**Particle Effect System**

The Particle System in Unity is a robust particle effect system where you can simulate moving liquids, smoke, clouds, flames, magic spells, and a whole slew of other effects. The system makes it possible to represent effects that are normally difficult to portray, using meshes or Sprites since they often represent effects that are fluid and intangible in nature.

Adding a particle system to our project:

1. From the top menu dropdown, select: **GameObject** > **Effects** > **Particle System**.

2. To add a Particle System to an existing GameObject, select that GameObject, and in the Inspector window, select the **Add Component** button, and type Particle System in the search field.

## **Particle System Properties:**

You will see the Particle System as a game object in the Hierarchy window. In the Scene window, you will observe particle Sprites populate into the Scene randomly in the upward direction.

1. You can select this Particle System either by selecting it within the Scene or using the Hierarchy

The Particle System has some base settings that you can configure these are called Modules

The Main Module which contains the most common particle settings.

* **Duration**: If Looping isn’t checked, this determines how long the Particle will play.
* **Looping**: Determines if the Particle loops or plays only once.
* **Prewarm**: Only used when looping is enabled. The Particle System will act if it's already completed a full cycle on startup.
* **Start Delay**: The delay in seconds before the particle system starts emitting.
* **Start Lifetime**: The initial lifetime in seconds for the particles. The particle is destroyed after this elapsed time.
* **Start Speed**: The initial speed of the particles. The greater the speed of the particles, the more spread out they will be.
* **3D Start Size**: Changes Start Size from a constant to a vector.
* **Start Size**: The initial size of the particles.
* **3D Start Rotation**: Changes Start Rotation from a Constant to a Vector.
* **Start Rotation**: The initial rotation angle of the particles.
* **Flip Rotation**: Setting the value to 1 flips the rotation angle along the horizontal axis.
* **Start Color**: The initial color of the particles.
* **Gravity Modifier**: Scales the Gravity value set in Unity’s Physics Manager window. If it’s set to 0, the gravity will be turned off.
* **Simulation Space**: Move the particles in Local Space along with the Particle System. World Space particles move freely once emitted.
* **Simulation Speed**: Affects the speed of the simulation of the particles.
* **Delta Time**: Changes the Delta Time from Scaled to Non-Scaled. Scaled time is the time passed in relation to the frames, so it will only count while the game is running. Non- Scaled is time passing since the start of the game, useful for playing effects while the game is paused.
* **Scaling Mod**e: Changes the scaling mode from Local, Shape or Hierarchy.
* **Play on Awake**: Starts emitting immediately when enabled. If this is turned off, you will have to manually activate the particle system with a script or Animation system.
* **Emitter Velocity**: Allows you to change the velocity of the emitter from Rigidbody to Transform.
* **Max Particles**: The upper limit of how many particles can be alive at a once for the given Particle System. It will not emit any more particles after the limit has been reached.
* **Auto Random Seed**: Enables the spawning to be auto seeded to give a more random creation of the particles.
* **Stop Action**: Determines if the Particle System will disable or destroy itself when the Particle System is stopped and all particles have died.
* **Culling Mode**: Choose whether to pause the Particle System simulation when particles are offscreen. Culling when offscreen is an efficient strategy when considering performance, especially if there are many particle effects in your scene.
* **Ring Buffer Mode**: Keeps particles alive until they reach the Max Particles count, at which point new particles recycle the oldest ones, instead of removing particles when their lifetimes elapse.

you’ll also be able to access, modify, disable Particle Systems via scripts.

ParticleSystem Part=GetComponent<ParticleSystem>();

Var part=GetComponent<particleSystem> ();

part.Play();

Destroy(gameObject,part.main.duration);