Unity Scripting 4

Unity Components overview
Particle components
Interaction – Key and Button input
Parenting
CAVE2 Interaction – Wand / Wanda™ VR Input Devices
Project Organization
Prefabs
Instantiate
# Unity Components

<table>
<thead>
<tr>
<th>Animation components</th>
<th>Mesh components</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asset components</td>
<td>Network group</td>
</tr>
<tr>
<td>Audio components</td>
<td>Particle components</td>
</tr>
<tr>
<td>Physics components</td>
<td>Rendering components</td>
</tr>
<tr>
<td>The GameObject</td>
<td>Transform component</td>
</tr>
<tr>
<td>Image effect scripts</td>
<td>UnityGUI group</td>
</tr>
<tr>
<td>Settings managers</td>
<td>Wizards</td>
</tr>
</tbody>
</table>
Unity Components

Audio components

Implement sound in Unity

- Audio Listener
  Add this to a Camera to get 3D positional sound.
- Audio Source
  Add this Component to a GameObject to make it play a sound
- Audio Effects
Unity Components

Physics components
Physic Material
Rigidbody
Box Collider
Capsule Collider
Character Controller
Character Joint
Configurable Joint
Constant Force
Fixed Joint
Hinge Joint
Spring Joint
Interactive Cloth
Skinned Cloth...
Unity Components

The Game Object

GameObjects are containers for all other Components
All objects in your scene are inherently GameObjects

They are containers that hold Components, which implement actual functionality. Ex. a Light is a Component which is attached to a GameObject.
Unity Components

Mesh components

3D Meshes are the main graphics primitive of Unity. Various components exist in Unity to render regular or skinned meshes, trails or 3D lines.

- Mesh Filter
- Mesh Renderer
- Skinned Mesh Renderer
- Text Mesh
Particle components

Particle Systems are used to make effects
Smoke
Steam
Fire
Atmospheric effects

Particle systems in Unity work by using one or two textures (2D), and drawing them many times, creating a chaotic random effect.
Unity Components

Particle components

Ellipsoid Particle Emitter
Line Renderer
Mesh Particle Emitter
Particle Animator
Particle Renderer
Trail Renderer
Particle Collider
Unity Components

Particle components

Ellipsoid Particle Emitter
  spawns particles inside a sphere
  Use the Ellipsoid property to scale & stretch the sphere
Unity Components

Particle components
Line Renderer

draws a straight line between two or more points in 3D space can be used to draw anything from a simple straight line, to a complex spiral renders billboard lines that have width and can be textured uses the same algorithm for line rendering as the Trail Renderer
Particle components
Trail Renderer

makes trails behind moving objects in the scene
Unity Components

Particle components
Particle animator

Move particles over time
Used to apply wind, drag and color cycling to particle systems
Unity Components

Particle components
Mesh particle emitter

emits particles around a mesh
particles are spawned from the surface of the mesh
Particle component

Open your Unity scene GameObject > Particle System

Rename it “Confetti”
Particle component

Particle Effect window / Inspector

Particle Effect window / Inspector
Particle component

Open Particle Editor
Adjust Size over Lifetime parameter in the graph view
Adjust Color over Lifetime in the Editor
Particle component

Create a script “ParticleScript”

```javascript
public var confettiEmitter : ParticleSystem;

function Start() {
    confettiEmitter = GameObject.Find("Confetti").GetComponent(ParticleSystem);
}

function Update() {
    if (Input.GetButtonDown("Fire1")) {
        confettiEmitter.Emit(30); //emits 30 particles
    }
}
```
Particle component

Assign “ParticleScript” to a cylinder (drag and drop)

Assign Confetti Particles to Confetti Emitter Var in the Inspector (drag and drop from Hierarchy)

Uncheck “Play on Awake” box in the Inspector

With selected Confetti object in Hierarchy
Assign “ParticleScript” to a cylinder (drag and drop).
Assign Confetti Particles to Confetti Emitter Var in the Inspector (drag and drop from Hierarchy).
Assign “ParticleScript” to a cylinder (drag and drop)
Assign Confetti Particles to Confetti Emitter Var in the Inspector (drag and drop from Hierarchy)
Particle component

Test the mouse button input and interaction
The particle should emit 30 particles on mouse click
Interaction – Key and Button input

Add Rigidbody component to the Cylinder
In the Inspector
private var orange : Color = Color(0.8, 0.4, 0.0, 0.7);
private var green : Color = Color(0.0, 0.9, 0.2, 0.7);
var newMaterial : Material;
var newMaterial2 : Material;

function Update()
{
if (Input.GetButtonDown("Fire1"))
{
GetComponent.<Renderer>().material.color = orange;
newMaterial.color = orange;
}
if (Input.GetButtonDown("Fire2"))
{
GetComponent.<Renderer>().material.color = green;
newMaterial2.color = green;
}
Interaction – Key and Button input

Create addForceScript and assign it to the Cylinder
Detects mouse clicks on an element

#pragma strict

function OnMouseDown ()
{
    GetComponent.<Rigidbody>().AddForce(transform.forward * 500f);

    GetComponent.<Rigidbody>().useGravity = true;
}

Interaction – Key and Button input

Use ESC key to test mouse input and the Add Force function
Interaction – Key and Button input

Use ESC key to test mouse input and the Add Force function
Parenting

Parent-Child Relationship in Unity
The Child Game Object will inherit the behaviour of its parent. It will move, rotate, scale exactly as its Parent does.
Similar to Arm/Body relationship—whenever your body moves, your arm moves along with it.

- This is a way of grouping objects together
- Children can have their own children and etc.
- Complex parent-child structure
- Represented in the Hierarchy window
Parenting

Drag and drop Confetti particles inside the Cylinder object in the Hierarchy window

Test the game
Drag and drop Confetti particles inside the Cylinder object in the Hierarchy window. Test the game.
The wand is the major input device used to interact with and control a virtual reality (VR) experience in the CAVE and other VR systems. It is essentially a 3D mouse, with a receiving antenna attached which provides the computer with information about the wand’s position and orientation.

buttons and a joystick
Interaction – Wand / Wanda™ VR Input Devices

The joystick is used primarily for navigation in combination with the position and orientation information.

The buttons are used to set modes and select options.
Project Assets

Assets are the models, textures, sounds and all other media files from which you make your 3D project

- Audio Files
- Materials
- Meshes
- Textures
- Prefabs
- Scripts
- Prefabs
Project Organization

Project > Create > Folder
Project Organization

Assets > Folder >

Rename

- Materials
- Scripts
- Scenes
- Prefabs
- Textures
- Sounds
Prefabs and reusable assets

Create Prefab:

- Project > Create > Prefab
- Rename it “PrefabSphere”
- Game Object > 3D > Sphere
- Inspector : Add RigidBody Component
- Drag and drop sphere into PrefabSphere
- Delete Sphere n the Hierarchy window / scene
Prefabs
Prefabs

particles.unity - unity2 - PC, Mac & Linux Standalone (Personal) <OpenGL 4.1>
Prefabs

Create InstantiateScript

```javascript
var prefabSphere : Transform;

function Update () {
    if (Input.GetButtonDown("Jump")) {
        var instanceObject = Instantiate(prefabSphere, transform.position, transform.rotation);
    }
}
```
Prefabs

Assign script to FP character Controller (add one if your scene has no FPC)

Assign prefabSphere to New Object in the Inspector

Test the game ad press Space bar to instantiate spheres
Prefabs

Assign script to FP character Controller (add one if your scene has no FPC)

Assign prefabShpere to New Object in the Inspector

Test the game and press Space bar to instantiate spheres
Prefabs

Assign script to FP character Controller (add one if your scene has no FPC)

Assign prefabShpere to New Object in the Inspector

Test the game and press Space bar to instantiate spheres during run-time
**Instantiate**

1. **Which object to instantiate?**
   
   The best way is to expose a variable.

   We can state which object to instantiate by using drag and drop to assign a game object to this variable prefab.

2. **Where to instantiate it?**

   Create the new game object wherever the user (FPC) is currently located whenever the Jump button is pressed.
Instantiate

Instantiate is to create objects during run-time (as the game is being played)

Instantiate function takes three parameters;
(1) the object we want to create
(2) the 3D position of the object
(3) the rotation of the object

```csharp
if (Input.GetButtonDown("Jump")) {
    var instanceObject = Instantiate(prefabSphere, transform.position, transform.rotation);
}// the transform that the script is attached to (FP Character)
```
To clean generated prefabs from the scene, attach the following script “destroyPrefabs” to the prefabSphere

```javascript
var timeRemaining = 3.0;

function Update()
{
    timeRemaining -= Time.deltaTime;
    if (timeRemaining <= 0.0)
    {
        Destroy(gameObject);
    }

```
To add a direction in which the prefabs are moving, add a forward pointing force to the script:

```javascript
var forwardForce = 1000;

instanceObject.GetComponent.<Rigidbody>().AddForce(transform.forward * forwardForce);
```
CAVE2 Unity Tutorial