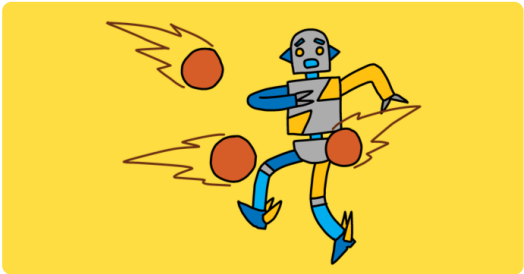
Projects

Dodgeball

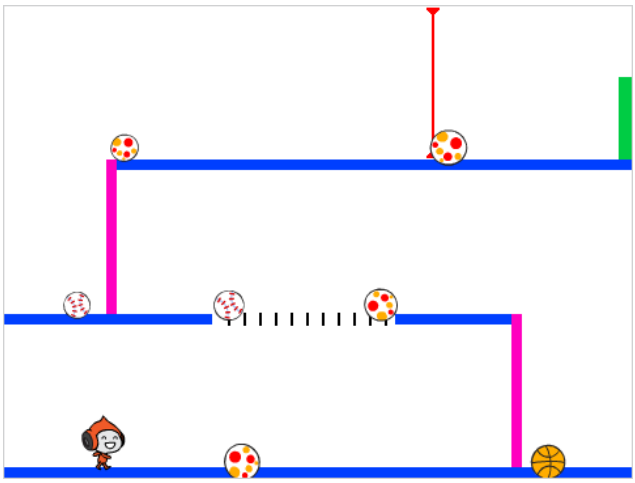
Create a platform game in which you dodge moving balls to reach the end of the level

Scratch



Step 1 Introduction

You'll learn how to create a platform game in which the player has to dodge moving balls to reach the end of the level.



What you will need

Hardware

- A computer capable of running Scratch 3

Software

- Scratch 3 (either **online** (https://scratch.mit.edu/projects/editor/) or **offline** (https://scratch.mit.edu/download/))

Downloads

You can **find the downloads here** (http://rpf.io/p/en/dodgeball-go).

What you will learn

- How to use the keyboard to control a sprite
- How to use the `if, then, else` Scratch block
- How to clone a sprite

Additional notes for educators

If you need the solution to this project, **you can find it here** (http://rpf.io/p/en/dodgeball-get).

Step 2 Character movement

Start by creating a character that can move left and right, and can climb up ladders.

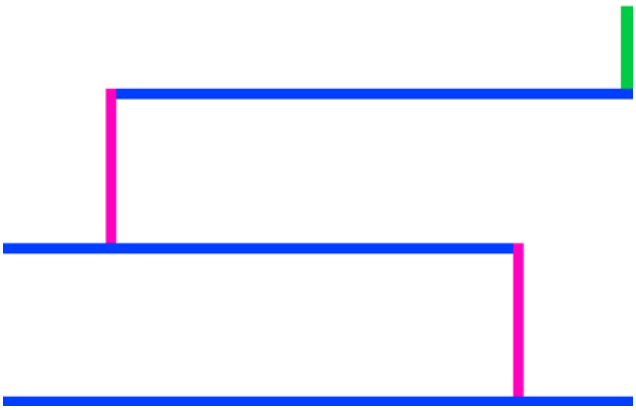
Open the 'Dodgeball' Scratch starter project.

Online: open the starter project at **rpf.io/dodgeball-on** (http://rpf.io/dodgeball-on).

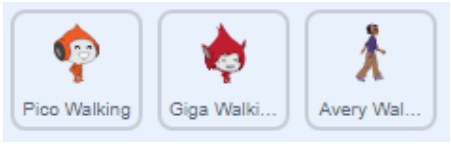
If you have a Scratch account you can make a copy by clicking **Remix**.

Offline: download the starter project from **rpf.io/p/en/dodgeball-get** (http://rpf.io/p/en/dodgeball-get) and then open it using the offline editor.

The project contains a backdrop with platforms:

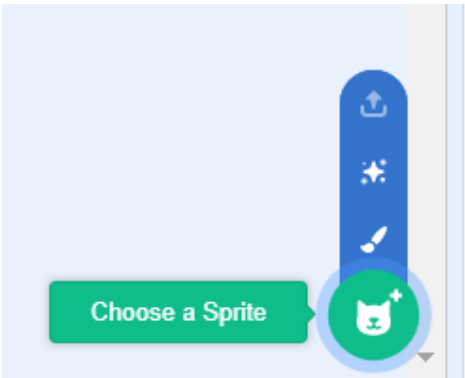


Choose a new sprite as the character the player will control, and add it to your project. It's best if you choose a sprite with multiple costumes, so that you can make it look as though it's walking.

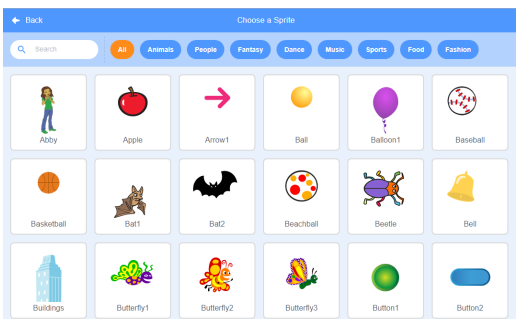


Adding a Scratch sprite from the Library

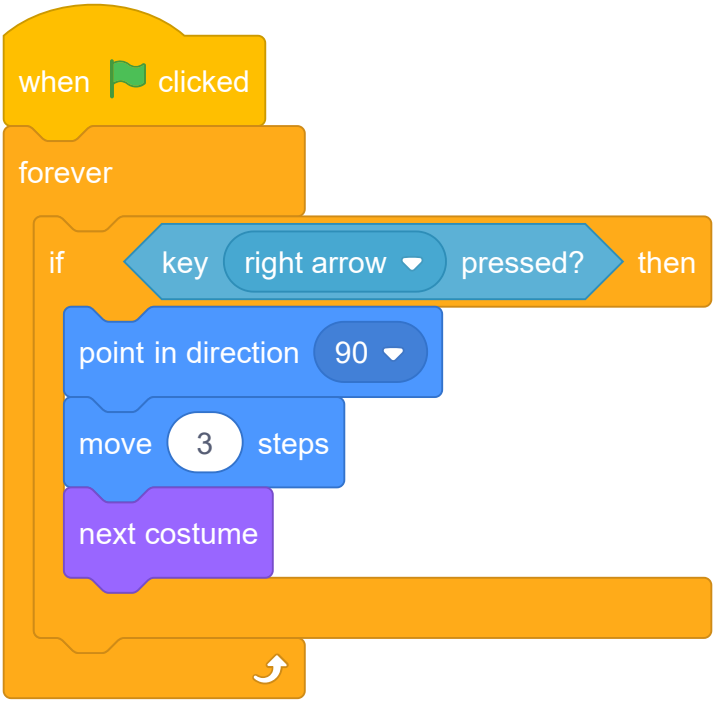
- Click **Choose a sprite** to see the library of all Scratch sprites.



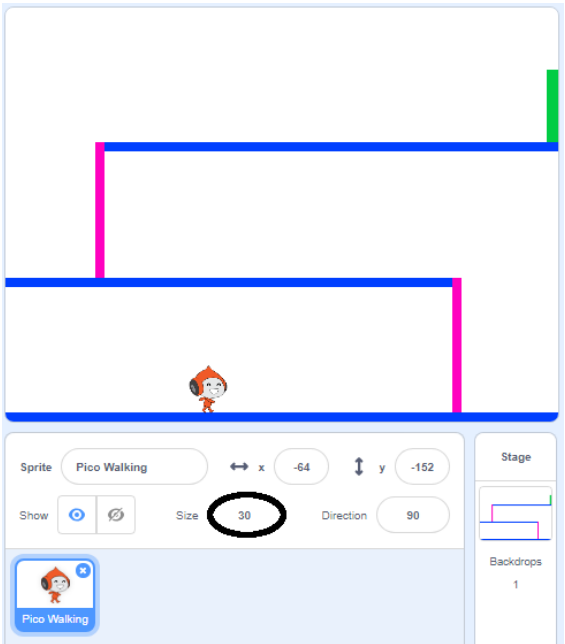
- You can search or browse sprites by theme. Click on a sprite to add it to your project.



Add code blocks to your character sprite so that the player can use the arrow keys to move the character around. When the player presses the right arrow, the character should point right, move a few steps, and change to the next costume:



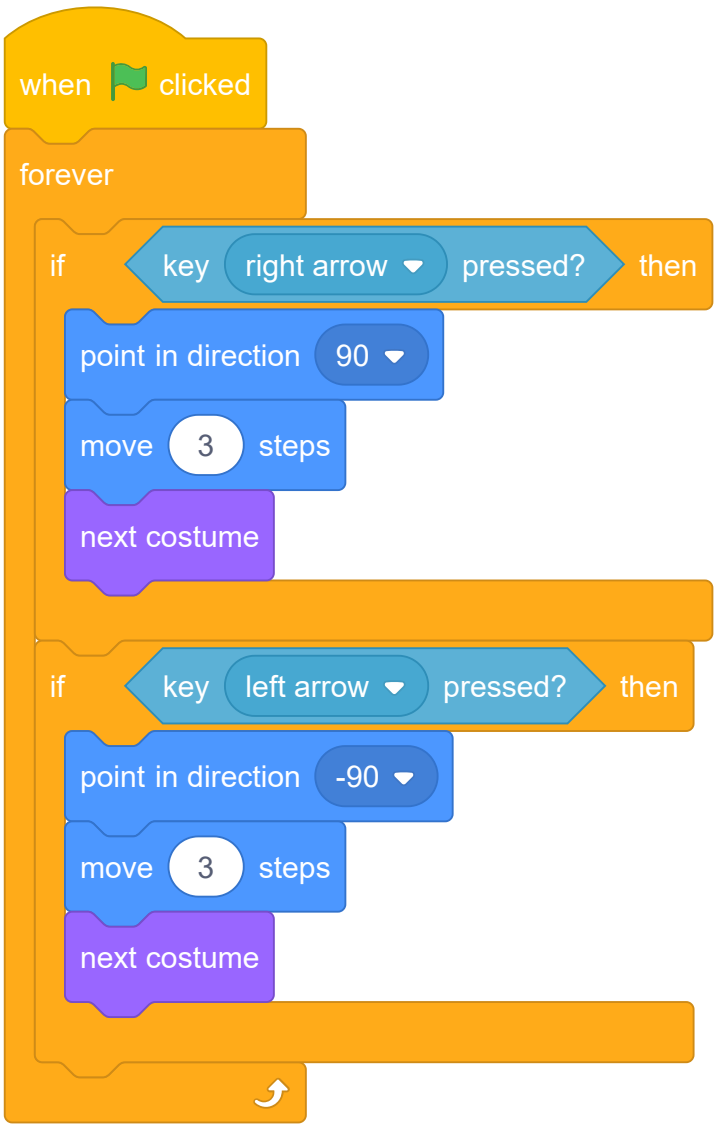
If your sprite doesn't fit, adjust its size.



Test out your character by clicking the flag and then holding down the right arrow key. Does your character move to the right? Does your character look like it is walking?

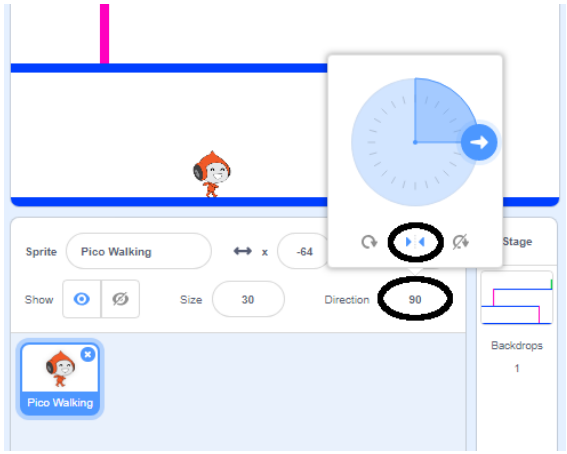
Add code blocks to the character sprite's `forever` loop so that it walks left if the left arrow key is pressed.

Your code should look like this now:

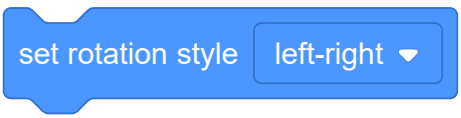


Test your new code to make sure that it works. Does your character turn upside-down when walking to the left?

If so, you can fix this by clicking on the **direction** of your character sprite, and then clicking on the left-right arrow.

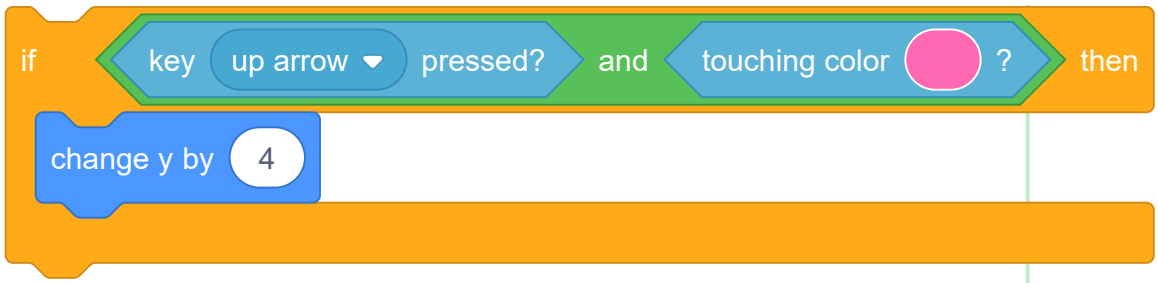


Or if you prefer, you can also fix the problem by adding this block to the start of your character's script:

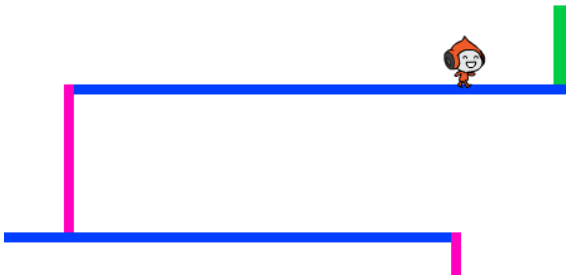


To climb a pink ladder, your character sprite should move a few steps upwards on the Stage whenever the up arrow is pressed **and** the character is touching the correct colour.

Add inside your character's `forever` loop to `change` the character's `y` (vertical) position `if` the `up arrow is pressed` and the character is `touching the colour pink`.



Test your code. Can you make the character climb the pink ladders and get to the end of the level?



Challenge!

Challenge: completing the level

Can you add more code blocks to your character sprite to make the sprite say something if it gets to the green door?

Step 3 Gravity and jumping

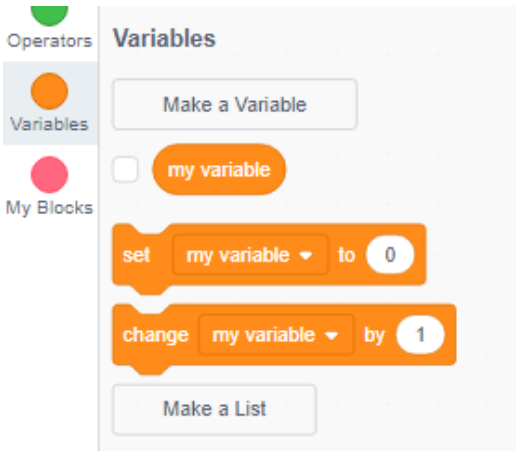
Now you're going to make your character move more realistically: you're going to add gravity to your game and give the character the ability to jump.

In the game, move your character so that it walks off a platform. Do you see that it can walk into empty space?

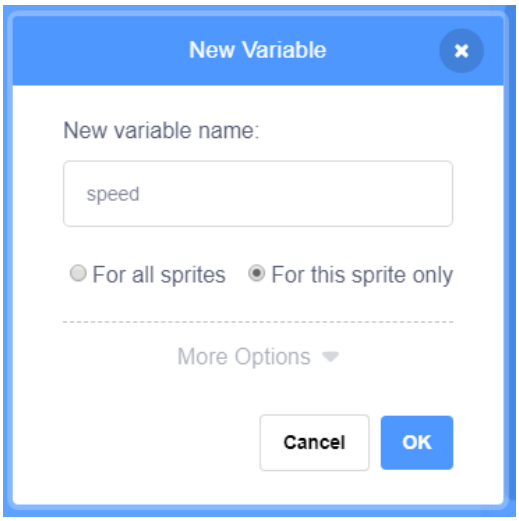
To fix this, add gravity to your game. To do this, create a new variable called `gravity`.

Add a variable in Scratch

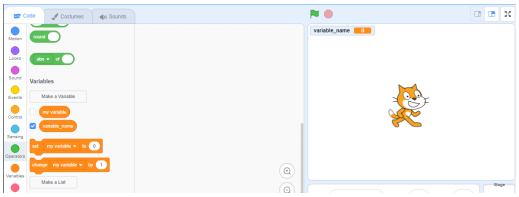
- Click on **Variables** in the Code tab, then click on **Make a Variable**.



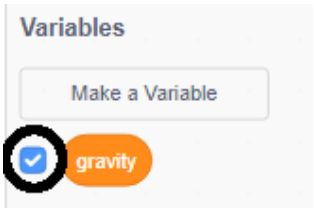
- Type in the name of your variable. You can choose whether you would like your variable to be available to all sprites, or to only this sprite. Press **OK**.



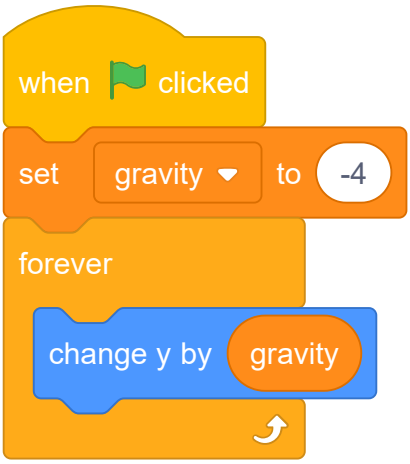
- Once you have created the variable, it will be displayed on the Stage, or you can untick the variable in the Scripts tab to hide it.



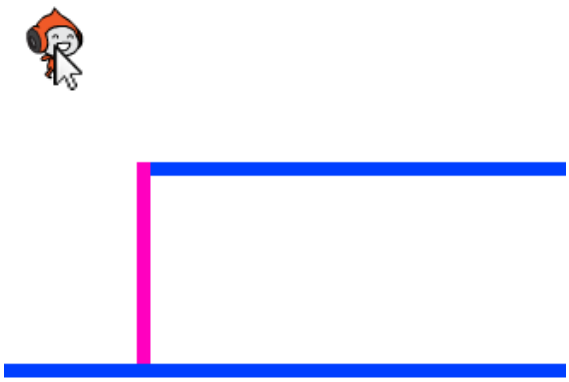
You can hide this variable from your Stage if you want to.



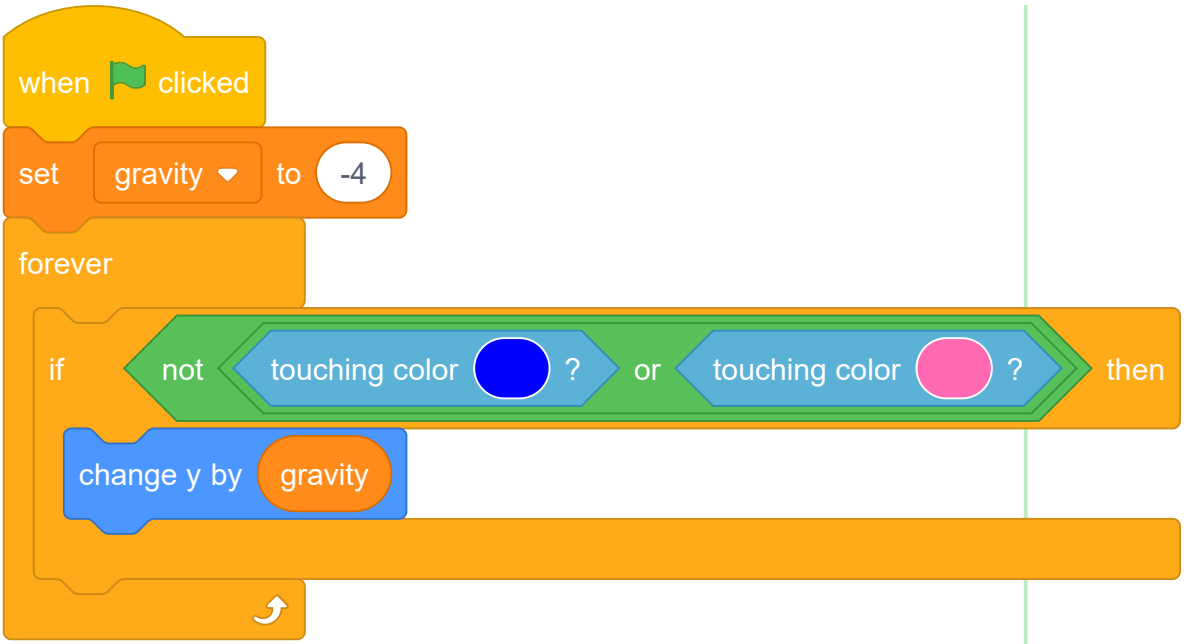
Add these new code blocks that set `gravity` to a negative number and use the value of `gravity` to repeatedly change your character's y-coordinate:



Click the flag, and then drag your character to the top of the Stage. What happens? Does the gravity work as you expect?

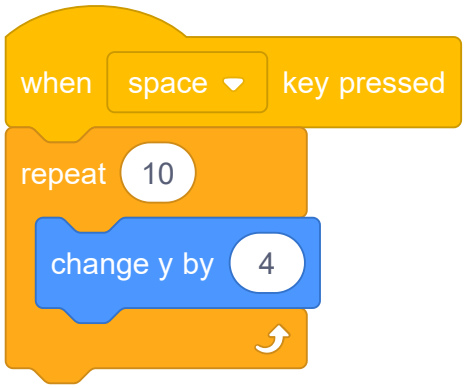


Gravity shouldn't move the character sprite through a platform or a ladder! Add an if block to your code to only let the gravity work when the character is in mid-air. The gravity code should then look like this:



Test the game again to see whether gravity works correctly now. Does your character sprite stop falling when it touches a platform or a ladder? Can you make the character walk off the edge of platforms and fall onto the level below?

Now add code to make your character jump whenever the player presses the `space` key. One very easy way to do this is to move your character up a few times:

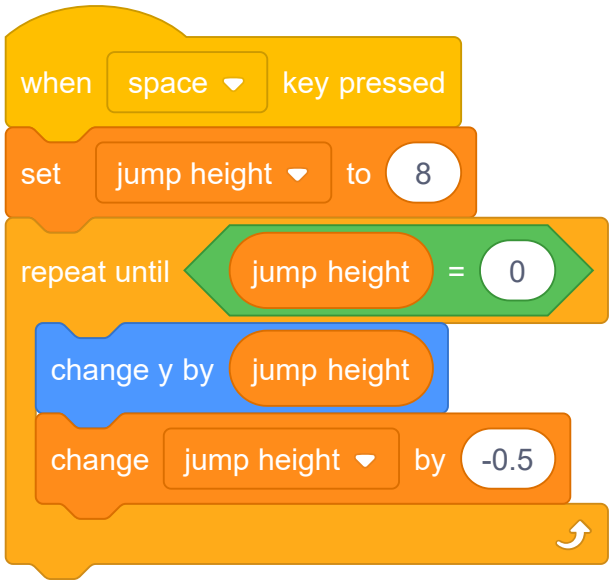


Because gravity is constantly pushing your character down by 4 pixels, you need to choose a number greater than `4` in your `change y by (4)` block. Change the number until you're happy with the height the character jumps.

Test out your code. Notice that the jumping movement isn't very smooth. To make jumping look smoother, you need to move your character sprite by smaller and smaller amounts, until it is not rising any higher.

To do this, create a new variable called `jump height`. Again, you can hide this variable if you prefer.

Delete the jumping code you added to your character sprite, and add this code instead:



This code moves your character up by 8 pixels, then 7.5 pixels, then 7 pixels, and so on, until it does not rise any higher. This makes jumping look much smoother.

Change the value of the `jump height` variable that is set before the `repeat` starts. Then test your game.

Repeat these two steps until you're happy with how high the character jumps.

Challenge!

Challenge: better jumping

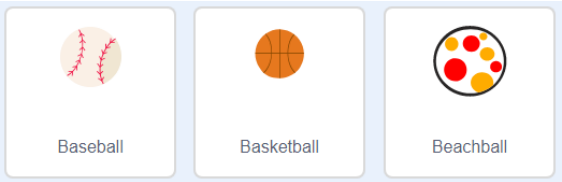
Even if the character is already in mid-air, it jumps whenever the player presses the `space` bar. You can see this behaviour if you hold down the `space` bar.

Can you change the character sprite's code for jumping so that your character can only jump `if` it touches a blue platform?

Step 4 Dodging balls

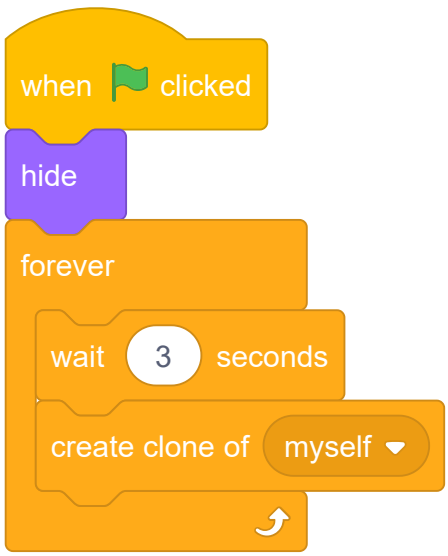
Your character can move and jump now, so it's time to add some balls that the character has to avoid.

Create a new ball sprite. You can choose any type of ball you like.



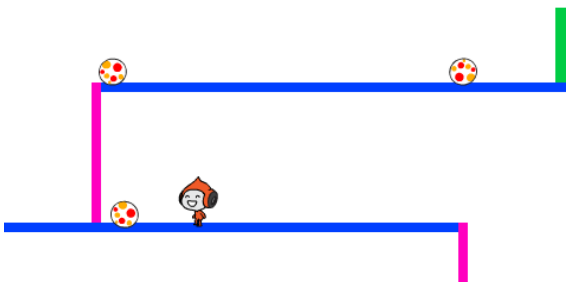
Resize the ball sprite so that the character can jump over it. Try making the character jump over the ball to test whether the ball is the right size.

Add this code to your ball sprite:

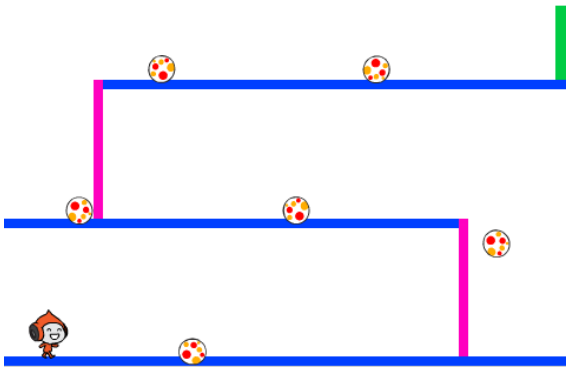


This code creates a new clone of the ball sprite every three seconds. Each new clone moves along the top platform and then drops.

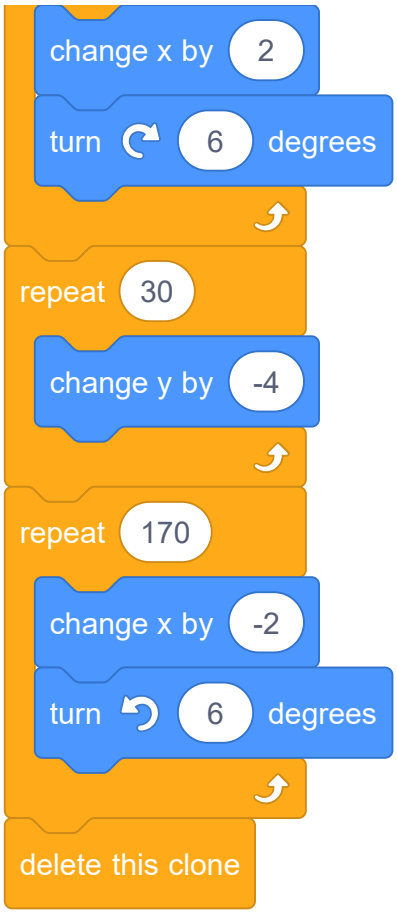
Click the flag to test the game.



Add more code to your ball sprite so that clones of it move across all three platforms.



The code for your ball sprite clones should look like this:



Now add some code blocks to broadcast (send) a message if your character gets hit by a ball!

Add this code to your ball sprite:

Finally, add code blocks to your character sprite to make it move back to its starting position when it receives the `hit` message:

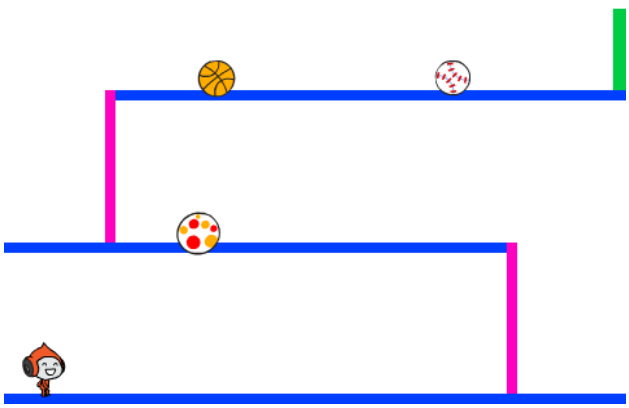
Test out your code. Check whether the character moves back to the start after touching a ball.

Challenge!

Challenge: random balls

The balls that your character has to dodge all look the same, and they appear at regular three-second intervals. Can you add code to your game so that the balls:

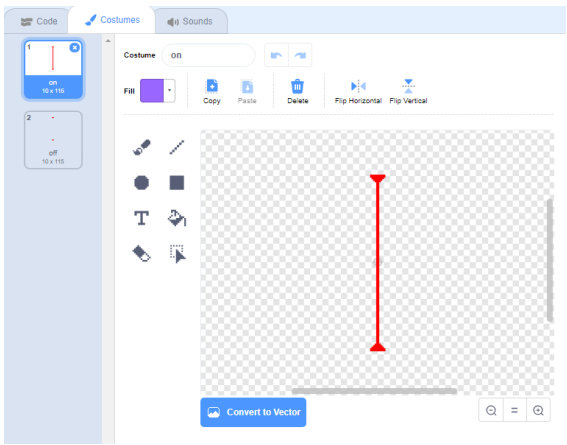
- Don't all look the same?
- Appear after a `random` amount of time?
- Are a random size?

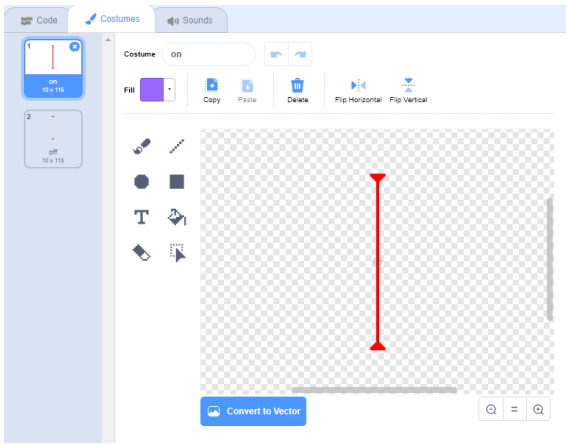


Step 5 Lasers!

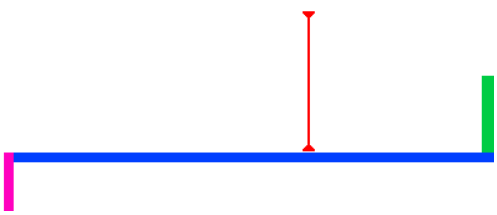
To your game a little harder to complete, you are going to add lasers!

Add a new sprite to your game and call it `laser`. It should have two costumes: one called 'on', and one called 'off'.

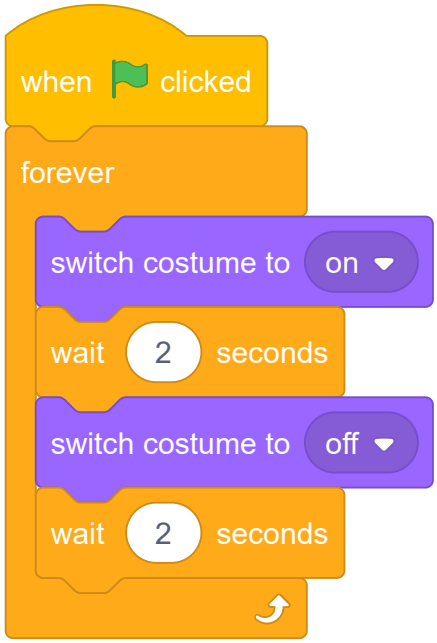




Place your new laser sprite between two platforms.



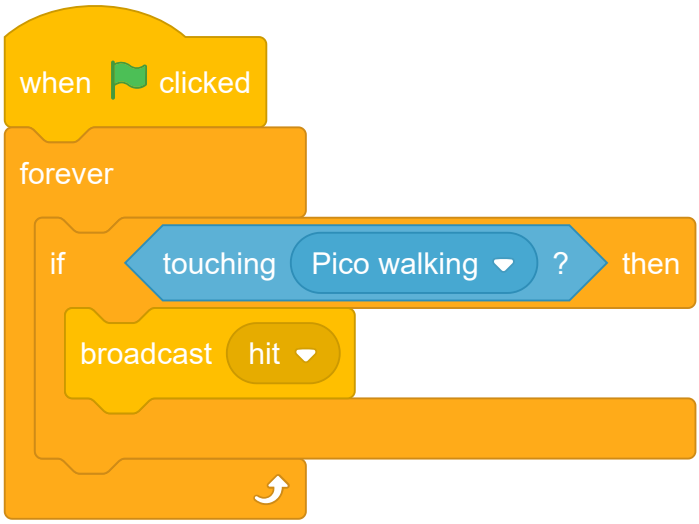
Add code to your laser sprite to make it switch between its two costumes.



If you prefer, you can change the code shown above so that the sprite `waits` a `random` amount of time between costume changes.

Finally, add code to your laser sprite so that the laser sprite broadcasts a 'hit' message when it touches the character sprite.

This is the code you should add:



You don't need to add any extra code to your characters sprite, because the character sprite already knows what to do when it receives the `broadcast 'hit'`!

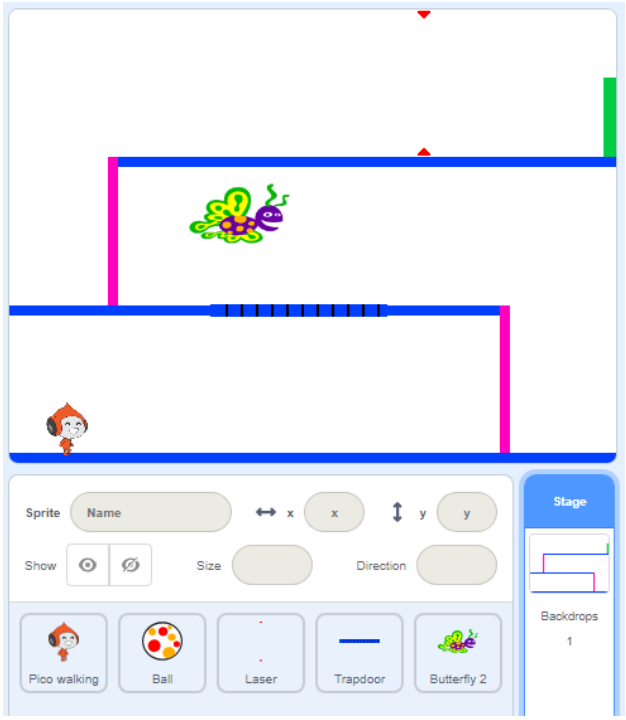
Test out your game to see if you can move the character past the laser. If the laser is too easy or too hard to avoid, change the `wait` times in the code for the laser sprite.

Challenge!

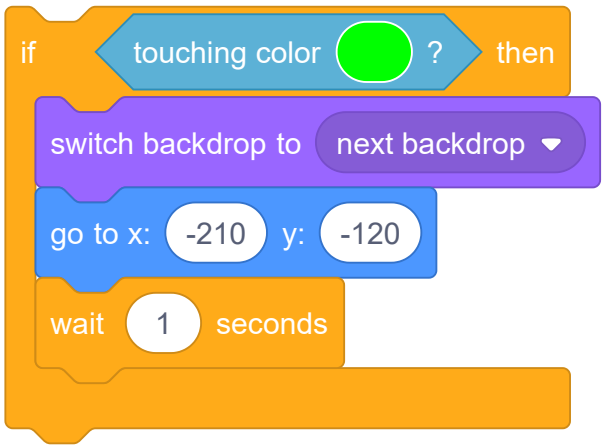
Challenge: more obstacles

If you think your game is still too easy, you can add more obstacles to it. The obstacles can be anything you like! Here are some ideas:

- A dangerous butterfly
- Platforms that appear and disappear
- Falling tennis balls that must be avoided



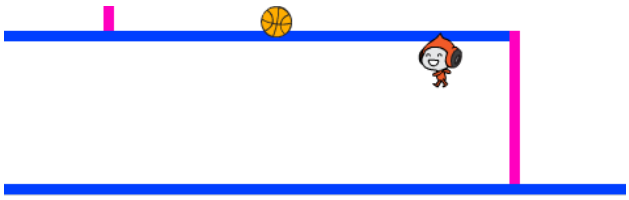
You could even design another backdrop to create the next level. Then add code so that, when your character reaches the green door, the game switches to the new background:



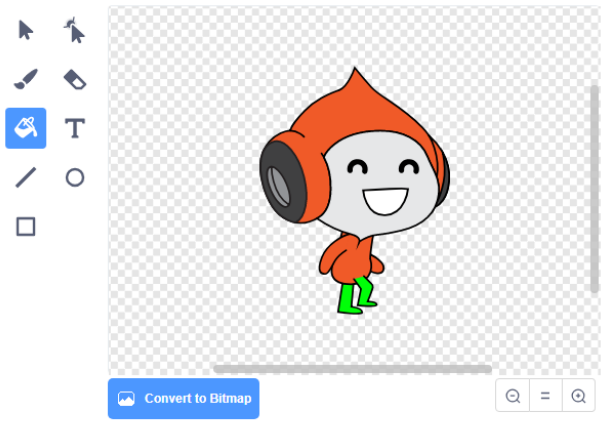
Challenge!

Challenge: improved gravity

There's one other small bug in your game: gravity doesn't pull the character sprite downwards if **any** part of the sprite is touching a blue platform. So even if the sprites head touches a platform, the sprite doesn't fall! You can test this yourself: make your character climb most of the way up a ladder, and then move the character sideways beneath a platform:



To fix the bug, you first need to give your character sprite new trousers that have a different colour (on **all** costumes).



Then replace this code block:

with this code block:

To make sure you've fixed the bug, test the game after you've made these changes!

Challenge!

Challenge: more lives

Right now, your character goes back to its starting position when it gets hit. Can you also give the character sprite three `lives`, and make it lose one life when it gets hit? Here's how your game could work:

- The game starts with three lives for the character
- Whenever the character gets hit, it loses one life **and** moves back to the start
- If there are no lives left, the game ends

Step 6 What next?

Have a go at our **Brain game** (https://projects.raspberrypi.org/en/projects/brain-game?utm_source=pathway&utm_medium=whatnext&utm_campaign=projects) project, in which you can create your own maths quiz.