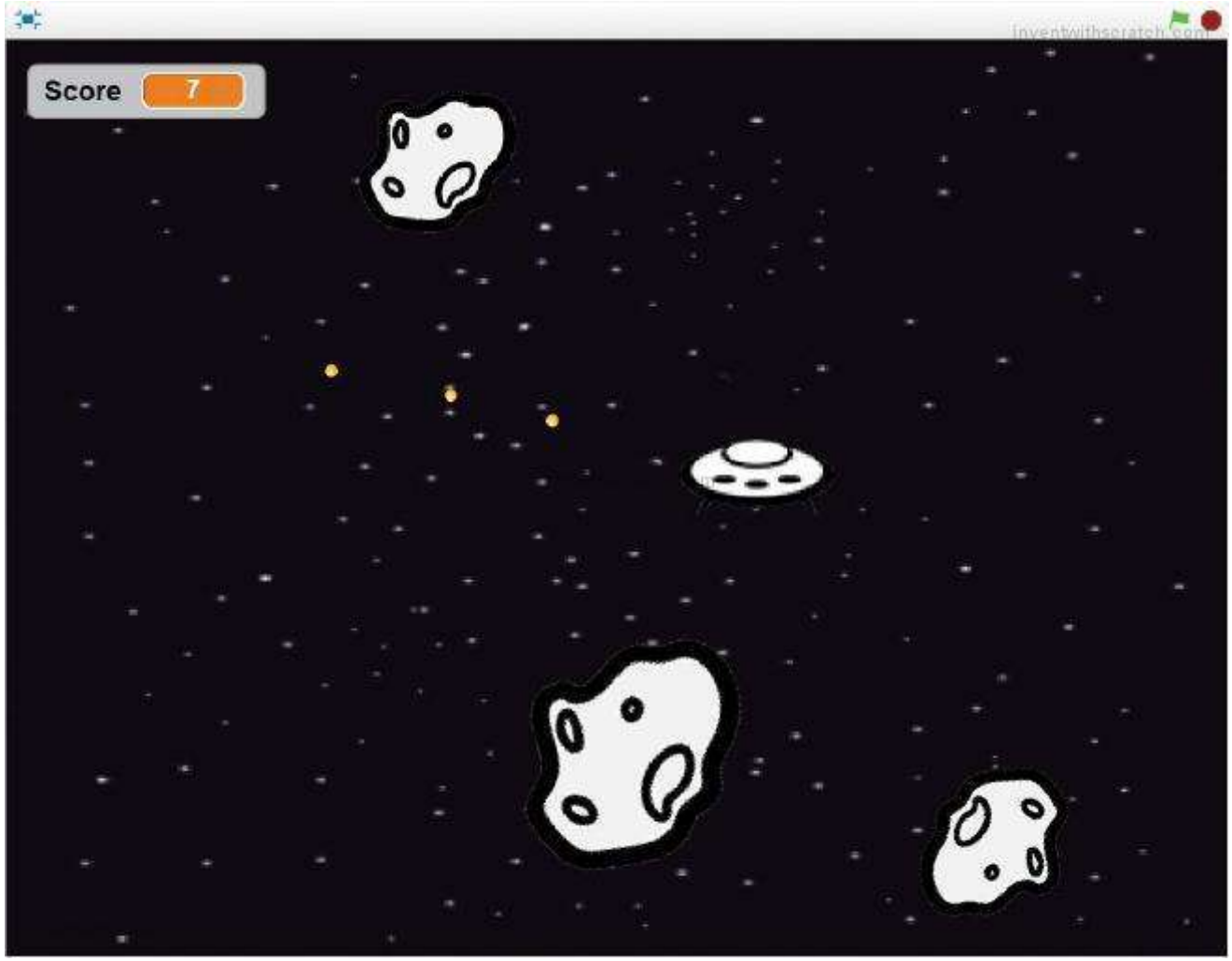
8
ASTEROID BREAKER . . . IN SPACE!

Asteroids is a classic game developed in 1979 by Atari. Since then, many programmers have remade the game, and it's a great programming project to make in Scratch, too. The player pilots a spaceship that must destroy space asteroids while avoiding the space pieces that break off . . . in space! (It's a well-known fact that adding ". . . in space!" makes everything more exciting.)

Instead of directly controlling where the spaceship moves, the player *pushes* the spaceship like a hockey puck on ice; because the player's ship has inertia, it slides around the Stage. To slow down the spaceship, players must push it in the opposite direction. It takes skill to move the spaceship without losing control, but that's half the fun of the game. The other half of the fun is blowing up asteroids.

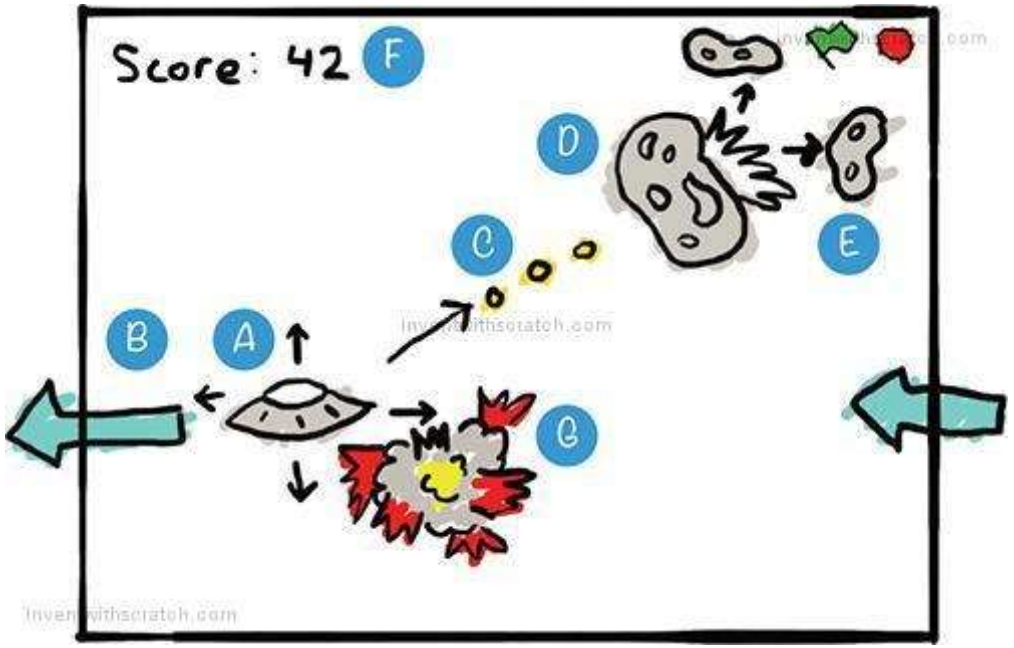
Before you start coding, look at the final *Asteroid Breaker* program at *https://www.nostarch.com/scratchplayground/*.



SKETCH OUT THE DESIGN

Let’s draw on paper what the game should look like. In our version of the game, the player controls their spaceship with the WASD keys and aims at incoming asteroids with the mouse.

Here is what my sketch looks like:



And here's what we'll be doing in each part:

A. Make a spaceship that is pushed around

B. Make the spaceship wrap around the edges

C. Aim with the mouse and fire with the spacebar

D. Make asteroids that float around

E. Make asteroids split in two when hit

F. Keep score and make a timer

G. Make the spaceship explode if it is hit

If you want to save time, you can start from the skeleton project file, named *asteroidbreaker-skeleton.sb2*, in the resources ZIP file. Go to *https://www.nostarch.com/scratchplayground/* and download the ZIP file to your computer by right-clicking the link and selecting **Save link as** or **Save target as**. Extract all the files from the ZIP file. The skeleton project file has all the sprites already loaded, so you'll only need to drag the code blocks into each sprite.

A MAKE A SPACESHIP THAT IS PUSHED AROUND

Before we code the exciting parts of the game, we need to set up the backdrop and sprite. We'll make this game spacey by adding the stars backdrop and the spaceship sprite. Click the **Choose backdrop from library** button under New backdrop, select **stars**, and click **OK**.

We won't use the cat sprite that Scratch starts with, so right-click that sprite and select **delete** before continuing. Start a new project in the Scratch editor, and enter *Asteroid Breaker* as the project name.

1. Create the Spaceship Sprite

We'll use a flying saucer image for the spaceship, which you'll find in the resources ZIP file.

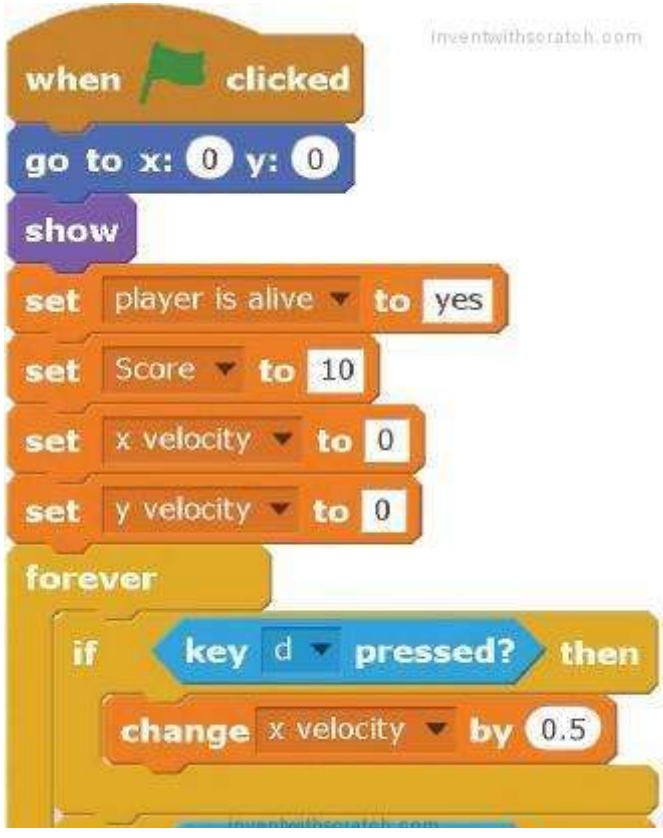
Click the **Upload sprite from file** button next to New sprite. Then select the *Spaceship.png* image file from the resources ZIP file.

In the orange *Data* category, click **Make a Variable** and create a variable named x velocity. Make x velocity a For this sprite only variable. Repeat the preceding steps to create a variable named y velocity.

> **NOTE**
>
> *If For this sprite only does not appear, the Stage is selected instead of the Spaceship sprite. Close the New Variable window, select the Spaceship sprite, and click **Make a Variable** again.*

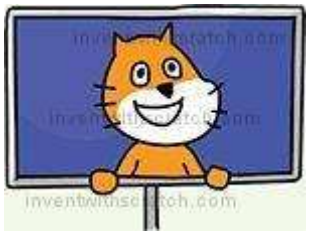
You'll also need to create two variables named Score and player is alive, but make these variables For all sprites.

Next, add the following code to the Spaceship sprite. The code defines the ship's starting position and the initial values of variables; it also contains the logic that defines the user's controls.



You might be wondering why we didn't reuse code from previous games in which we used the **change x by** or **change y by** block. In this program, holding down one of the WASD keys adds to or subtracts from the x velocity and y velocity variables. Then the code at the bottom of the script changes the x and y position of the Spaceship sprite by using the values in these variables. Even after the player lets go of the key, the variables still reflect the updated position, so the spaceship continues to move.

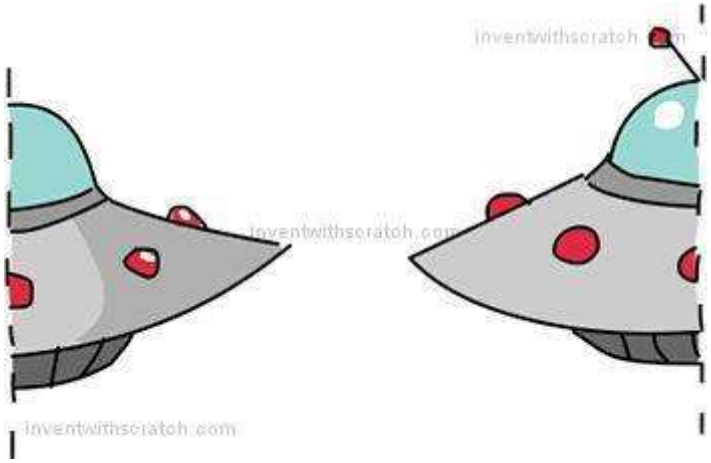
SAVE POINT



Click the green flag to test the code so far. Press the WASD keys to see how the spaceship gets pushed. Make sure all four WASD keys push the spaceship in the correct directions. Then click the red stop sign and save your program.

B MAKE THE SPACESHIP WRAP AROUND THE EDGES

When you tested the code, did you notice that the Spaceship sprite stops immediately when it runs into the edge of the Stage? The reason is that Scratch prevents sprites from moving off the Stage, which is helpful in most Scratch programs. But in *Asteroid Breaker*, we want sprites to go off the side of the Stage and *wrap around* to the other side.



2. Add the Wrap-Around Code to the Spaceship Sprite

The following code will make the spaceship travel to the other side of the Stage whenever it reaches an edge. Add this code now.

The left and right edges of the Stage are at x-coordinates –240 and 240, respectively. The bottom and top edges of the Stage are at the y-coordinates –180 and 180. We use these boundaries to write code that changes the position of the Spaceship sprite when it goes past these four coordinates. Whenever the **x** or **y position** of the Spaceship sprite gets within 5 steps of these edges, this new code will move the Spaceship sprite to the other side of the Stage. Because the x velocity and y velocity variables will still be moving the Spaceship sprite at the same speed and in the same direction, the Spaceship sprite will look like it's moving continuously around the Stage.

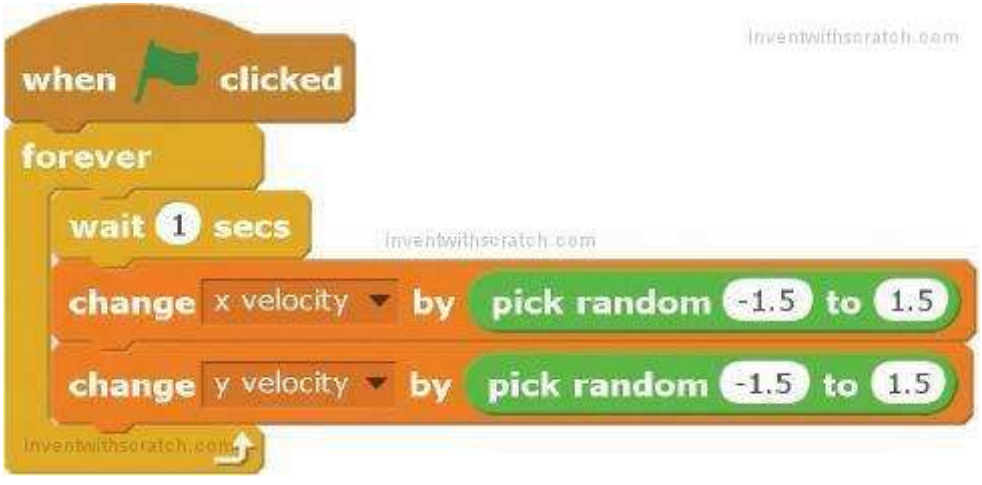
SAVE POINT

Click the green flag to test the code so far. Make sure that all four edges send the Spaceship sprite to the other side. Then click the red stop sign and save your program.

3. Add the Random-Push Code to the Spaceship Sprite

The controls for this game offer a challenge, but let's make playing the game even more difficult. We'll add random little pushes to the Spaceship sprite so that the player can't just stay in the center without moving at all.

Add the following code to the Spaceship sprite to make random pushes happen every second:



Inside the **forever** loop, the x velocity and y velocity variables are changed by a small, random amount after a 1 second pause. This means that every second the spaceship's movement receives a random push.

SAVE POINT

Click the green flag to test the code so far. Don't press any of the WASD keys. Wait to see if the spaceship starts moving a little on its own. Then click the red stop sign and save your program.

AIM WITH THE MOUSE AND FIRE WITH THE SPACEBAR

The code that controls the Spaceship sprite is complete, so let's add energy blasts. These blasts will bust up those dangerous space asteroids! In space!

4. Create the Energy Blast Sprite

Scratch's Sprite Library has a sprite we can use for the energy blasts. Click the **Choose sprite from library** button next to New sprite. Select the Ball sprite and click **OK**. Open the sprite's Info Area by clicking the i button, and rename it Energy Blast.

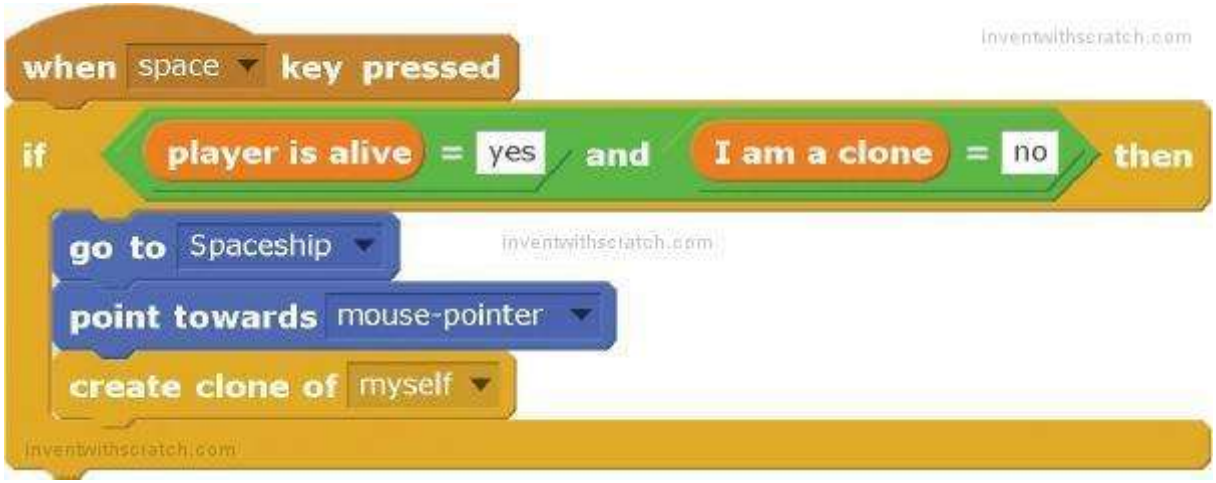
We want the Energy Blast sprite to make a laser sound when the Spaceship sprite fires it. Click the **Sounds** tab above the Blocks Area. Then click the **Choose sound from library** button under New sound. Select the laser1 sound and click **OK**.

We'll make clones of the Energy Blast sprite, but the clones and original sprite will run different code. There is only one Energy Blast sprite, but the player should be able to fire many energy blasts at once. We'll create clones of the original Energy Blast sprite so that more than one energy blast can be on the Stage. The original sprite will remain hidden; all the Energy Blast sprites that appear on the Stage will be clones.

We'll use a variable named I am a clone to keep track of which is the original sprite and which are clones. Click the **Scripts** tab to go back to the Scripts Area. In the orange *Data* category, click the **Make a Variable** button. Create a For this sprite only variable named I am a clone. The original Energy Blast sprite will set this variable to no, and the clones will set it to yes. Add the following code to the Energy Blast sprite:

The original sprite hides itself at the start of the game and remains hidden. The clones it makes will appear on the Stage. Also, the sprite we're using is too big for the game, so set its size to 10 percent to make it smaller.

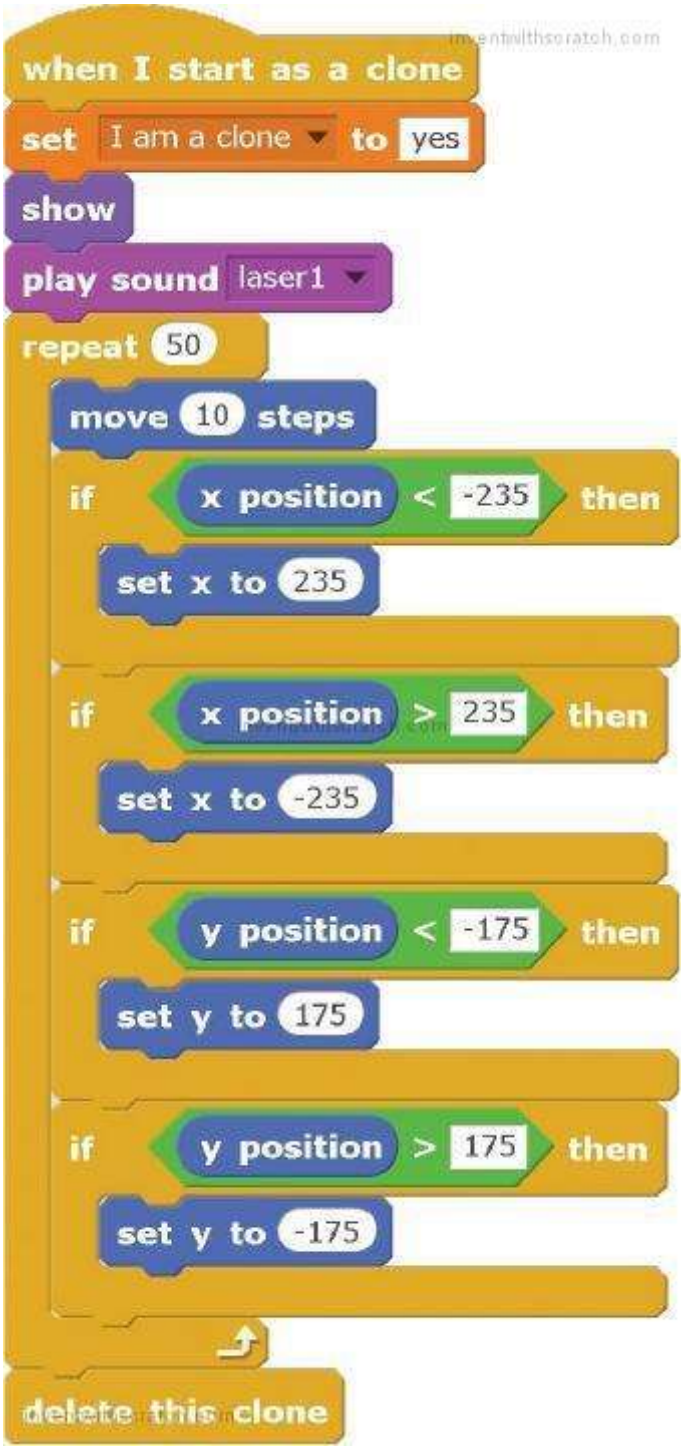
Now add the following script to the Energy Blast sprite. The player will fire an energy blast by pressing the spacebar. The original Energy Blast sprite will create clones that show themselves and move toward the mouse. Since the Energy Blast clones move toward the mouse, the player can move the mouse to aim the energy blasts.



The code under the **when space key pressed** block will run for the original sprite *and* the clones if we don't give instructions that this code should only run for the original sprite. But we don't want the existing clones to create new clones. The **if then** block checks that the I am a clone variable is set to no so that only the original Energy Blast sprite runs this code and creates clones.

Obviously, the Spaceship sprite can fire Energy Blast clones only if the player is alive, so the code also checks that the player is alive variable is set to yes.

Next, add the following code to the Energy Blast sprite so that the clones move toward the mouse after they've been created.

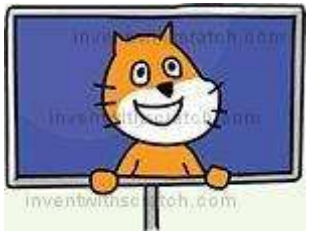


The clone starts by showing itself and moving forward. The clones should also wrap around the edges of the Stage just like the Spaceship sprite, so we use similar code to do that here.

Notice that the clones will set their own I am a clone variable to yes. This is so that the clones do not run the code in the **when space key pressed** script. The **if then** block in that script runs the code only if **I am a clone** is set to no.

The clones move forward 10 steps at a time 50 times. That means the Energy Blast clones have a limited range and won't keep moving forever. After the loop finishes repeating 50 times, the clone deletes itself and disappears from the Stage.

SAVE POINT



Click the green flag to test the code so far. Aim with the mouse and press the spacebar to fire. Make sure the Energy Blast clones start at the Spaceship sprite and move toward the mouse. The clones should wrap around the edges of the Stage and eventually disappear. Then click the red stop sign and save your program.

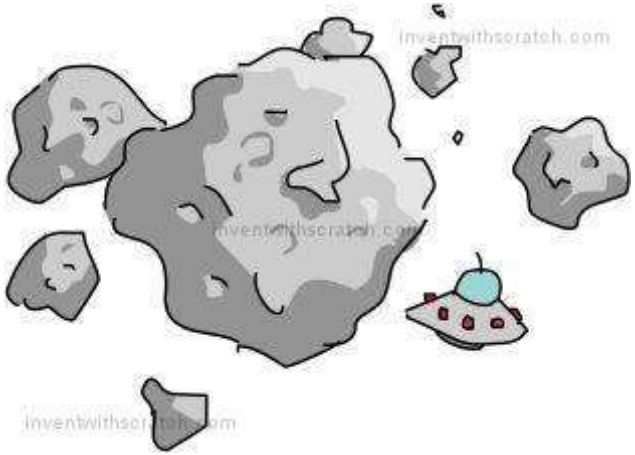
MAKE ASTEROIDS THAT FLOAT AROUND

Now we need targets for the player to hit. The asteroids in this game float around until an Energy Blast clone hits them. They'll repeatedly break apart into two smaller asteroids until they're small enough to be vaporized.

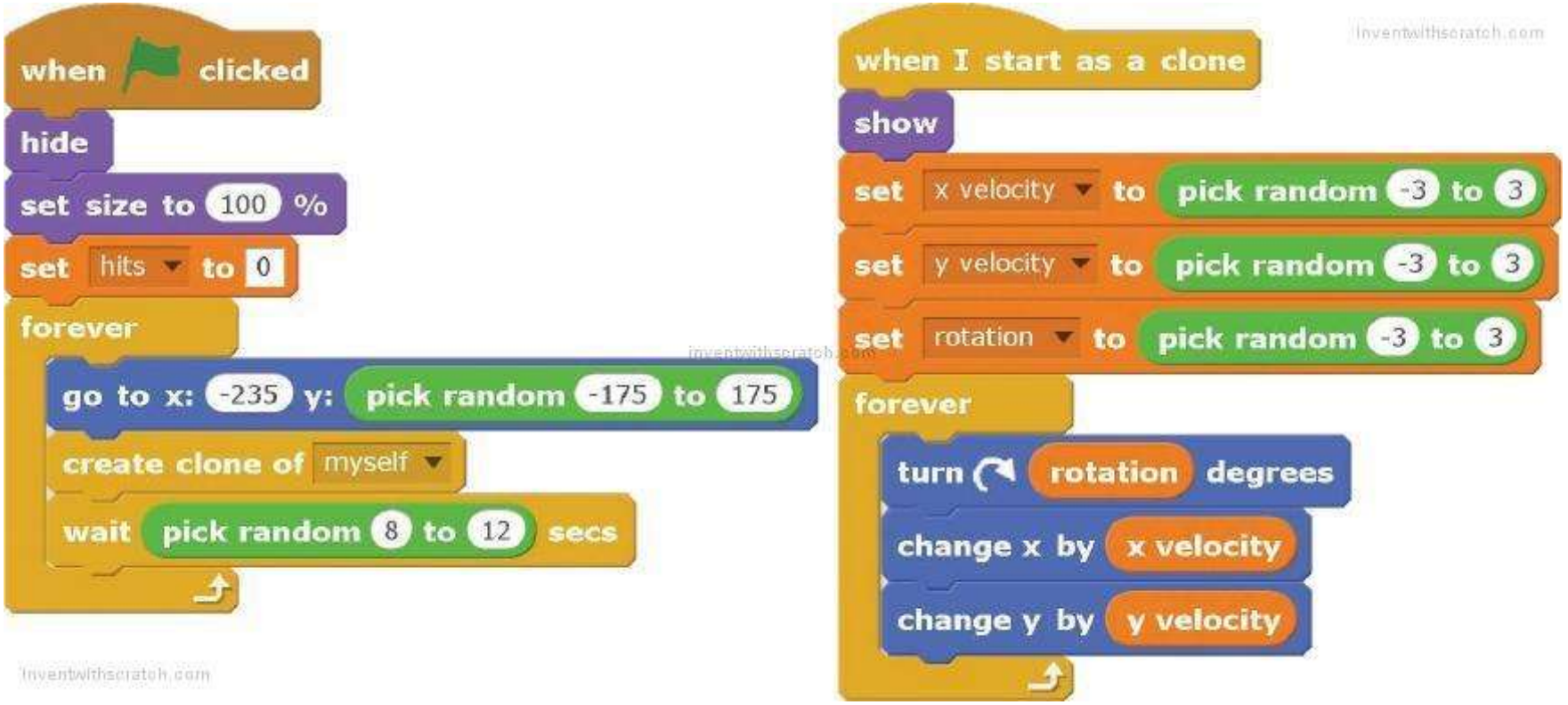
5. Create the Asteroid Sprite

Add the new asteroid sprite by clicking the **Upload sprite from file** button and selecting the *asteroid.png* file. This file is in the resources ZIP file. Click the orange *Data* category, and then click the **Make a Variable** button. Create a For this sprite only variable named hits. Repeat these steps to make variables named x velocity, y velocity, and rotation. All are For this sprite only variables. In Step 6, we'll use the hits variable to keep track of how many times the asteroid has been hit and the size of the asteroid.

Let's write some code that makes the Asteroid sprite create new clones that appear on the Stage; each clone will have a random velocity and rotation. The result will be an unpredictable asteroid swarm . . . in space!



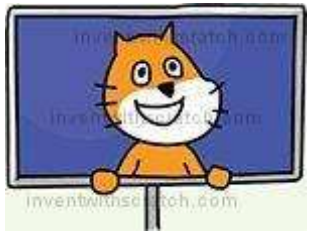
Add the following scripts to the Asteroid sprite.



Like the Energy Blast sprite, the original Asteroid sprite will hide itself and generate clones. New clones are created every 8 to 12 seconds. When the clone is created, it shows itself, is assigned random velocities and rotations, and begins moving.

Also, like the Spaceship and Energy Blast sprites, the Asteroid sprites will wrap around the edges of the Stage. Add the following script to the Asteroid sprite.

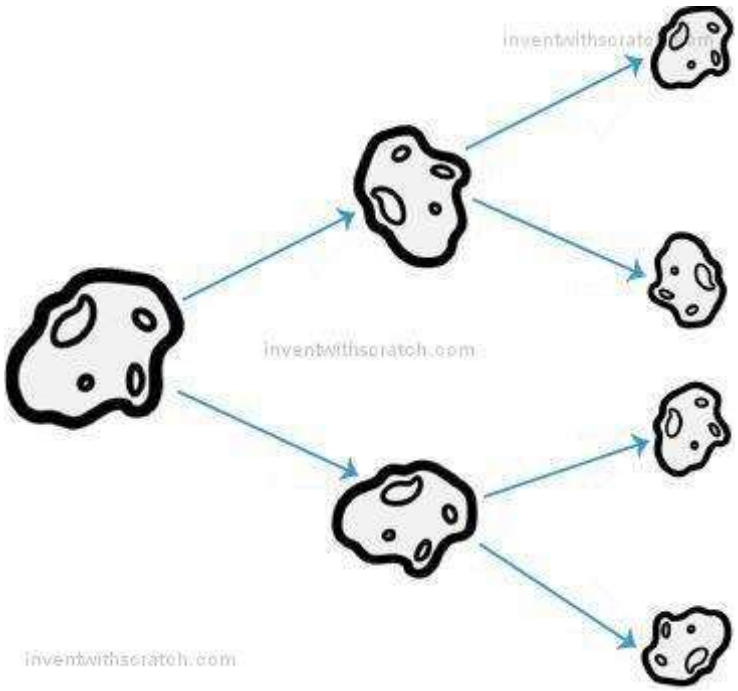
SAVE POINT



Click the green flag to test the code so far. Make sure the Asteroid sprite slowly moves and rotates and that it wraps around the edges of the Stage. Also, make sure that new clones appear every 8 to 12 seconds. Then click the red stop sign and save your program.

MAKE ASTEROIDS SPLIT IN TWO WHEN HIT

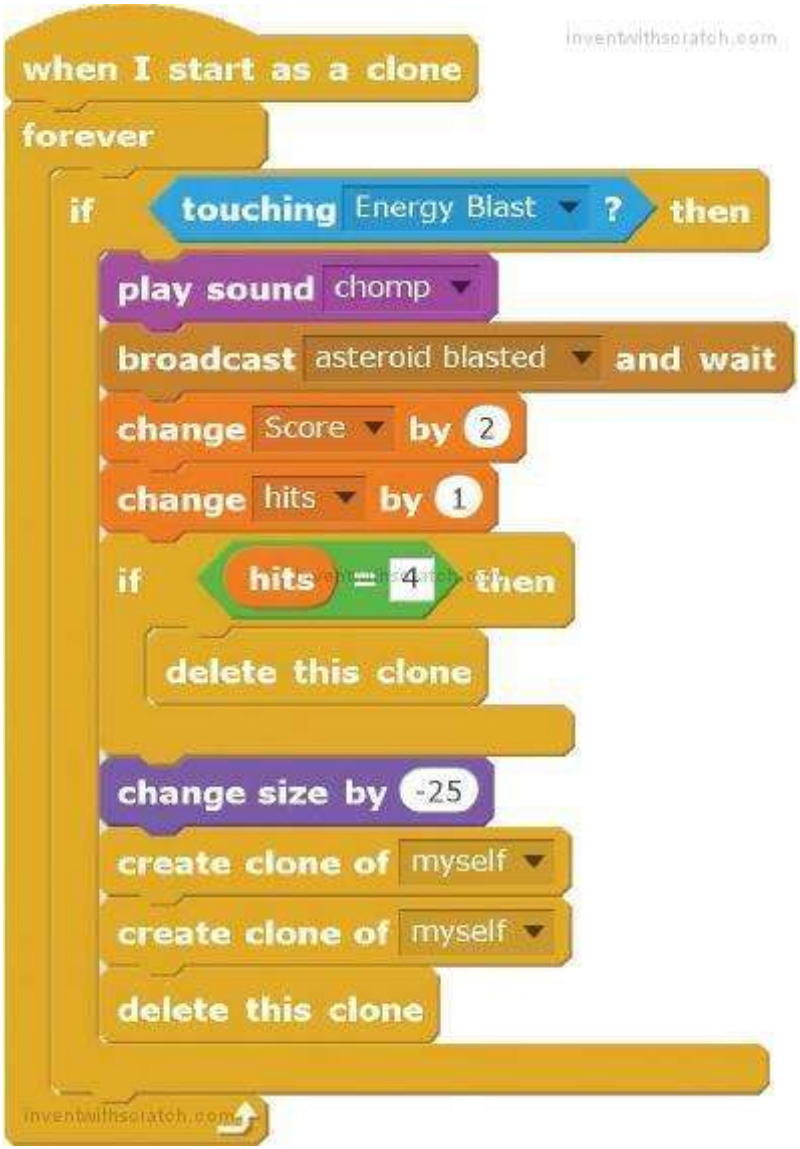
When an Energy Blast clone hits an Asteroid, the Asteroid will create two new smaller clones of itself, making it look like the Asteroid on the Stage split in two.



6. Add the Asteroid's Splitting Code

From the Sounds tab, click the **Choose sound from library** button. Then select chomp and click **OK**. The chomp sound will play whenever a clone hits the asteroid.

Add the following script to the Asteroid sprite. You'll need to create a new broadcast message named asteroid blasted.



When an Energy Blast clone hits an Asteroid clone, the Asteroid clone will play the chomp sound effect and broadcast the asteroid blasted message.

Then the Asteroid adds 2 points to the Score variable and increases the hits variable by 1. The hit Asteroid sprite will also shrink a little, changing its size by -25, so that when it (the “parent”) clones itself twice, its “child” clones will be the smaller size. Finally, the Asteroid clone deletes itself.

The two smaller child clones will have hits variables that are one more than what the parent started with. The fourth time an Asteroid clone is hit, the hits variable is 4, and the code deletes the clone instead of creating two new clones. (The code after the **delete this clone** block doesn't run, because the clone no longer exists.) This prevents 1 Asteroid sprite from becoming 2, then 4, then 8, then 16, then 32, and exponentially more Asteroid sprites forever.

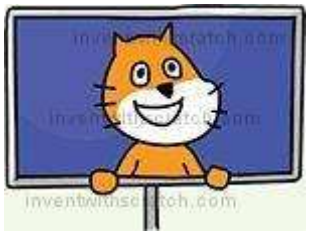
However, if you *do* want exponentially more Asteroid sprites, increase the number in the **if hits = 4 then** blocks.

7. Add the Asteroid Blasted Message Code to the Energy Blast Sprite

Select the Energy Blast sprite in the Sprite List, and add this script to it:

All the Energy Blast clones will receive the asteroid blasted message, but only those currently touching an Asteroid sprite will be deleted. (One will be touching an Asteroid, because the message is only broadcast by the Asteroid sprite when it is touching an Energy Blast sprite.) This is how an Energy Blast sprite disappears after hitting an Asteroid.

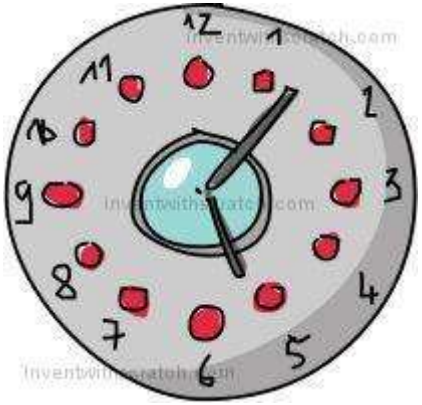
SAVE POINT



Click the green flag to test the code so far. Try blasting some of the asteroids. Make sure the energy blast disappears and the Asteroid sprite is replaced by two smaller clones. The fourth time an asteroid is hit, it should just disappear. Then click the red stop sign and save your program.

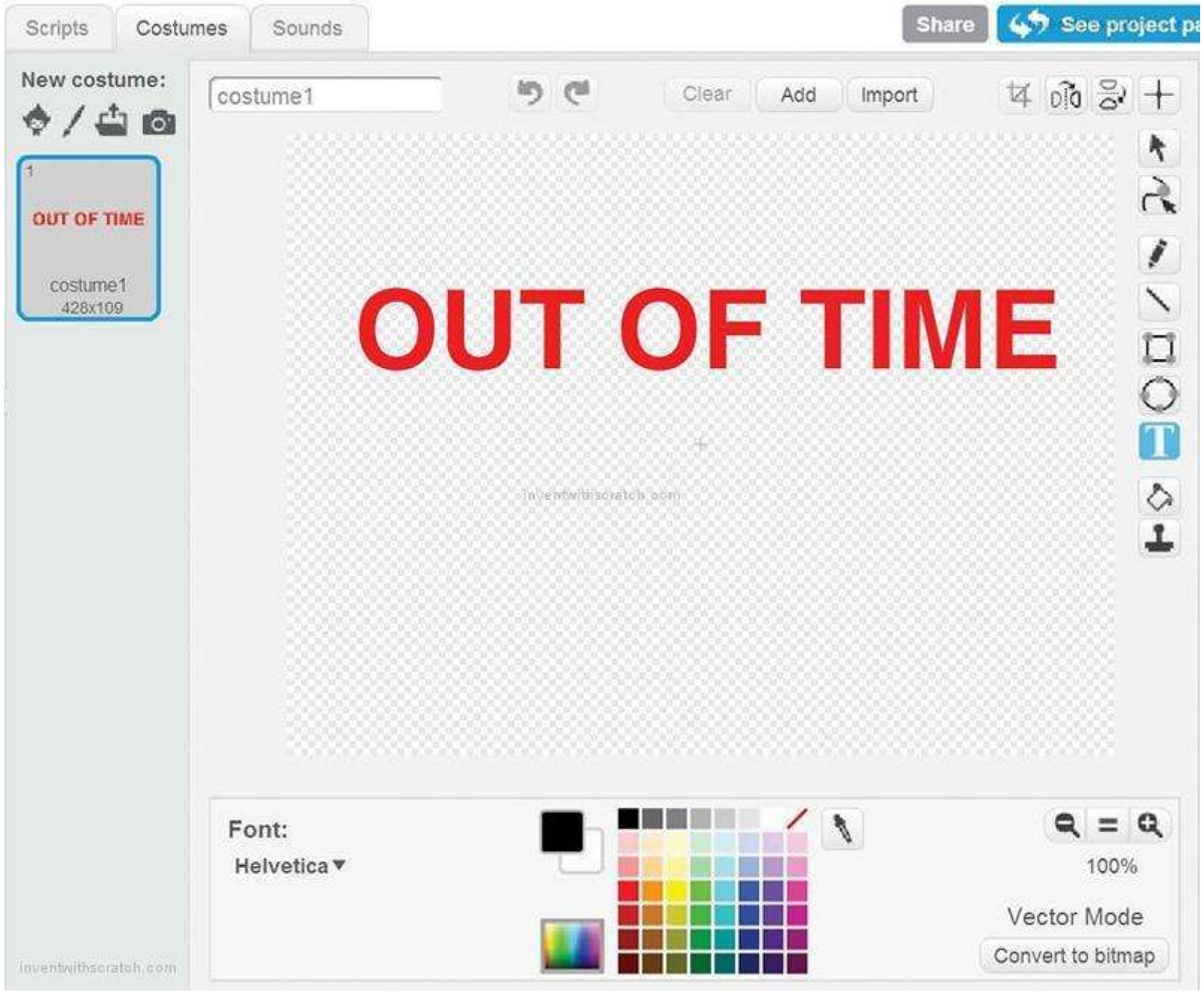
KEEP SCORE AND MAKE A TIMER

The *Asteroid Breaker* game quickly becomes challenging when many tiny asteroids are flying around the Stage. A good game strategy is to be slow and careful, finishing off small Asteroid sprites before firing at larger ones. But we also want to put some pressure on the player, so let's make the Score variable start dropping by 1 point every second and have the game end when the Score is 0. This way, the player will have to keep up a quicker pace when blasting Asteroid sprites.

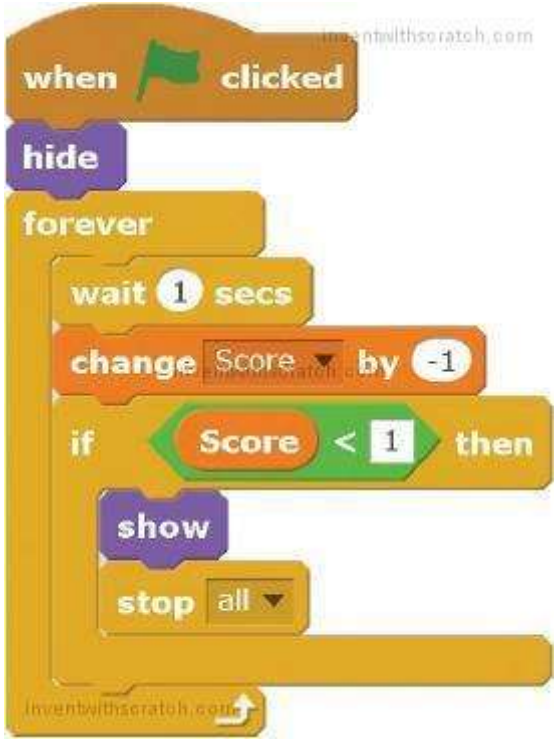


8. Create the Out of Time Sprite

Click the **Paint new sprite** button next to New sprite. In the Paint Editor, use the Text tool to write *OUT OF TIME* in red capital letters.

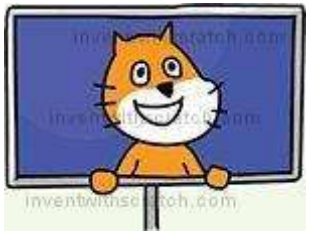


Click this new sprite's i button in the Sprite List to open its Info Area. Rename the sprite Out of Time.
Add the following script to the Out of Time sprite.



This code hides the Out of Time sprite at the start of the game and decreases the Score variable by 1 after a 1 second pause. When the Score variable reaches 0, the code shows the Out of Time sprite.

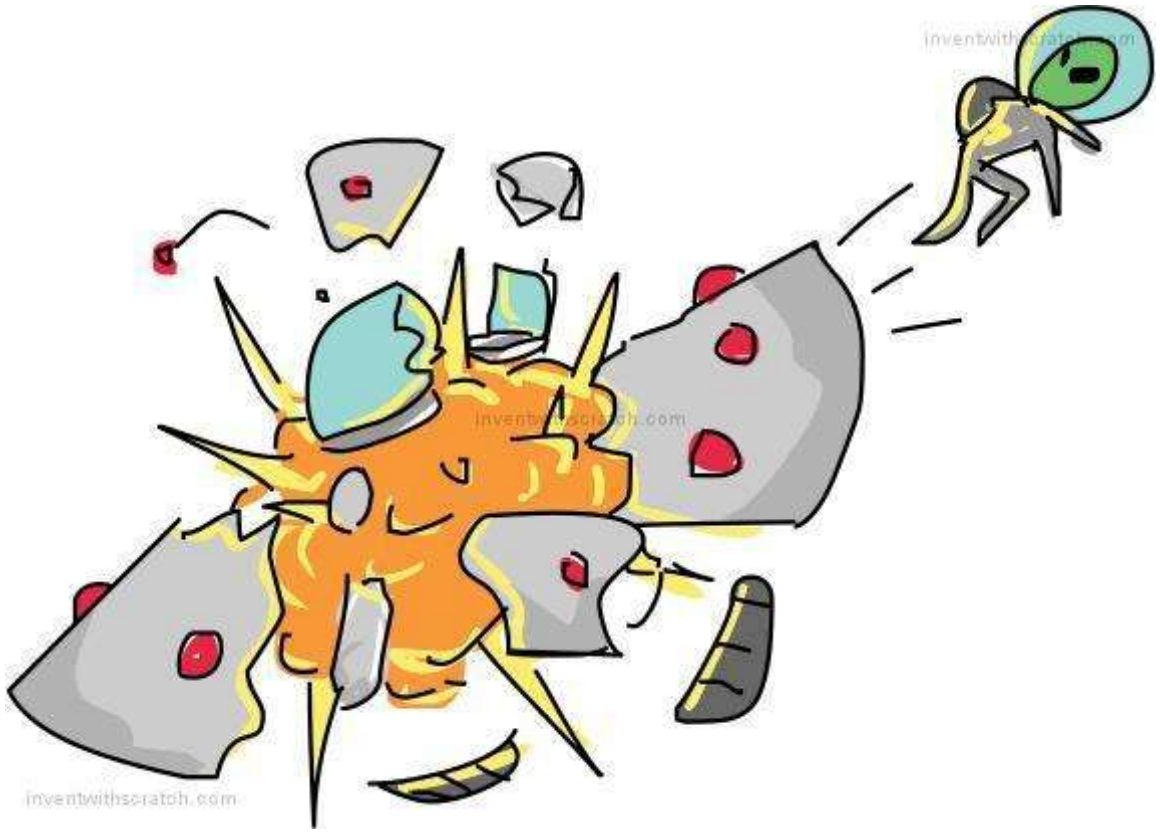
SAVE POINT



Click the green flag to test the code so far. Make sure the Score variable decreases by 1 point each second. When the Score variable is 0, the Out of Time sprite should display. Then click the red stop sign and save your program.

MAKE THE SPACESHIP EXPLODE IF IT IS HIT

A player can lose the game if they don’t blast asteroids fast enough to prevent their Score from reaching 0. But they can also lose if an asteroid hits the spaceship. Let’s add code to detect when this happens and display a cool explosion animation.



9. Upload the Explosion Sprite

Eight images are available for the frames of the explosion animation. These costumes are in the *Explosion.sprite2* file in the resources ZIP file.

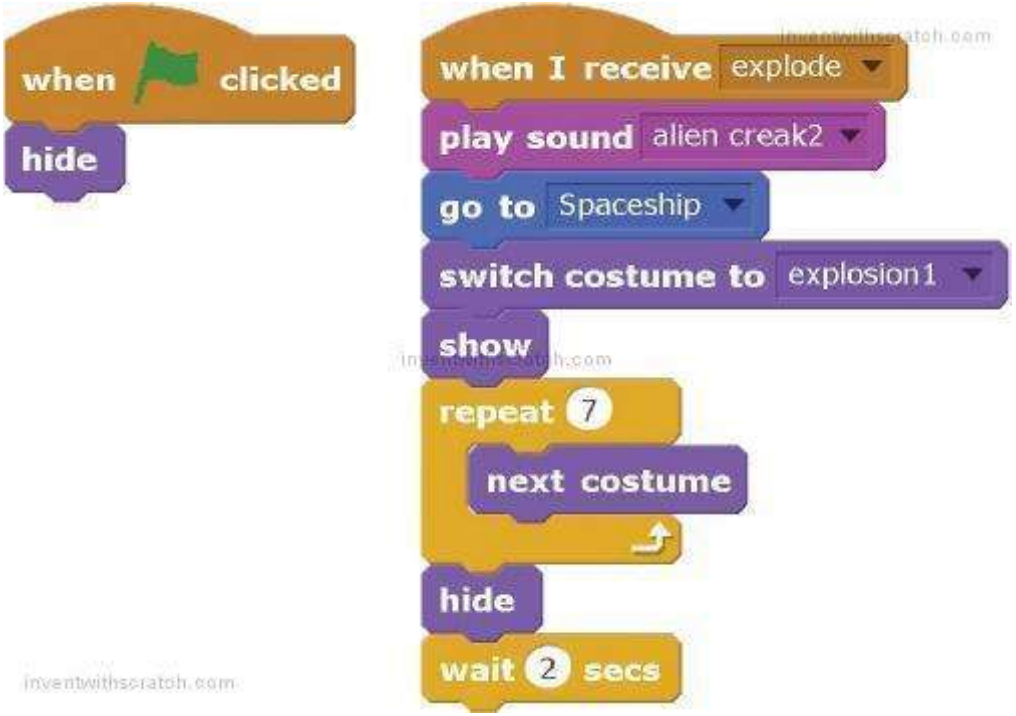
In the Scratch editor, click the **Upload sprite from file** button next to New sprite. Select *Explosion.sprite2* and click **OK**. The eight costumes for the explosion animation appear on the sprite’s Costumes tab.

10. Add the Code for the Explosion Sprite

For the Explosion sprite, you’ll create a new broadcast message named explode. When the Explosion sprite receives this message, it will appear and switch through its costumes to display the explosion animation.

The Explosion sprite will also play a sound effect when the explosion happens. Load the sound by clicking the **Sounds** tab above the Blocks Area. Then click the **Choose sound from library** button under New sound. Select the alien creak2 sound and click **OK**.

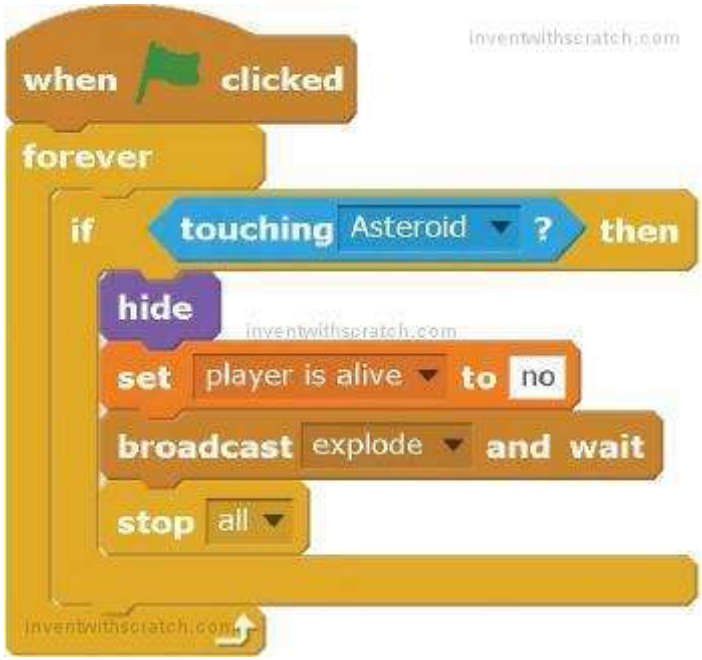
Add the following code to the Explosion sprite:



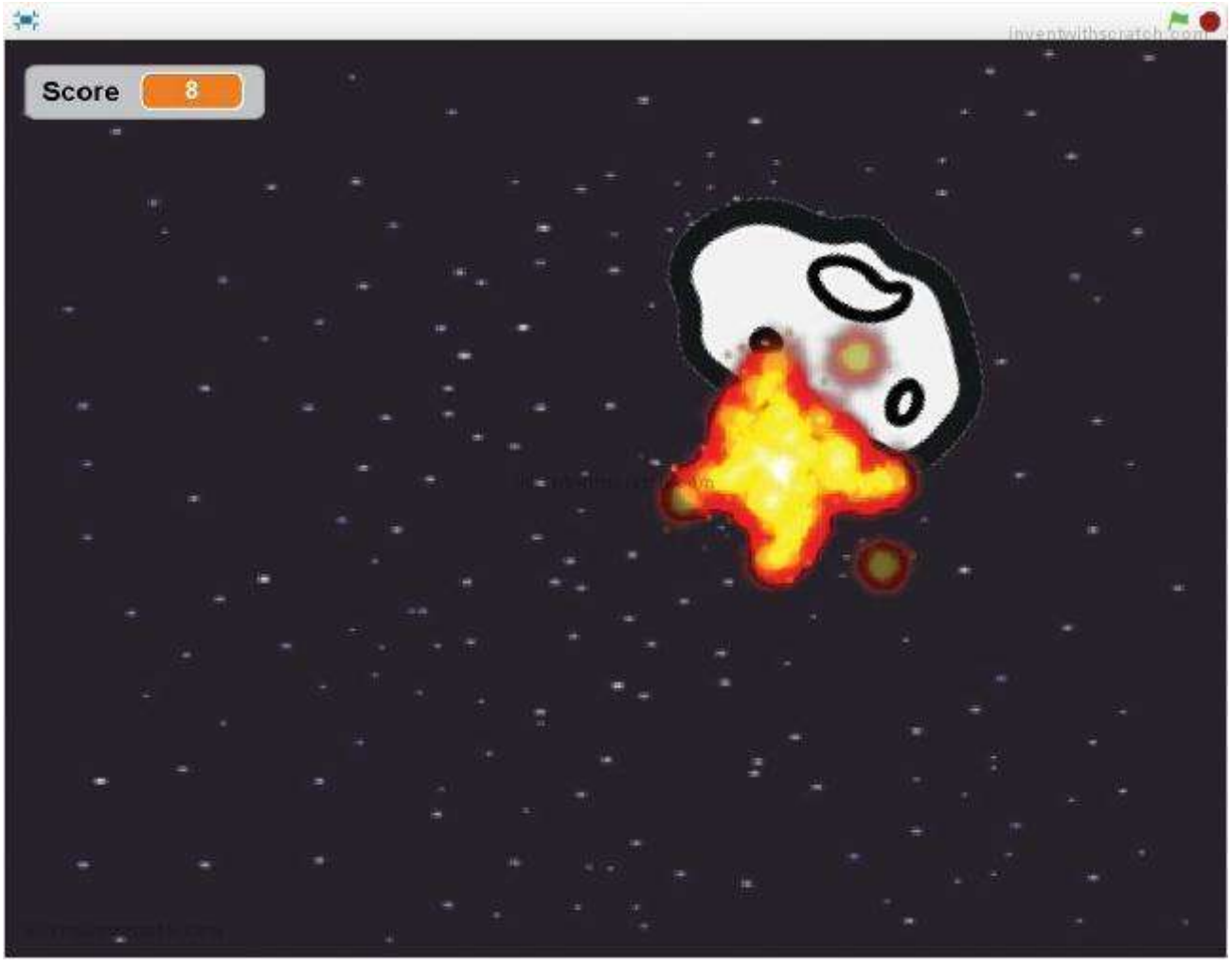
The Explosion sprite hides until it receives the explode broadcast. Then it plays a sound, goes to the position where the spaceship is, and switches its costumes seven times to create an awesome explosion animation!

11. Add the Explode Code to the Spaceship Sprite

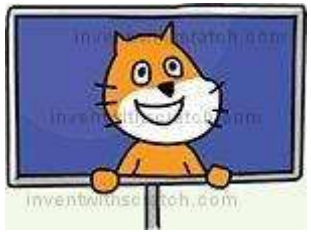
The Spaceship sprite will broadcast the explode message when it touches one of the Asteroid clones. Add the following script to the Spaceship sprite:



The explosion animation works by briefly showing one of the costumes before moving to the next costume. This is similar to the frame-by-frame animation that cartoons and flipbooks use. Each costume is a frame, and the code quickly changes costumes to make the explosion look real.



SAVE POINT

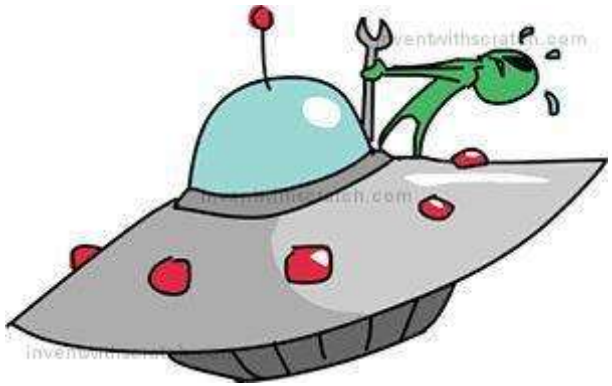


Click the green flag to test the code so far. Make sure the Explosion sprite is not visible when the game starts. Fly into an asteroid, and make sure the Explosion sprite appears where the Spaceship sprite is. Then click the red stop sign and save your program.

The code for this program is too large to show here, but you can view the completed code in the resources ZIP file—the filename is *asteroidbreaker.sb2*.

VERSION 2.0: LIMITED AMMO

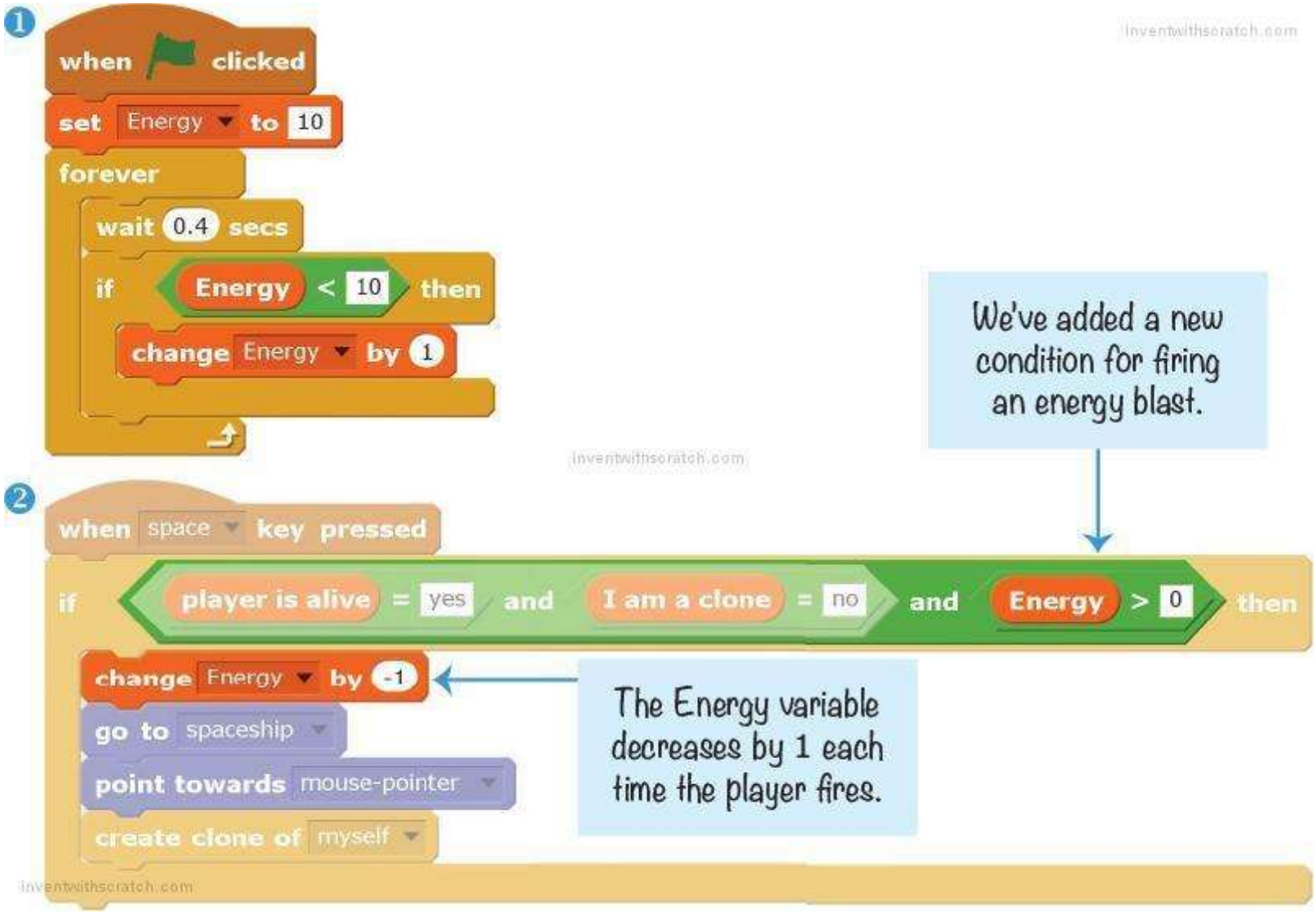
Once you get the hang of the game, *Asteroid Breaker* can become too easy. The one feature that makes it easy is that you can fire as fast as you can press the spacebar. This action lets the player fire indiscriminately instead of carefully aiming at the asteroids. But we can change that behavior by adding a new Energy variable. Firing an energy blast will reduce this variable by 1 each time. If the Energy variable is 0, the spaceship can't fire. The Energy variable will increase slowly over time, but it forces the player to carefully aim their shots and not waste them.



You'll need a variable to keep track of the spaceship's energy level. Select the orange *Data* category, and click the **Make a Variable** button. Make a variable named Energy, and select **For all sprites**. In the Blocks Area, make sure the checkbox next to Energy is checked (just like the checkbox for Score is checked) so that it will appear on the Stage.

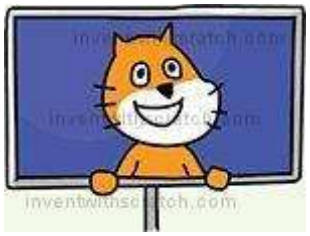
The Energy variable will start at 10 at the beginning of the game and then decrease by 1 each time the player fires an energy blast. The player should be able to fire only if the Energy variable is larger than 0.

Modify the code in the Energy Blast sprite to match the following.



Script ❶ is brand new. It first sets the Energy variable to 10 before entering the **forever** loop. Inside the loop, if Energy is less than 10, it waits for 0.4 seconds before increasing Energy by 1. This way, the Energy value never goes above 10. Script ❷ is slightly modified so that Energy must be greater than 0 in order for the player to fire. When an energy blast is fired, the **change Energy by -1** block decreases the Energy value.

SAVE POINT



Click the green flag to test the code so far. Make sure the Energy variable is visible on the Stage near the Score variable. The Energy variable should be 10 at the start of the game and decrease by 1 each time the player presses the spacebar. When the Energy variable is at 0, pressing the spacebar should not fire any more Energy Blast clones. Also, make sure the Energy variable recharges by 1 about every half second. Then click the red stop sign and save your program.

CHEAT MODE: STARBURST BOMB

The limited energy in *Asteroid Breaker* version 2.0 is more challenging, but let's add a secret cheat to work around it. A cheat to have unlimited energy would be boring, so instead we'll add a special energy bomb that fires in a starburst pattern all around the spaceship.

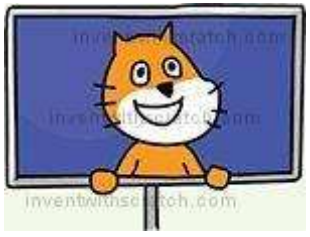
The starburst will fire when the player presses the X key. This code is similar to the regular firing code when the player presses the spacebar.

Add the following code to the Energy Blast sprite:

Like the **when space key pressed** script, this script checks that the player is alive and that the sprite is not a clone. Only the original sprite should run this code, not the clones.

Inside the **if then** block, the Energy Blast sprite moves to the spaceship and points in the direction of the mouse. Then, the sprite clones itself 24 times. After each clone is made, the sprite changes direction by 15 degrees counterclockwise. This results in a starburst of energy blasts in all directions.

SAVE POINT



Click the green flag to test the code so far. Press the X key and watch Energy Blast clones fly out of the Spaceship sprite in all directions. Because this cheat move is a secret, make sure the player can use it no matter what the energy level is. Then click the red stop sign and save your program.

SUMMARY

In this chapter, you built a game that does the following:

- Uses a hockey-puck push style to control a spaceship
- Has x velocity and y velocity variables to keep track of how fast the Spaceship sprite is moving
- Lets sprites wrap around the edges of the Stage
- Has Asteroid clones that can create two smaller clones of themselves
- Has variables Score and Energy that constantly decrease and increase, respectively, over time
- Has frame-by-frame animation for an explosion

This game offers players a real challenge, but as programmers, we had to add those features one by one! The player doesn't directly control the spaceship but instead pushes it. If we stopped coding the Spaceship sprite at that point, the player could just hide in a corner, safe from asteroids, so we made all the sprites wrap around the edges. Even with that addition, the player might have tried to stay still in the center of the Stage. That's when we added random small pushes to the spaceship.

It's difficult to avoid many small asteroids, so the player could have taken it slow and carefully finished off small asteroids before targeting large ones. At that point, we made the score decrease over time to encourage the player to fire faster. The player could also keep blasting willy-nilly without aiming carefully, so we made an Energy variable to limit how fast the player could fire.

Each time you add a feature to your games, keep in mind how it affects gameplay. A game that is too hard to play is frustrating, but a game that is too easy is boring. It's all about finding a balance.

The next game is the most advanced program yet: it's a platformer game like *Super Mario Bros.* or *Super Meat Boy*. Not only will it have jumping and gravity like the *Basketball* game, but you'll be able to design custom levels for it without changing the code!

REVIEW QUESTIONS

Try to answer the following practice questions to test what you've learned. You probably won't know all the answers off the top of your head, but you can explore the Scratch editor to figure out the answers. (The answers are also online at *http://www.nostarch.com/scratchplayground/*.)

1. How does the wrap-around code work?
2. Why does the Energy Blast sprite have an I am a clone variable?
3. What prevents the Asteroid clone from breaking into exponentially more pieces forever?
4. How does the Explosion sprite's code make it look like the spaceship explodes?