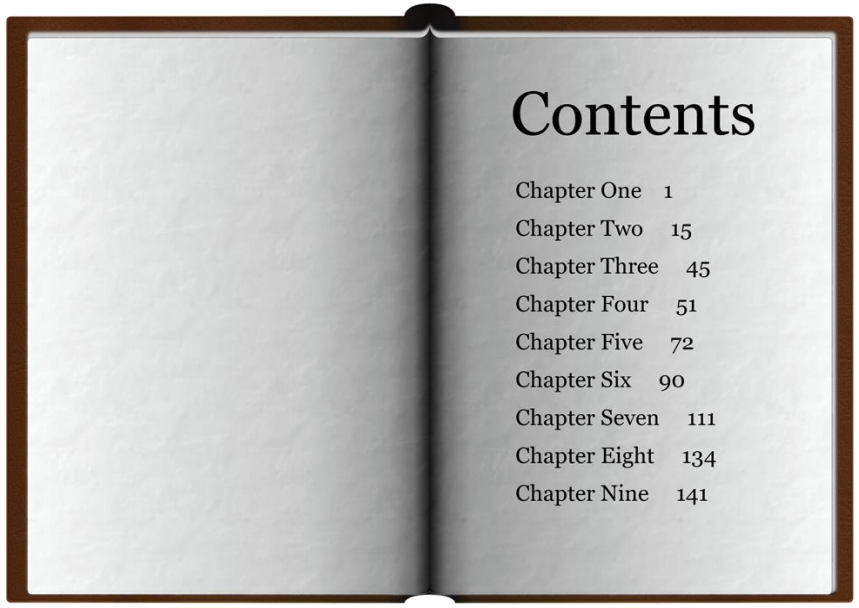
1. Create **"DestroyOutOfBounds" script** and apply it to the projectile in the prefabs folder
   a) How will you determine if the projectile is out of bounds?
   b) Use the Destroy() method to destroy the current object
2. Check that it works and, if so, **modify the script and use it also to destroy the animals** when they go out of "their" bounds

Destroy GameObjects

```
public class DestroyOutOfBounds : MonoBehaviour
{
    1 reference
    private float topBound = 30.0f;
    1 reference
    private float bottomBound = -15.0f;
    // Start is called before the first frame up
    0 references
    void Start()
    {

    }
    // Update is called once per frame
    0 references
    void Update()
    {
        if (transform.position.z > topBound)
        {
            Destroy(gameObject);
        }
        else if (transform.position.z < bottomBound)
        {
            Destroy(gameObject);
        }
    }
}
```

Content

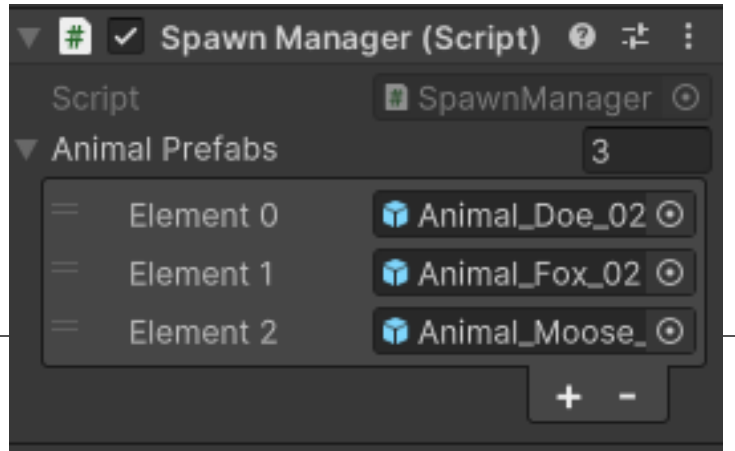
1. Introduction
2. Build the scene
3. Keep the player inbounds: IF statement
4. Prefabs
5. Launch projectiles: Instantiate()
6. Destroy Game Objects
7. Spawn manager: Arrays
8. Randomize spawning
9. Timed intervals: InvokeRepeating()
10. Collider and trigger components
11. Destroy on collision: OnTriggerEnter()
12. ACTIVITY



Spawn Manager: Arrays

- If we are going to be doing all of this complex spawning of animals, we should have a **dedicated script to manage the process**
- We will use a script attached to an empty object

1. **Create an Empty object and a Script** both called "SpawnManager" and assign the script to the empty object
2. Declare a **public array of GameObjects** --> How will you do that?
3. In the Inspector, change the **Array size** to match your animal count and then **assign your animals**



Spawn Manager: Arrays

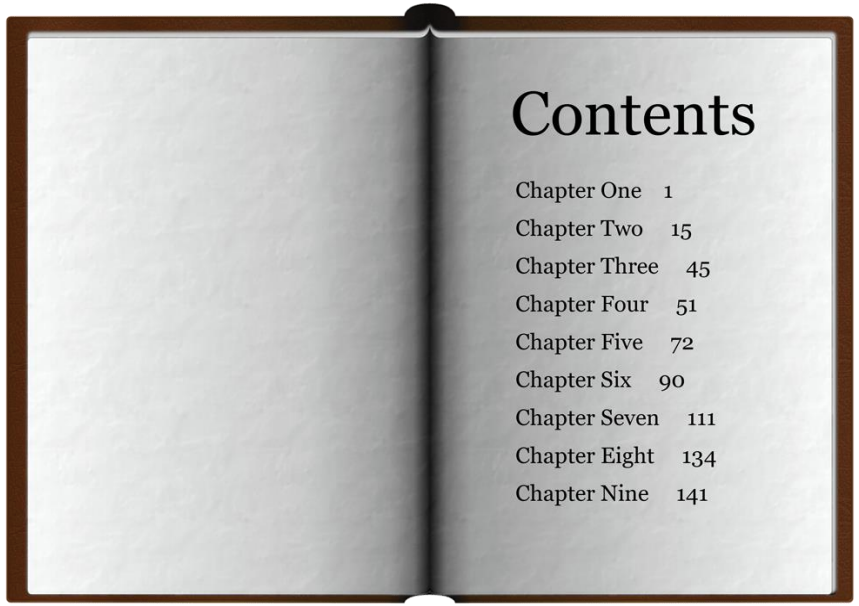
- Let's create a temporary solution for choosing and spawning the animals when 'S' is pressed

1. In *Update()*, write the code to **instantiate a new animal** at the top of the screen **if 'S' is pressed**
2. The **type of animal should be selected from the array** using a public variable
3. Test it and check that one can spawn different types of animals

```
void Update() {
  if (Input.GetKeyDown(KeyCode.S)) {
    Instantiate(animalPrefabs[animalIndex], new Vector3(0, 0, 20),
    animalPrefabs[animalIndex].transform.rotation);
  }
}
```

Content

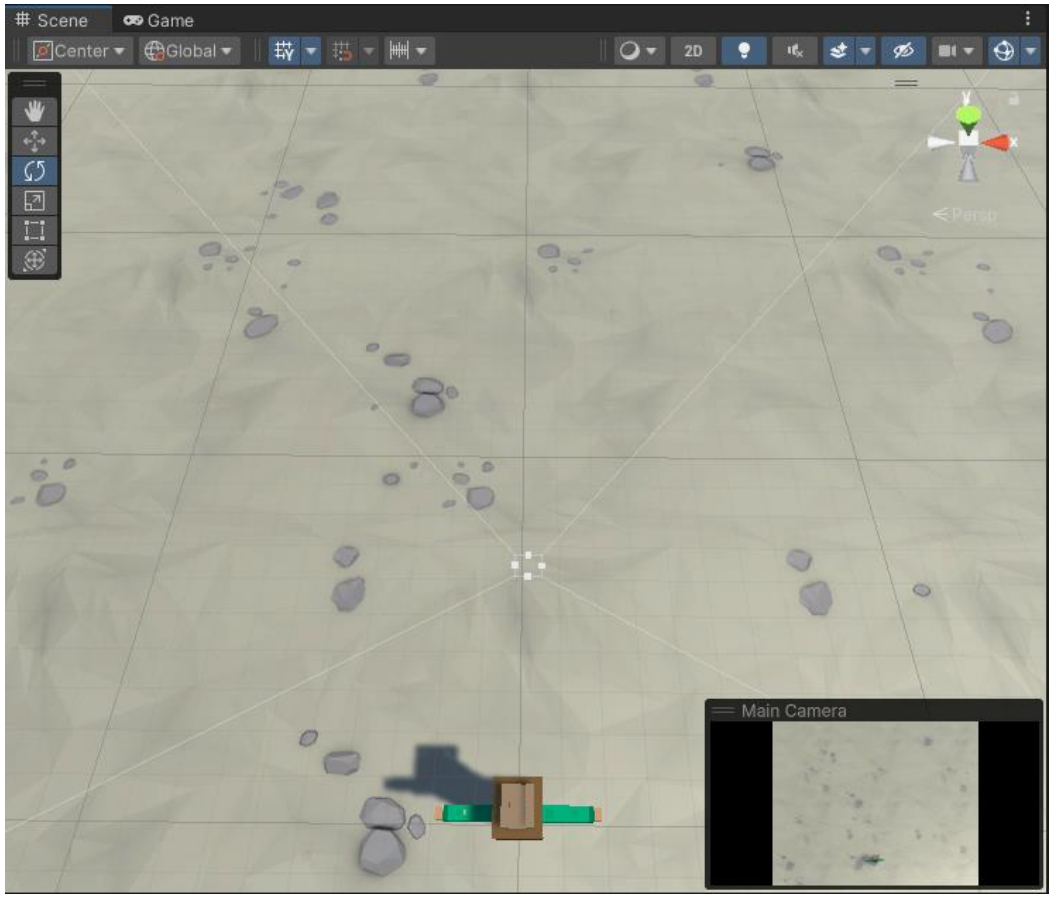
1. Introduction
2. Build the scene
3. Keep the player inbounds: IF statement
4. Prefabs
5. Launch projectiles: Instantiate()
6. Destroy Game Objects
7. Spawn manager: Arrays
8. Randomize spawning
9. Timed intervals: InvokeRepeating()
10. Collider and trigger components
11. Destroy on collision: OnTriggerEnter()
12. ACTIVITY



Randomize spawning

- Modify the script to randomize the animals spawning

1. The **type of animal** must be random
2. The animal **horizontal position** must be random
3. Change the camera's perspective to offer a more appropriate view for this top-down game
4. Select the **Main Camera** and change the Projection from "Perspective" to "**Orthographic**"

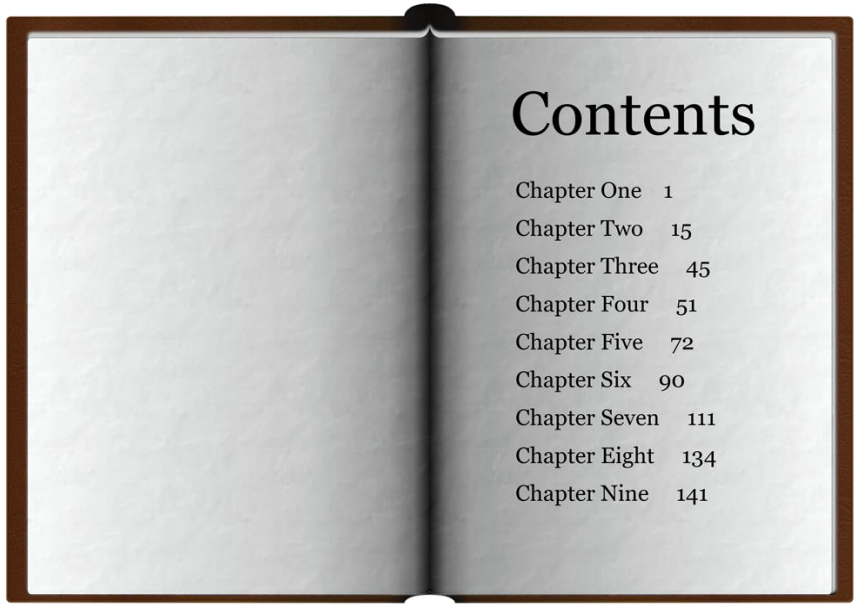


Randomize spawning

```
private float spawnRangeX = 20;
private float spawnPosZ = 20;

void Update() {
  if (Input.GetKeyDown(KeyCode.S)) {
    // Randomly generate animal index and spawn position
    Vector3 spawnPos = new Vector3(Random.Range(-spawnRangeX, spawnRangeX),
    0, spawnPosZ);
    int animalIndex = Random.Range(0, animalPrefabs.Length);
    Instantiate(animalPrefabs[animalIndex], spawnPos,
    animalPrefabs[animalIndex].transform.rotation); }}
```

Content

1. Introduction
2. Build the scene
3. Keep the player inbounds: IF statement
4. Prefabs
5. Launch projectiles: Instantiate()
6. Destroy Game Objects
7. Spawn manager: Arrays
8. Randomize spawning
9. Timed intervals: InvokeRepeating()
10. Collider and trigger components
11. Destroy on collision: OnTriggerEnter()
12. ACTIVITY



Timed intervals: InvokeRepeating()

- It makes no sense pressing a key to spawn enemies in a game
- We need to **spawn the animals on a timer**, so they randomly appear every few seconds

1. Create a **new method *SpawnRandomAnimal()*** and move the *if()* content into it. Leave the Update() empty.
2. In *Start()*, use ***InvokeRepeating()*** method to **spawn animals based on an interval**
3. *InvokeRepeating()* will call a given method after a time delay and repeat the call on a given time interval <-- How will you use it?

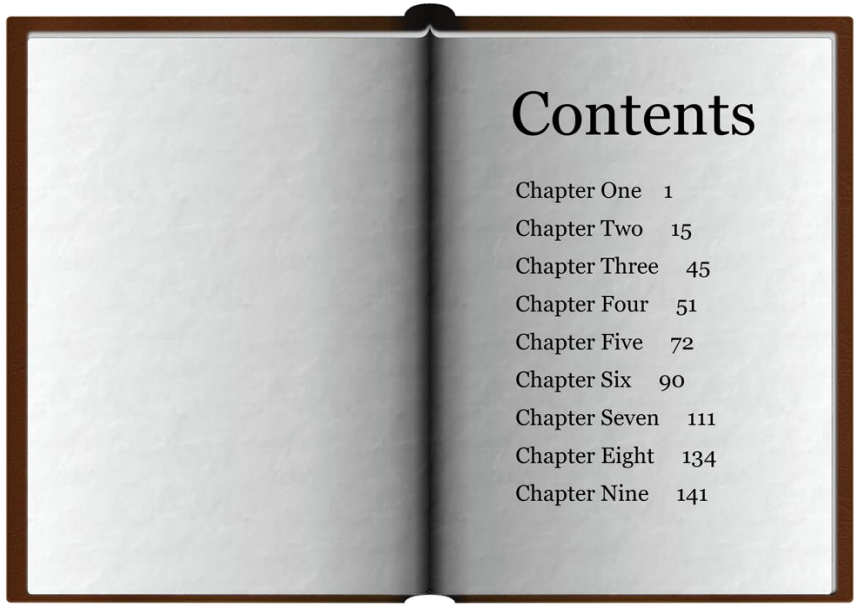
Timed intervals: InvokeRepeating()

```
public class SpawnManager : MonoBehaviour
{
    3 references
    public GameObject[] animalPrefabs;
    2 references
    private float spawnRangeX = 15;
    1 reference
    private float spawnPosZ = 20;
    1 reference
    private float startDelay = 2.0f;
    1 reference
    private float spawnInterval = 1.5f;

    void Start()
    {
        InvokeRepeating("SpawnRandomAnimal", startDelay, spawnInterval);
    }
    0 references
    void SpawnRandomAnimal()
    {
        Vector3 spawnPos = new Vector3(Random.Range(-spawnRangeX, spawnRangeX), 0, spawnPosZ);
        int animalIndex = Random.Range(0, animalPrefabs.Length);

        Instantiate(animalPrefabs[animalIndex], spawnPos,
            animalPrefabs[animalIndex].transform.rotation);
    }
}
```

Timed intervals: InvokeRepeating()

Content

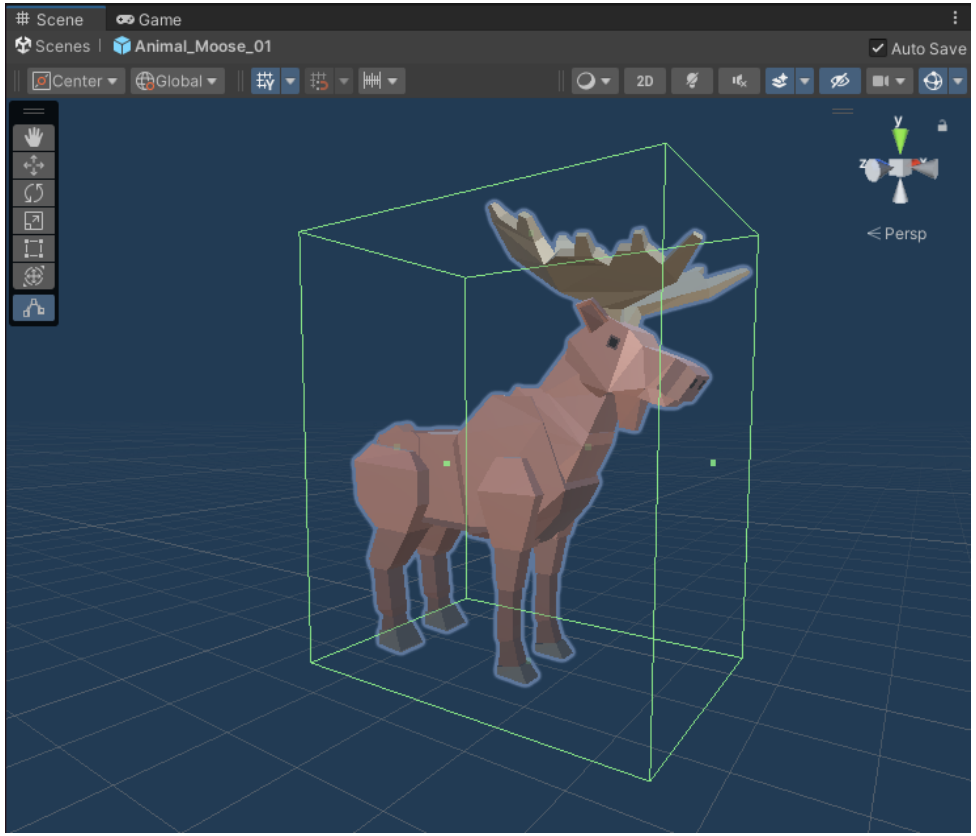
1. Introduction
2. Build the scene
3. Keep the player inbounds: IF statement
4. Prefabs
5. Launch projectiles: Instantiate()
6. Destroy Game Objects
7. Spawn manager: Arrays
8. Randomize spawning
9. Timed intervals: InvokeRepeating()
10. Collider and trigger components
11. Destroy on collision: OnTriggerEnter()
12. ACTIVITY



Collider and Trigger components

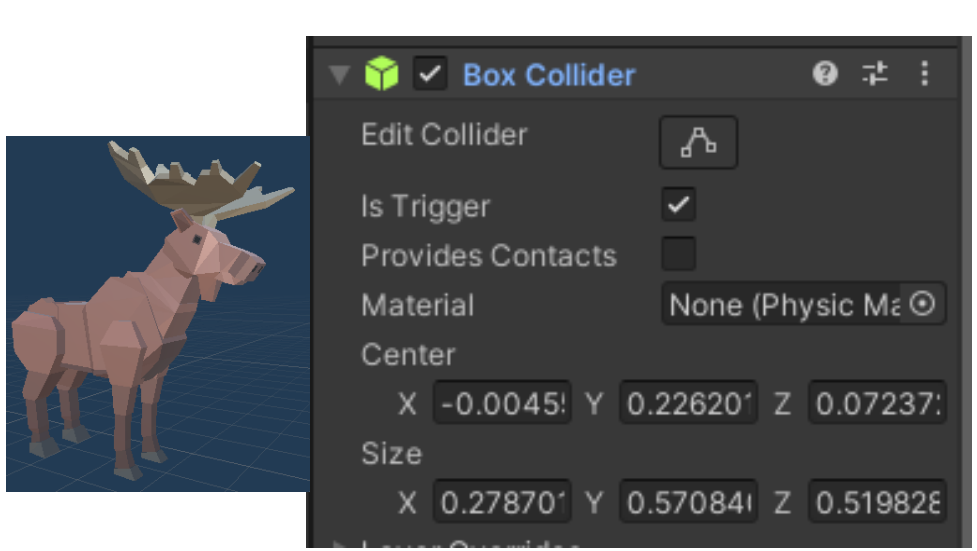
- Animals spawn perfectly and the player can fire projectiles at them but **nothing happens when the two collide**
- If **we want the projectiles and animals to be destroyed on collision**, we need to give them some familiar components --> *Colliders*

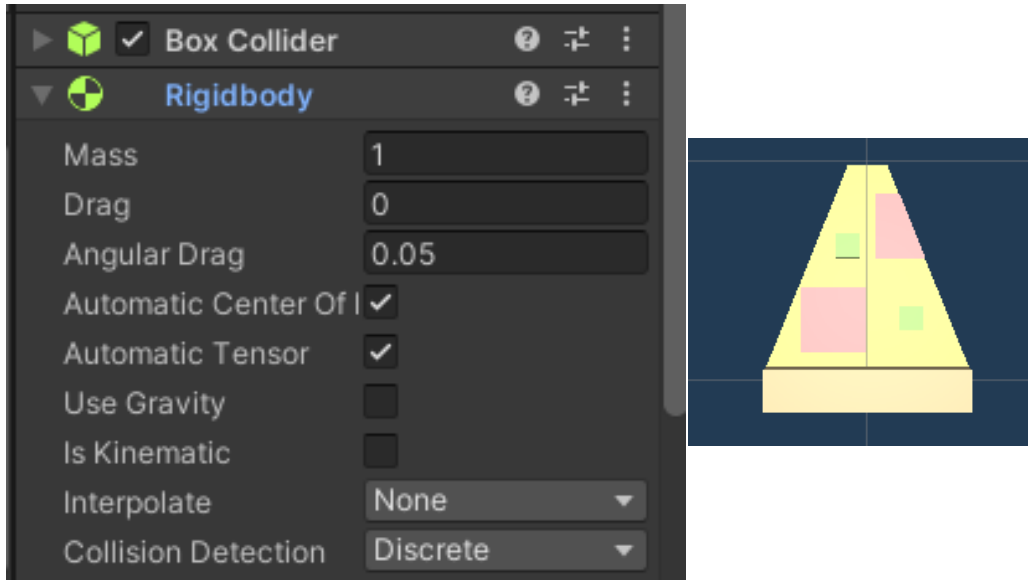
1. **Open the prefabs editor** on one of the animal prefabs **and add a *Box Collider***
2. Click *Edit Collider* and **drag the collider handles** to encompass the object



Collider and Trigger components

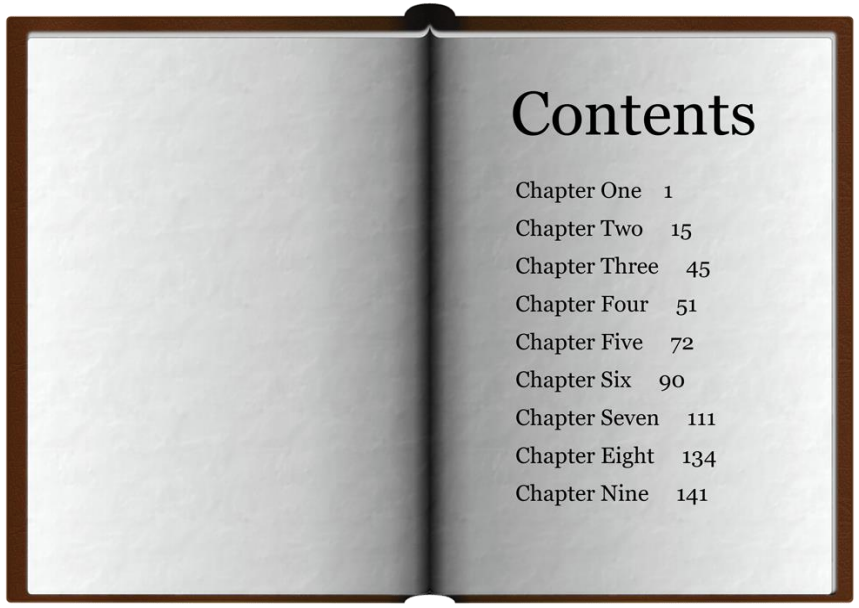
1. Check the "**Is Trigger**" checkbox
2. **Repeat** this process **for each of the animals and the projectile**
3. **Add a RigidBody** component **to** the **projectile** and **uncheck "use gravity"** (we don't want the projectile to fall on the ground)





Content

1. Introduction
2. Build the scene
3. Keep the player inbounds: IF statement
4. Prefabs
5. Launch projectiles: Instantiate()
6. Destroy Game Objects
7. Spawn manager: Arrays
8. Randomize spawning
9. Timed intervals: InvokeRepeating()
10. Collider and trigger components
11. Destroy on collision: OnTriggerEnter()
12. ACTIVITY



Destroy on collision: OnTriggerEnter()

- We need to code a new script in order to **destroy animals and projectiles on impact**

1. Create a new ***DetectCollisions*** **script, add it to each animal** prefab
2. Add the Unity **OnTriggerEnter method**

   void OnTriggerEnter(Collider other){ } <-- Can you guess what it does?

3. **Destroy both objects** on collision trigger

```
private void OnTriggerEnter(Collider other)
{
    Destroy(gameObject);
    Destroy(other);
}
```

Do we have to add it also to the projectile prefab? Why?

Destroy on collision: OnTriggerEnter()

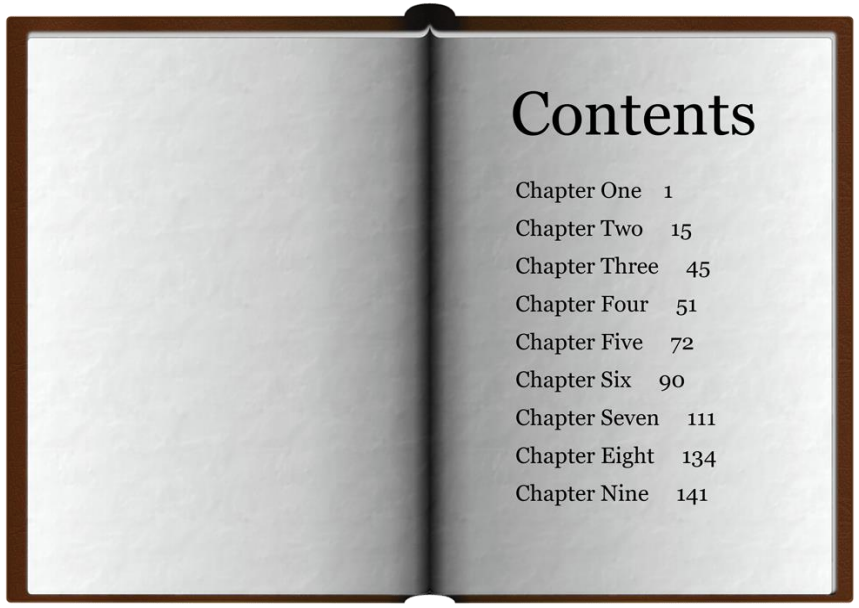
- Is this an **endless game**?
- We should let the players know they've **lost when any animals get past the player**
- For the time being, **a simple "Game Over" message** will be enough
- In following activities we'll stop spawning enemies and the action

1. In *DestroyOutOfBounds* script, add a Debug.Log() message in the appropriate place

```
else if (transform.position.z < bottomBound)
{
    Destroy(gameObject);
    Debug.Log("Game Over!");
}
```

Content

1. Introduction
2. Build the scene
3. Keep the player inbounds: IF statement
4. Prefabs
5. Launch projectiles: Instantiate()
6. Destroy Game Objects
7. Spawn manager: Arrays
8. Randomize spawning
9. Timed intervals: InvokeRepeating()
10. Collider and trigger components
11. Destroy on collision: OnTriggerEnter()
12. ACTIVITY



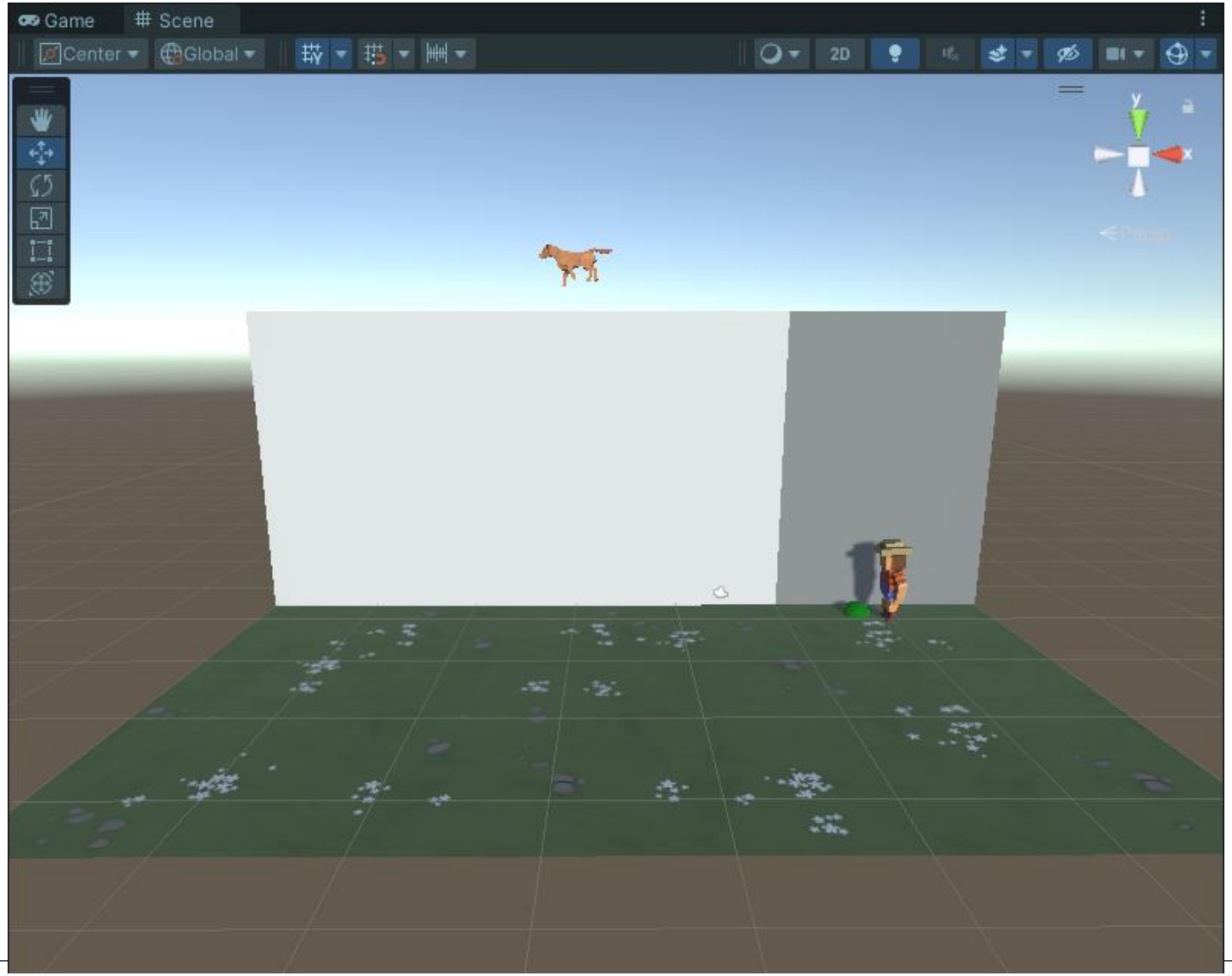
ACTIVITY

Fix the errors of the game (Dog.unitypackage)

1. A random ball (of 3) is generated at a random position above the screen
2. Make the spawn interval a random value between 3 and 5 seconds
3. Make the player spawn dogs on spacebar
4. The balls should be destroyed on direct contact with a dog (not near)
5. If the ball hits the ground, a “Game Over” message is displayed
6. Balls and dogs should be removed when they leave the screen

ACTIVITY

Initial state



ACTIVITY

How it should be

EXTRA ACTIVITY

Add the following modifications to the Animals game:

1. Allow the player to move forward and backwards
2. Also spawn animals from the left and right sides
3. If any hits the player, "Game Over" should be logged to the console
4. Display in the console the player's Lives and Score
   a) If the player feeds an animal, increase the Score
   b) If the player misses an animal or is hit by one, decrease Lives
   c) When the number of Lives reaches 0, log "Game Over" in the console

LICENSE

Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0)

You are free to:

Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

NonCommercial — You may not use the material for commercial purposes.

ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

https://creativecommons.org/licenses/by-nc-sa/3.0/