HOUR 16

Particle Systems

What You'll Learn in This Hour:

- The basics of particle systems
- How to work with modules
- How to use the Curves Editor

In this hour, you'll learn how to use Unity's particle system. You'll start by learning about particle systems in general and how they work. You'll experiment with the many different particle system modules. You'll wrap up the hour by experimenting with the Unity Curves Editor.

Particle Systems

A particle system is basically an object or component that emits other objects, commonly referred to as *particles*. These particles can be fast, slow, flat, shaped, small, large, and so on. The definition is very generic because these systems can achieve a great variety of effects with the proper settings. They can make jets of fire, plumes of billowing smoke, fireflies, rain, fog, or anything else you can think of. These effects are commonly referred to as *particle effects*.

Particles

A *particle* is a single entity that is emitted by a particle system. Because many particles are generally emitted quickly, it is important for particles to be as efficient as possible. This is the reason that most particles are 2D billboards. A

billboard is a flat image that always faces the camera. This gives the illusion that billboards are 3D, while still giving great performance.

Unity Particle Systems

To create a particle system in a scene, you can either create a particle system object or add a particle system component to an existing object. To create a particle system object, select **GameObject > Effects > Particle System**. To add a particle system component to an existing object, select the object and click **Add Component > Effects > Particle System**.

▼ TRY IT YOURSELF

Creating a Particle System

Follow these steps to create a particle system object in a scene:

1. Create a new project or scene.
2. Add a particle system by selecting **GameObject > Effects > Particle System**.
3. Notice how the particle system emits white particles in the Scene view (see Figure 16.1). This is the basic particle system. Try rotating and scaling the particle system to see how it reacts.

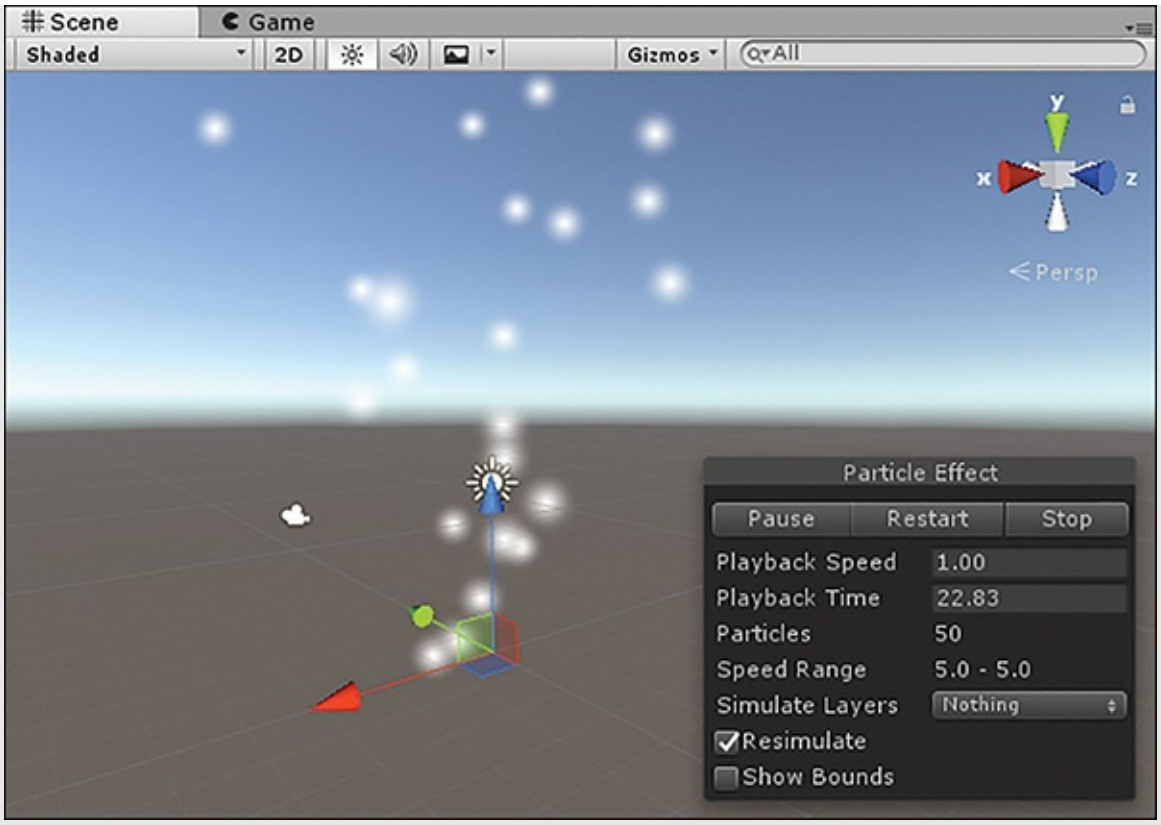

FIGURE 16.1
The basic particle system.

NOTE

Custom Particles

By default, the particles in Unity are small white spheres that fade into transparency. This is a really useful generic particle, but it can only take you so far. Sometimes you want something more specific (to make fire, for example). If you want, you can make your own particles out of any 2D image to create effects that exactly suit your needs.

Particle System Controls

You might have noticed that when you added a particle system to your scene, it began emitting particles in the Scene view. You may also have noticed that particle effect controls appeared (see Figure 16.2). These controls allow you to pause, stop, and restart the particle animation in a scene. This can be very

helpful when tweaking the behavioral components of a particle system.

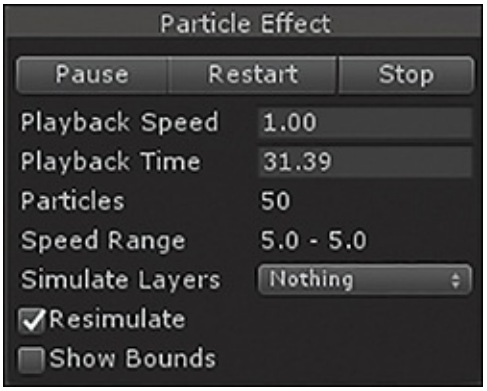

FIGURE 16.2
The particle effect control.

These controls also allow you to speed up the playback and tell you how long an effect has been playing. This can be very useful when testing duration effects. Note that the controls show the playback speed and playback time only when the game is stopped.

NOTE

Particle Effects

To create complex and visually appealing effects, you can have several particle systems work together (a smoke and a fire system, for example). Multiple particle systems working together creates a *particle effect*. In Unity, creating a particle effect is achieved by nesting particle systems together. One particle system can be the child of another, or they can both be children of a different object. The result of a particle effect in Unity is that the effects are treated as one system, and you can use the particle effect controls to control the entire particle effect as one unit.

Particle System Modules

At its root, a particle system is just a point in space that emits particle objects. How particles look and behave and the effects they cause are all determined by modules. *Modules* are various properties that define some form of behavior. In Unity's particle system, modules are an integrated and essential component. This section lists the modules and explains briefly what each one does.

Note that with the exception of the default module (covered first), all modules can be turned on and off. To turn modules on or off, put a check mark by the module's name. To hide or show modules, click the plus sign (+) next to Particle System (see Figure 16.3). You can also click the name of a particle system in the list to toggle its visibility. By default, all modules are visible, and only the Emission, Shape, and Renderer modules are enabled. To expand a module, simply click its title.

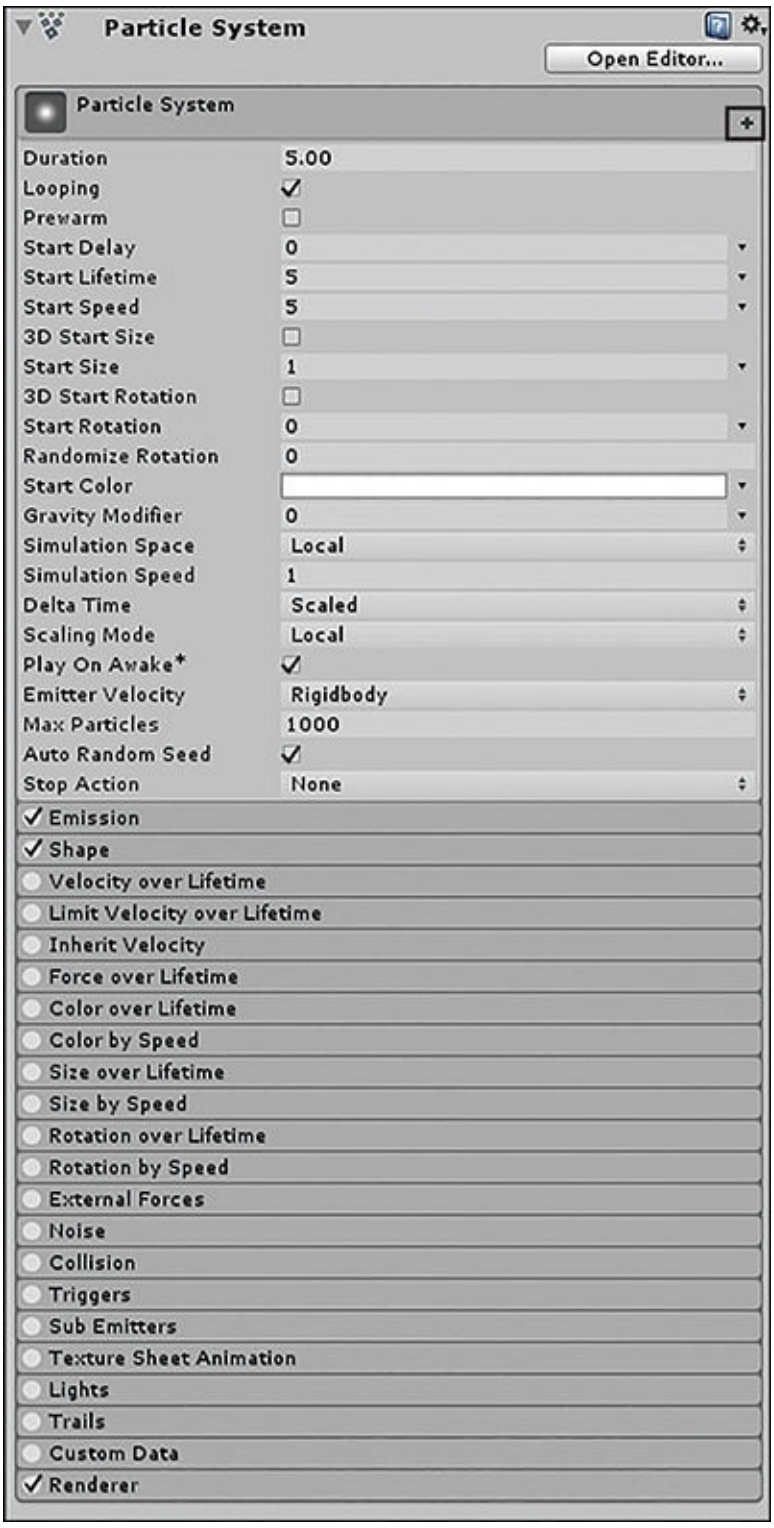

FIGURE 16.3
Showing all modules.

NOTE

Properties Overview, Briefly

Several modules have properties that are either self-explanatory (like the length and width property of a rectangle) or have been covered previously. For the sake of simplicity (and to prevent this hour from being 50 pages), these properties are not described in this section. So if you see more properties on your screen than are covered in this text, don’t worry; that is intentional.

NOTE

Constant, Curve, Random

A value curve allows you to change the value of a property over the lifetime of a particle system or an individual particle. You can tell that a property can use curves because there is a downward-facing arrow next to the value. The options you see are Constant, Curve, Random Between Two Constants, and Random Between Two Curves. In this section, the value is always Constant. Later in this hour, you'll get a chance to explore the Curves Editor in detail.

Default Module

The default module is simply labeled Particle System. This module contains all the specific information that every particle system requires. Table 16.1 describes the properties of the default module.

TABLE 16.1 Default Module Properties

Property	Description
Duration	Specifies how long, in seconds, the particle system runs.
Looping	Determines whether the particle system starts over when the Duration value has been reached.
Prewarm	Specifies whether the particle system should start as if it had already emitted particles from a previous cycle.
Start Delay	Specifies how long, in seconds, the system waits before emitting particles.
Start Lifetime	Specifies how long, in seconds, each particle lives.

Start Speed	Determines the initial speed of particles.
Start Size	Specifies the initial size of a particle. If you check the property 3D Start Size, you can provide different size values along the three axes. Otherwise, it is one size fits all.
Start Rotation	Specifies the initial rotation of particles. If you check the property 3D Start Rotation, you can specify a rotation value about the three axes. Otherwise, only one value is used.
Randomize Rotation	Causes some particles to rotate in the opposite direction.
Start Color	Specifies the color of emitted particles.
Gravity Modifier	Specifies how much of the world's gravity is applied to the particles.
Simulation Space	Determines whether the coordinate and axes values are based on the world coordinate system or the local coordinate system of a parent game object.
Simulation Speed	Allows fine-tuning of the speed for an entire particle system.
Delta Time	Determines whether the timing of a particle system is based on scaled time or unscaled time.
Scaling Mode	Determines whether the scale is based on the game object, the object's parent, or the shape of the emitter.
Play on Wake	Determines whether the particle system begins emitting particles immediately when created.
Emitter Velocity	Allows you to choose whether velocity is calculated from the object's transform or its rigidbody (if it has one).
Max Particles	Specifies the total number of particles that can exist for a system at a time. If this number is reached, the system ceases emitting until some particles die.
Auto Random Seed	Determines whether the particle system will look different each time it is played.
Stop	Allows you to specify what happens when the particle system

Action	finishes. For instance, you could disable or destroy the game object or request a script callback to occur.

Emission Module

The Emission module is used to determine the rate at which particles are emitted. Using this module, you can dictate whether particles stream at a constant rate, in bursts, or somewhere in between. Table 16.2 describes the Emission module properties.

TABLE 16.2 Emission Module Properties

Property	Description
Rate over Time	Specifies the number of particles emitted over time.
Rate over Distance	Specifies the number of particles emitted over distance.
Bursts	Specifies bursts of particles at specific time intervals. You can create a burst by clicking the plus sign (+) and remove a burst by clicking the minus sign (−) (see Figure 16.4).

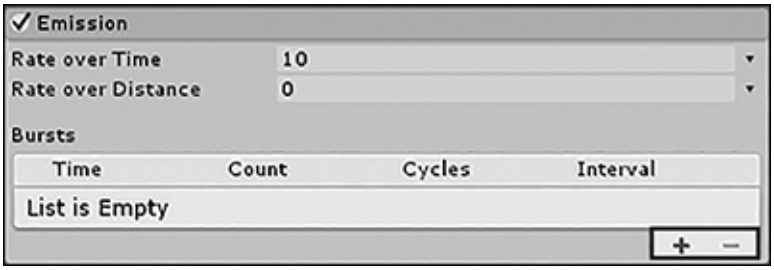

FIGURE 16.4
The Emission module.

Shape Module

Just as its name implies, the Shape module determines the shape formed by the emitted particles. The shape options are Sphere, Hemisphere, Cone, Donut, Box, Mesh, Mesh Renderer, Skinned Mesh Renderer, Circle, and Edge (whew!). In addition, each shape has a set of properties used to define it, such as radius for

cones and spheres. These properties are fairly self-explanatory, so they are not covered here.

Velocity over Lifetime Module

The Velocity over Lifetime module directly animates each particle by applying an x, y, and z axis velocity to it. Note that this is a velocity change of each particle over the lifetime of the particle, not over the lifetime of the particle system. Table 16.3 describes the properties of the Velocity over Lifetime module.

TABLE 16.3 Velocity over Lifetime Module Properties

Property	Description
XYZ	Specifies the velocity applied to each particle. This can be a constant, curve, or random number between a constant or curve.
Space	Dictates whether the velocity is added based on local or world space.
Speed Modifier	Allows you to scale the individual velocities all at once.

Limit Velocity over Lifetime Module

This long-named module can be used to dampen or clamp the velocity of a particle. Basically, it prevents, or slows down, particles that exceed a threshold speed on one or all of the axes. Table 16.4 describes the properties for the Limit Velocity over Lifetime module.

TABLE 16.4 Limit Velocity over Lifetime Module Properties

Property	Description
Separate Axis	If unchecked, uses the same value for each axis. If checked, uses speed properties for each axis as well as a property for local or world space.
Speed	Specifies the threshold speed for each or all axes.
Dampen	Specifies the value, between 0 and 1, by which a particle will be slowed if it exceeds the threshold, as determined by the Speed

	property. A value of 0 does not slow a particle at all, but a value of 1 slows the particle 100%.
Drag	Specifies the amount of linear drag to apply to particles.
Multiply by Size	Determines whether larger particles will be slowed more.
Multiply by Velocity	Determines whether faster particles will be slowed more.

Inherit Velocity Module

The Inherit Velocity Module is very simple and determines how much, if any, of the velocity of the emitter should be applied to the particle. The first property, Mode, specifies if only the initial velocity is applied or if the particle should continue receiving the velocity of the emitter. Finally, the Multiplier property determines the proportion of velocity to apply.

Force over Lifetime Module

The Force over Lifetime module is similar to the Velocity over Lifetime module. The difference is that this module applies a force, not a velocity, to each particle. This means the particle will continue to accelerate in the specified direction. This module also allows you to randomize the force each frame, as opposed to all up front.

Color over Lifetime Module

The Color over Lifetime module allows you to change the color of a particle as time passes. This is useful for creating effects like sparks, which start out bright orange and end a dark red before disappearing. To use this module, you must specify a gradient of color. You can also specify two gradients and have Unity randomly pick a color between them. Gradients can be edited using Unity's Gradient Editor (see Figure 16.5).

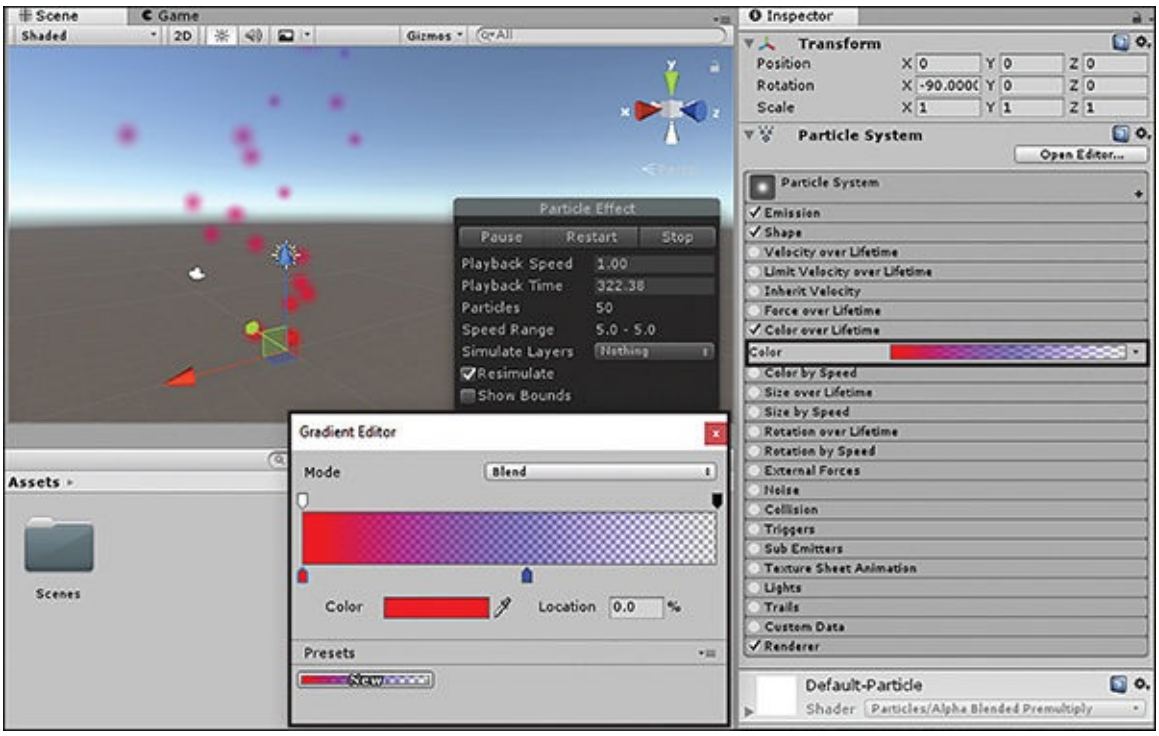

FIGURE 16.5
The Gradient Editor.

Note that the color of the gradient is multiplied by the Start Color property of the default module. Therefore, if your start color is black, the Color over Lifetime module will have no effect.

Color by Speed Module

The Color by Speed module allows you to change the color of a particle based on its speed. Table 16.5 describes the properties of the Color by Speed module.

TABLE 16.5 Color by Speed Module Properties

Property	Description
Color	Specifies a gradient (or two gradients for random colors) that is used to dictate the color of the particle.
Speed Range	Specifies the minimum and maximum speed values that are mapped to the color gradient. Particles going the minimum speed are mapped to the left side of the gradient, and colors at the maximum speed (or beyond) are mapped to the right side of the gradient.

Size over Lifetime Module

The Size over Lifetime module allows you to specify a change in the size of a particle. The size value must be a curve, and it dictates whether the particle grows or shrinks as time elapses.

Size by Speed Module

Much like the Color by Speed module, the Size by Speed module changes the size of a particle based on its speed between minimum and maximum values.

Rotation over Lifetime Module

The Rotation over Lifetime module allows you to specify a rotation over the life of a particle. Note that the rotation is of the particle itself, not a curve in the world coordinate system. This means that if your particle is a plain circle, you will not be able to see the rotation. If the particle has some detail, however, you will notice it spin. The values for the rotation can be given as a constant, curve, or random number.

Rotation by Speed Module

The Rotation by Speed module is the same as the Rotation over Lifetime module except that it changes values based on the speed of the particle. Rotation changes based on minimum and maximum speed values.

External Forces Module

The External Forces module allows you to apply a multiplier to any forces that exist outside the particle system. A good example of this is any wind forces that exist in a scene. The Multiplier property scales the forces either up or down, depending on its value.

Noise Module

The Noise module is a relatively new module for Unity’s particle system. This module allows you to apply some randomization to the movement of a particle (think lightning bolts, for example). It accomplishes this by generating a *Perlin noise* image to use as a lookup table. You can see the noise being used in the

Preview window of the module. Table 16.6 lists the properties of the Noise module.

TABLE 16.6 Noise Module Properties

Property	Description
Separate Axes	Dictates whether the noise will be applied equally to all axes or whether different values will be derived for each.
Strength	Defines how strong the noise effect is on a particle over its lifetime. Higher values make particles move faster and farther.
Frequency	Specifies how often particles change their direction of travel. Low values create soft, smooth noise, and high values create rapidly changing noise.
Scroll Speed	Causes the noise field to move over time to cause more unpredictable and erratic particle movement.
Damping	Determines whether strength is proportional to frequency.
Octaves	Determines how many overlapping layers of noise are applied. Higher numbers give more rich and interesting noise but at the cost of performance.
Octave Multiplier	Reduces the strength of each additional noise layer.
Octave Scale	Adjusts the frequency for each additional noise layer.
Quality	Allows you to adjust the quality of the noise to regain some performance.
Remap and Remap Curve	Allows you to remap the final value of the noise into something else. You can use a curve to specify which noise values should be translated into other values.
Position, Rotation, and Size Amount	Control how much the noise affects position, rotation, and scale of a particle.

Collision Module

The Collision module allows you to set up collisions for particles. This is useful

for all sorts of collision effects, like fire rolling off a wall or rain hitting the ground. You can set the collision to work with predetermined planes (Planes mode is the most efficient) or with objects in the scene (World mode slows performance). The Collision module has some common properties and some unique properties, depending on the collision type chosen. Table 16.7 describes the common properties of the Collision module. Tables 16.8 and 16.9 describe the properties of Planes mode and World mode, respectively.

TABLE 16.7 Common Collision Module Properties

Property	Description
Planes and World	Dictates the type of collision used. Planes collide off predetermined planes. World mode collides off any object in a scene.
Dampen	Determines the amount a particle is slowed when it collides. Values range from 0 to 1.
Bounce	Determines what fraction of the component of velocity is kept. Unlike Dampen, this only affects the axes the particle bounces on. Values range between 0 and 1.
Lifetime Loss	Determines how much life of a particle is lost on collision. Values range from 0 to 1.
Min Kill Speed	Specifies the minimum speed of a particle before it is killed by collision.
Max Kill Speed	Specifies the speed above which particles that collide will be removed from the system.
Radius Scale	Adjusts the radius of the particle collision spheres so it more closely fits the visual edges of the particle graphic.
Send Collision Messages	Determines whether collision messages are sent to objects that collide with particles.
Visualize Bounds	Renders the collision bounds of each particle as a wireframe shape in the Scene view.

TABLE 16.8 Planes Mode Properties

Property	Description

Planes	Determines where the particles can collide. The y axis of the transforms provided determines the rotation of the plane.
Visualization	Determines how the planes are drawn in the Scene view. They can either be solid or grid.
Scale Plane	Resizes the visualization of the planes.

TABLE 16.9 World Mode Properties

Property	Description
Collision Mode	Specifies whether to use 2D or 3D.
Collision Quality	Indicates the quality of the world collision. The values are High, Medium, and Low. Obviously, High is the most CPU intensive and most accurate, and Low is the least.
Collides With	Determines which layers the particles collide with. This is set to Everything by default.
Max Collision Shapes	Specifies the number of shapes that can be considered for collision. Excess shapes are ignored, and terrains take priority.
Enabled Dynamic Colliders	Determines whether particles can collide with non-static (non-kinematic) colliders.
Collider Force and Multiply Options	Allow particles to apply force to objects they collide with. This allows particles to push objects. Additional check boxes allow more force to be applied based on angle of collision, speed of particle, and size of particle.

▼ TRY IT YOURSELF

Making Particles Collide

In this exercise, you'll set up collision with a particle system. This exercise uses both Planes and World collision modes. Follow these steps:

1. Create a new project or scene. Add a sphere to the scene and place it at (0, 5, 0) with a scale of (3, 3, 3). Give the sphere a Rigidbody component.

2. Add a particle system to the scene and place it at (0, 0, 0). Under the Emission module in the Inspector, set Rate over Time to 100.

3. Enable the Collision module by clicking the circle next to its name. Set Type to World and set Collider Force to 20 (see Figure 16.6). Notice how the particles are already bouncing off the sphere.

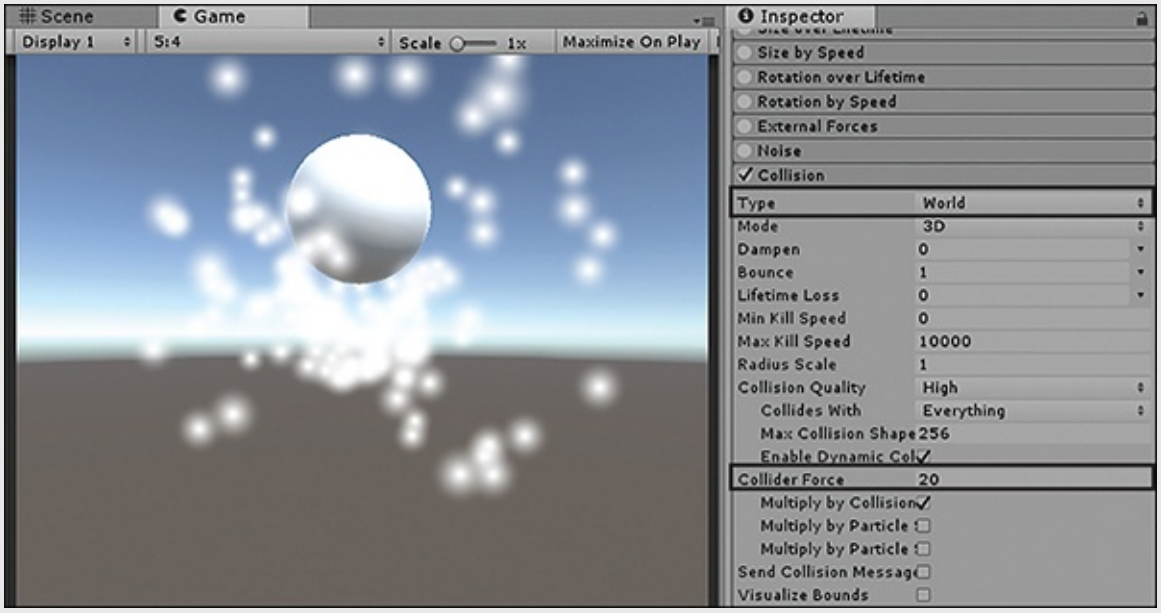

FIGURE 16.6
Adding a plane transform.

4. Enter Play mode and notice how the sphere is buoyed by the particles.

5. Experiment with the various emission, shape, and collision settings. See how long you can keep the sphere in the air.

TIP

Emitter Versus Particle Settings

Some modules modify the emitter, while others modify the particles. You may wonder why there is a Color property and also a Color over Lifetime property. One controls the color over the life of the emitter, while the other makes particles change color over their life.

Triggers Module

The Triggers module "triggers" a response to a particle entering a collider volume. You can respond to the event of a particle being inside a volume, outside a volume, entering a volume, and exiting a volume. When any of these happens, you can ignore the event, destroy the particle, or call a method in some code and define custom behavior.

Sub Emitter Module

The Sub Emitter module is an incredibly powerful module that enables you to spawn a new particle system at certain events for each particle of the current system. You can create a new particle system every time a particle is created, dies, or collides. In doing this, you can generate complex and intricate effects, such as fireworks. This module has three properties: Birth, Death, and Collision. Each of these properties holds zero or more particle systems to be created on the respective events.

Texture Sheet Module

The Texture Sheet module allows you to change the texture coordinates used for a particle over the life of the particle. In essence, this means you can put several textures for a particle in a single image and then switch between them during the life of a particle (as you did with the sprite animation in Hour 15, "Game 3: *Captain Blaster*"). Table 16.10 describes the properties of the Texture Sheet module.

TABLE 16.10 Texture Sheet Module Properties

Property	Description
Mode	Determines whether you will use the tradition texture sheet method or provide individual sprite to cycle through.
Tiles	Specifies the number of tiles the texture is divided into in the x (horizontal) and y (vertical) directions.
Animation	Determines whether the whole image contains textures for the particle or whether only a single row does.
Cycles	Specifies the speed of the animation.
Flip U and Flip V	Determine whether some of the particles will be flipped horizontally or vertically.

Lights Module

The Lights module allows a portion of the particles to also contain a point light. This allows the particle systems to add illumination to a scene (think of a torch effect, for example). Most of the properties for this module are self-explanatory, but a very important one is the Ratio property. A value of 0 means that no particles will have a light, and a value of 1 means that all will have a light. It is important to note that adding lights to too many particles will greatly slow down your scene, so use this module sparingly.

Trails Module

The Trails module allows particles to leave a trail behind them. Using the module is a great way to create streaked effects, such as fireworks or lightning bolts. Almost all properties of this module have been covered in other modules or are self-explanatory. The only property that needs to be covered here is the Minimum Vertex Distance property. This property determines how far a particle needs to travel before its trail gets a new vertex. Lower numbers make for smoother trails but are also less efficient.

Custom Data Module

The Custom Data module is really beyond the scope of this book because it performs a very technical and very powerful function. Essentially, this module allows you to pass data from the particle system into a custom shader you've written to utilize that data.

Renderer Module

The Renderer module dictates how the particles are actually drawn. It is here that you can specify the texture used for the particles and their other drawing properties. Table 16.11 describes some of the properties of the Renderer module.

TABLE 16.11 Renderer Module Properties

Property	Description
Render Mode	Determines how the particles are actually drawn. The modes are Billboard, Stretched Billboard, Horizontal Billboard, Vertical Billboard, and Mesh. All the billboard modes cause the particles to align with either the camera or two out of three axes. The Mesh

	align with either the camera or two out of three axes. The Mesh mode causes the particles to be drawn in 3D, as determined by a mesh.
Normal Direction	Dictates how much the particles face the camera. A value of 1 causes the particles to look directly at the camera.
Material and Trail Material	Specify the material used to draw the particle and the particle's trail, respectively.
Sort Mode	Specifies the order in which particles are drawn. Can be None, By Distance, Youngest First, or Oldest First.
Sorting Fudge	Determines the order in which the particle system is drawn. The lower the value, the more likely the system is to be drawn on top of other particles.
Min Particle Size and Max Particle Size	Specify the smallest or largest particle size (regardless of other settings), expressed as a fraction of the viewport size. Note that this setting is only applied when Rendering Mode is set to Billboard.
Render Alignment	Determines whether particles are aligned with the camera, the world, their own transform, or the direct position of the camera (useful for VR).
Pivot	Defines a custom pivot point for particles.
Masking	Determines whether particles interact with a 2D mask.
Custom Vertex Stream	In conjunction with the Custom Data module, determines which particle system properties are passed into custom vertex shaders.
Cast Shadows	Determines whether particles cast shadows.
Receive Shadows	Determines whether particles receive shadows.
Motion Vectors	Determines whether particles use motion vectors for rendering. Leave the default setting for now.
Sorting	Allow particles to be sorted using the sprite sorting layer system.

Layer and Order in Layer	
Light and Reflection Probes	Allow particles to work with light and reflection probes (if they exist).

The Curves Editor

Several values in the various modules listed previously can be set as Constant or Curve. The Constant option is fairly self-explanatory: You give it a value, and it is that value. What if you want that value to change over a period of time, though? That is where the curve system comes in very handy. This feature gives you a very fine level of control over how a value behaves. You can see the Curves Editor at the bottom of the Inspector view (see Figure 16.7). You may need to drag it up by the horizontal handle.

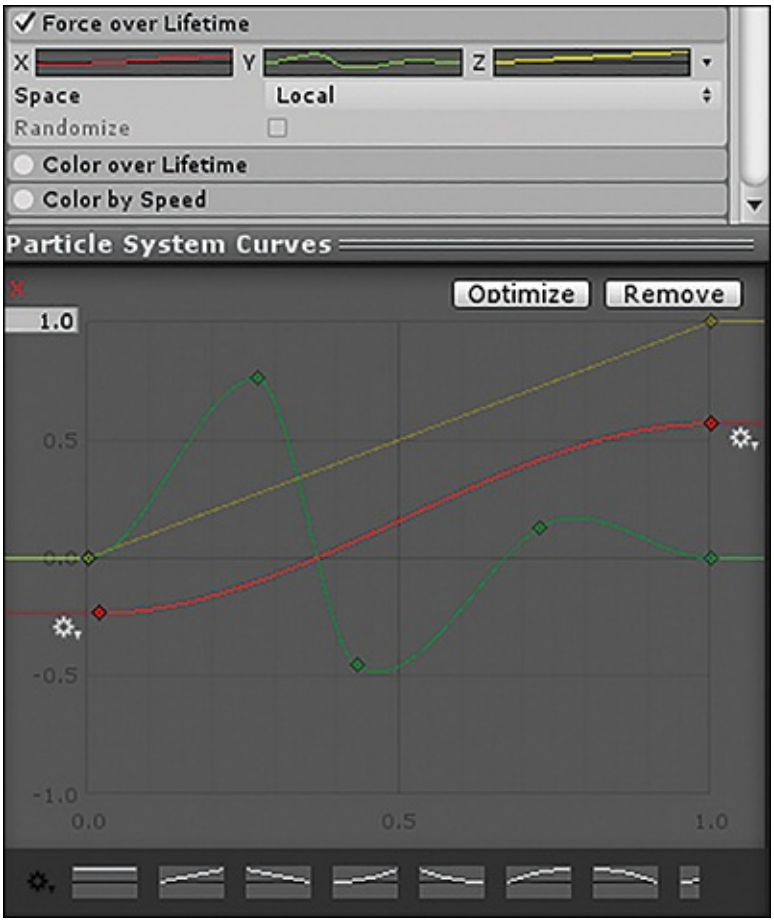

FIGURE 16.7

The Curves Editor in the Inspector.

The title of the curve is whatever value you are determining. In Figure 16.7, the value is for the force applied along the x axis in the Force over Lifetime module. The range dictates the minimum and maximum values available. This can be changed to allow for a greater or lesser range. The curve is the values themselves over a given course of time, and the presets are generic shapes that you can give to the curve.

The curve is movable at any of the key points. These key points are show as visible points along the curve. By default, there are only two key points: one at the beginning and one at the end. You can add a new key point anywhere on the curve by right-clicking it and choosing **Add Key Point** or by double-clicking the curve.

You can get an even larger Curves Editor by clicking the **Open Editor** button at the top right of the Particle System component or by right-clicking the title bar of the Curves Editor.

TRY IT YOURSELF

Using the Curves Editor

To get familiar with the Curves Editor, in this exercise, you'll change the size of the particles emitted over the duration of one cycle of the particle system. Follow these steps: **1.** Create a new project or scene. Add a particle system and position it at (0, 0, 0).

2. Click the drop-down arrow next to the Start Size property and choose **Curve**.

3. Change the range of the curve from 1.0 to 2.0 by changing the value in the top left of the Curves Editor.

4. Right-click the curve at about the midpoint and add a key. Now drag the start and end points of the curve down to 0 (see Figure 16.8). Notice how the particles emitted change in size over the 5-second cycle of the particle system.

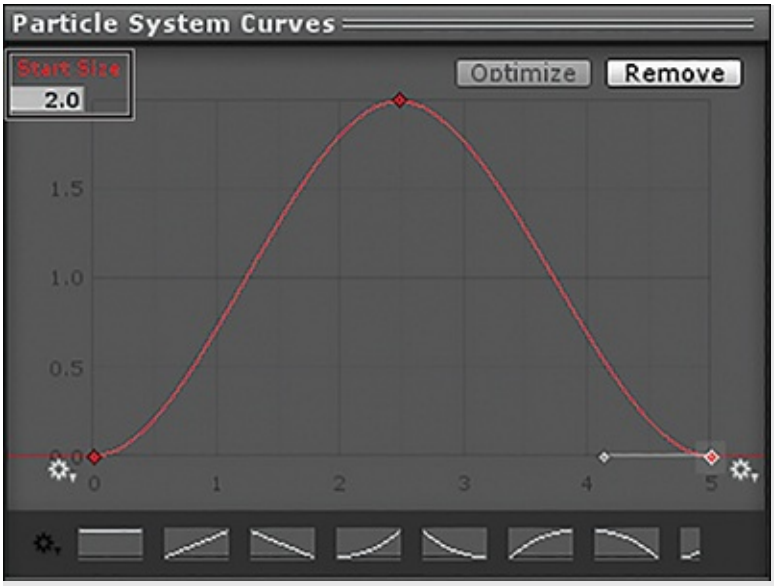

FIGURE 16.8
Start Size curve settings.

Summary

In this hour, you have learned the basics of particles and particle systems in Unity. You have also learned about the many modules that make up the Unity particle system. You wrapped up the hour by looking at the functionality of the Curves Editor.

Q&A

Q. Are particle systems inefficient?

A. They can be, depending on the settings you give them. A good rule of thumb is to use a particle system only if it provides some value to you. Particle systems can be great visually, but don’t overdo it.

Workshop

Take some time to work through the questions here to ensure that you have a firm grasp of the material.

Quiz

1. What is the term for a 2D image that always faces the camera?
2. How do you open a larger particle effect editor window?
3. Which module controls how a particle is drawn?
4. True or False: The Curves Editor is used for creating curves that change values over time.

Answers

1. Billboard 2. Click the Open Editor button at the top of the Particle System component in the Inspector.
3. Renderer module 4. True

Exercise

In this exercise, you'll experiment with some exciting particle effects provided as standard packages with Unity. This exercise is a chance both to play around with existing effects and to create your own. There is no correct solution for you to look at. Just follow the steps here and use your imagination:

1. Import the particle effects package by selecting Assets > Import Package > ParticleSystems. Be sure to leave all assets checked and click **Import**.
2. Navigate to Assets\Standard Assets\ParticleSystems\Prefabs. Click and drag the FireComplex and Smoke prefabs into the Hierarchy view. Experiment with the positioning and settings of these effects. Click **Play** to see the effects.
3. Continue experimenting with the rest of the provided particle effects. (Be sure to check out the Explosion and Fireworks effects at least.) **4.** Now that you have seen what is possible, see what you can create yourself. Try out the various modules and try to come up with your own custom effects.