## Getting started

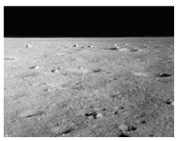
### Open Scratch 3.0

You can do this online here: https://scratch.mit.edu/ or use a local version.

This project only uses resources available in Scratch, so there's no need for students to sign in. They will use the following sprites and backgrounds:

Dot

Robot



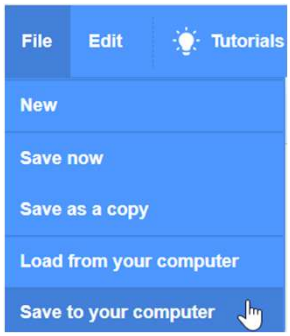
Moon

There is also a video available to show you how to get started with this activity.

### Saving projects

Use this to save projects to the computer if you are not logged in to Scratch (recommended):

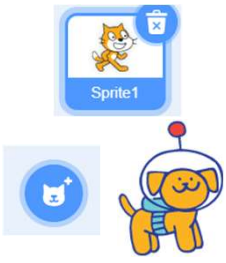
In the menu click File > Save to your computer



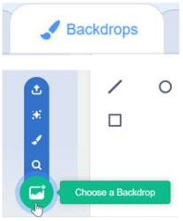



## Set the scene

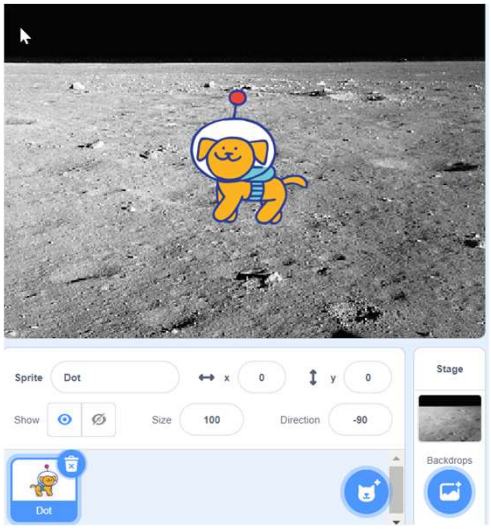
1. **Remove the cat sprite**
   Felix the Scratch cat sprite doesn't want to go to space today, so let's replace them with Dot the space dog! Click on the bin icon, top right of the cat, to delete 'Sprite1'. Click on the 'Choose a Sprite' button and use the search to find and add 'Dot' instead.




2. **Change the backdrop**
   In the Stage area (bottom right) click on the 'Choose a Backdrop' icon and use the search to find the 'moon'.




3. **Now it's time to begin your Coding Steps**



## Challenge 1 – Moon Walk

**Can you teach Dot the space dog to walk on the moon?**

### Coding Step 1

Drag the **move 10 steps** block from the **Motion** block menu into the empty space.

Add **switch costume to dot-b** from **Looks** menu so it clicks.

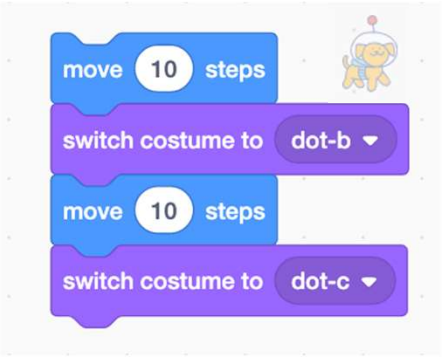
Add another **move 10 steps** block and **switch costume to dot-c** (you can click on the down arrow to change from dot-a)

Note: 'Move 10 steps' will make Dot move 10 pixels across the screen. 'Switch costume' is what will create the illusion of Dot walking. Also, you can put things back on the block menu to delete. Watch the video if you'd like to see how to start…

### Teaching point:

Test by clicking on the sequence of blocks. Notice how Dot moves but you cannot see the costume is changing. This is because it's so fast it can't be seen – see next slide.

### Code solution:




3

## Challenge 1 - Debug

### Coding Step 2

We need to use a **wait** control block to slow the process down so we can see the costume change.

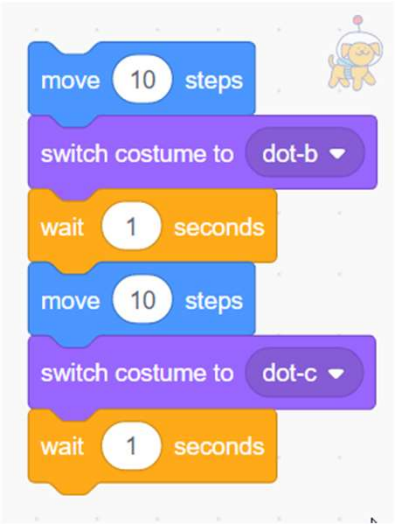
Add in **wait 1 second** after the costume changes and test again by clicking on the sequence of blocks.

Note: You need to drag the block down from the sequence if you want to break it apart to remove a block – make sure you move it far enough so they don't clip back together again!

Did the algorithm work for this test run? If so, well done! You have made Dot walk on the moon!

On the next slide, let's see how we can make her turn around when she gets to the edge of the screen.

### Code solution:




4

## Challenge 1 – Event blocks and repetition

### Coding Step 3

Set how the algorithm is triggered and make Dot walk back and forth across the screen forever!

Add a forever loop from the **Control** block menu around the entire sequence. This is an example of repetition.

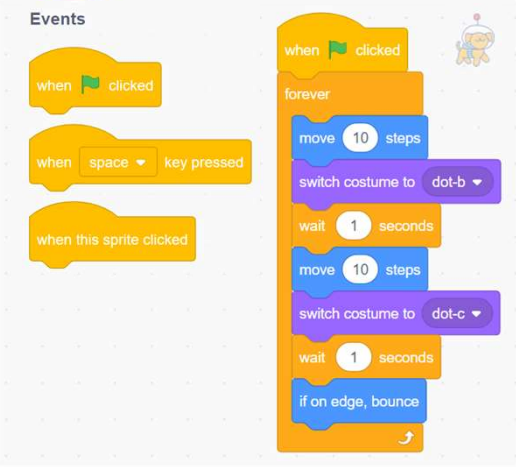
Add **if on edge, bounce** from the **Motion** block menu to the bottom of this sequence but inside the forever loop.

Choose an event block from the **Event** block menu to trigger getting this algorithm to run. This could be **when green flag clicked**, **when this sprite clicked** or **when space key pressed**. Note: pressing on a keyboard key or using the mouse is an example of input into your algorithm.

### Teaching point:

Test by clicking/pressing on your chosen trigger action to start the program (eg clicking the green flag). Notice how the dog sprite turns upside down when it touches the side of the screen. Let's debug this in the next step.

### Code solution:




5

## Challenge 1 - Add in more event blocks

The 'rotation style' needs to be set at the start so the sprite will only turn left and right (not upside down), and the sprite needs to be reset at the beginning of each test run.

### Coding Step 4

Add in further blocks from the **Motion** block menu and **Looks** block menu under a new **when green flag clicked** event block. Dot should always:

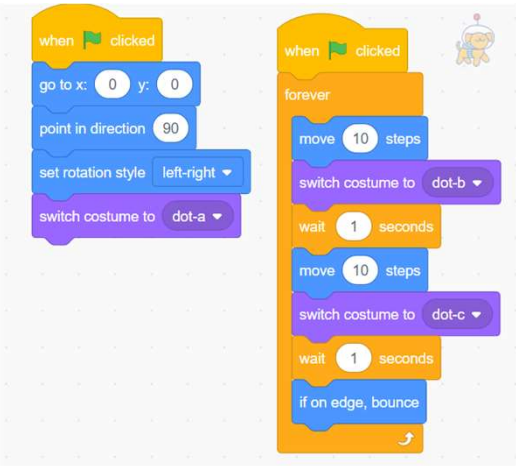
- return to the same start position of 0,0, (the middle)
- point in the right direction
- only move left and right
- start with the first costume dot-a

### Teaching Point:

Note it is possible to use an event block more than once to organise the code on the screen. Here the second green flag sequence resets things - this is often called *initialisation.*

Test the algorithm by clicking the green flag.

### Code solution:




6

## Challenge 1 - Investigate

Now let's try changing some things to see what happens!

### Investigate

- What happens if you increase/decrease the number of steps Dot takes?
- What happens if you change 1 second to 0.5 second or 2 seconds?
- What happens if you change the coordinates to go to x: -100 y: 0 ?
- What about go to x: 0 y: 100 ?

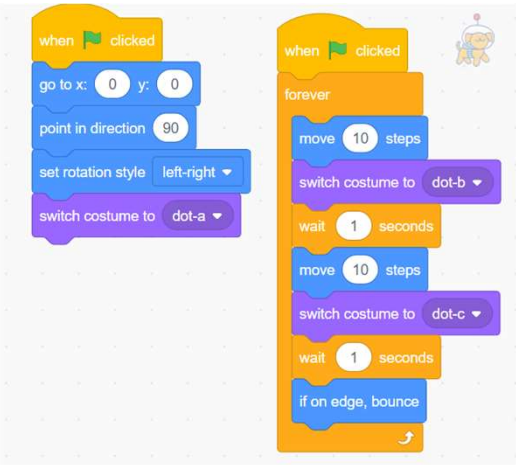
Can you explain how the coordinates work?

### Teaching Point:

Use the Xy-grid backdrop to help show how the coordinates work in Scratch. x:0 y:0 will always come back to the middle.

**Note: If you are planning to save your work (file > save to your computer), do it now before resetting the project for the next Challenge.**

### Code solution:






## Challenge 2 - Conversation

### Coding Step 1

Select the Dot sprite.

In the **code** tab, add **when green flag clicked** from the **Events** block menu.

Add **say Hello! for 2 seconds** from the **Looks** menu.

Perhaps you might like Dot to say something different? Type over Hello! to add in your own words.

Add **wait 2 seconds** block from **Control** menu.

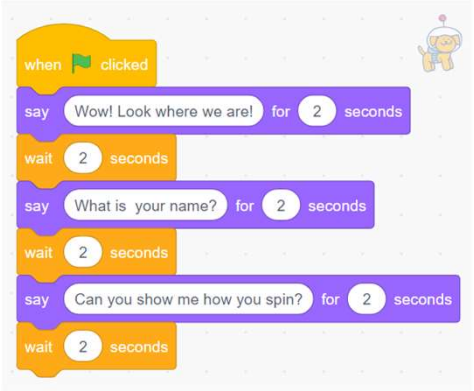
Right click on the **say** block and duplicate the block sequence to add in 2 further parts of conversation.

### Teaching point:

Note that the **wait** block will be when the Robot will reply.

It may be helpful to think of questions that Dot asks so the Robot can then answer them.

### Code solution:

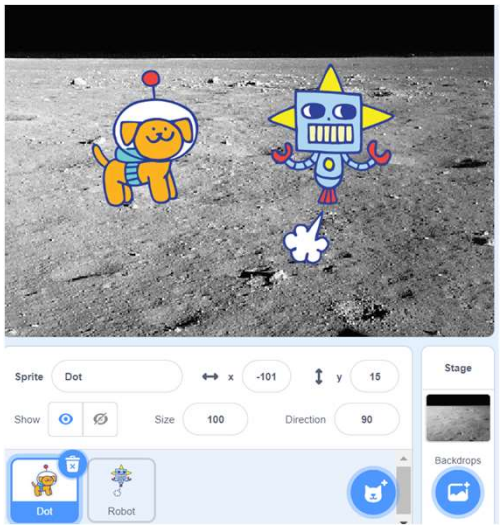


## Challenge 2 - Conversation

**Now we're on to Challenge 2.**

**Dot has bumped into a friendly Robot on the moon! Can you make them talk to each other?**

1. Click 'file' and then 'new' at the top of your screen. This will reset the project.
2. Follow the instructions on slide 5 ('Set the scene') to add the 'Dot' sprite and the 'moon' background again. This time, you will also need to search for and add the 'Robot' sprite.
3. You are now ready to begin your **Coding Steps**.






## Challenge 2 - Make the Robot reply

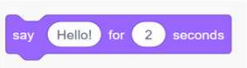
### Coding Step 2

Select the Robot sprite.

In the **code** tab, add **when green flag clicked** from the **Events** block menu.

Add **wait 2 seconds** block from the **Control** menu

Add **say Hello! for 2 seconds** from the **Looks** menu.




What should the Robot say in reply to Dot? Type over Hello! to add in your own words.

Right click on the **wait** block and duplicate the sequence to add in 2 further parts of conversation.

### Teaching Point:

Make sure that you have put a **wait** block first so that Dot and the Robot don't speak at the same time! The timing makes them take it in turns to talk.

Test the algorithm by clicking the green flag.

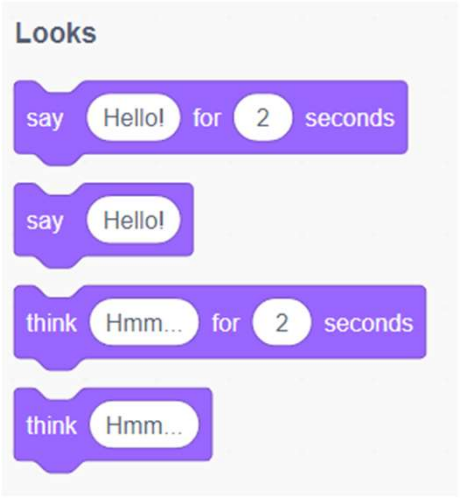
### Code solution:

## Challenge 2 - Investigate

Now have a play to see what difference it makes when you change things. Here are some ideas:

- Experiment with replacing **say Hello! for 2 seconds** with the **say Hello!** block which does not have a timer control. What does that do to the conversation?
- Explore using the **think** blocks instead of **say**. What do you notice happens on the screen?
- Extend the conversation further.
- Change the Event that triggers the sequence.
- If you have used Scratch before and know about **broadcast** message and **when I receive** message Event blocks, you could try using them instead of **wait.**

**If you are saving your work (file > save to your computer), do so now before you move onto the next Challenge. You will find these files as .sb3 files, usually called something like 'Scratch Project(1).sb3' in downloads.**



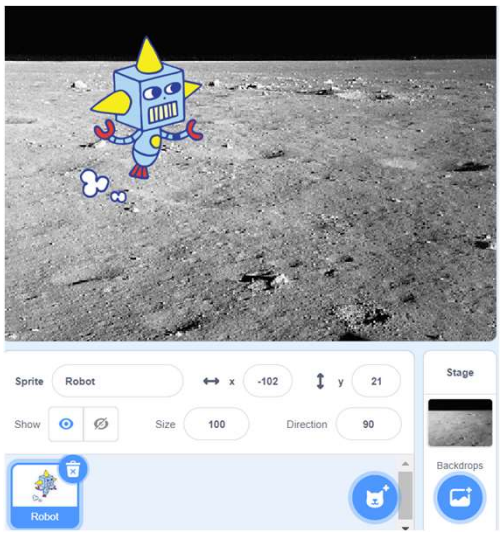



## Challenge 3 – Robot spin!

**Now we're on to Challenge 3.**

**After having a lovely chat with Dot the moon dog, our Robot wants to celebrate meeting a new friend. Can you create a project to make them spin?**

1. Click 'file' and then 'new' at the top of your screen. This will reset the project.
2. As before, add the moon background. For this challenge, you only need to add the 'Robot' sprite.
3. You are now ready to begin your **Coding Steps**.



# Challenge 3 – Make the Robot move and spin

## Coding Step 1

To make the Robot spin right using the right arrow, add the **when space key pressed** block from the **Events** block menu. Change **space key** to **right arrow** using the dropdown.

Add **change x by 10** from **Motion** menu.

Add **next costume** block from **Looks** menu

Test the sequence of blocks. Remember pressing on a keyboard key or using the mouse is an input to your algorithm. Does it look like the Robot is spinning round and round?

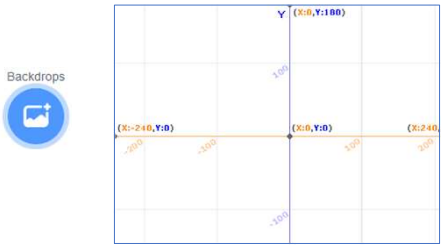
Now, right click and duplicate the code, change the sequence to be the left arrow and change x by –10.

## Teaching Point:

Remember that the x axis runs from left to right across the screen. 0,0 is in the middle so anything to the left is negative and anything to the right is positive.

(Use the Xy-grid backdrop to help show this).

## Code solution:




# Challenge 3 - Investigate

Our Robot can now be controlled with the arrow keys. What happens if we change some things?

- What happens if you increase the number of steps Robot moves?
- What happens if you decrease the number of steps Robot moves?
- What happens if you change the order of the costumes?

## Code solution: