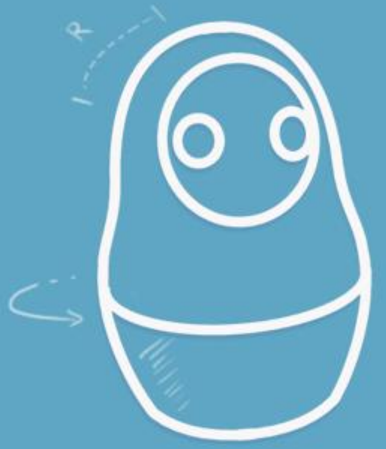
Sperasoft

Your game dev partner



Unity Programming

Introducing Unity3D

Unity3D is a full 3D engine that supports most modern technologies. The main features of Unity3D is cross-platform and very powerful customization tools to create the game

Unity3D Features:

- **Full 3D engine**
- **Cross-platform**
  - **Windows**
  - **Mac**
  - **Web player**
  - **iOS**
  - **Android**
  - **Windows RT**
  - **Blackberry**
  - **Xbox 360, PC 3, WII**
- **Very flexible and convenient**

Introducing Unity3D

Unity3D Key Technologies:

- Extensible plug-in architecture
- Language for scripting game logic
- Shader support
- 3D animator (Mecanim)
- Particle system (Shuriken)
- Physic (PhysX)
- Audio system
- Profiler

Extensible Plug-in Architecture

Popular third-party developers plug-ins used to increase Unity3D tools capacity:

- **2DToolkit**
- **NGUI**
- **PlayMaker**
- **Prime31 plugins**
- **GamesAnalytics**
- **UnityVS**

Language for Scripting Game Logic

Scripting languages used for game logic and default library.

Languages:

- C#
- Javascript (subset)
- Boo

Frameforks:

- Mono 2.6
- UnityEngine
- .NET for Win8/WinRT

Unity3D Shader Support

Use the following languages for writing shaders:

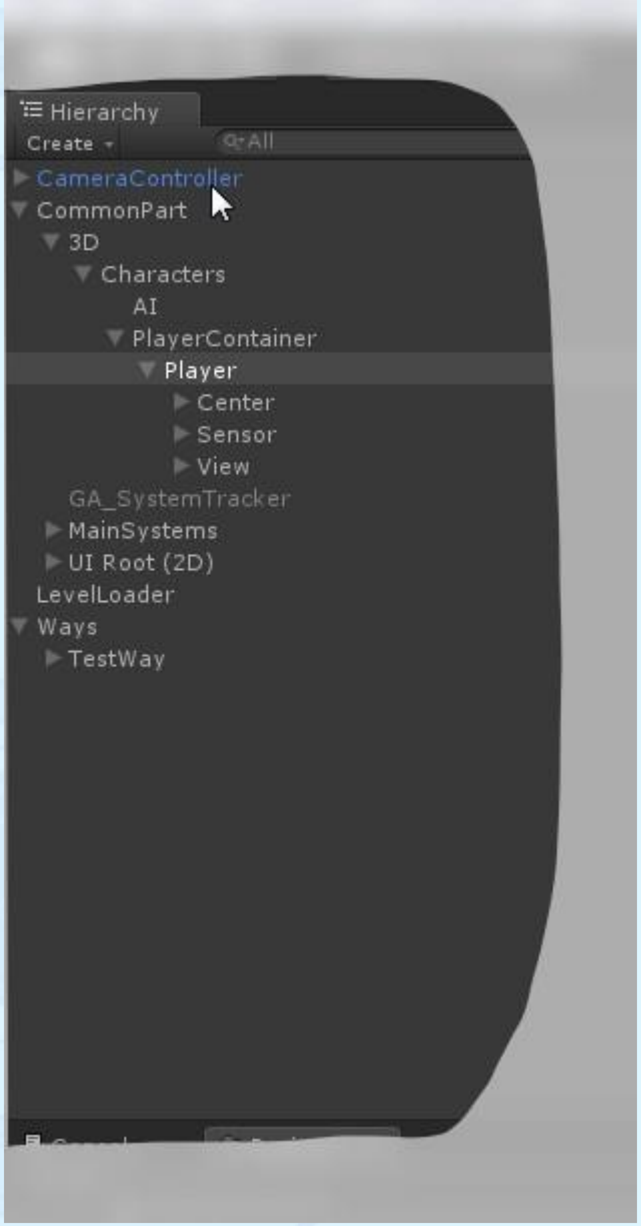
- **ShaderLab**
- **CG**
- **HLSL**

Unity Engine Specifics

The main features of Unity3D:

- Tree hierarchy of objects
- Modular properties of the object
  - Scripts
  - Physics
  - Render
  - Audio
- Standard physic system
- Prefab system

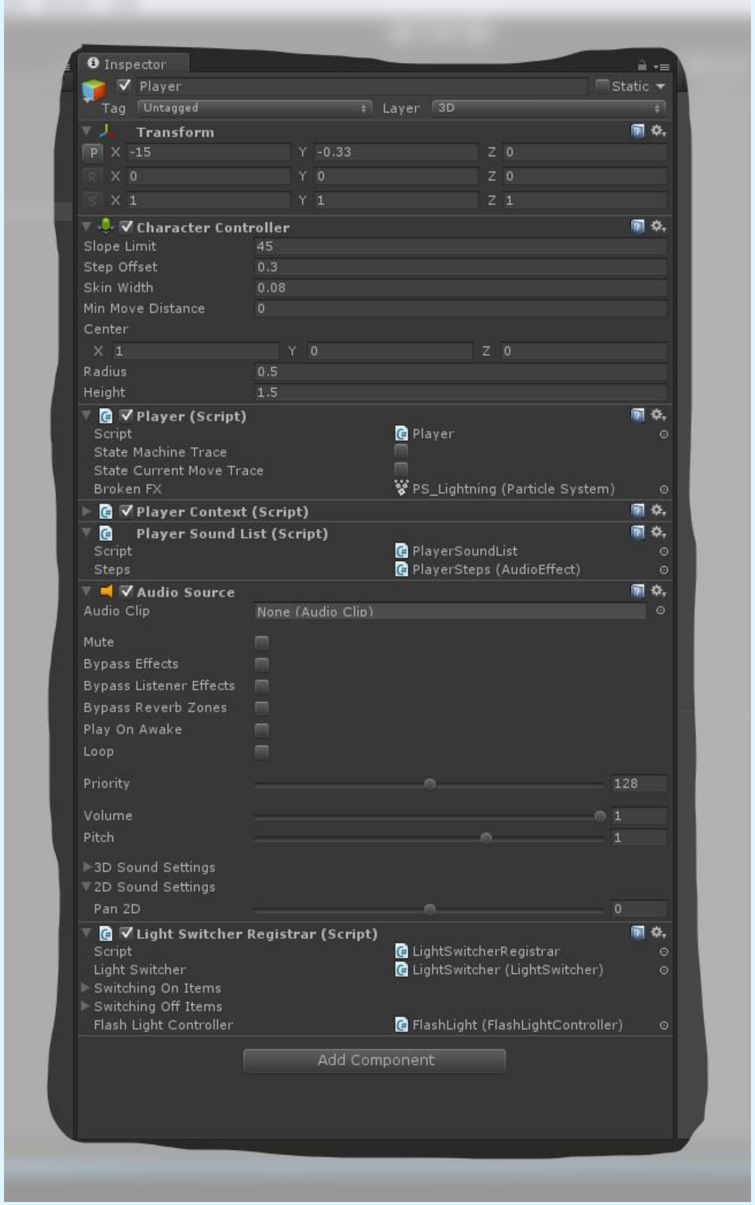
Hierarchy



```
▶ CameraController
▼ CommonPart
  ▼ 3D
    ▼ Characters
        AI
      ▼ PlayerContainer
        ▼ Player
          ▶ Center
          ▶ Sensor
          ▶ View
    GA_SystemTracker
  ▶ MainSystems
  ▶ UI Root (2D)
  LevelLoader
▼ Ways
  ▶ TestWay
```

Example of a hierarchy of objects in the scene. View from the editor.

Components of The Object



Inspector

☑ Player □ Static

Tag Untagged Layer 3D

Transform

P X -15 Y -0.33 Z 0

R X 0 Y 0 Z 0

S X 1 Y 1 Z 1

☑ Character Controller

Slope Limit 45

Step Offset 0.3

Skin Width 0.08

Min Move Distance 0

Center

X 1 Y 0 Z 0

Radius 0.5

Height 1.5

☑ Player (Script)

Script Player

State Machine Trace □

State Current Move Trace □

Broken FX PS_Lightning (Particle System)

☑ Player Context (Script)

Player Sound List (Script)

Script PlayerSoundList

Steps PlayerSteps (AudioEffect)

☑ Audio Source

Audio Clip None (Audio Clip)

Mute □

Bypass Effects □

Bypass Listener Effects □

Bypass Reverb Zones □

Play On Awake □

Loop □

Priority 128

Volume 1

Pitch 1

3D Sound Settings

2D Sound Settings

Pan 2D 0

☑ Light Switcher Registrar (Script)

Script LightSwitcherRegistrar

Light Switcher LightSwitcher (LightSwitcher)

Switching On Items

Switching Off Items

Flash Light Controller FlashLight (FlashLightController)

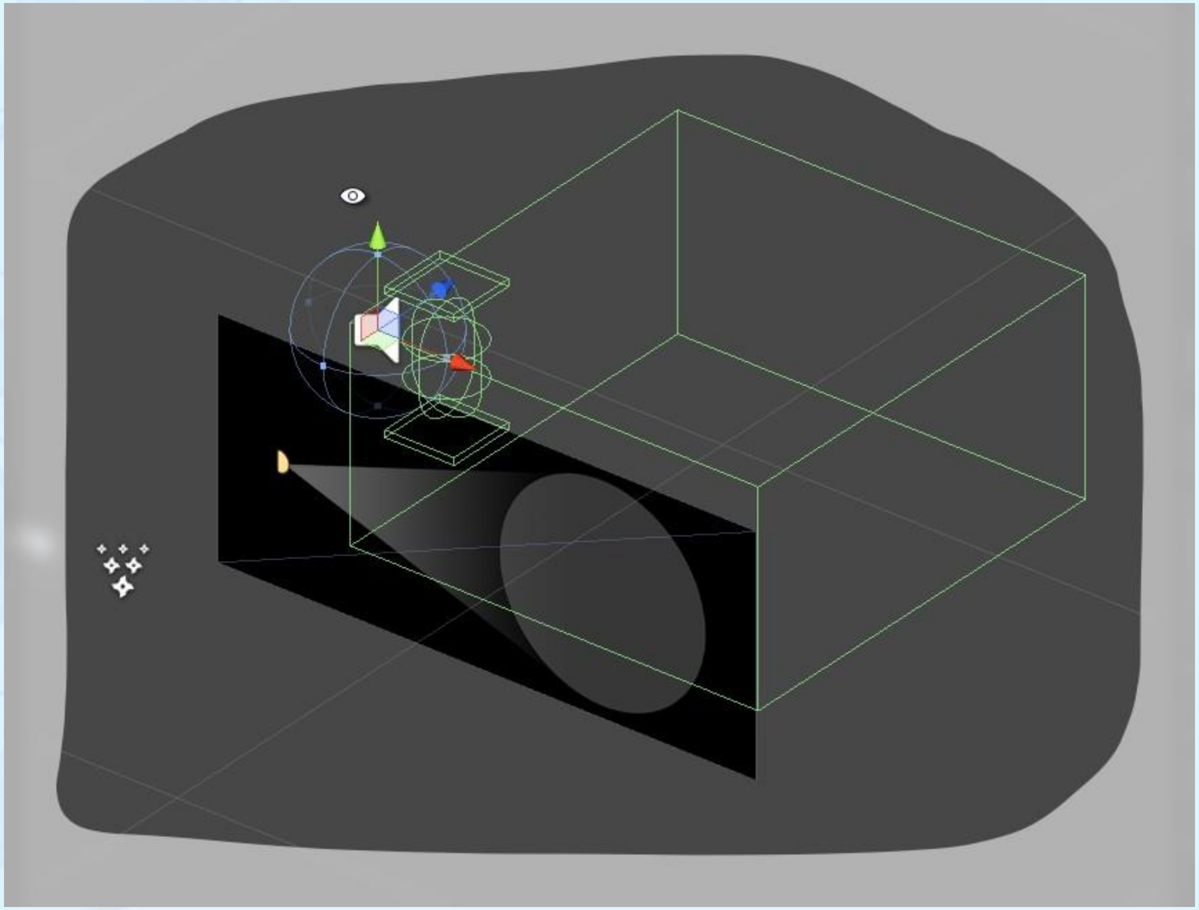
Example of a list of components that are attached to the object.

These components define the behavior and the gaming facility. View from the editor.

Physic System

Implementation of physics in the game engine:

- Collider
- Rigidbody
- Collision / Trigger event system

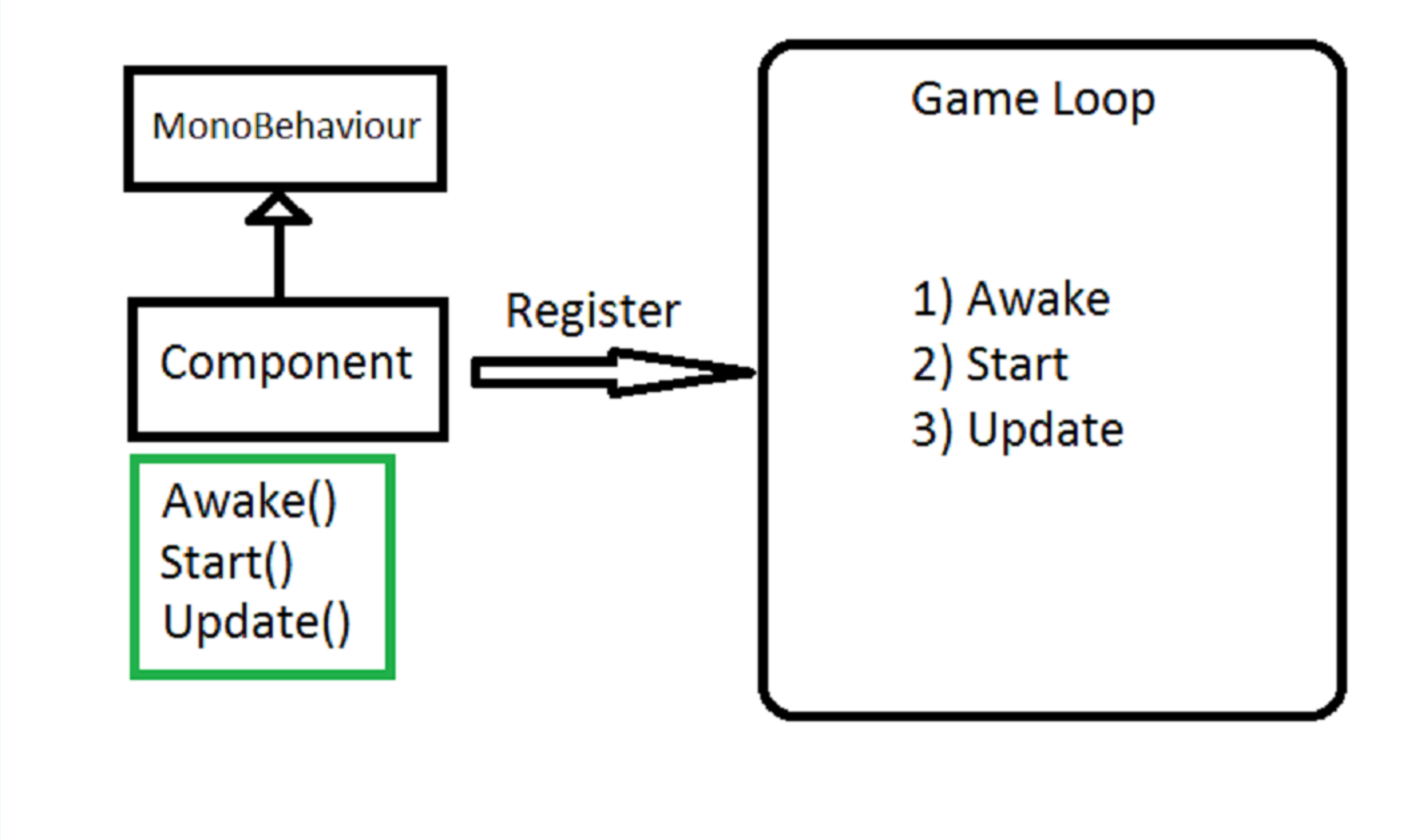


Scripting

All game components (scripts, etc.) are inherited from MonoBehaviour class.

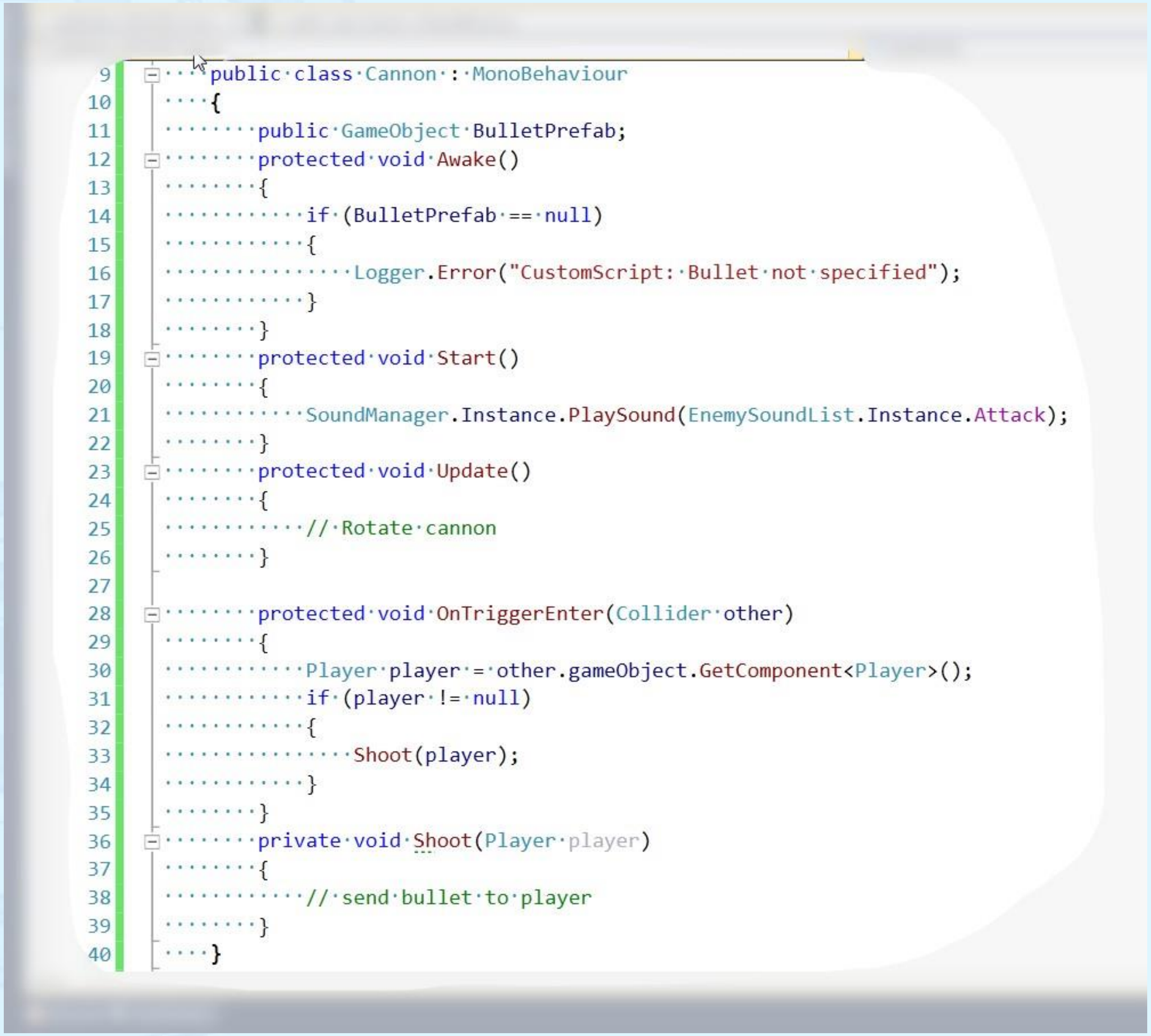
Engine causes certain methods (Awake (), Start () , etc.)

They can override (works through Reflection) and MĀ custom scripts.



Script Example

After all, the main feature of Unity 3D is the easiness of writing. Here is an example of script for gun control:



```
    public class Cannon : MonoBehaviour
    {
        public GameObject BulletPrefab;
        protected void Awake()
        {
            if (BulletPrefab == null)
            {
                Logger.Error("CustomScript: Bullet not specified");
            }
        }
        protected void Start()
        {
            SoundManager.Instance.PlaySound(EnemySoundList.Instance.Attack);
        }
        protected void Update()
        {
            // Rotate cannon
        }

        protected void OnTriggerEnter(Collider other)
        {
            Player player = other.gameObject.GetComponent<Player>();
            if (player != null)
            {
                Shoot(player);
            }
        }
        private void Shoot(Player player)
        {
            // send bullet to player
        }
    }
```