Focus on the prize

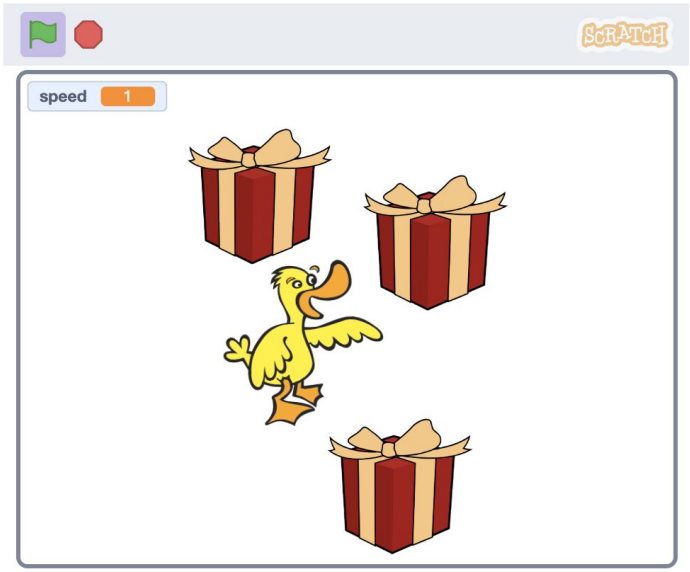
Introduction

What will you make

In this project, you will make a game that will help you to improve your concentration. You will have to watch really carefully to try to follow the gift box with the diamond inside, while the other boxes and sprites try to distract you.

You will learn to -

- use `broadcasts` to communicate between sprites
- detect when two sprites `touch` each other
- use `and` and `not` Boolean operators



How will you open the project

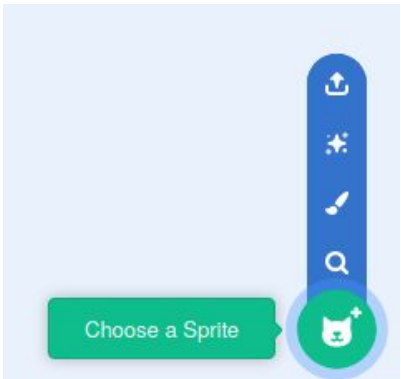
- **Online:** Open a new online Scratch project at https://rpf.io/scratch-new .
- **Offline:** Open a new project in the Scratch offline editor. If you need to, you can download Scratch using this link https://rpf.io/scratchof .

1. Add a gift in a box

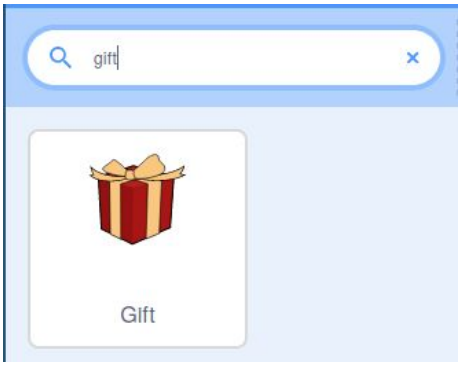
In this step, you will add a gift box that will briefly show what is hidden inside it.

Step 1:
Delete the Scratch cat sprite. To do this, click on the Delete icon in the corner of the image in the Sprite list.

Step 2:
Click on Choose a Sprite to search for a new sprite to add to your game.

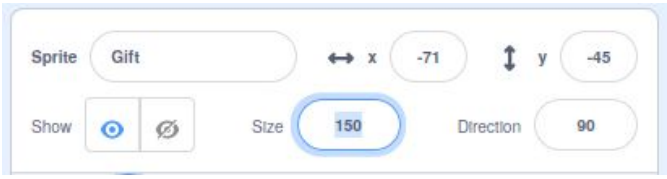


Step 3:
Search for the **Gift** sprite, then add it to your game.

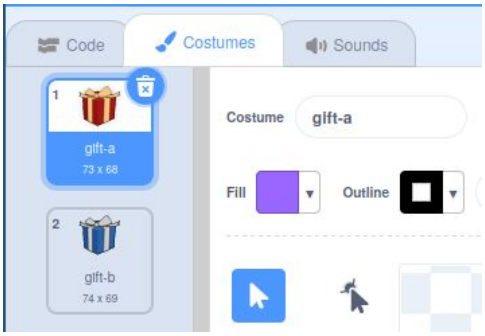


The sprite is a little small at the moment. To increase the sprite's size, you can either use the `set size to` block, or you can change the sprite's **properties**.

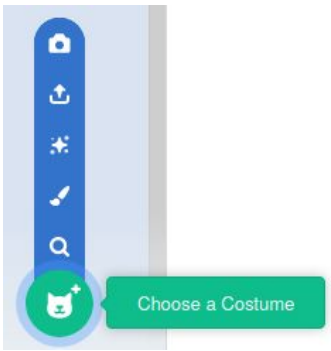
Step 4:
Set the sprite's size to `150` (percent).



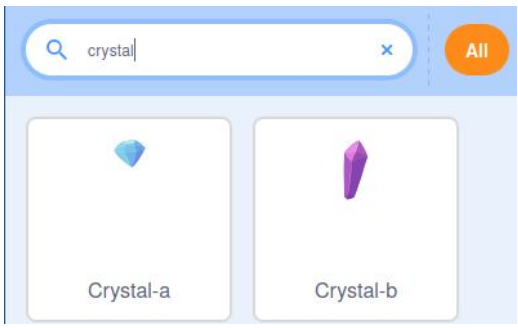
Step 5:
Go to the **Costumes** tab to look at the costumes for the **Gift** sprite. You will see two costumes called **gift-a** and **gift-b**.



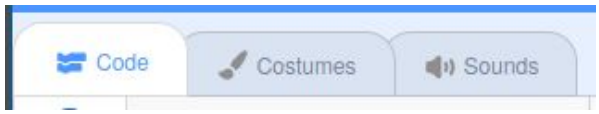
Step 6:
Click on the **Choose a Costume** icon to add a new costume to the sprite.



Step 7:
Choose the gift that you would like to put inside the gift box. In this example, we will use the **Crystal-a** costume.



Step 8:
Switch back to the **Code** tab so that you can begin to work on the program.



Step 9:
Add some blocks to make the starting costume for the **Gift** sprite `switch` to `Crystal-a` and then back to `gift-a` again. Use `wait` blocks in between the costume changes.

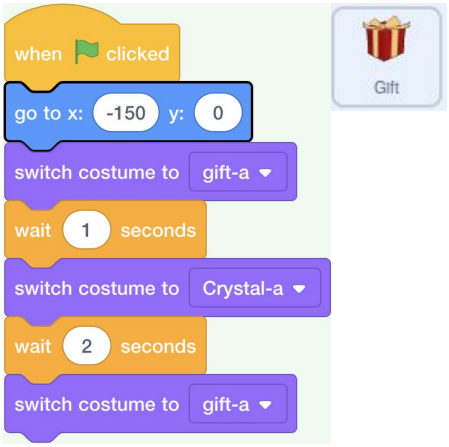
The `wait` blocks give the player a chance to see the costume changes.

Step 10:
Click on the green flag, and make sure that your gift box changes costumes.

2. Move the gift box

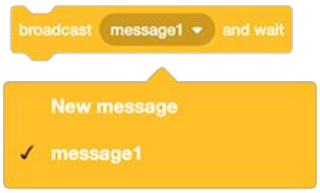
Now that you have a box with a gift inside it, you need to make it move around the screen.

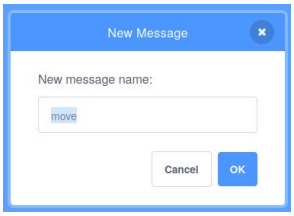
Step 1:
First, use a `go to x: y:` block to give the gift a starting position to the left of the Stage.



In the `Events` blocks menu, select the `broadcast message1 and wait` block.

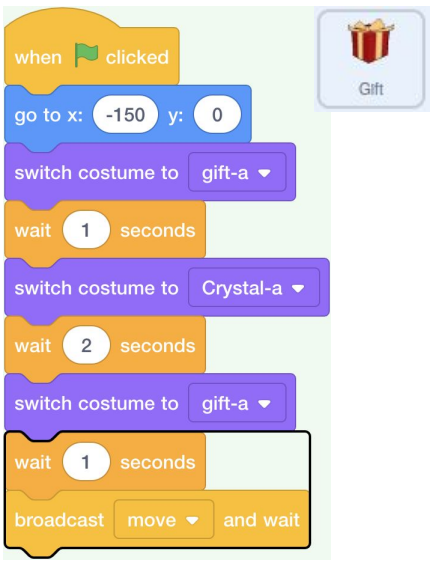
Step 2:
Create a new `broadcast` to start moving. Call your new broadcast `move`.



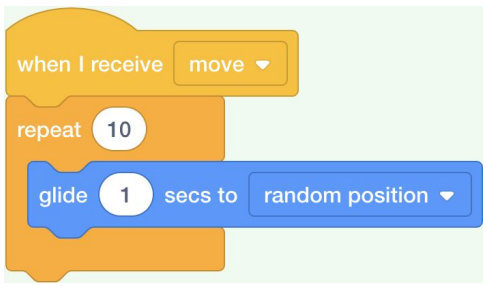


Broadcasts are messages that are sent by a sprite for some or all other sprites to receive.
Broadcasts are very useful for making sure that events in your program all happen simultaneously.

Step 3:
Add a `wait` block, then add the `broadcast move and wait` block.



Step 4:
When the sprite `receives` the `move` broadcast, the sprite can start to `glide` in random directions. You can use a `repeat` block to make this happen `10` times.



Step 5:
Click on the green flag, and check that your sprite starts moving randomly, after the costume changes.

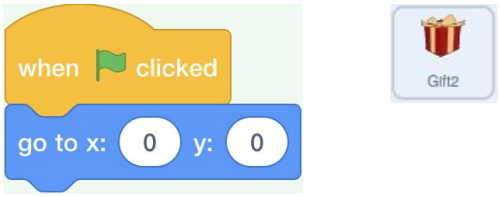
3. Create an empty box

In this step, you will add another gift box, but this one will be empty! It will also move around in random directions.

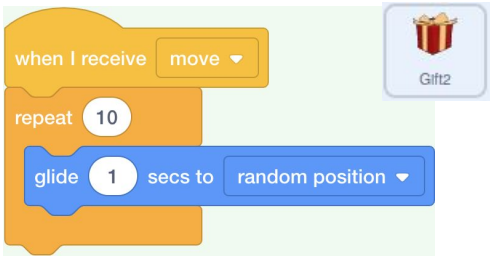
Step 1:
Add a second **Gift** sprite. This one will be named **Gift2** automatically.

Step 2:
Increase the size of **Gift2** to `150` (percent).

Step 3:
Set the **Gift2** sprite to start in the centre of the Stage.



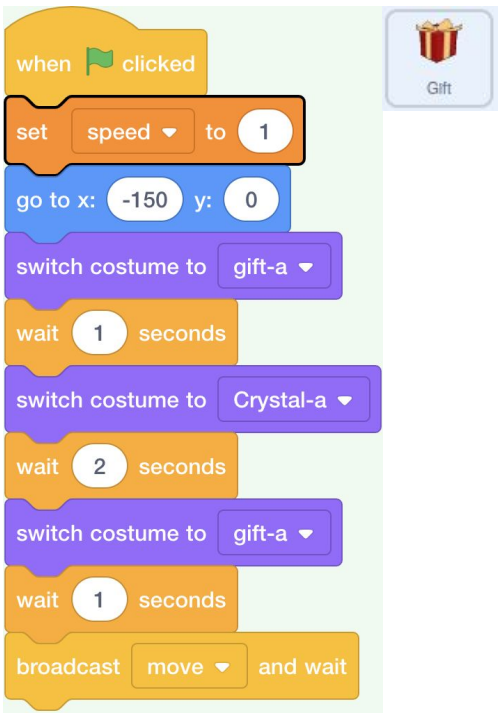
Step 4
Add blocks so that when this sprite `receives` `move`, it also starts to `glide` in random directions.



Step 5:
Click on the green flag, and watch the two sprites move around the Stage.

Step 6:
Create a new `variable` called `speed`, and add it to the `glide 1 secs to random position` block. Add this block to both the **Gift** and **Gift2** sprites.

Step 7:
In the code for the **Gift** sprite, you can add code to `set` the `speed` variable. Choose any value you like.

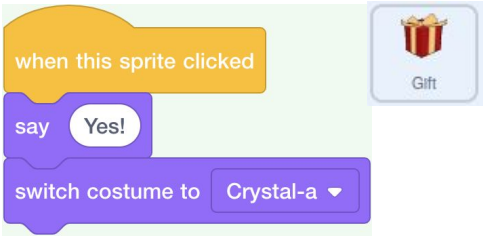


Step 8:
Experiment with different values, until you find a speed you like.

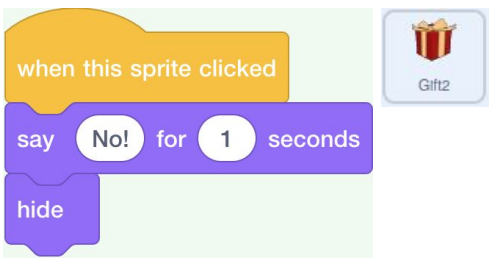
4. Reveal the answers

In this step, you will add code so that the player can click on a gift box to see if it has the crystal inside it.

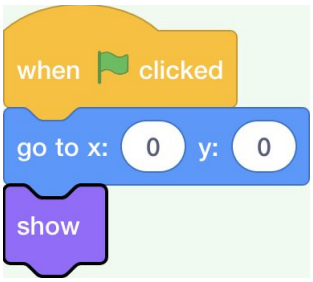
Step 1:
In the code for the **Gift** sprite, add code so that `when this sprite clicked`, it `says` `Yes!` and `switches costume to` the crystal.



Step 2:
Click on the **Gift2** sprite and add blocks so that it `says` `No!`. Change the value of the `say` block to `1` second. Add a `hide` block so the sprite then disappears.



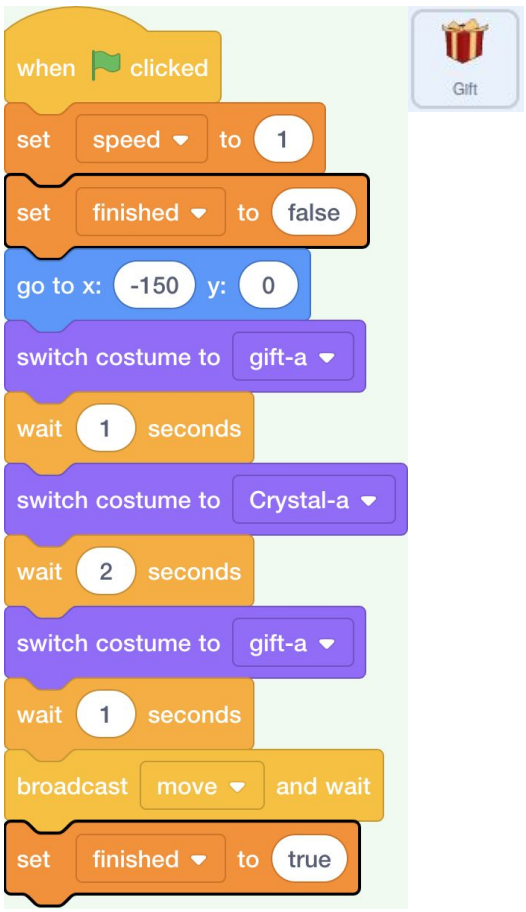
Step 3:
As the **Gift2** sprite can now be hidden, you need to add a `show` block to make sure that it is visible when the program starts.



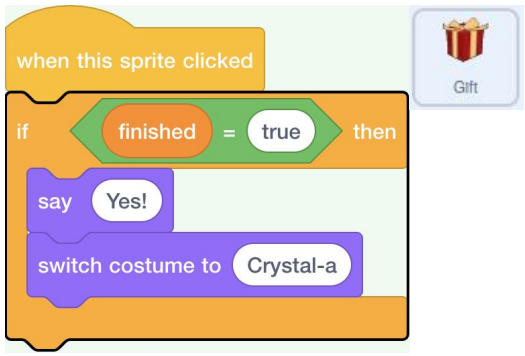
Step 4:
Click on the green flag to run your code, and test what happens when you click on the boxes.

You may notice that you can click on the boxes while they are moving around. You can use a `variable` to stop players cheating like this.

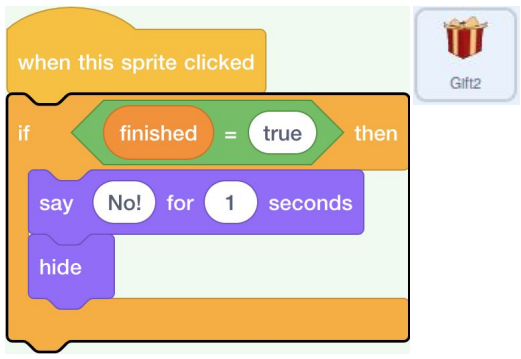
Step 5:
Create a new `variable` called `finished`. Add blocks to `set` the `finished` variable to `false` when the gifts start moving, and then to `true` when the gifts stop moving.



Step 6:
Now, you can use an `if … then` block to check if `finished` is true or false. Add an `Operators` block so that the clicking only has any effect if `finished` = `true`.



Step 7:
Add the same `if` condition to the **Gift2** sprite.



Step 8:
Test your program, and you should see that you can only click on the sprites when they have finished moving around.

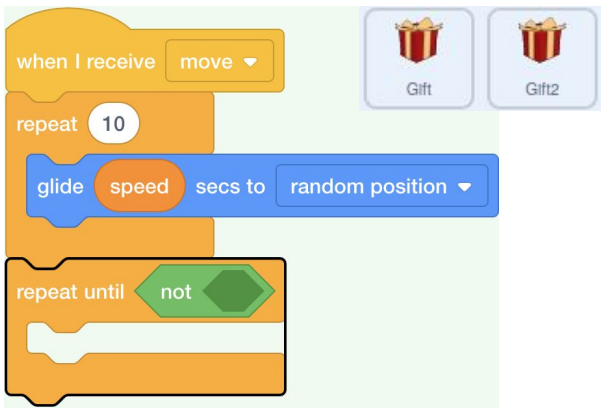
5. Add more gifts

In this step, you will make sure that the gifts are not touching each other, or the edge of the screen, when they stop moving. Then, you will add a third gift. The colour of the gift box's ribbon can be used to detect if two boxes are touching each other.

You need to add the scripts shown below to **both** the **Gift** and **Gift2** sprites.

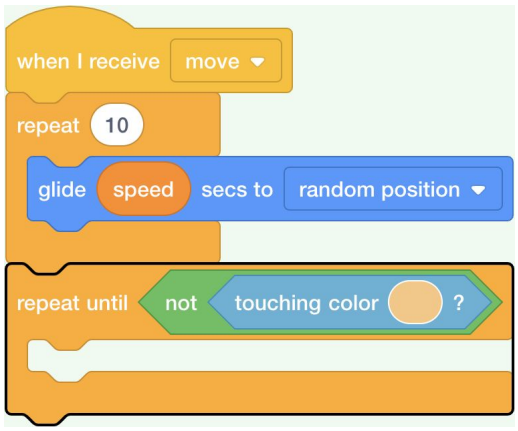
Step 1:
You can use the `not` operator to turn **true** into **false**, or **false** into **true**.

Add a `repeat until` block to the bottom of your script, and place a `not` block inside it.

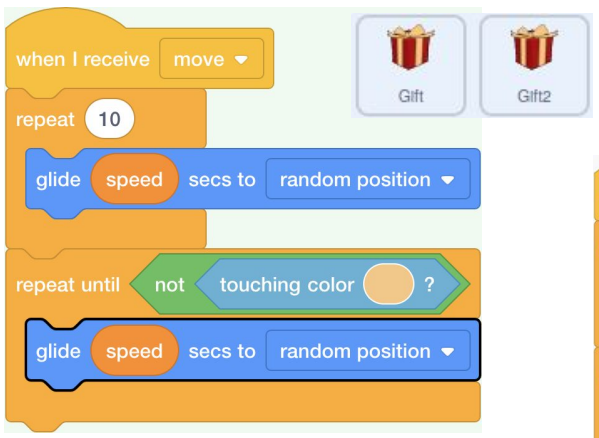


Step 2:
A `Sensing` block that detects if a sprite is touching a particular colour would return true when the sprite touches that colour.

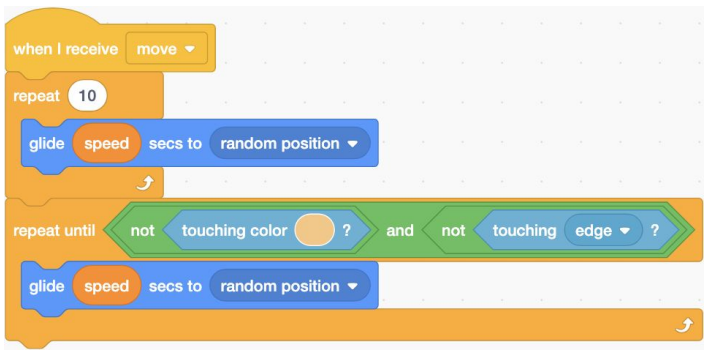
Add in a `touching color` block, and use the **Colour picker** tool to select the colour of the ribbon.



Step 3:
Within the repeat until block, add another block to keep the box moving to a random position.



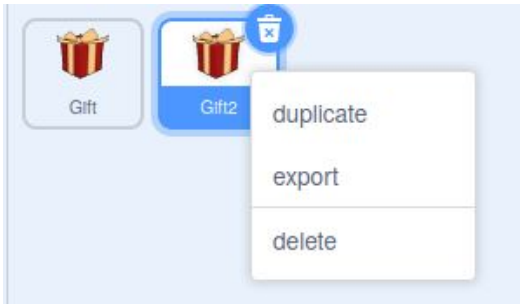
Step 4:
Add an and block into the repeat until condition. Then, add in a not block and a touching edge block to go with the not touching color blocks.



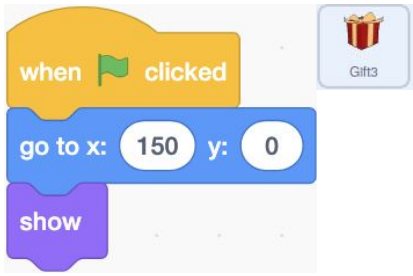
Step 5:
Test your code, to make sure that the boxes now do not stop at the edge of the Stage.

Now that the code for the **Gift** and **Gift2** sprites is complete, you can duplicate the **Gift2** sprite to add a third box to the game.

Step 6:
Right-click on the **Gift2** sprite in the Sprite list and select **duplicate** in the drop-down menu.



Step 7:
All the code has been duplicated as well, but you need to give the **Gift3** sprite a different starting position.



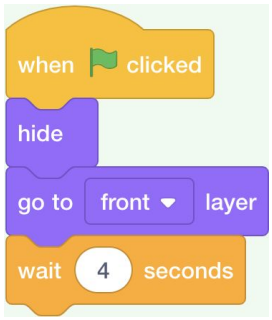
Step 8:
To test your game again, click on the green flag.

6. Make a distraction

To finish the game, you can add a distraction, to try to stop the player keeping their eyes on the gift box.

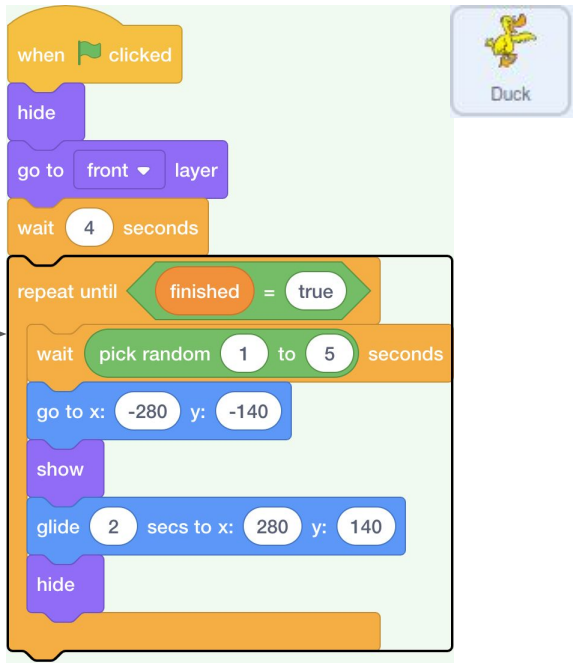
Step 1:
Add a new sprite to your game. You can choose any sprite you like, but in this example, we will use the **Duck** sprite.

Step 2:
Add the following blocks to the **Duck** sprite.



Step 3:
Next, the **Duck** sprite should `wait` a `random` number of seconds, then `glide` across the Stage and `hide`. This can keep happening until the game has finished.

Step 4:
Test your game and see if you can keep track of the gift box with a distracting duck flying across the Stage.



7. Add more

Can you add more distractions to your game to make it even harder to keep your eye on the correct gift box?

Think about some of the things that you have learned in other Scratch projects that might make the game even harder, such as changing costumes, adding graphic effects, changing sizes, and moving and turning.