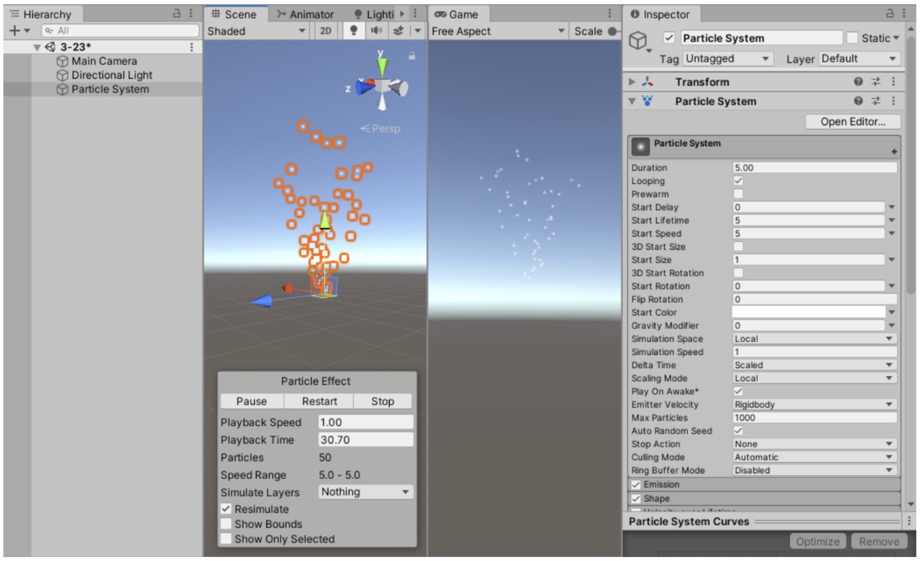
PARTICLE SYSTEM

Create new game object -> Effects -> Particle System



Particle Effect

Pause | Restart | Stop

Playback Speed	1.00
Playback Time	30.70
Particles	50
Speed Range	5.0 - 5.0
Simulate Layers	Nothing

☑ Resimulate
□ Show Bounds
□ Show Only Selected

☑ Particle System □ Static
Tag Untagged Layer Default

Particle System

Duration	5.00
Looping	☑
Prewarm	□
Start Delay	0
Start Lifetime	5
Start Speed	5
3D Start Size	□
Start Size	1
3D Start Rotation	□
Start Rotation	0
Flip Rotation	0
Start Color	
Gravity Modifier	0
Simulation Space	Local
Simulation Speed	1
Delta Time	Scaled
Scaling Mode	Local
Play On Awake*	☑
Emitter Velocity	Rigidbody
Max Particles	1000
Auto Random Seed	☑
Stop Action	None
Culling Mode	Automatic
Ring Buffer Mode	Disabled

☑ Emission
☑ Shape

Figure 01

It should appear as above Figure. The floating box in the scene allows you to control the particle animation, with 'Play' 'Restart' 'Stop'. In the inspector, there are many properties that will allow you to simulate *liquid-like effects*, fire, smoke, magic, weather and so on!

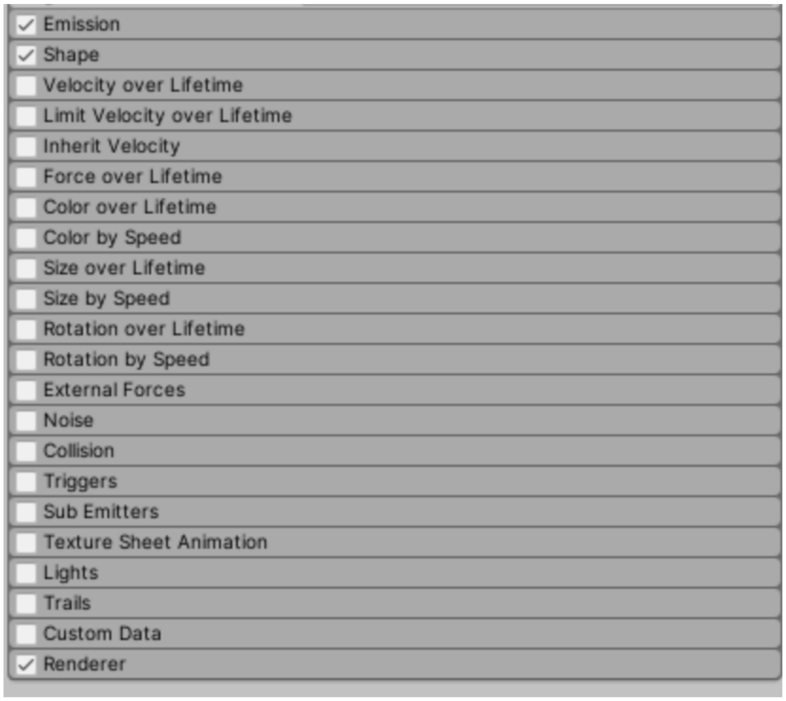
Here are some of the definitions of the essential for you to get started:

- **Duration**: If Looping isn't checked, this determines how long the Particles will play (be created)
- **Looping**: Determines if the Particle loops or plays only once.
- **Start Lifetime**: The initial lifetime in seconds for the particles. The particles get destroyed after this elapsed time, not in consideration of Duration time.
- **Gravity Modifier**: Scales the Gravity value set in Unity's Physics Manager window. If 0, gravity is off, if 1, it falls to the ground (rain, fireworks…)
- **Simulation Space**: Local Space moves the particles along with the Particle System or parent object. World Space moves the particles' transformation of creation in relation to the world position

Here is the full list of definitions for the properties.

The rest of the list has to be checked to activate (Refer to Figure 02).

Some notable ones definitely includes:



☑ Emission
☑ Shape
□ Velocity over Lifetime
□ Limit Velocity over Lifetime
□ Inherit Velocity
□ Force over Lifetime
□ Color over Lifetime
□ Color by Speed
□ Size over Lifetime
□ Size by Speed
□ Rotation over Lifetime
□ Rotation by Speed
□ External Forces
□ Noise
□ Collision
□ Triggers
□ Sub Emitters
□ Texture Sheet Animation
□ Lights
□ Trails
□ Custom Data
☑ Renderer

Figure 02

Emission: Allows you to create bursts effects

Shapes: includes many choices like cone, box, donut, hemisphere, etc. Within, experiment with the angle, especially when you are in 2D instead of 3D, the cone shape appears like a triangle if you shift the angles around.

External Forces: Means that the particles will be affected by events in your scene say if you have wind blowing in your scene, the particles will be blown away by it.

Sub Emitters: Allows you to spawn particles in child emitters from the position of particles in the parent system. Imagine creating a waterfall with particle systems, it wouldn't just be the water falling, there will be the splash at the end and there will be some mist floating from the splashes too, that's where you create child particle effects.

Trails: This feature adds lines/trails behind the particles, think fireworks, dashing and shooting stars!

Renderer: This is important, you can add your desired mesh in here, if you have smoke sprites, any other mesh or objects you can replace the original default sprites here. In 2D, it is important which sorting layers you are on so to be able to see your particles in relation to your player.

Everything with a **Lifetime** behind corresponds to the Start Lifetime duration. For example, you can set what colors the particle may change to through the lifetime of each particle.

In all the **drop down triangles**, it allows you to randomize values. You may add a curve to indicate how you want your particles to orbit, change colors, speed, whatever. Everything can be randomized between two values and or be kept constant.

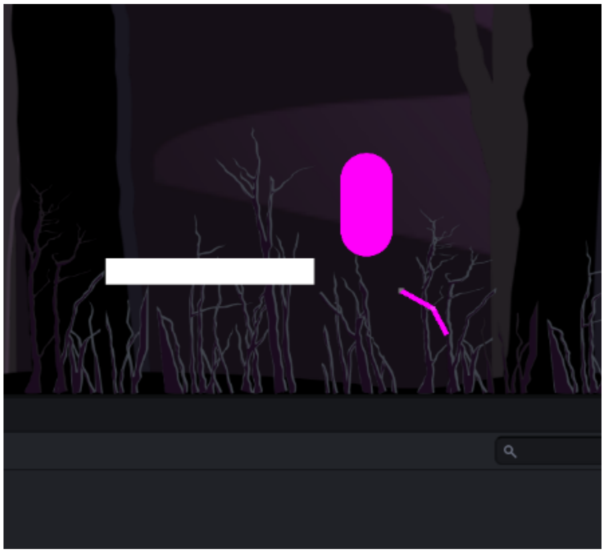


Figure 03

Simplest Implementation (Scripting)

1. Turn off 'Loop'
2. Set your desired particle effects

Then,
In player actions like a dash, jump or flip:

- Declare Particle System as a container for you to fill in the inspector later
- Create a method to play the particle effect
    - particle.Play();
- Call the method within your jump if statements, or dash, or flip etc.
- After saving, drag the desired particle effects into the inspector where you had the container
- Adjust to liking when playtesting, I suggest changing Simulation Space to World instead of Local so it leaves a trail naturally in relation to the world!

Here is the Particle System Scripting Documentation to reference all the methods you could use.

Some fun videos to watch: Making waterfall , Platformer wall jump effects , Crazy Plexus Effect
References: UnityVid Badduck Brackeys TheRealTimeEssentials