Project

Beginners Course To Scratch
yorkpijam.com
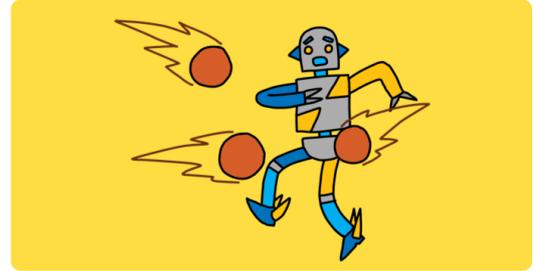
Projects

Dodgeball

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

Scratch

Step 1   **Introduction**

In this project you'll learn how to create a platform game, in which you
have to dodge the moving balls and reach the end of the level.

Additional information for club leaders

If you need to print this project, please use the **Printer friendly
version** [(https://projects.raspberrypi.org/en/projects/dodgeball/pri
nt)](https://projects.raspberrypi.org/en/projects/dodgeball/print).

 Club leader notes

Introduction:

In this project, children will learn how to make a platform game,
making use of Scratch concepts and skills that have been learnt in
previous projects.

Resources

For this project, Scratch 2 should be used. Scratch 2 can either be
used online at **jumpto.cc/scratch-on** [(http://jumpto.cc/scratch-o](http://jumpto.cc/scratch-on)
[n)](http://jumpto.cc/scratch-on) or can be downloaded from **jumpto.cc/scratch-off** [(http://jumpt](http://jumpto.cc/scratch-off)
[o.cc/scratch-off)](http://jumpto.cc/scratch-off) and used offline.

The 'Project Materials' link for this project contains the following
resources:

Club leader Resources

You can find a completed version of this project **online** [(http://scratc](http://scratch.mit.edu/projects/39740618/#editor) [h.mit.edu/projects/39740618/#editor)](http://scratch.mit.edu/projects/39740618/#editor), or it can be downloaded by clicking the 'Project Materials' link for this project, which contains:

- Dodgeball.sb2

Project Resources

For this project, club members can make use of external resources that aren't available in the Scratch library:

- background.png.

For club members that aren't able to access these project resources, there is a Scratch project with the external resources pre-loaded. This is available online at **jumpto.cc/dodge-resources** [(http://jumpto.c](http://jumpto.cc/dodge-resources) [c/dodge-resources)](http://jumpto.cc/dodge-resources), or in the downloadable project materials, which contains:

- DodgeballResources.sb2

Learning Objectives

- This project consolidates learning of previous programming skills learnt.

This project covers elements from the following strands of the **Raspberry Pi Digital Making Curriculum** [(http://rpf.io/curriculum)](http://rpf.io/curriculum):

- **Combine programming constructs to solve a problem.** [(http](https://www.raspberrypi.org/curriculum/programming/builder) [s://www.raspberrypi.org/curriculum/programming/builder)](https://www.raspberrypi.org/curriculum/programming/builder)

Challenges

- "Improved jumping" - using an additional `if` block to fix a multiple-jump bug;
- "Random balls" - using the `random` block to vary the ball costume, size and delay between clones;
- "More obstacles" - adding additional sprites to the game;
- "Improved gravity" - using an additional `if` block to fix a bug in the `gravity`;
- "More lives" - addition of a `lives` variable.

    ⓘ   **Project materials**

Project resources

- **Online Scratch 2 project containing external resources** (htt
  p://jumpto.cc/dodge-resources)
- **Downloadable Scratch 2 project containing external
  resources** (https://projects-static.raspberrypi.org/projects/d
  odgeball/cb86c08603640312bb2cbc42511ec07997705dcc/e
  n/resources/DodgeballResources.sb2)
- **background.png** (https://projects-static.raspberrypi.org/proj
  ects/dodgeball/cb86c08603640312bb2cbc42511ec0799770
  5dcc/en/resources/background.png)

Club leader resources

- **Online completed Scratch 2 project** (http://scratch.mit.edu/pr
  ojects/39740618/#editor)
- **Downloadable completed Scratch 2 project** (https://projects-
  static.raspberrypi.org/projects/dodgeball/cb86c086036403
 12bb2cbc42511ec07997705dcc/en/resources/Dodgeball.sb2)

| Step 2 | **Character movement** |

Let's start by creating a character that can move left and right, as well
as climb up ladders.

- Open the 'Dodgeball' Scratch project online at **jumpto.cc/dodge-
  go** (http://jumpto.cc/dodge-go) or download from
  jumpto.cc/dodge-get (http://jumpto.cc/dodge-get) and then
  open if you are using the offline editor.

  The project contains a background with platforms:

- Add a new sprite, which will be your character. It's better if you
  choose a sprite with multiple costumes, so that you can make it look
  as though it's walking.

- Let's use the arrow keys to move your character around. When the
  player presses the right arrow, you want your character to point right,
 move a few steps and change to the next costume:
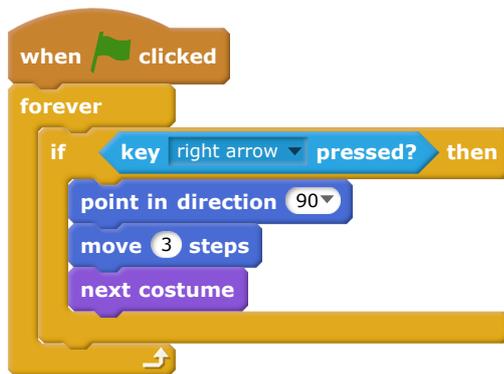
- Test out your character by clicking the flag and then holding down the right arrow key. Does your player move to the right? Does your character look like they are walking?

- To move your character to the left, you'll need to add another `if` block inside your `forever` loop, which moves your character to the left.

- 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 bug by clicking the (i) icon on your character sprite, and then clicking the left-right arrow.

  Or, if you prefer you can add this block to the start of your character's script:

  ```
  set rotation style [left-right v]
  ```

- To climb a pink ladder, your character should move up slightly whenever the up arrow is pressed and they're touching the correct colour. Add this code inside your character's `forever` loop:
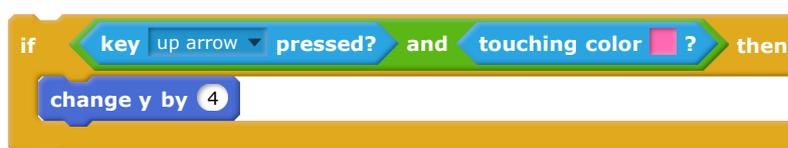
- Test your character - can you climb the pink ladders and get to the end of your level?

Step 3   Challenge: Completing the level

Can you add more code to your character, so that they say something **if** they get to the green door?

## Step 4   Gravity and jumping

Let's make your character move more realistically, by adding gravity and allowing them to jump.

- You may have noticed that your character can walk off a platform into mid-air. Try to walk off of a platform and see what happens.

- To fix this, let's add gravity to your game. Create a new variable called **gravity**. You can hide this variable from your stage if you want to.

- Add this new code block, which sets the gravity to a negative number, and then uses this to repeatedly change your character's y-coordinate.

```
when ⚑ clicked
set gravity ▾ to -4
forever
    change y by gravity
```
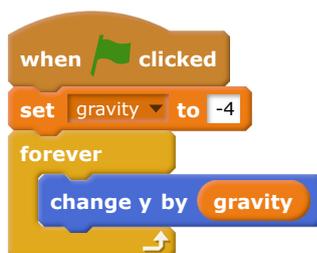
- Click the flag, and then drag your character to the top of the stage. What happens? Does the gravity work as you expected?

- Gravity shouldn't move your character through a platform or a ladder! Add an `if` block to your code, so that the gravity only works when your character is in mid-air. The gravity code should now look like this:
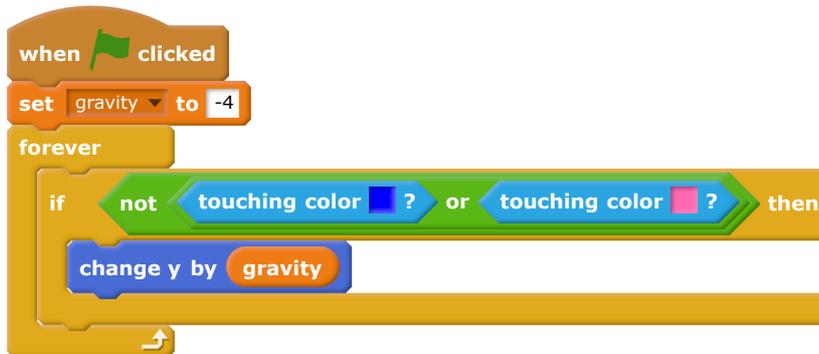

- Test the gravity again. Does your character stop when they are on a platform or a ladder? Can you walk off the edge of platforms to the level below?

- Let's also make your character jump when the player presses the space bar. One very easy way to do this is to move your character up a few times, using this code:
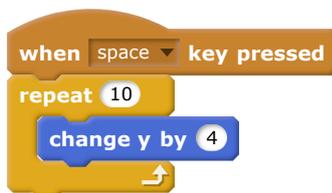
As gravity is constantly pushing your character down by 4 pixels, you need to choose a number greater than 4 in your `change y by (4)` block. Change this number until you're happy with the height your character jumps.

- If you test out this code, you'll notice that it works, but the movement isn't very smooth. To make jumping look smoother, you'll need to move your character by smaller and smaller amounts, until they're not jumping anymore.

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

- Delete the jumping code you added to your character, and replace it with this code:
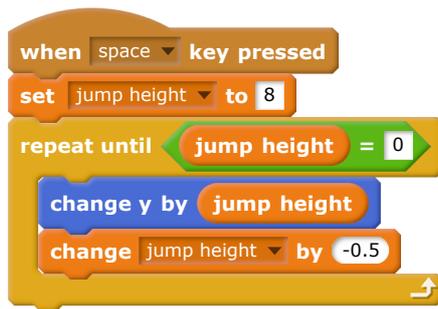
This code moves your character up by 8 pixels, then 7.5 pixels, then 7 pixels, and so on, until your character has finished jumping. This makes jumping look much smoother.

- Change the starting value of your `jump height` variable and test it until you're happy with the height your character jumps.

Step 5   Challenge: Improved jumping

Your character is able to jump whenever the spacebar is pressed, even if they're already in mid-air. You can test this by just holding down the spacebar. Can you fix this, so that your character can only jump `if` they're touching a blue platform?

Step 6   Dodging balls

Now that you've got your character moving around, let's add some balls for your character to avoid.

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

- Resize your ball, so that your character can jump over it. Try jumping over the ball to test it.

- Add this code to your ball:

  This code creates a new ball clone every 3 seconds. Each new clone moves along the top platform.

- Click the flag to test this out.

- Add more code to your ball sprite, so that they move across all 3 platforms.

- Finally, you'll need code for when your character gets hit by a ball! Add this code to your ball sprite:

- You'll also need to add code to your character, to move back to the start when they're hit:
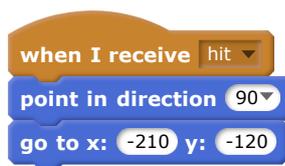
- Test out your character and see if they go back to the start when they've been hit by a ball.

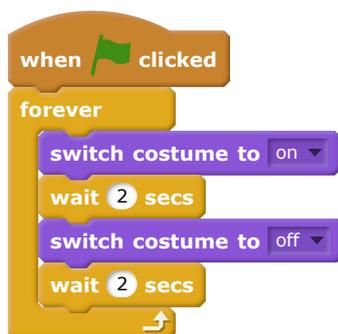## Step 7   Challenge: Random balls

The balls your character has to dodge all look the same, and always appear every 3 seconds. Can you improve them, so that they:

- don't all look the same?
- appear after a random amount of time?
- are a random size?

Step 8   Lasers!

Let's make your game a little harder to complete, by adding lasers!

- Add a new sprite to your game, called 'Laser'. It should have 2 costumes, called 'on' and 'off'.

- Place your new laser anywhere you like, between 2 platforms.

- Add code to your laser, to make it switch between the 2 costumes.

```
when [flag] clicked
forever
    switch costume to on
    wait 2 secs
    switch costume to off
    wait 2 secs
```

If you prefer, you can `wait` a `random` amount of time between costume changes.

- Finally, add code to your laser, so that the 'hit' message is broadcast when the laser touches your character. This code will be very similar to the code you added to your ball sprite.

  You don't need to add any more code to your character - it already knows what to do when it is hit!
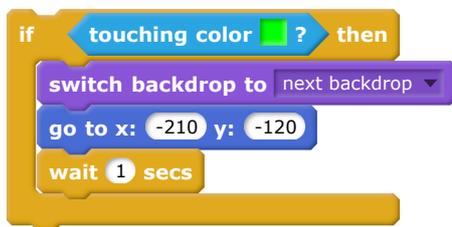
- Test out your game to see if you can get past the laser. Change the `wait` times in your code if the lasers are too easy or too hard.

Step 9   Challenge: More obstacles

If you think your game is still too easy, you can add more obstacles to your level. You can add anything you like, but here are some ideas:

- A flying killer butterfly;
- Platforms that appear and disappear;
- Falling tennis balls that must be avoided.

You could even create more than one backdrop, and move to the next level when your character reaches the green door:

```
if     touching color ⬜ ? then
  switch backdrop to  next backdrop ▼
  go to x: -210 y: -120
  wait 1 secs
```

Step 10   Challenge: Improved gravity

There's one other small bug in your game: gravity doesn't pull your character downwards if *any* part of it is touching a blue platform – even its head! You can test this out by climbing most of the way up a ladder and then moving to the left.

Can you fix this bug? To do this, you need to give your character different coloured trousers (on *all* costumes)…

…and then replace the code:

```
touching color 🟦 ?
```

with:

Remember to test your improvements to make sure you've fixed the bug!

Step 11   Challenge: More lives

Can you give your player 3 **lives**, instead of just sending them back to the beginning each time? Here's how your game could work:

- Your player starts with 3 lives;
- Whenever your player gets hit, one life is lost and they go back to the start;
- If there are no lives left, the game ends.