Learning something new is always a little daunting at first, but things will start to become familiar in no time. In fact, by the end of this chapter, you’ll have completed your very first gaming masterpiece!

**Designing the Game: Evil Clutches**

Before you start making a game, it’s a good idea to have an idea of what you’re aiming for. Commercial game developers usually prepare long design documents before they start creating a game. Nonetheless, writing documents isn’t a fun way to learn how to make games, so we’ll keep our designs as short as possible. We’re calling the game in this chapter *Evil Clutches*, and this is its design:

> You play a mother dragon who must rescue her hatchlings from an unpleasant band of demons that have kidnapped them. The band’s boss sends a stream of demons to destroy the dragon as the hatchlings make their escape. The mother can fend off the boss’s minions by shooting fireballs, but must be careful not to accidentally shoot the hatchlings!

> The arrow keys will move the dragon up and down and the spacebar will shoot fireballs. The player will gain points for shooting demons and rescuing young dragons, but will lose points for any hatchlings that accidentally get shot. The game is over if the dragon is hit by a demon, and a high-score table will be displayed. Figure 2-1 shows an impression of what the final game will look like.

Using this description, we can list all the different elements needed to create our game: a dragon, a boss, demons, hatchlings, and fireballs. Making the game will require pictures of each of these as well as a background image, some sound effects, and music. We call all these different parts that make up the game *resources*, and the resources for this game have already been created for you in the Resources/Chapter02 folder on the CD. For the remainder of the chapter, we will learn how to put these resources together into a game and bring them to life.
Sprites
In Game Maker, pictures of dragons, demons, and other game objects are all called sprites. Sprites are one kind of resource used in games, and they can be made from images that have been created in art packages or downloaded from the Internet. Game Maker includes a simple sprite editor for drawing your own sprites, but you can use any drawing package you like for this purpose. However, creating sprites is time consuming, so we’ve already provided professionally drawn sprites for each game.

If you’ve not done so already, start up Game Maker. Figure 2-2 shows the (rather empty) main window that appears.

**Note** If your window doesn’t look exactly the same as shown in Figure 2-2, then you’re probably running Game Maker in Advanced mode. To switch to Simple mode, choose **Advanced Mode** from the **File** menu and the checkmark beside it will disappear.
In the main window of Game Maker (in Simple mode), the menu and toolbar runs along the top of the window and a list of resources down the left side. The left side of the window shows the different types of resources that make up the game: sprites, backgrounds, sounds, and so forth. These are currently empty, but the names of new resources will appear here as they are added to the game. The menu bar along the top of the window contains all the commands that allow us to work with resources—although most common tasks can also be accessed using the buttons on the toolbar. We'll begin by using the Create Sprite command to create a new sprite.

Creating a new sprite resource for the game:

1. From the Resources menu, choose Create Sprite. The Sprite Properties form appears, like the one shown in Figure 2-3.
2. Click the Name field, where it currently says sprite0. This is the default name created by Game Maker for the new sprite, but you should rename it to sprite_dragon.

3. Click the Load Sprite button. This opens the standard Windows file requester.

4. Select the required image using the file requester. The image for the dragon is called Dragon.gif, and you’ll find it in the Resources/Chapter02 folder on the CD. Your Sprite Properties form should now look like Figure 2-4.

**Note** Always avoid using spaces and punctuation in names for resources as they will confuse Game Maker when you try to use some of its more advanced functions later on. You can use the underscore (_) symbol instead of spaces, which is usually found on the same key as the minus symbol (press Shift and the minus key).

5. Click OK to close the form. You have now created a sprite.

The dragon sprite should now have been added to the list of sprites in the resource list. If you ever need to change a resource, you can reopen its properties form by double-clicking on its name in the resource list. Do this now and take another look at the dragon sprite’s properties (Figure 2-4).

The form shows that there are six subimages to this sprite. Sprites often consist of several images shown one after the other to create the illusion of movement. If you move through the subimages using the blue arrow button, you will notice that there are actually only two different images for this sprite. The extra copies make sure that the dragon doesn’t flap its wings too quickly when it’s animating.

The checkmark next to the Transparent property means that the background of the dragon sprite is see-through. Most sprites are set to transparent so that the surrounding rectangle won’t be drawn when the sprite appears in the game. Figure 2-5 shows the difference that the Transparent option makes—the advantages are obvious to see!
Here's the dragon sprite with the Transparent option set (left) compared to the same dragon without the Transparent option (right).

**Note** Game Maker works out which color to make transparent based on the color in the bottom leftmost corner of each image. This is worth remembering when you want to create your own sprites.

Okay, let's create the other sprites for the game in the same way.

**Creating the remaining Evil Clutches sprites:**

1. From the Resources menu, choose Create Sprite.
2. In the Name field in the Sprite Properties form, type the name sprite_boss.
3. Click the Load Sprite button and choose the file Boss.gif.
4. Click OK to close the Sprite Properties form.
5. Now create a demon sprite, baby sprite, and fireball sprite using the Demon.gif, Baby.gif, and Fireball.gif files in the same way. Give each sprite an appropriate name (using only letters and the underscore symbol).

This completes all the sprites needed to create the Evil Clutches game.

**Objects**

Sprites don't do anything on their own; they just store pictures of the different elements in the game. **Objects** are the parts of the game that control how these elements move around and react to each other. We'll begin by creating our first object to tell Game Maker how we want the demon boss to behave.

**The Boss Object**

The following steps create a new object and assign it a sprite so that Game Maker knows how it should look on the screen.
Creating a new object and assigning it a sprite:

1. From the Resources menu, choose Create Object. An Object Properties form like the one in Figure 2-6 appears.

![Figure 2-6. Open the Object Properties form for your new object.](image)

2. In the Name field, give the object a name. You should call this one object_boss.

3. Click the icon at the end of the sprite field and a list of all the available sprites will appear. Select the sprite_boss sprite.

---

**Caution** Always make sure that you give your object resources names that are different from your sprite resources. Ending up with an object and a sprite both called “dragon,” or two objects called “demon,” can confuse Game Maker when you try to use its more advanced functions later on. Adding prefixes like “sprite_” or “object_” to names is a good way to achieve this without having to think of new names.

---

**Events and Actions**

Game Maker uses *events* and *actions* to specify how objects should behave. *Events* are important things that happen in the game, such as when objects collide or when the player presses a key on the keyboard. *Actions* are things that happen in response to an event, such as changing an object's direction, setting the score, or playing a sound. Game Maker games are basically just a collection of objects with actions to tell them how they should react to different events. Therefore, to set the behavior of an object in Game Maker you must define which events the object should react to and what actions they should perform in response.

The boss object's lists of events and actions are currently empty. We're going to begin by adding an event and action that will start the boss moving up the screen at the beginning of
the game. This will be complemented by an action that reverses the vertical direction of the boss in the event that it collides with the edge of the screen. As a result, the boss will continually move up and down between the top and bottom of the screen.

**Adding a create event for the boss object:**

1. Click the **Add Event** button. The Event Selector appears, as shown in Figure 2-7.

   ![Figure 2-7.](image)
   
   **Figure 2-7.** Click **Add Event** to open the Event Selector pop-up form.

2. Click the **Create** event to add it to the list of events. A new event is automatically selected (with a blue highlight) in the event list, as shown in Figure 2-8. This means we're already looking at this event's **Actions** list alongside it (which is currently empty).

   ![Figure 2-8.](image)
   
   **Figure 2-8.** This is how the Object Properties form should look once the name, sprite, and **Create** event have been added.

3. Next you need to include the **Move Fixed** action in the list of actions. To do this, press and hold the left mouse button on the action image with eight red arrows, drag it to the empty **Actions** list box, and release the mouse button. An action form will then pop up asking for particular information about this action (see Figure 2-9).
4. Select the up arrow and enter a value of 8 for the Speed. This will make the object move vertically 8 pixels (the tiny squares that make up a monitor display) for every step that it takes.

5. Press OK to close the action form and it will be included in the list of actions.

**Note** Whenever we use an action in the instructions, that action's image is shown in the left margin to help you find the correct one.

---

This event should start the boss moving upward. Now we'll add an event to reverse an object's vertical direction when it collides with the edge of the screen. This event is called the Intersect Boundary event because it gets called when the object's sprite intersects the screen's boundary by being partly in and partly out of the screen.

### Adding an intersect boundary event for the boss object:

1. Click the Add Event button.

2. Choose Other from the Event Selector pop-up form and select Intersect boundary from the drop-down menu that appears. This action will then be added and selected in the list of events.
3. Include the **Reverse Vertical** action in the list of actions for this event. You’ll now see the form shown in Figure 2-10.

![Figure 2-10. The action form for the Reverse Direction action looks like this.](image)

4. Nothing needs changing on this form, so just click **OK**. The Object Properties form for the boss object now looks like the one shown in Figure 2-11.

![Figure 2-11. In the Object Properties form for the boss object, we’ve added two events, along with their corresponding actions.](image)
These are all the events and actions we need for the boss right now. You can switch between the different events by clicking on them in the Events list. The selected event is highlighted in blue and the actions for that event are then shown in the Actions list. You can edit the properties of each action by double-clicking on them, but we're done with the boss object for now.

5. Click OK at the bottom left of the form to close it.

The Dragon Object

Now let's turn our attention to the heroine of the game. We'll begin by creating an object for the dragon in the same way as for the boss.

Creating a dragon object:

1. From the Resources menu, choose Create Object.
2. Give the object a name by entering object_dragon in the Name field.
3. Select the sprite_dragon sprite from the drop-down sprite menu.

The dragon also needs actions to make it move up and down the screen, but this time only when the appropriate keys are pressed on the keyboard. We do this by using keyboard events.

Adding keyboard events for the dragon object:

1. Click the Add Event button.
2. Choose a Keyboard event and select <Up> from the pop-up menu (to indicate the up arrow key).
3. Include the Move Fixed action in the Actions list.
4. In the action form, select the upward direction and set Speed to 16.
5. Repeat the previous process to add a Keyboard event for the <Down> key that includes a Move Fixed action with a downward direction and a speed of 16. The Object Properties form should now look like the one shown in Figure 2-12.

We just need one more event and action to make the dragon's movement work correctly. Our Keyboard events will start the dragon moving when the player presses the arrow keys, but there are currently no events to stop it from moving again when the keys are no longer being pressed. We use the Keyboard, <no key> event to test for when the player is no longer pressing any keys.
Adding a no key event for the dragon object:

1. Click the Add Event button.
2. Choose a Keyboard event and select <no key> from the pop-up menu.
3. Include the Move Fixed action in the Actions list for this event.
4. Select the center square in the action form, to indicate no movement, and set Speed to 0. The form should now look like Figure 2-13.
5. That’s all the actions we need to make our dragon move up and down, so click OK to close the Object Properties form for the dragon object.
Caution When setting a Move Fixed action with a speed of 0, you must also select the center square of the direction grid. If no direction square is selected at all, then the action is ignored!

Rooms

Our dragon and boss objects are all ready to go now, but in order to see them we need to put them into a level. Levels in Game Maker are made using rooms, and putting objects into a room defines where they will appear at the start of the game. However, not all objects need to be there at the start of the game, and they can be created on the fly as well (fireballs, for example). Let's create a new room.

Creating a new room resource:

1. Select Create Room from the Resources menu. A Room Properties form will appear (see Figure 2-14).

![Room Properties form for a new room opens.](image)

Note If there are sliders along the edges of the room grid, then the window is not currently large enough to see the entire room. Maximize the Game Maker window and the Room Properties form to see more of the room, or use the sliders to scroll around the entire room.
2. Click the **settings** tab in the top left of the form.

3. Enter a name for the room in the **Name** field. Call this one **room_first**.

4. Enter a caption for the title bar of the window when the game is running. "Evil Clutches" seems appropriate for this game. The room settings should now look like Figure 2-15.

![Figure 2-15. Here's the settings tab of the Room Properties form, with the name and caption filled in.](image)

Now we can place our objects in the new room.

**Adding a dragon and boss to the room:**

1. Click the **objects** tab in the top left of the form. You should see that the dragon object is already selected as the object “to add with left mouse.”

2. Click on the room grid with the left mouse button. An instance of the dragon object will be placed with its top-left corner at the point at which you click. The position you place the dragon becomes its starting position in the game, so put just one dragon close to the left boundary of the room area. If you add it in the wrong place, use the right mouse button to remove it again.

3. Click on the dragon’s image on the **objects** tab (or on the image of the pop-up menu next to where it says **object_dragon**) and select the boss object from the menu that appears.

4. Place an instance of the boss close to the right edge of the room, but make sure that the whole of his sprite is completely inside the room—otherwise his events will not work correctly! The room should now look something like Figure 2-16.
Figure 2-16. The room with the dragon and the boss looks like this.

Our very first version of the game is now ready. Click the green checkmark in the top-left corner of the form to close it and you can see the results of your labor . . .

Tip You can also click and hold the mouse button to move instances within a room.

Save and Run

It’s always a good idea to save your work as often as possible—just in case your computer crashes. In case you haven’t already worked it out for yourself, then the steps for this process are given here. However, in the future you’ll have to remember to save your work regularly yourself! This works in the same way as most programs.

Saving your work and running the game:

1. Choose Save from the File menu (or click the disk icon).

2. The first time you save the game, you will be prompted for a location and filename in the normal way. Note that Game Maker files always end with the extension .gm6. Save this game in a place where you can easily find it again (on the desktop, for example).
3. To run the game, select **Run Normally** from the **Run** menu. After a brief pause, a game window should appear, like the one shown in Figure 2-17.

![Figure 2-17. Here's the first version of the Evil Clutches game in action.](image)

You should now be able to move the dragon up and down using the arrow keys, and the boss should float up and down by itself. If your game doesn't work in this way, then you might want to check through all your steps in the previous sections. You may also need to ensure that your game window is selected (by clicking on it with the mouse) before your keyboard input has any effect. All games we make in the book are stored on the CD in stages, and the current version of the game can be found on the CD in the file `Games/Chapter02/evil1.gm6`.

Although we now have a running game, it’s not much fun to play yet as there are no goals or challenges. We’ll spend the remainder of the chapter turning it into a playable game. Press Esc to stop the game.

---

**Tip** Pressing F4 while the game is running will maximize the game to fill the entire screen. Press F4 again to return to the windowed version.
Instances and Objects

So far we have two object resources in our game and two characters appearing on the screen. However, there is an important distinction to be made between object resources and instances of objects that appear on screen. It may seem odd, but now that we have made dragon and boss objects, we can put as many instances of dragons and bosses on the screen as we like. Try it—go back and place more dragons and bosses in the room. If you run the game, you will find that they all behave in exactly the same way as the original instances! (Don't forget to remove them again afterward using the right mouse button.) A good way to think of the relationship between objects and instances is to think of objects as jelly molds and instances as the jellies that you make with them. You only need one mold to make any number of jellies, yet the mold defines the appearance of all of them (see Figure 2-18). From now on we will talk about instances and objects in this way, so it is important that you appreciate the difference.

Figure 2-18. Object resources are like jelly molds, and they can be used to create any number of object instances on the screen at once.

Demons, Baby Dragons, and Fireballs

To create some challenges and goals, we’re going to need to bring our remaining objects into the game. Let’s start by giving the dragon the ability to breathe fireballs—as dragons often do!

The Fireball Object

To create the fireball object you’ll need the fireball sprite. If you didn’t get around to doing this earlier, then quickly flick back a few pages and add it in the way that was described in the “Sprites” section. You should remember the basic steps required to making a new object by now, but here they are one more time, just in case.
Creating the fireball object:

1. Select Create Object from the Resources menu.
2. Call the object object_fireball.
3. Select the fireball sprite.

We now need to think about how we want fireballs to behave. When the dragon creates a fireball, we want it to move across the screen toward the boss and get destroyed when it reaches the other side of the screen.

Adding the fireball object’s events:

1. Click the Add Event button and choose the Create event.
2. Include the Move Fixed action in the Actions list. Select the right arrow to indicate the direction and set Speed to 32 (fireballs fly fast!).
3. Click the Add Event button again, select Other events, and pick Outside room.
4. Select the main1 action tab and include the Destroy Instance action in the Actions list. In the action form that pops up, simply click OK. The fireball Object Properties form should now look like Figure 2-19.
5. Click OK to close the fireball Object Properties form.

Figure 2-19. The properties form for the fireball object should now look like this.

Caution It is always a good idea to make sure that instances are deleted when they’re not needed any more (when they go off the edge of the screen, for example). Even though you can’t see them, Game Maker still has to spend time updating them, and too many instances will eventually slow down the program.
Now we need to tell the dragon object to create instances of the fireball object when the player presses the spacebar. We do this in a similar way to the events that make the dragon move, but this time using a **Key Press** event rather than a **Keyboard** event. **Keyboard** events happen as long as the player continues to hold down the key, but **Key Press** events happen only once when the key is first pressed. Using a **Keyboard** event for the fireballs would create a continuous stream of fireballs and make the game too easy, so that’s why we’re using **Key Press** instead.

**Creating a Key Press event for the dragon object:**

1. Double-click the dragon object in the resource list (not the dragon sprite). This will bring back the Object Properties form for the dragon object.

2. Click the **Add Event** button. Select the **Key Press** event and then choose `<Space>` from the pop-up menu.

3. Select the main1 action tab and include the Create Instance action in the Actions list.

4. In the action form that appears, we need to specify which type of instance to create and where on the screen it should be created. Select the fireball object from the menu, enter a value of 100 into X and 10 into Y, and select the Relative checkbox. Figure 2-20 shows what the completed action form should look like.

5. Click OK to close the action form and click OK again to close the Object Properties form.

![Create Instance form](image)

**Figure 2-20.** Note that we checked the Relative property to make the fireball appear relative to the position of the dragon.

The x and y values you just entered are screen coordinates, which are used to indicate positions on the game screen. Screen coordinates are measured in pixels (the tiny squares that make up a monitor display), with x values indicating the number of pixels horizontally, and y values indicating the number of pixels vertically.
We need to select the Relative option because the fireball needs to be created on the screen in front of the dragon, in other words, relative to the dragon’s position. However, the dragon’s position is measured from the top-left corner of its sprite—just above its wings—and this would be a crazy place for the fireball to appear! Giving an x-coordinate of 100 moves the fireball across 100 pixels to the right (to just above its head) and a y-coordinate of 10 brings it 10 pixels down. This creates the fireball right in front of the dragon’s mouth and exactly where we need it (see Figure 2-21). Test the game now to check that you can use the spacebar to shoot fireballs, and that they appear in the correct position.

![Figure 2-21. The fireball needs to appear from the dragon’s mouth, which is 100 pixels across and 10 pixels down from the origin of the dragon’s sprite.](image)

The Demon Object

The demon object will work in the same way as the fireball, except that demons fly from right to left and are created by the boss. Also, to make demons a bit more interesting, we’ll start some moving diagonally as well as horizontally. Those that head diagonally for the top or bottom of the screen will need to reverse their vertical direction when they intersect the boundary—like the boss object does. We’ll also need to destroy demons when they go outside the room, like the fireball. Next we provide the steps you need to do all of this; notice that we’ve started to shorten the steps that you should be familiar with by now.

Creating the demon object:

1. Create a new object called object_demon and give it the demon sprite.
2. Add a Create event and include the Move Fixed action.
3. Select all three left-pointing direction arrows and set Speed to 12. Selecting more than one direction causes Game Maker to randomly choose between them when an instance is created. The action form should now look like Figure 2-22.
4. Add an **Intersect boundary** event (in the **Other** events) and include the **Reverse Vertical** action in it.

5. Add an **Outside room** event (also in the **Other** events) and include a **Destroy Instance** action in it.

The demons will now bounce back and forth between the top and the bottom of the screen, but we also need them to react to collisions with other instances. For this we use a **collision event**, which happens when two sprites of different objects overlap on the screen. The first collision event we need is for when a demon collides with a fireball. This event should destroy the demon, and reward the player by increasing their score. There are a number of different actions dealing with scores, health, and lives in the **score** actions tab. As soon as the score changes, it will automatically be displayed in the game window caption.

**Adding an event to the demon object for colliding with the fireball:**

1. Click the **Add Event** button, choose the **Collision** event, and select the fireball object from the pop-up menu.

2. Include the **Destroy Instance** action from the **main** action tab.

3. Also include a **Set Score** action from the **score** tab. This should automatically appear below the **Destroy Instance** action in the **Actions** list. Lists of actions like this are carried out one after another, starting from the top of the list and working down.

4. Enter a value of **100** in the **Set Score** action form, and click the **Relative** property. This property makes the action set the score *relative* to the current score, so 100 will be added to the score rather than setting the score to 100. See Figure 2-23.

If a demon collides with the dragon, then the game is over. When this happens, we want to bring up a high-score table and (when appropriate) let the player enter their name. After showing the high-score table, we want to restart the game. Conveniently, Game Maker provides a **Show Highscore** event that handles most of this automatically.
Adding an event to the demon object for colliding with the dragon:

1. Add a **Collision** event for colliding with the dragon object.
2. Include a **Show Highscore** action from the **score** tab.
3. Click **OK** to keep the default settings for this action's properties.
4. Also include a **Restart Game** action from the **main2** tab. This action has no properties.
5. The object properties form for the demon should now look like Figure 2-24. Check that you have included all the demon object's events. We're done with this object for now, so click **OK**.

---

**Figure 2-23.** We add 100 to the score by setting the **Relative** property.

---

**Figure 2-24.** The Object Properties form for the demon object now looks like this.
Summoning Demons

That's it for the demon, but we still need the boss to create instances of the demon in the first place. However, we don't want the demons to appear at regular intervals because this would make the game too easy. Instead, we want there to be a random chance that a demon is created at each “step” of the game. A step is essentially just a short period of time in which everything on the screen moves a very small distance. There are normally 30 steps in every second, so we only need there to be a very small chance that a demon is created in each step. We achieve this by using a Test Chance action, which acts like throwing a die with many sides (see Figure 2-25). In each step we throw the die, but only one side will trigger the chance action and create a demon. In this way, we create a steady, but unpredictable, flow of demons.

![Figure 2-25. The more sides a die has, the less often Game Maker will throw the one side that triggers the Test Chance action.](image)

Adding a step event to the boss object:

1. Double-click the boss object in the resource list to bring back its Object Properties form.

2. Click the Add Event button, select the Step event, and choose Step again from the pop-up menu.

3. Include the Test Chance action from the control tab. Set the sides of the die to 50 in the action's properties.

4. Also include the Create Instance action in the Actions list for this event. Set the properties to create a demon object and select the Relative option, so that the demon is created relative to the boss's position. The event should now look like Figure 2-26.
The Test Chance action is an example of a conditional action. Conditional actions control the action that immediately follows them so that it is only performed if some condition is met. So in this case the Create Instance action is only performed if the Test Chance action rolls a 1 using a 50-sided die—otherwise it is skipped.

Click OK, save your work, and run the game to test it. Demons should now be appearing, and you should be able to shoot them with your fireballs to rack up your score in the window caption. When you eventually get hit by a demon, the high-score table will be displayed and the game restarts. How long can you survive?

The Baby Dragon Object

We now have a game with two goals: shooting demons and staying alive. However, it’s still not much fun to play as it’s far too easy to provide any real challenge. To increase the challenge, we’re going to occasionally throw in a baby dragon along with the demons. If the player shoots a baby dragon, they will lose 300 points, but if they rescue one they will gain 500 points. This will mean that the player will have to be much more careful about when they shoot, thereby increasing the challenge of the game.

Creating a new baby dragon object and its events:

1. Create a new object called object_baby, and give it the baby dragon sprite.
2. Add a Create event for the object and include a Move Fixed action in it. Set it to move left with a Speed of 8 (slower than the demons to make life harder).
3. Add an Outside room event (in Other events) and include a Destroy Instance action from the main1 tab.
4. Add a Collision event with the fireball object and include a Destroy Instance action in that as well.
5. Also include a Set Score action in the collision event with a value of -300 and the Relative property selected. This will subtract 300 from the player’s score.

6. Add a Collision event with the dragon object and include the Destroy Instance action in it.

7. Also include the Set Score action with a value of 500 and the Relative property selected. This will add 500 to the player’s score. The baby dragon object should now look like Figure 2-27.

8. Click OK to close the properties form.

![Object Properties form for the baby dragon object looks like this.](image)

Figure 2-27. The Object Properties form for the baby dragon object looks like this.

Now we need to make the boss randomly release baby dragons as well as demons. This is exactly the same as for the demons except we will use a value of 100 for the die so that they are created less often.

**Editing the boss object to randomly create baby dragons:**

1. Reopen the Object Properties form for the boss object.

2. Click on the existing Step event to select it and view its actions.

3. Include another Test Chance action in the Step event. Set the sides of the die to be 100 in the action’s properties.

4. Include the Create Instance action below the new Test Chance action in the Actions list. Set the properties to create a baby object and select the Relative option.

That completes the second phase of our game! All the gameplay elements are now in place. Save the game and carefully test it to make sure it works correctly. You’ll also find the current version of the game on the CD in the file `Games/Chapter02/evil2.gm6`.
Backgrounds and Sounds

In this section we’ll finish off the look and feel of our game by adding background graphics, sound effects, and music. As you’ll see, these finishing touches have quite a dramatic effect on how professional the game seems.

A Background Image

The first improvement we’ll make is to add a background to the room. Backgrounds are another type of resource, like sprites, rooms, and objects. We’ve created an image that is exactly the same size as the game window (640×480 pixels). This needs to be loaded into a new background resource, which can then be assigned to a room.

Creating a new background resource and assigning it to a room:

1. Select **Create Background** from the **Resources** menu.
2. Call the background **background_cave**, and click the **Load Background** button. Select the **Background.bmp** image from the Resources/Chapter02 folder on the CD. The Background Properties form should now look like Figure 2-28.
3. Click **OK** to close the Background Properties form.
4. Reopen the properties form for the room by double-clicking on it.
5. Select the **backgrounds** tab in the Room Properties form. Click the menu icon to the right of where it says <no background> and select the new background from the pop-up menu. The Room Properties form now looks like Figure 2-29.
6. Close the Room Properties form by clicking the green checkmark in the top-left corner of the form.

![Figure 2-28. The Background Properties form allows you to load and edit backgrounds.](image)
Background Music

The next step is to add some atmospheric music. Sounds are another kind of Game Maker resource for including both sound effects and music. We need to create a sound resource for the music and then set up an action to start the music playing. We’ll include this action in the Create event of the boss object so that it starts playing at the beginning of the game, but it would work just as well in the dragon object too.

Creating a music sound resource and playing it in the boss object:

1. Select Create Sound from the Resources menu and call it sound_music.
2. In the properties form that appears, click Load Sound and select the Music.mp3 file from Resources/Chapter02 on the CD. The Sound Properties form should now look like Figure 2-30.
3. Close the Sound Properties form by clicking OK.
4. Reopen the Object Properties form for the boss object.
5. Click the existing Create event to select it and view its actions.
6. Include a Play Sound action (main1 tab) in the Create event.
7. In the action properties, select the music sound and set the **Loop** property to true. This makes the music loop back to the start when it finishes. The sound action form should then look like Figure 2-31.

8. Click **OK** to close the action, and click **OK** again to close the boss object.

![Figure 2-30. The Sound Properties form allows you to load, preview, and save sound files.](image)

![Figure 2-31. This sound action loops the background music.](image)

**Sound Effects**

Adding sound effects is another way to enhance the atmosphere of a game, but they also help to inform the player about their actions. For now, we’ll just add two sound effects to our game: one for shooting a demon and one for shooting a baby dragon. The baby’s sound effect will be much higher-pitched and cuter than the demon’s so that the player instantly knows they have done something wrong.

**Creating and playing sound effects for shooting babies and demons:**

1. Create a new sound resource called `sound_demon`.
2. Load the `Demon.wav` file from `Resources/Chapter02` on the CD.
3. Close the Sound Properties form.
4. Reopen the Object Properties form for the demon object and select the existing Collision event with the fireball object.

5. Include a Play Sound action in the collision event and select the new sound. Leave the Loop property set to false.


7. Repeat the previous steps to create a sound resource for Baby.wav. Include an action to play it in the baby dragon object's collision event with the fireball.

Congratulations

Congratulations on completing your very first game using Game Maker! If you need it, then you’ll also find the final version of the game in the file Games/Chapter02/evil3.gm6, on the CD. When you’ve finished a game, you can turn it into an executable by choosing Create Executable from the File menu. Executables don’t need Game Maker to run, so it’s easy to give them to your friends or put them on a website.

Now that you’re a bit more familiar with Game Maker, why not try making some changes to the game to see what effects they have? You could add new objects to the game—there’s an image for an “evil baby” in the resources directory that you can use. Perhaps these could be demons in disguise? Also try changing the movement speeds of the different objects. This can have a big impact on the difficulty of the game, as can changing the number of sides on the dice in the Test Chance actions. Balancing the settings for all these parameters is one of a game designer’s most important jobs, and we’ll talk more about this in Chapter 11.

This chapter has introduced you to the basic elements of Game Maker. We’ve looked at different kinds of game resources and seen how events and actions are used to control the behavior of objects. However, we’ve only just scratched the surface; there is still much more to discover about Game Maker and lots of even better games to make. Our journey continues in the next chapter with a trip to a moon or two as we learn more about events and actions by playing with spaceships.