CRE ATE C# Workshop

Inhoud

[1. Make a game object move 1](#_Toc33378859)

[2. Make a game object move through player input 5](#_Toc33378860)

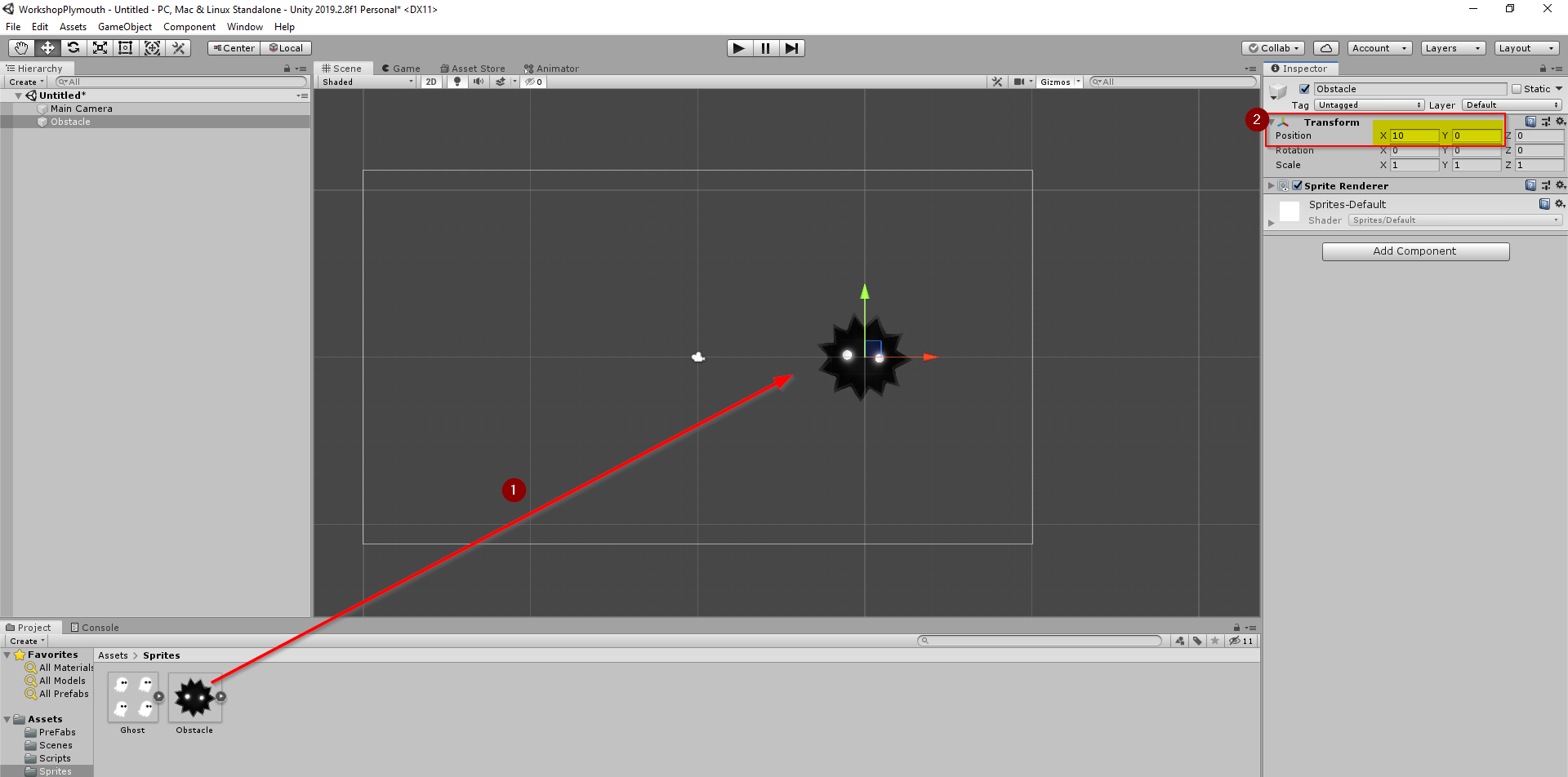
[3. Make game object able to collide to each other 6](#_Toc33378861)

[4. Add health to game object 8](#_Toc33378862)

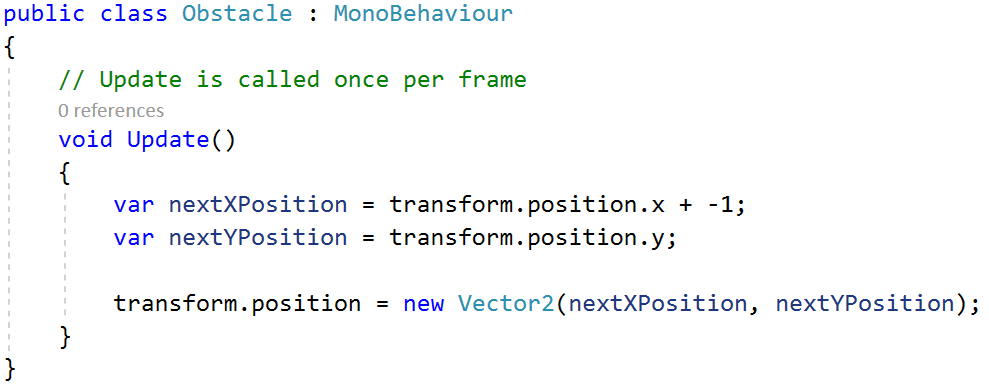
[5. Add spawn points 9](#_Toc33378863)

[6. Create a new spawn point 11](#_Toc33378864)

# Make a game object move

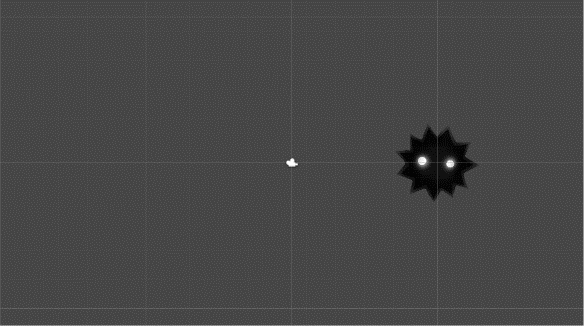


1. Drag Sprite On the Scene.
2. Every Game object has a ‘Transform’. With a transform we can set the position of the object on the scene.
3. Create a new script to make the object move.
4. Update method; for every frame this code will be ran. We want the object to move to the left.
   1. The position can be changed by creating a new ‘Vector2’ with the next x and y position.



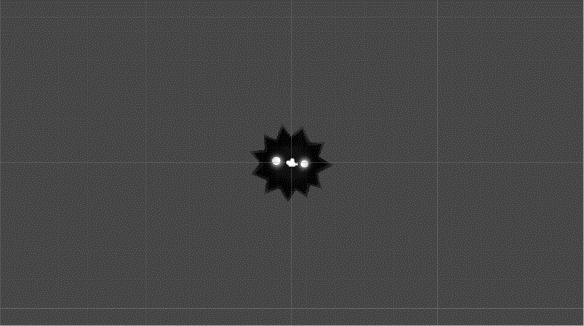
1. Attach the script to the game object by dragging the script into the inspector view.



Example

**Frame1:**

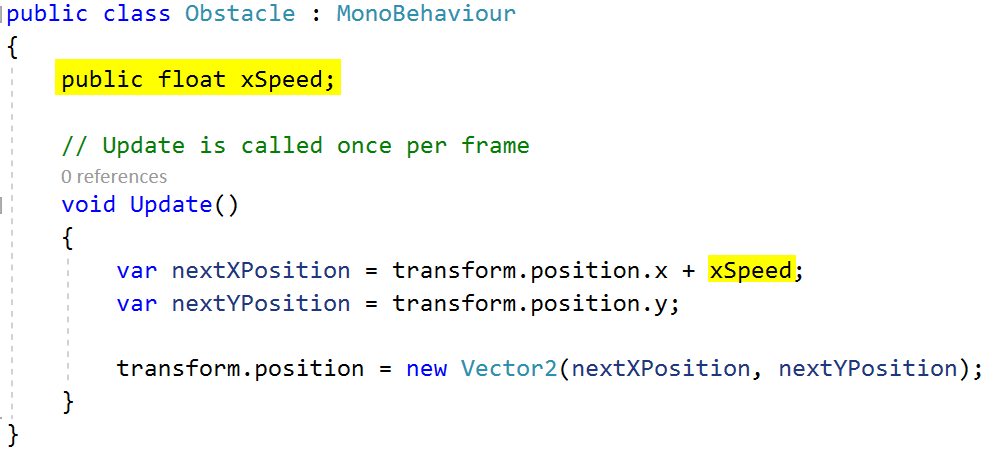
Transform.postion.x = 10;   
Transform.postion.y = 0;



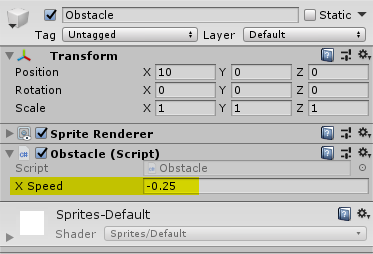
**Frame2:**

Transform.postion.x = 0;   
Transform.postion.y = 0;

We can regulate the speed of the X position in the Unity Environment by declaring a public variable.

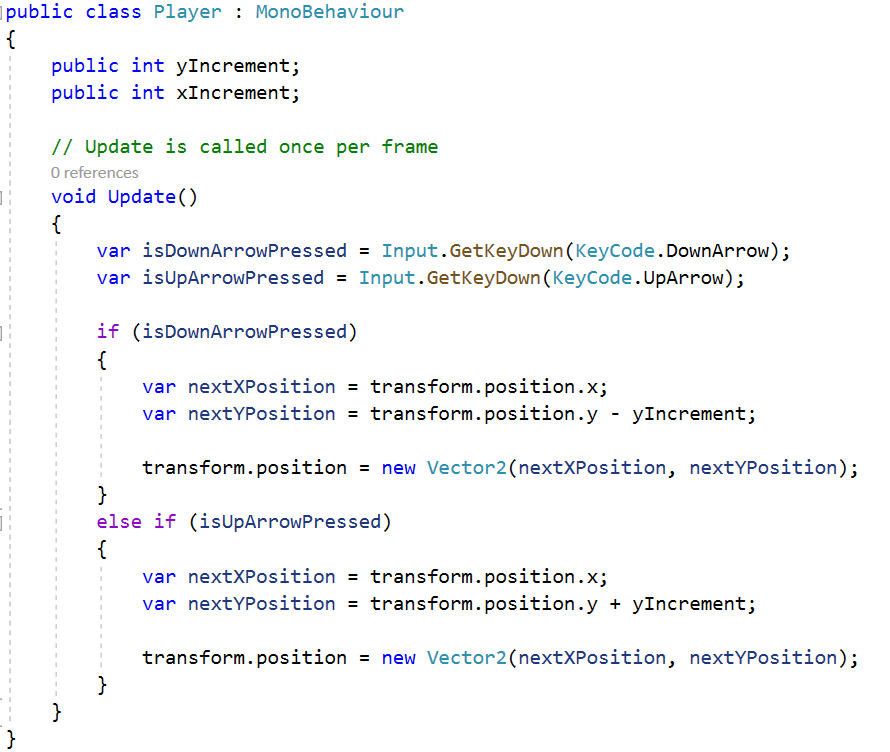


Now we can change the xSpeed of the obstacle in the Unity Environment.



# Make a game object move through player input

1. Drag the Ghost sprite in the scene.
2. Create a new script ‘player’
   1. Two public variables
      1. **yIncrement**: control the amount of units the ghost will increment in the Y position
      2. **xIncrement**: control the amount of units the ghost will increment in the X position.
   2. Update method
      1. Check if the UP or DOWN arrow key is pressed.
         1. When Down is pressed we decrease the Y position by the **yIncrement**.
         2. When Down is pressed we increase the Y position by the **yIncrement**.
3. Attach the script to the the Ghost.



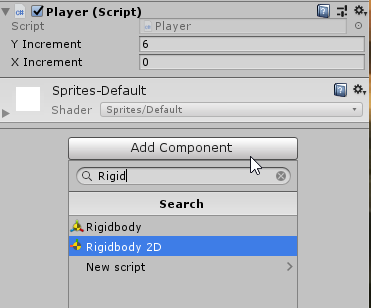
# Make game object able to collide to each other

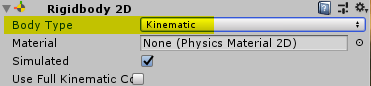
1. Select the ghost object and add following components to the object.

Add components to Ghost object:

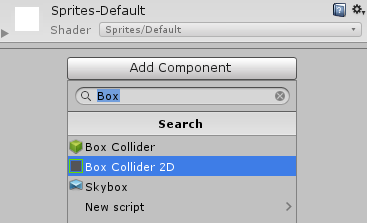
* Rigidbody 2D
* Box Collider 2D

Add **Rigid body 2D** and set its **Body type** to **Kinematic**

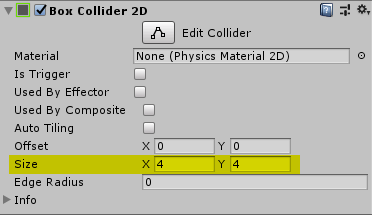
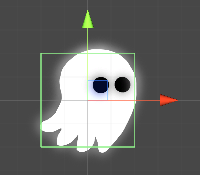




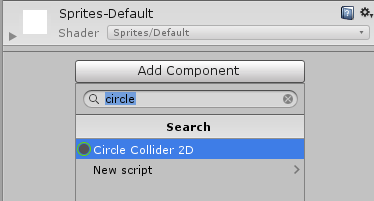
Add **box Colider 2D**



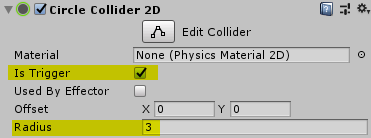
You can adjust the size of the box collider to make it fit the ghost sprite.

1. Add a circle collider to the obstacle object

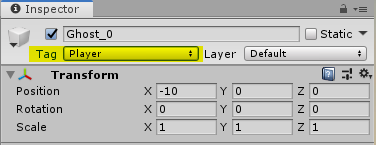


Set the **‘Is Trigger’** checkbox **ON,** this will allow us to tell the obstacle what to do when it collides with the player.

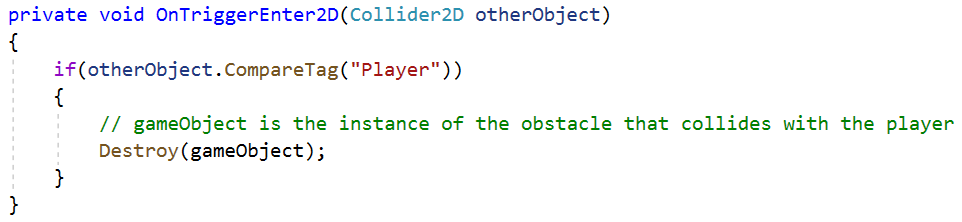


1. Set the ‘Player’ tag to the Ghost

Select the ghost, go to the inspector window. The first component has a property ‘Tag’. Change it to ‘Player’.

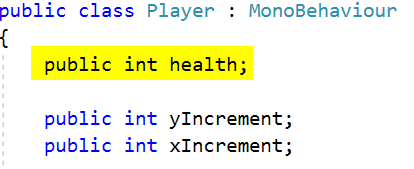


1. Writing the code in the obstacle script
   1. Add a new function ‘OnTriggerEnter2D’
      1. Check if the object you are colliding with is a Player. (It is possible that the obstacle is colliding with another obstacle, so we need to make sure the other object is the player/ghost)
      2. When it is a player we are colliding with, we destroy the obstacle instance.

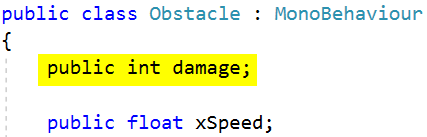


# Add health to game object

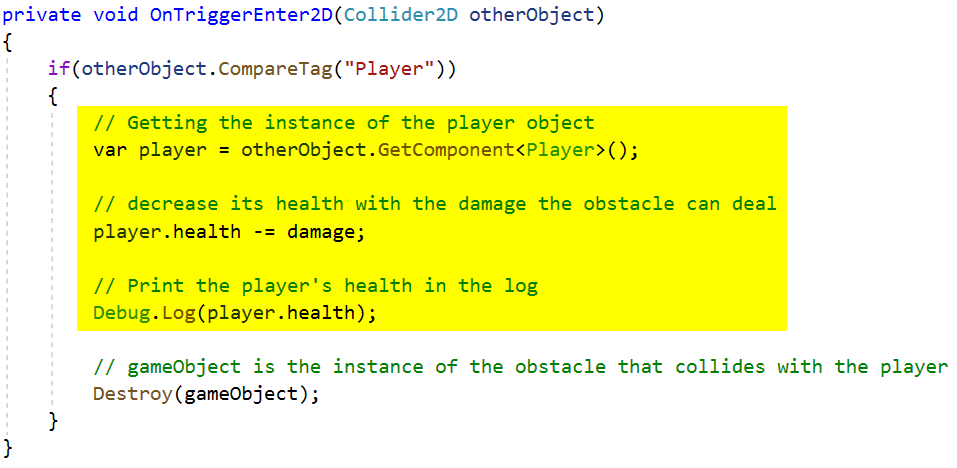
1. Add a new variable, ‘Health’, in the player script.



1. Set the health variable to 3 in the Unity environment
2. In the obstacle script, add a new variable ‘Damage’



1. In the obstacle script. Remove 1 health point of the obstacle.



# Add spawn points

1. Create a **PreFab** of the obstacle

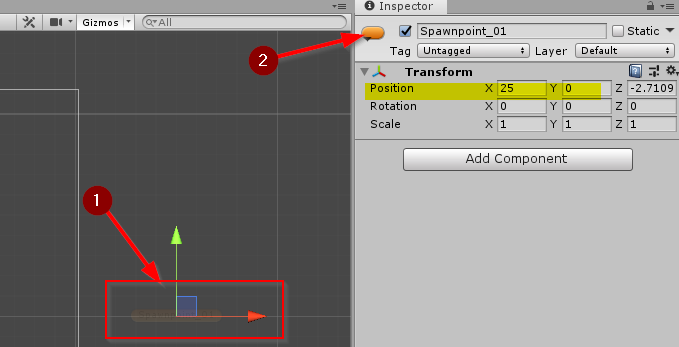
Because we want eventually to have many obstacles in our game, we need to create a PreFab out of it.

Select the Obstacle object in the hierarchy view and drag it in the prefabs folder. Now the Obstacle will be highlighted blue in the hierarchy view.   
Delete the Obstacle out of the hierarchy view.

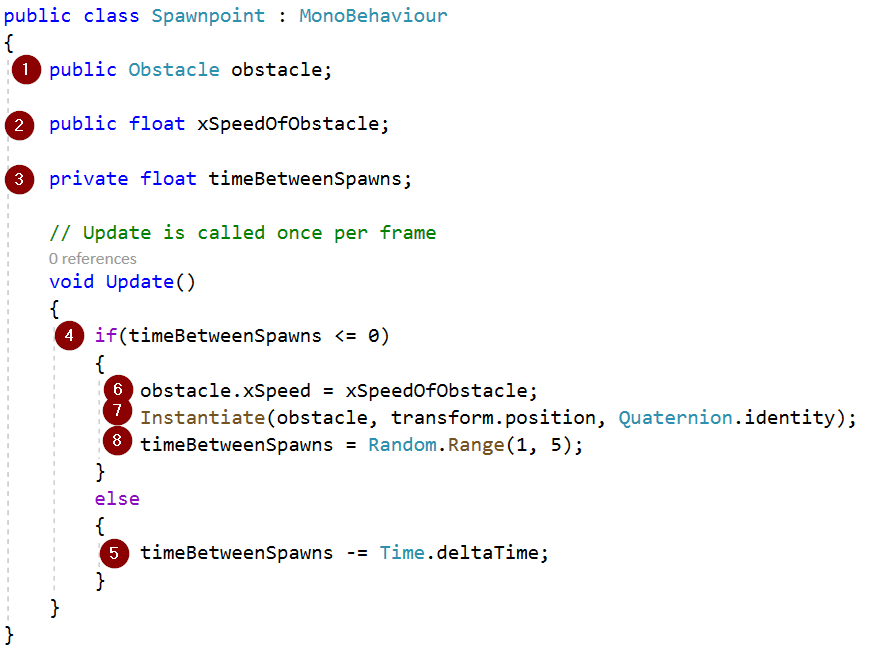
1. Add a new empty game object in the hierarchy view, call it ‘Spawnpoint\_01’.

Set its X and Y position to the height where the obstacles needs spawn.

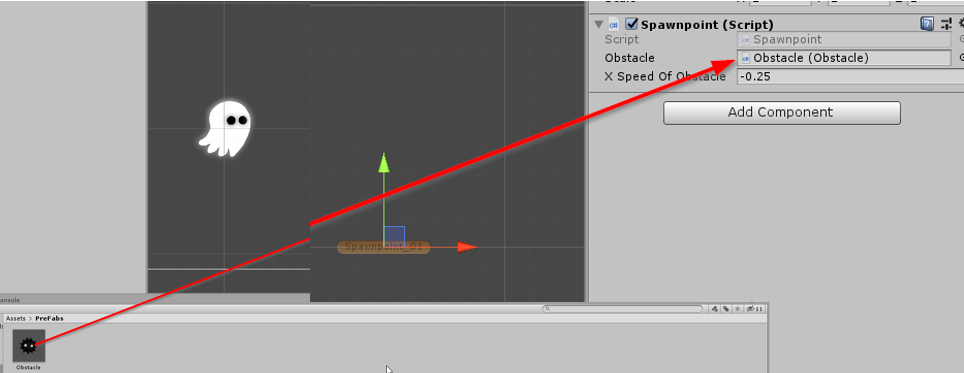
* + - 1. The Spawn point 01
      2. Change the appearance in the Scene so we can see the object in the scene.



1. Create a new script ‘SpawnPoint’
   * + 1. Obstacle: a variable to store the prefab of our obstacle object
       2. When creating a new obstacle we could set the xSpeedOfTheObstacle before spawning it.
       3. This is the time between the two spawning objects
       4. If the time between the two spawing objects is below zero, we have to spawn a new obstacle.
       5. Else, we need to subtract the time from ‘timeBetweenSpawns’
       6. Set the xSpeed of the obstacle to te xSpeed that the spawnpoint defines
       7. Placing the new obstacle at the current position of ‘spawnpoint\_01’.
       8. Reset the next spawn time to a random time between 1 and 5 seconds.



1. Set the xSpeed and attache the Obstacle



# Create a new spawn point

It would be more challenging that we have more than one spawn point. Perhaps an obstacle that spawns from behind the ghost.

1. Duplicate ‘spawnpoint\_01’ and call it ‘spawnpoint\_02’
2. Set ‘spawnpoint\_02’ to the location you want.
3. Adjust the values of the script component for ‘spawnpoint\_02’

