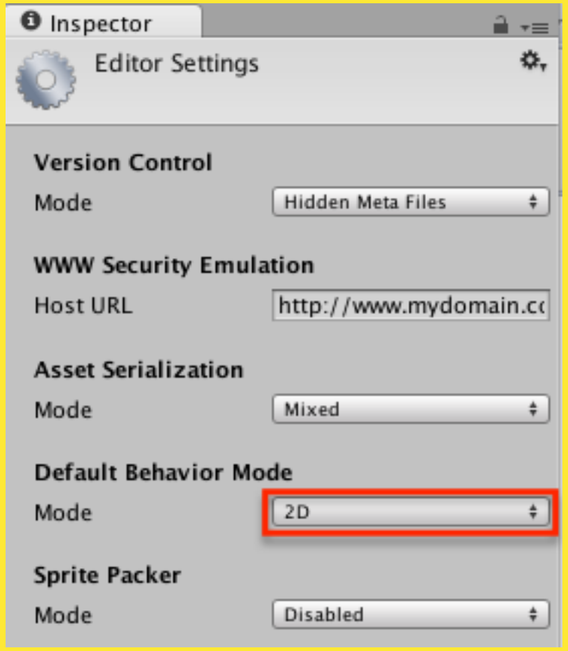
ROAD TO GAMEDEV

WELCOME !

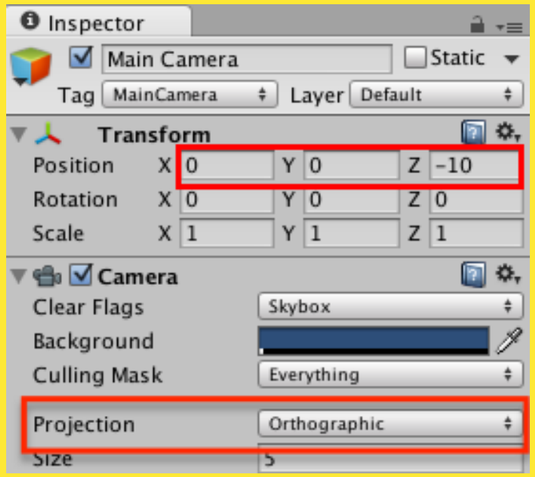
Asegurémonos de que estamos trabajando en 2D

- Edit > Project Settings > Editor > Default Behavior Mode
- Scene view



Main Camera

- **Position:** `(0, 0, -10)`
- **Projection:** `Orthographic`
- **Size:** `3.2`



Project

Creamos una carpeta para las escenas

Guardamos la escena (`^S` / `⌘S`*) como* `JetJerry.unity`

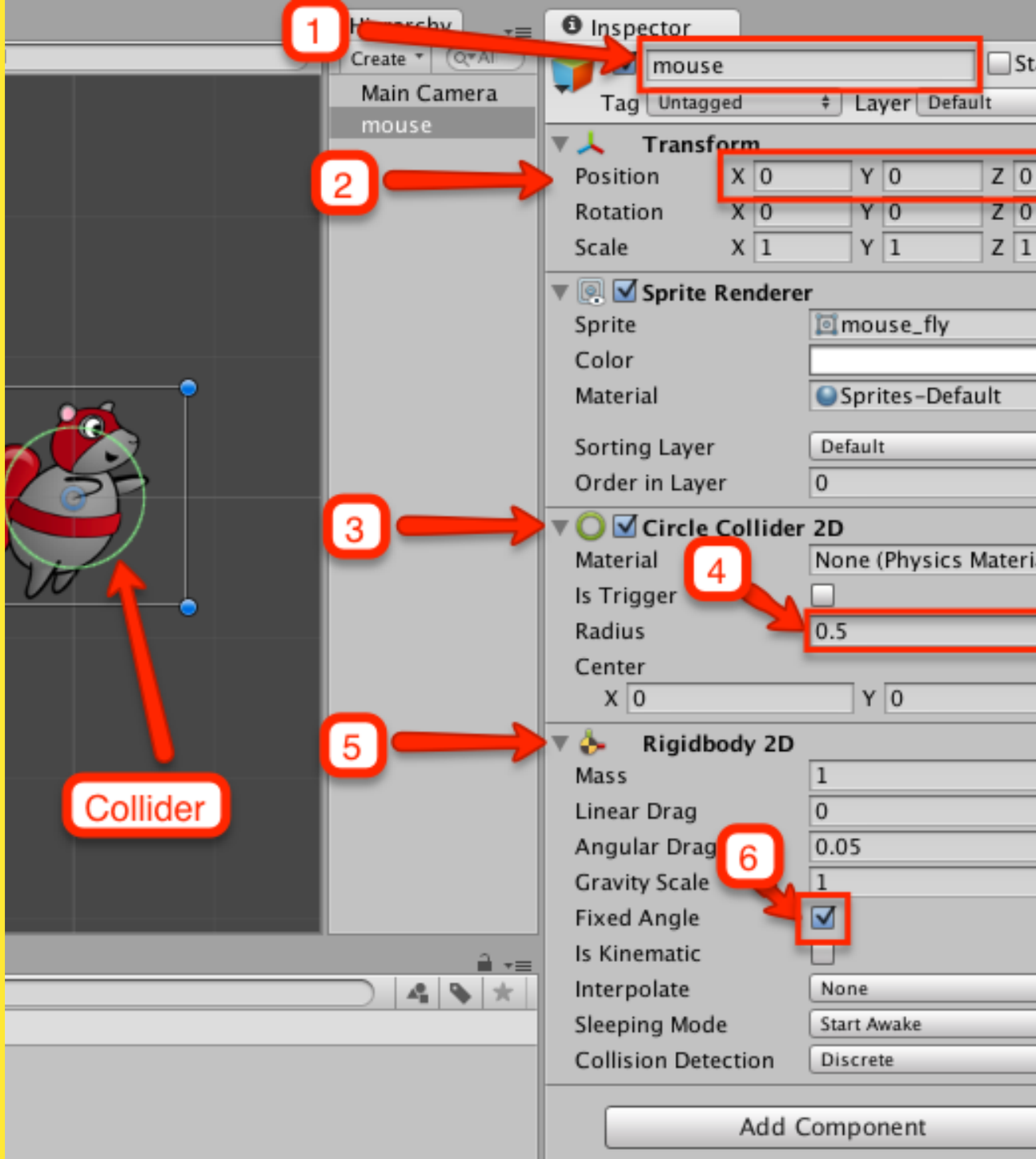
Resolución

Creamos una nueva:

- iPhone Landscape: 1136x640

Añadimos el sprite `mouse_fly` *a la escena:*

1. *Lo renombramos a* `mouse`
2. *Position:* `(0, 0, 0)`
3. *Añadimos Circle Collider 2D / Radius:* `0.5`
4. *Añadimos Rigidbody 2D / Habilitamos* `Freeze Rotation`



EL RATÓN SE VA A LA MIER**!!

Creamos una carpeta para los scripts

Creamos un script de C#, `MouseController.cs`

Vinculamos el script con la instancia de `mouse` *y abrimos el editor*

HERE COMES THE CODE !

DEVNOTES: USAR CAFEÍNA PARA POTENCIAR LOS EFECTOS DEL CÓDIGO

```
public float jetpackForce = 75.0f;
private Rigidbody2D rb;

void Start() {
    rb = GetComponent<Rigidbody2D>();
}

void FixedUpdate() {
    bool jetpackActive = Input.GetButton("Fire1");

    if (jetpackActive) {
        rb.AddForce(new Vector2(0, jetpackForce));
    }
}
```

Vamos a ajustar la gravedad, no ?

- Edit > Project Settings > Physics 2D
- Gravity Y = -15

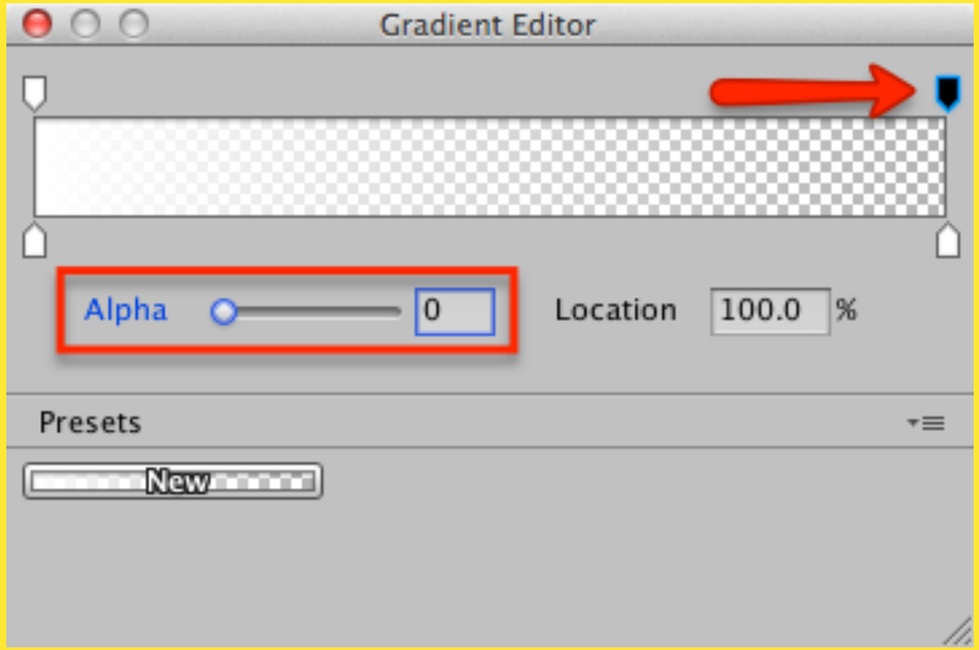
Vamos a añadir partículas !

- *Click derecho en Hierarchy > Particle System*
- *Anidarlo en la instancia de* `mouse`
- *Renombrar a* `jetpackFlames`

- ***Position:*** `(-0.62, -0.33, 0)`
- ***Rotation:*** `(65, 270, 270)`
- ***Start Lifetime:*** `0.5`
- ***Start Size:*** `0.3`
- ***Start Color:*** `rgba(255, 135, 40, 255)`
- ***Emission Rate:*** `300`
- ***Shape:*** Cone ***/ Angle:*** `12` ***/ Radius:*** `0.1` ***/ Random Direction***

Vamos a mejorarlas un poco...

- **Color Over Lifetime**
- **Le bajamos la opacidad a 0 al slider de arriba a la derecha**



Creamos un nuevo `Empty GameObject` *en la Hierarchy y lo nombramos* `floor`

1. *Position:* `(0, -3.5, 0)`
2. *Scale:* `(14.4, 1, 1)`
3. *Añadimos* Box Collider 2D

CHALLENGE !

VAMOS A HACER UN TECHO

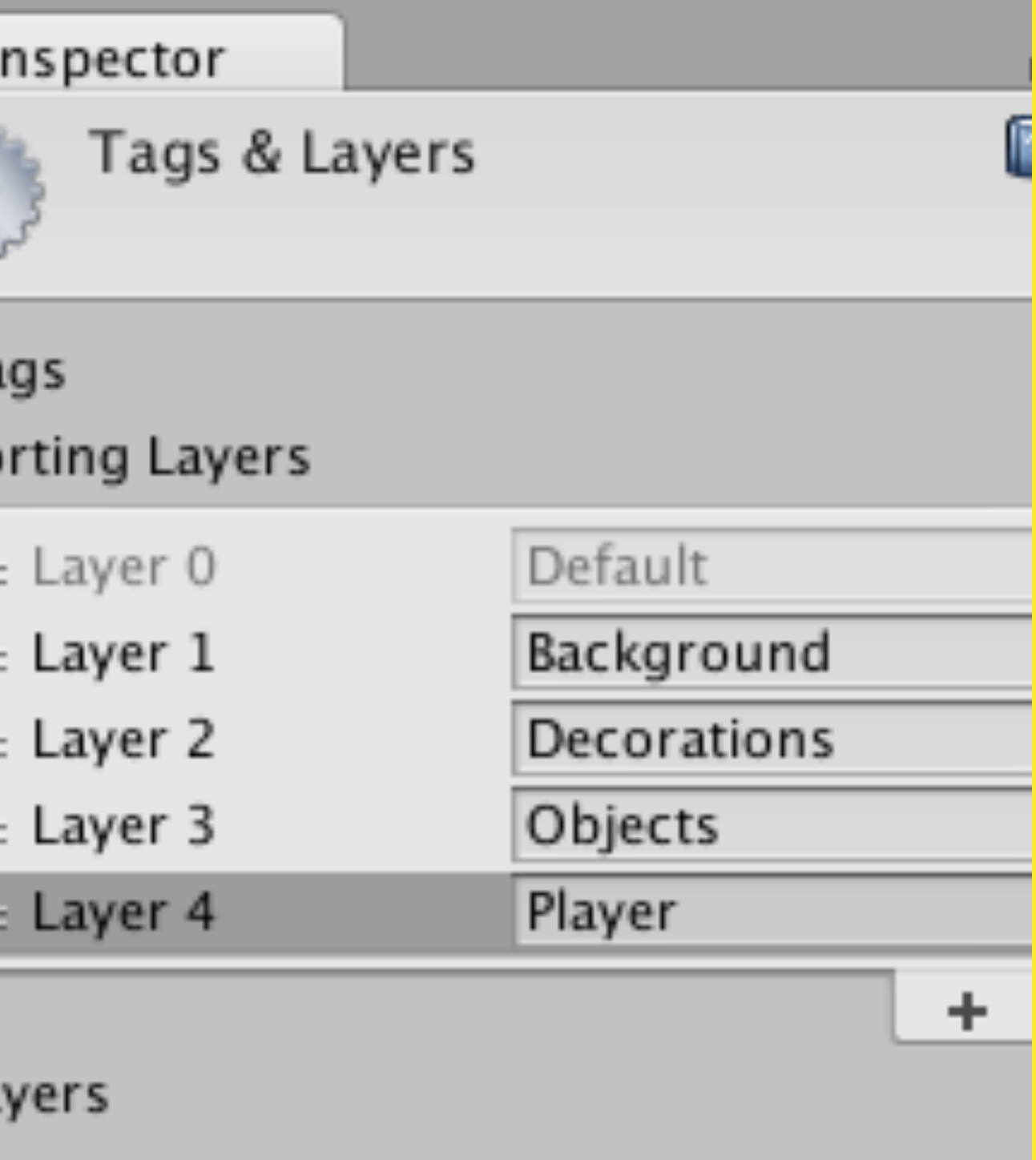
HABITACIONES

1. *Arrastramos el sprite* `bg_window` *a la Scene View*
2. *Le reseteamos la posición a:* `(0, 0, 0)`
3. *Arrastramos dos copias del sprite* `bg` *a la Scene View*
4. *Usando vertex snapping ajustamos las posiciones de estos (V)*

EL RATÓN ESTÁ MARGI... 

SORTING LAYERS !

FANFARRIA



Seleccionamos la instancia de `mouse` *y buscamos el componente* Sprite Renderer

Añadimos las siguientes Sorting Layers:

- *Background*
- *Decorations*
- *Objects*
- *Player*

- *Establecemos la Sorting Layer de la instancia de* `mouse` *a* `Player`
- *Establecemos la Sorting Layer de las instancias* `bg` *y* `bg_window` *a* `Background`
- *Establecemos la Sorting Layer del Particle System a* `Objects`

DISEÑO DE INTERIORES

DÉMOSLE VIDA A NUESTRO HÉROE !

Abrimos el script `MouseController.cs` ***y añadimos la siguiente variable:***

```
public float forwardMovementSpeed = 3.0f;
```

Añadimos el siguiente código al final de `FixedUpdate()`*:*

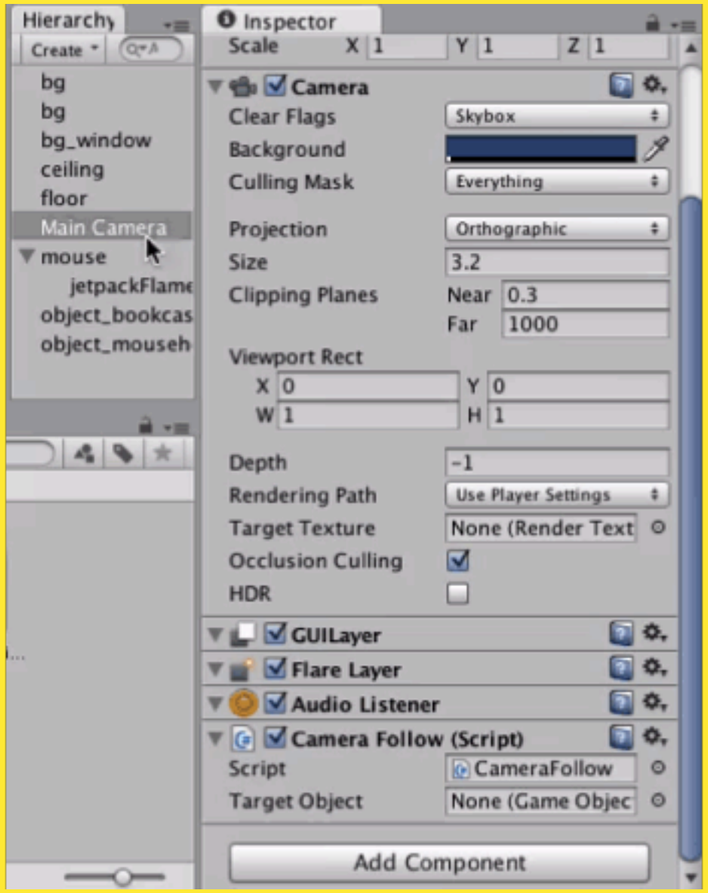
```
Vector2 newVelocity = rb.velocity;
newVelocity.x = forwardMovementSpeed;
rb.velocity = newVelocity;
```

CÁMARAS...
ACCIÓN !

Creamos un nuevo script, `CameraFollow.cs`***, lo vinculamos en la instancia*** `Main Camera` ***y abrimos el editor...***

```
public GameObject targetObject;

public Update() {
    float targetObjectX = targetObject.transform.position.x;

    Vector3 newCameraPosition = transform.position;
    newCameraPosition.x = targetObjectX;
    transform.position = newCameraPosition;
}
```

Establecemos la instancia de `mouse` *como* `targetObject` *en el inspector de Unity*



Hagamos que el ratón esté en una posición adecuada para que podamos ver los obstáculos a tiempo:

Seleccionamos la instancia `mouse`

- **Position:** `(-3.5, 0, 0)`

Añadimos la siguiente variable al script `CameraFollow.cs`*:*

```
private flat distanceToTarget;
```

Y la siguiente línea al principio de `Start()`*:*

```
distanceToTarget = transform.position.x - targetObject.transform.position.x;
```

Modificamos Update():

```
void Update() {
    float targetObjectX = targetObject.transform.position.x;

    Vector3 newCameraPosition = transform.position;
    newCameraPosition.x = targetObjectX + distanceToTarget;
    transform.position = newCameraPosition;
}
```

NIVELES INFINITOS !

IT'S OVER 9000 !!!!!

- *Creamos un* `Empty GameObject` *en la Hierarchy*
  - *Lo renombramos a* `room1`
  - *Le reseteamos Position:* `(0, 0, 0)`
- *Anidamos todos los componentes de la habitación en la nueva instancia*

- *Creamos una carpeta* `Prefabs`
- *Arrastramos la instancia* `room1` *a dicha carpeta*

NUESTRO PRIMER PREFAB !

- *Creamos un nuevo script,* `GeneratorScript.cs`
- *Lo vinculamos con la instancia de* `mouse`
- *Abrimos el editor de código...*

```
// Import !
using System.Collections.Generic;

// Class
public GameObject[] availableRooms;
public List<GameObject> currentRooms;
private float screenWidthInPoints;

void Start() {
    float height = 2.0f * Camera.main.orthographicSize;
    screenWidthInPoints = height * Camera.main.aspect;
}
```

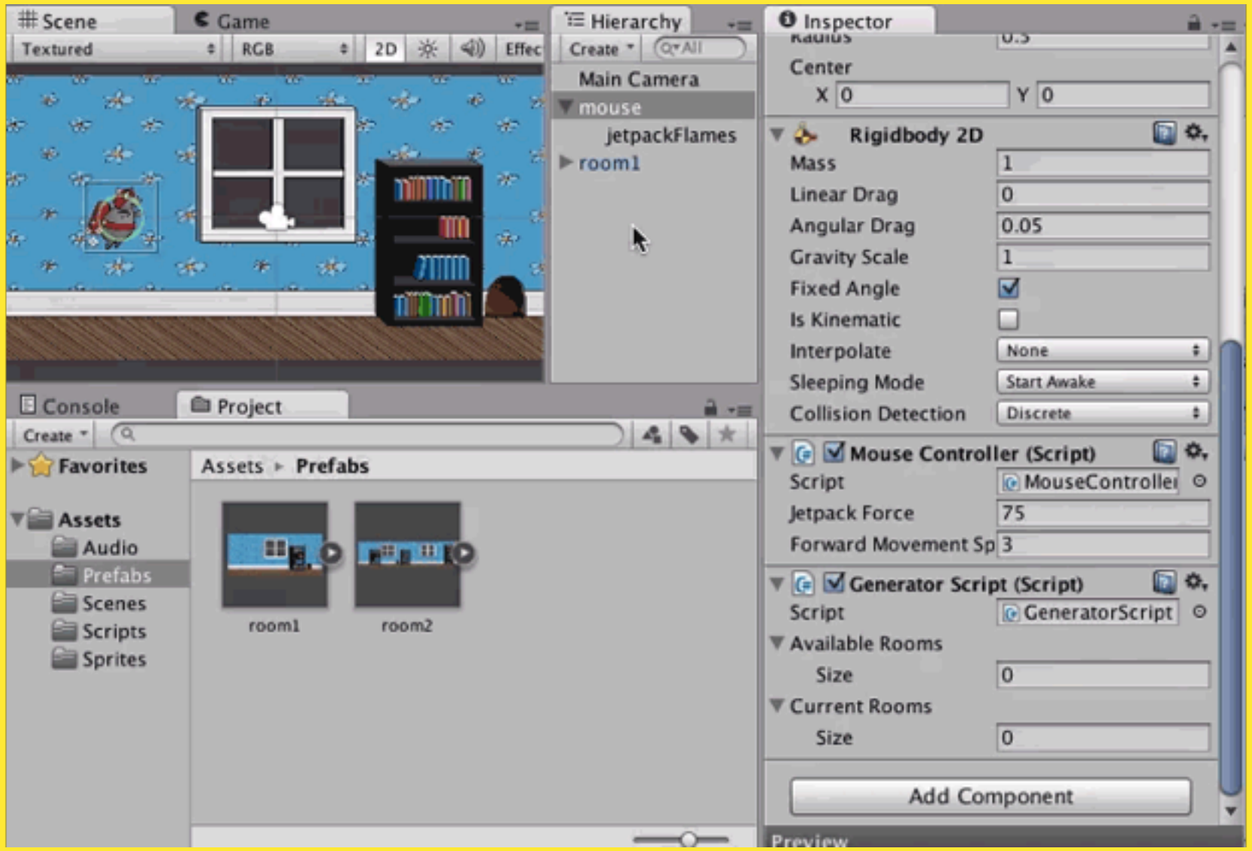
```
void AddRoom(float farthestRoomEndX) {
    int randomRoomIndex = Random.Range(0, availableRooms.Length);
    GameObject room = (GameObject) Instantiate(availableRooms[randomRoomIndex]);
    float roomWidth = room.transform.FindChild("floor").localScale.x;
    float roomCenter = farthestRoomEndX + roomWidth * 0.5f;

    room.transform.position = new Vector3(roomCenter, 0, 0);
    currentRooms.Add(room);
}
```

```
void GenerateRoomIfRequired() {
    List<GameObject> roomsToRemove = new List<GameObject>();
    bool addRooms = true;
    float playerX = transform.position.x;
    float removeRoomX = playerX - screenWidthInPoints;
    float addRoomX = playerX + screenWidthInPoints;
    float farthestRoomEndX = 0;

    foreach (var room in currentRooms) {
        float roomWidth = room.transform.FindChild("floor").localScale.x;
        float roomStartX = room.transform.position.x - (roomWidth * 0.5f);
        float roomEndX = roomStartX + roomWidth;

        if (roomStartX > addRoomX)
            addRooms = false;

        if (roomEndX < removeRoomX)
            roomsToRemove.Add(room);

        farthestRoomEndX = Mathf.Max(farthestRoomEndX, roomEndX);
    }

    foreach (var room in roomsToRemove) {
        currentRooms.Remove(room);
        Destroy(room);
    }

    if (addRooms)
        AddRoom(farthestRoomEndX);
}
```

Y por último, pero no menos importante...

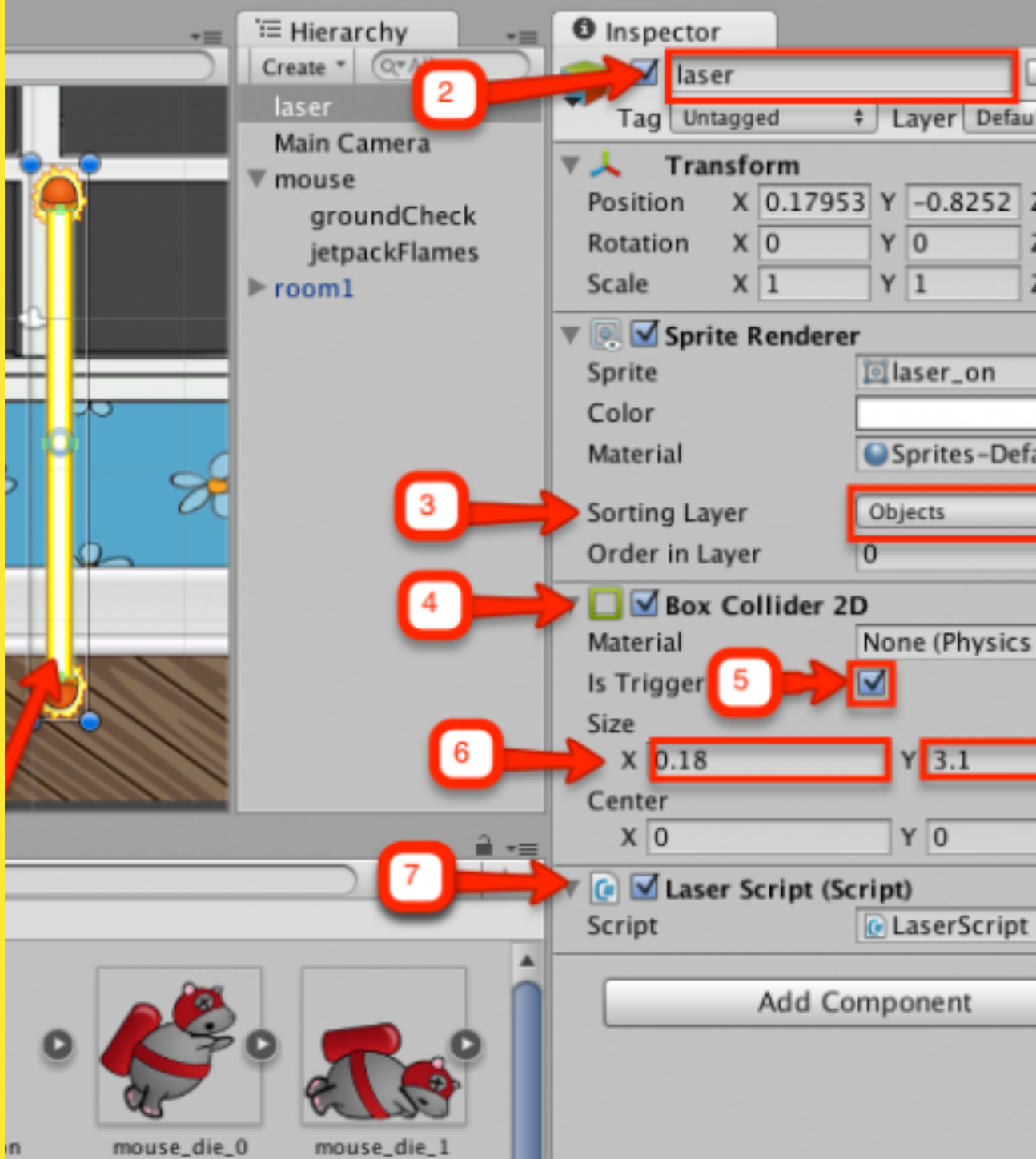
```
void FixedUpdate() {
    GenerateRoomIfRequired();
}
```

- *Arrastramos la instancia* `room1` *a* `currentRooms`
- *Arrastramos el prefab* `room1` *a* `availableRooms`



IMMA FIRIN MA LAZERS !!!

1. *Arrastramos el sprite* `laser_on` *a la escena*
2. *Lo renombramos a* `laser`
3. *Le establecemos la Sorting Layer a* `Objects`
4. *Le añadimos un Box Collider 2D / Activar* `Is Trigger` */ Size:* `(0.18, 3.1)`
5. *Creamos un script* `LaserScript.cs`*, lo vinculamos y abrimos el editor...*



```
public Sprite laserOnSprite;
public Sprite laserOffSprite;

public float interval = 0.5f;
public float rotationSpeed = 0.0f;

private bool isLaserOn = true;
private float timeUntilNextToggle;
private Collider2D collider;

void Start() {
    timeUntilNextToggle = interval;
    collider = GetComponent<Collider2D>();
}
```

```
void FixedUpdate() {
    timeUntilNextToggle -= Time.fixedDeltaTime;

    if (timeUntilNextToggle <= 0) {
        isLaserOn = !isLaserOn;
        collider.enabled = isLaserOn;

        SpriteRenderer spriteRenderer = GetComponent<SpriteRenderer>();
        if (isLaserOn)
            spriteRenderer.sprite = laserOnSprite;
        else
            spriteRenderer.sprite = laserOffSprite;

        timeUntilNextToggle = interval;
    }

    transform.RotateAround(transform.position, Vector3.forward, rotationSpeed * Time.fixedDeltaTime);
}
```

- ***Arrastramos el sprite*** `laser_on` ***al campo*** `Laser On Sprite` ***del script***
- ***Arrastramos el sprite*** `laser_off` ***al campo*** `Laser Off Sprite` ***del script***
- ***Establecemos la velocidad de rotación a 30***
- ***Position:*** `(2, 0.25, 0)`

EL RATÓN DEBE MORIR

ABRIMOS EL SCRIPT `MouseController.cs`

```
// Variables
private bool dead = false;

void OnTriggerEnter2D(Collider2D collider) {
    HitByLaser(collider);
}

void HitByLaser(Collider2D laserCollider) {
    dead = true;
}
```

```
void FixedUpdate() {
    bool jetpackActive = Input.GetButton("Fire1");
    jetpackActive = jetpackActive && !dead;

    if (jetpackActive)
        rb.AddForce(new Vector2(0, jetpackForce));

    if (!dead) {
        Vector2 newVelocity = rb.velocity;
        newVelocity.x = forwardMovementSpeed;
        rb.velocity = newVelocity;
    }
}
```

MONEY, MONEY, MONEY, MONEY

1. *Arrastramos el sprite* `coin` *a la Hierarchy*
2. *Añadimos un Box Collider 2D / Activar* `Is Trigger` */ Radius:* `0.23`
3. *Lo guardamos como prefab*

Ahora puedes usar tantos prefabs `coin` *como quieras !*

LE PUDO LA AVARICIA...

1. *Abrimos el desplegable* `Tag`
2. *Añadimos un nuevo tag* `Coins`
3. *Le establecemos dicho tag al prefab* `coin`
4. *Editamos el script* `MouseController.cs`*…*

```
// Variables
private uint coins = 0;

void CollectCoin(Collider2D coinCollider) {
    coins++;
    Destroy(coinCollider.gameObject);
}

void OnTriggerEnter2D(Collider2D collider) {
    if (collider.gameObject.CompareTag("Coins"))
        CollectCoin(collider);
    else
        HitByLaser(collider);
}
```

GENERACIÓN ESPONTÁNEA DE DINERO

1. *Creamos un nuevo* `Empty GameObject`
2. *Lo renombramos* `coin_group1`
3. *Ponemos varios prefab* `coin` *en la posición que más nos guste y los anidamos en el* `coin_group1`
4. *Guardamos* `coin_group1` *como prefab*
5. *Editamos el script* `GeneratorScript.cs`*…*

```
public GameObject[] availableObjects;
public List<GameObject> objects;

public float objectsMinDistance = 5.0f;
public float objectsMaxDistance = 10.0f;

public float objectsMinY = -1.4f;
public float objectsMaxY = 1.4f;

public float objectsMinRotation = -45.0f;
public float objectsMaxRotation = 45.0f;
```

```
void AddObject(float lastObjectX) {
    int randomIndex = Random.Range(0, availableObjects.Length);
    GameObject obj = (GameObject) Instantiate(availableObjects[randomIndex]);

    float objectPositionX = lastObjectX + Random.Range(objectsMinDistance, objectsMaxDistance);
    float randomY = Random.Range(objectsMinY, objectsMaxY);
    obj.transform.position = new Vector3(objectPositionX, randomY, 0);

    float rotation = Random.Range(objectsMinRotation, objectsMaxRotation);
    obj.transform.rotation = Quaternion.Euler(Vector3.forward * rotation);

    objects.Add(obj);
}
```

```
void GenerateObjectsIfRequired() {
    float playerX = transform.position.x;
    float removeObjectsX = playerX - screenWidthInPoints;
    float addObjectX = playerX + screenWidthInPoints;
    float farthestObjectX = 0;
    List<GameObject> objectsToRemove = new List<GameObject>();

    foreach (var obj in objects) {
        float objX = obj.transform.position.x;
        farthestObjectX = Mathf.Max(farthestObjectX, objX);

        if (objX < removeObjectsX)
            objectsToRemove.Add(obj);
    }

    foreach (var obj in objectsToRemove) {
        objects.Remove(obj);
        Destroy(obj);
    }

    if (farthestObjectX < addObjectX)
        AddObject(farthestObjectX);
}
```

```
void FixedUpdate() {
    GenerateRoomIfRequired();
    GenerateObjectsIfRequired();
}
```

Configuremos el script:

1. *Arrastrar las instancias de objetos (monedas, lásers...) al array* `Objects`
2. *Arrastrar los prefabs de objetos a la lista* `Available Objects`

WELL DONE !