A Game Design Vocabulary

Exploring the Foundational Principles Behind Good Game Design

Anna Anthropy
Naomi Clark

Addison-Wesley

Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

CHAPTER 3

SCENES

Every game is made up of rules, which we've divided into verbs and objects. We've talked about how important it is to develop these game characters, but how, and where, do we do that? We do it with scenes, which is what we call the units of gameplay experience that unfold during and create the pacing of a game as it's played. Our players are performers, and naturally, we can't always anticipate how they will perform every given scene. But when we create a game, we have the capacity to shape those scenes and frame the choices the players get to make.

Rules in Scenes

In the previous chapter, we introduced the main characters in our story: the player's verbs, which are the rules that allow the player to interact with the game. In this chapter, we're going to talk about the ways we develop them. Our verbs and objects—the two elements of our game that comprise our system of rules—are the actors in the story. These actors perform in *scenes*.

Depending on the game, the speaker, and the position of the stars, we might call scenes levels, stages, rounds, waves, boards, missions, or screens. They're not necessarily equivalent in time, content, or presentation, though. A scene is the most basic unit of pacing in a game.

Every game might use completely different sizes and shapes for scenes. In fact, some games might have only a single scene. That's fine—it just means there isn't going to be a lot of designed character development. Character development might be the development of the player's understanding of the rules and how they interact. Consider the most basic games of the 1970s, such as *Breakout* and *Pong*. In both these games, the player moves a paddle back and forth to intercept a ball and send it back to the other side of the screen, toward another player's paddle (in *Pong*) or a wall of bricks that can be broken (in *Breakout*). These games work just fine with only one scene, and the verb—moving the paddle to hit or miss the ball—is developed as the player practices and comes to understand the ball's trajectory.

Often, a game can be broken into scenes that are even smaller than its own self-demarcation, if we accept that a scene is the most basic unit of pacing. *Super Mario Bros.* has a world 1-1, a world 1-2, and a world 2-2. Within world 1-1, we can identify and talk about a number of different scenes: the part with the monsters and pipes, the part where Mario climbs the stairs and jumps over the pit, the part where he jumps and tries to grab the flagpole. In each of these parts, a different kind of development is going on, so it's useful to consider each one a scene of its own. A scene is a more atomic, fundamental unit of gameplay than a level, or a world, or a region in a game world.

How can scenes develop a verb? Let's visualize, in our enormous brains, a game whose protagonist is from the future, wild though the concept may be. In the future, every citizen is equipped with a personal teleporter. Just think about going to a place, and you're there. Naturally, in this golden age of laziness, human legs have atrophied to the point that they more closely resemble the rockers on a rocking chair in appearance and function.

The player's verb is "teleport," and the physical layer is the mouse. Click somewhere, BING! The protagonist is there (see Figure 3.1). It's a simple game.

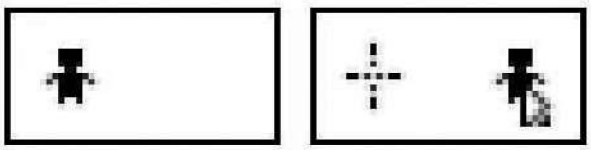

Figure 3.1 Teleportation as a verb: just click and you're there!

What situations—or scenes—could we engineer to develop this verb? Well, we need a reason for teleporting to be important. So now we have our first object. It's an electrified force-field, the kind of thing that appears everywhere in the future. The player doesn't want to touch it because it looks dangerous. We'll talk more about using context to explain the rules of the game in Chapter 4, "Context." Just know that there are sparks coming off this thing; its every inch sizzles with a dangerous green, clearly unfriendly to life.

This force-field is moving toward the protagonist, from the top of the screen to the bottom, and it stretches the entire width of the screen. Now teleportation has purpose: the player has to teleport to get to the other side of the force-field safely (see Figure 3.2). Maybe a food capsule appears on the other side of the force-field, if the player needs a stronger hint. Those futuristic food capsules are delicious! When the player teleports to the food, she's also teleporting past the force-field and to safety.

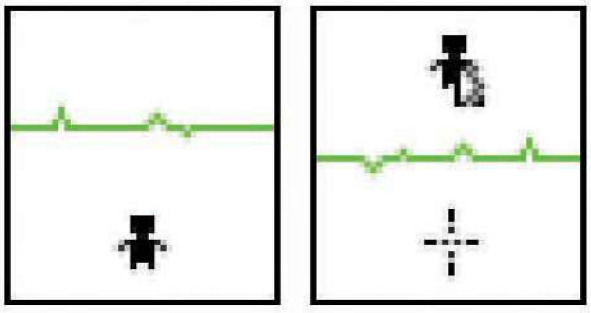

Figure 3.2 Using teleportation to dodge a deadly fence.

That was a first scene in a story about teleportation. What could be a next scene? How do we develop teleportation further? Let's add a stronger element of timing to teleportation. The next deadly fence—we'll have it come from the bottom of the screen this time, since the first

one forced the player to warp to the top—might be really, really thick. In fact, it might be so thick that not all of it fits on the screen at once. The player has to wait until it's all on the screen before she can teleport to the safe space beyond it. Its near side is closing in on the player at the top of the screen when she finally gets her opportunity (see Figure 3.3).

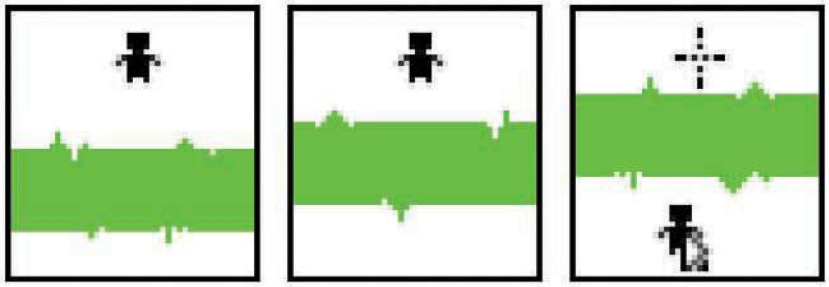

Figure 3.3 Timing the teleportation to dodge a thicker fence.

If we use timing to develop our "teleport" verb further, what does a later scene look like? Imagine an electric fence that covers the entire screen, with holes that appear long enough only for the player to wait for the next hole to appear onscreen (see Figure 3.4). How important is timing to teleportation in that scene?

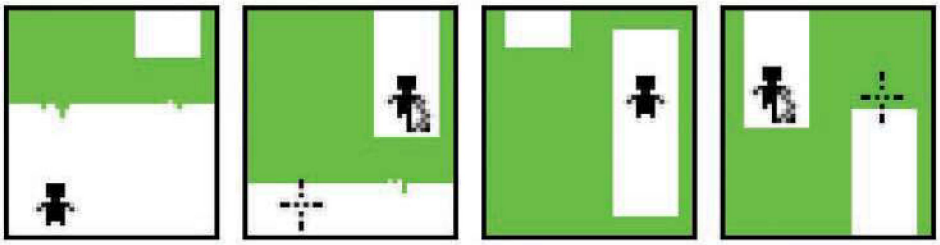

Figure 3.4 Timing is critical when jumping from hole to hole in the electric fence.

Each scene revolves around a particular development of the game's verbs. A scene can introduce a new development, like the previous examples, but it's often useful to revisit how to use a verb, or what aspects of using the verb to focus on (like timing) again and again—just as recurring themes often appear in a written story or a piece of music. As we add scenes to the game, we can also overlap these ways of developing a verb to create a more complex and rich experience for the player.

The ordering of these scenes is crucial, as you might imagine. In our teleporting game, it probably makes sense to show the player how to teleport past a nonmoving force-field first, then a thin force-field, and eventually a thick one. This sequence creates the *pacing* of the game as well as its *resistance*—concepts we'll be discussing later in this book.

The Cast

The purpose of scenes is to introduce or develop rules, to give a chance for the game's cast—its verbs and objects—to shine, and a chance for the player to understand something new about them. Objects are the building blocks of scenes. While verbs are often our main characters throughout much of a game (at least if they're robust!), the selection and arrangement of objects in a scene are often what makes the scene unique. They're our most basic tools for creating choices for the player and setting up encounters between verbs and objects. The timing of an object's appearance on stage is important.

In a game called *Ducks* (www.glorioustrainwrecks.com/node/3833) by Nick Scalzi, the player guides a mother duck around a pond: point the mouse at a position, and hold the left button to move toward it (see Figure 3.5). The verbs are simple: the player can move the mother duck around the pond. Her five ducklings follow in a line behind her, repeating every motion she makes.

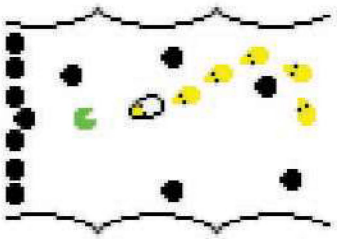

Figure 3.5 Layout of the pond in the game *Ducks*.

In the game's first scene, the mother duck swims around the pond and collects food for her ducklings. The objects here are the rocks in the pond that make impassable objects the ducks have to swim around and the moving lily pads that represent food. You can see the layout of this pond in Figure 3.5. It's a mostly open space with a few scattered rocks. There's a wall of rocks along the left border of the screen, and the banks of the river on the top and bottom.

Food lily pads drift onto the screen from the left and right. They represent moving targets—things that the player wants to catch. They're prompts; they give the player an incentive to "move" and, in doing so, to learn how the mother duck moves. The player also learns how the

baby ducks move behind the mother duck. The player brushes against a rock and learns that she has to plan paths around rocks.

After collecting several pieces of food and gaining a basic familiarity with the main verb and its limitations, something changes. A new scene introduces a sudden new object.

The new objects are bullets, fired downward from the top of the screen, beyond which, we presume, a hunter lurks on the riverbank. Suddenly the movement of the ducklings behind their mother is important—something the player has to pay attention to. As she tries to evade the hunter's bullets, she may also try and keep the ducklings out of the way of those bullets. The player's "move" verb now has an added responsibility: it's become a more developed character in the game's cast of verbs. By stopping at the right time, the player can position her ducklings so that the bullet will pass harmlessly through the space between one duckling and the next. Naturally, this is easier when moving horizontally than vertically, something the layout of the scene is designed around.

Figure 3.6 depicts the getaway scene, the length of the river to the right of the pond. The river to the left of the pond is closed off by rocks, remember. A big arrow flashes on the screen when the bullets start to fire. This book began with a complaint about the use of arrows to indicate where players should go in a game, but given the suddenness of the attack and the panic it induces, it's clear the author felt it was most fair to make absolutely clear where the path of escape lies.

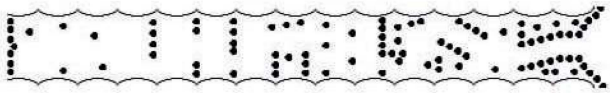

Figure 3.6 The player navigates around objects to escape the scene.

The path is constructed out of rocks, the game's most basic object. You can see from the figure that the first significant obstacle, near the left side of the screen, is a wall with a gap at the bottom. The next wall is fairly wide open for maneuvering before a third wall with a narrow gap that's slightly higher up. Why does the gap start at the bottom and then move up? Because the bullets are firing from the top. The purpose of this sequence is to gradually escalate the tension between moving and keeping your ducks out of the path of the bullets.

The bullets, not just the rocks, are a critical part of this scene. Both objects drive the development of the player's verb forward, and their relationship gets much hairier toward the close of the game. Eventually, at the right end of the river, the space the mother duck has to navigate and dodge bullets in becomes greatly constrained. This progression develops the relationship between the danger of the bullets and the player's main verb: "swim."

Making Introductions

One of the most important responsibilities of a scene is not just to develop a rule but to introduce it. The introduction of any new rule is critical. This is the player's first encounter with that rule or object, and what the player takes away from this encounter will inform every future encounter. It'll dictate her expectations every time she sees that object. What we as creators want to be sure of is that she understands the rule and its implications as completely as possible. If we want the player to be totally intimidated by a scene starring a dangerous object later in the game, we'd better make sure she leaves her first encounter with that object convinced of its danger.

We also want to teach the player this as efficiently as possible. The player's most important resource is time. The player can discover an object is dangerous by having it hurt her, but if that means the player is forced to repeat the last five minutes of the game, we've just wasted five minutes of the player's time. (We'll talk more about repetition and punishment in Chapter 6, "Resistance.") In the space mining game discussed at the end of the previous chapter, one successful enemy attack means that the expendable mining slave has been killed, and the next expendable slave is sent in her place. In other words, the player has to try the scene again from the start. This being the case, we want to avoid teaching the player by death.

You might recall some of the creatures we designed for the space mining game in the previous chapter: a monster that falls when the ground supporting it has been destroyed, and a robot turret that shoots bullets at the player when she's nearby. There were various other interactions we built using these creatures; the turret's bullets could destroy the earth beneath a rock monster, causing it to fall. But the player can't anticipate these interactions if she doesn't understand the implications of these objects in the first place. When designing a possible first scene for our game, how do we introduce these objects and what they mean?

For starters, we have to introduce the player's primary verb: the "bomb." Specifically, what we need the player to know about it is the duration of its fuse and the radius of its explosion: the specific characteristics of the verb that will guide all the player's choices about when and where to place a bomb. The solution for that—at least, for now—could be a round room with thick walls and just enough room for the player to stand outside the explosion of a bomb placed next to the walls (see Figure 3.7). There are no other creatures or items in this room. It exists just for the player to try a bomb and to get a sense of how far away and how long she has to stand to avoid being caught in the explosion.

It's important that the player develops a sense of how objects work, because the introduction of a falling rock creature hinges on this knowledge (see Figure 3.8). It's important that the player sets off this first falling rock guy with her own primary verb: her bomb. That's because the player needs to be conscious that she has the choice to release these creatures by herself (and

eventually to exploit them as weapons). But the creatures will fall quickly. This is a characteristic we decided the falling creatures should have to make it easier to line up other creatures with them and use them as weapons. We want the player to cause this creature to fall, but not to be under the creature when it falls.

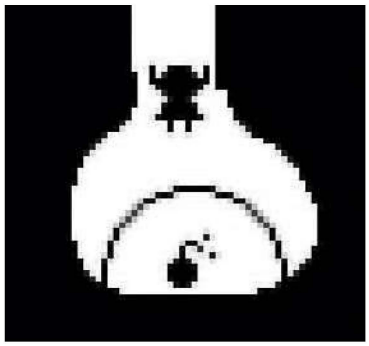

Figure 3.7 The player needs a safe place to test how objects work.

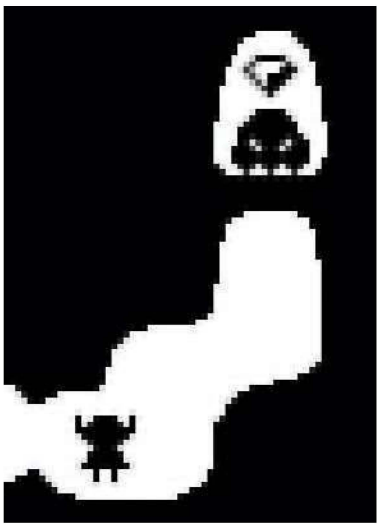

Figure 3.8 The player should be able to use previous knowledge to figure out how to interact with new objects, like falling rock creatures.

And so we exploit the player's existing knowledge of the radius and danger of her own bomb, things she has already learned. We place the first falling rock creature in a small chamber at the end of a winding corridor. The final stretch of the corridor—the part vertically below the rock monster, and thus in range of its falling attack—is too short for the player to share it with

a bomb without being caught in the explosion. So the player has to drop the bomb and then go around a corner to get outside the range of that bomb—also moving her out of reach of the falling creature. When the bomb goes off, she can observe the creature's behavior—that it falls, where it falls, and how quickly it falls—from a safe distance.

Why does the player want to bomb and release the creature in the first place? If you remember our previous discussion of this game, the player's stated goal is to collect crystals. As you can see in Figure 3.8, there is a crystal in the small chamber behind the rock creature.

We introduce the robot turret the same way. We let the player observe its behavior in relative safety, but from a close position, so that she can understand her own relationship to the creature's behavior. Just as with the rock monster, we want the player to understand how her own verbs interact with the turret so that she can eventually exploit its behavior and use it as a weapon or tool. In this case, though, the verb that the turret interacts with isn't the bomb (because the turret itself can dig through the ground, using its bullets) but simply "walking." The turret awakens, visibly unfolding and opening its mouth, when the player walks into its range of fire, and the turret goes back to sleep when the player leaves that range. It fires its bullets toward the player's current position (see Figure 3.9).

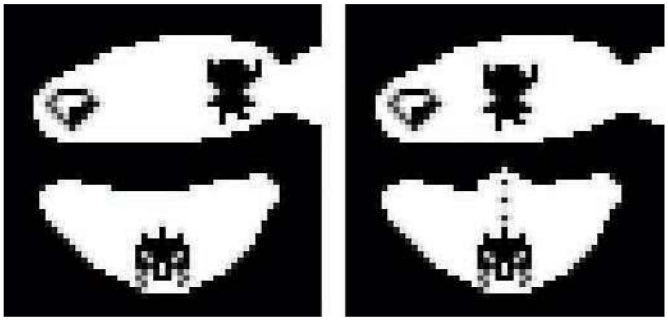

Figure 3.9 The player learns about the robot turret from a safe position, since its shots will hit a wall first.

The bait, again, is a crystal. It's at the far end of a short hallway that's within the turret's range, so when the player enters the hallway, the turret awakens. The wall is thick enough that it'll take several rounds of fire from the turret to break through and actually hit the player—giving her ample time to get the crystal and leave, especially if she's in a hurry (which she likely will be, because of the turret).

While the player is grabbing the crystal, she has the opportunity to learn things about the turret. Specifically, she learns how often it fires (currently, three bullets in a quick burst, then a short pause, then another burst), where it fires (at the player's miner), and what the effects

these bursts have on the terrain of the level (each bullet destroys a small piece of dirt that it hits). By the time the player grabs the crystal and gets out, she's learned quite a lot.

Subsequent scenes develop the relationship between these objects and the player's verbs that were introduced here, by allowing the player to actually use these objects as digging tools and weapons against other creatures.

Performance and Expression

Players should be able to make various choices about how to interact and move through a game. But isn't it a contradiction to talk about designing choices for a player while at the same time talking about how a scene should play out in a specific way? To talk about telling a story when we can't predict what our lead performer, the player or players, is going to do? Storytelling in games is like storytelling in theater: that's why we use the word *scene*.

In theater, we write a script, but the actor's performance of it is up to her. But we nevertheless compose the shape of the scene. We give the player (another theater word) liberty to perform a scene; this is where the choices come in. The choices that a player makes come from the verbs we give her, so we have the ability to constrain and design what those choices are. A fictionalized ideal of the videogame is that it gives the player the ability to do anything, to choose anything. There are games that aim for this kind of wish fulfillment and mostly result in weak experiences, lacking cohesion and focus.

We create choices that serve our stories using the verbs the player has access to. *Bioshock* (2007) is a game about rampaging through an underwater city and shooting objectivists; the player's central verb is "shooting." And yet, a critical part of the game's written story involves periodic choices over whether to "murder" or "rescue" orphaned girls. This is a sociopath's idea of a moral crisis: kill a girl to farm her for game resources, or magically transform her from a zombie girl into a ruddy-cheeked white girl.

Aside from the absurdity of having to frame a "moral" choice as being between the furthest possible extremes of human behavior, the deeper awkwardness of these choices is that they have no connection to the rest of the game. They aren't made using the verbs that the player has already been given to communicate with the game. The player is taken away from the game and presented with an arbitrary choice: press A to rescue, press B to murder. What is the relationship between these choices and the rest of the choices the player is making in the game—exploring, shooting? There is none. Although the game acts as if there's a benefit to "doing evil"—murder gives you more resources to spend on powers—it turns out that rescuing them rewards you with even more resources later on in the game. Even the game's supposed moral calculus undercuts itself.

I put a love story into a quick-draw game I made in 2008, *Calamity Annie*. Annie's verbs are "aiming" and "firing" her pistol. When Annie flirts with her love interest, the player lights her cigarette by aiming and firing (see Figure 3.10).

Figure 3.10 In *Calamity Annie,* the player is given a choice to light a cigarette by firing a pistol.

But most of the choices a player makes in a game are much more subtle. Let's go back to the future, to our game of teleportation, fences, and food capsules. When the player teleports to avoid that first electric fence, she's actually making a few choices. First: she decides when to teleport. When the fence is about to touch her? As soon as there's room behind it? When's the safest moment to teleport? Or should she try to get past the fence as quickly as possible? That's a choice.

Second, where does the player teleport? The player has to teleport somewhere behind the electric fence to avoid getting shocked by it; that's her only requirement. She could teleport to the left side of the screen, the right side, right behind the fence, as far away as possible. She could teleport in front of the fence, realize her error, and quickly click behind the fence to teleport there. As long as she ends up somewhere behind it, all that space is equally valuable.

Well, what if we make some spaces more valuable than others? Those food capsules are a way we can do that. Maybe the game keeps track of how many food capsules the player's collected, as a kind of score. Teleport onto a food capsule to collect it. The capsules are small, so the player will have to teleport to specific positions on the screen to collect them.

Now the player has the choice between teleporting to a food capsule or teleporting somewhere else. Well, of course the player's going to take the food capsule; that's not really much of a choice. So let's make it a choice: let's put another, smaller fence behind the food capsule. If the player wants the food capsule, she has to teleport not once but twice: she has to teleport to the capsule and then teleport away before this second fence can touch her (see Figure 3.11).

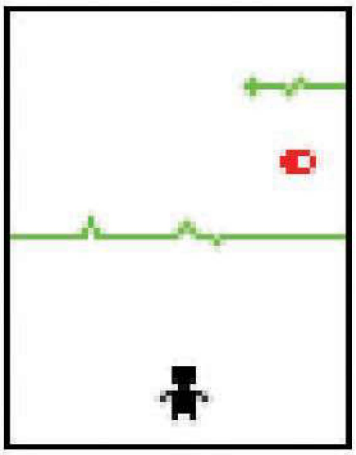

Figure 3.11 The player might have to make dangerous choices to collect some food capsules.

This is an interesting choice: take the easier path, or take a harder path for which you'll be rewarded. Giving the player meaningful choices makes her performance more meaningful. As game creators, we're little Rube Goldbergs, bolting together all sorts of interesting predicaments for the player until we have an interesting system in which she can perform by playing, and in doing so express herself through choices.

Shaping and Pacing

Shape is a word that can have a lot of meaning for us as creators of games. Think about what the shape is of each scene in your game. Remember, we're writing a script for our player, we're setting up the props in such a way as to guide the performance, but ultimately we can't police the details of the player's performance. So we can't always ask, "What happens in this scene?" Instead, we can ask, "What is the shape of this scene?" When we determine that shape, we're also defining the space of possibilities within it, much as an architect creates the space of a room or hallway by deciding where the walls go. The space of possibilities in a scene encompasses all the ways that the player has to move within the game's shape by making choices.

The shape of a scene, accordingly, might look a little like a "floor plan" or a map of the choices available to the player. In the game *Mr. Do!* (1982), the player digs tunnels to get to patches of buried cherries. Monsters then escape into the tunnels and navigate them to catch the player. The tunnels become areas the player must defend, staging areas for booby traps. This pursuit is resolved either by the player collecting all the cherries or by the monsters catching the player (see Figure 3.12).

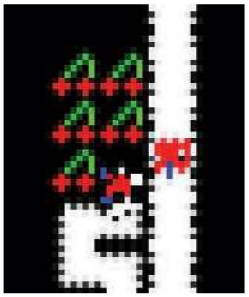

Figure 3.12 *Mr. Do!* is played in a back-and-forth tunnel pattern.

Each of these moments—digging to the cherries and eluding the monsters—has a number of small choices embedded into them that branch into many more choices. Do I dig up? Down? Left or right? Do I build my tunnel in a back-and-forth pattern to ensure that pursuers have to take the slowest path possible to reach me? When they chase me, will I dig under one of the heavy apples in the scene, hoping it will crush them when it drops? When do I drop the apple? Will I wait under it until the perfect moment? Will I dig a vertical tunnel long and straight to maximize the chances of a bunch of monsters being in it when I drop the crushing apple on all of them?

Mapping out the networks of possible player choices in a scene like this would be an impossible task, a combinatorial explosion. But as creators, we can think of the shape of the scene and perceive the relationships between different parts of that shape. We know that the player will dig tunnels and that the monsters will enter those tunnels. Having that shape in mind, we can place objects in such a way as to encourage particular interactions.

Figure 3.13 shows the starting position for the first scene of *Mr. Do!* You can see that every patch of cherries has one or two apples somewhere in relation to it. Apples fall downward if there's nothing under them, crushing monsters and potentially the protagonist, Mr. Do. The lower-left patch is the safest for a starting player to pursue, as Mr. Do starts at the bottom center of the screen. There's space between that patch and the apple above so that the player has room to construct a trap for pursuing monsters after she collects the cherries. In the patch immediately above that one, the apple is directly over the cherries. The player has to take its presence into account *while* she collects those cherries. In designing the rules of *Mr. Do!*, its creators made it

possible for many shapes to emerge—safer and more dangerous juxtapositions of cherries and apples, patterns of tunnels that players can turn to their advantage or be trapped in.

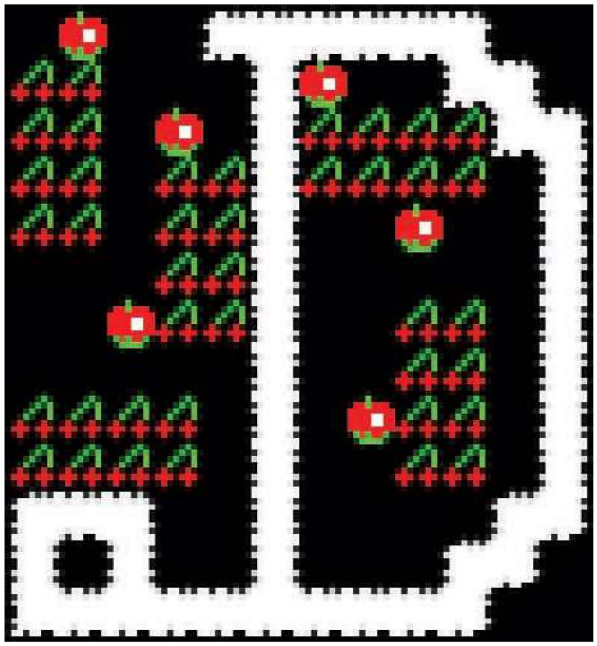

Figure 3.13 The first scene of *Mr. Do!* provides the player with numerous choices.

As we consider the shape of a scene, we can also think how the space of possibilities in that scene changes. Remember the first image in this book, the opening scene of *Super Mario Bros.*, which is shown again in Figure 3.14. The possibility space—the cloud of possible player choices—starts open but small. In this initial open space, a jump in one place or another is inconsequential. Then there's the first monster, whom the player can deal with in one of a few ways—jump on top of it, jump over it, jump while moving, jump straight up and rely on the monster's motion to bring it under Mario—but it must be dealt with. The space narrows.

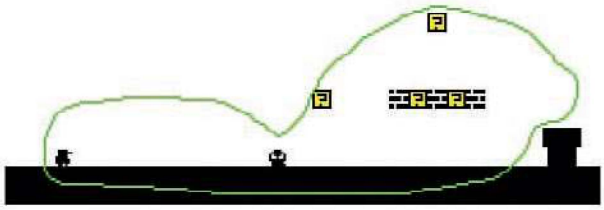

Figure 3.14 The possibility space in the opening scene of *Super Mario Bros.*

Past the monster, the shape opens even wider: there are a lot of tiny choices to make on and in between these hanging platforms. Will the player climb up to the top or the middle? Will the player collect the mushroom? Will she break some of the blocks? Break all the blocks? Just run straight through? And then, after that, it narrows again. Mario has to get over this pipe to see the rest of the game. There are a few different choices again. The player can jump from next to the pipe, from on top of the platforms, she can run and jump, or she can jump from a standing position. But she must prove she understands that she can use her verb, Mario's "jump," to navigate obstacles.

As we change the shape of a scene from beginning to end, alternating wide spaces of choice and narrow spaces, we create the pacing of the game. We can guide the motion of a scene in a particular direction, toward a particular point, without taking away the player's ability to choose. We can open our scenes wide and then slowly narrow them down until the player is performing our script word-for-word. We can choose the shape that is truest to the purpose of a scene. Then we can decide what kind of scene comes next and how the shape of that scene will relate to and continue from the scenes that came before. Will the next scene be more difficult, pushing back against the player's desire to continue? Will we develop new verbs, opening more possibilities?

We can also think of the shape of a scene in terms of how it is presented visually, how it leads the eye around and draws the player's attention to the most important elements of a scene. We talk about that in Chapter 4.

Scenes with Purpose

For every scene in your game, you should be able to answer two questions: what's the purpose of this scene, and how can you accomplish it using established game vocabulary? If you want the player to feel tense in this scene, what rules and objects are in the game that will allow you to create that feeling? Are there objects you can use to add an element of timing, for example? As we saw with the examples of the electric fence at the beginning of this chapter, timing can also help develop a verb. Are there aspects of a verb or other rules that a scene could help the player understand? When we fail to tell the story we want to tell using the vocabulary of our game, remember this: that's when we resort to using devices that have no connection to our game, like movies with no player interaction.

The game *Condensity (*www.newgrounds.com/portal/view/598331) is about guiding a group of water droplets, who all move as one but may be in different positions on the screen, to an exit that they all must occupy at the same time. This is the critical idea of the game: that the water drops may be navigating spaces that are arranged differently, with different configurations of obstacles, using the same instructions from the player. To emphasize this disparity, the droplets can be transformed (by heated or chilling elements) between two different states: the liquid state, which is affected by gravity, and the gas state, which is able to fly (see Figure 3.15).

Naturally, it is possible to have droplets that are in liquid form and droplets that are in gas form at the same time, which makes coordinating them trickier.

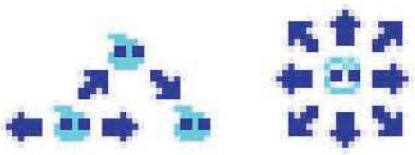

Figure 3.15 Liquid and gas states of the water droplets in *Condensity*; a liquid droplet will fall, but a gas droplet can move in any direction.

Droplets in gas form have properties that are different from the liquid form. There are fan objects that blow them around, pushing them relentlessly in a single direction. Usually these fans function like gates: to pass a fan, a gas droplet has to find a way to become liquid, or a liquid droplet needs to avoid being transformed into gas. But one scene (see Figure 3.16) finds another use for the fan: it propels a gas droplet helplessly along a path while a water droplet below, steered by the player, tries to keep up so that they can meet at the exit (where the gas droplet is transformed to water, and will fall past the exit unless the other droplet is there to catch it).

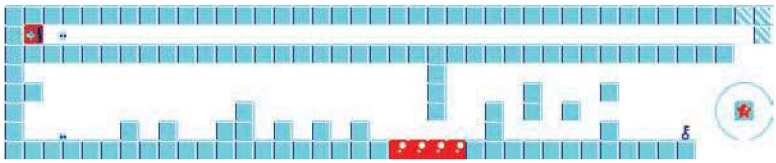

Figure 3.16 Introducing timing in a scene puts pressure on the player.

This introduces an element of timing—of racing, of struggling to keep up—into a game that doesn't normally have that kind of pressure. And it does it using rules that are already established. The player understands the interactions that conspire to create this situation for her; she knows why she's running.

Remember the game *Tombed* we looked at in Chapter 2, "Verbs and Objects"? The one with Danger Jane and the descending spiked wall? Figure 3.17 is an early scene from that game. The purpose of this scene is to make the player aware that she can use the spiked wall's destruction of objects as a strategy. Rules that have been established so far: "soft" blocks, the cyan, blue, and green, can be dug through. Metal blocks cannot—except by the spiked wall. It's the latter rule that we want to develop in this scene to give it a relationship to the player that's nontrivial.

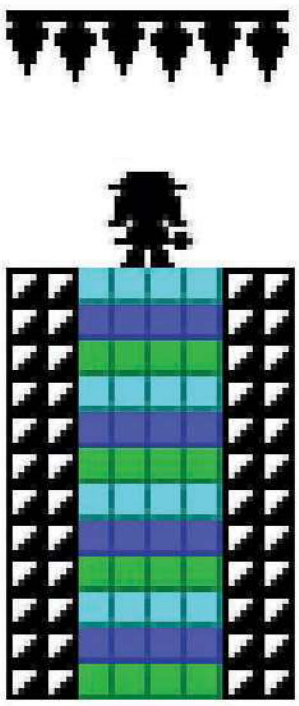

Figure 3.17 An early scene in *Tombed* shows the player how to use spiked walls strategically.

How do we develop a rule like this—one that's not a verb, but which governs the relationship between objects like the spikes and the various kinds of blocks? By giving the player a choice. Chased by the descending spikes, Jane has to dig through the soft blocks in the middle because she can't dig through the metal barriers on the sides. That's not much of a choice.

The choice comes as the spikes grow closer—and destroy the metal barriers. Once the spikes touch any part of a continuous shape, remember, the whole thing crumbles instantly (see Figure 3.18). Jane is likely somewhere in the middle of the soft-block column at this point. There are many thin layers of soft blocks, so digging through them is slow work. In fact, it's so slow that the spikes will almost certainly catch up to Jane should she continue this route. (It might be possible to dig through all the soft blocks, if one's timing is good enough to dig at a rapid pace. But it's not easy.)

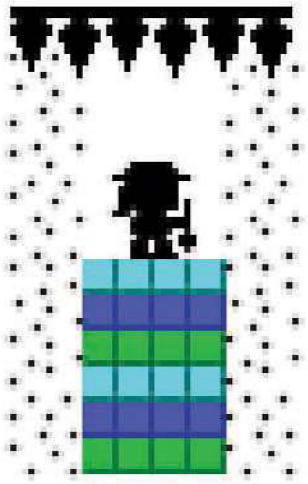

Figure 3.18 The metal blocks crumble as soon as the spikes touch them.

The solution is to walk left or right off the soft blocks into the open columns that have been cleared up by the crumbling of the metal barriers: to become cognizant of the fact that the spikes' transformation of the game world can be exploited by the player. The player's choice here is whether to keep digging or to escape to the side. One of these choices seems right and the other wrong, but making the "correct" choice requires cognition of the fact that destruction by the spikes presents opportunities for the player, and having to make a choice based on that knowledge helps the player internalize the rule more than merely observing the rule would.

Later in the game, other scenes revolve around the player's understanding that some of her options come from the spike wall's destruction of other game objects. Early in the game, this scene's purpose is to develop the "spike wall destroys things" rule to the point where that's clear, to establish the player's understanding of that rule.

Layering Objects

I learned an important lesson in 2005 while tinkering with one of my first games. The game started out being about a pig that projected her astral form into a higher plane to somehow disrupt the machinations of slaughterhouses and ended up being about a squid in a pond being pursued by fish. The game is called *Pond Squid* (see Figure 3.19).

In the space of this small pond, the player has to keep the squid out of the reach of the relentlessly pursuing fish. Eventually (though unpredictably), she gains opportunities to trap fish and use them as projectiles to remove other fish, who by this point have consolidated into a giant lump that is chasing the squid around.

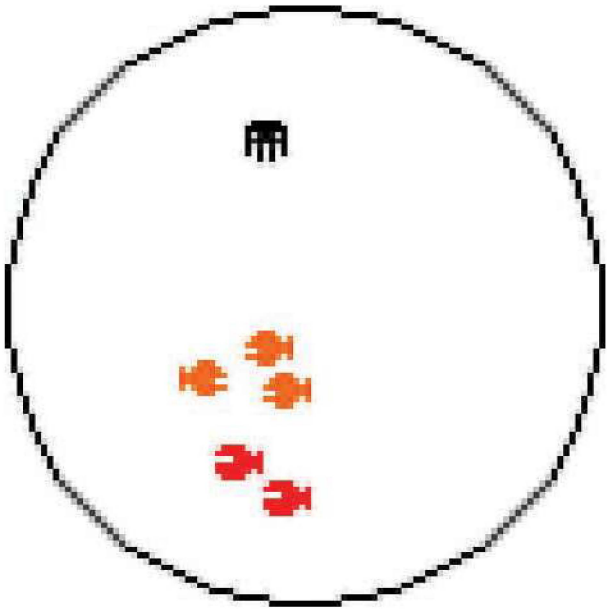

Figure 3.19 Fish pursue a squid in *Pond Squid*.

All the fish follow a really simple rule, you see. They just move directly toward the squid. So though their numbers grow continuously, they all merge into a single cloud that basically moves as one. Now, at that point I wasn't the Baba Yaga of game design I am today, but I realized the solution called for another object, another kind of fish. The fish that I added followed the same rule for pursuit as the first one—move directly toward the squid—but at half the speed.

And that worked, or it would have if I hadn't made the type of fish that appears totally random—meaning the player could see fish that move at the same speed for a really long time without ever seeing a slower fish. But when there's a mix of fast and slow fish, the game is much more interesting to play than when there's just fast fish or slow fish. You can see similar design in "bullet curtain" shooting games—different layers of bullets that are moving at different speeds at the same time are much more complicated to navigate than a field of bullets moving at the same speed.

The lesson is that *layering* is important. Having objects that stack in interesting ways creates more interesting choices. Six years after I made *Pond Squid*, I was confronted with the importance of this rule again while working on *Lesbian Spider-Queens of Mars* (http://games.adultswim.com/lesbian-spider-queens-of-mars-twitchy-online-game.html). In this game, a Martian spider-queen pursues her escaped slaves through a maze, attempting to zap them with a bondage laser and recapture them. The slaves are armed, so if one of them manages to get

the drop on the queen—sneak up on her without getting zapped by the bondage laser—the slaves can turn the tables.

The most basic kind of runaway slave simply picks a new direction at random every time she approaches an intersection (without doubling back—see Figure 3.20). This slave makes up the bulk of the queen's opposition in the game—since she chooses at random, she doesn't always move toward the queen, giving her opportunities to sneak up from behind, and she may avoid opportunities to really corner the queen. She's more difficult in greater numbers—they fill more space, all picking different paths—but she's easily manageable in small numbers.

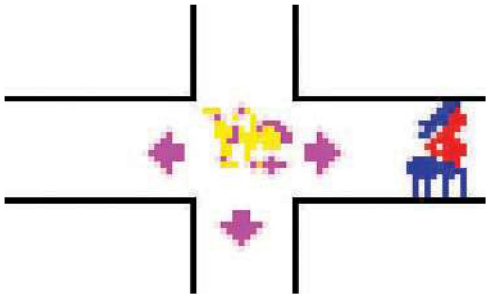

Figure 3.20 Simple slaves pick random directions in *Lesbian Spider-Queens of Mars*.

Midway through the game I wanted to introduce a smarter opponent. This one is called a gladiator. Whenever a gladiator reaches an intersection, she looks at the queen's position in the maze relative to hers and picks the direction that will lead her closer to the queen the quickest (see Figure 3.21).

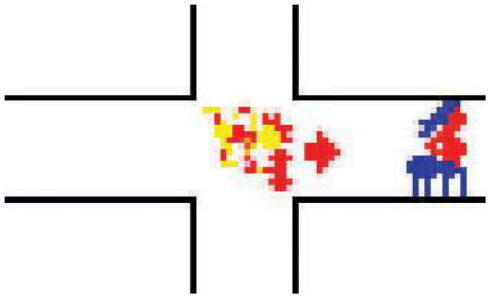

Figure 3.21 The gladiator picks a direction that leads her to the queen.

Originally, I had imagined gladiators becoming the replacement for the original slaves in the later scenes of the game—a harder-to-deal-with version of the same behavior. That's what I thought. In practice, though, it became apparent that they weren't as overwhelming in large numbers, that they didn't gain as much from numbers as the original slaves. You see, they

were predictable: since they always chose to move toward the queen, if a bunch of them were coming from the same place, they'd all march toward her in a line, blundering straight into her bondage laser.

It was much more interesting, I discovered, to mix the random-moving slaves with the queen-chasing gladiators. That made for much more complex situations and much more meaningful choices.

Having objects that complement each other is important to constructing situations. Look at *DOOM* (1993). The game introduces early on a monster called an imp that throws fireballs at the player (which travel in a straight line) and a monster called a pinky demon that attacks by biting and takes circuitous paths to reach the player to try to dodge her fire. Many of the most precarious scenes in the game require the player to manage these two different avenues of attack simultaneously.

For the mining game discussed previously, we wanted to be sure to design objects that could be combined to create interesting scenes and meaningful choices. For example, in Figure 3.22, the turrets potentially set off a chain reaction of falling rock creatures that ultimately destroys the turrets and clears the way to a bunch of crystals, but the player must dodge both the falling rocks and the turrets' bullets. The player could set off the chain reaction with her own bomb. There are many ways, involving conservative and risky choices, to play out this scene.

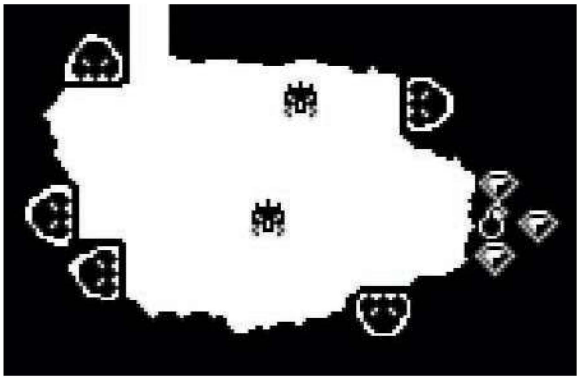

Figure 3.22 A scene in the mining game with a combination of turrets and rock monsters.

This scene doesn't appear until after both the turrets and the rock monsters have been introduced individually (something we did in earlier scenes). The player is unable to make those choices—conservative versus risky, whether to initiate the chain reaction with a bomb, with a bullet, or try to avoid it entirely—unless she first understands how the individual objects—the rock monster and the turret—work. Without that knowledge, based on earlier experiences, this room is a trap, not an arena for player choice.

Moments of Inversion

An encounter with an overwhelming force, a sudden shift in the focus of the game, is a *moment of inversion*. Moments where a rule in the background suddenly comes to the foreground—where a hunter becomes the hunted or vice versa—these are moments of climax in the pacing of a game. Thinking about the shape of a scene, moments of inversion make a scene's shape more dynamic.

Wizard of Wor (1981), the game that inspired *Lesbian Spider-Queens,* is paced by regular moments of inversion. *Wor* is a game in which two human players ("worriors") attempt to survive mazes full of monsters—the arena of the titular villain, the Wizard of Wor. The mazes are symmetrical labyrinths of walls and corridors; a marked door on either side allows a player to "wrap around" to the other side of the maze. A number of monsters patrol each maze, stalking the players. They can fire bullets at the worriors and are invisible when a worrior doesn't occupy the same hallway. There's a radar on the bottom of the screen—a smaller map of the maze—that shows the positions of all the monsters. The players have to rely on it to track hidden monsters. The monsters start much slower than the players and gradually pick up speed until they're much faster.

Fending off the horde of monsters is tricky. The worriors are outnumbered—monsters come from every direction, and, as in *Space Invaders*, only one bullet is allowed on the screen at a time. That means missed shots can be disastrous, and a worrior who acts rashly is easily overwhelmed. Worriors can also shoot one another—by accident or on purpose. (The game awards a big point bonus to a player who shoots the other, to add temptation to the dynamic between the players.) Players often split up, working different parts of the maze, finding advantageous positions and working to defend them from enemies often coming from many different places. Figure 3.23 depicts an average maze in *Wizard of Wor.*

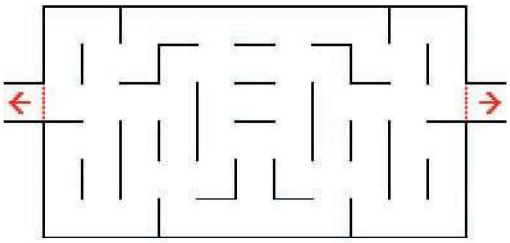

Figure 3.23 Worriors fight in a maze in *Wizard of Wor.*

When the last monster is destroyed, something happens—suddenly. The Worluk, the wizard's pet, appears somewhere in the maze. This creature will not shoot at the worriors. The Worluk

is always visible on the screen. It attempts to reach one of the doors—the wraparound doors on the left and right side of the screen—and exit the maze. Now the players, after fending off a huge mob of aggressive monsters, have the opportunity to be hunters. The Worluk can kill a worrior on contact, but it doesn't pursue him—it just tries to reach an exit door quickly.

There are two exit doors. The Worluk moves much too fast for a single worrior to pursue it. Hunting the Worluk requires teamwork. The reward for catching the Worluk, in fact, benefits both players; it causes all points to be doubled for both players in the next maze. The players outnumber their enemy for the first time and are given an incentive to work together.

Shooting the Worluk sometimes incites the Wizard himself—who spends most of the game taunting the players vocally, unseen—to appear in the maze and attempt to kill a worrior as punishment. Successfully shooting the Wizard is a difficult task, but it's one of the greatest moments of role reversal in the game.

The pace of *Wizard of Wor* is defined by the contrast between these long scenes of the players defending themselves against many enemies and these fast quick ones where they have the opportunity to strike back and be empowered, possibly to humiliate their antagonist. Were it not for these moments—if there was no Wizard or Worluk, only endless mazes and monsters—the game would have no changes in pacing; the space of possibilities would remain the same throughout. It would be a single, unbroken scene, essentially. The inversions that happen in the Worluk scenes not only give a narrative shape to each maze—they mark the end of a scene and the release of the tension that has been building in that scene—but give the game a more dynamic shape (see Figure 3.24). Be slowly hunted for a villain's amusement, have a brief, frantic opportunity to strike back at that villain. We talk more about reasons to give the player a "change of pace," to switch things up in the shape of the game and thus the space of possibilities, in Chapter 4.

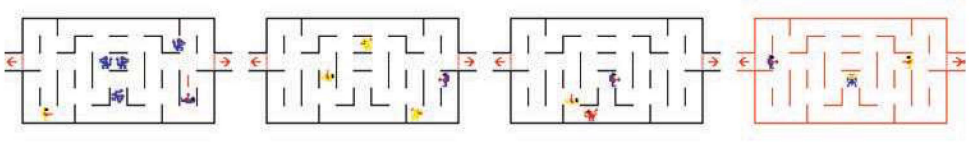

Figure 3.24 Timeline of a single maze in *Wizard of Wor*, with the Worluk battle in the rightmost frame.

Chance

Since we're working with computers, it's easy to incorporate *chance* into many different areas of design. It's hard for a computer to produce a truly random number, but it's capable of generating what are practically random numbers of any size on the fly. A game creator can use these to many different ends.

Chance's usefulness in scene design is to shift the focus of the play onto the rules themselves, not on how they're encountered. Randomness works best when the player is engaged with a complicated network of rules; it can be used to provide myriad combinations of those rules and objects. Randomness isn't a shortcut to design. Its function has much less effect when dealing with more succinct rule sets. Chance is best used for combinatorial ends.

Take the game *Crosstown* (2009), an Xbox Indie Game that's a descendant of *Wizard of Wor*. The players—up to four—explore a maze filled with creatures, attempting to collect four power objects. There are more than 15 creatures in the game's cast, which all interact with the players and with each other in different ways. One builds walls, one destroys them, one only attacks other monsters (not players), one seeks out and defends the objects the players are seeking.

The interactions between these creatures are the center of the play. The layouts of each maze are important only in that they force the creatures—and the four players—to interact. As a result, each maze is a simple, randomly drawn pattern, with left–right symmetry and lots of branching paths (see Figure 3.25), that allows for lots of interaction but also for individual encounters to be isolated to different hallways.

I played an early version of *Crosstown* in which the mazes were much bigger. The creatures and the players didn't interact as much because of the larger space. When the author made the mazes smaller, the game started to really come together—characters interacted much more often.

Crosstown doesn't have much to gain from designed mazes. *Wizard of Wor*, which has a smaller cast of characters that don't interact with each other, only the player, uses designed mazes. Some of those mazes are in fact moments of inversion: the arena scene, which features a large open, wall-less area in the center, and the pit, which is entirely wall-less. Because walls are so important for cover, these scenes represent major upsets for the player.

But chance doesn't have to be all or nothing. Consider the Worluk in *Wizard of Wor*. When it appears in the maze at the end of a scene, it does so at a randomly chosen position (see Figure 3.26).

What would happen if the Worluk appeared at the same position every game? The players could anticipate its appearance and be ready for it each time, defusing its potency as a situation that requires the players to collaborate.

So chance has the capacity to break stagnation in a game. The monsters in *Wor* appear in different locations every time, ensuring players can't simply memorize each scene. This shifts the focus from learning the scenes to dealing with the monsters. The Wizard himself, when he appears, teleports to random positions, making him less like a foe that can be outfoxed than a force of nature to be endured. The rules of interacting with monsters, with staying alive, become more important, not the particulars of any one encounter.

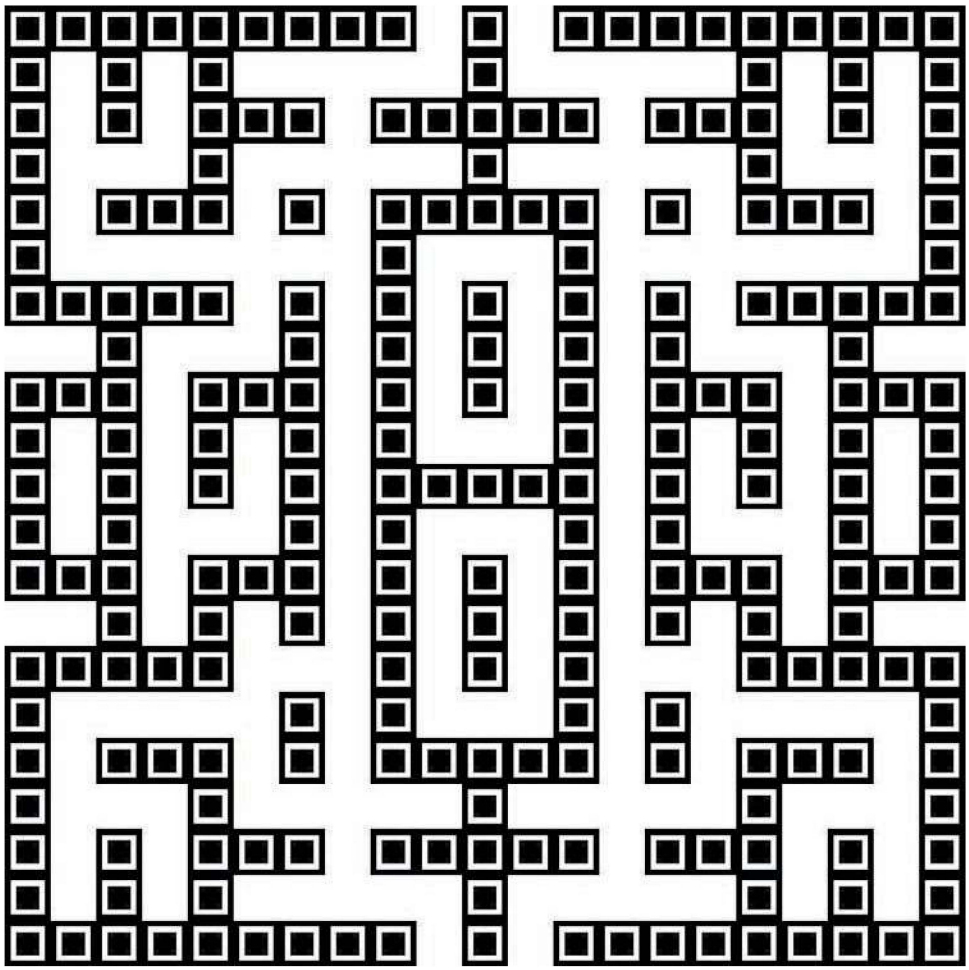

Figure 3.25 A typical *Crosstown* maze layout is a simple random pattern.

Chance is also useful for breaking symmetry. In Michael Brough's *Glitch Tank* (2011) for the iPad, two players attempt to destroy each other by choosing commands for robot tanks. Commands include "turn right," "go forward," and "fire a laser." But the commands available to each player—four at a time—are chosen at random like cards from a shuffled deck (see Figure 3.27). So an ideal move isn't always available, and players often have to choose and incorporate into their strategies less-than-ideal moves.

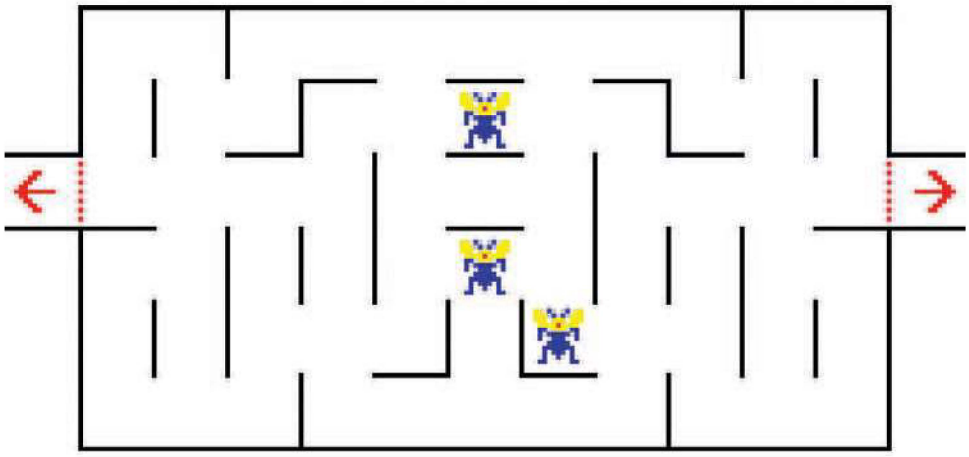

Figure 3.26 The Worluk appears at one of these locations, randomly chosen, in *Wizard of Wor*.

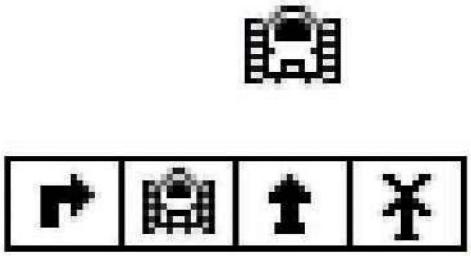

Figure 3.27 The player is given a set of four random commands at a time in *Glitch Tank*.

The effect of this is that it's impossible for both players to play the same series of moves. If that were possible, the result would be stagnation: the player who played fastest would have the advantage. And the game isn't about who can play fastest, although thinking fast is an important skill in *Glitch Tank*. The focus of the game is on outmaneuvering one's opponent with the resources one is given. Chance, in this case, maintains a dynamic between the players by preventing them from performing symmetrically.

Real Talk

Let's talk about how a scene I designed went from conception to its final version. The scene is from the game *REDDER*, which I made in 2010. *REDDER*'s protagonist, Hannah, is an astronaut

who lands on Mars when her spaceship's supply of fuel gems is extinguished. To escape, she must find replacement gems scattered throughout the ruins of an abandoned civilization—abandoned by life, that is, but its electronic defenses are still online—electric fences still sizzle, robot guardians still patrol.

Verbs, primary: Hannah can "move" left and right, on the ground or midair. She can "jump" to four-and-a-half times her height, which means she can "climb" anything up to four blocks—a block being a solid object of varying appearance as tall as Hannah and slightly wider (see Figure 3.28).

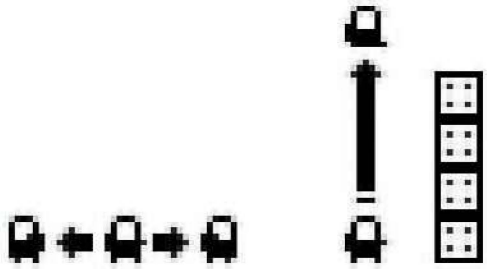

Figure 3.28 The primary verbs of *REDDER*.

Verbs, secondary: Hannah can "trigger" switches that open or close off her way forward. Passage through the old Martian crypts and laboratories is regulated by gates that are more like electronic walls. These things come in two states: extended and receded. When they're extended, they function as solid blocks, serving as wall, floor, ceiling—physical obstacles Hannah cannot pass through. When they've receded into the background, they're as intangible as air, solid space that Hannah can pass right through.

Electronic walls come in two colors: red and green. One color always is extended and one receded. Touching a green switch makes green walls recede and red walls extend: touching a red switch does the opposite. The switches are scanners: they trigger the moment Hannah passes through them, whether she wants them to or not (see Figure 3.29).

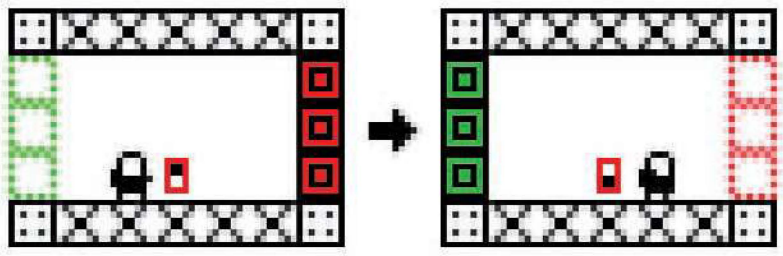

Figure 3.29 "Triggering" switches and gates are the secondary verbs.

The first function of these is to guide the player's exploration of the world: a red wall on one screen might be opened by a red switch a few screens away, requiring the player to go find it, which creates continuity between individual scenes of the game. Switches and walls can also be one-way doors: just passing in front of a switch activates it, regardless of whether it's to the player's benefit. Maybe this switch creates a wall in the hall Hannah just passed through, preventing her immediate return. You can see how switches could even be traps: since extended walls function as solid objects, imagine a midair platform built out of them. If Hannah touched a switch, the floor would open. She would have to avoid switches as she travelled across the platform (see Figure 3.30).

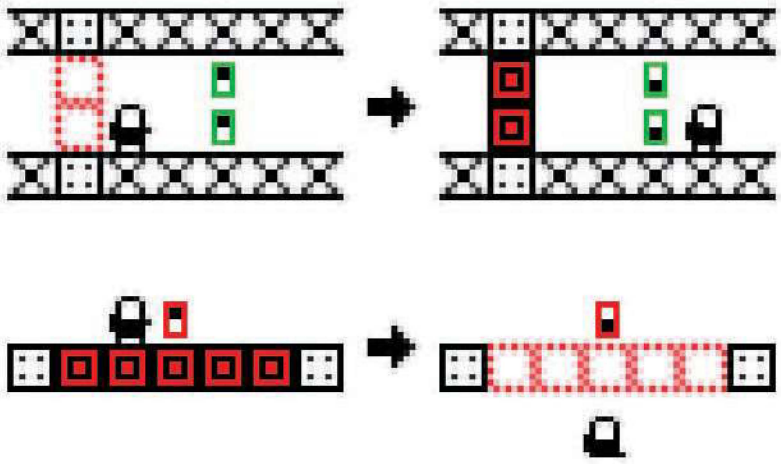

Figure 3.30 One-way door and gate platforms sometimes act as traps in *REDDER*.

REDDER's Mars is divided into distinct areas to help the player focus her exploration. Each area is distinct not only visually (that is, the blocks that make up that area look noticeably different from those of other areas) but in shape. For example, one area is an underground city, divided into two areas, with a gate separating the upper city from the lower city. Another area is a kind of puzzle box, with red and green gates separating each screen. Making progress requires manipulating switches in the right order.

One of the deepest areas is an open cavern consisting of many rooms that can be traversed with a system of rails and catwalks. Passage from room to room, once again, is governed by switches and gates. Whereas the puzzle box area is a maze, this area is straightforward: each room contains a switch opening the entrance to the next room, usually placed in a tricky-to-reach position. When Hannah touches it, she can advance to the next room in the path. There are two paths that the player can follow to navigate the cavern: a clockwise circle and

a counterclockwise circle that join in the lower middle. In the upper-middle room, shown in green in Figure 3.31, the paths divide, giving the player a choice.

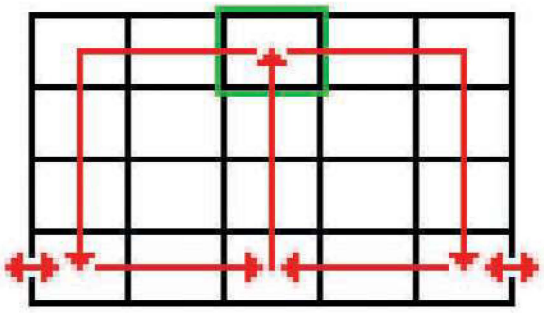

Figure 3.31 Layout of two paths (one clockwise, the other counterclockwise) connecting various rooms.

Figure 3.32 shows that upper-middle room, where the paths split. At least, this is what that room originally looked like, in an early version of the game. It's symmetrical because the scene's role is symmetrical: the player has a choice of entering the left or right path. She enters the room through the bottom. She has to climb up on one of the ledges at the left or right and then jump over the nearest electric fence-topped pillar to touch the switch above it. (The electric fences are dangerous. Hannah will have to return to the previous scene and start again if she lands on one.) The switch opens the left and right exits and closes the path through which she entered, turning it into solid floor.

So this was the room for a while. But I needed to connect this area—the cavern of catwalks and one-way doors—with a different area above it, the aforementioned underground city. In the interest of player mobility, there needed to be a way for the player to pass between these two areas in either direction. This scene, the upper-middle room, seemed the most logical place for the player to enter from above, being an intersection between this area's two paths. A connection here would afford the player the most choices.

I had to create a vertical connection between this room and the one above it. I regretted having to break the visual symmetry of the scene, which perfectly matched the symmetrical left/right choices available to the player. What if the broken symmetry was part of this room's apparent history? A hole in the upper catwalk could be the vertical connection to the above area, and the presence of the hole could signify a structural decay that destroyed the room's ostensible original symmetry.

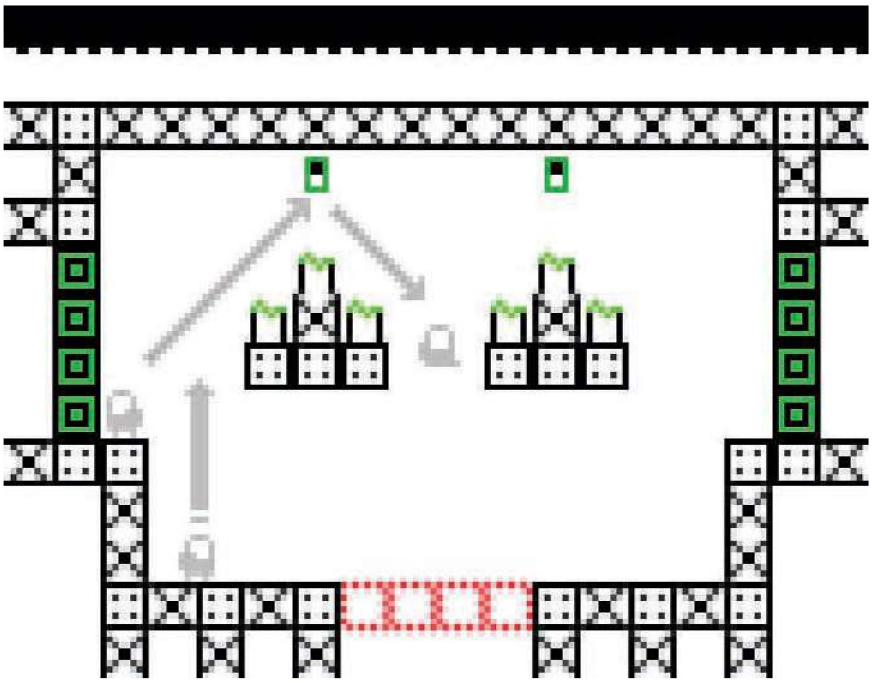

Figure 3.32 Original version of the room where the path splits to the left and right.

Figure 3.33 shows the next iteration of the scene. The piece I took out of the ceiling now rests on top of the platform below, where the electric fences can be found on its mirror platform. The implication is that part of the ceiling collapsed, changing the layout of the room and creating a path upward. This platform-on-top-of-a-platform provides enough height that Hannah can jump through the hole to the top of the above catwalk and travel along it—two rooms to the right is a passage to the city area above.

But this change introduced a new problem. The player can still hit the green switch by jumping from the left ledge over the fences. But because the right platform is so tall, if she tries to jump and hit the switch from the right, she'll hit the ceiling and descend early, landing on the electric fence (see Figure 3.34). This was a bad break in the symmetry of the room that wouldn't do. There's no visual indication that jumping from the right should be less valid or more dangerous than jumping from the left.

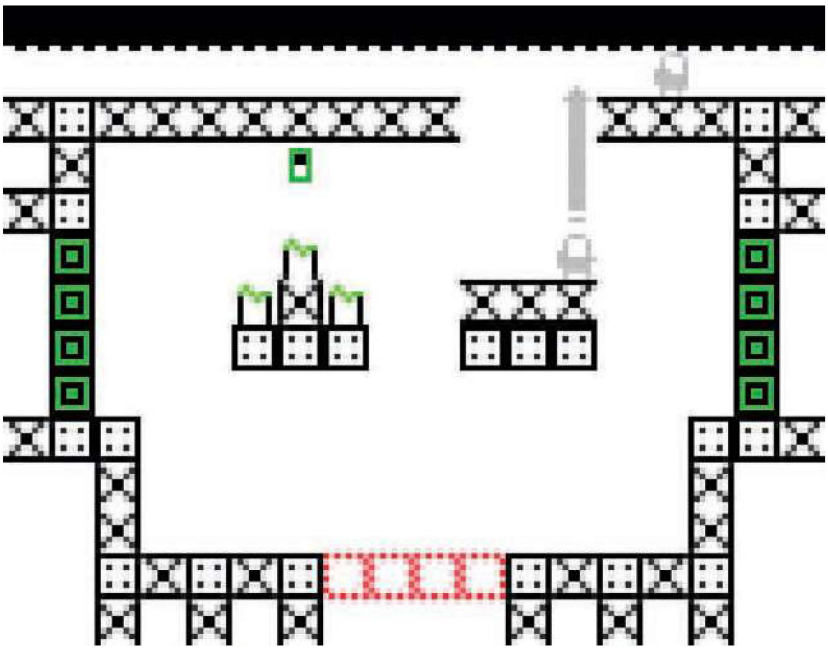

Figure 3.33 Updated scene with fallen platform opening a hole at the top.

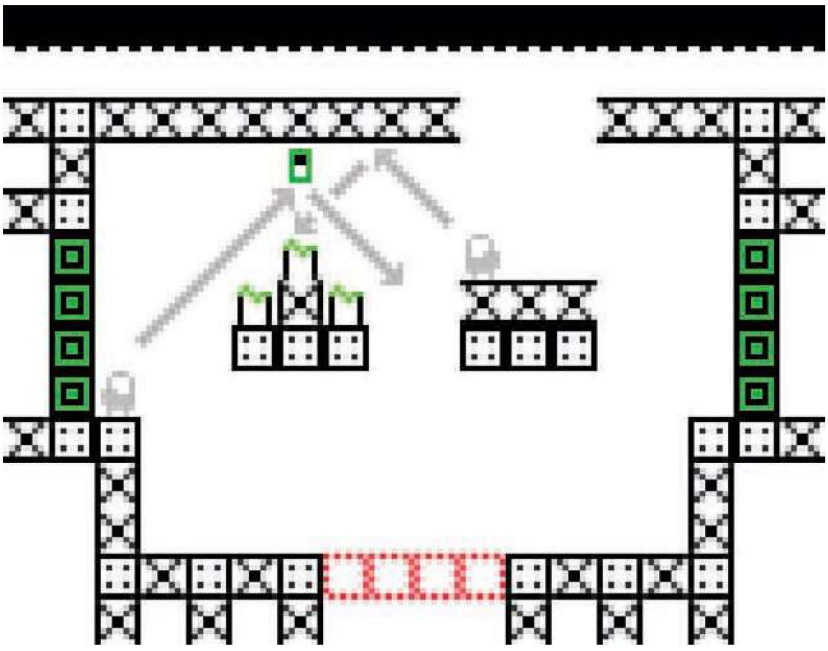

Figure 3.34 A bad break in symmetry makes jumping from the right deadly.

So how could I make the switch accessible from both the left and the right? If I got rid of the fallen catwalk—lowering the height of the right platform—Hannah could make the jump to the switch, but she couldn't reach the upper hole. Well, this is a dislodged chunk of the ceiling, right? There's no reason it needs to be flush with the platform it's lying on. Why would a piece of busted architecture fall in such a neat position?

Figure 3.35 is the finished scene. I shifted the fallen catwalk over one block so it overhangs the platform. As a nice bonus, it's visually even more asymmetrical—a sign that it's an exit from the symmetry of the whole left/right area shown in Figure 3.31. Jumping from the top of the fallen catwalk, the player can reach the hole above and the passage to the city. Jumping from the exposed piece of platform beside it, the player can hit the switch and clear the electric fence. Now the room fills every function I want from it. I created, played, identified problems, changed, played again, discovered new problems, changed again, played again, and solved the problem. That's design.

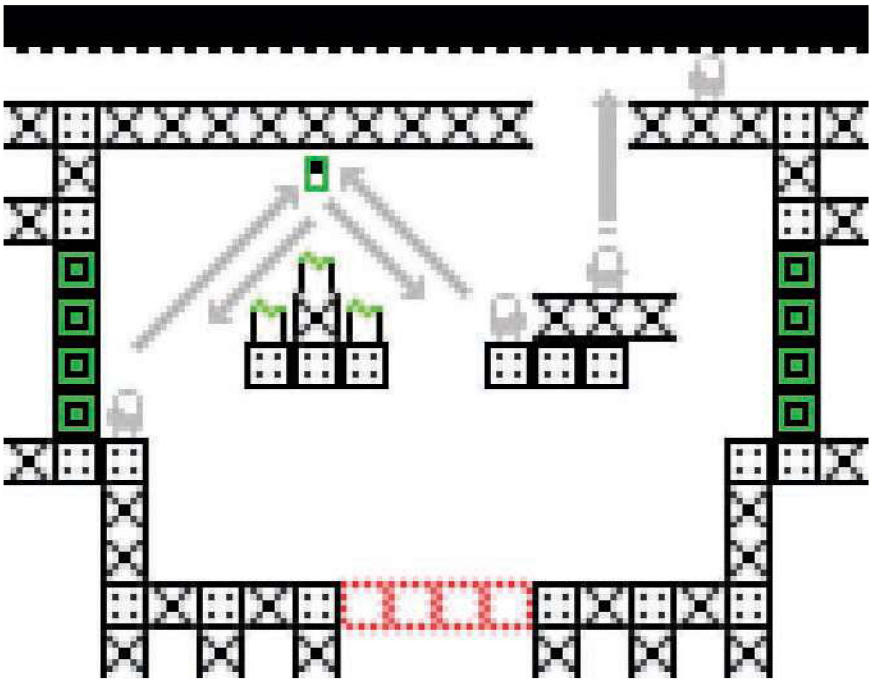

Figure 3.35 The finished scene with all problems fixed.

Review

- A scene is the most basic unit of pacing in any given game. What constitutes a scene might be different in every game.
- The responsibilities of a scene are to introduce and develop rules of the game. Objects are the building blocks of our scenes, what we use to create choices.
- It's important to introduce rules clearly so that the player understands what they mean and represent when she encounters them in successive scenes.
- We should use the rules we've established—verbs, objects, their relationships—to create choices for the player. Otherwise, we're creating choices that have no relationship to the game.
- Scenes are often made of many tiny choices, exclusive to each player's individual performance of a scene. That's why it's often useful to think and design in terms of the overall shape of a scene.
- Each scene should have a purpose that we can identify: to develop a specific rule or to present a specific idea. Design means using the rules the player already understands to communicate that idea.
- Layering—including different rules that work well together in a scene, like making the player track two different kinds of movement simultaneously—can create stronger, more effective scenes.
- Moments of inversion or climax give a scene a more dynamic shape and can draw attention to different aspects of the player's verbs.
- Chance is useful for focusing the game on the interactions between rules rather than how they're presented. Using randomized patterns for scene layouts, for example, is at its most useful when it's the interactions between many rules that are important, not the encounters that can be staged with them.
- Chance allows us to break stagnation by interfering with the player's ability to predict the game and to break symmetry by forcing competing players to play differently.
- Designing a scene often involves several drafts. Between drafts, play, identify problems, and make changes to solve those problems.

Discussion Activities

1. Choose a game—the game your group used for the previous chapter's discussion, if possible. Give yourself 10 minutes to play. If you're in a group, only one person should play at a time. When the 10 minutes are up, finish the scene the player is currently on. The group can decide what constitutes the end of the scene. Discuss that last scene.

2. Identify any rules introduced in that scene, and any ways in which objects interact in that scene. Most especially, what are the ways that appear in this scene for the first time? What is the focus of the scene? What rules does the scene introduce? What rules does it develop? What's the purpose of this scene?
3. Using graph paper, design a scene for the teleporting/laser game described in this chapter. Assume that six squares wide by eight squares tall is the space the player has to move around in: the scene you're designing will scroll onscreen into that space, from a direction and at a speed that's up to you.

   A line along any grid line is a laser fence and cannot be touched. A box that's filled in is an irradiated laser area and cannot be teleported to. A dot in any box is a delicious food capsule (see Figure 3.36).

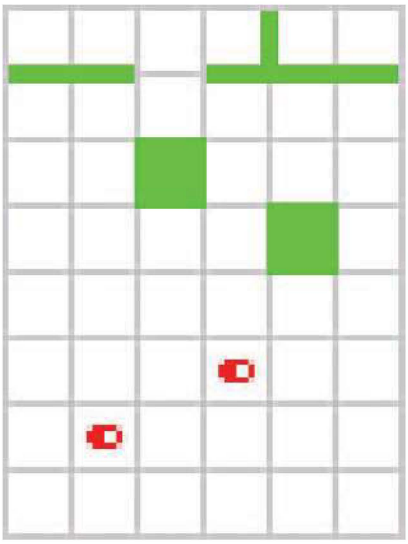

Figure 3.36 Reference for designing a scene in the teleporting/laser fence game.

4. Design a scene around a meaningful choice for the player to make. There will probably be more than one choice the player makes in your scene, but there should be at least one major choice—one the player recognizes as a choice with consequences.

   You may introduce a new object, but you must communicate the rules and ramifications of that object prior to the major choice.
5. Choose one of the following scenarios:
   - A masquerade ball celebrating the protagonist's birthday. She is now old enough to inherit the Imperial throne. At least one of the guests is an assassin and will attempt to strike before the night is through.

- Shipwrecked! The protagonist's vessel has been shattered by a terrible storm, and she must find her way to dry land. Once there, she must somehow find shelter.
- This old house cleaned up so well, no one will even realize it's haunted. Just a few more tasks before the protagonist can put it on the market—but then something goes unexpectedly awry!

Now, imagine that you're going to shape the space of possibilities in the scenario you chose—the space of choices the player can make. Following is a list of a few possible shapes. Each of these shapes changes over time. When the shape is more open, the player has a wider space in which to decide what to do. For a masquerade ball, that might be the informal part of the evening where guests are free to mingle as they choose and talk to all sorts of characters. When the shape is tighter, more closed, the possible choices are more constrained, with fewer options open—perhaps because of tension or danger. In the masquerade ball, this might involve the moment when the assassin strikes, when the player must react immediately.

Here are some possible shapes for your scenario. How might these play out? What would happen at each change in shape?

- A wide-open space in the beginning and for the first half, then narrowing to a tight space for the second half.
- A wide-open space followed by smaller and smaller ones, but ending in a large open space of possibilities.
- A narrow space from the beginning that periodically gets wider and then contracts again at the end.

Defining the rules of the game (and the larger story into which this scene fits, if you so desire) is up to you.

Group Activity

Knytt Stories, created by Nicklas Nygren in 2007, is a platform for creating playable "stories" about a protagonist, named Juni, whose primary verbs are "running," "jumping," and "climbing." You can download it freely at http://nifflas.ni2.se/?page=Knytt+Stories, though it will only run on Windows computers.

Have someone in the group play through the included "Tutorial" story and 10 minutes or so of the other included story, "The Machine." Then, working as a group or breaking into smaller groups, look at the level editor. To understand the level editor, be sure to check out the following posts in the Knytt Stories support forum:

- **Level Editor FAQ:** http://nifflas.lpchip.nl/index.php?topic=28.0
- **Level Editor Manual:** http://nifflas.lpchip.nl/index.php?action=dlattach;topic=18.0;attach=5

Figure 3.37 shows the level editor. The numbers 0–7 on the right represent different layers of the game. Layer 3 is where solid objects go: anything placed here Juni will be able to jump to and touch and climb on. Layers 0, 1, and 2 are the background: anything placed here Juni will pass right over. Click on a layer, and then click on one of the tiles on the bottom of the window. Then you can use the mouse to draw in the scene.

Figure 3.37 The *Knytt Stories* level editor; click the brightly colored numbers to change which layer of the game world you're working with.

Layers 4–7 are for "objects," which in *Knytt Stories* refers to any object more complex than a wall or floor tile. At the far right of the screen, you can select from the many objects in *Knytt Stories*: click on "bank" to cycle through categories of objects and "obj" to cycle through specific objects. Right-clicking cycles in the opposite direction. (You can also cycle tilesets and backgrounds by left- and right-clicking on the appropriate parts of the editor.)

Knytt Stories has a variety of objects. Each group should confer, experiment, and pick three of these. These objects could be a block that's only visible when the player is close, a spiky creature that drives back and forth along the ground, and a button that needs to be held for a while to open a gate.

(Also, don't forget about save points—bank 0, object 1! Juni starts over at these if she touches something dangerous like a spiky creature. Also, be sure to turn on the abilities that you want the player to start with. You can choose these after clicking "set start pos," which allows you to set the story's starting position. I recommend starting the player with the ability to run and climb, at least.)

Create a story about the three objects you've chosen. Start by introducing the first one. Develop it a little in another scene. Then introduce the second object by itself, and create a scene that combines the first and second objects. Introduce the third object by itself. Create a scene that combines the first and third objects, and then one that combines the second and third. Finally, combine all three.

Make sure that your combinations don't merely contain the different objects, but actually interact with each other. Find as many uses for each object—as much utility—as possible.