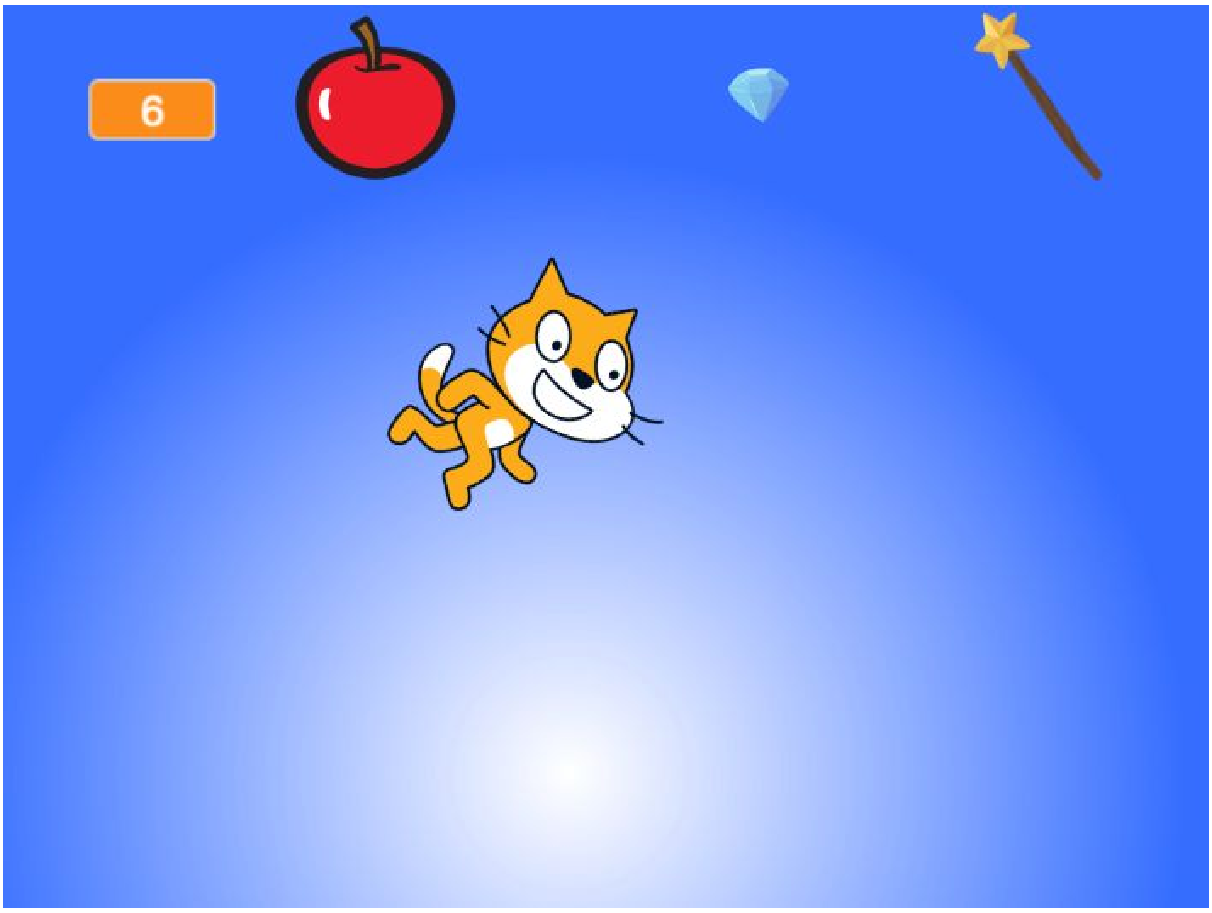
Phoenix



You will need:

Access to the Scratch website, from a computer or laptop.
(It might also work from an ipad or tablet)

This tutorial is mainly for children and people who have not used Scratch much before. If you get stuck, let us know in the comments section for the example project.

Setup:

If you haven't used Scratch before, click join scratch at the top and create an account. You might need your parent's email to do this, so ask for help. If you can't make an account, you can follow along, but you won't be able to save your work.

Once you've made an account follow the Phoenix Scratch studio here. This is where you can find the template versions of all the workshops we create. Feel free to remix them and post the new things you create.

Wait, what is Scratch?

Scratch is a free, online, block-based code editor. You might have used it in school. If you've been to one of our family workshops, you've probably seen us using it there.

There are lots of different tutorials available for using Scratch. Here we are going to focus on showing you ways of using Scratch that link to the work we do at Phoenix. This will include things like animation and drawing.

Virtual Pet

This tutorial will teach you how to make an interactive pet, that you can feed and look after. There is quite a lot of code for this one so it's probably easiest to remix our example project before starting from an empty project.

To get started, click **create**, or **remix** this project,

https://scratch.mit.edu/projects/379161910/

Step 1

First of all, let's think about what we want our virtual pet to be able to do.

We want the pet to:

-move around
-react to us

We want to monitor:

-how hungry the pet is

We want to be able to:

-feed the pet
-clean up any mess the pet makes
-play with the pet

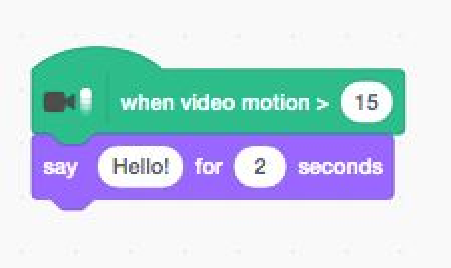
Step 2

This project uses a webcam. If your device doesn't have a webcam you can skip this step. Anywhere that we use the video sensing block you can use when this sprite clicked instead.

If you are starting from an empty project, go to add extension and choose video sensing.

Step 3

Now we can test out our first interaction. We will be creating interactions where our movement on the webcam makes something happen to the sprite we are touching.



```
when video motion > 15
say Hello! for 2 seconds
```

With the Scratch Cat selected, try to copy these blocks as shown in the image. You can find each of these blocks in the menu on the left. Blocks are split up by category. You'll find most of the ones for this tutorial in, Events, Control, Motion and Video Sensing.

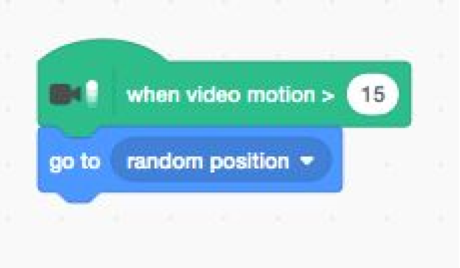
Click the Green Flag to run the program and then click on the button to see what happens.

Hint

The blocks are stored in different sections. When you are looking for a block we have used, match the colour of the block to the circle on the left side. For example, movement related blocks are always blue.

Step 4

So that is how we will start to build our interactions. Let's try another one. Disconnect your Looks block and try this Motion one instead.



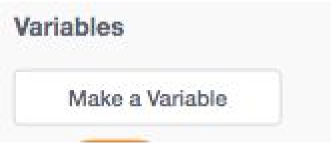
```
when video motion > 15
go to random position
```

Make sure you disconnect this block before moving on to the next step. We put ours back to how it was in Step 3.

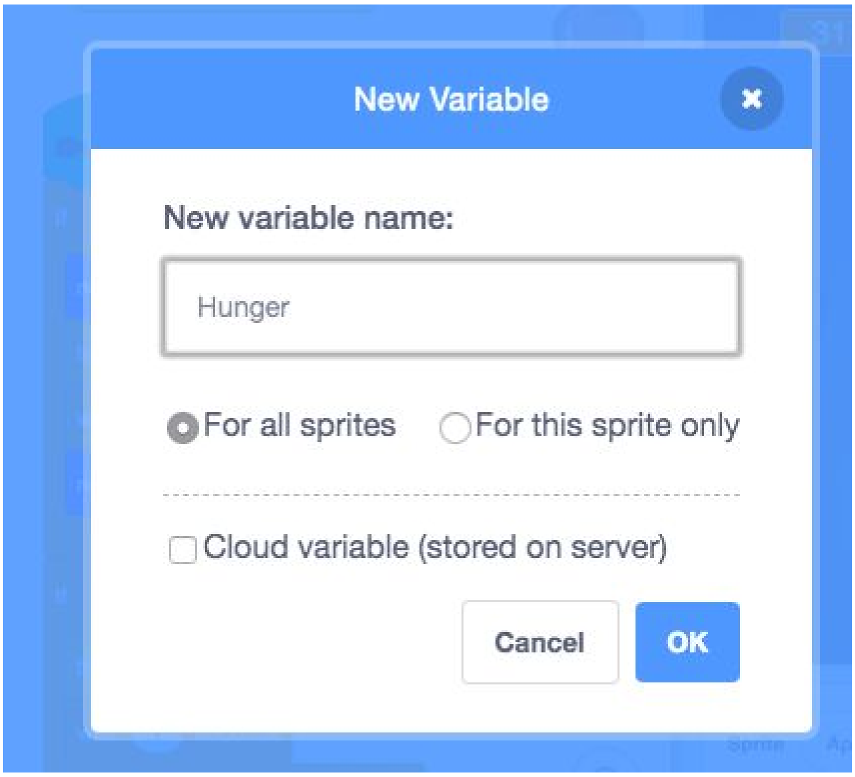
Step 5

Now let’s think about how we can check if the pet is hungry.

We can do this by creating a variable. To do this choose Variables and then click the Make a Variable button.



Fill in a name for the new variable.



New Variable

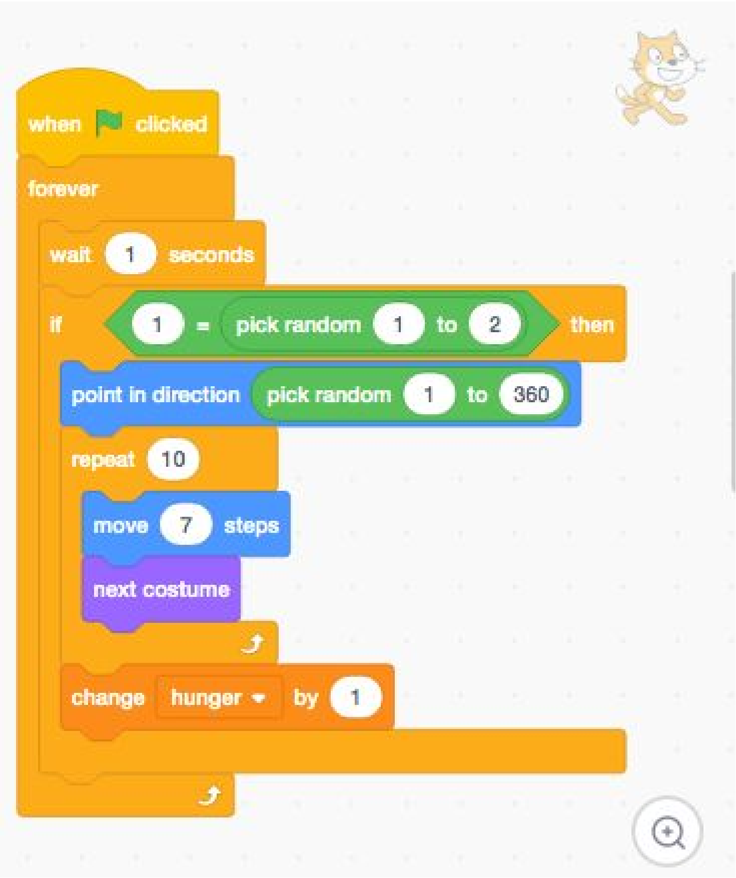
New variable name: Hunger

● For all sprites ○ For this sprite only

□ Cloud variable (stored on server)

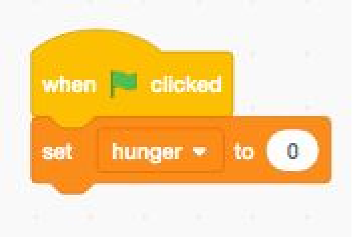
Step 6

Now let’s put that variable to use. Try to assemble the code below.



```
when ⚑ clicked
forever
    wait 1 seconds
    if 1 = pick random 1 to 2 then
        point in direction pick random 1 to 360
        repeat 10
            move 7 steps
            next costume
        change hunger ▾ by 1
```

We’ll also need to make this block, to make sure the variable starts in the right place each time.



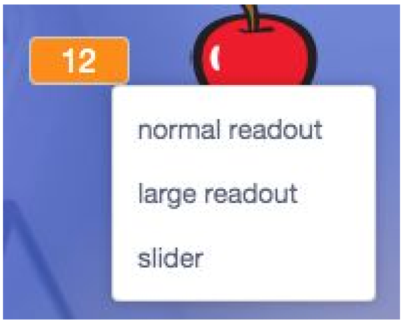
```
when ⚑ clicked
set hunger ▾ to 0
```

So now, when we press the green flag, the pet will make a decision, whether or not to move. We’ve put that in the pick random block that we will be using a lot. At the moment it is set to 1 to 2, so half the time, the pet will choose to move. You can come back to this later and adjust it however you like.

When the pet moves, it takes a few steps, changes costume using next costume and repeats that. This makes the pet appear animated.

Finally, when the pet has finished moving, it registers that it is now hungrier than it was before it moved. So now the hunger is increased using change hunger by 1.

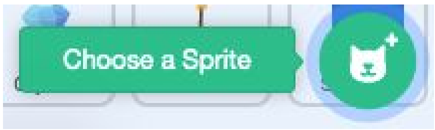
The hunger variable should be visible in the stage area. You can choose how it looks by right clicking on it. To begin with the normal readout is probably easiest. But once you have everything in place the large readout is clearer.



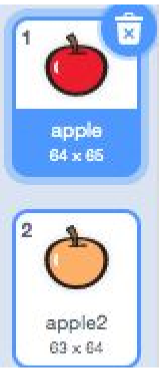
Step 7

Ok, so now we have a pet that is getting increasingly hungry.
Let's make some food to feed it with.

Click on 'choose a sprite' and then pick some food. We've chosen an apple. Set the position of the food to x -90, y 140.

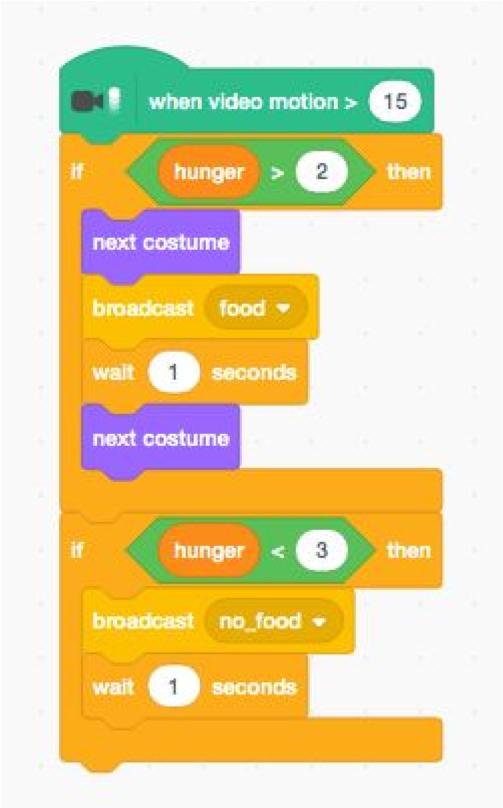


Now select the food sprite.
First of all open up the costume tab. Using the drawing tools, make a second costume for the food.



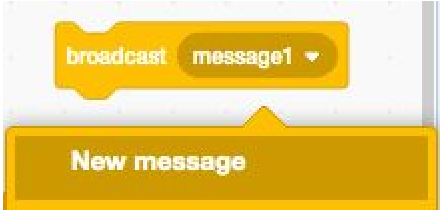
It can be anything really. To make this apple we duplicated the first one and the used the 'fill' tool, to just change the colour.

Next open up the code. We want to build some code that will allow us to press the apple whenever we want to feed the pet. Try to copy the code below.



```
when video motion > 15
if hunger > 2 then
    next costume
    broadcast food ▾
    wait 1 seconds
    next costume
if hunger < 3 then
    broadcast no_food ▾
    wait 1 seconds
```

There is a new code block in here. Broadcast message1. You have to click on the dropdown option next to message1 and select new message, to modify this.



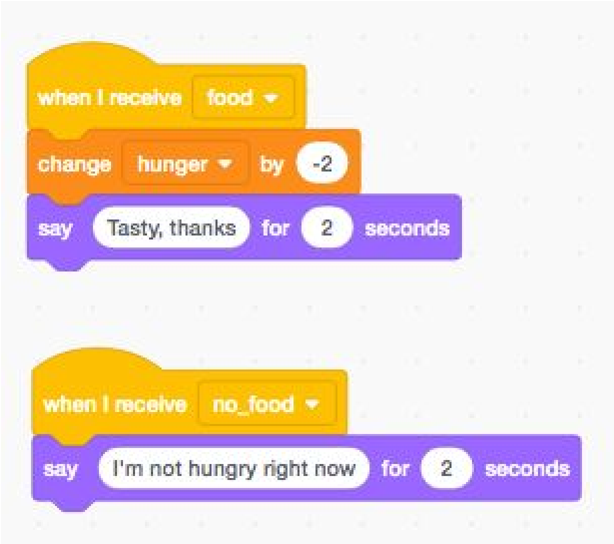
```
broadcast message1 ▾
```

We have made two messages, ‘food’ and no_food’.

So if you test the code now you should already be able to see that the food is starting to work!

Step 8

To make the pet more interesting though we are going to use the messages we set up, to make the pet respond. It will say something to us, and also we will make sure that the hunger variable decreases.



```
when I receive food
change hunger by -2
say Tasty, thanks for 2 seconds

when I receive no_food
say I'm not hungry right now for 2 seconds
```

Step 9

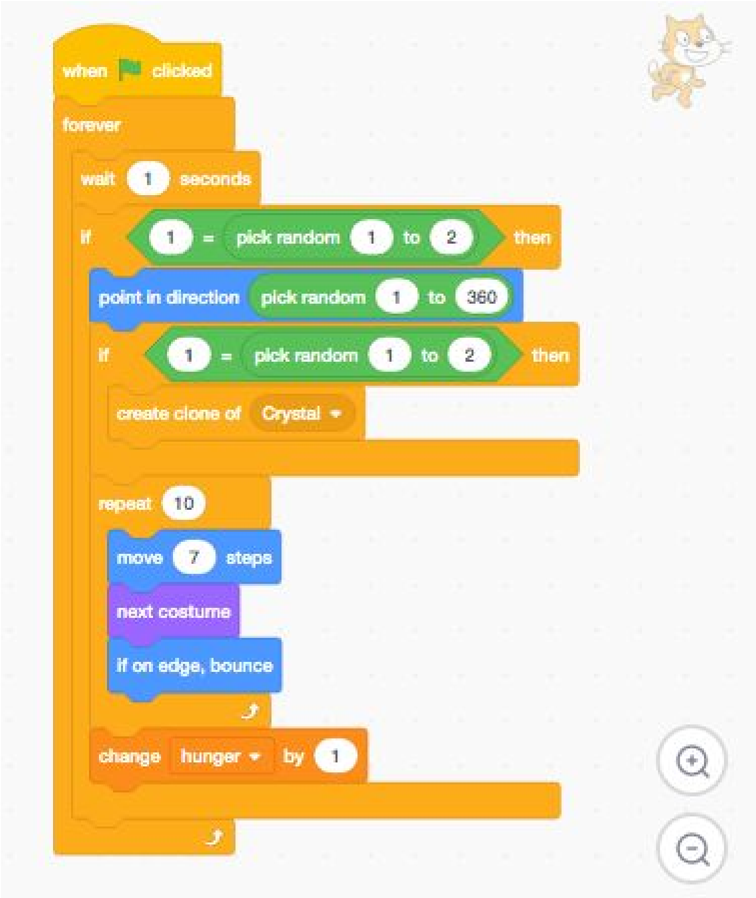
So now we have our first interaction sorted, we can start experimenting with others. If you have had any real pets, there is one thing you know they all do. They poop.

And that needs cleaning up.

So how can we build this?

First of all, choose a new sprite. Our pet is quite special so we've chosen a crystal. Place the crystal at x 60, y 140.

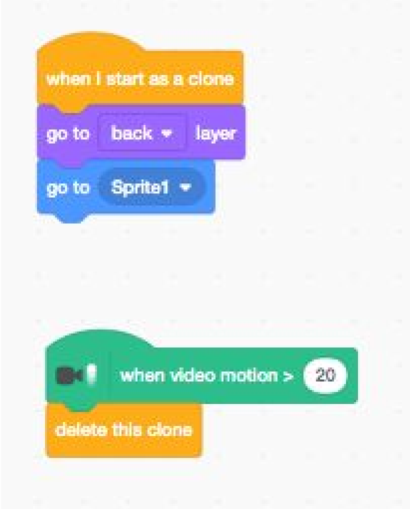
Now try changing your pet code to look like this.



```
when ⚑ clicked
forever
    wait 1 seconds
    if <1 = pick random 1 to 2> then
        point in direction pick random 1 to 360
        if <1 = pick random 1 to 2> then
            create clone of Crystal ▾
        repeat 10
            move 7 steps
            next costume
            if on edge, bounce
        change hunger ▾ by 1
```

We have added an extra decision. Half of the time, when the pet moves it will also do a poop. It does this by creating a clone of the Crystal sprite. Creating a clone allows us to make lots of copies of the sprite if we need and also get rid of them when we need to.

Back in the Crystal code, try to make the following.



```
when I start as a clone
go to back ▾ layer
go to Sprite1 ▾

when video motion > 20
delete this clone
```

So, this code makes the clone of the Crystal appear underneath the pet, and also delete itself if we wave over it.

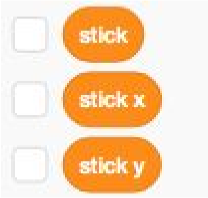
Click the green flag and check everything is working as you expect it to.

Step 9

Excellent. We now have most of our aims sorted out. To complete the next one we're going to put everything that we've used so far together. We're going to make a stick that we can throw for the pet to fetch.

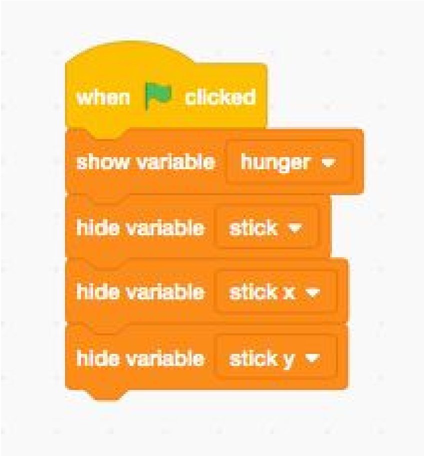
Make a new sprite. We've chosen the wand, which looks a bit like a stick.

Now make three new variables.



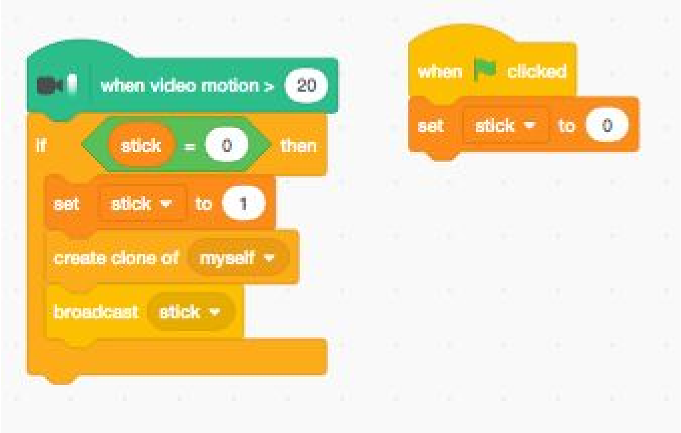
□ stick
□ stick x
□ stick y

By default these will now appear on your stage. This is useful whilst we are testing, but when you are ready you can add in a block of code like this to hide them.



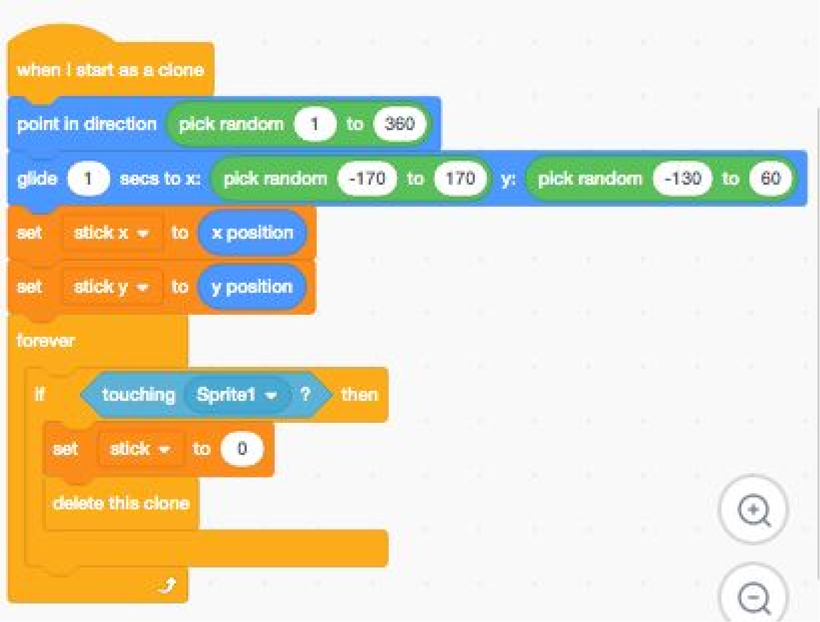
```
when 🏴 clicked
show variable hunger ▾
hide variable stick ▾
hide variable stick x ▾
hide variable stick y ▾
```

Go back to your stick sprite and create the following code.



```
when video motion > 20
if stick = 0 then
    set stick ▾ to 1
    create clone of myself ▾
    broadcast stick ▾
```

```
when 🏴 clicked
set stick ▾ to 0
```



```
when I start as a clone
point in direction pick random 1 to 360
glide 1 secs to x: pick random -170 to 170 y: pick random -130 to 60
set stick x to x position
set stick y to y position
forever
    if touching Sprite1 ? then
        set stick to 0
        delete this clone
```

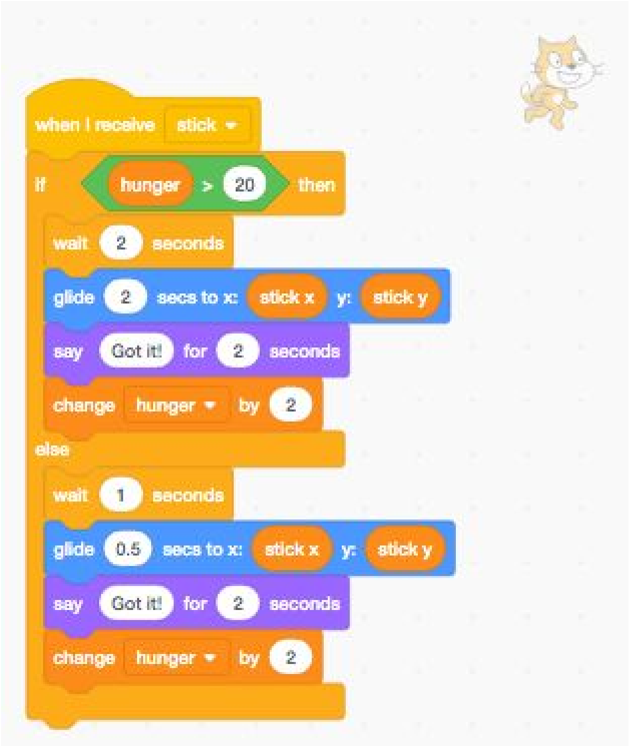
There is quite a bit going on here.

When we wave over the stick we make a clone of it, but only if there isn’t already a stick waiting to be fetched. We also broadcast a new message and set our stick variable to show that there is now a stick.

When the stick is created as a clone, we point it in a random direction and then make it move to a random position within the stage. We then set the new variables to record the location of that stick.

Finally, as soon as the clone is touched by the pet, it is deleted and we reset our stick variable.

Now we are halfway there. But we need to make our pet fetch the stick. To do this we will use the new message we created.



```
when I receive stick ▾
if hunger > 20 then
    wait 2 seconds
    glide 2 secs to x: stick x y: stick y
    say Got it! for 2 seconds
    change hunger ▾ by 2
else
    wait 1 seconds
    glide 0.5 secs to x: stick x y: stick y
    say Got it! for 2 seconds
    change hunger ▾ by 2
```

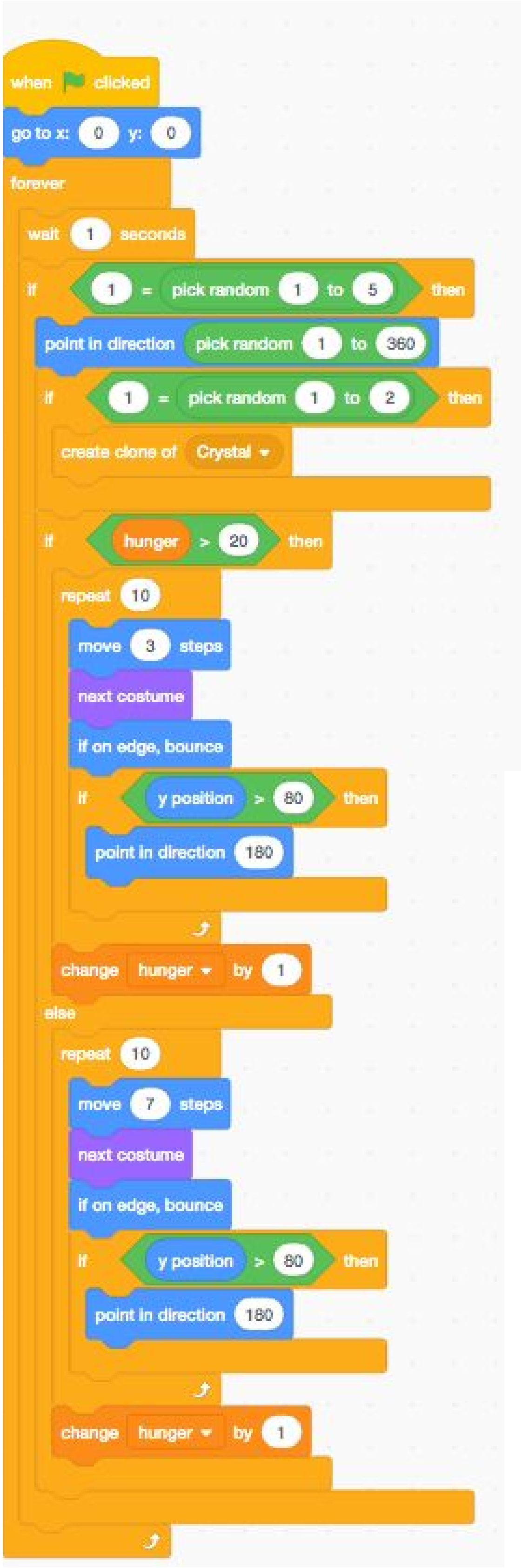
So now we have a method for making our pet find the stick.

You'll notice that it is also checking to see how hungry the pet is first. If it is too hungry it will move more slowly.

See if you can add a hunger check in to the main movement of the pet. We'll show you how we did it in the next step.

Step 10

So now we can feed the pet, clean up and play fetch. Let's have a final look at our pet movement code to see how it ended up.



```
when ⚑ clicked
go to x: 0 y: 0
forever
    wait 1 seconds
    if 1 = pick random 1 to 5 then
        point in direction pick random 1 to 360
        if 1 = pick random 1 to 2 then
            create clone of Crystal ▾
        if hunger > 20 then
            repeat 10
                move 3 steps
                next costume
                if on edge, bounce
                if y position > 80 then
                    point in direction 180
            change hunger ▾ by 1
        else
            repeat 10
                move 7 steps
                next costume
                if on edge, bounce
                if y position > 80 then
                    point in direction 180
            change hunger ▾ by 1
```

Extra steps

Wow, congratulations on making your own virtual pet. If you made it this far, you can probably handle some extra steps.

- In the example project there is a hidden dance party mode. See if you can connect it up.
- What about adding a happiness variable and designing an interaction for that?
- Can you add some more sounds?
- Pets often sleep quite a lot...

Remember to share your project to the Phoenix Scratch studio!

Phoenix.org.uk
@PhoenixLeic