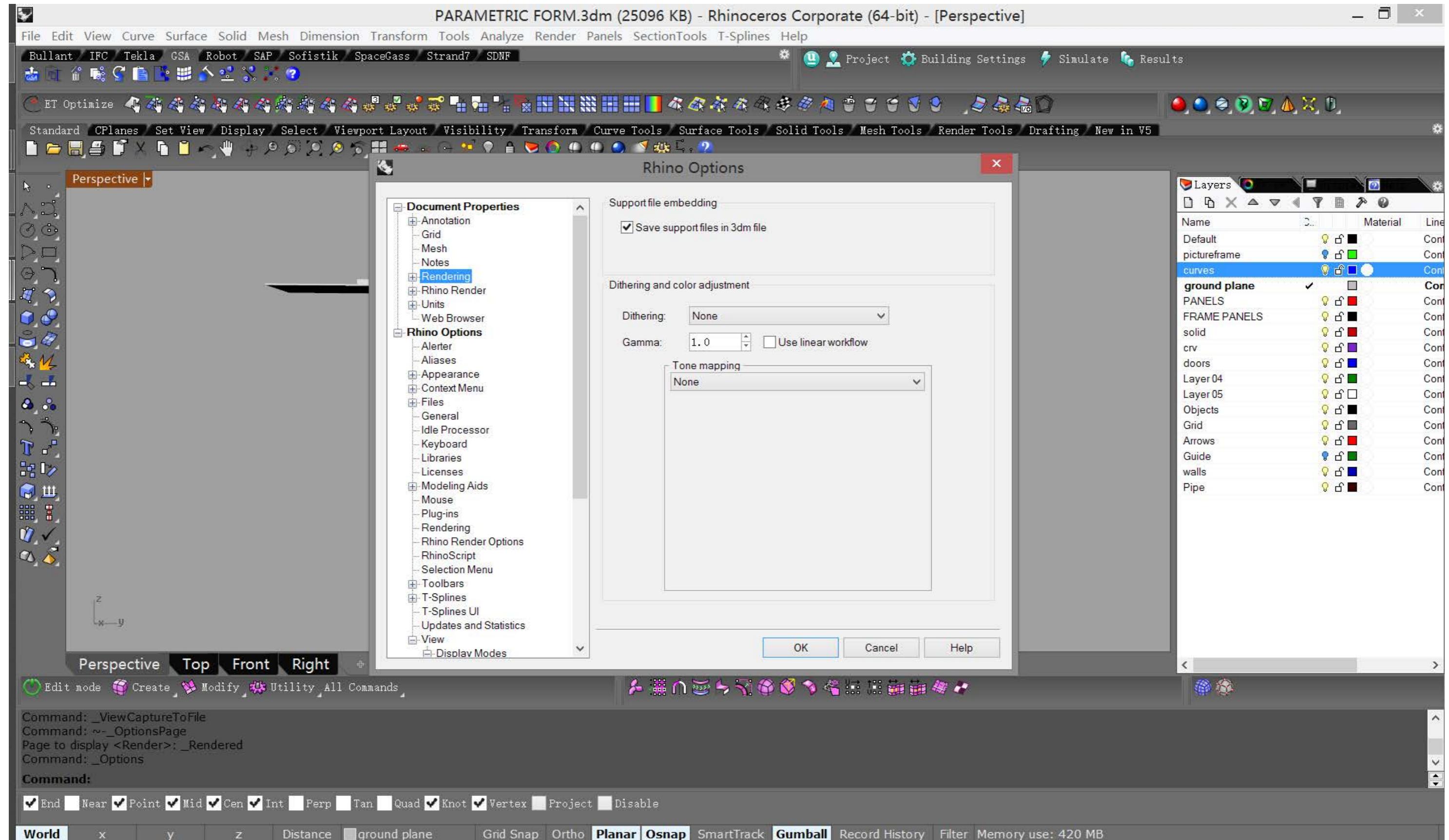


PARAMETRIC FORM .COM

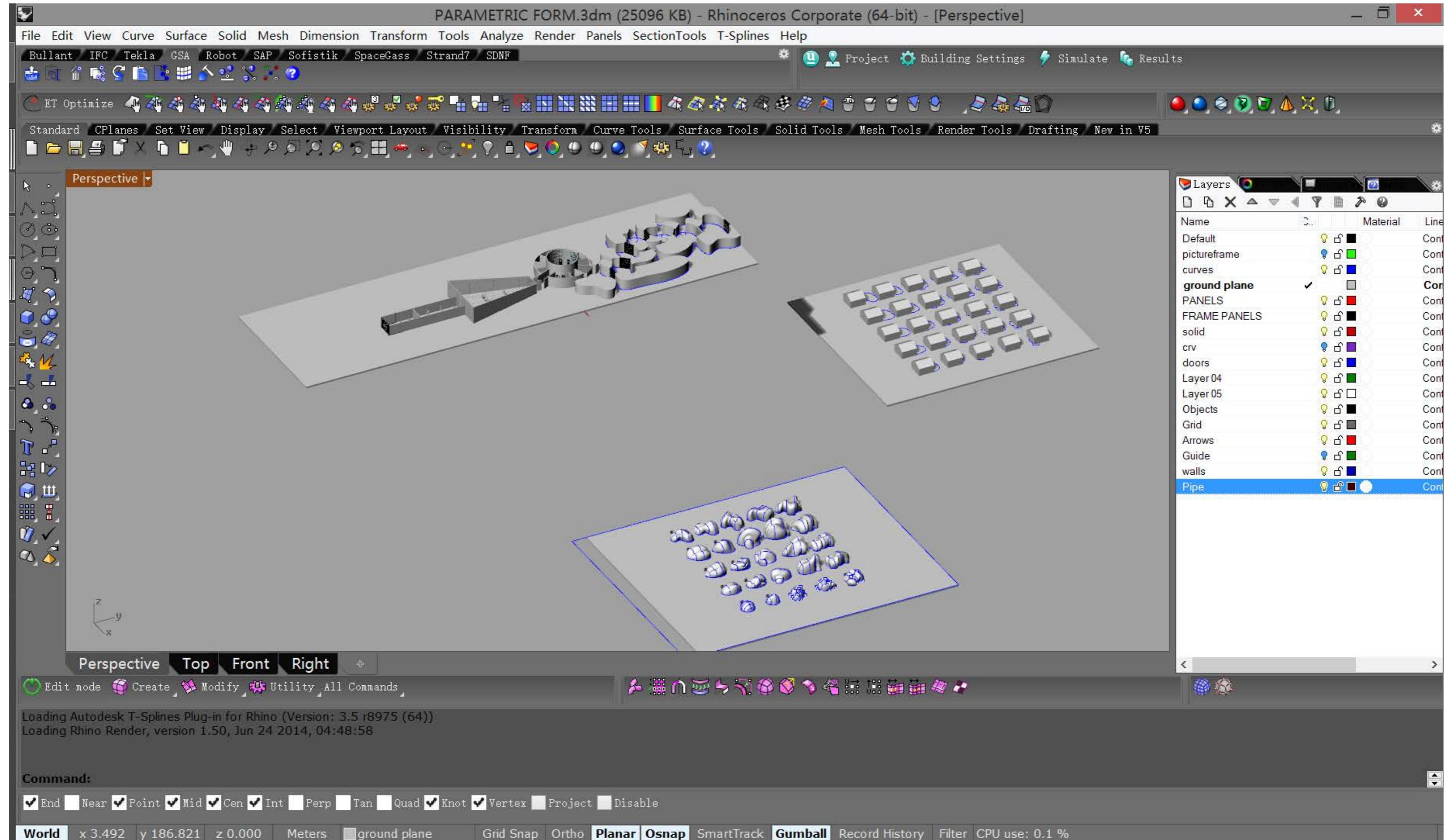
Gameify! (Unity 3D)  
Digital Tools  
Josep Alcover Llubiá + Nohelia Gonzalez + Jinyang Han **Iaac**

## Setting up Rhino

To be able to work in different files (with textures applied) and then join them without losing the textures we enable “save support files in 3dm file” in the options menu.

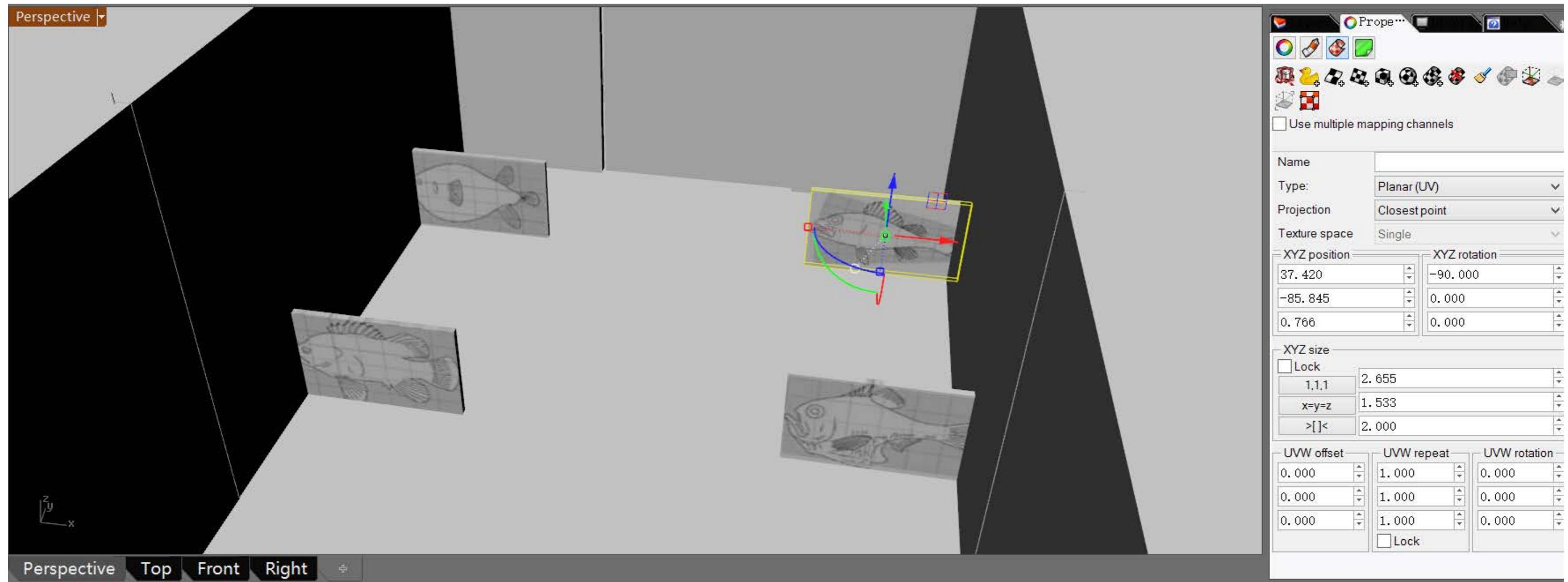


# Modeling a virtual world in Rhino

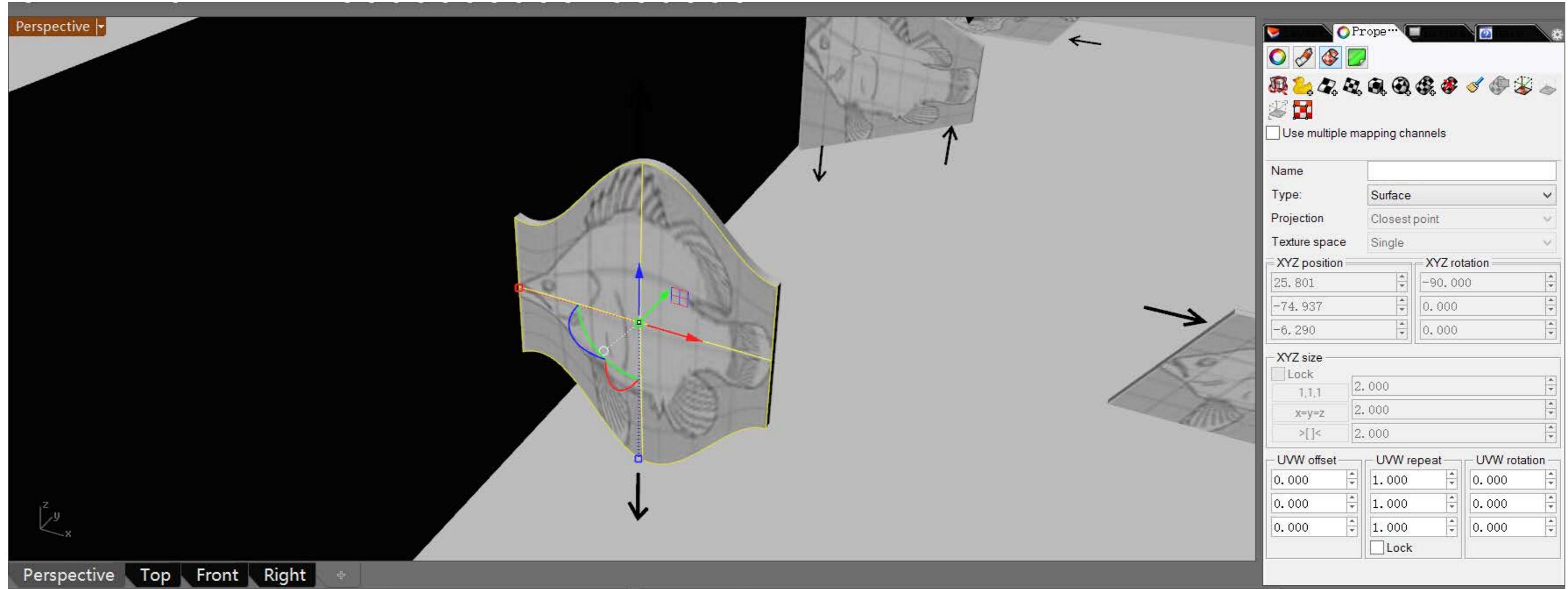


## Applying textures to the objects

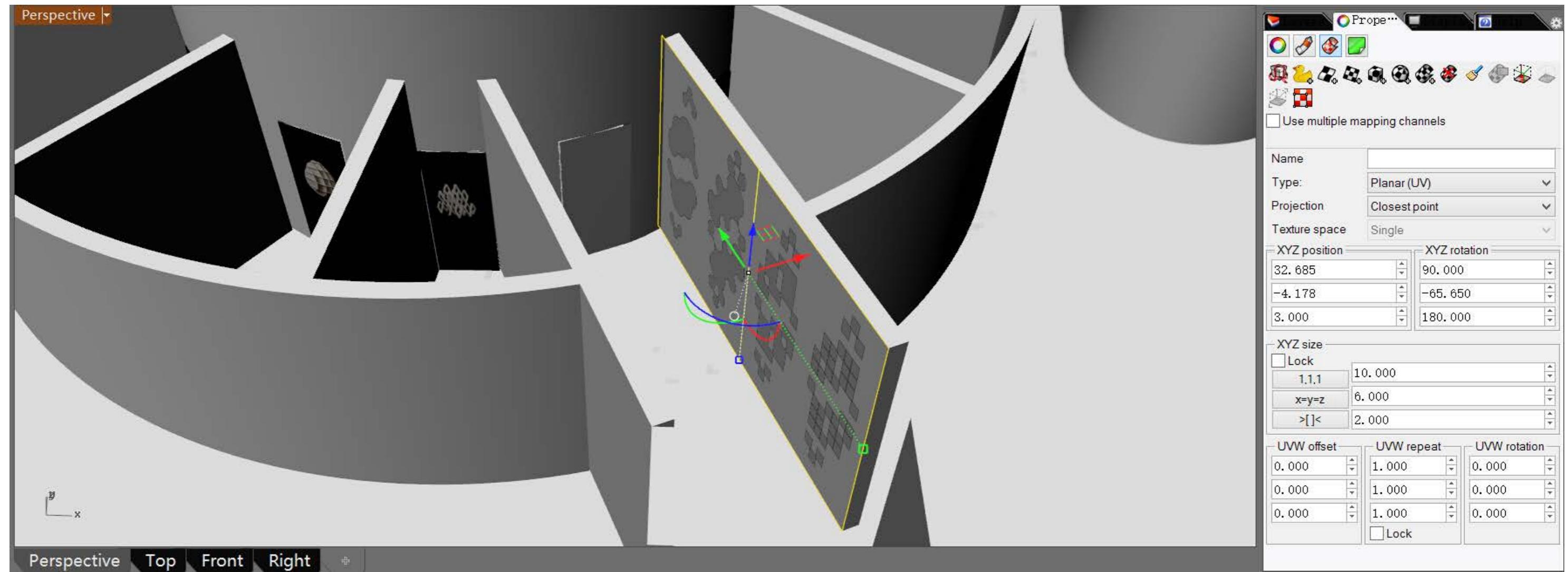
We used different types of UV mapping  
(Box, Surface, Plane)



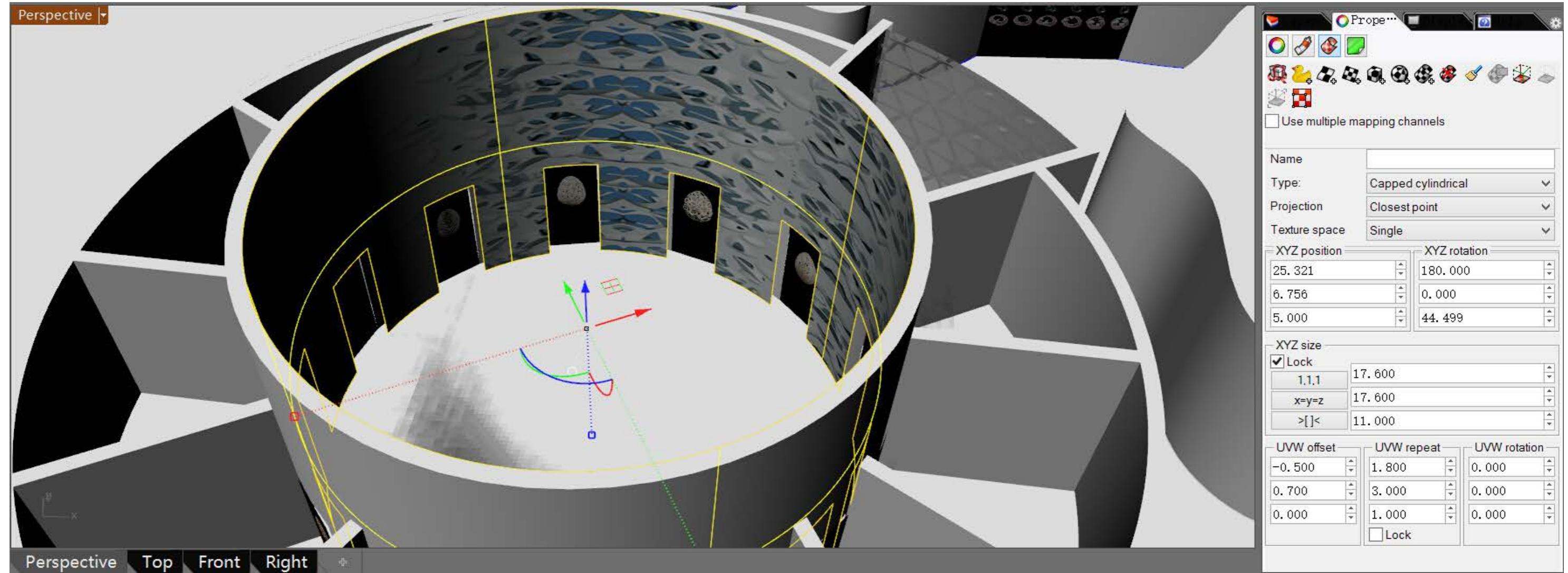
## Applying textures to the objects



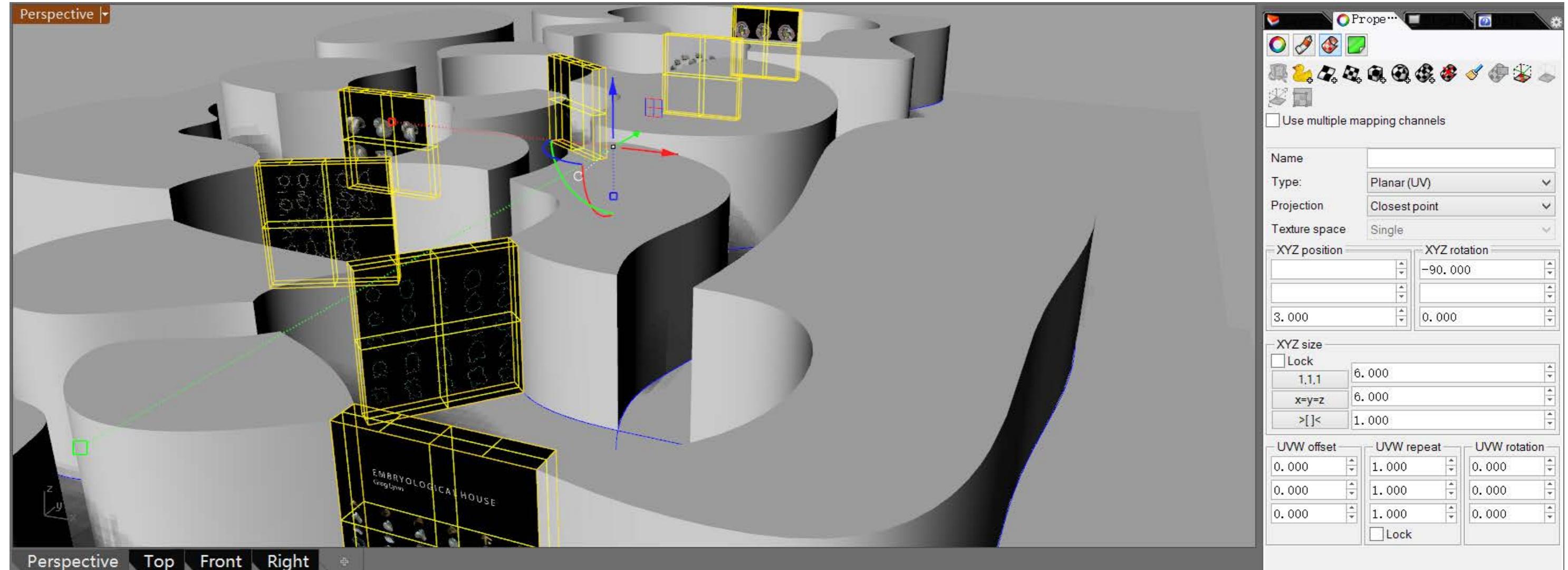
## Applying textures to the objects



## Applying textures to the objects

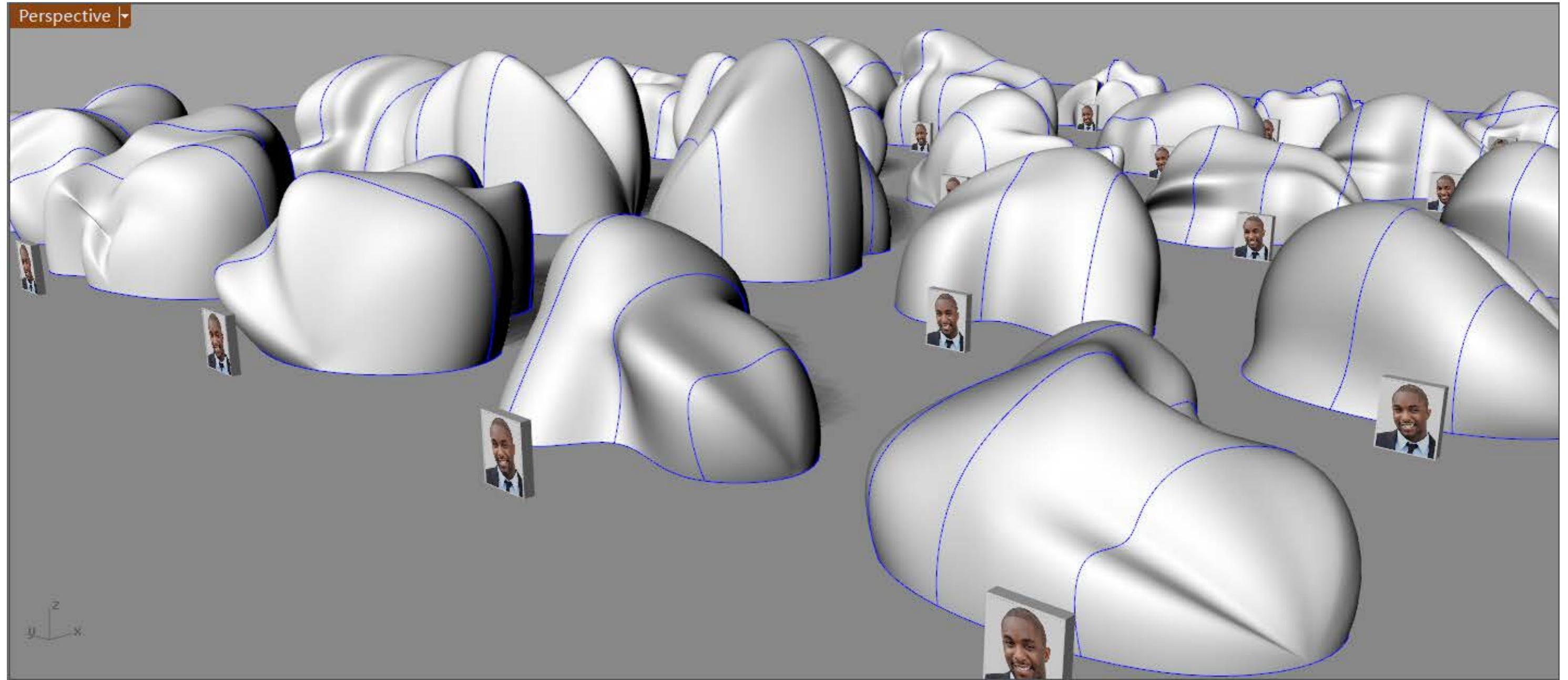


## Applying textures to the objects



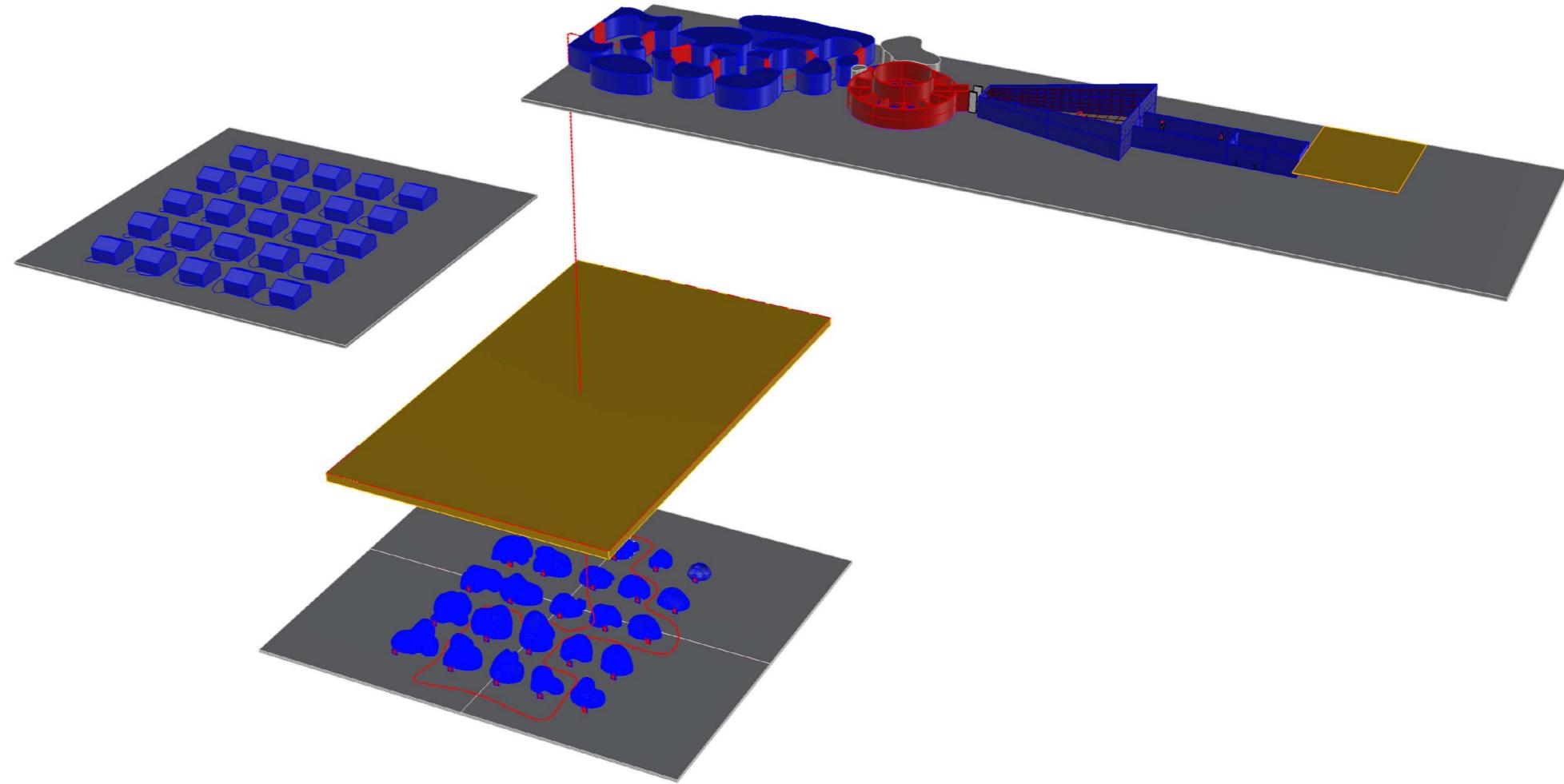
## Applying textures to the objects

Since the textures will have to be applied again in Unity,  
we apply the same texture to all the objects with the  
same proportion just to create the UV projector.

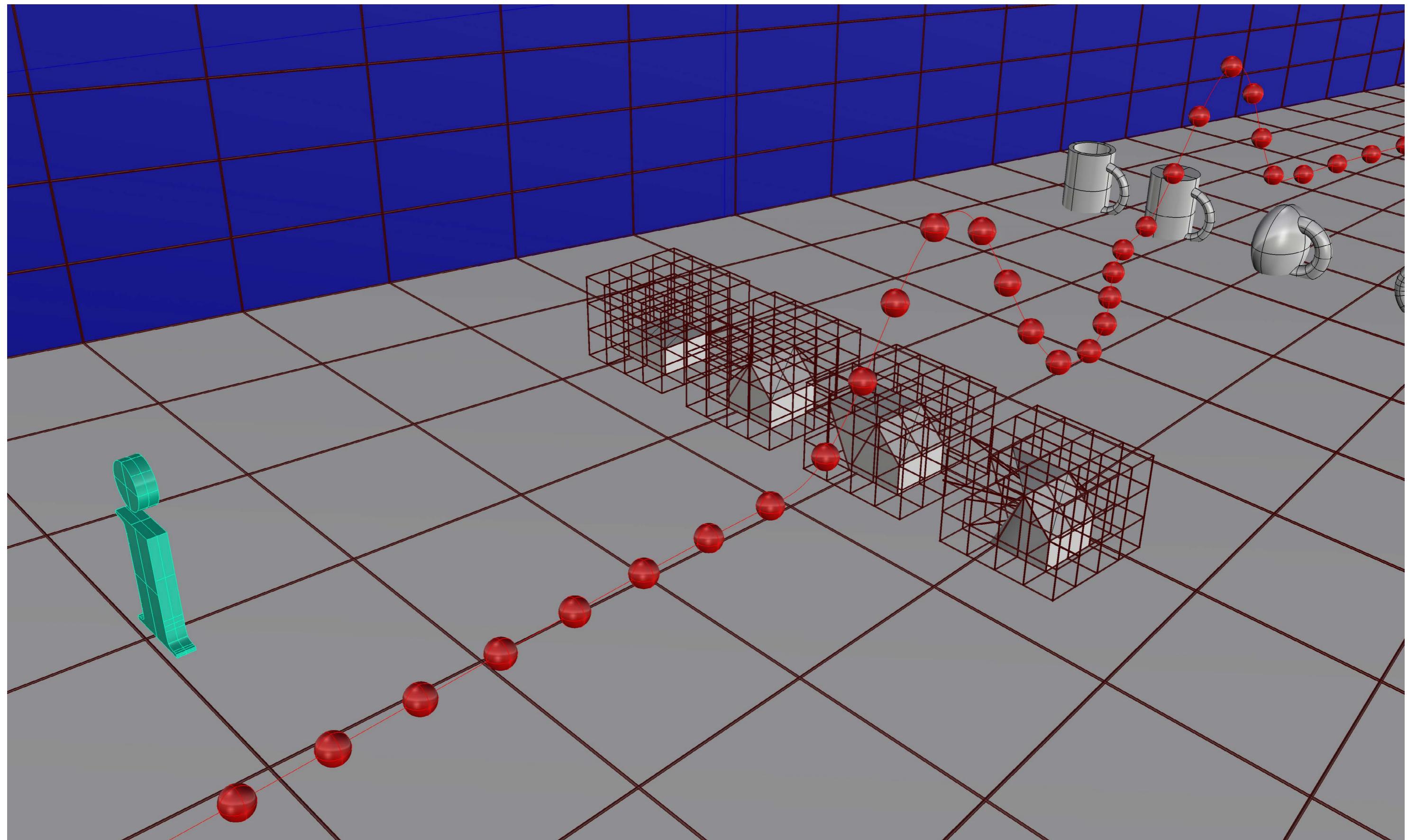


## Modeling the “guiding” objects

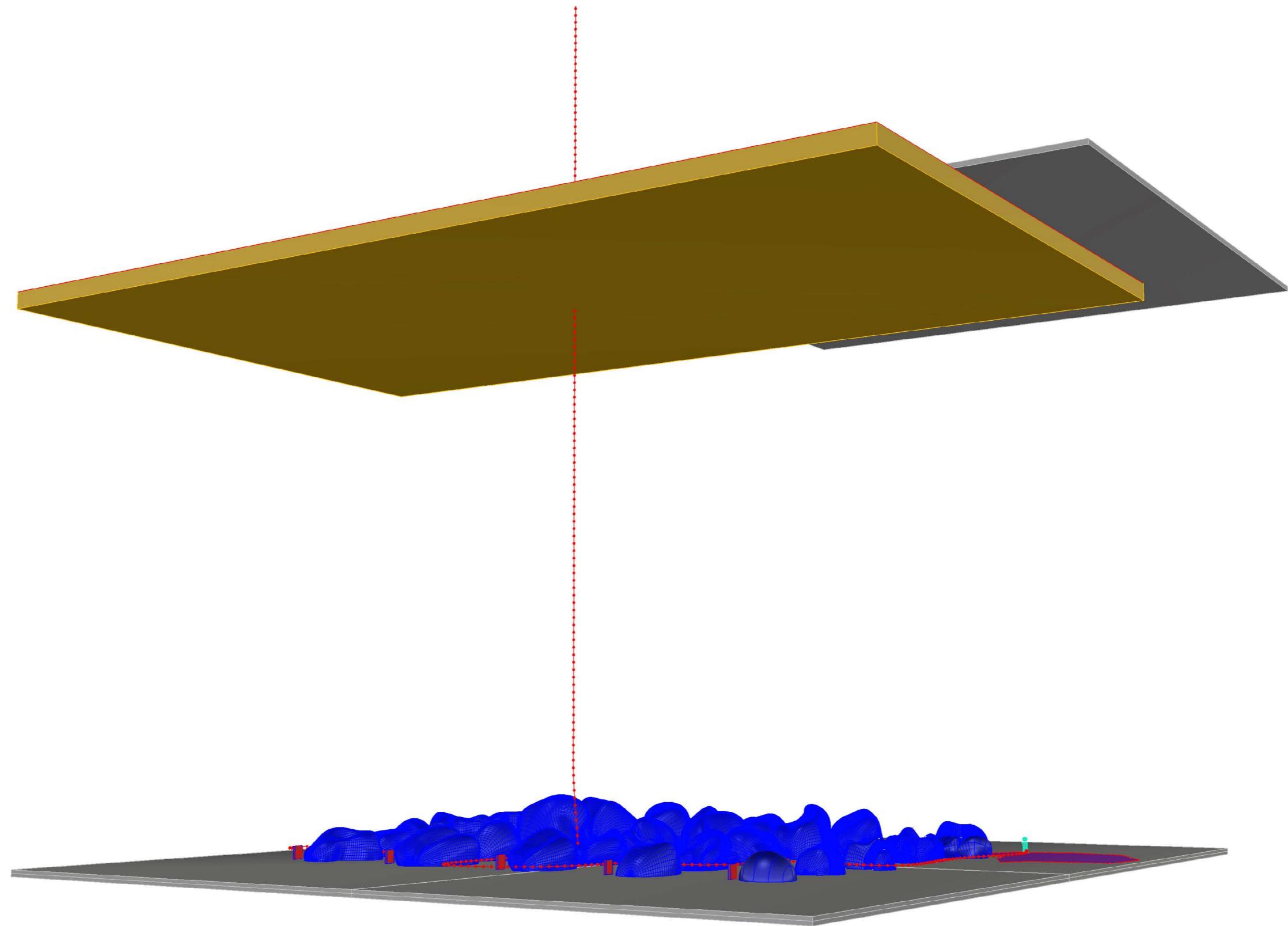
To guide the player through the game we create a path of floating spheres, some information points with floating “i”, and some objects that will be transparent to hold a message that will be displayed when the player collides with them.



## Modeling the “guiding” objects

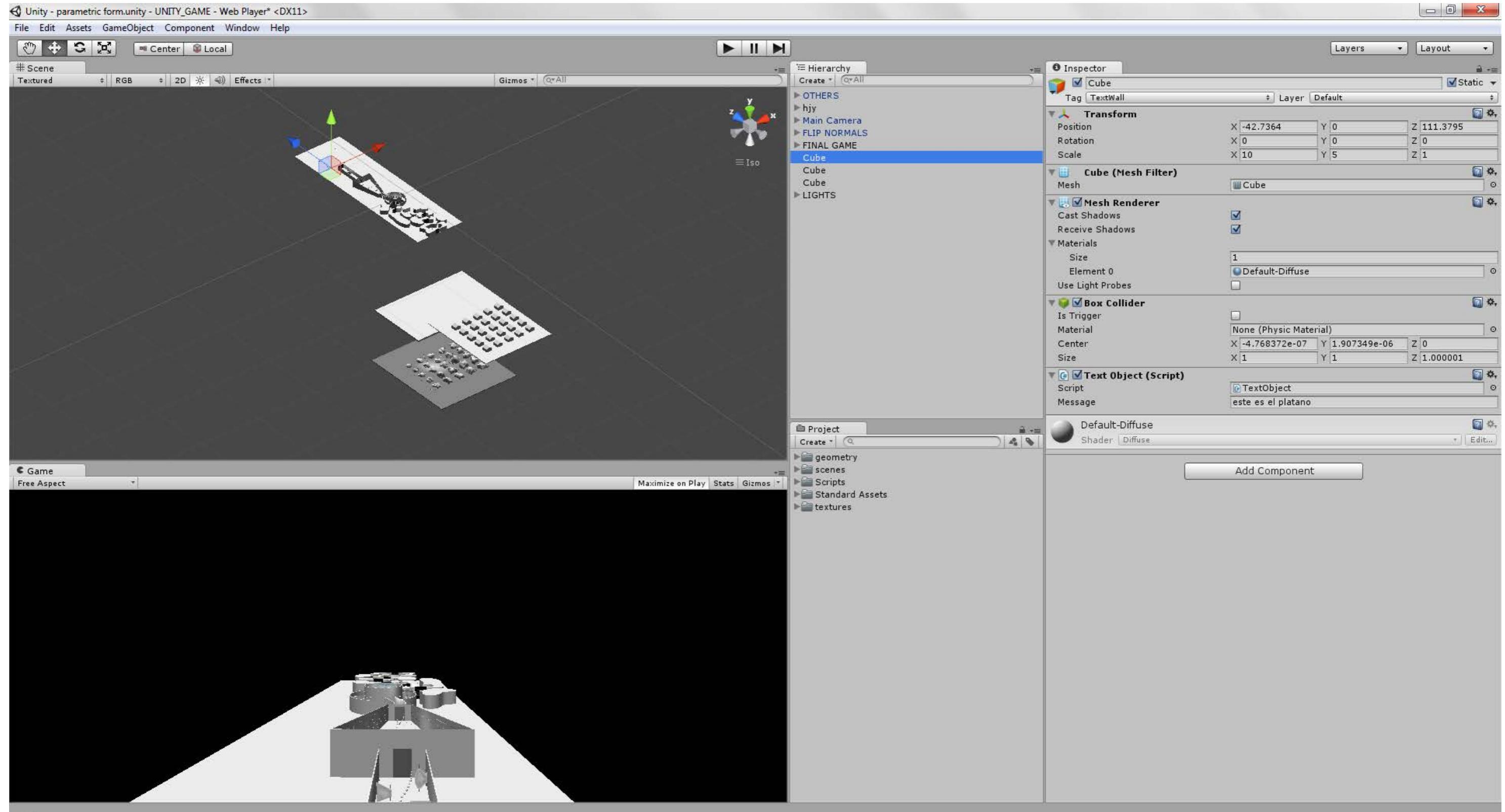


## Modeling the “guiding” objects



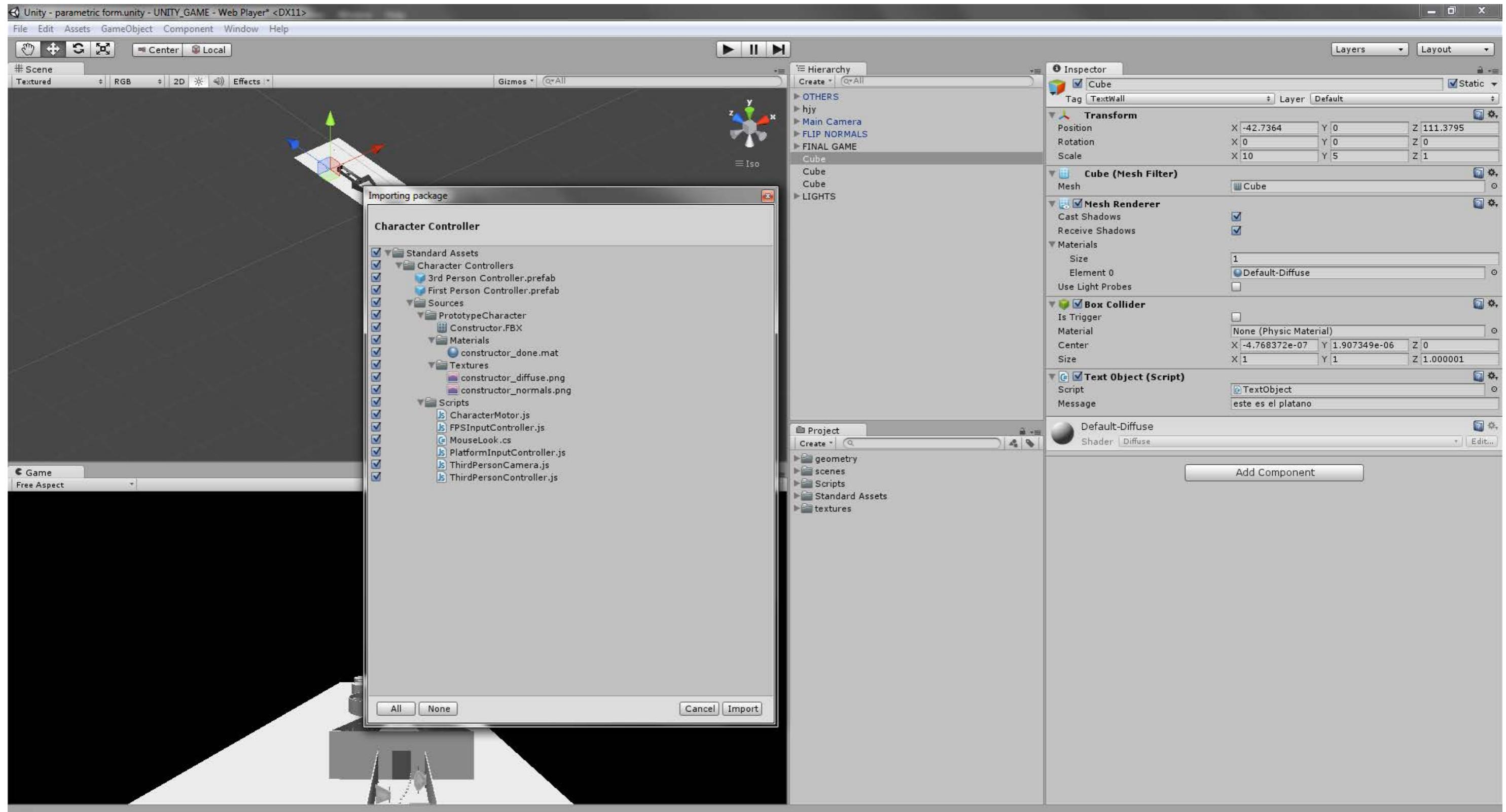
## Importing the geometry into Unity

We export the geometry from Rhino in .OBJ format (that generates a folder with all the textures) and we import it as a new asset into Unity.  
We generate colliders for all those objects that we don't want to be "physically transparent" (especially ground planes).



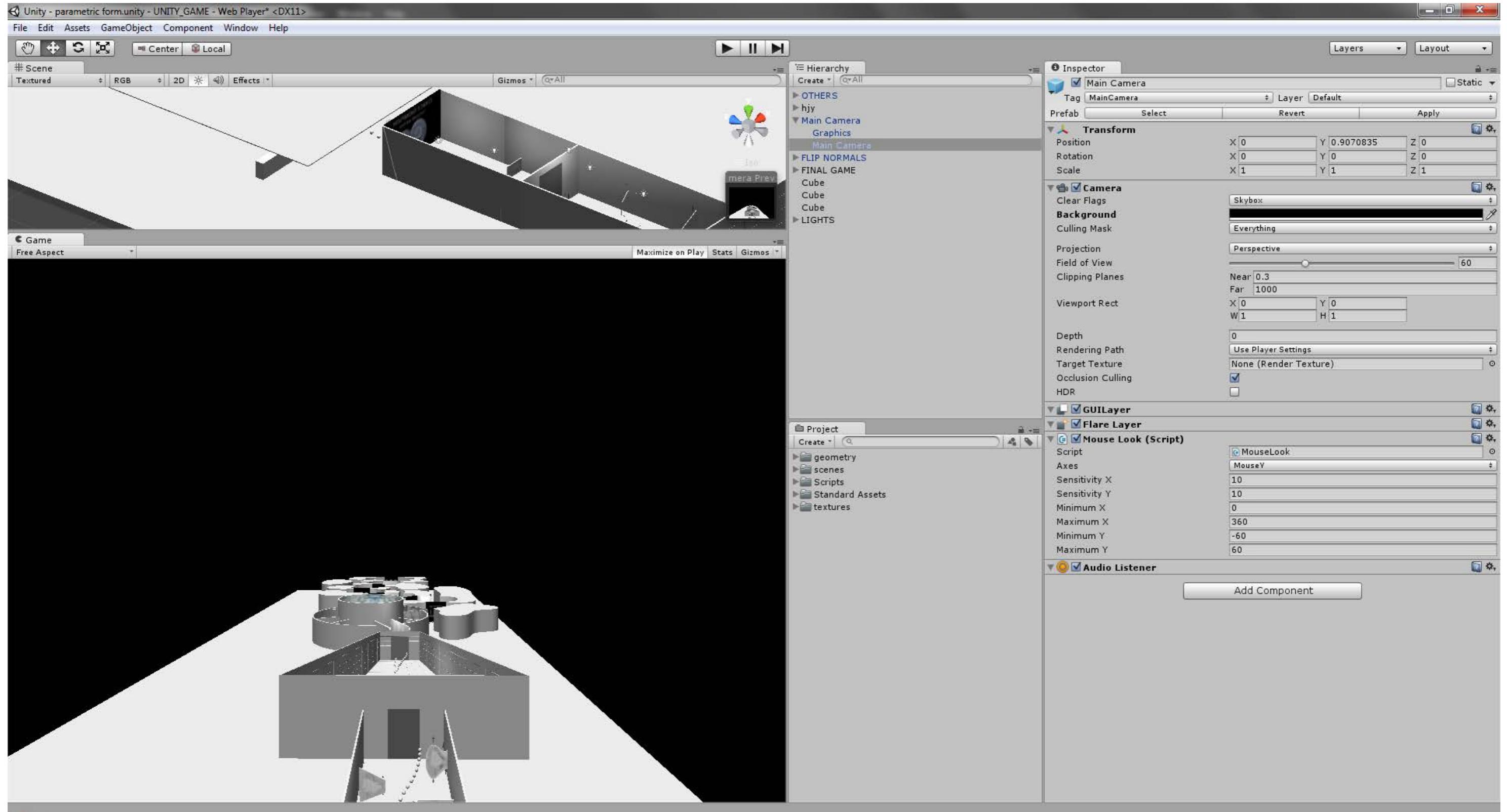
## Importing the Character Controller package

To be able to use the mouse and keyboard to move around the game we import the "Character Controller package" and we use the first person controller.



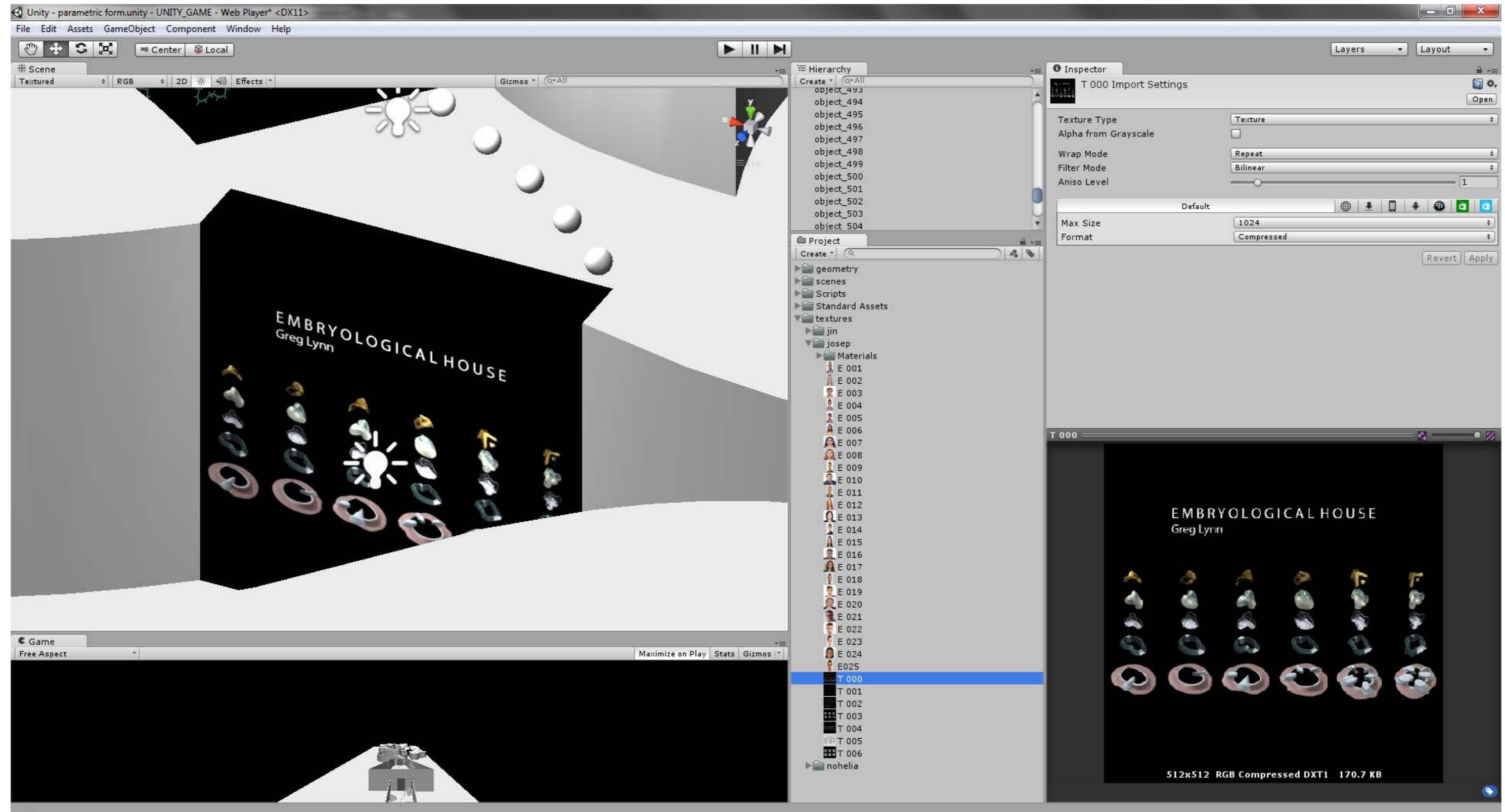
## Changing the Background Color

To create a more abstract world we change the background color to a dark grey instead of blue.



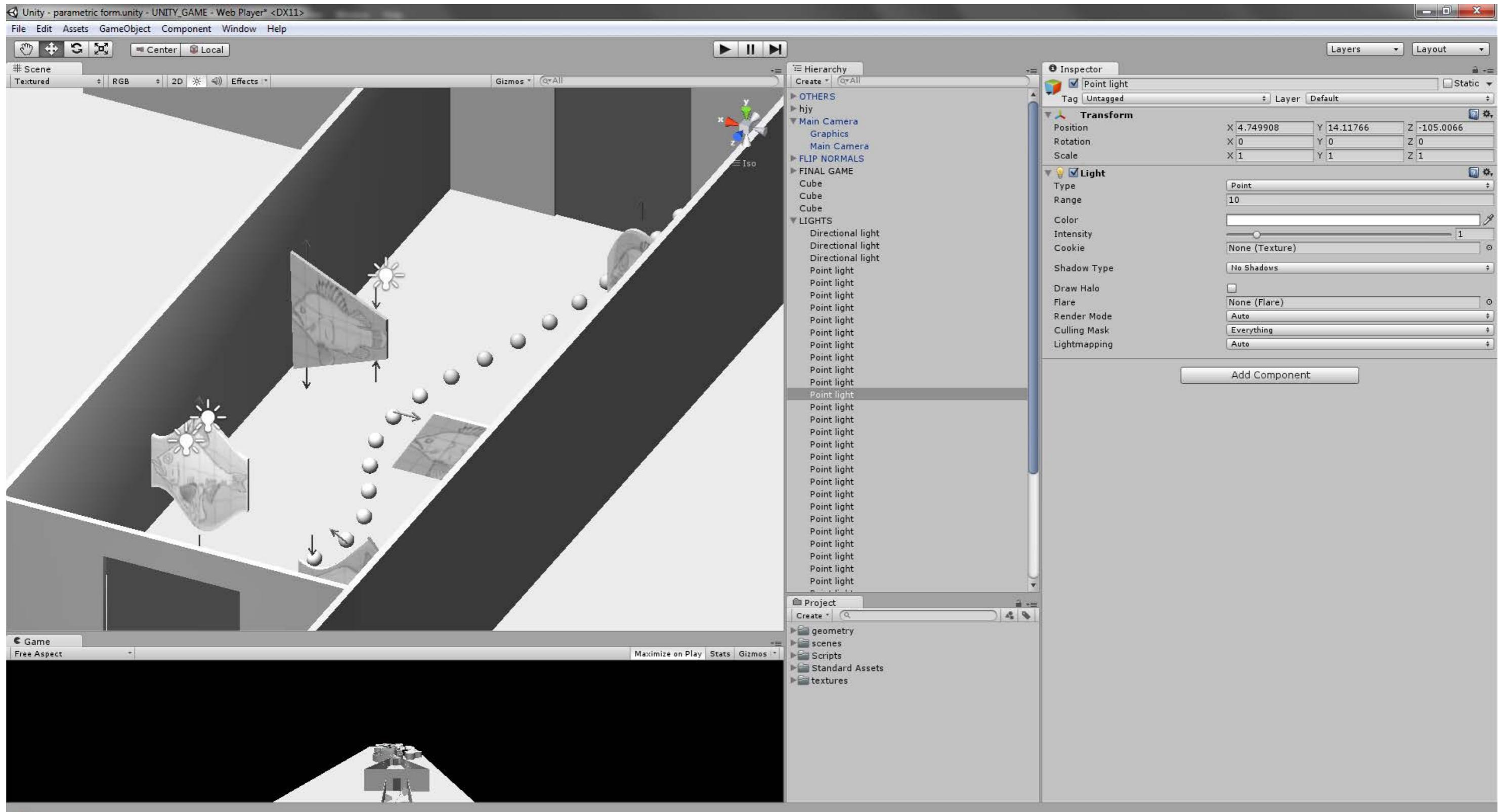
## Reassigning Textures

Each texture has to be reassigned manually to its object.



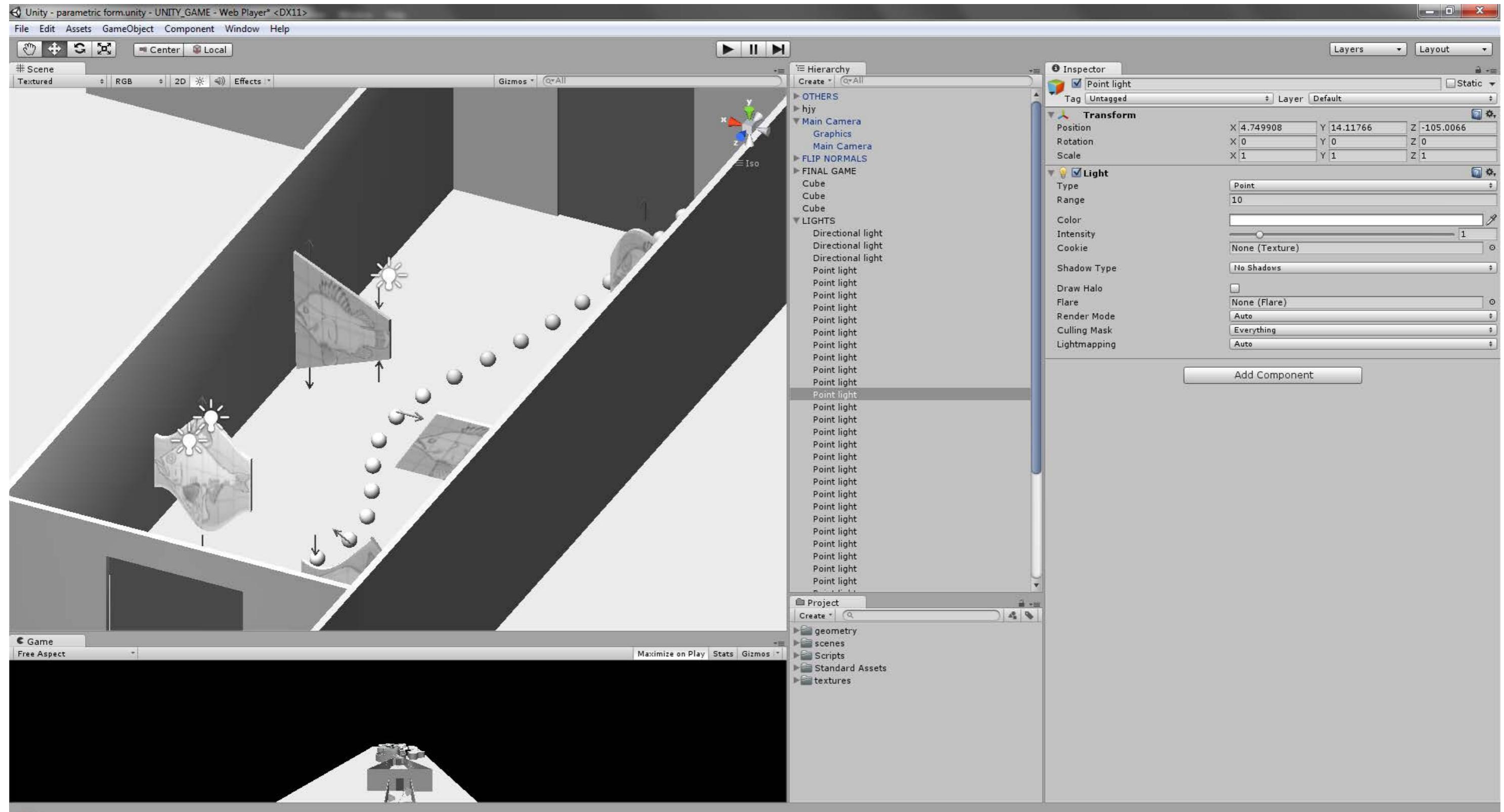
## Creating Lights

To light the scene we use some directional and spot lights.



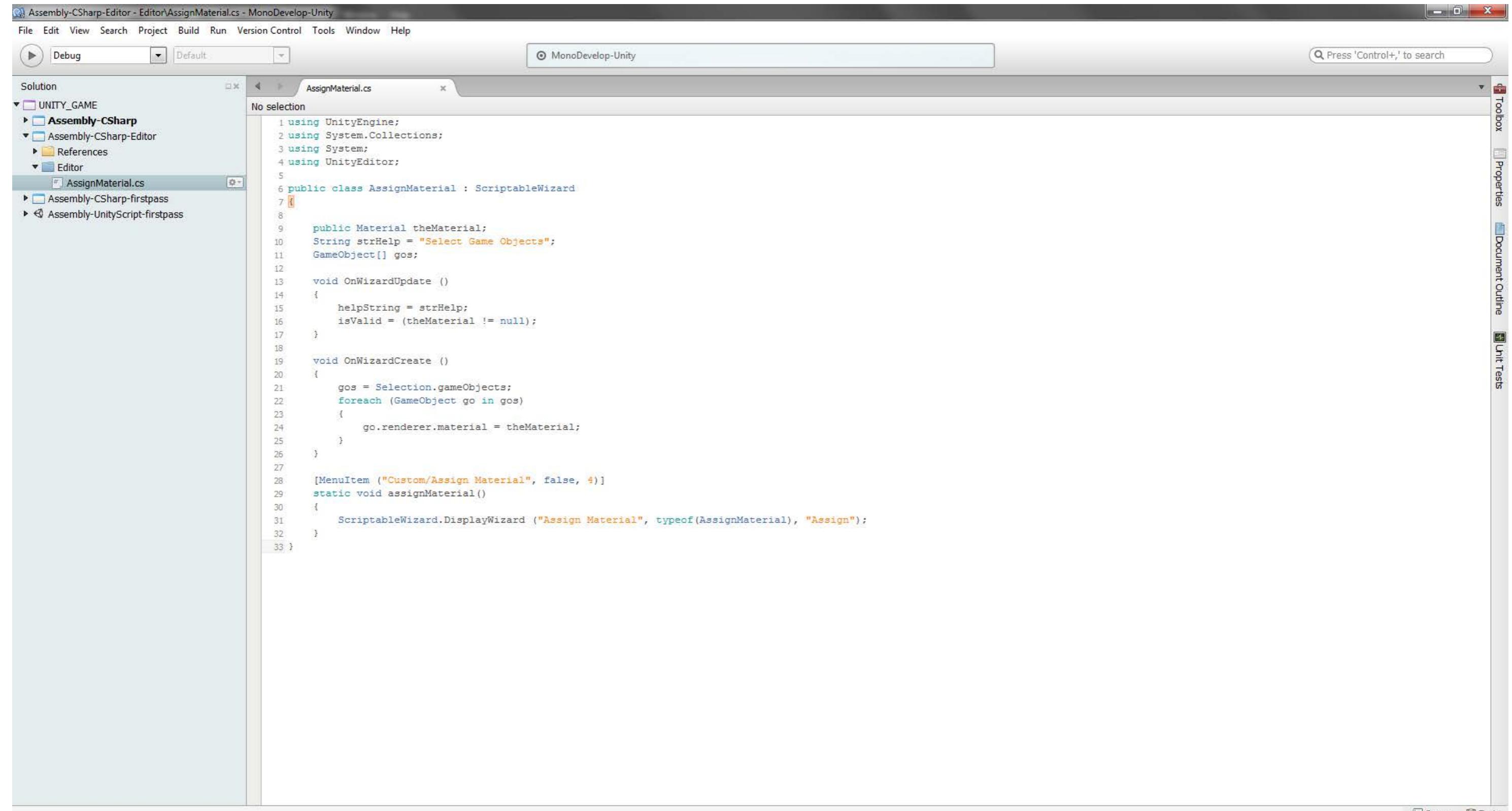
## Assigning materials to multiple objects

Unity doesn't have an option to assign one material to multiple objects at the same time. However we need to assign the same material to all the spheres that create the path to guide the player. The solution is creating a script and save it in a folder called "Editor" inside the "Assets" Folder. By doing so we will be able to access the command from the top toolbar.



# Assigning materials to multiple objects

## AssignMaterial Script



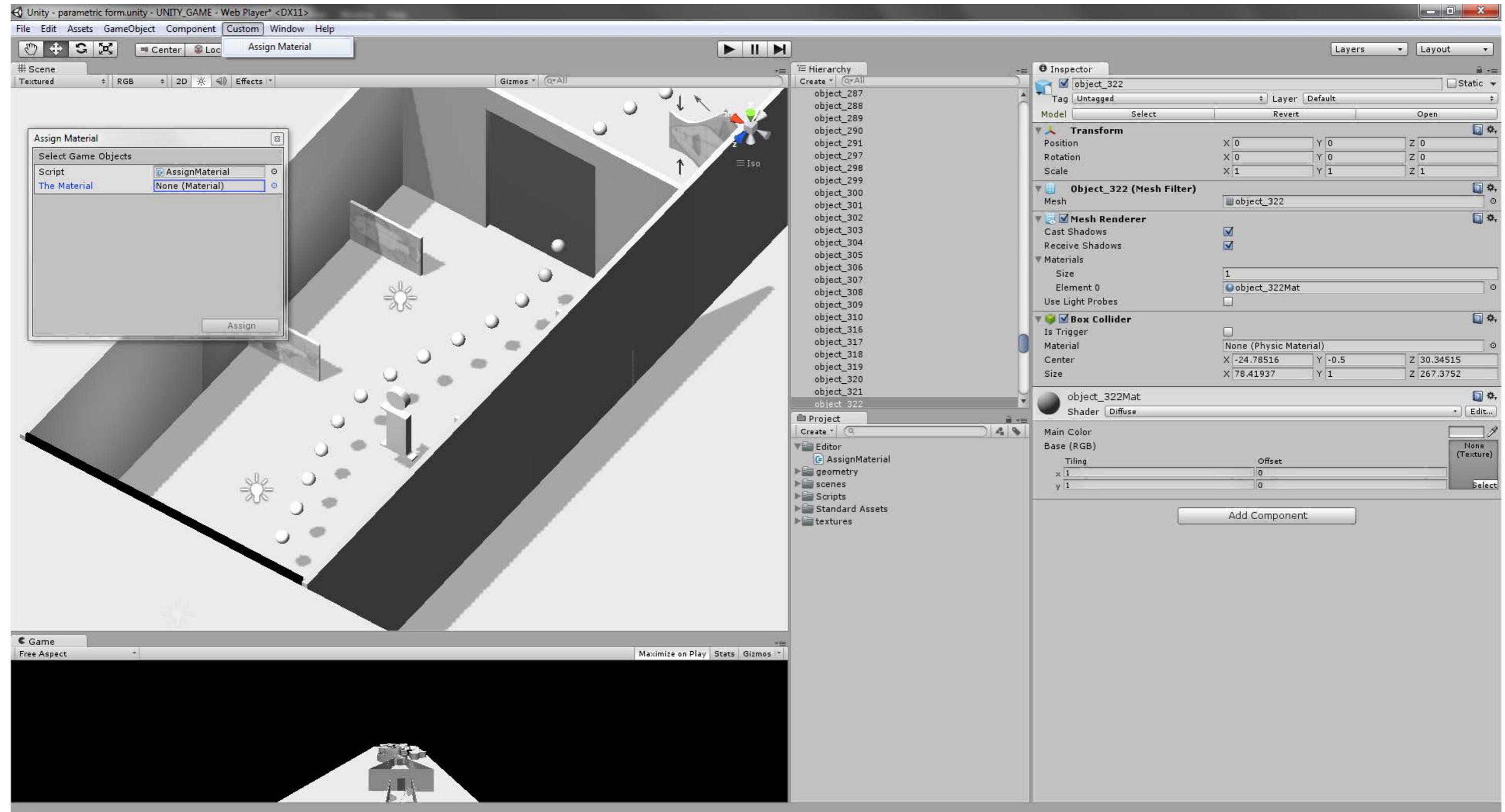
The screenshot shows the MonoDevelop-Unity IDE interface. The title bar reads "Assembly-CSharp-Editor - Editor\AssignMaterial.cs - MonoDevelop-Unity". The menu bar includes File, Edit, View, Search, Project, Build, Run, Version Control, Tools, Window, Help. The toolbar has a play button, "Debug", and a dropdown for "Default". The main window shows the "AssignMaterial.cs" file under the "Solution" tree, which contains the following C# code:

```
1 using UnityEngine;
2 using System.Collections;
3 using System;
4 using UnityEditor;
5
6 public class AssignMaterial : ScriptableWizard
7 {
8
9     public Material theMaterial;
10    String strHelp = "Select Game Objects";
11    GameObject[] gos;
12
13    void OnWizardUpdate ()
14    {
15        helpString = strHelp;
16        isValid = (theMaterial != null);
17    }
18
19    void OnWizardCreate ()
20    {
21        gos = Selection.gameObjects;
22        foreach (GameObject go in gos)
23        {
24            go.renderer.material = theMaterial;
25        }
26    }
27
28    [MenuItem ("Custom/Assign Material", false, 4)]
29    static void assignMaterial()
30    {
31        ScriptableWizard.DisplayWizard ("Assign Material", typeof(AssignMaterial), "Assign");
32    }
33 }
```

The right side of the interface includes tabs for "MonoDevelop-Unity", "Press 'Control+' to search", and toolbars for "Tools", "Properties", "Document Outline", and "Unit Tests". The bottom right corner shows "Errors" and "Tasks".

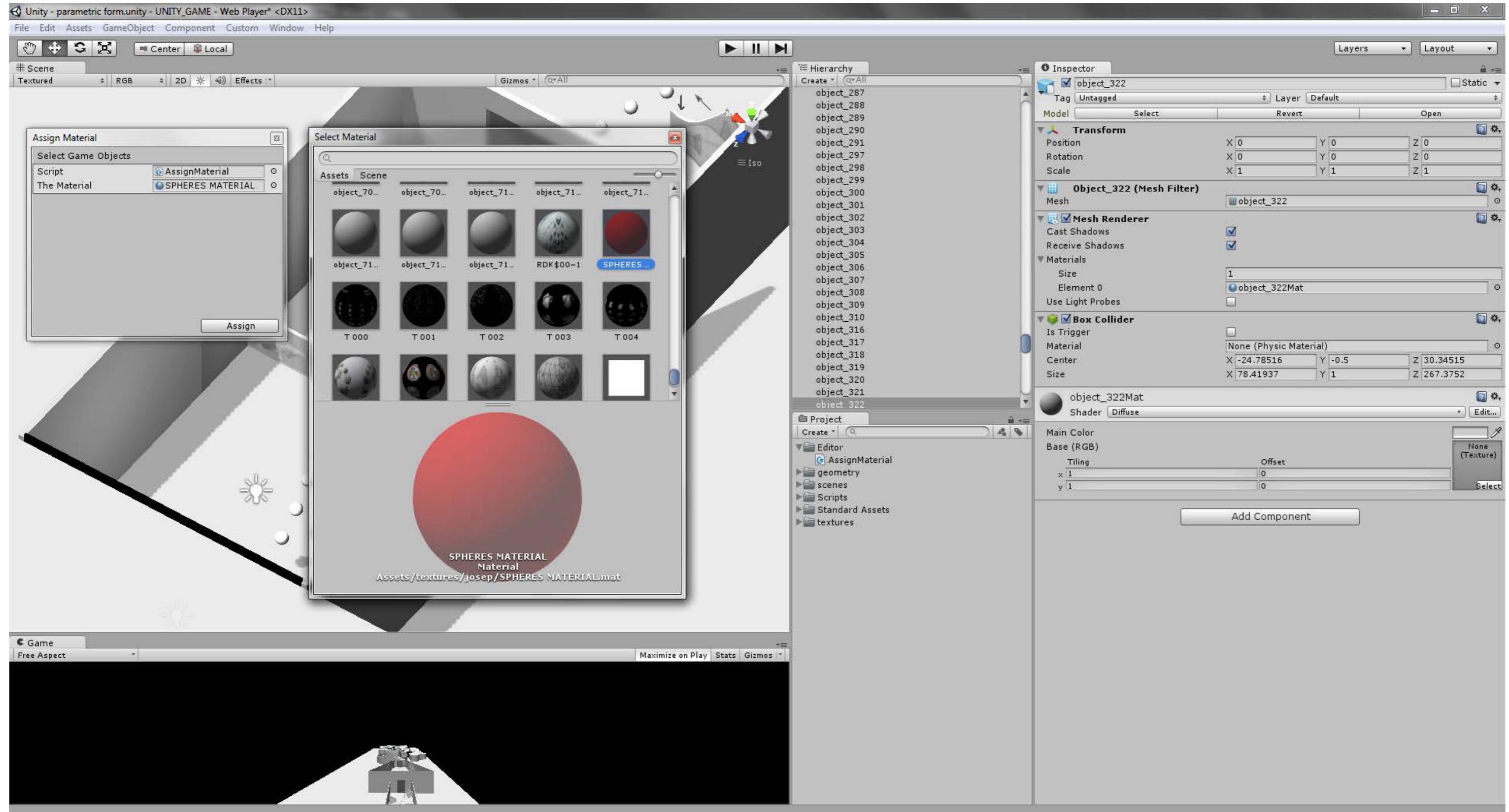
## Assigning materials to multiple objects

Running the Assign Material script from the top toolbar.

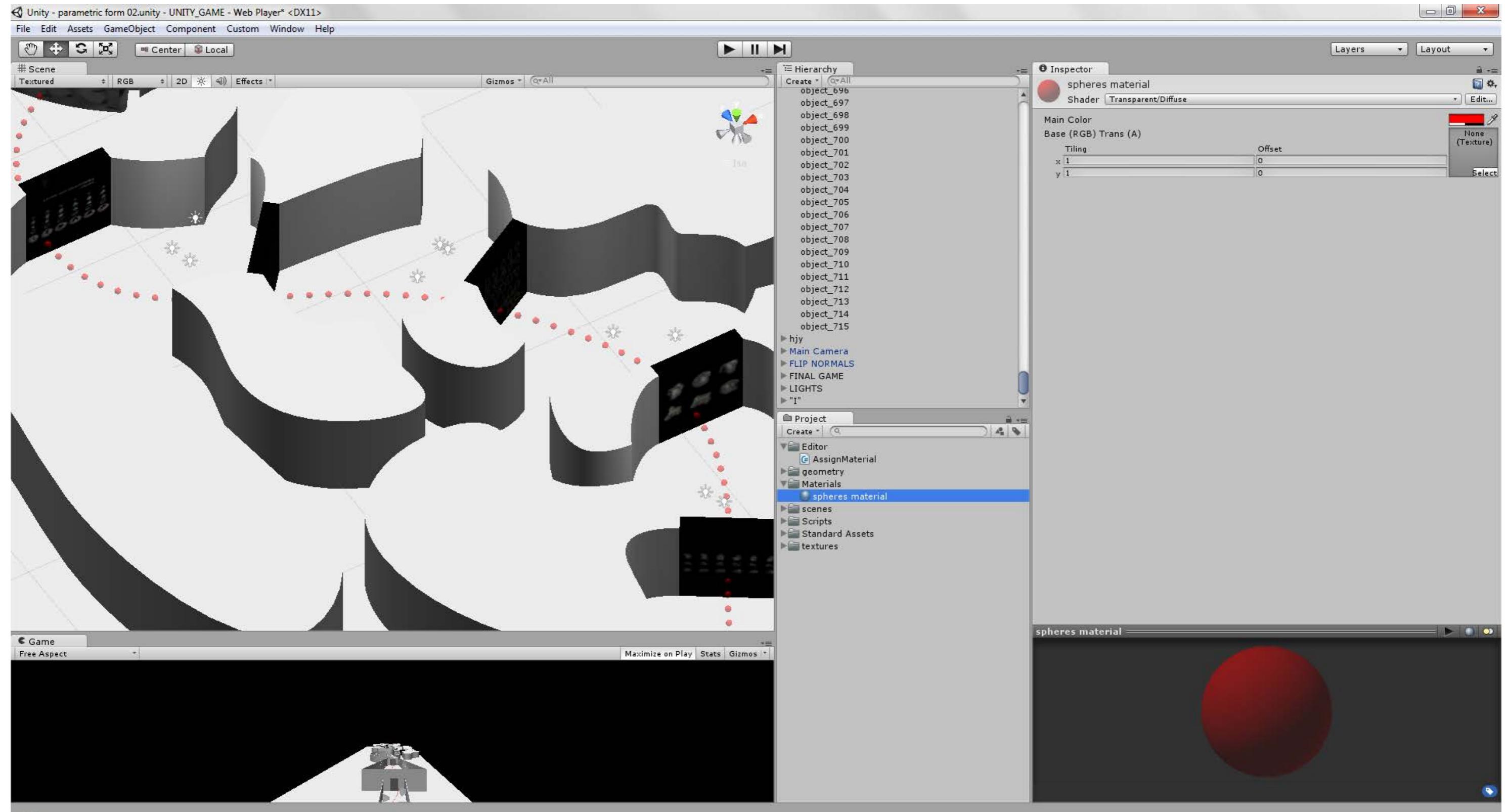


## Assigning materials to multiple objects

We select the material that we previously created (transparent diffuse red material) for the spheres



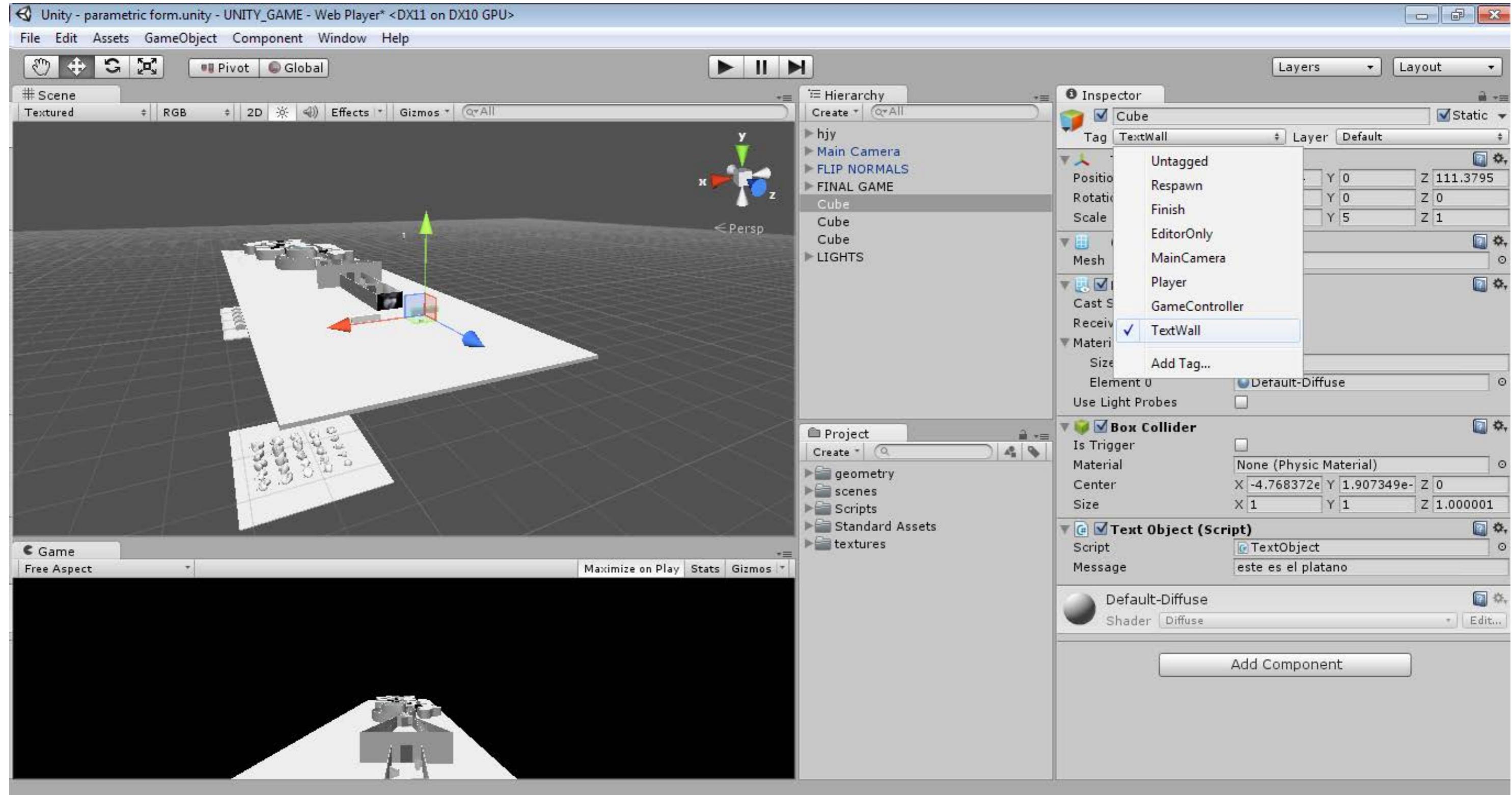
## Assigning materials to multiple objects



Assignment 3 - Gameify! (Unity 3D)  
Josep Alcover Llubiá + Nohelia Gonzalez + Jinyang Han

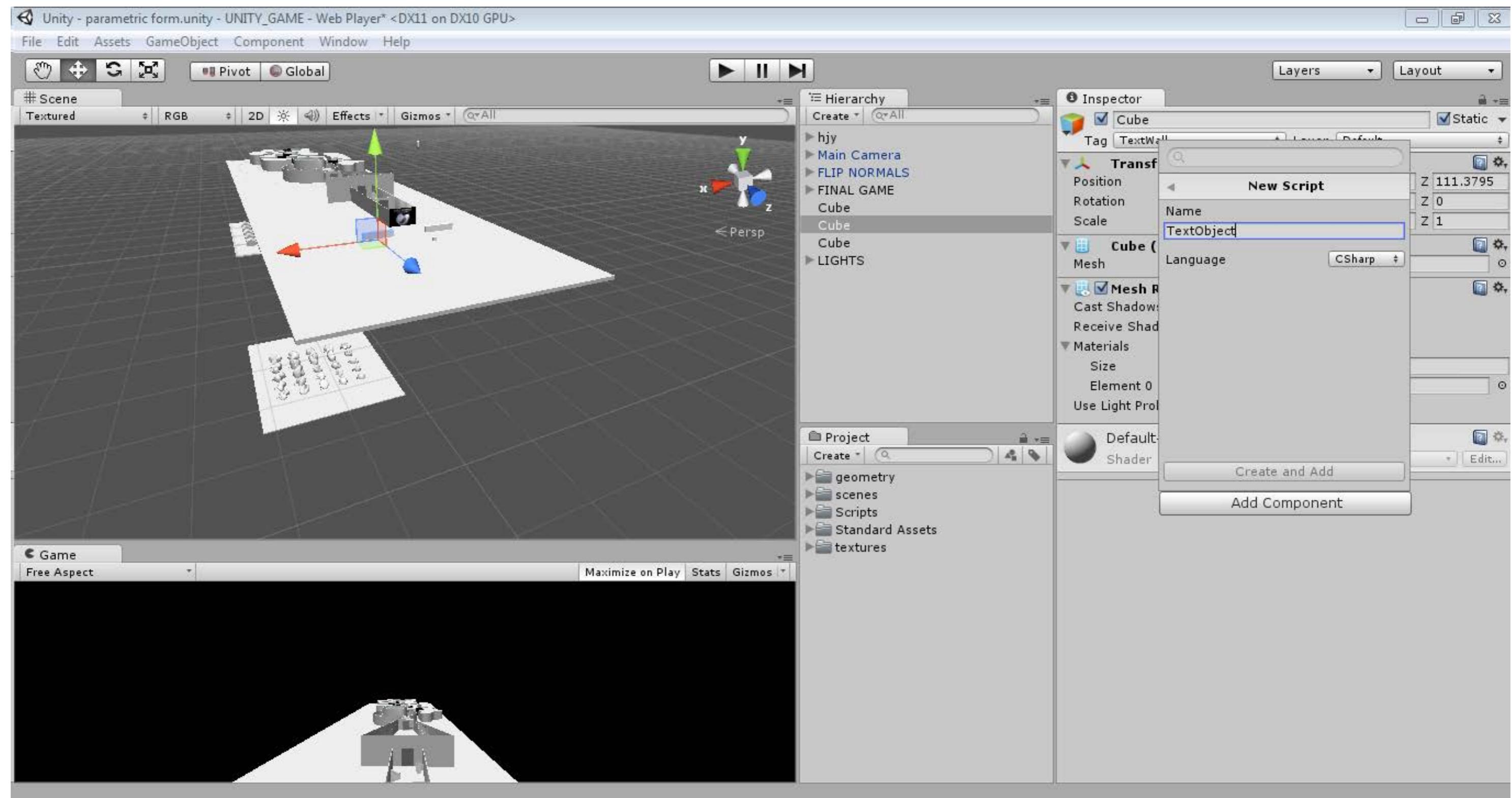
## Creating objects that display a message when the player collides with them

We create the tag "TextWall" to differentiate objects that contain information from those that don't.



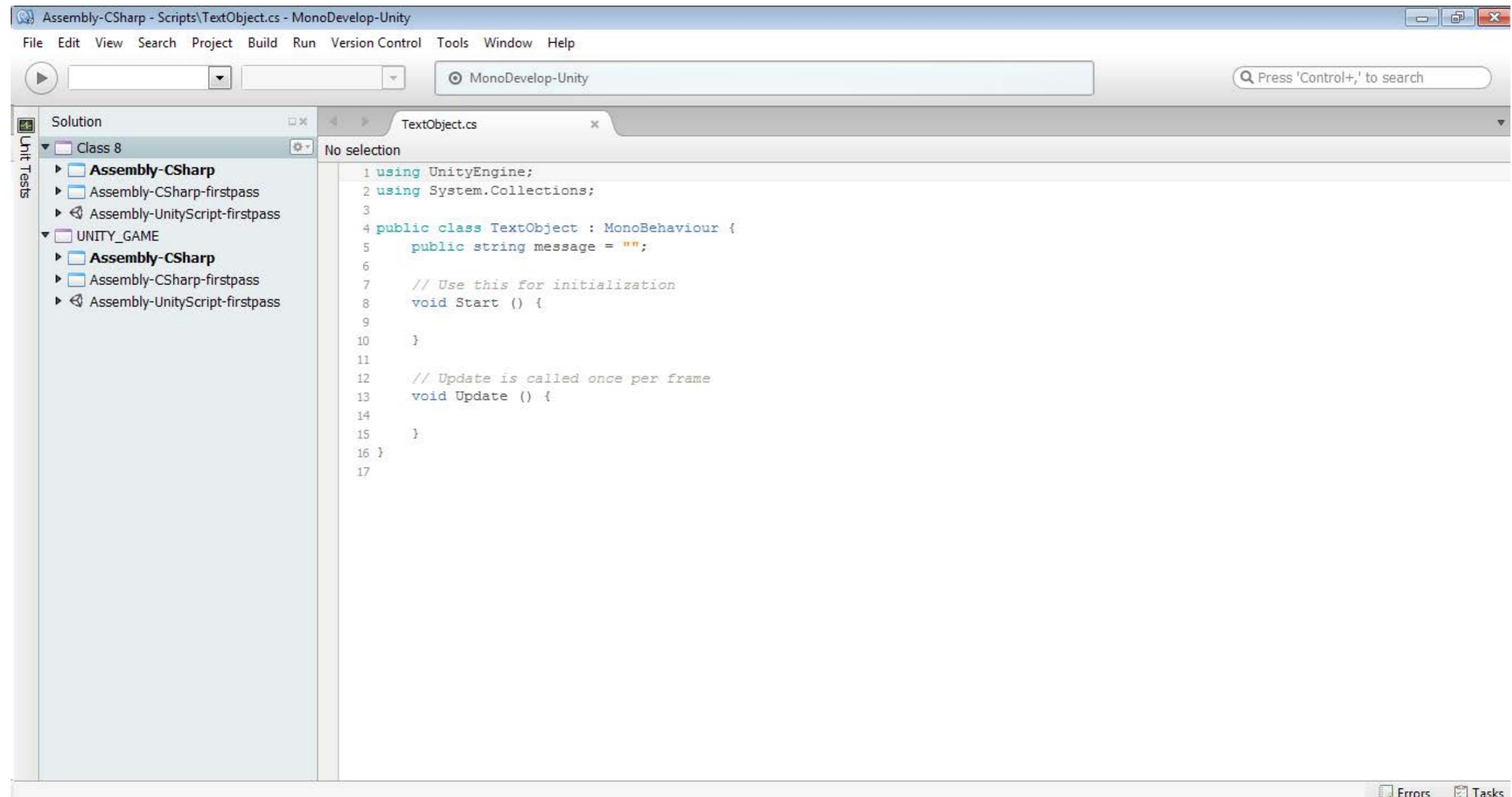
## Creating objects that display a message when the player collides with them

We create the script "TextObject" that will store the message to be displayed.



## Creating objects that display a message when the player collides with them

The script "TextObject" only contains a public message that can be accessed from the player controller. Every object has its own message.

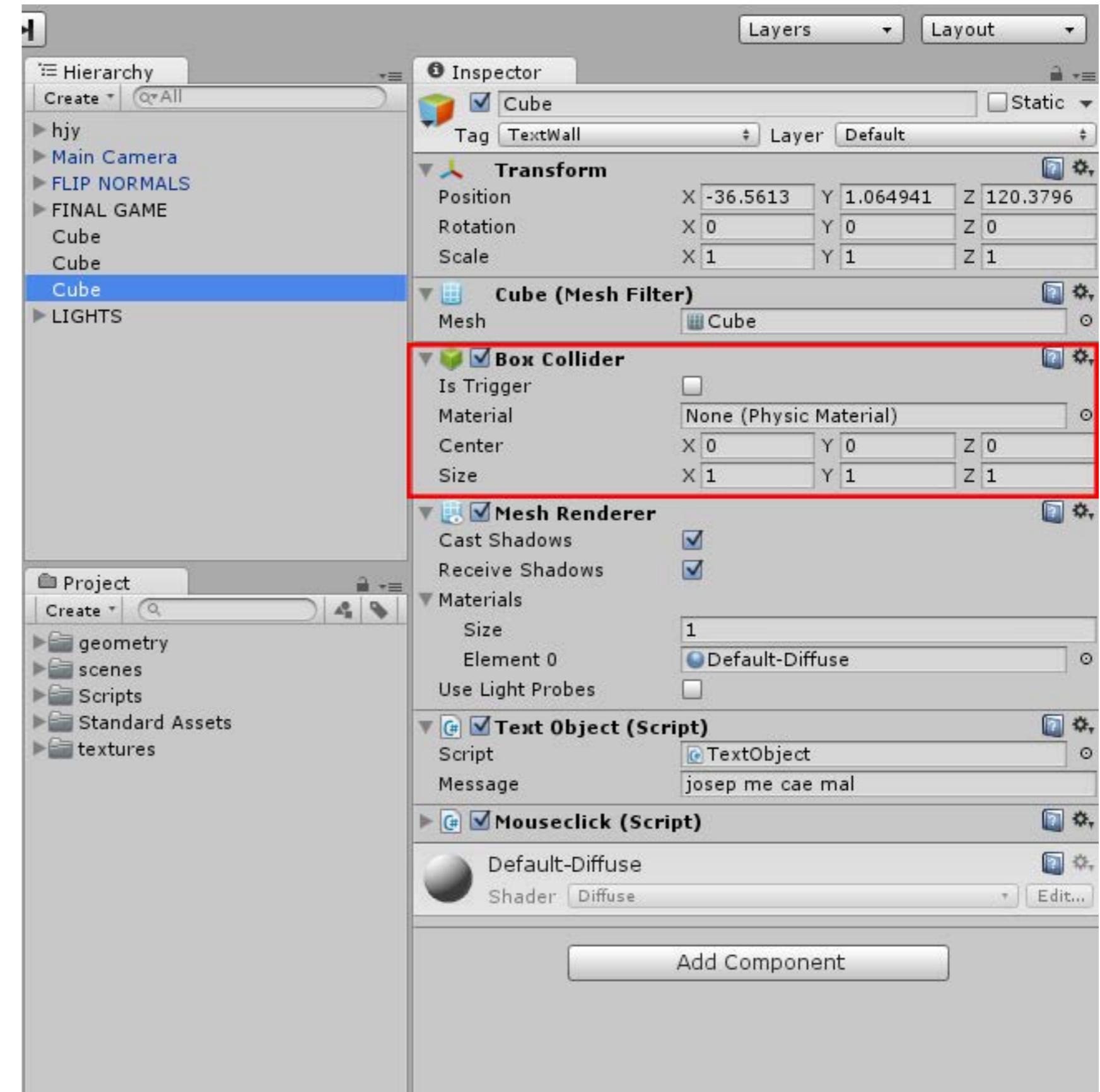


The screenshot shows the MonoDevelop-Unity IDE interface. The title bar reads "Assembly-CSharp - Scripts\TextObject.cs - MonoDevelop-Unity". The menu bar includes File, Edit, View, Search, Project, Build, Run, Version Control, Tools, Window, and Help. A toolbar with a play button and dropdown menus is visible. The status bar at the bottom right says "Press 'Control+', to search". The left sidebar shows a "Solution" tree with nodes for Class 8, Assembly-CSharp, Assembly-CSharp-firstpass, Assembly-UnityScript-firstpass, and UNITY\_GAME. The main editor window displays the "TextObject.cs" script:

```
1 using UnityEngine;
2 using System.Collections;
3
4 public class TextObject : MonoBehaviour {
5     public string message = "";
6
7     // Use this for initialization
8     void Start () {
9
10    }
11
12     // Update is called once per frame
13     void Update () {
14
15    }
16 }
17
```

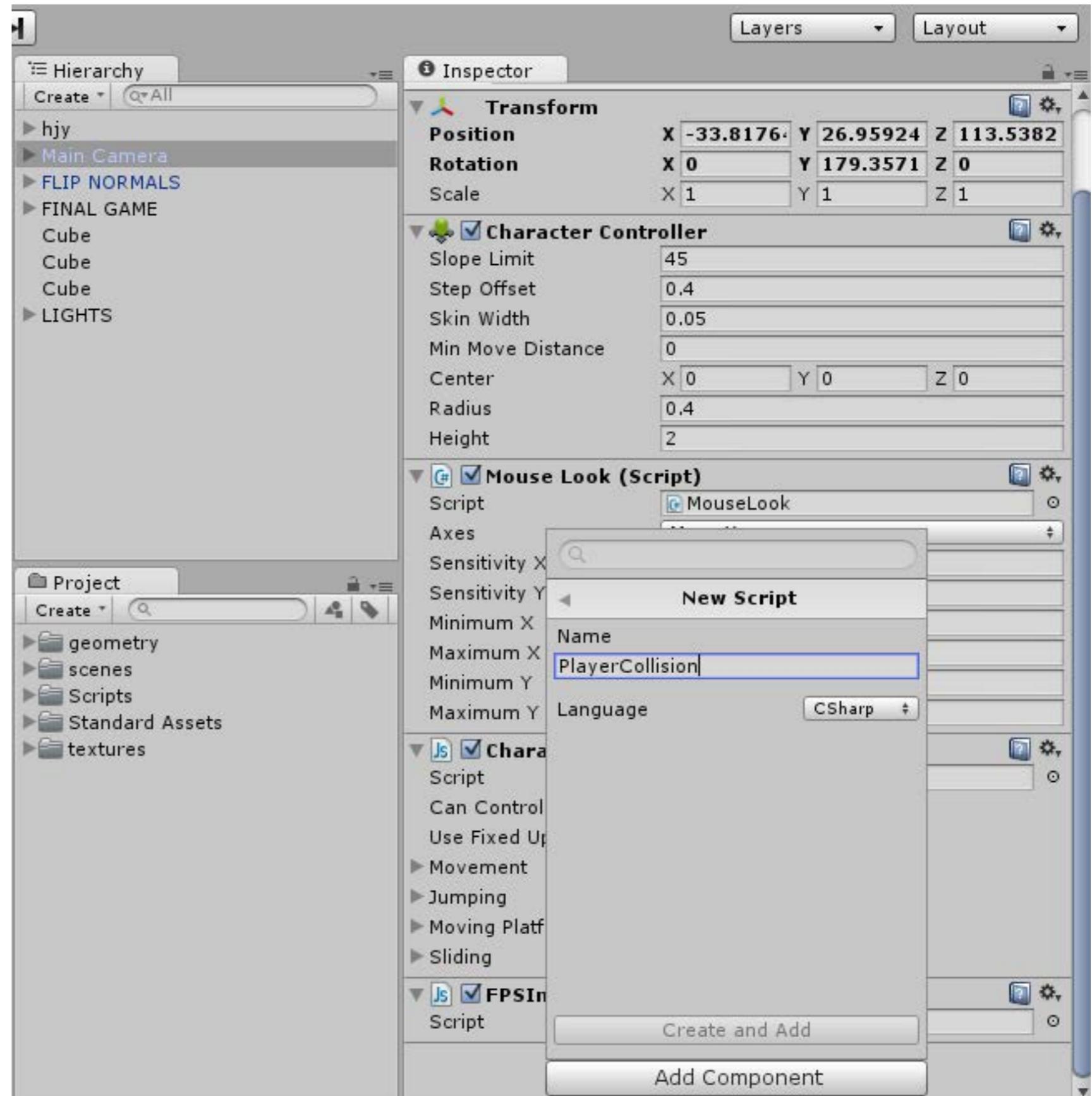
## Creating objects that display a message when the player collides with them

Every object from this class must have a proper collider. The best collider is the simplest one that fits the object. For example mesh collider, which is heavier, is useful for highly irregular objects, for simple geometric forms a geometric collider can be used.



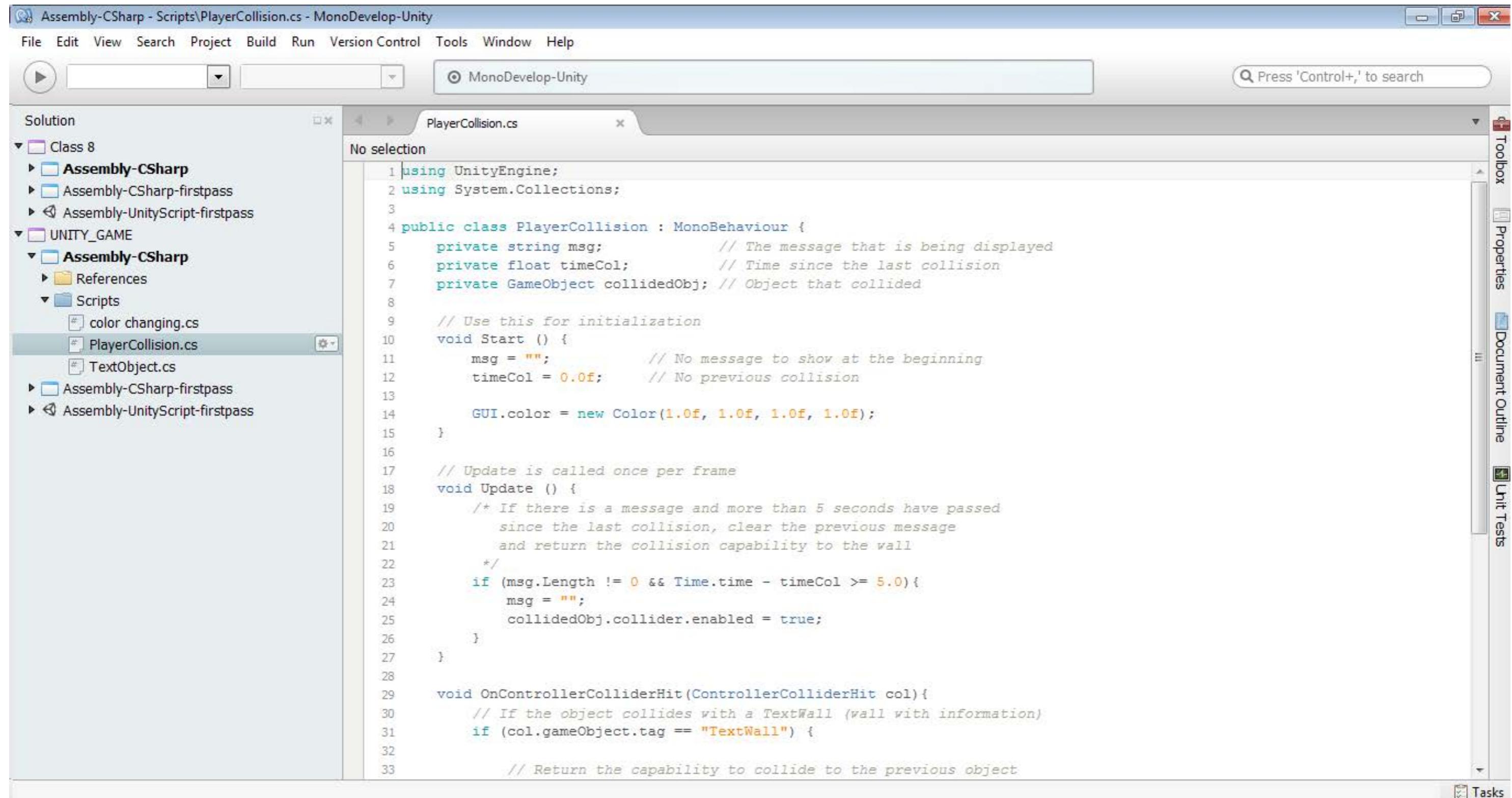
## Creating objects that display a message when the player collides with them

We create the script "PlayerCollision" that will handle the collision with the objects that contain information.



## Creating objects that display a message when the player collides with them

"PlayerCollision" script.



The screenshot shows the MonoDevelop-Unity IDE interface. The title bar reads "Assembly-CSharp - Scripts\PlayerCollision.cs - MonoDevelop-Unity". The menu bar includes File, Edit, View, Search, Project, Build, Run, Version Control, Tools, Window, and Help. A toolbar with various icons is visible above the main window. The left sidebar is the Solution Explorer, showing a project structure with nodes like Class 8, Assembly-CSharp, Assembly-CSharp-firstpass, Assembly-UnityScript-firstpass, UNITY\_GAME, Assembly-CSharp, References, Scripts, color changing.cs, PlayerCollision.cs (which is selected), TextObject.cs, Assembly-CSharp-firstpass, and Assembly-UnityScript-firstpass. The main editor area displays the C# code for the PlayerCollision.cs script:

```
1 using UnityEngine;
2 using System.Collections;
3
4 public class PlayerCollision : MonoBehaviour {
5     private string msg; // The message that is being displayed
6     private float timeCol; // Time since the last collision
7     private GameObject collidedObj; // Object that collided
8
9     // Use this for initialization
10    void Start () {
11        msg = ""; // No message to show at the beginning
12        timeCol = 0.0f; // No previous collision
13
14        GUI.color = new Color(1.0f, 1.0f, 1.0f, 1.0f);
15    }
16
17    // Update is called once per frame
18    void Update () {
19        /* If there is a message and more than 5 seconds have passed
         * since the last collision, clear the previous message
         * and return the collision capability to the wall
         */
20        if (msg.Length != 0 && Time.time - timeCol >= 5.0){
21            msg = "";
22            collidedObj.collider.enabled = true;
23        }
24    }
25
26    void OnControllerColliderHit(ControllerColliderHit col){
27        // If the object collides with a TextWall (wall with information)
28        if (col.gameObject.tag == "TextWall") {
29            // Return the capability to collide to the previous object
30        }
31    }
32
33}
```

## Creating objects that display a message when the player collides with them

Three variables are needed: the message, the time since previous collision (to make the message disappear after a established period of time) and the object that collides with the player.

```
1 using UnityEngine;
2 using System.Collections;
3
4 public class PlayerCollision : MonoBehaviour {
5     private string msg;           // The message that is being displayed
6     private float timeCol;        // Time since the last collision
7     private GameObject collidedObj; // Object that collided
8
9     // Use this for initialization
10    void Start () {
11        msg = "";                // No message to show at the beginning
12        timeCol = 0.0f;           // No previous collision
13
14        GUI.color = new Color(1.0f, 1.0f, 1.0f, 1.0f);
15    }
}
```

At the beginning there is nothing to show, so the message should be empty and the time 0.  
GUI.color changes the default color of the font to be fully opaque

```
5     private string msg;           // The message that is being displayed
6     private float timeCol;        // Time since the last collision
7     private GameObject collidedObj; // Object that collided
8
9     // Use this for initialization
10    void Start () {
11        msg = "";                // No message to show at the beginning
12        timeCol = 0.0f;           // No previous collision
13
14        GUI.color = new Color(1.0f, 1.0f, 1.0f, 1.0f);
15    }
16
17    // Update is called once per frame
18    void Update () {
19        /* If there is a message and more than 5 seconds have passed
          since the last collision. clear the previous message
20    }
```

## Creating objects that display a message when the player collides with them

If we are displaying a message and more than 5 seconds have passed since the last collision, the message is cleared and the other object's collider are activated again.  
The collider is deactivated when the player hits the object in order to let the player pass through the object.

```
14     GUI.color = new Color(1.0f, 1.0f, 1.0f, 1.0f);
15 }
16
17 // Update is called once per frame
18 void Update () {
19     /* If there is a message and more than 5 seconds have passed
20         since the last collision, clear the previous message
21         and return the collision capability to the wall
22     */
23     if (msg.Length != 0 && Time.time - timeCol >= 5.0){
24         msg = "";
25         collidedObj.collider.enabled = true;
26     }
27 }
28
29 void OnControllerColliderHit(ControllerColliderHit col){
```

## Creating objects that display a message when the player collides with them

This is the function that controls the collision between the player and other objects. It reacts when the other object has the tag "TexWall". This "if" condition ensures that the following code is applied only to objects that store a message.

```
27    }
28
29    void OnControllerColliderHit(ControllerColliderHit col){
30        // If the object collides with a TextWall (wall with information)
31        if (col.gameObject.tag == "TextWall") {
32
32            // Return the capability to collide to the previous object
34            if ( msg.Length != 0 ){
35                collidedObj.collider.enabled = true;
36            }
37
38            // Update the information
39            var box = col.gameObject;
40            msg = box.GetComponent<TextObject> ().message; // Retrieve the message
41            collidedObj = col.gameObject; // Update the colliding object
42            collidedObj.collider.enabled = false; // Disable the other object's collider
43            timeCol = Time.time; // Update the time
44
45        }
46    }
47
48    void OnGUI () {
```

## Creating objects that display a message when the player collides with them

The message contained in the object is retrieved and stored, the object's collider is disabled and the time since the last collision is updated.

```
34         if ( msg.Length != 0 ){
35             collidedObj.collider.enabled = true;
36         }
37
38         // Update the information
39         var box = col.gameObject;
40         msg = box.GetComponent<TextObject> ().message;    // Retrieve the message
41         collidedObj = col.gameObject;                      // Update the colliding object
42         collidedObj.collider.enabled = false;              // Disable the other object's collider
43         timeCol = Time.time;                             // Update the time
44
45     }
46 }
47
48 void OnGUI() {
```

If the message of another object is being displayed, the collider is enabled again to allow future collisions.

```
27     }
28
29     void OnControllerColliderHit(ControllerColliderHit col){
30         // If the object collides with a TextWall (wall with information)
31         if (col.gameObject.tag == "TextWall") {
32
33             // Return the capability to collide to the previous object
34             if ( msg.Length != 0 ){
35                 collidedObj.collider.enabled = true;
36             }
37
38             // Update the information
39             var box = col.gameObject;
40             msg = box.GetComponent<TextObject> ().message;    // Retrieve the message
41             collidedObj = col.gameObject;                      // Update the colliding object
```

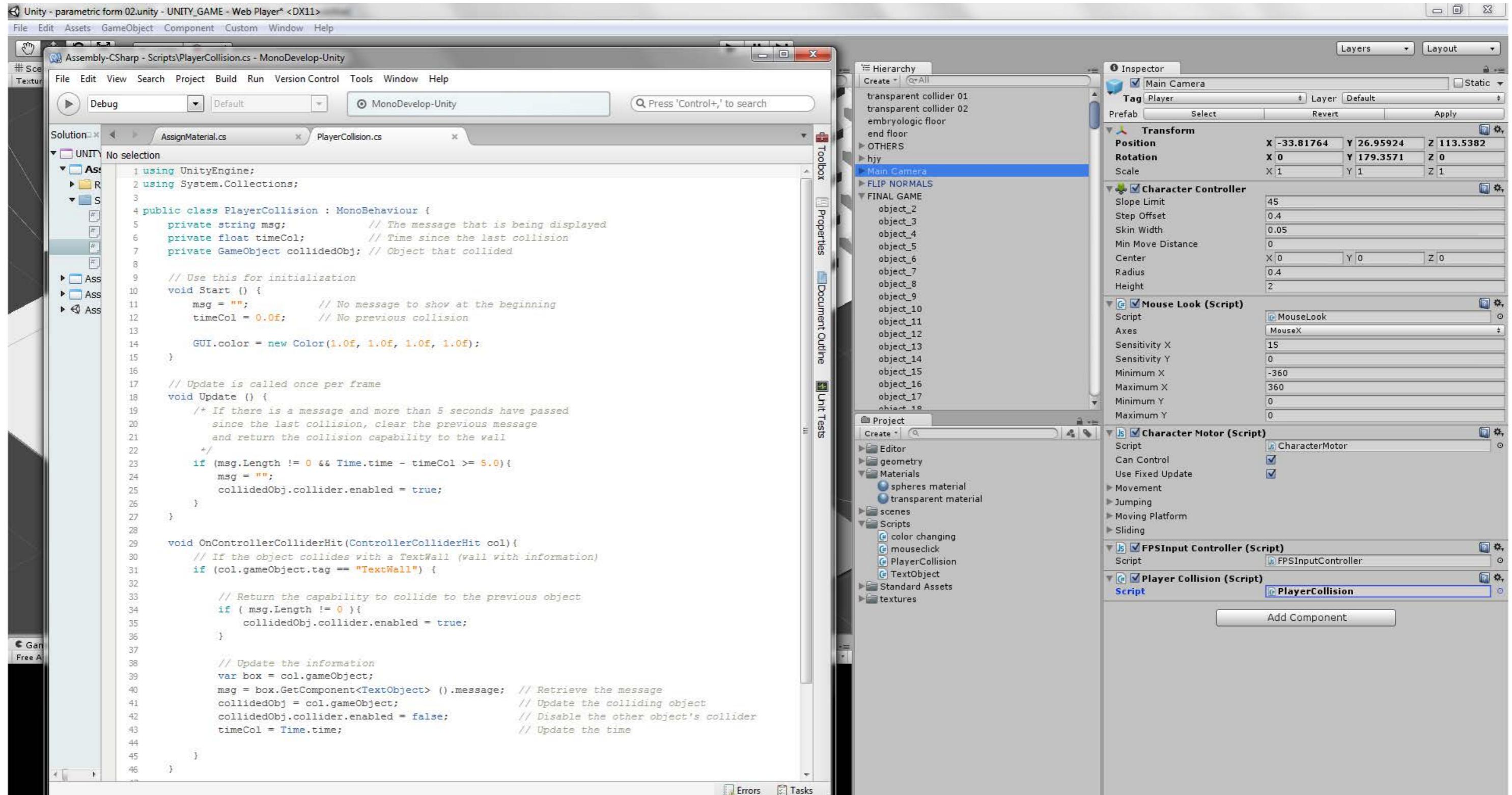
## Creating objects that display a message when the player collides with them

The message is displayed in the upper left of the screen (starts in 20,20 and ends in 1/4 of the screen width horizontally and 1/2 of the screen height vertically.

```
39     var box = col.gameObject;
40     msg = box.GetComponent<TextObject> ().message; // Retrieve the message
41     collidedObj = col.gameObject; // Update the colliding object
42     collidedObj.collider.enabled = false; // Disable the other object's collider
43     timeCol = Time.time; // Update the time
44
45 }
46 }
47
48 void OnGUI() {
49     if (msg.Length != 0) {
50         GUI.Box (new Rect (20, 20, Screen.width / 4, Screen.height / 2), msg);
51     }
52 }
53 }
54 }
```

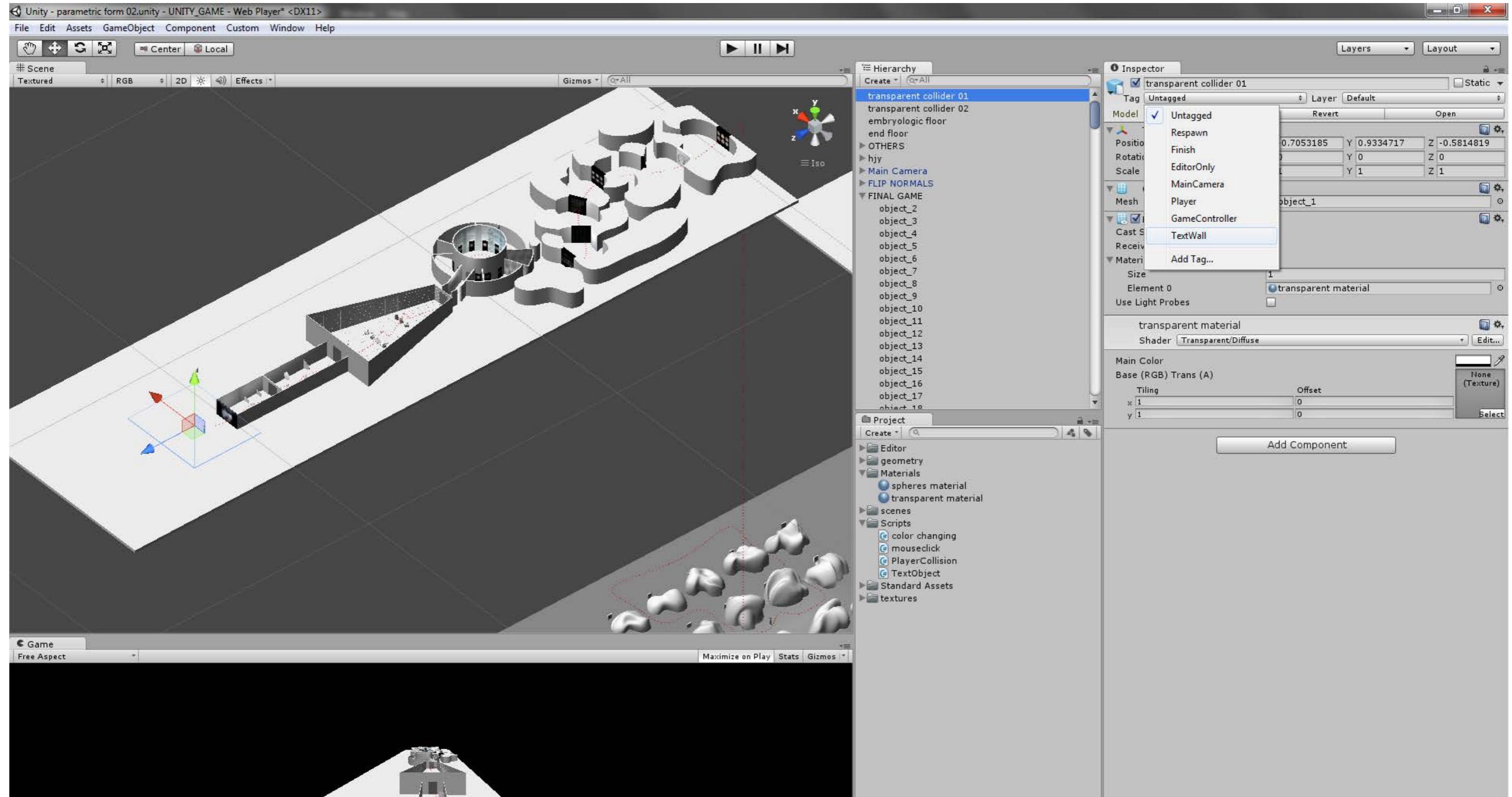
## Creating objects that display a message when the player collides with them

We apply the PlayerCollision script to the camera.



## Creating objects that display a message when the player collides with them

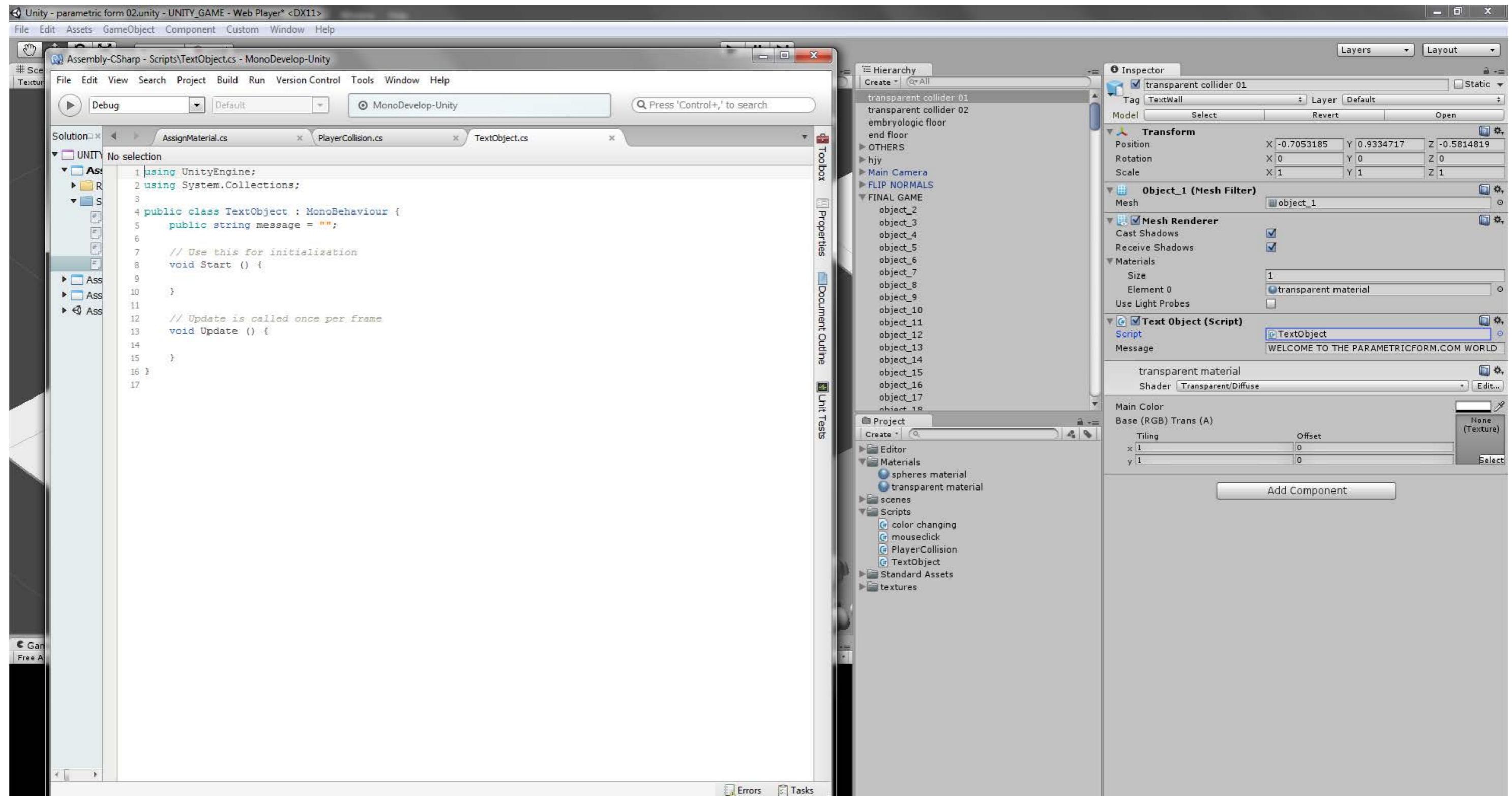
We tag the object that we want to contain a message with the 'Text Wall' tag.



## Creating objects that display a message when the player collides with them

We apply the "TextObject" script to the chosen objects and write the message in the Inspector.

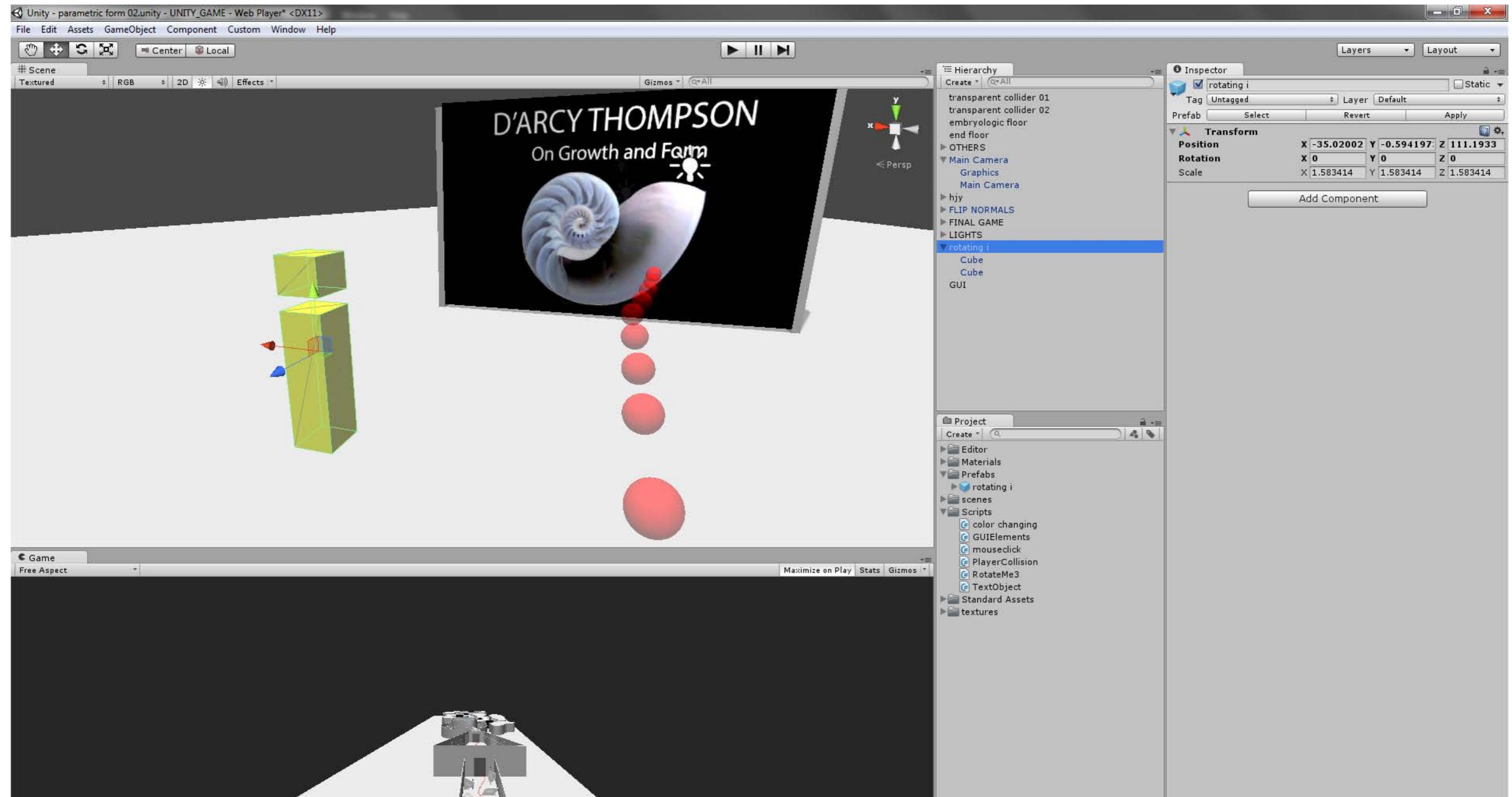
We don't need to open the script because the message is a public variable.



## Creating the information points

The information points are 3D floating rotating "i" with 3 features.

1. They rotate around and the rotation speed can be controlled with the slider.
2. They display a message when the player collides with them.
3. Their color changes when we hover the mouse over them.



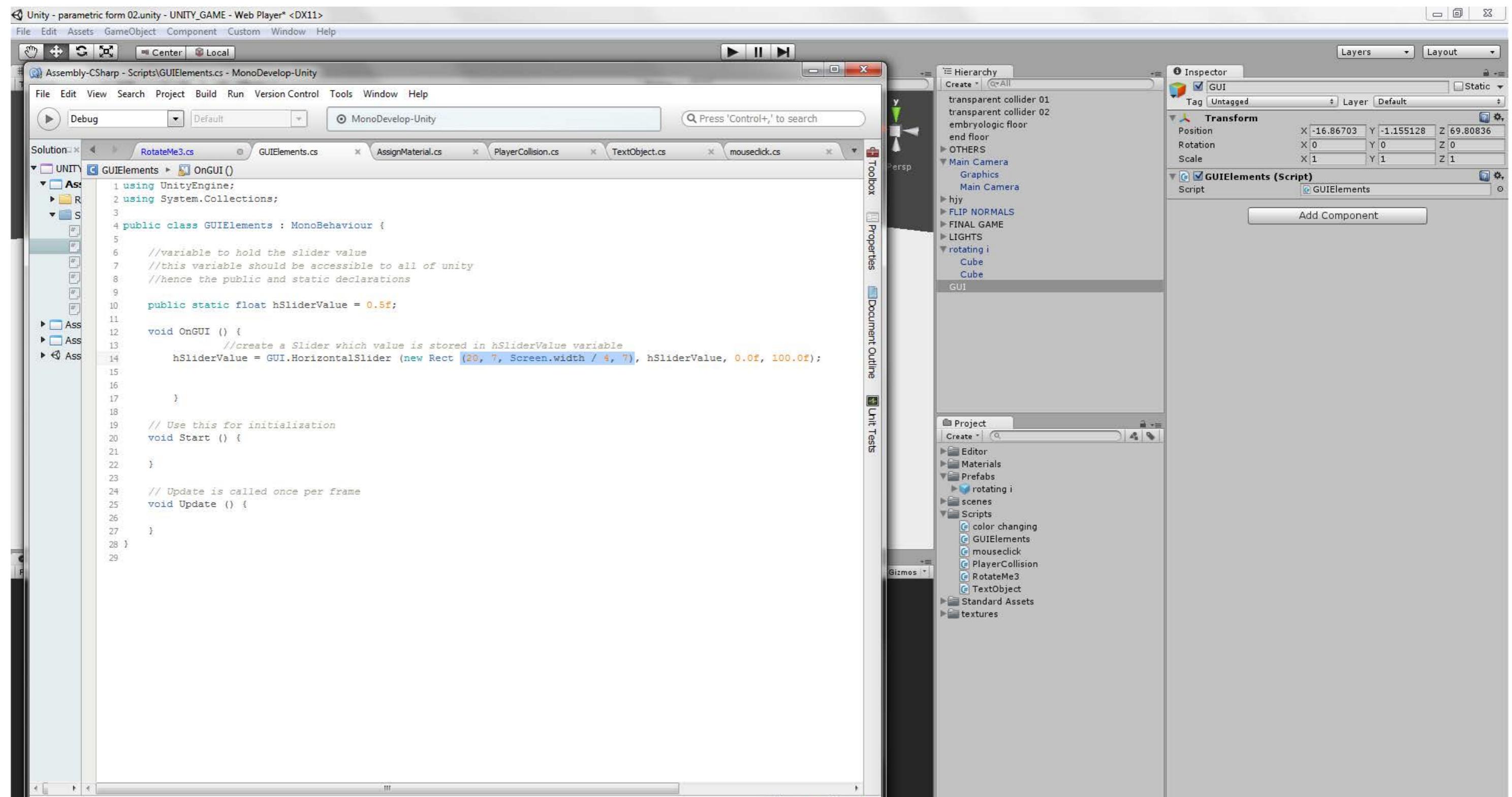
## Creating the information points

### ROTATION

We create the GUIElements script and apply it to an empty Game Object.

The script creates a slider and stores the value of the slider in a public static variable.

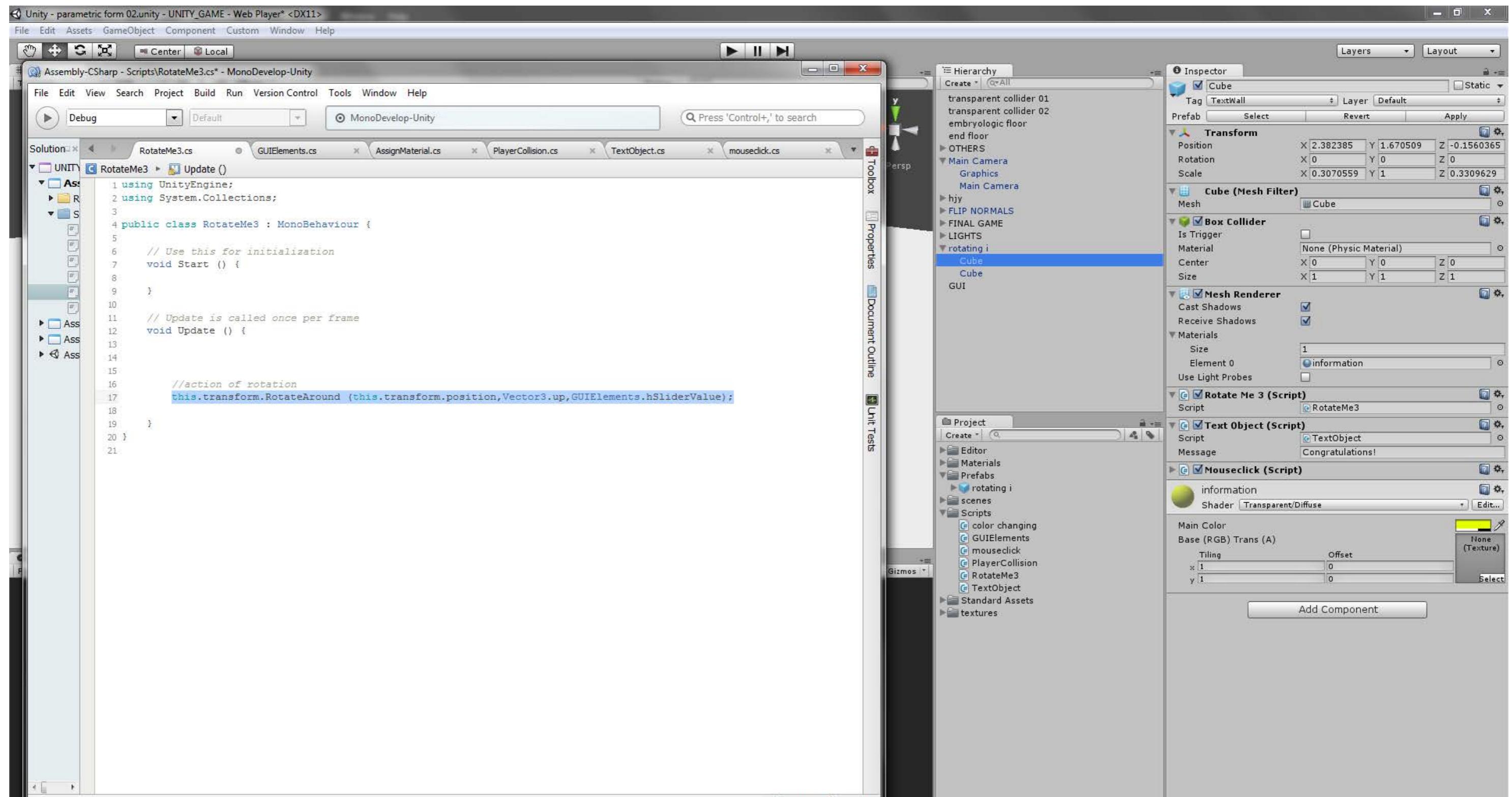
We match the size of the slider with the size of the text box that displays the messages.



## Creating the information points

### ROTATION

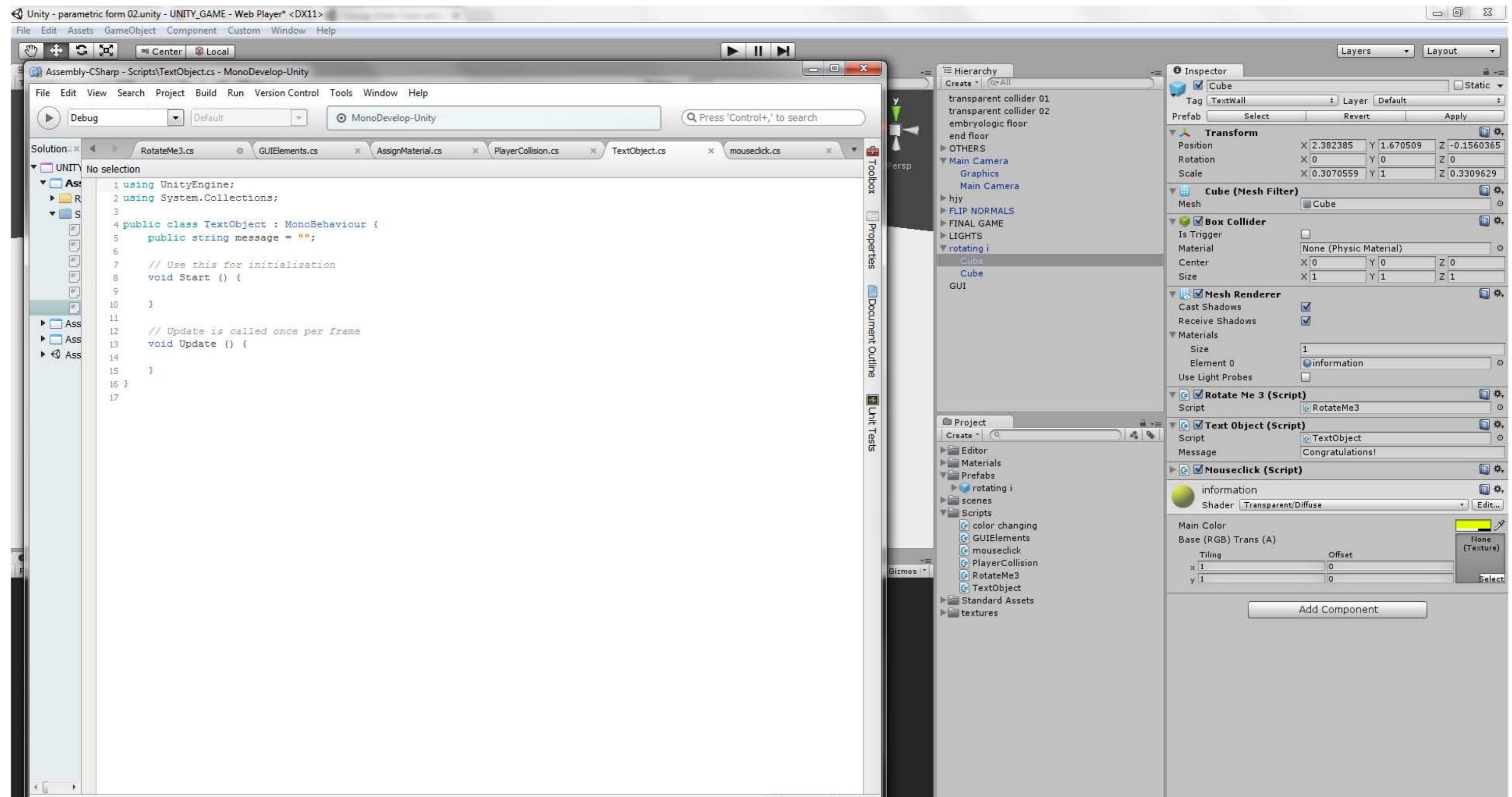
We create the RotateMe script and apply it to the 'i' object.  
The rotation speed is controlled by the value of the slider.



## Creating the information points

### DISPLAY MESSAGE

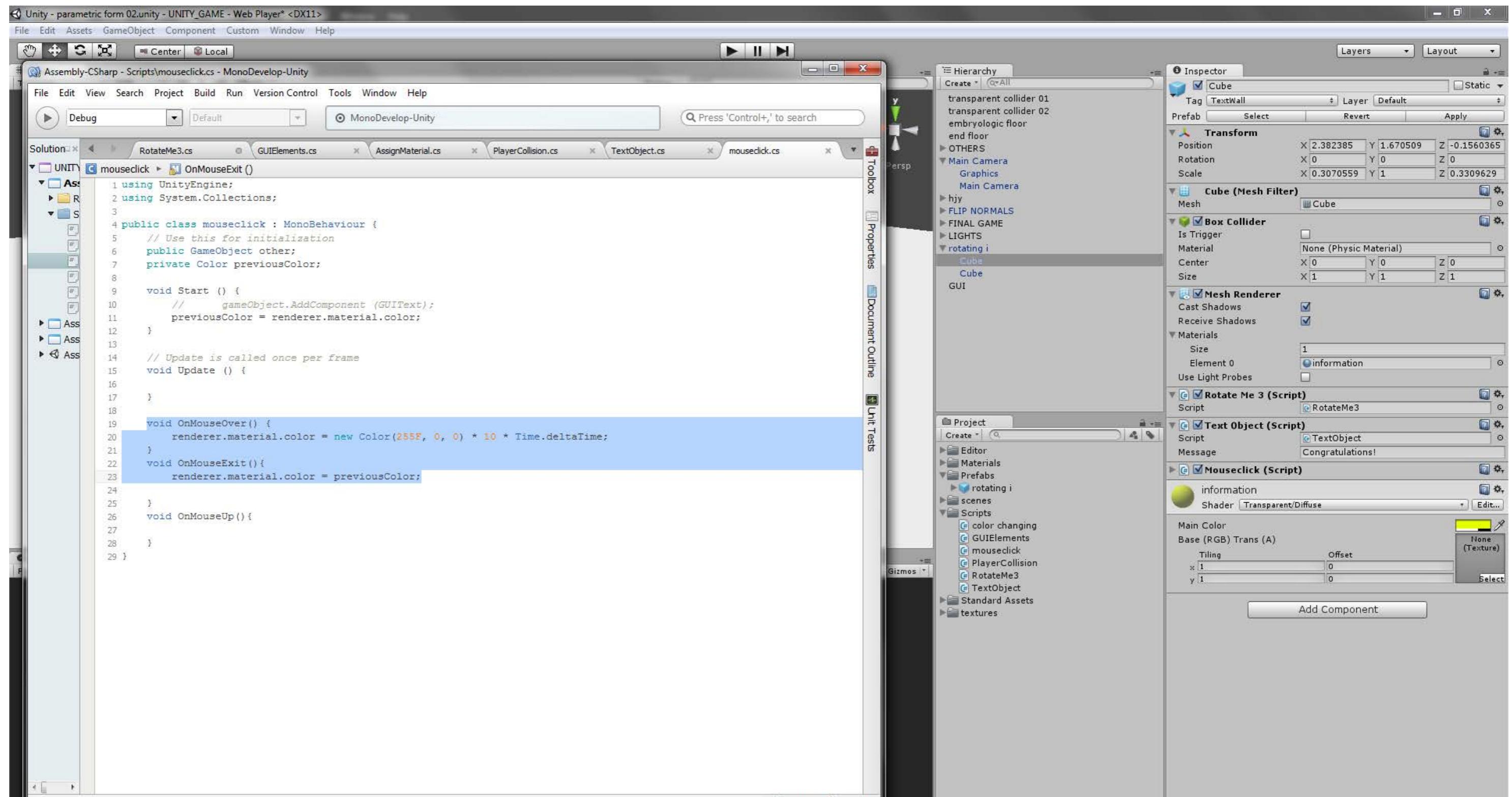
We apply the TextObject script to the "I".



# Creating the information points

## CHANGING COLOR

We apply the Mouseclick script to the "i" object.



## Creating the information points

### CHANGING COLOR

Two actions are considered in this script: mouse over and mouse exit.  
When the mouse is on an object, it changes its color.

```
18     void OnMouseOver() {
19         renderer.material.color -= new Color(0.1F, 0, 0) * 10 * Time.deltaTime;
20
21     void OnMouseExit() {
22         renderer.material.color = previousColor;
```

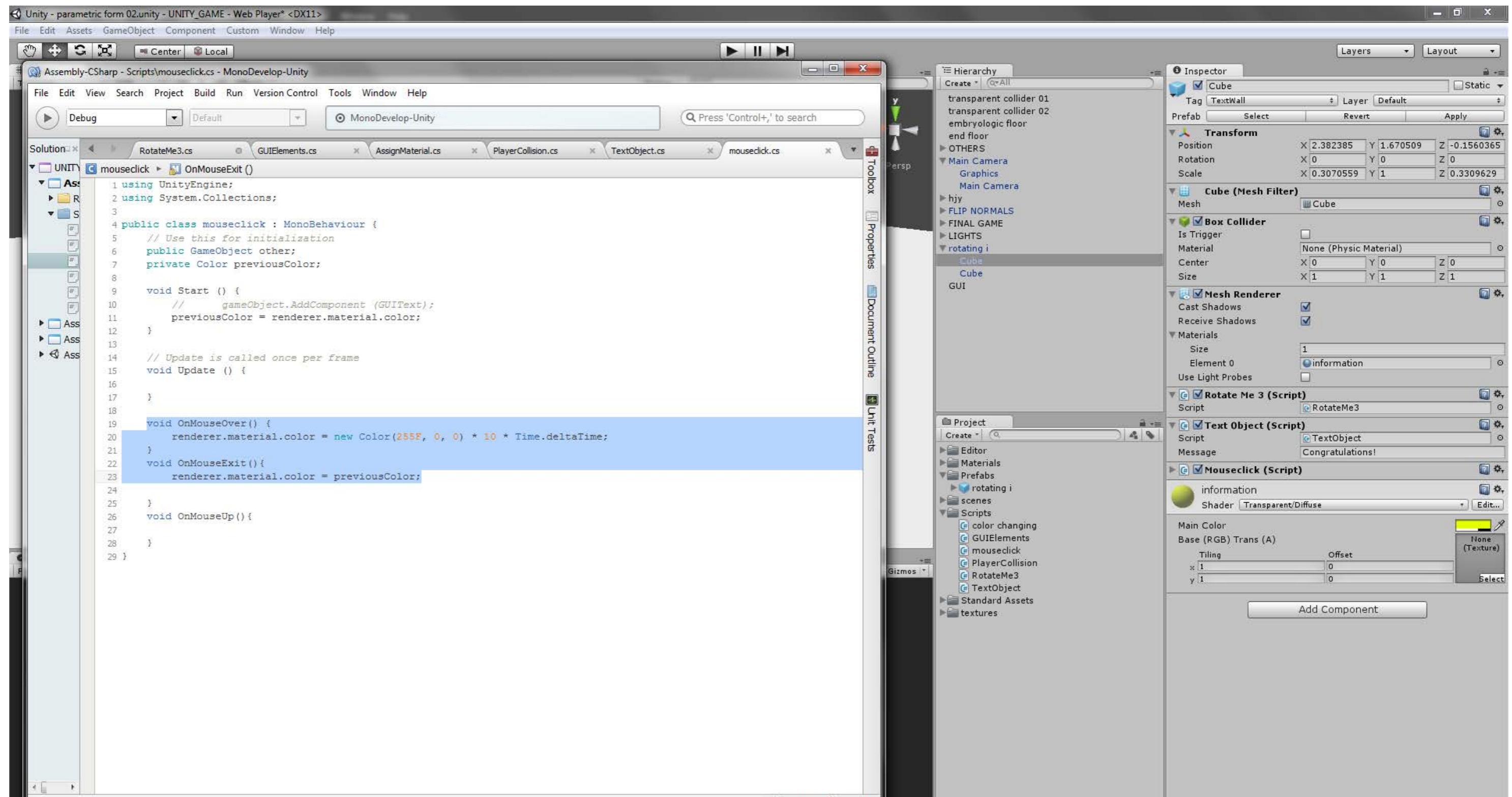
When the mouse leaves the object, the object changes to its original color.  
To achieve this we saved its previous color before.

```
7     private Color previousColor;
8
9     void Start () {
10
11     previousColor = renderer.material.color;
12 }
```

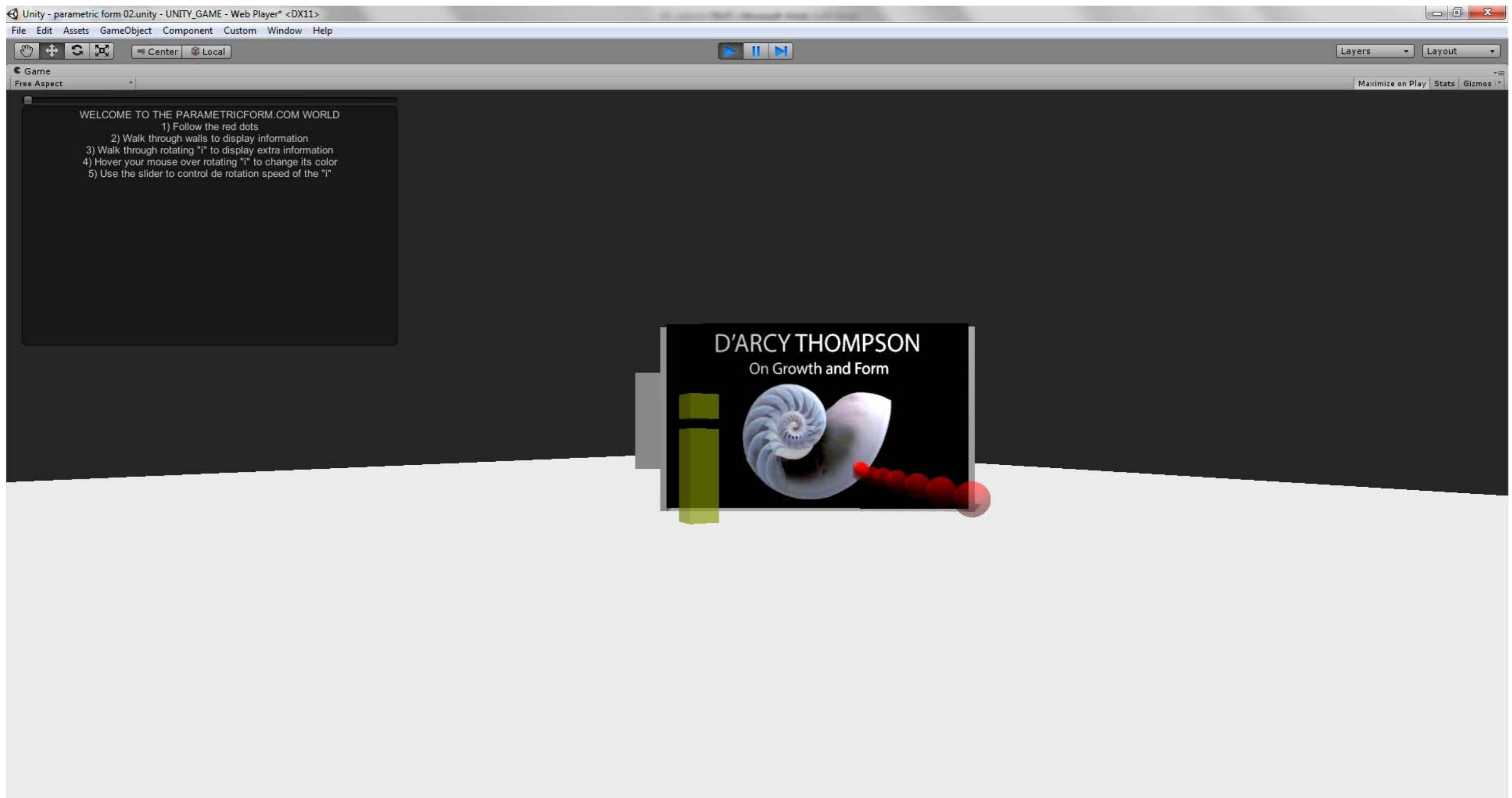
# Creating the information points

## CHANGING COLOR

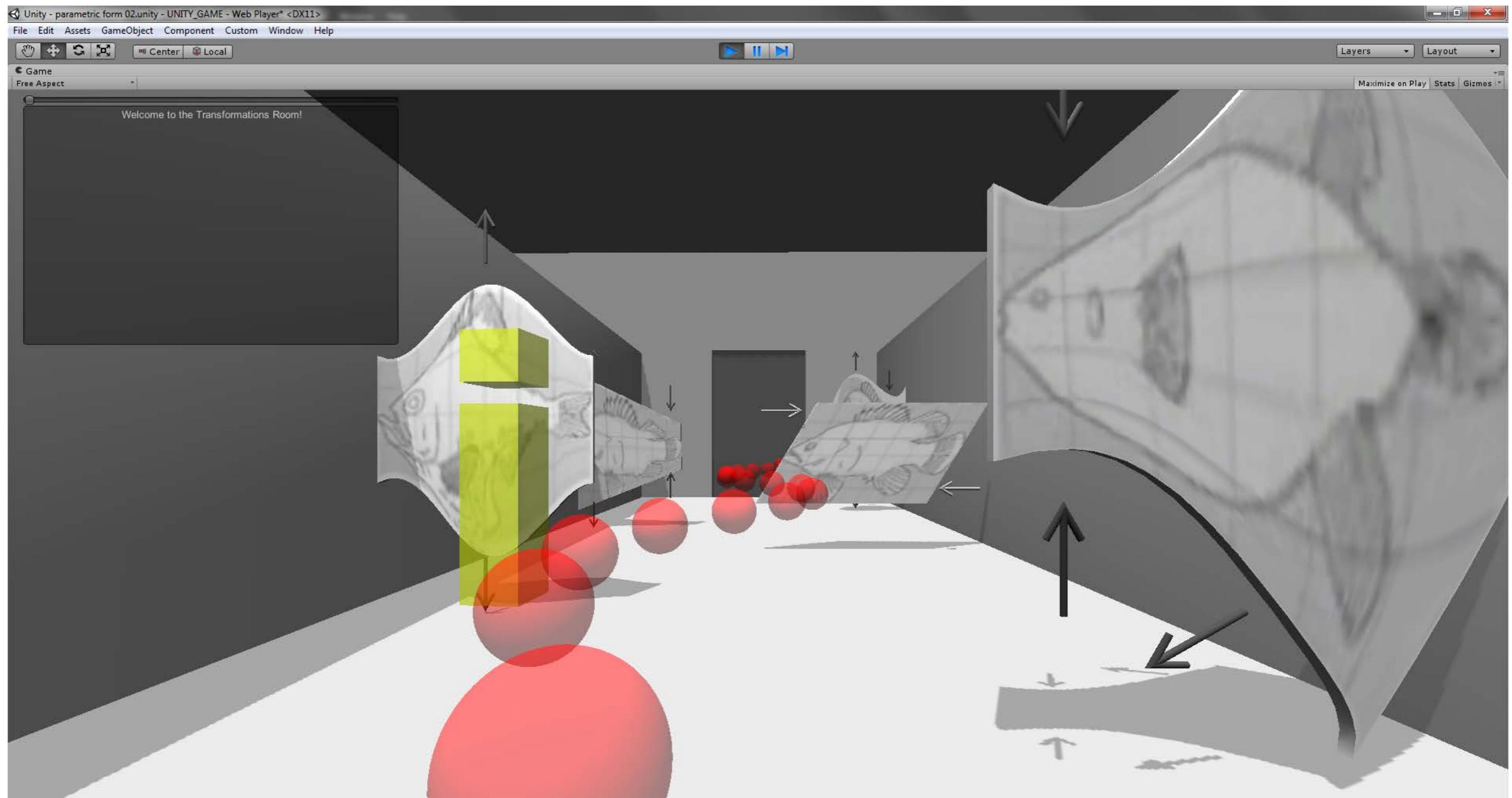
We apply the Mouseclick script to the "i" object.



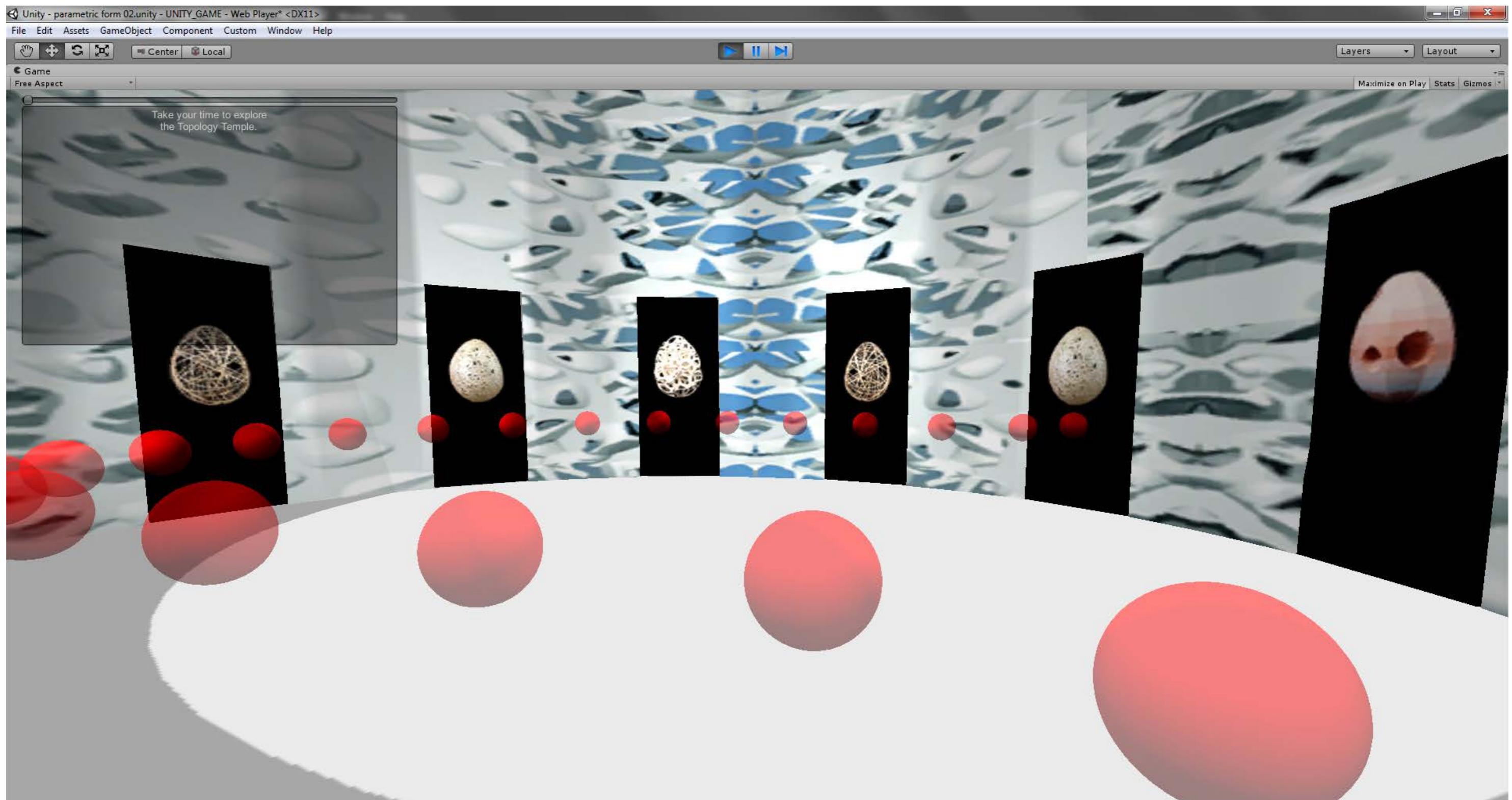
## Playing the game in Unity



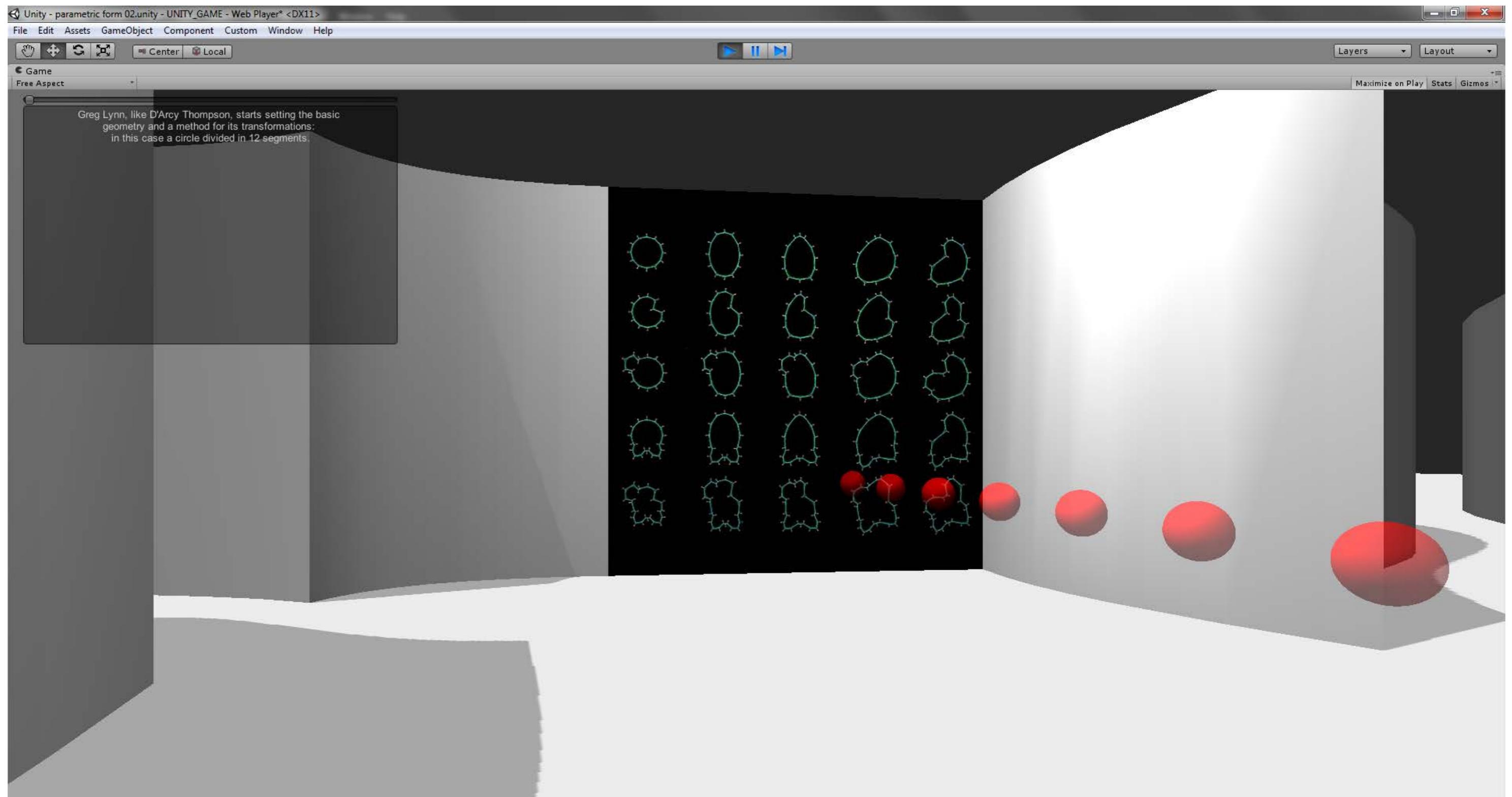
# Playing the game in Unity



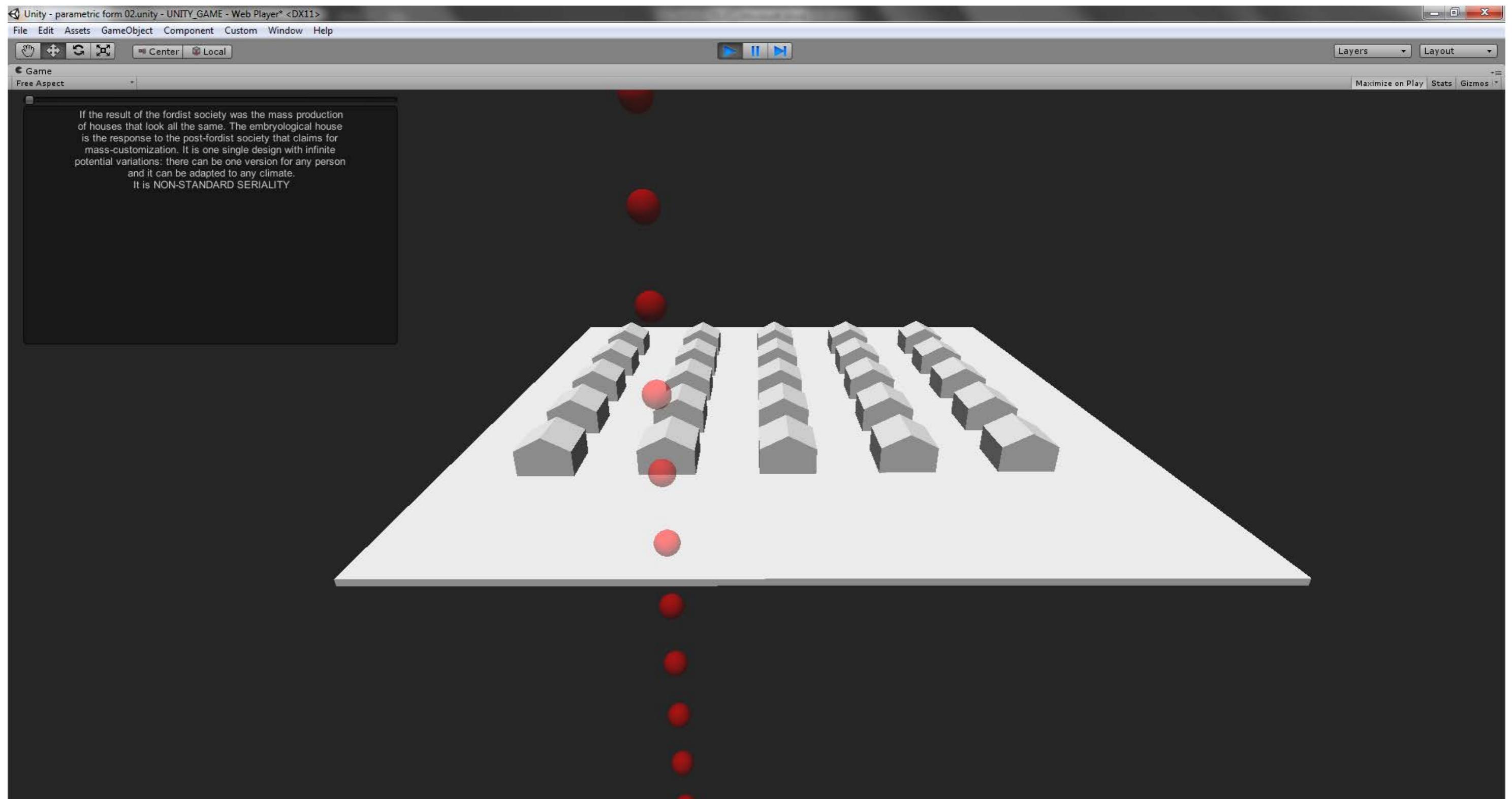
## Playing the game in Unity



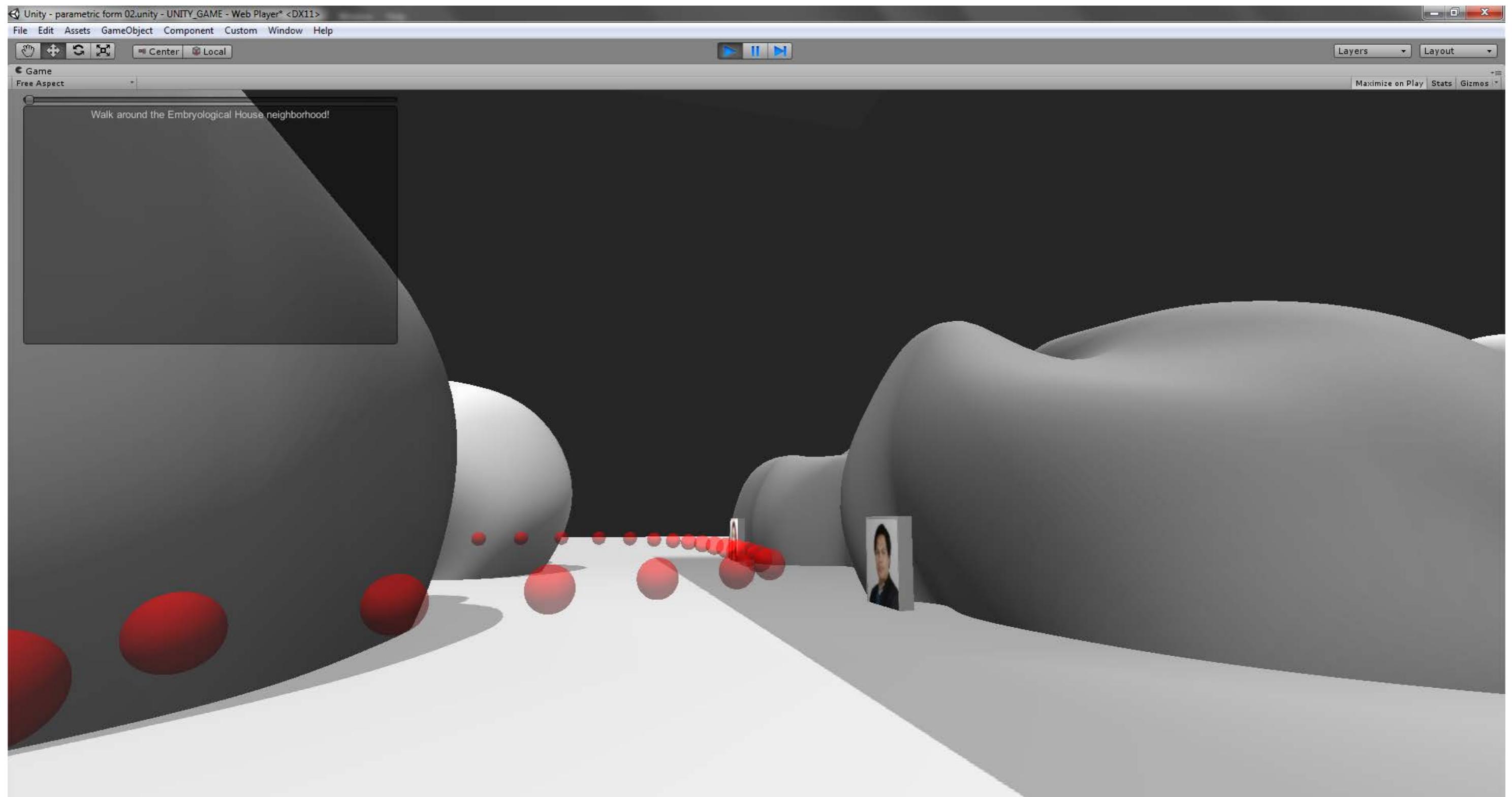
## Playing the game in Unity



## Playing the game in Unity

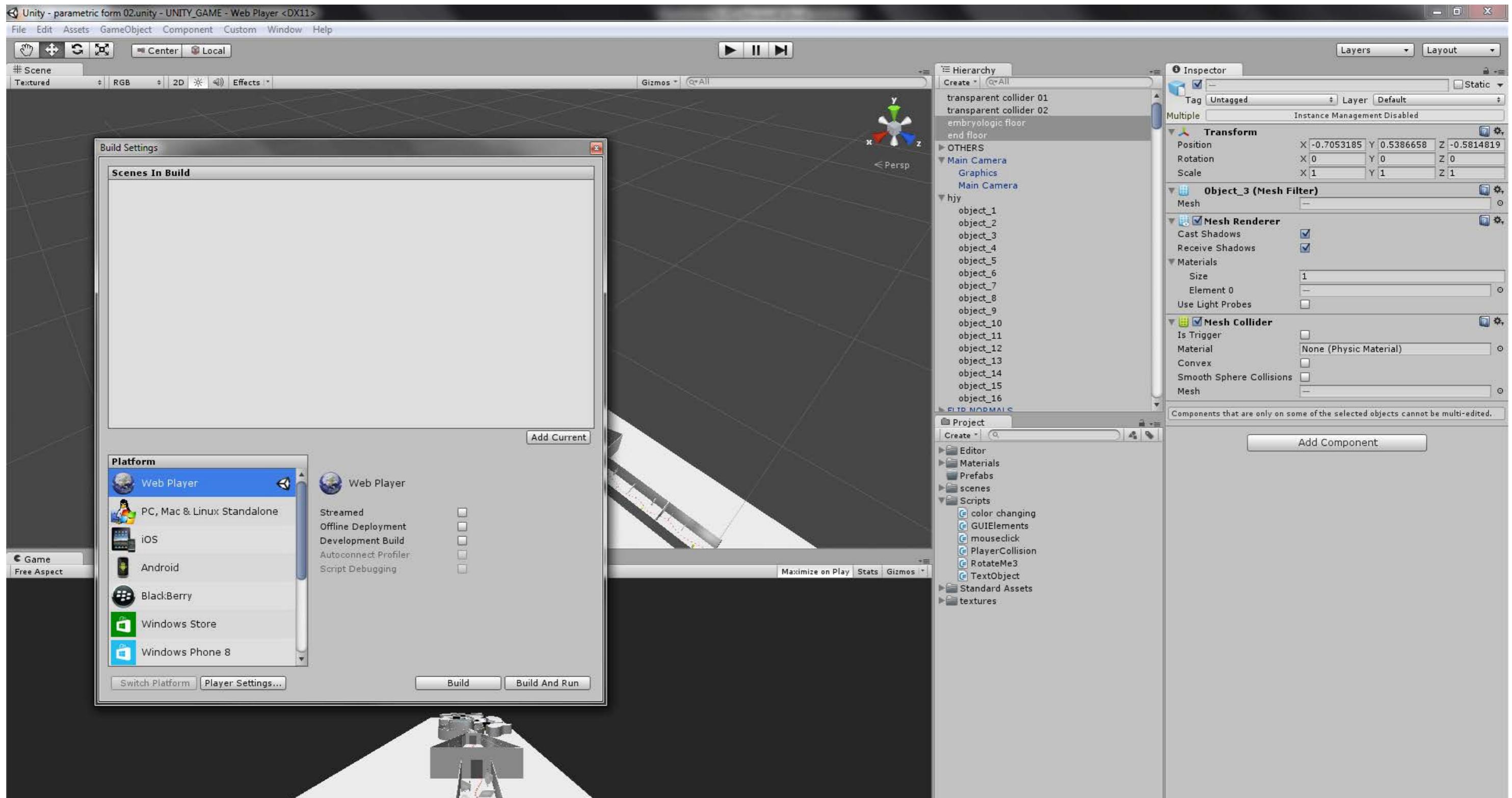


## Playing the game in Unity



# Exporting the game for the web

We build the game using Web Player format.

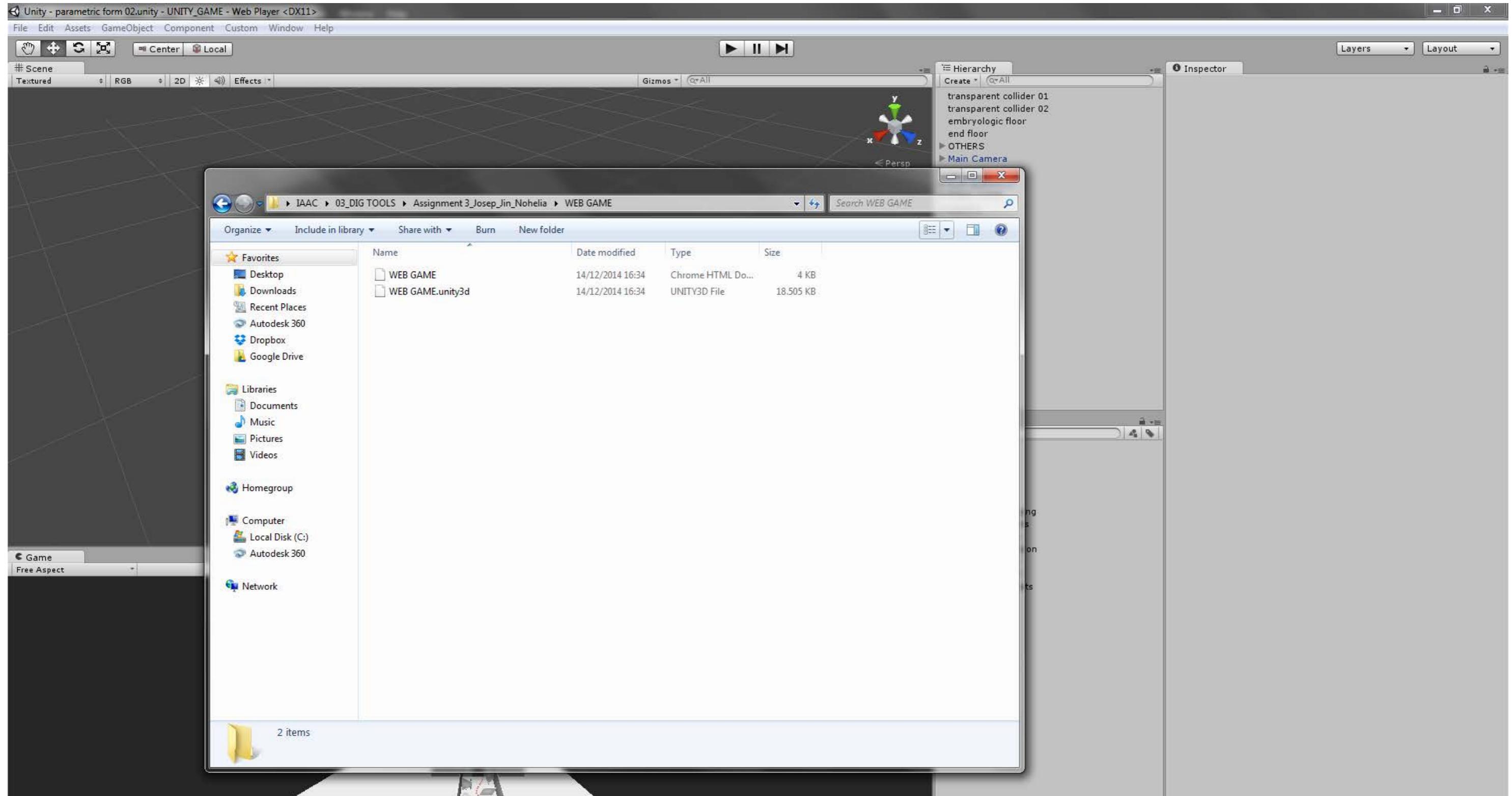


## Exporting the game for the web

Two files are created.

One .html file and one .unity3d file.

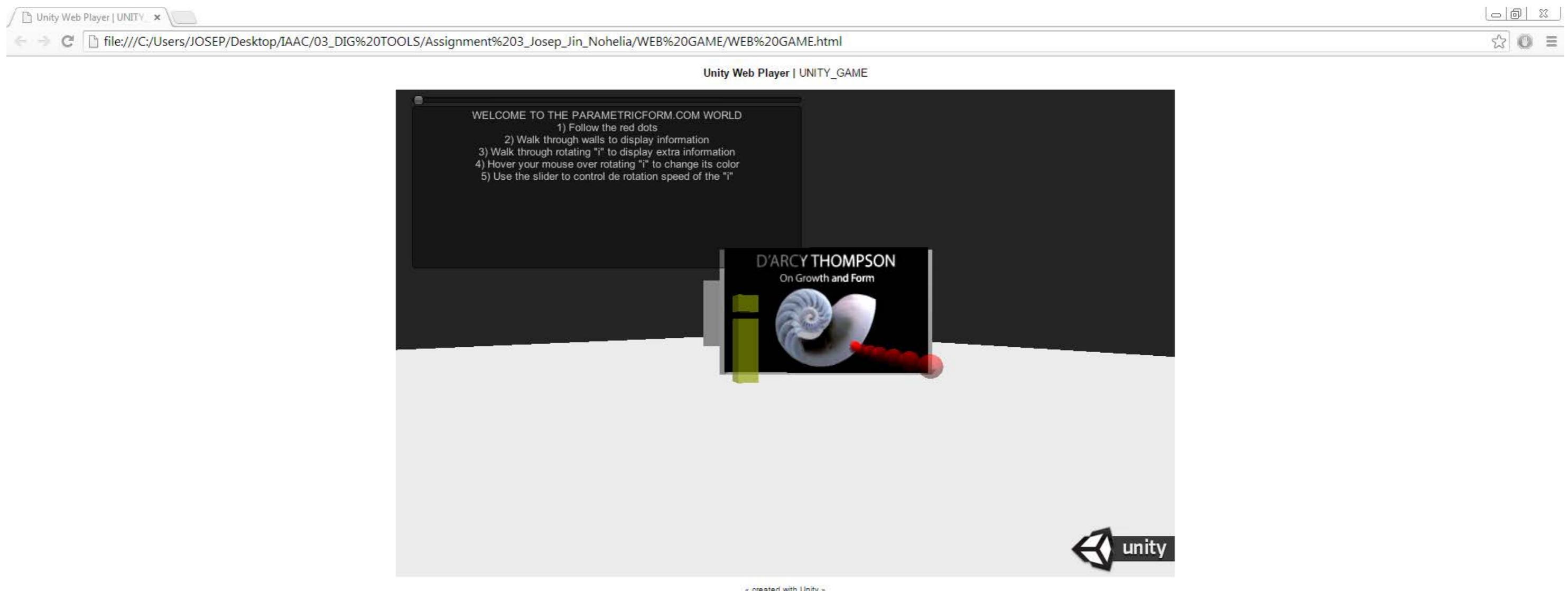
They have to be in the same location in order to be able to run the game.



## Playing the game from the computer with a web browser

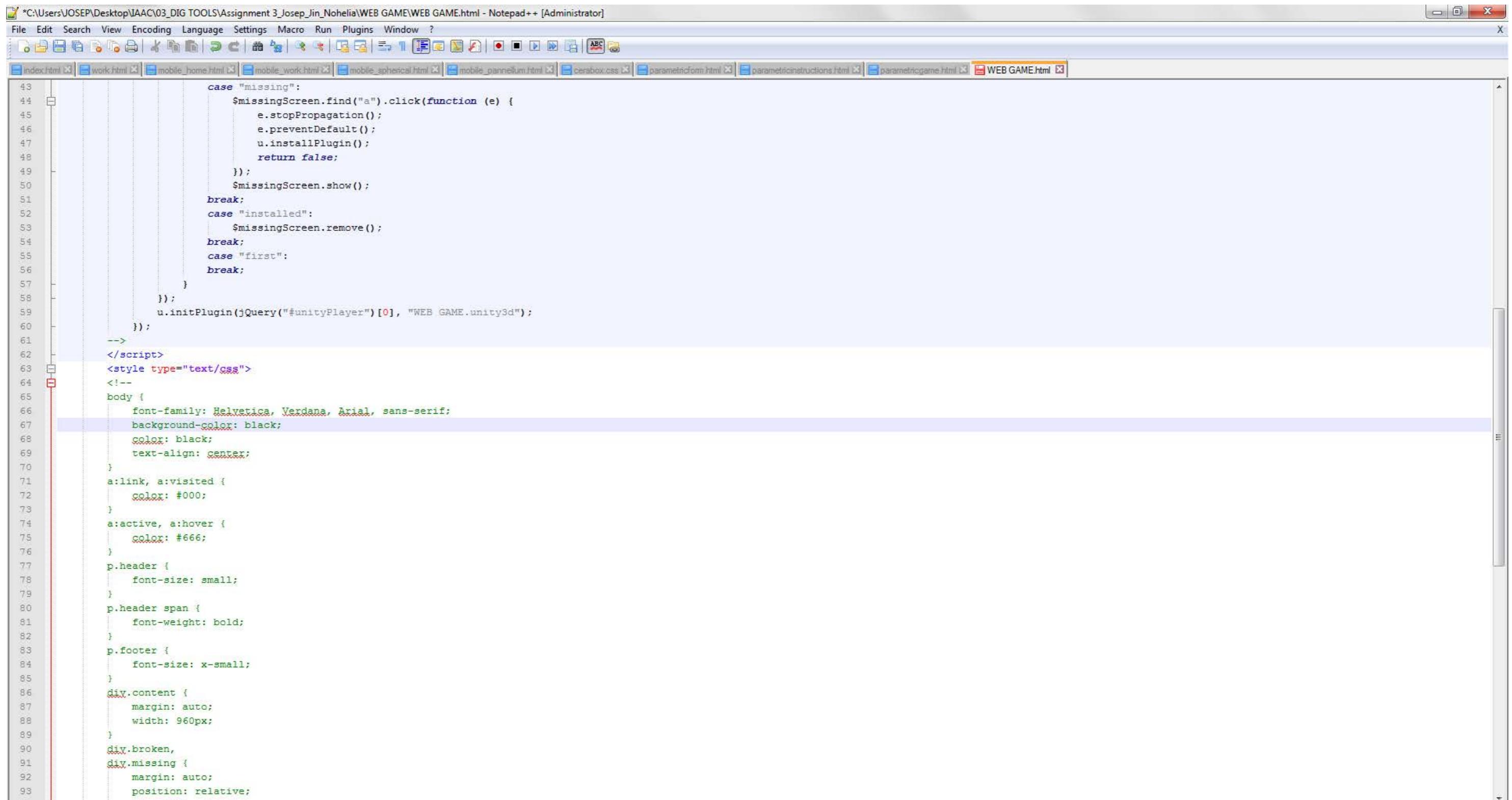
The .html file can be opened with a web browser that support WebGL.

Despite running it with a web browser the game is not online.



## Changing the background color of the .html file

We open the .html file of the game with a code editor and change the background color from white to black. Therefore the black texts won't be displayed and the page will look cleaner.

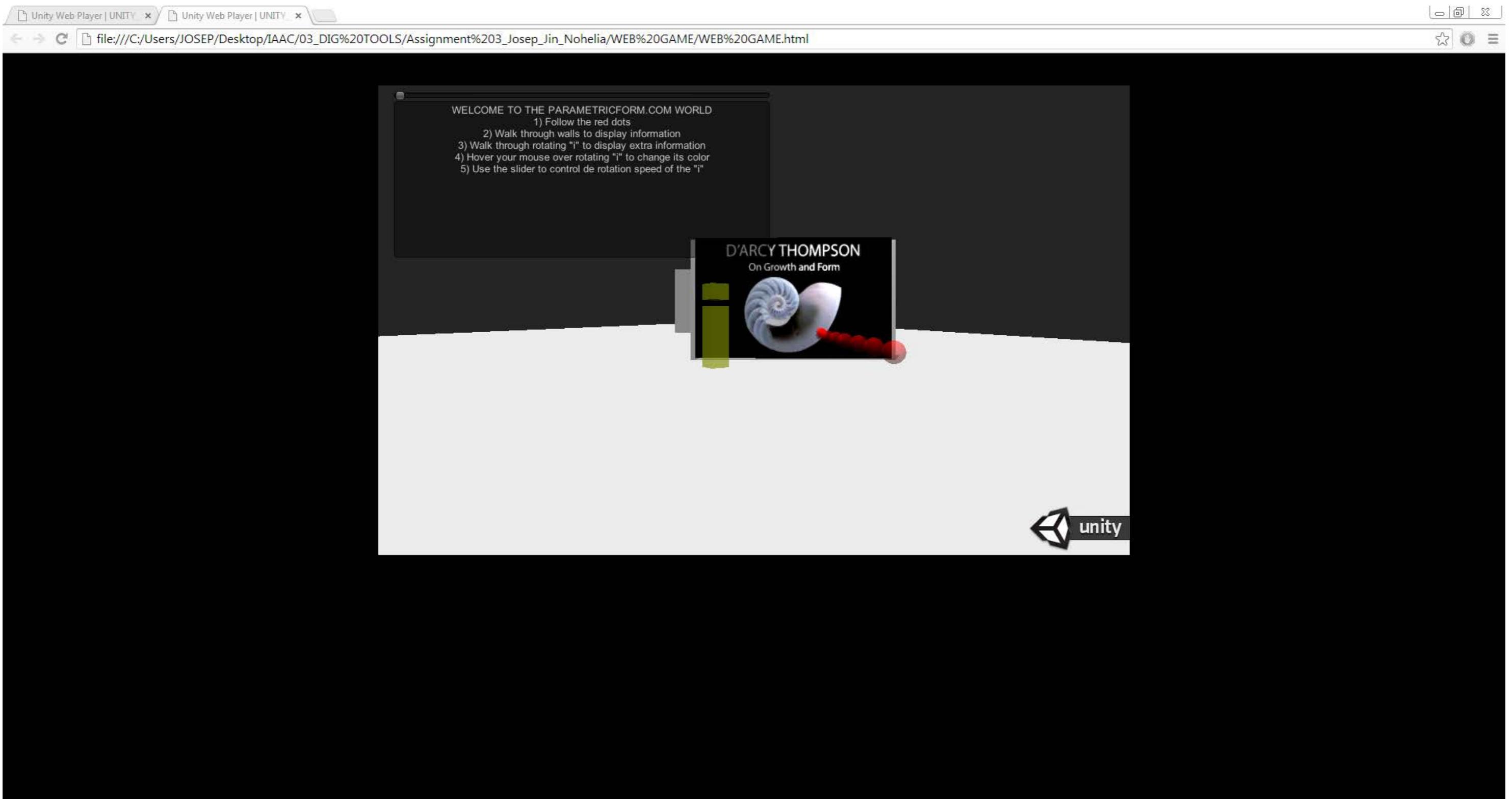


```
*C:\Users\JOSEP\Desktop\JAAC\03_DIG TOOLS\Assignment 3_Josep_Jin_Nohelia\WEB GAME\WEB GAME.html - Notepad++ [Administrator]
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
index.html mobile_home.html mobile_work.html mobile_spherical.html mobile_pannellum.html cerabox.css parametricform.html parametricinstructions.html parametricgame.html WEB GAME.html

43
44     case "missing":
45         $missingScreen.find("a").click(function (e) {
46             e.stopPropagation();
47             e.preventDefault();
48             u.installPlugin();
49             return false;
50         });
51         $missingScreen.show();
52     break;
53     case "installed":
54         $missingScreen.remove();
55     break;
56     case "first":
57     break;
58 }
59 u.initPlugin(jQuery("#unityPlayer")[0], "WEB GAME.unity3d");
60 });
-->
</script>
<style type="text/css">
<!--
body {
    font-family: Helvetica, Verdana, Arial, sans-serif;
    background-color: black;
    color: black;
    text-align: center;
}
a:link, a:visited {
    color: #000;
}
a:active, a:hover {
    color: #666;
}
p.header {
    font-size: small;
}
p.header span {
    font-weight: bold;
}
p.footer {
    font-size: x-small;
}
div.content {
    margin: auto;
    width: 960px;
}
div.broken,
div.missing {
    margin: auto;
    position: relative;
}

44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
```

## Changing the background color of the .html file



## Creating other .html pages for the website

We want to upload the game to the internet but we don't want it to be directly accessible.

We create a home page to receive the visitors and a page with the basic instructions of the game.

The structure of the site will be HOME PAGE click INSTRUCTIONS PAGE click GAME.

The image shows a dual-screen setup. On the left, a Notepad++ window displays the code for 'parametricform.html'. The code includes an HTML structure with a title 'PARAMETRIC FORM', a CSS section defining a 'fullscreen' style, and a body containing an anchor tag that links to 'parametricinstructions.html' and points to an image named 'parametrichome.jpg' with the id 'fullscreen'. On the right, a web browser window titled 'PARAMETRIC FORM' shows the final website. The website features a dark background with numerous abstract, organic, and geometric shapes resembling parametrically generated forms. Overlaid on these shapes is the text 'PARAMETRIC FORM .COM' in large, bold, white letters. Below it, a smaller text reads 'A virtual world to explore in a non-linear way the origin, concepts and applications of parametric design.' At the bottom center is a button labeled '[ CLICK TO CONTINUE ]'. The bottom right corner of the browser window contains the 'taac' logo and the text 'Advanced Architecture Concepts Jinyang Han + Josep Alcover + Nohelia Gonzalez'.

```
<!DOCTYPE html>
<html>
  <head>
    <title>PARAMETRIC FORM</title>
    <style type="text/css">
      #fullscreen {
        z-index: -2;
        height: 100%;
        width: auto;
        position: fixed;
        top: 0;
        left: 0px;
        margin-left: -2px;
      }
    </style>
  </head>
  <body>
    <a href="parametricinstructions.html">  </a>
  </body>
</html>
```

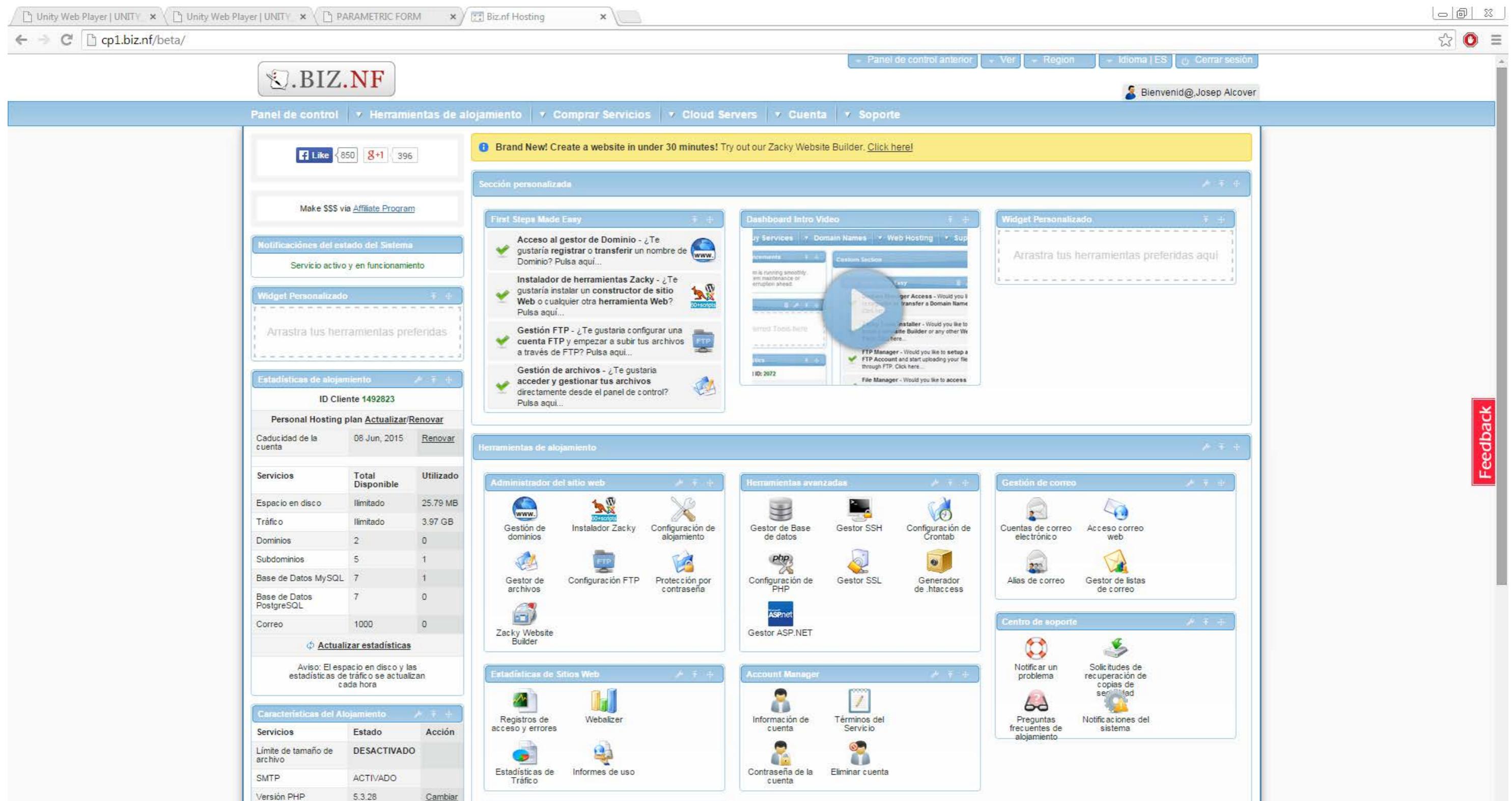
## Creating other .html pages for the website

The image shows a dual-screen setup. On the left, a Notepad++ window displays the code for a file named 'parametricinstructions.html'. The code is an HTML document with a CSS section defining a 'fullscreen' class for a background image. On the right, a browser window shows the resulting web page titled 'PARAMETRIC FORM .COM'. The page features a black background with several circular parametric shapes. It includes instructions for interacting with the content: 'Go Fullscreen' (with a double arrow icon), 'Look' (with a camera icon), and 'Move' (with a 3D cursor icon). A button at the bottom right says '[ CLICK TO CONTINUE]'. The browser's tab bar shows multiple other files like 'index.html', 'work.html', etc., indicating a larger project.

```
<!DOCTYPE html>
<html>
  <head>
    <title>PARAMETRIC FORM</title>
    <style type="text/css">
      #fullscreen {
        z-index: -2;
        height: 100%;
        width: auto;
        position: fixed;
        top: 0;
        left: 0px;
        margin-left: -2px;
      }
    </style>
  </head>
  <body>
    <a href="WEB GAME.html">  </a>
  </body>
</html>
```

## Web Hosting

We sign up for a free web hosting service in [www.biz.nf](http://www.biz.nf).  
 This gives us space in the internet.  
 We get a free .co.nf domain.



The screenshot shows the Biz.nf Hosting control panel interface. At the top, there are tabs for 'Panel de control', 'Herramientas de alojamiento', 'Comprar Servicios', 'Cloud Servers', 'Cuenta', and 'Soporte'. A banner at the top right says 'Brand New! Create a website in under 30 minutes! Try out our Zacky Website Builder. Click here!' Below this, there's a 'Sección personalizada' section with three boxes: 'First Steps Made Easy', 'Dashboard Intro Video', and 'Widget Personalizado'. The 'First Steps Made Easy' box contains links to 'Acceso al gestor de Dominio', 'Instalador de herramientas Zacky', 'Gestión FTP', 'Gestión de archivos', and 'Widget Personalizado'. The 'Dashboard Intro Video' box has a play button over it. The 'Widget Personalizado' box says 'Arrastra tus herramientas preferidas aquí'. The main area is divided into several sections: 'Herramientas de alojamiento' (with sub-sections like 'Administrador del sitio web', 'Estadísticas de Sitios Web', 'Herramientas avanzadas', and 'Account Manager'), 'Gestión de correo' (with sub-sections like 'Cuentas de correo electrónico', 'Alias de correo', and 'Centro de soporte'), and 'Características del Alojamiento' (with sub-sections like 'Servicios', 'Características del Alojamiento', and 'Notificaciones del estado del Sistema'). On the far right, there's a red 'Feedback' button.

## .com Domain

We buy a .com domain name from [www.godaddy.com](http://www.godaddy.com).

it is **PARAMETRICFORM.COM**

The screenshot shows the 'Detalles del Dominio' (Domain Details) page for the domain **PARAMETRICFORM.COM**. The domain is listed as **Activo** (Active), created on 06/12/2014, and expires on 06/12/2015. It is associated with the **Ninguno** folder and profile. Action buttons include **Renovar**, **Mejorar**, **Cambio de Cuenta**, and **Eliminar**.

**Configuración de Dominio** (Domain Configuration) section:

- Renovación Automática**: Estándar Activado (Standard Renewal Enabled)
- Bloquear**: Activado (Blocked)
- Servidores de nombres**: NS35.DOMAINCONTROL.COM, NS36.DOMAINCONTROL.COM (Updated 06/12/2014)
- Reenvío**: **Domino:** <http://josepalcover.co.nf/parametricform.html> Reenviar con Masking
- Premium DNS**: No propio (Own)
- Registros DS**: 0 registros DS creados
- Nombres del Host**: 0 nombres de host creados

**Código de Autorización**: Transferencia bloqueada hasta el 05/02/2015 Registro de dominio nuevo  
[Enviar mi código por correo electrónico](#)

**Instantánea de Cuenta** (Account Snapshot) sidebar:

- HOSTING**: Más información
- CORREO ELECTRÓNICO**: Office 365 no comprado. Más información
- CREADOR DE SITIOS WEB**: Más información

**Mejoras de Dominios** (Domain Enhancements) sidebar:

- PRIVACIDAD**: No propio. Agregar
- PROTECCIÓN DE LA PROPIEDAD DE UN DOMINIO**: No propio. Agregar
- DOMINIO CERTIFICADO**: No propio. Agregar
- GODADDY AUCTIONS®**: No propio. Agregar
- CERTIFICADO SSL**: No propio. Agregar
- REGISTRO DE EMPRESA**: No propio. Agregar

**Extras** sidebar:

- PAQUETE DE DOMINIO PERSONALIZADO**: Variaciones Disponibles. Agregar

## Forwarding parametricform.com to the .co.nf domain

We forward parametricform.com to the .co.nf domain that is linked to the web host server where we will upload the game and other .html files of the website.

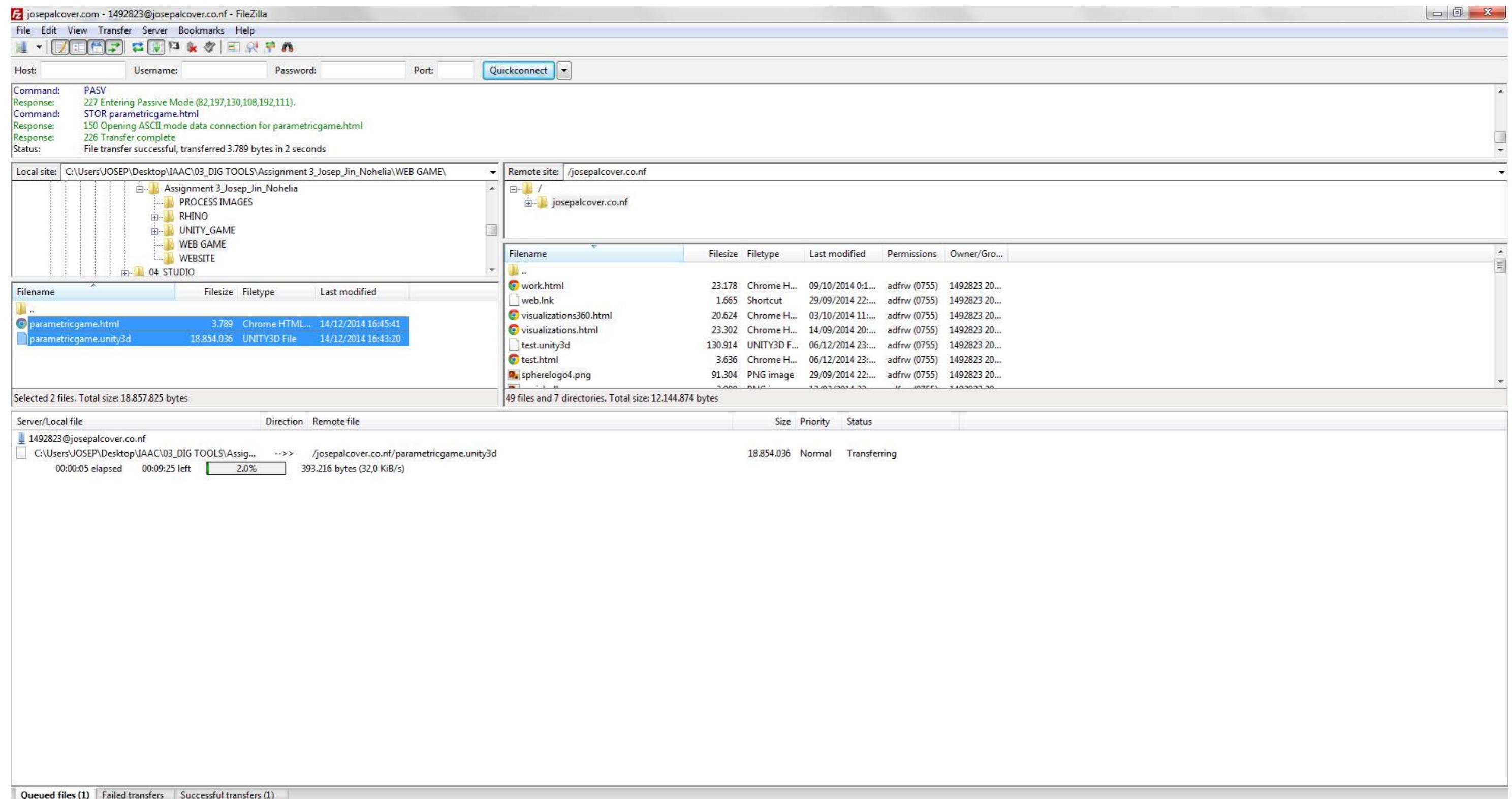
We activate the masking. This means that in the address bar of the browser we will always see parametricform.com.

The screenshot shows the GoDaddy.com domain management interface. The main menu at the top includes 'Mi cuenta', 'Dominios', 'Asistencia técnica', and 'Josep'. Below the menu, there are tabs for 'Dominios', 'Comprar y Vender', 'Configuraciones', and 'Ayuda'. The central area displays the 'Detalles del Dominio' (Domain Details) for 'PARAMETRICFORM.COM'. The domain status is 'Activo' (Active), with creation and expiration dates of 06/12/2014. Below this, there are sections for 'Configuración de Dominio' (Domain Configuration), 'Bloquear' (Lock), 'Servidores de nombres' (Name Servers), 'Reenvío' (Forwarding), 'Premium DNS', 'Registros DS', and 'Nombres del Host'. A modal dialog box titled 'Reenvío y Masking' is open over the configuration section. It contains a sub-section for 'PARAMETRICFORM.COM' with the text: 'El reenvío de nombre de dominio te permite dirigir automáticamente a los visitantes de tu nombre de dominio a un sitio web diferente.' It has tabs for 'Dominio' and 'Subdominio'. Under 'Dominio', there is a table with three columns: 'Reenviar a:', 'redirección', and 'Tipo'. The first row shows 'http://josepalcover.co.nf/paramet...' as the redirect URL, '301 (Permanente)' as the redirection type, and a checked checkbox for 'Reenviar con Masking'. Below the table is a green 'Actualizar Reenvío' button. At the bottom of the dialog are 'Guardar' and 'Cancelar' buttons. To the right of the main content area, there is a sidebar titled 'Instantánea de Cuenta' (Account Snapshot) with sections for 'HOSTING', 'CORREO ELECTRÓNICO', 'CREADOR DE SITIOS WEB', 'Mejoras de Dominios', and 'Extras' (Extras). The 'Mejoras de Dominios' section lists options like 'PRIVACIDAD', 'PROTECCIÓN DE LA PROPIEDAD DE UN DOMINIO', 'DOMINIO CERTIFICADO', 'GODADDY AUCTIONS', 'CERTIFICADO SSL', and 'REGISTRO DE EMPRESA'. The 'Extras' section lists 'PAQUETE DE DOMINIO PERSONALIZADO'.

## Uploading the files with FileZilla

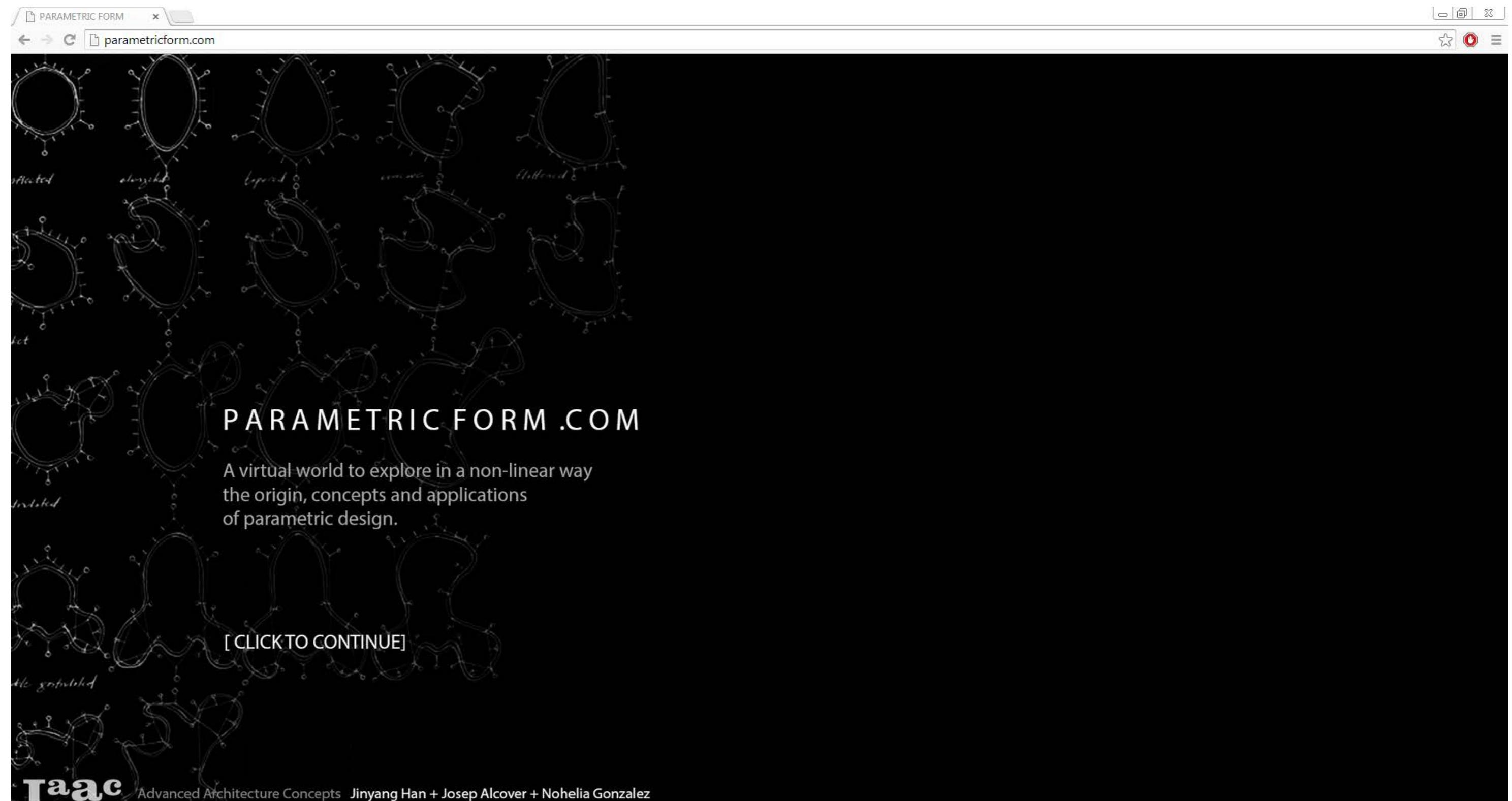
We use FileZilla to upload the files to our web host server.

We upload 3 .html files (the home page, the instruccions page and the game page) and 1 .unity3d file that stores information of the game.

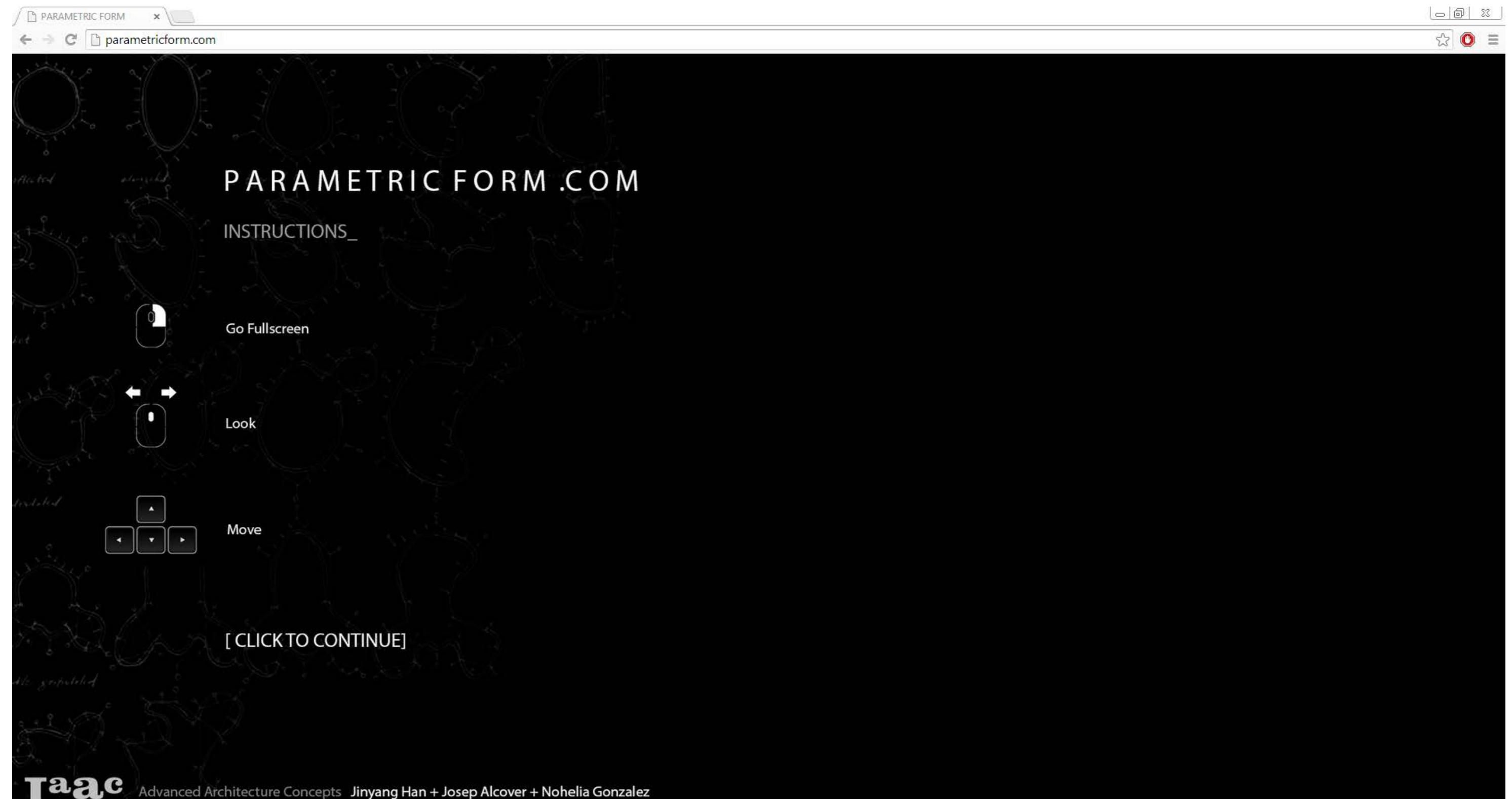


## Playing the game online

Now the game is in PARAMETRICFORM.COM  
and everyone can play it.



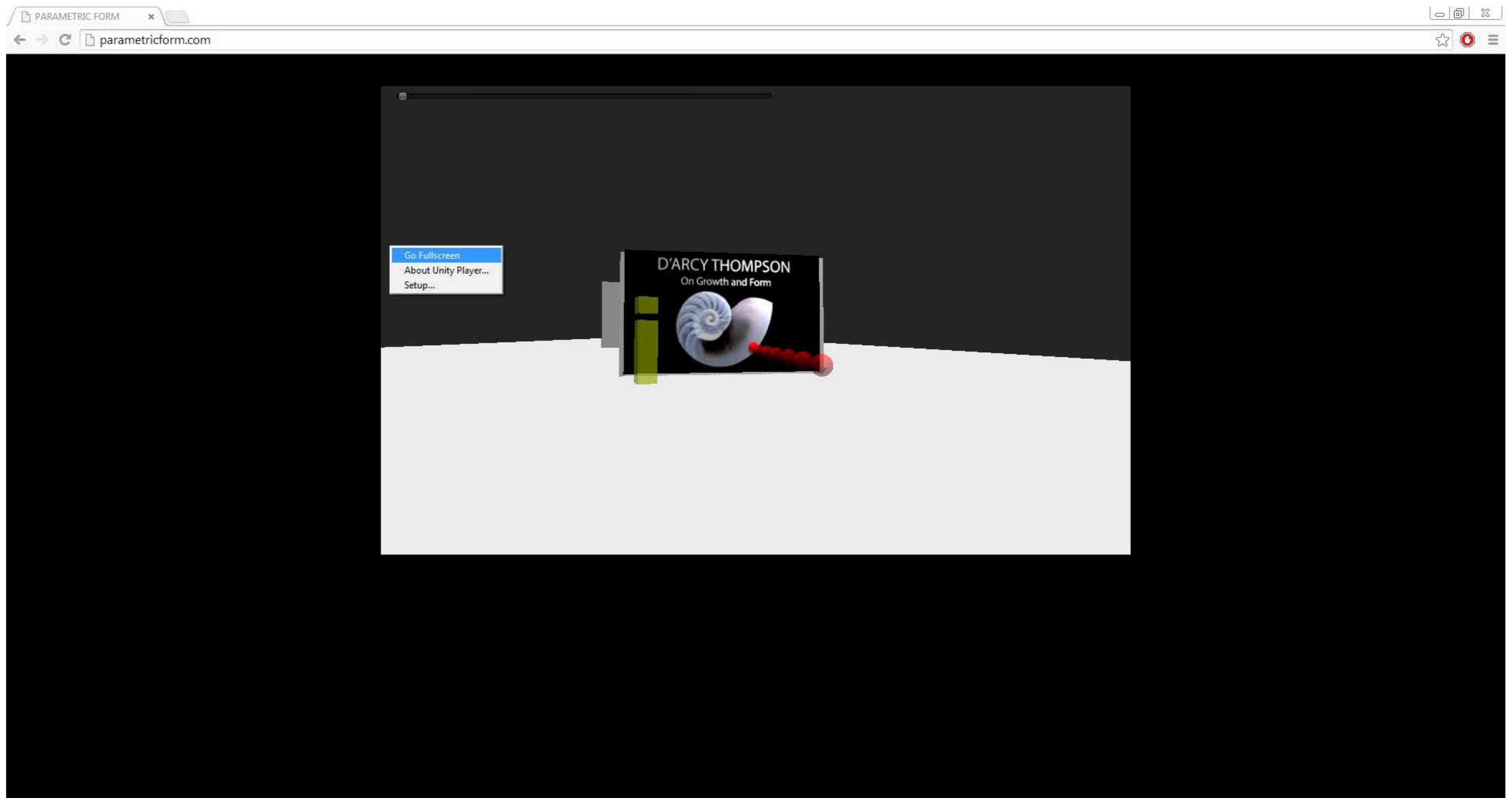
Playing the game online



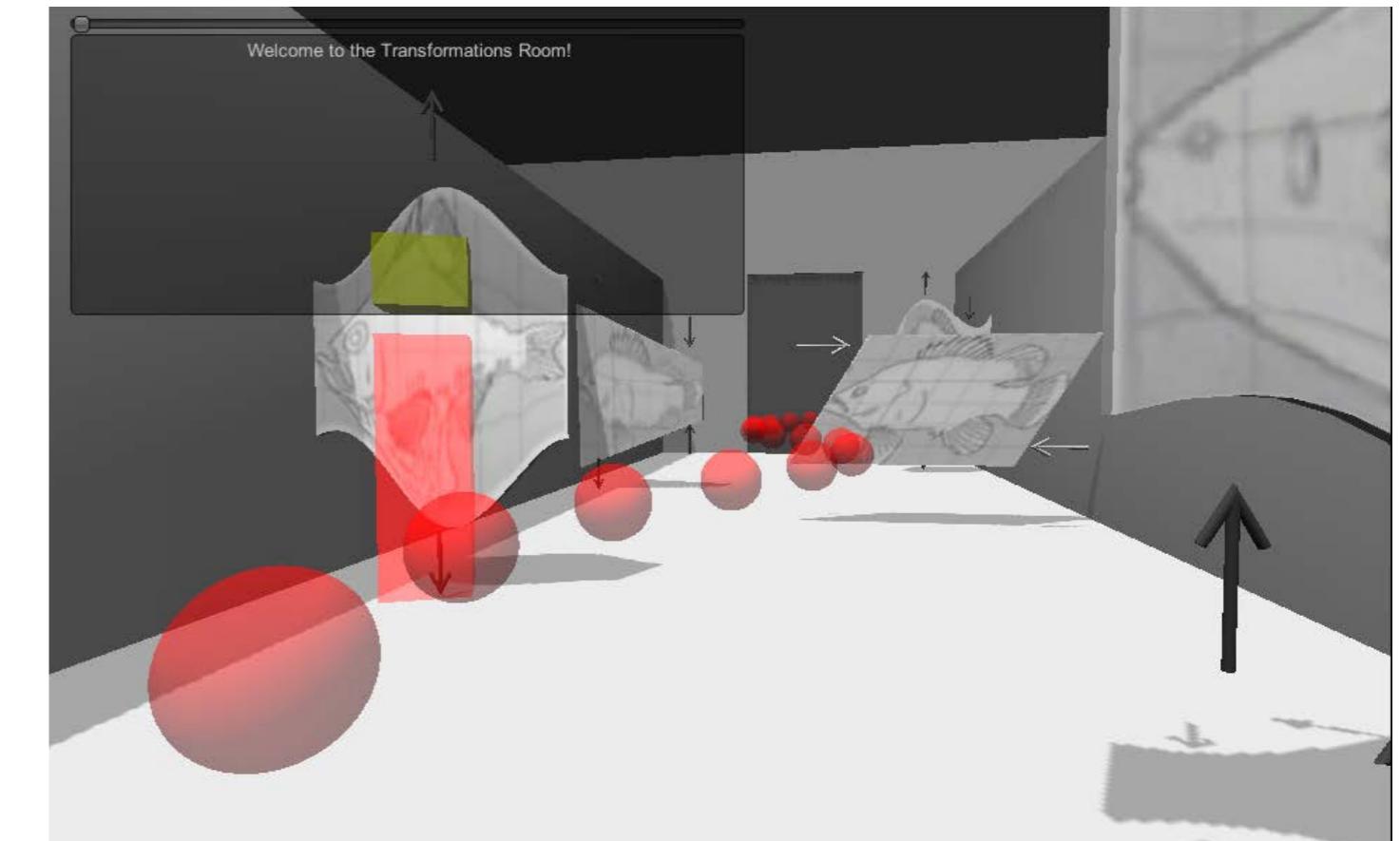
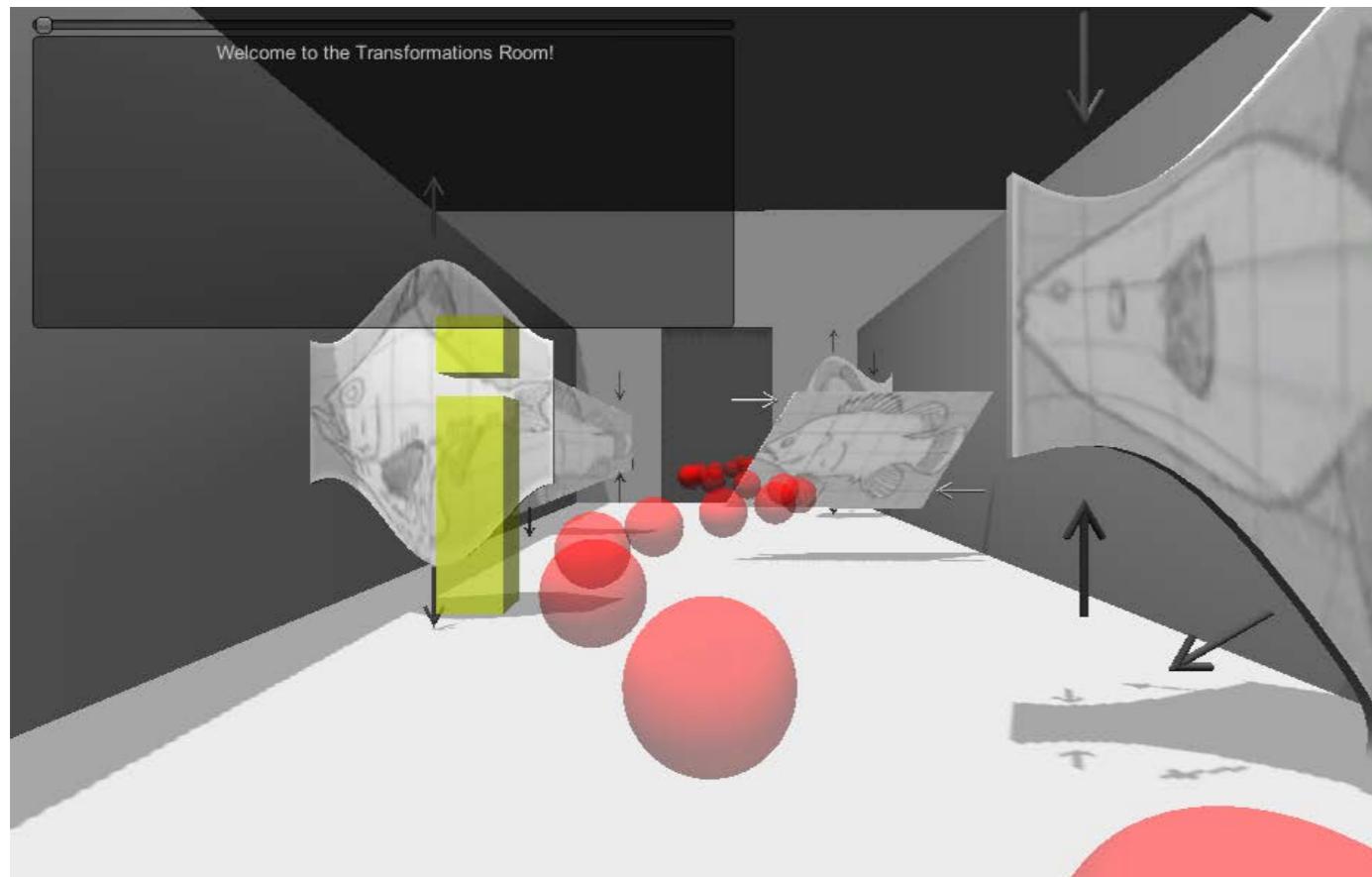
## Playing the game online



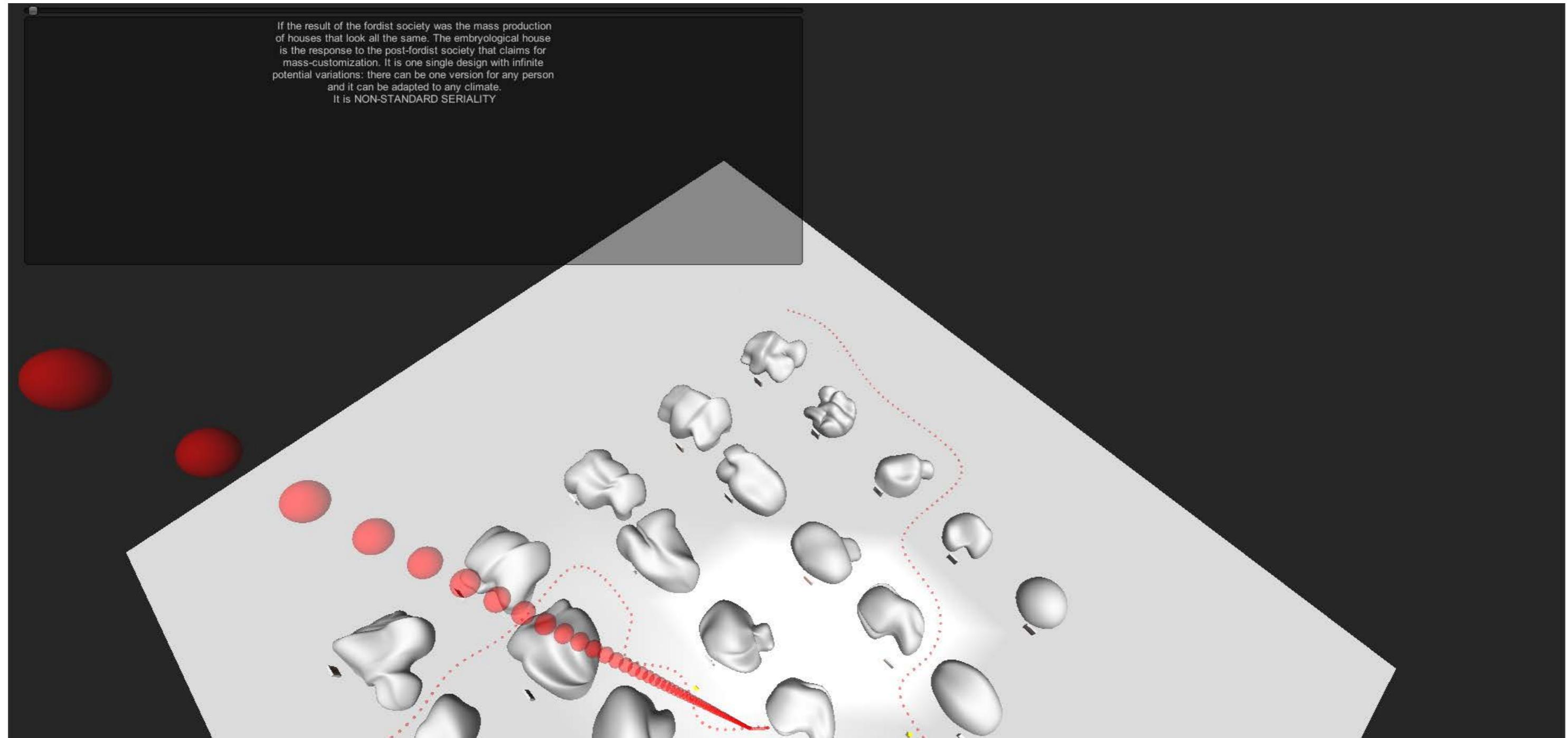
## Playing the game online

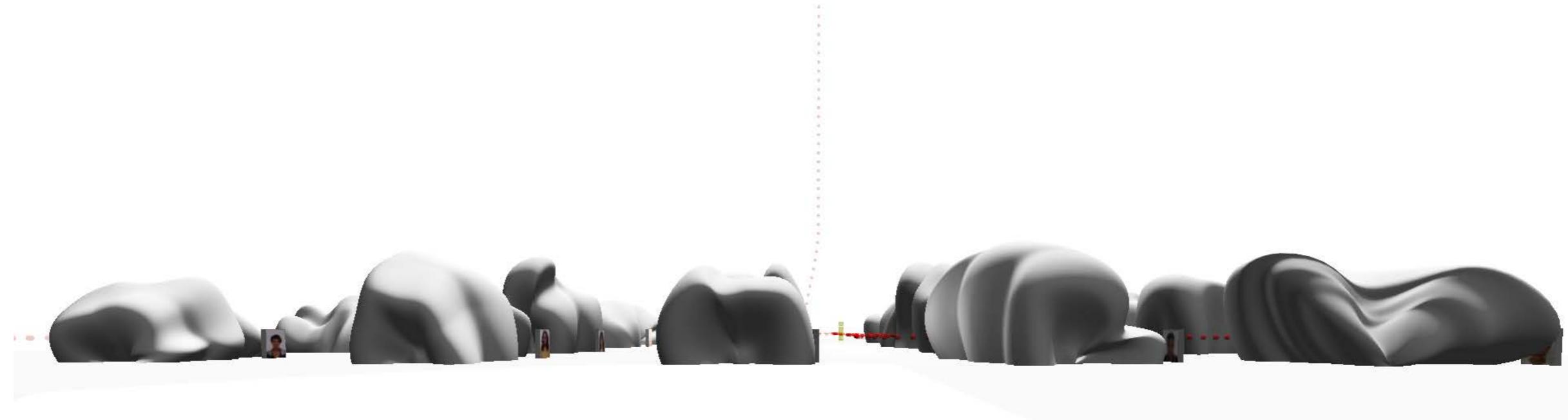


## Playing the game online



## Playing the game online





PARAMETRIC FORM .COM

Gameify! (Unity 3D)  
Digital Tools  
Josep Alcover Llubiá + Nohelia Gonzalez + Jinyang Han **Iaac**