

Introduction to Unity

Jaanus Jaggo

Unity



- Largest community
- 2D / 3D
- C#

Unity is a cross-platform game engine developed by Unity Technologies.

First released in 2005.

20+ platforms including: Windows, Mac, Linux, Android, Iphone, WebGL, Consoles...

53% of top 1000 grossing mobile games globally

Architecture

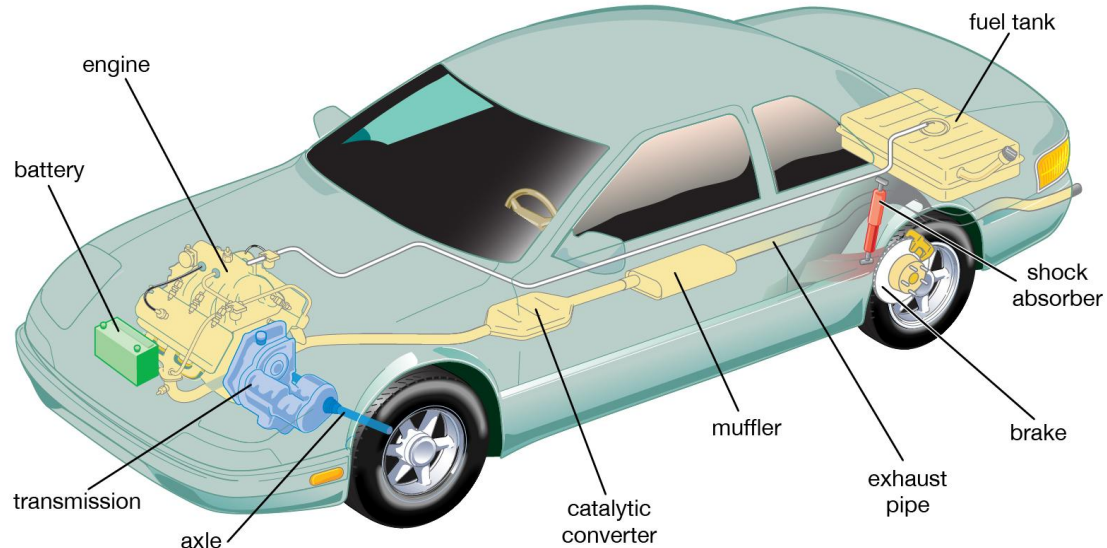
- Components
- Scriptable objects
- Unity and C# events
- Assembly files
- Coroutines
- Execution order

Architecture

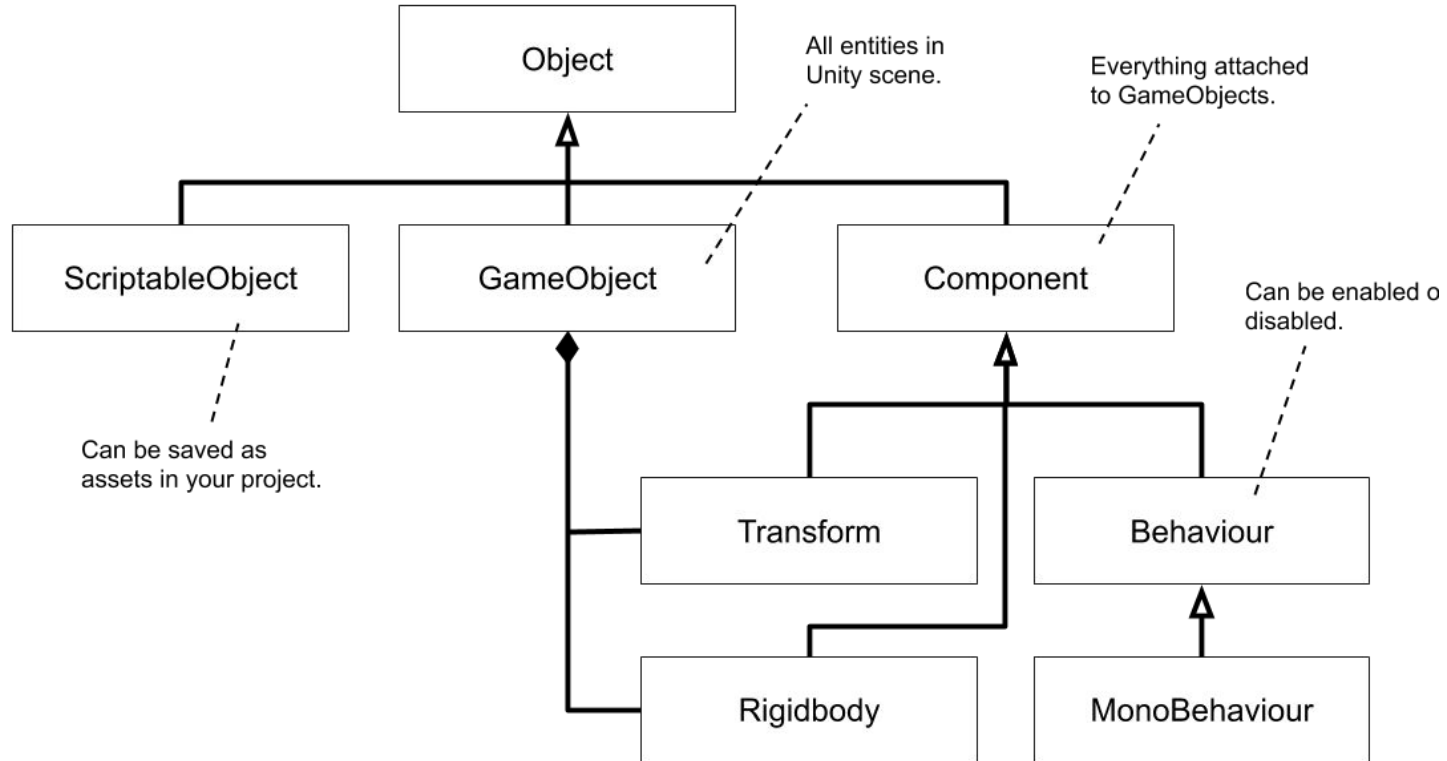
- **Components**
- Scriptable objects
- Unity and C# events
- Assembly files
- Coroutines
- Execution order

Components

Project is made of **scenes**, scenes are made of **game objects**, game objects are made of **components**.



Engine classes



Architecture

- Components
- **Scriptable objects**
- Unity and C# events
- Assembly files
- Coroutines
- Execution order

Scriptable objects

Scriptable objects are the data containers that can be saved as assets in the project.



Architecture

- Components
- Scriptable objects
- **Unity and C# events**
- Assembly files
- Coroutines
- Execution order

Unity and C# events

Unity events

- Connectable in inspector
- Can only have certain types
- Slower

```
import UnityEngine.Events  
  
[Serializable]  
  
public class SomeEvent : UnityEvent<float> { }
```

C# events

- Can be invoked in the same class
- Can have a return value with Func
- Faster

```
public event Action<float> OnSet;  
  
public event Func<float, float> OnRequest;  
  
OnSet?.Invoke(value);
```

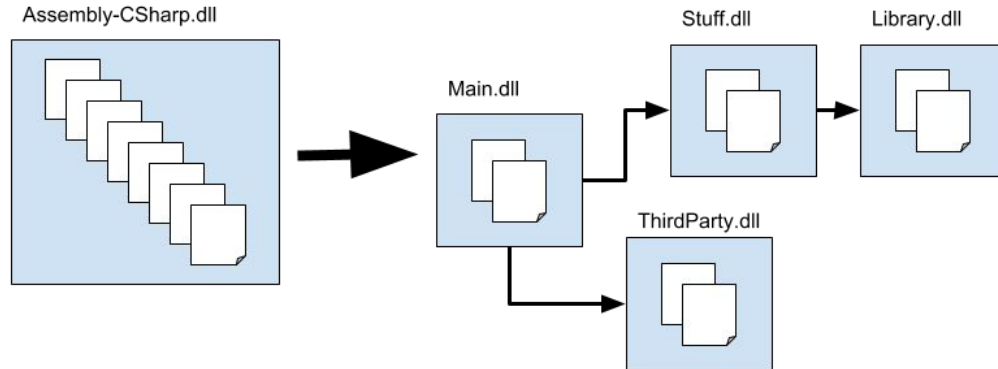
Architecture

- Components
- Scriptable objects
- Unity and C# events
- **Assembly files**
- Coroutines
- Execution order

Assembly files

For compiling code in separate dll files.

- Faster compile times
- Guarantees decoupling



Architecture

- Components
- Scriptable objects
- Unity and C# events
- Assembly files
- **Coroutines**
- Execution order

Coroutines

Like a function that has the ability to pause execution and return control to Unity.

They are not using multithreading!

```
void Update()
{
    if (Input.GetKeyDown("f"))
    {
        StartCoroutine("Fade");
    }
}
```

```
IEnumerator Fade()
{
    for (float ft = 1f; ft >= 0; ft -= 0.1f)
    {
        Color c = renderer.material.color;
        c.a = ft;
        renderer.material.color = c;
        yield return new WaitForSeconds(.1f);
    }
}
```

<https://docs.unity3d.com/Manual/Coroutines.html>

Architecture

- Components
- Scriptable objects
- Unity and C# events
- Assembly files
- Coroutines
- **Execution order**

Execution order

Script lifecycle (ordered in first to last):

1. Awake
2. OnEnable
3. Start
4. FixedUpdate
5. Update
6. LateUpdate
7. OnRender(...)
8. OnGUI
9. OnDisable
10. OnDestroy

More info: <https://docs.unity3d.com/Manual/ExecutionOrder.html>

Execution order

Execution order of scripts: **Edit > Project Settings**

Script Execution Order

Add scripts to the custom order and drag them to reorder.

Scripts in the custom order can execute before or after the default time and are executed from top to bottom. All other scripts execute at the default time in the order they are loaded.

(Changing the order of a script may modify the meta data for more than one script.)

Default Time

UnityEngine.EventSystems.HoloLensInput	100	-
LoadBundle	200	-
UnityEngine.XR.WSA.SpatialMappingBase	250	-
LoadTextures	300	-
BuildiOSAppSlices	400	-
		+ -

Revert Apply