HOUR 13

2D Tilemap

What You'll Learn in This Hour:

- What tilemaps are
- How to create palettes
- How to create and place tiles
- How to add physics to tilemaps

You learned to created 3D worlds using Unity's terrain system in Hour 4, "Terrain and Environments." Now it is time to extend that knowledge into 2D games, using Unity's new 2D Tilemap system. Using this system, you can create worlds quickly and easily for more interesting and compelling game experiences. In this hour you'll start by learning about tilemaps. From there you'll examine palettes, which you use to hold your tiles. Finally, you'll create tiles and paint them onto your tilemaps before adding collision to make your tilemaps interactable.

The Basics of Tilemaps

As the name implies, a *tilemap* is simply a "map" of tiles (in the same way old-school bitmaps are maps of bits). Tilemaps sit on a *grid* object that defines the tile sizes and spacing common to all tilemaps. *Tiles* are individual sprite elements used to draw a world. These tiles are placed on a *palette*, which is then used to draw the tiles onto the tilemap. It may seem like a lot, but it is basically a "paint, palette, brush, canvas" type of setup, and once you begin using it, it will seem very natural.

If it isn't already painfully obvious, it should be noted that tilemaps are meant to be used for 2D games. While they could technically be used for 3D, using them that way would not be very effective. It is also a good idea to use sprite sheets in conjunction with tilemaps. You can create a sheet of the various environment parts and then easily convert them into tiles.

Creating a Tilemap

You can have as many tilemaps in a scene as you like, and you will often find yourself creating several of them and layering them together. This method enables you to set up background, midground, and foreground tilemaps for things like parallax effects. To create a tilemap in a scene, you can select **GameObject > 2D Object > Tilemap**. Unity adds two game objects, named Grid and Tilemap, to your scene (see Figure 13.1).

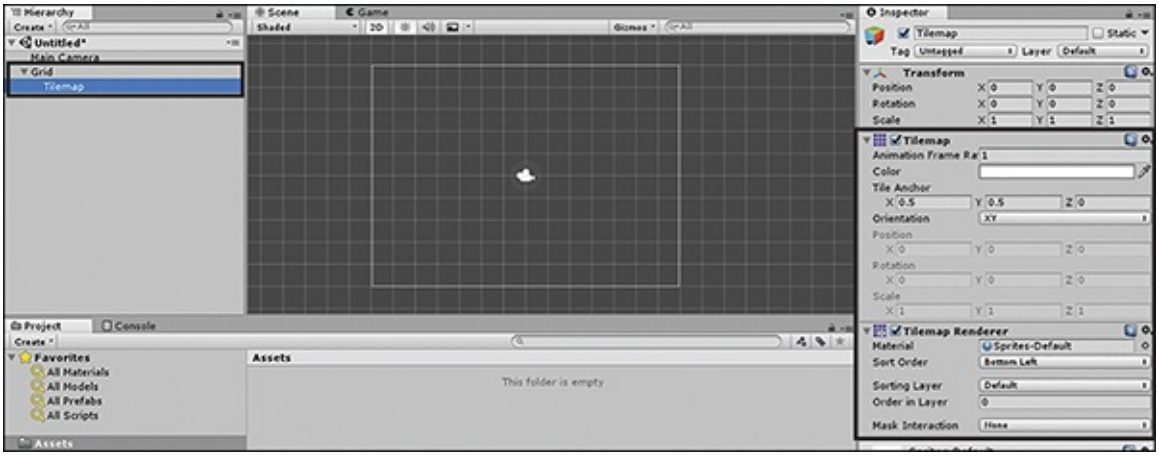

FIGURE 13.1
Adding a tilemap.

The Tilemap game object has two components worth noting: Tilemap and Tilemap Renderer. The Tilemap component has to do with tile placement, anchor positions, and overall color tinting. The Tilemap Renderer allows you to specify sorting orders so that you can ensure that your tilemaps draw in the correct order.

▼ TRY IT YOURSELF

Adding a Tilemap to a Scene

In this exercise, you will add a tilemap to a scene. Be sure to save this scene because you will be using it more later in this hour. Follow these

scene because you will be using it more later in this hour. Follow these steps:

1. Create a new 2D project. Create a new folder named **Scenes** and save your scene into it.
2. Add a tilemap to your scene by selecting **GameObject > 2D Object > Tilemap**.
3. Eventually, this tilemap will be the background of your scene, so rename the new Tilemap game object **Background**.
4. To create another tilemap without also creating another grid, right-click the Grid game object in the Hierarchy window and select **2D Object > Tilemap** (see Figure 13.2). Name the new tilemap **Platforms**.

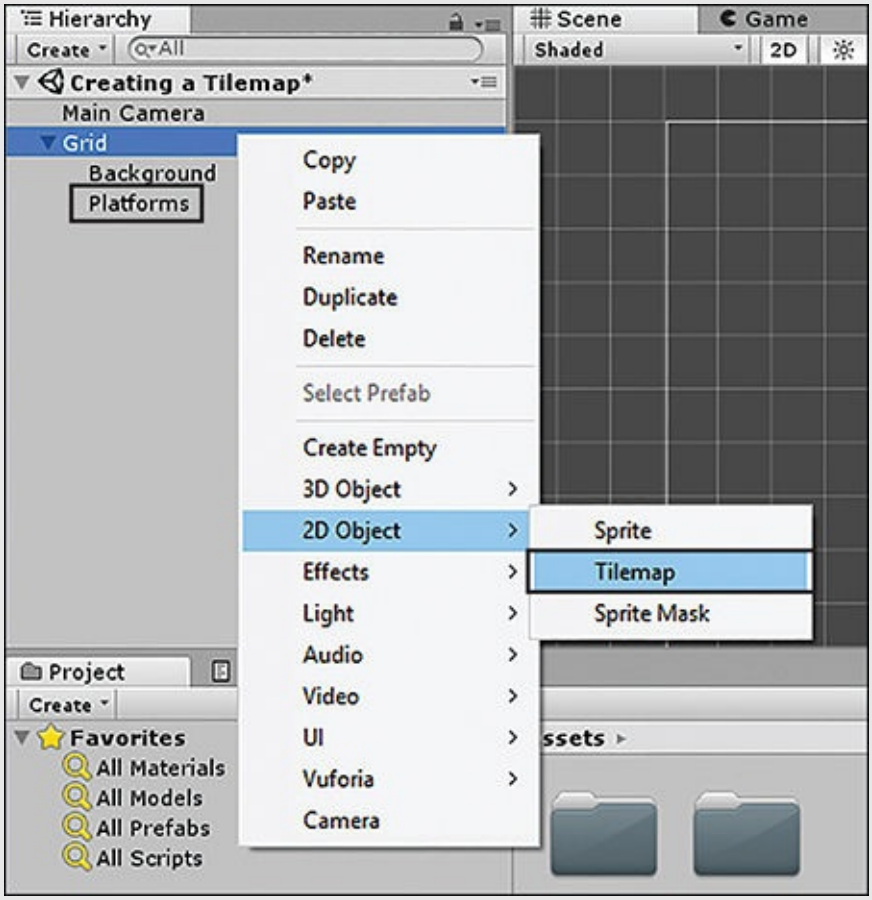

FIGURE 13.2
Adding a new tilemap from the Hierarchy view.

5. Add two new sorting layers to your project and name them

Background and **Foreground**. (Review Hour 12, “2D Game Tools,” if you do not remember how to add sorting layers to a project.)

6. Select the **Background** tilemap and set the Sorting Layer property of the Tilemap Renderer component to **Background**. Select the **Platforms** tilemap and set the Sorting Layer property of the Tilemap Renderer component to **Foreground**.

The Grid

As you saw earlier, when you add a tilemap to a scene, you also get a Grid game object (see Figure 13.3). The grid manages settings that are common across all similar tilemaps. Specifically, the grid manages the cell size and cell gap for your tilemaps. Therefore, if all your tilemaps need to be the same size, you need only a single grid for all of them to sit under. Otherwise, you can have multiple grids for multiple tilemap sizes. Notice that the default cell size is 1. This information will become important later, so be sure to remember that.

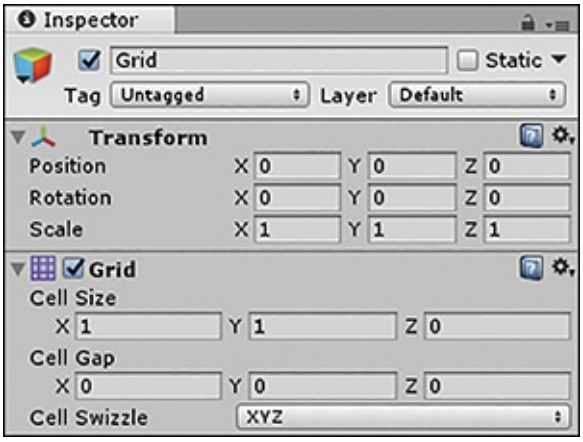

FIGURE 13.3
The Grid object.

TIP

Angled Tilemaps

Usually, tilemaps are aligned with each other. However, that doesn’t have to be the case. If you want one of your background tilemaps to be at an angle, you can simply rotate it in the Scene view. You can even move tilemaps to offset their positions for staggered tiles or push them back on the z axis for a built-in parallax effect.

Palettes

In order to paint tiles onto tilemaps, you need to first assemble them on a palette. You can think of the palette as a painter's palette, where all your painting choices take place. The palette comes with many tools that help you sculpt worlds exactly as you see fit. You access the tile palette by going to **Window > Tile Palette**. The Tile Palette window, shown in Figure 13.4, appears.

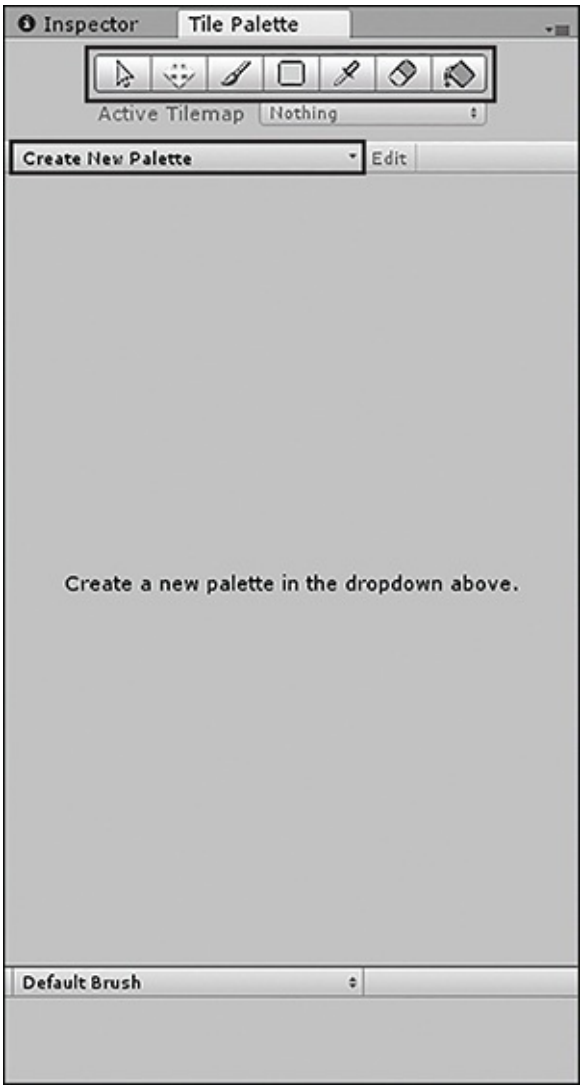

FIGURE 13.4
The Tile Palette window.

Tile Palette Window

The Tile Palette window has several tools for painting and a main middle area where all your tiles are laid out. By default, a project doesn't have any palettes to work with, but a new one can be created by clicking the **Create New Palette** drop-down.

TRY IT YOURSELF

Creating Palettes

Now it is time to add a couple palettes to your project. You will be using the project created in the Try It Yourself "Adding a Tilemap to a Scene," so if you haven't done that one yet, go ahead and complete it now. Be sure to save the scene because you will be using it more later in this hour. When you're ready to create a palette, follow these steps:

1. Open the scene you created in the Try It Yourself "Adding a Tilemap to a Scene." Open the Tile Palette window (by selecting **Window > Tile Palette**) and dock it next to the Inspector view (refer to Figure 13.4).

2. Add a palette by clicking **Create New Palette** and name the palette **Jungle Tiles**. Leave the rest of the palette settings at their defaults (see Figure 13.5). Click **Create**.

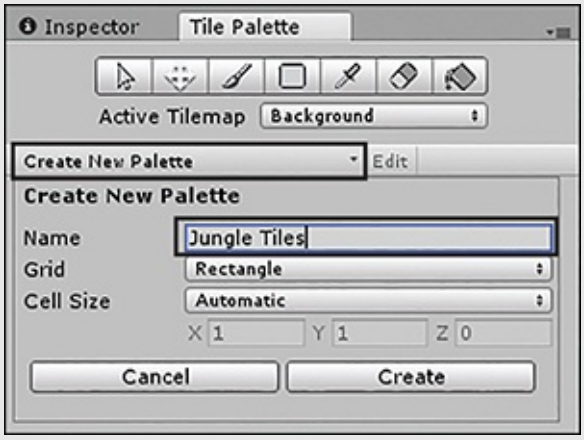

FIGURE 13.5
Creating a new palette.

3. In the Create Palette into Folder dialog that appears, create a new folder, name it **Palettes**, and click **Select Folder**.

4. Repeat steps 2 and 3 to create another palette named **Grass Tiles**.

When you are done, you should have two palettes in the palette drop-down and two tilemaps in the Active Tilemap drop-down (see Figure 13.6).

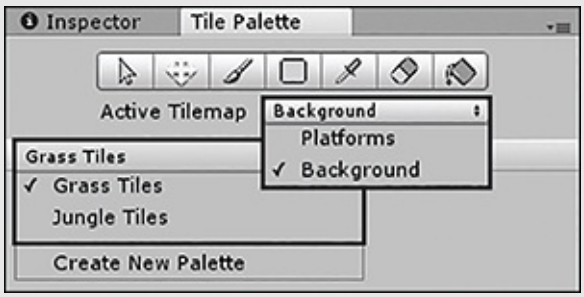

FIGURE 13.6
The correct palettes and tilemaps.

Tiles

So far in this hour, you have been doing a lot of prep work that will allow you to use tiles. Now it is time to dig in and make the tiles to paint with. In essence, tiles are sprites that are specially configured to be used with tilemaps. Sprites that are being used as tiles can still be used as regular sprites if you happen to need them for both. Once sprites are imported and configured, they can be turned into tiles, added to a palette, and then painted onto tilemaps.

NOTE

Crazy Custom Tiles

In this hour you are going to be working with and painting basic tiles. While there is a ton of functionality in these built-in tile options, there are a lot of further customizations that can be done with the new 2D Tilemap feature. If you want to take your expertise even further—creating things like animated tiles, smart tiles, or even custom brushes with game object logic built into the tiles—you should check out Unity's 2D Extras package. At the time this book was published, this package was located at https://github.com/Unity-Technologies/2d-extras. Eventually, though, this package will be migrated into the Unity engine's package manager and will be available like all the other assets you've been using in this book. So much to do, so much to see!

Configuring Sprites

Configuring Sprites

There isn't much you need to do to prepare sprites for use as tiles. There are two major steps:

1. Ensure that the Pixels per Unit property of your sprites is configured to be exactly the same size as the Cell Size property of your grid. (You'll learn more about this shortly.)
2. Slice up the sprites (assuming that they are in a sprite sheet) so that there is as little extra space around them as possible. Where possible, having no extra space around a tile is preferred.

The first step in preparing a sprite for use as a tile may seem complex, but it is really rather straightforward. For example, in this hour you will be using sprite sheets that have multiple tiles in them. These tiles are 64 pixels by 64 pixels (because that's how the artist made them). Since your grid's Cell Size property is 1 unit by 1 unit, you will want to set the Pixels per Unit property to 64. That way, every 64 pixels in your sprite will equal 1 unit, which is the cell size.

Creating Tiles

Once your sprites are prepared correctly, it is time to make some tiles. Doing so simply requires dragging the sprites onto the correct palette in the Tile Palette window and choosing where you'd like to save the resulting tiles. The original sprites remain where they are, unchanged. New tile assets are created that reference the original sprites.

▼TRY IT YOURSELF

Configuring Sprites and Creating Tiles

This exercise shows you how to configure your sprites and use them to make tiles. You will be using the project created earlier in this hour, so if you haven't done the Try It Yourself "Creating Palettes," go ahead and get caught up. If you've forgotten how to complete any of these steps, you can go back and review Hour 12. Be sure to save this scene because you will be using it more later in this hour. Follow these steps:

1. Open the scene you created in the Try It Yourself "Creating Palettes." Create a new folder and name it **Sprites**. Locate the two sprites GrassPlatform_TileSet and Jungle_Tileset in the book files for this

hour. Drag them into the newly created Sprites folder.

2. Select the **GrassPlatform_Tileset** sprite in the Project view and look at its properties in the Inspector view. Set Sprite Mode to **Multiple** and set Pixels per Unit to **64.** Click **Apply**.

3. Open the Sprite Editor and click **Slice** in the upper-left corner. Set Type to **Grid By Cell Size** and set the X and Y Pixel Size properties to **64** (see Figure 13.7).

4. Click **Slice** and then click **Apply**. Then close the Sprite Editor window.

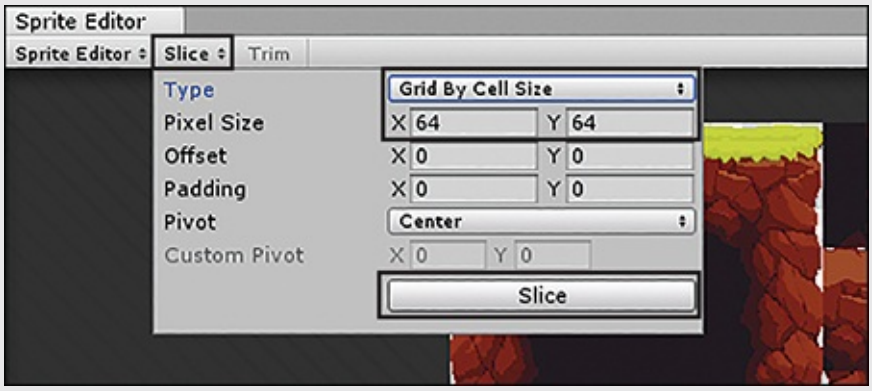

FIGURE 13.7
Slicing the sprite sheet.

5. Repeat steps 2 through 4 for the **Jungle_Tileset** sprite. This sprite is a bit larger than the grass tiles, though, so set Pixels per Unit and the X and Y Pixel Size properties to **128**.

6. Ensure that the Tile Palette window is open and docked next to the Inspector view. Also ensure that the Grass Tiles palette is currently active. Drag the **GrassPlatform_Tileset** sprite into the center area of the Tile Palette window (see Figure 13.8).

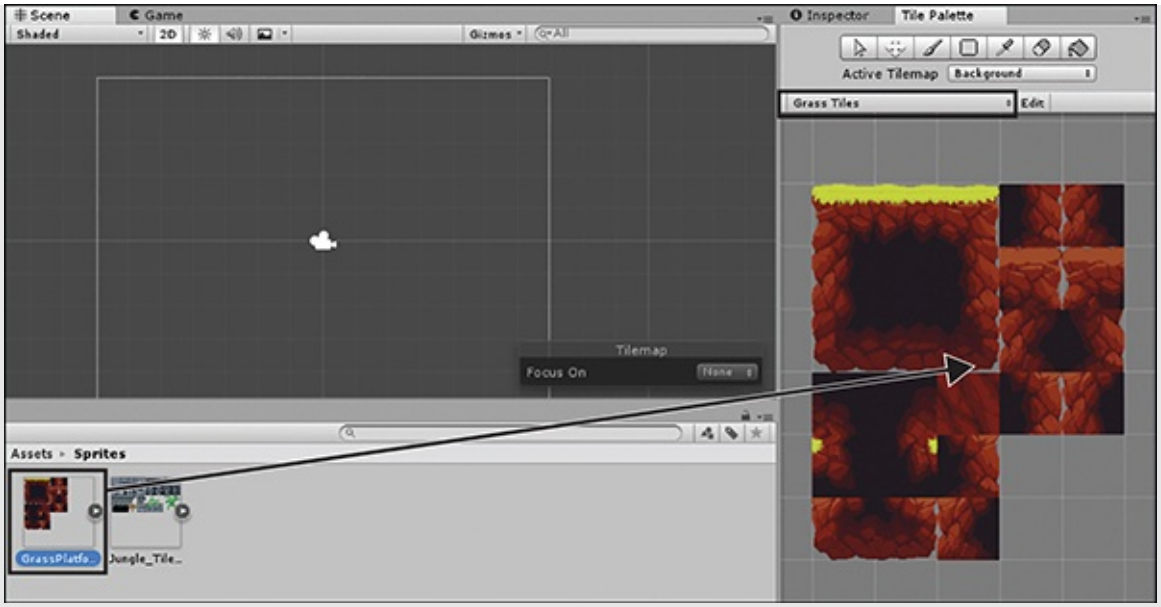

FIGURE 13.8
Turning sprites into tiles.

7. In the Generate Tiles into Folder dialog that appears, create a new folder, name it **Tiles**, and then click **Select Folder**.
8. In the Tile Palette window, change the active palette to Jungle Tiles and then repeat steps 6 and 7 for **Jungle_Tileset**.

Now that you have your sprites configured and your tiles created, it is time to start painting!

Painting Tiles

To paint tiles onto a tilemap, you need to pay attention to three things: the selected tile, the active tilemap, and the selected tool (see Figure 13.9). When selecting which tile to paint, you can click a single tile, or you can drag to grab a selection of tiles. This is useful if you want to paint a section of tiles that go together (like a complex roof piece).

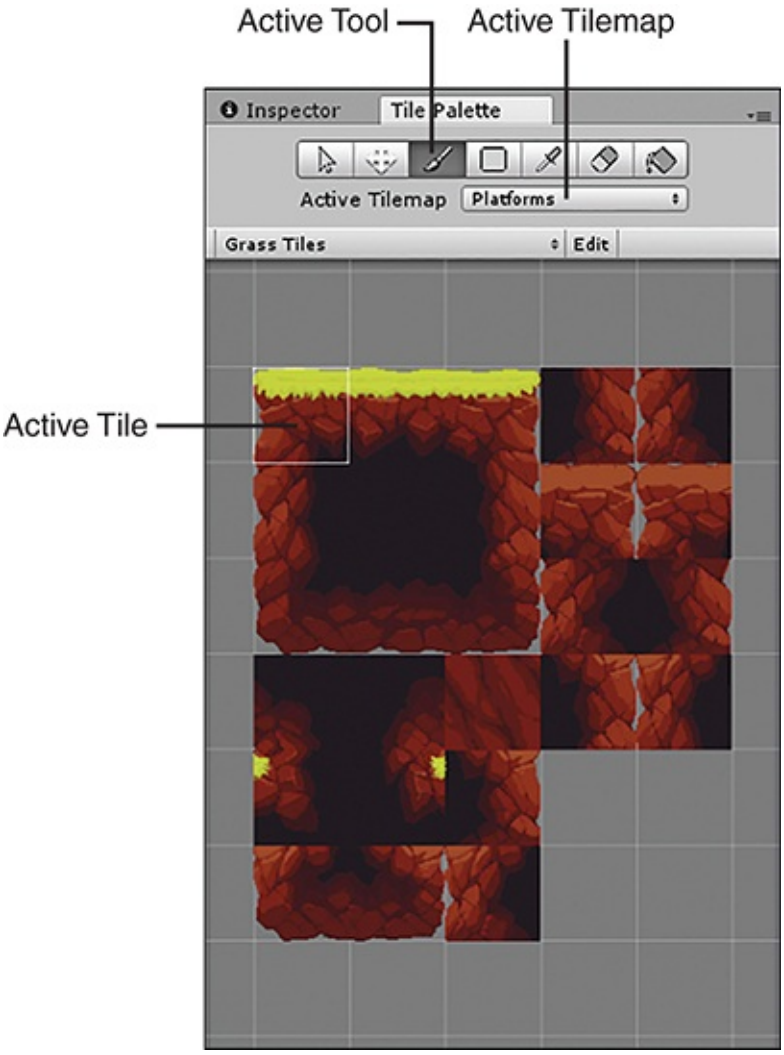

FIGURE 13.9
Preparing to paint.

When you're ready, click in the Scene view to begin painting on the tilemap. Table 13.1 lists the tools that appear in the Tile Palette window (from left to right).

TABLE 13.1 Tile Palette Window tools

Collider	Description
Select tool	Used to select a tile or group of tiles on a tilemap.
Move tool	Used to move a selection on a tilemap from one location to another.
Paint tool	Used to paint the currently highlighted tile (on the palette) onto the

Paint tool	Used to paint the currently highlighted tile (on the palette) onto the active tilemap. Clicking and dragging paints multiple tiles at once. Holding Shift while painting toggles this tool with the Erase tool. Holding Ctrl (Command on Mac) while painting toggles this tool with the Picker tool.
Rectangle tool	Used to paint a rectangular shape on the tilemap and fills it with the currently highlighted tile.
Picker tool	Used to select a tile from a tilemap to paint with (instead of highlighting it on the palette). This tool speeds up painting repeating complex tile groups.
Erase tool	Used to erase a tile or group of tiles from the active tilemap.
Fill tool	Used to fill an area with the currently highlighted tile.

▼ TRY IT YOURSELF

Painting Tiles

It is time to start painting tiles. You will be using the project created earlier in this hour, so if you haven't done that yet, go ahead and get caught up with all the Try It Yourself exercises. Be sure to save this scene because you will be using it more later in this hour. Follow these steps:

1. Open the scene created in the Try It Yourself "Configuring Sprites and Creating Tiles."
2. With the Tile Palette window open, select the **Jungle Tiles** palette and ensure that the Active Tilemap is set to **Background**.
3. Begin selecting tiles and painting them in the Scene view (see Figure 13.10). Continue to select tiles and paint until you've created a jungle background that you like.

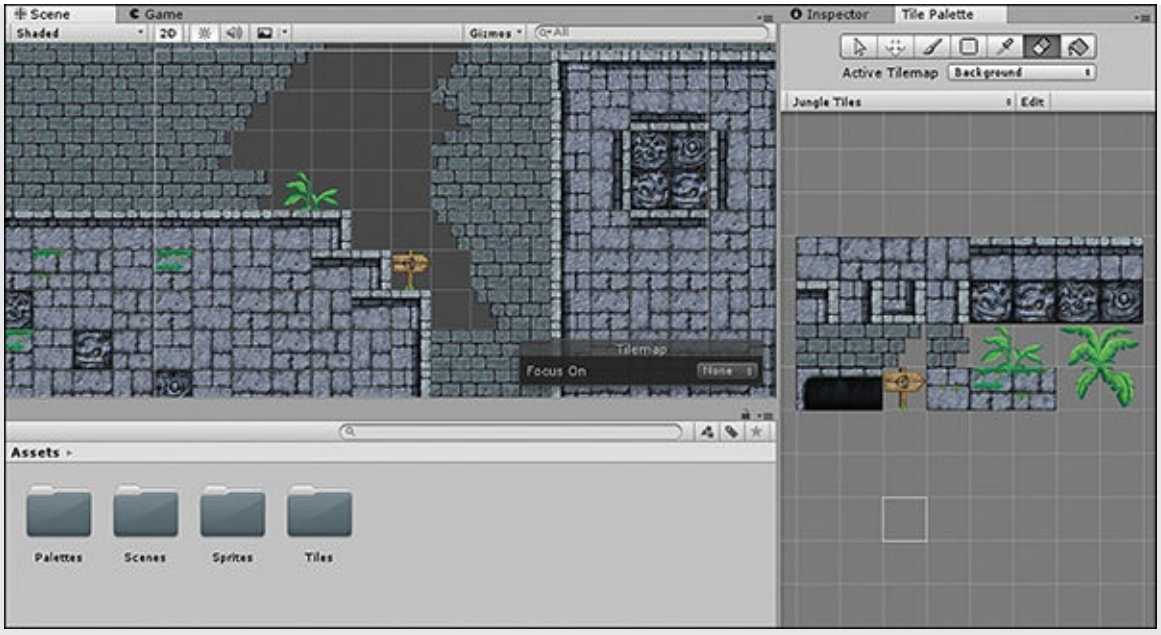

FIGURE 13.10
Painting the background.

4. Switch to the **Grass Tiles** palette and change Active Tilemap to **Platforms**.

5. Paint some grass platforms for your level. You will be using these platforms with a character controller soon, so be sure to make something a player can jump around on (see Figure 13.11).

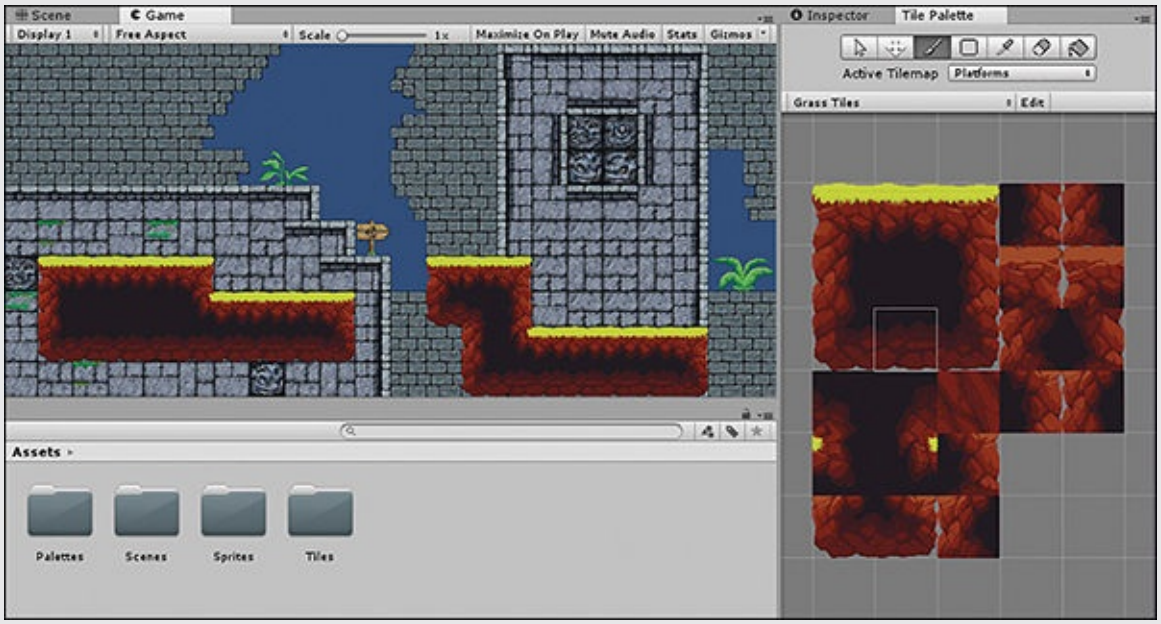

FIGURE 13.11

The finished level.

TIP

Enhanced Controls

Several hotkeys can assist you in finding and painting the correct tiles. First, the Tile Palette window uses the same controls for navigation as the 2D Scene view. This means you can use the scroll wheel to zoom as well as right-clicking and dragging to pan around. When painting tiles, you can rotate and flip them to generate new and interesting designs. Simply use the **,** (comma) and **.** (period) keys to rotate the tile before painting. Likewise, use **Shift+,** to flip the tiles horizontally and **Shift+.** to flip the tiles vertically.

Customizing the Palettes

You might have noticed that the palettes aren't exactly laid out in the most convenient manner. When you created the tiles by dragging sprites onto the palette, Unity placed them in a straightforward, but not necessarily intuitive, manner. Luckily, you can customize the palette to suit your needs. Simply click the **Edit** button in the Tile Palette window (see Figure 13.12) and use the palette tools to paint, move, or modify the tiles as you see fit. You can even create several copies of the same tile and rotate or flip them for convenient painting.

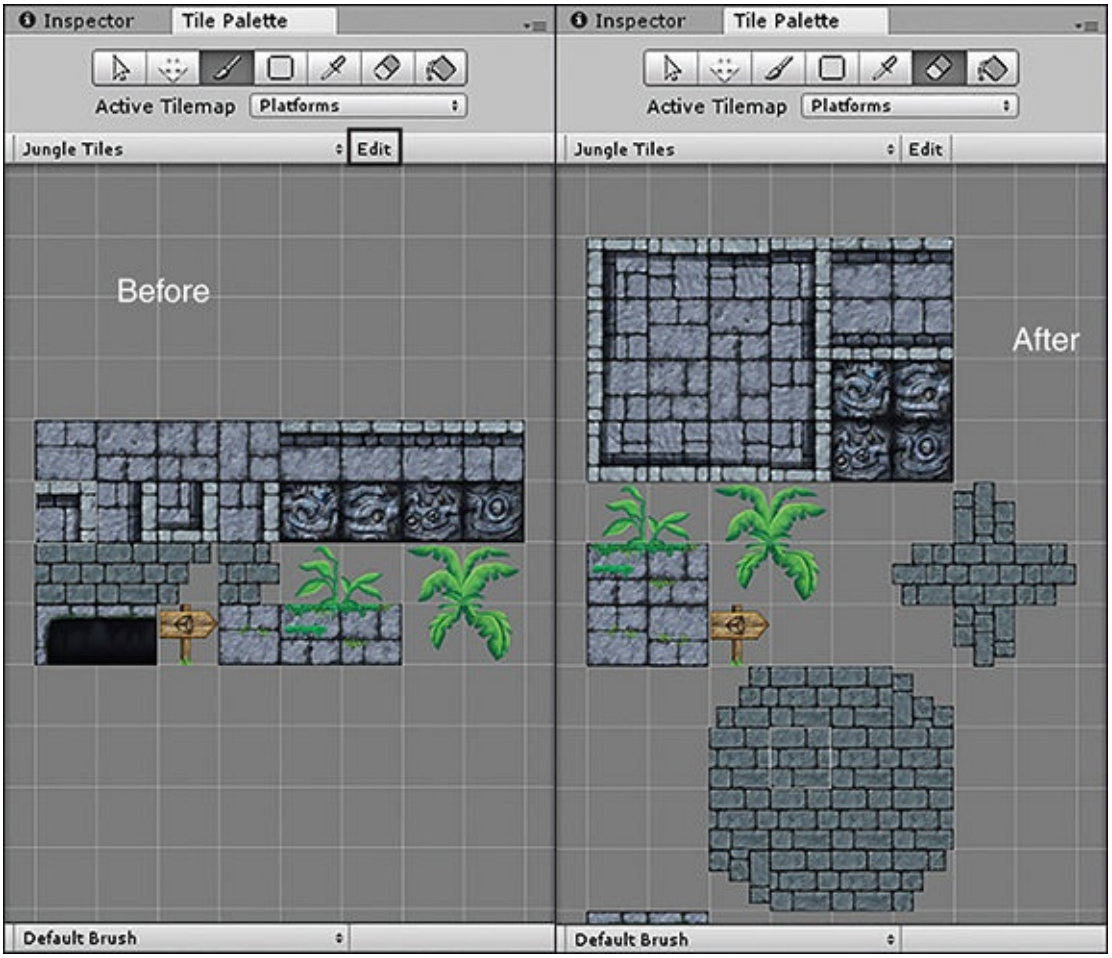

FIGURE 13.12
Editing the tile palette.

Tilemaps and Physics

You've now learned to paint on tilemaps to create brand-new 2D level designs. If you actually try to play with these levels, however, you'll find that your characters fall right through the floor. It's time to learn about using colliders with these new tilemaps.

Tilemap Colliders

You could add collision to your levels by placing box colliders around your tiles manually, but who wants to work that hard? I don't! Instead, you can use a Tilemap Collider 2D component to automatically handle collision. Besides working specifically for tilemaps, these colliders function just like any other colliders you've used in the previous hours. All you need to do is select the

tilemap you want to have collision and select **Component > Tilemap > Tilemap Collider 2D** to add the collider as a component.

TRY IT YOURSELF

Adding a Tilemap Collider 2D Component

In this exercise you are going to finish the scene you've been working with all throughout this hour by adding colliders to your platforms. You will be using the project created earlier in this hour, so if you haven't done that yet, go ahead and get caught up. Then follow these steps:

1. Open the scene you created in the Try It Yourself "Painting Tiles." Import the 2D standard assets (by selecting **Assets > Import Package > 2D**).
2. Locate the CharacterRobotBoy prefab (in the folder Assets\Standard Assets\2D\Prefabs) and drag it into your scene, above one of your platforms. You may need to change the scale of the prefab to (1, 1, 1).
3. Play your scene and notice that the robot falls right through the ground. Exit play mode.
4. Select the **Platforms** tilemap game object and then add a Tilemap Collider 2D component (by selecting **Add Component > Tilemap > Tilemap Collider 2D**). Notice that a collider is placed around each individual tile (see Figure 13.13).

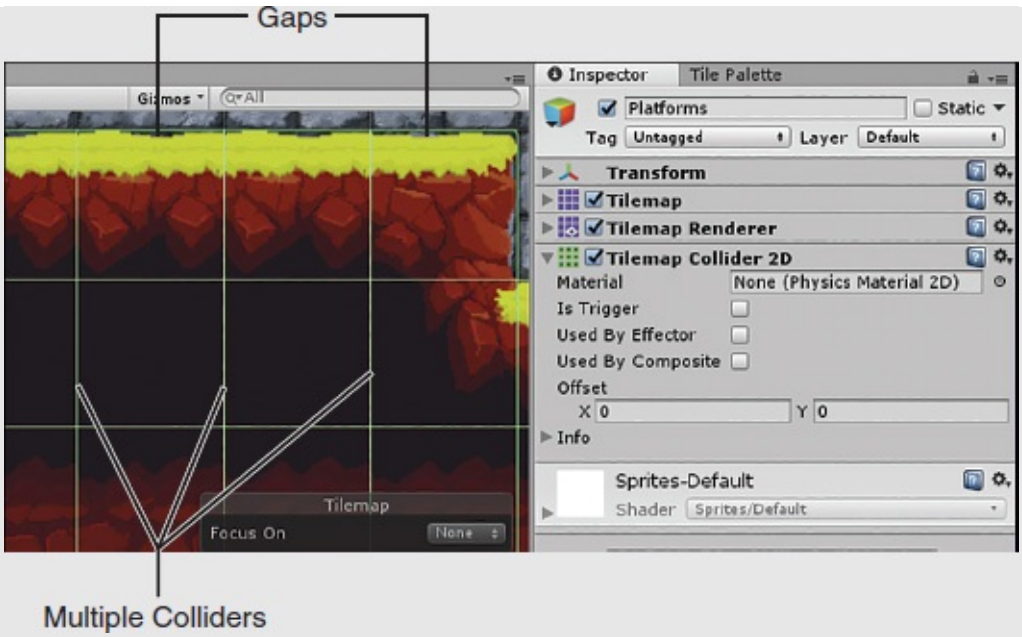

FIGURE 13.13
The Tilemap Collider 2D component.

5. Enter play mode again and notice that the robot lands on the platform now. If you zoom in, however, you will notice that because the collider fits around the tile, there are gaps between the grass platform and the collider (refer to Figure 13.13).

6. On the Tilemap Collider 2D component, set Y Offset to −**0.1** to lower the collider a little. Enter play mode again and notice that the robot now stands on the grass.

The colliders you now have on your tiles make the level complete and playable. One problem worth noting, however, is that placing a collider around each tile is very inefficient and can lead to performance issues. You can resolve this by using something called a Composite Collider 2D component.

TIP

Collision Accuracy

When you begin using colliders with tilemaps, you might need to make some changes to the rigidbody components of your moving objects. Because the edges of a Tilemap Collider 2D component and even a Composite Collider 2D component are very thin, you may notice that objects that have small

colliders or that move quickly can tend to get stuck on them or even fall through them. If you notice this behavior, you should set the Collision Detection property of the problematic rigidbody to **Continuous**. That should prevent any more of the collider issues with your tilemaps.

Using a Composite Collider 2D Component

A *composite* collider is a collider that is composed of many other colliders. In a way, it allows you to merge all the individual tile colliders into a single larger collider. The really cool thing about doing this is that whenever you add or change tiles, the collider is automatically updated for you. You can add a Composite Collider 2D component by selecting **Add Component > Physics 2D > Composite Collider 2D**. When you do, a Rigidbody 2D component is also added; it is required for the Composite Collider 2D component to work (see Figure 13.14). Obviously, if you don't want all your tiles to fall due to gravity, you will want to change the Body Type property of the Rigidbody 2D component to Static.

After you add the composite collider, nothing really changes with the tilemap. Each tile still has its own individual collider. The reason is that you need to tell the collider that it should be used in the composite. To do that, simply check the **Used By Composite** check box. After you do this, all the colliders are merged into one large (and more efficient) collider.

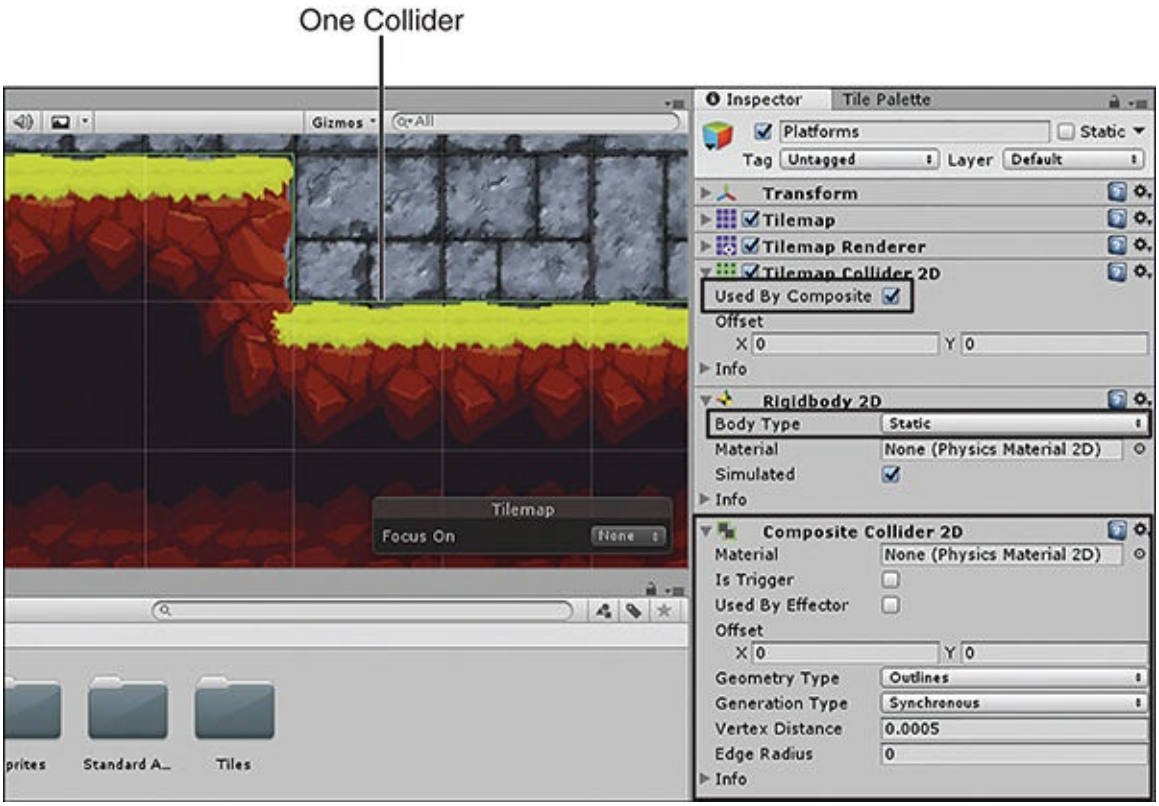

FIGURE 13.14
The Composite Collider 2D component.

Summary

In this hour, you have learned about creating 2D worlds using Unity's 2D Tilemap system. You began by learning about tilemaps in general. After that you created palettes and configured sprites to be added as tiles. Once your tiles were prepared, you painted a couple tilemaps to build a level. Finally, you learned how to add collision to your tilemaps to make them playable.

Q&A

Q. Can tilemaps be combined with regular sprites when building 2D worlds?

A. Yes, absolutely. A tile is really just a special kind of sprite.

Q. Are there any kinds of levels that tilemaps aren't good for?

A. Tilemaps are great for repetitive and modular levels. Scenes that involve a large number of different shapes or very unique and nonrepeating sprites

would be difficult to create with tilemaps.

Workshop

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

Quiz

1. What component defines properties (such as Cell Size) that tilemaps share?
2. Where are tiles placed prior to painting them onto a tilemap?
3. What collider type allows the combination of multiple colliders into one?

Answers

1. The Grid component
2. A palette
3. A Composite Collider 2D component

Exercise

In this exercise, you are going to experiment with the tilemaps you've created to enhance their appearance and usability. Here are some things to try:

- Try fully painting and modifying both of your tilemaps until you are satisfied with them.
- Try adding a Foreground tilemap to add more plant or rock elements to your scene.
- Try testing your full level by adding a 2D character and getting the camera to follow the character. The book files include a script called CameraFollow.cs to help get the camera to follow your player around.
- Try pushing your background tilemap away from the camera on the z axis. In this way, you can create a natural parallax effect. Remember that you will be able to see the effect only if you have a perspective camera.
- Try modifying the color property of the background's Tilemap component to make the background images look faded and distant.