Virtual Dog Program in Scratch

By Phil Bagge

@baggiepr

code-it.co.uk

How to use this planning

- Confident children could work independently through the instructions

- You could use the step by step guide to teach this to less independent children

- Show pupils a working model and get them to decompose what is happening before creation

- This plan shows one way of achieving this program your pupils may come up with better or different code, embrace this if it happens

Previous Knowledge

- Basic experimentation with Scratch is needed as pupils need to know how to attach and break blocks apart and understand how code links to a sprite or background

- Loops in real life
http://code-it.co.uk/resources/loops.pdf

- I would recommend creating a simpler program such as my Spider Maze
http://code-it.co.uk/year4/scratchmindmap.html
or Basic Quiz first
http://code-it.co.uk/year4/mathsquiz.pdf
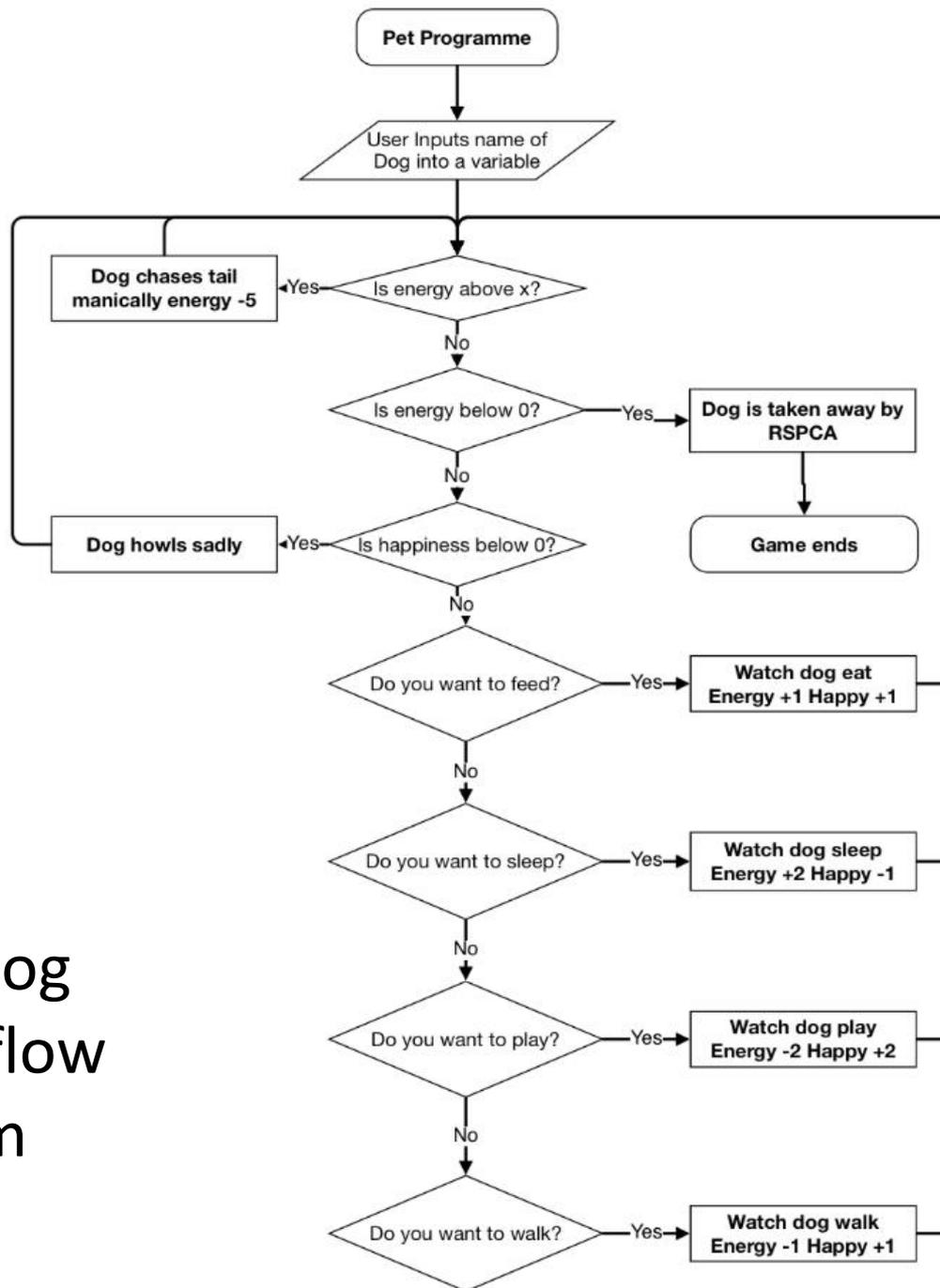
Programming Concepts

- Forever iteration loops
- Repeat n times iteration loops
- Forever if iteration loops
- Random numbers with max and minimum
- Starting blocks of code from within other blocks of code by broadcasting
- Less and more than > <
- Using variables to store numbers or words
- Changing a variable
- Using if not to control an event

Virtual Dog Plan

The program should

- Ask the player to name the dog

- Use the dogs name within the program by storing it within a variable

- Present the player with choices that the dog can do (eat, sleep, play, walk)

- Perform an short action when these choices are chosen

- Increase or decrease happiness or energy depending on which choices were chosen

- Grow the dog regularly as the game continues and report the pets age every time it gets a day older

- Perform involuntary actions if energy gets too much or happiness gets too low

- End the game if energy goes below 0

By Phil Bagge

@baggiepr

code-it.co.uk

Pet Programme

User Inputs name of Dog into a variable

Dog chases tail manically energy -5 ◄Yes— Is energy above x?

No

Is energy below 0? —Yes→ Dog is taken away by RSPCA

No

Dog howls sadly ◄Yes— Is happiness below 0?

Game ends

No

Do you want to feed? —Yes→ Watch dog eat Energy +1 Happy +1

No

Do you want to sleep? —Yes→ Watch dog sleep Energy +2 Happy -1

No

Do you want to play? —Yes→ Watch dog play Energy -2 Happy +2

No

Do you want to walk? —Yes→ Watch dog walk Energy -1 Happy +1

Virtual Dog Plan as a flow diagram

Download the dog sprites and backgrounds and understand what is available

- Go to http://code-it.co.uk/year4/scratchprojects.html and download the *virtualdog* file to your Documents area

- Start Scratch and using **File** and then **Open** open *virtualdog*

- This contains dog sprites for different actions, a ball sprite for the dog to play with and a variety of backgrounds

- Have a look at these.

- Can you see which sprites are pairs?

Stage 1
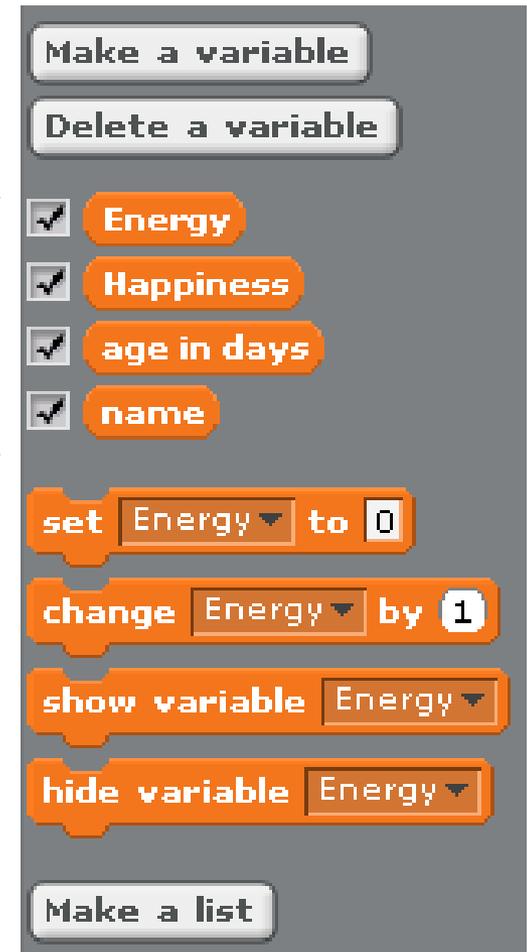# Setting up Variables to use throughout the program and getting the player to name the dog

Create Variables to use throughout the programme

Motion | Control
Looks | Sensing
Sound | Operators
Pen | **Variables**

First click on the variables button

Make a variable

Delete a variable

Secondly click on make a variable and name it **energy**

Thirdly repeat this three more times to make variables called **Happiness**, **age in days** & **name**. Watch your spelling and make sure they all have ticks in the box so they show on the screen

☑ Energy
☑ Happiness
☑ age in days
☑ name

set Energy to 0

change Energy by 1

show variable Energy

hide variable Energy

Make a list

We are going to write code that changes what is inside the variables as the game progresses and some events will be triggered by the amount in a variable

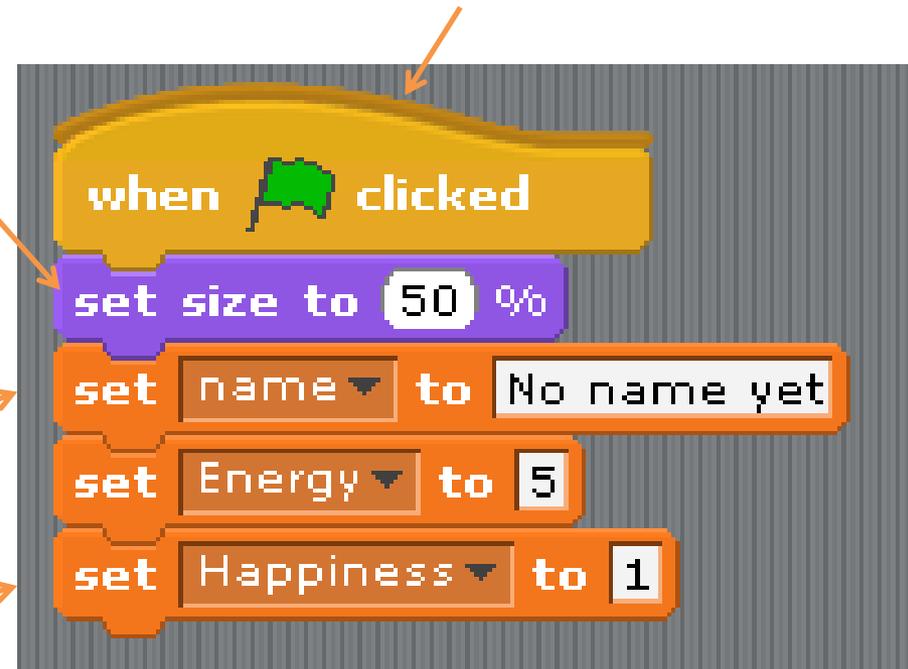**Finally** arrange these on your screen at the top

Set variables
Starting position

First set a flag starting block so the game will start when the green flag is clicked

Second set the size to 50%. The dog sprites are too big at the moment but this will shrink them, 50% is the same as ½ .

Thirdly set the name variable to have No name yet inside it. Later we will ask the player to put the dogs name into it.

when 🏳 clicked
set size to 50 %
set name to No name yet
set Energy to 5
set Happiness to 1

Finally set the dogs energy to 5. You can change this later when you play test. Set happiness to 1. You can change this later when you play test the game.
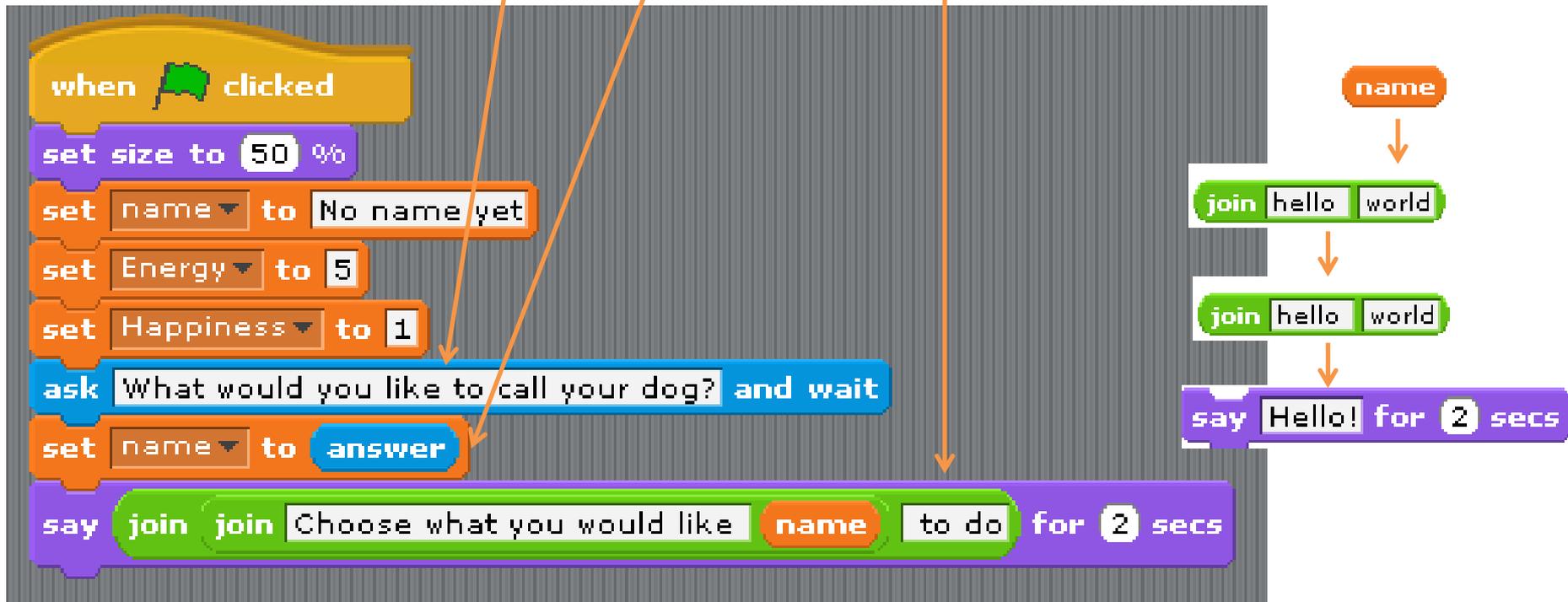
Ask the player to name the dog
The name is then put inside the variable called name

First ask the question and wait for the player to type an answer

Second put the answer the player types inside the name variable.

Finally use the name variable to ask the player what they would like to do.

Save your work and test it. Does the name end up inside the say command?

when 🚩 clicked

set size to 50 %

set name to No name yet

set Energy to 5

set Happiness to 1

ask What would you like to call your dog? and wait

set name to answer

say join join Choose what you would like name to do for 2 secs

name

join hello world

join hello world

say Hello! for 2 secs

```
set name to No name yet
set Energy to 5
set Happiness to 1
ask What would you like to call your dog? and wait
set name to answer
say join join Choose what you would like name to do for 2 secs
broadcast decision
```

Click on the black triangle and select new. Type in **decision** this block will start any other blocks which start when they receive the decision broadcast.

Start Virtual Dog Control Panel using a broadcast command

Stage 2
Create actions for the virtual dog

Adding a sleep action to the pet dog
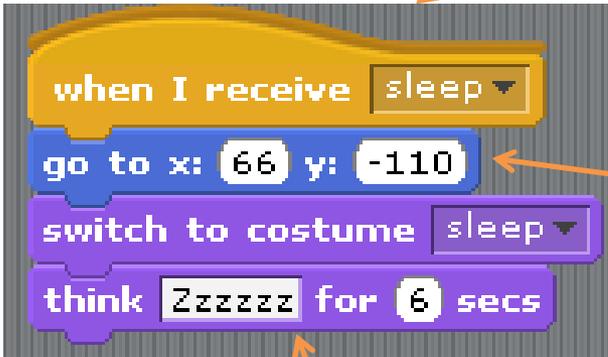
First create this block in the **dog sprite**.

Click on the black triangle and select new. Type in sleep this means this block will only run when it receives sleep broadcast from elsewhere in the programme.

```
when I receive sleep ▼
go to x: 66 y: -110
switch to costume sleep ▼
think Zzzzzz for 6 secs
```

Use these x and y coordinates or find your own. To place the dog at the bottom of the bed on the floor.

Switch to costume sleep which places the dog in its basket

Add some sleepy noises

Finally in the **dog sprite** remove the green flag from the dog naming block and add a sleep broadcast block. This is only a temporary block to test this and other action blocks.

```
when [green flag] clicked
broadcast sleep ▼
```

Second create this block in the **stage**

```
when I receive sleep ▼
switch to background sleep ▼
```

This block switches to a bedroom background picture called sleep

Adding an eat action to the pet dog

First create this block in the **dog sprite**.

Click on the black triangle and select new. Type in eat this means this block will only run when it receives eat broadcast from elsewhere in the programme.
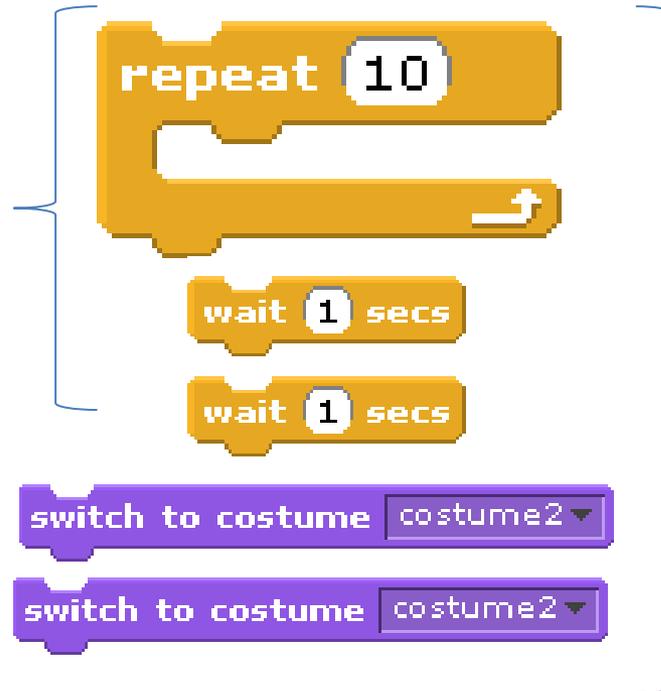
`when I receive eat`

`go to x: 124 y: -125`

Use these x and y coordinates or find your own. To place the dog on the floor in the kitchen.

Pop this block underneath

Change the number of repeats and the wait times so that the block lasts for about 6 seconds

`repeat 10`

`wait 1 secs`

`wait 1 secs`

Second arrange these so that the dog wags its tail for about 6 seconds

Use costumes **eata** and **eatb**

`switch to costume costume2`

`switch to costume costume2`

Adding a kitchen background to the eat action

Next create this block in the **stage**.

when I receive eat
switch to background eat

when 🚩 clicked
broadcast eat

Finally in the **dog sprite** change sleep to eat and test the blocks you have created.

Save your work and test it. Does the dog eat in the kitchen? Is it in the right place?

Adding a dog chases ball play action

First create this block in the **dog sprite**. This swaps costumes making the dog look like it is walking

```
when I receive play
repeat 38
  switch to costume walka
  wait 0.1 secs
  switch to costume walkb
  wait 0.1 secs
```

```
when I receive play
switch to background play
```

Fourthly create this block in the **stage** to change the background

```
when [green flag] clicked
broadcast play
```

Finally change this block in the **dog sprite** to test the play action
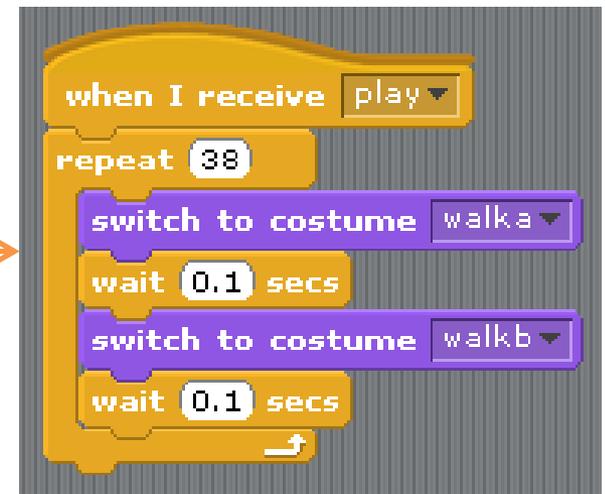
Second in the **dog sprite** create this block. The dog constantly points towards the ball sprite and moves forward.

```
when I receive play
repeat 400
  point towards ball
  move 2 steps
```

```
when I receive play
forever
  if  touching dog ?
    glide 1 secs to x: pick random -200 to 200 y: pick random -200 to 200
```

Thirdly create this block in the **ball sprite**. Forever if the ball touches the dog glide to a random location on the screen.

Adding a walk action

First create this block in the **dog sprite** to make the dog walk back and forth along the bottom of the screen.

```
when I receive walk
go to x: -165 y: -110
point in direction 90
glide 3 secs to x: 138 y: -110
point in direction -90
glide 3 secs to x: -162 y: -110
point in direction 90
```
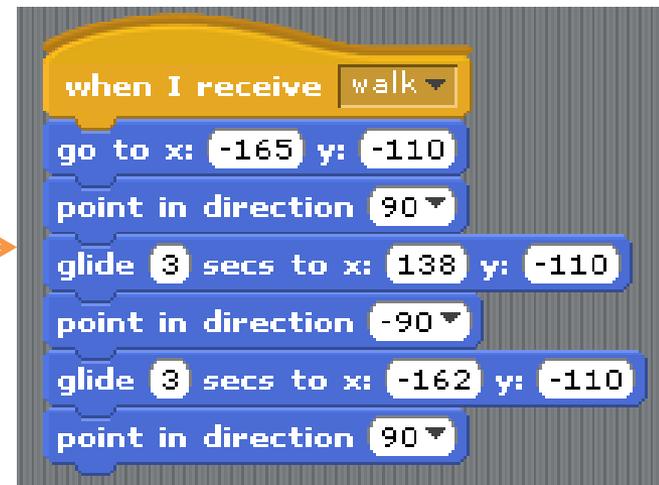
Secondly in the **dog sprite** create this block to change the costumes and make the dog look like it is walking.

```
when I receive walk
repeat 11
  switch to costume walka
  wait 0.25 secs
  switch to costume walkb
  wait 0.25 secs
```
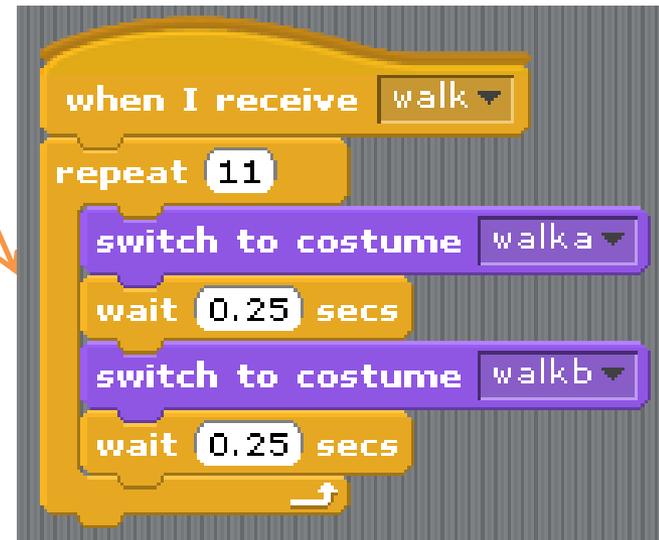
Thirdly in the **stage** create this block to change to a walking background.

```
when I receive walk
switch to background walk
```

```
when [flag] clicked
broadcast walk
```

Finally switch to walk and test the programme

Design a sad dog action

This action will use the **sita** and **sitb** costumes and will feature our dog sitting and howling for between 4-6 seconds. You could make this in one or more blocks all of which are started by a broadcast **sad** block. These blocks need to be created in the **dog sprite**.

switch to costume costume2 ▼

wait 1 secs

glide 1 secs to x: 0 y: 0

repeat 10

say Hello! for 2 secs

when I receive ▼

point towards ▼

Use these and any other blocks you want to design your action to make the dog look sad

Test your work when it is finished although this block will eventually be started when happiness goes below 0

when 🚩 clicked

broadcast sad ▼

Design a mad dog action

This action will use the **walka** and **walkb** costumes and will feature our dog chasing its tail for between 4-6 seconds. You could make this in one or more blocks all of which are started by a broadcast **mad** block. These blocks need to be created in the **dog sprite**.

wait 1 secs

switch to costume costume2

repeat 10

glide 1 secs to x: 0 y: 0

say Hello! for 2 secs

when I receive ▼

point towards ▼

Use these and any other blocks you want to design your action(s) to make the dog look mad

Test your work when it is finished although eventually this block will be started when energy goes over a certain level

when 🏳 clicked

broadcast mad ▼

Create an RSPCA action
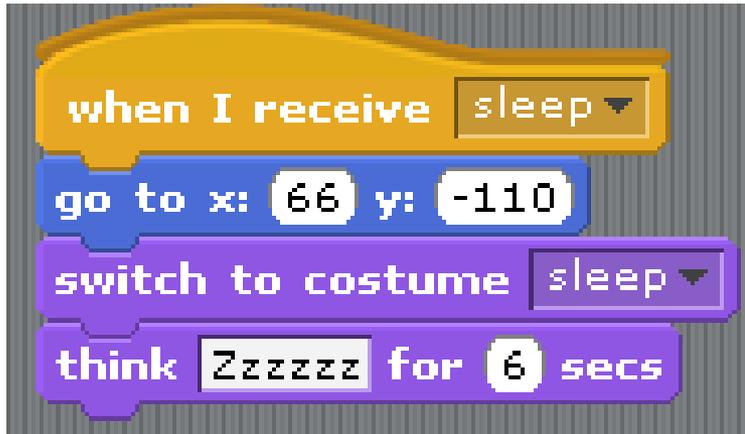
1. Find the sleep block you created earlier .
2. Right mouse click on it until a menu comes up.
3. Select duplicate which will make a new block.
4. Remove the think block at the bottom. Change the block to start when it receives rspca.
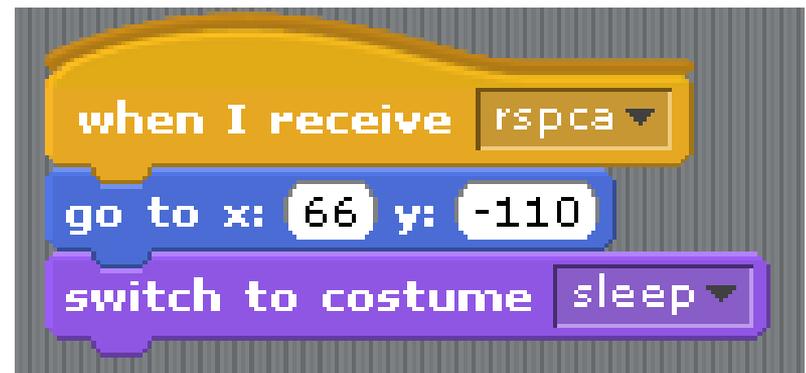
Create a choose action

Firstly in the **dog sprite** create this block. This will position the dog in the centre and change to the ponder costume where he/she looks like they are thinking.
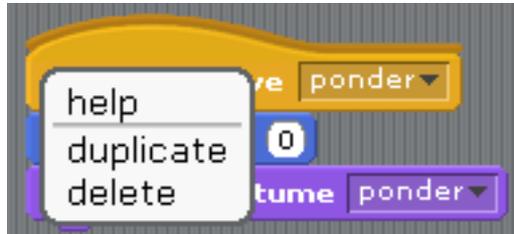
Secondly in the **stage** create this block. This will change the background to the grassy field.

Finally in the **dog sprite** change this block and use it to test if this block works. Once you have finished testing **delete** this block.

Stage 3
Create the Virtual Dog Control Panel

If energy levels drop below 0 then end the game

Create this block in the **dog sprite**. You will need this green operator block that uses a less than symbol. If amount inside the variable called energy is less than 0 then it broadcasts rspca which starts that block. Then it explains that the dog has been taken away before stopping the program.


```
when I receive decision
if     Energy < 0
    broadcast rspca
    say join name has been taken away by the RSPCA for 2 secs
    say as you didn't look after them well enough! for 2 secs
    stop all
```

The following blocks all attach under this one

Check to see if energy is greater than 15 and trigger mad action and reduce energy by 5 if it is
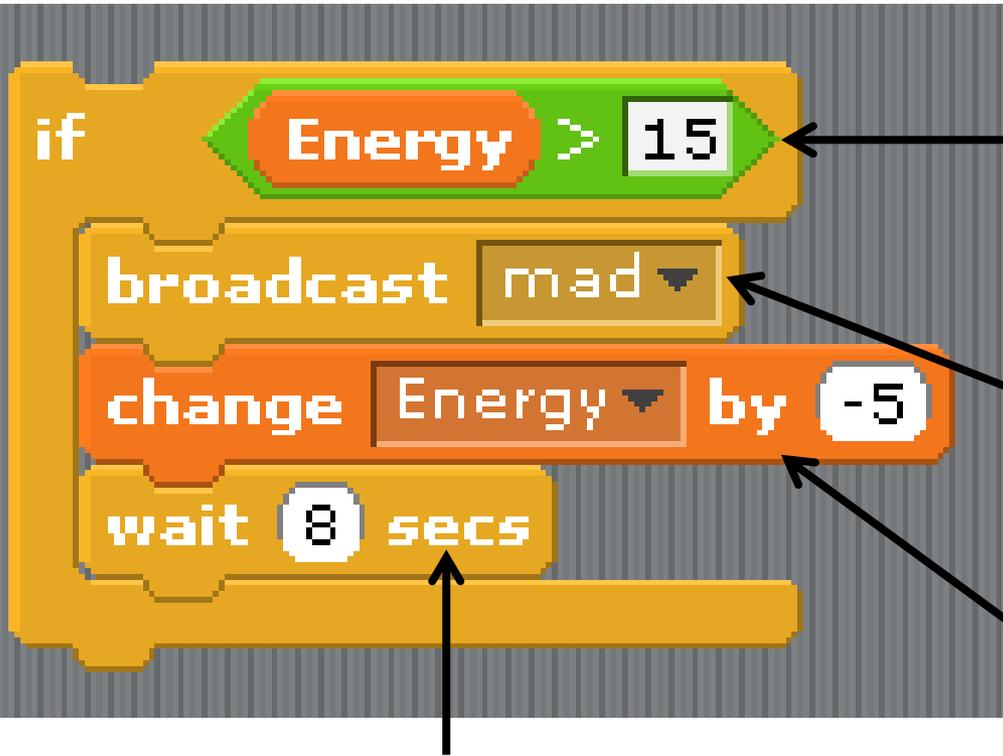
If energy variable is greater than 15

Trigger mad block

Reduce energy by -5

Wait while mad action runs (you may need to adjust this time interval depending on how long your mad action runs for)

NOTE
If the energy variable is less than 15 this block will be passed over and none of the code will be run

Check to see if happiness variable is less than 0 and trigger sad action if it is
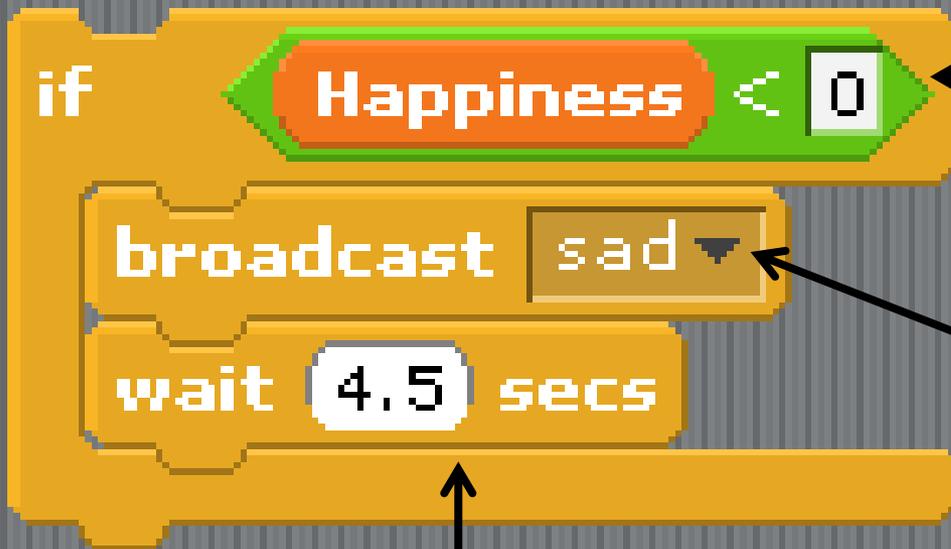
If happiness variable is less than 0

Trigger sad block

Wait while sad action runs (you may need to adjust this time interval depending on how long your mad action runs for)

NOTE
If the happiness variable is greater than 0 this block will be passed over and none of the code will be run

```
broadcast ponder▼
ask  Type f to feed, s to sleep, p to play or w to walk  and wait
if      answer = s
    broadcast sleep▼
    change Energy▼ by 2
    change Happiness▼ by -1

if      answer = f
    broadcast eat▼
    change Energy▼ by 3

if      answer = p
    broadcast play▼
    change Energy▼ by -1
    change Happiness▼ by 3

if      answer = w
    broadcast walk▼
    change Energy▼ by -1
    change Happiness▼ by 3
```

Ponder switches to the ponder block where the dog looks like he/she is making up his/her mind.

These blocks of code ask the user to make a choice which then goes into answer. The program then checks to see which letter is inside answer to determine which block to run

Once you have played the game a few times you may wish to change the variable changes for each block of code. For example you might think that a walk would use up more than one point of energy and change energy to -2 or -3.

Create the block like this
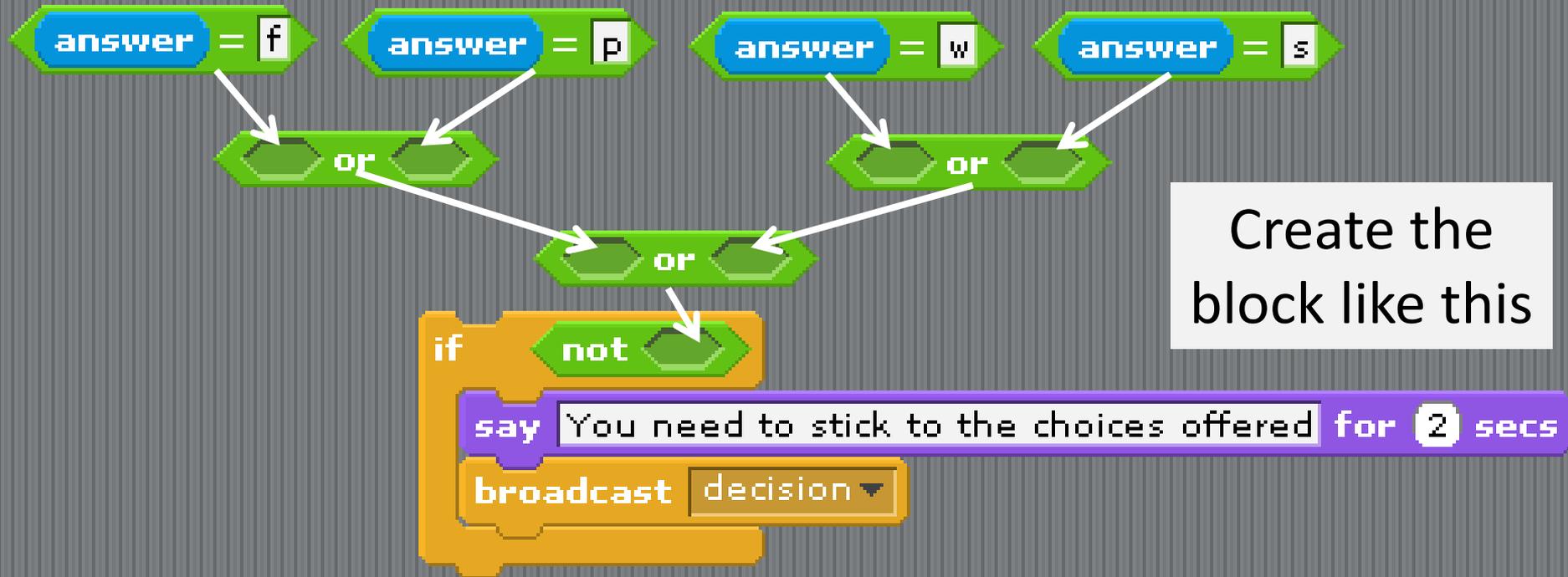
If the player types in an answer which is not an f, p, w or s then they will be told to stick to the choices offered and the decision block will be run again .

Good programming takes into account the player making a mistake.

Your finished block will look like this
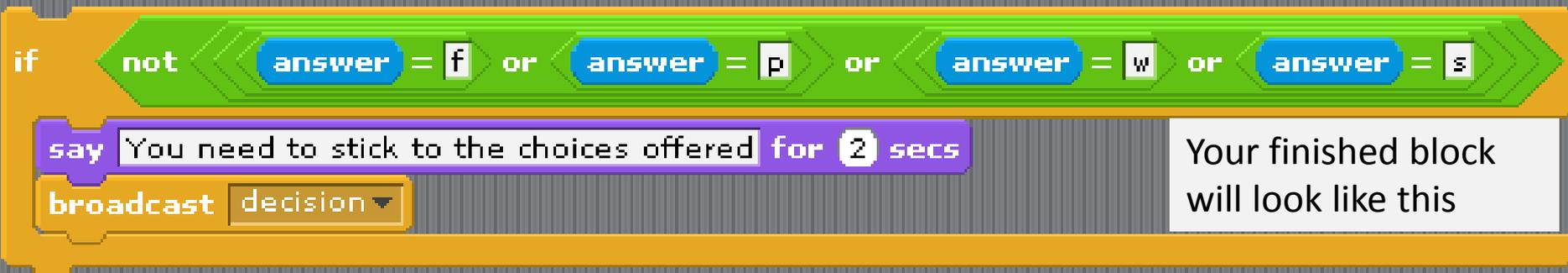
Make all the action blocks return to the decision control panel once they have finished running

when I receive [eat ▼]
go to x: (124) y: (-125)
repeat (10)
  switch to costume [eata ▼]
  wait (0.3) secs
  switch to costume [eatb ▼]
  wait (0.3) secs
broadcast [decision ▼]

when I receive [play ▼]
repeat (400)
  point towards [ball ▼]
  move (2) steps
broadcast [decision ▼]

when I receive [walk ▼]
go to x: (-165) y: (-110)
point in direction (90 ▼)
glide (3) secs to x: (138) y: (-110)
point in direction (-90 ▼)
glide (3) secs to x: (-162) y: (-110)
point in direction (90 ▼)
broadcast [decision ▼]

when I receive [mad ▼]
repeat (8)
  switch to costume [walka ▼]
  say [Where is my tail?] for (0.5) secs
  point in direction (90 ▼)
  wait (0.5) secs
  point in direction (-90 ▼)
broadcast [decision ▼]

when I receive [sleep ▼]
go to x: (66) y: (-110)
switch to costume [sleep ▼]
think [Zzzzzz] for (6) secs
broadcast [decision ▼]

Add a broadcast decision block to all of these blocks of code

Set the ball to mark the age of the dog

Create this block inside the **dog** sprite to make the dog grow

```
when [flag] clicked
set size to (64) %
set [age in days ▼] to (1)
forever
    wait (30) secs
    change [age in days ▼] by (1)
    say (join (join (join (name) (is)) (age in days)) (days old)) for (2) secs
    broadcast [grow ▼]
```

```
when I receive [grow ▼]
change size by (2)
```
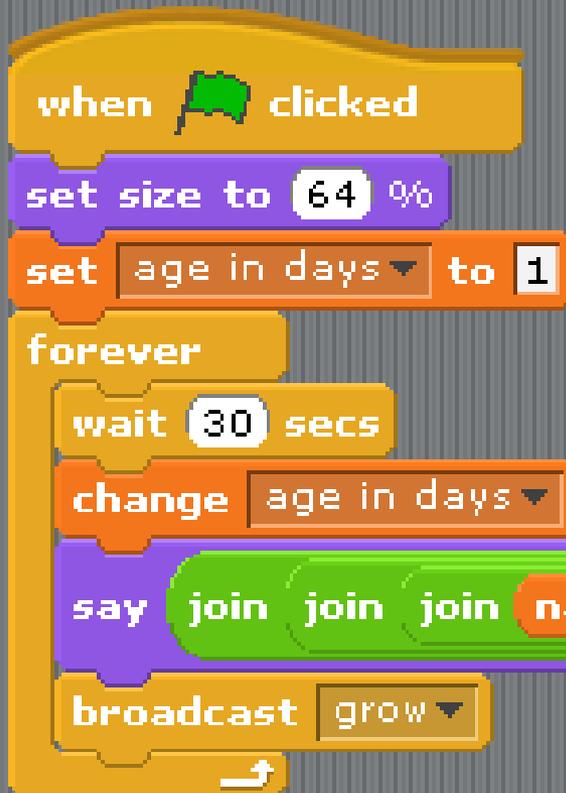
Create this block inside the ball sprite
Every 30 seconds the ball will report that the dog is x days old. Looking inside the age in days variable to see how many days to report.

Test and debug your code

If things don't work as they should
You can check your code against a full copy here

```
when I receive [ponder ▾]
switch to background [choose ▾]
```

```
when I receive [toomuchenergy ▾]
switch to background [choose ▾]
```

```
when I receive [eat ▾]
switch to background [eat ▾]
```

```
when I receive [play ▾]
switch to background [play ▾]
```

```
when I receive [sleep ▾]
switch to background [sleep ▾]
```

```
when I receive [sad ▾]
switch to background [choose ▾]
```

```
when I receive [walk ▾]
switch to background [walk ▾]
```

Code blocks in stage area

Code in ball sprite area

```
when [green flag] clicked
set size to (64) %
set [age in days ▼] to (1)
forever
    wait (30) secs
    change [age in days ▼] by (1)
    say (join (join (join (name) [is]) (age in days)) [days old]) for (2) secs
    broadcast [grow ▼]
```

```
when [green flag] clicked
forever if <touching [dog ▼]?>
    glide (1) secs to x: (pick random (-200) to (200)) y: (pick random (-200) to (200))
```

Code in dog sprite area 1/3

Script 1 — when I receive decision:

```
when I receive [decision]
if < Energy < 0 >
    broadcast [rspca]
    say (join (name) [ has been taken away by the RSPCA ]) for (2) secs
    say [as you didn't look after them well enough!] for (2) secs
    stop all ●
if < Energy > 15 >
    broadcast [mad]
    change [Energy] by (-5)
    wait (8) secs
if < Happiness < 0 >
    broadcast [sad]
    wait (4.5) secs
broadcast [ponder]
ask [Type f to feed, s to sleep, p to play or w to walk] and wait
if < answer = [s] >
    broadcast [sleep]
    change [Energy] by (2)
    change [Happiness] by (-1)
```

Script 2 — when I receive rspca:

```
when I receive [rspca]
go to x: (66) y: (-110)
switch to costume [sleep]
```

Script 3 — when I receive play:

```
when I receive [play]
repeat (38)
    switch to costume [walka]
    wait (0.1) secs
    switch to costume [walkb]
    wait (0.1) secs
```

Script 4 — when I receive ponder:

```
when I receive [ponder]
go to x: (0) y: (0)
switch to costume [ponder]
```

Script 5 — when I receive sad:

```
when I receive [sad]
switch to costume [sita]
say [Howl] for (2) secs
switch to costume [sitb]
say [Howl Howl!] for (2) secs
```

Script 6 — when I receive walk:

```
when I receive [walk]
repeat (11)
    switch to costume [walka]
    wait (0.25) secs
    switch to costume [walkb]
    wait (0.25) secs
```

Code in dog sprite area 2/3

```
if  answer = f
    broadcast eat
    change Energy by 3

if  answer = p
    broadcast play
    change Energy by -1
    change Happiness by 3

if  answer = w
    broadcast walk
    change Energy by -1
    change Happiness by 3

if  not ( answer = p or answer = w or answer = f or answer = s )
    say You need to stick to the choices offered for 2 secs
    broadcast decision
```