Computer Programming

with

Scratch

http://scratch.mit.edu

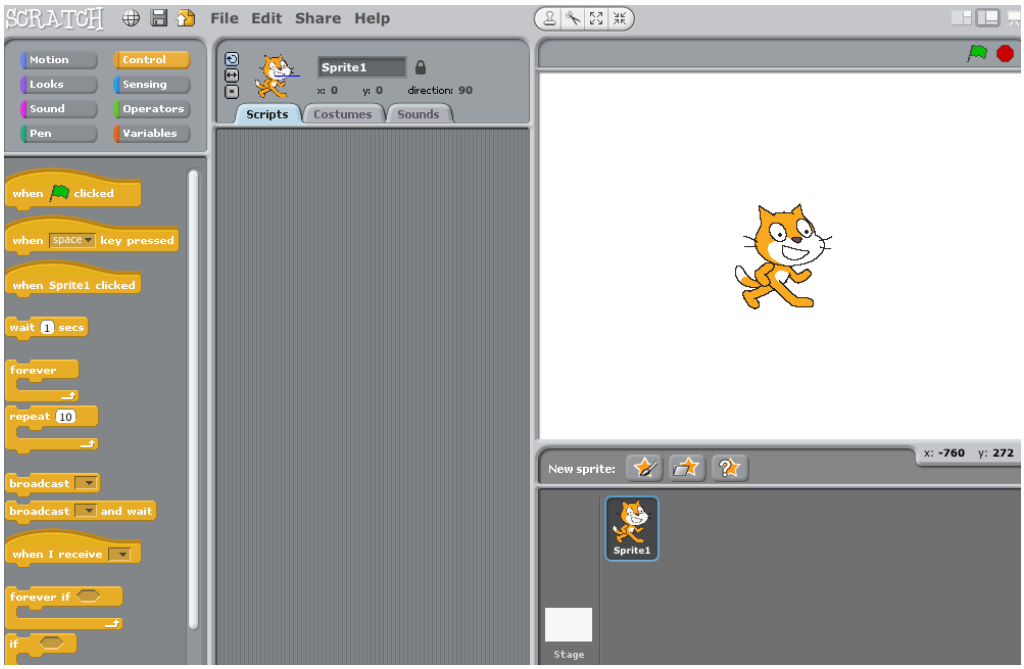
Brendan Smith, Digital Enterprise Research Institute(DERI) NUI Galway

Scratch is a programming language developed by the MIT Media Lab in the United States that makes it easy for users to create their own interactive stories, animations, games, music, art and to share these creations on the web.
It has a simple structure that is based around snapping together **visual blocks** of computer code that control sound, music and images. Hence it is ideally suited for children as it compliments the artistic elements of the primary school curriculum allowing users to bring their artistic skills into new digital dimension to create computer games, animations and stories.
As children create and share Scratch projects, they learn important mathematical and computational ideas such as geometry and logic, while also learning to think creatively, reason systematically, and work collaboratively.

What is a Programme?
A programme or program is a set of instructions that tells a computer or a device what to do. These instructions or commands are written in an artificial (non-speaking) language. The text used is often referred to as *code* or *computer code*.
Computer programming or *coding* is the process of writing code.

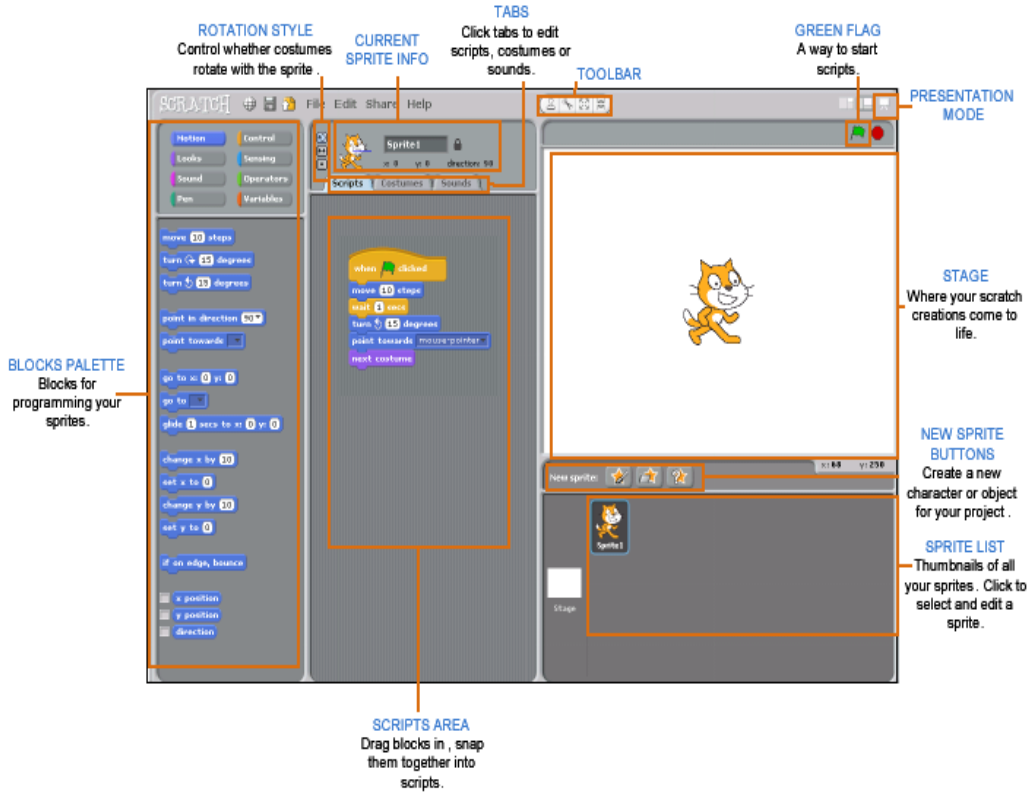
Where does the term Scratch come from?
Scratch is the name given to this very powerful but very easy to use programming language
The term Scratch was chosen by its inventors because of its similarities to a Disc Jockey's method of mixing different music tracks together to give new sounds.
For it is a key element of the Scratch website that members can download other people's computer games for instance and change the details (e.g. 'speed up' or 'slow down' animated characters in a game)

Introduction to Scratch Interface



Lesson 1: A Moving Talking Sprite

A ***Sprite*** is an animated character or object in your programme. The name originally comes from the word *Spirit* and is most commonly used to refer to an elf or fairy and is mentioned as such in the books *of Artemis Fowl* and the *Spiderwick Chronicles*.



In Scratch, Sprites can move around, be active or be objects that stay still. We will choose a Sprite character that will move about your screen.

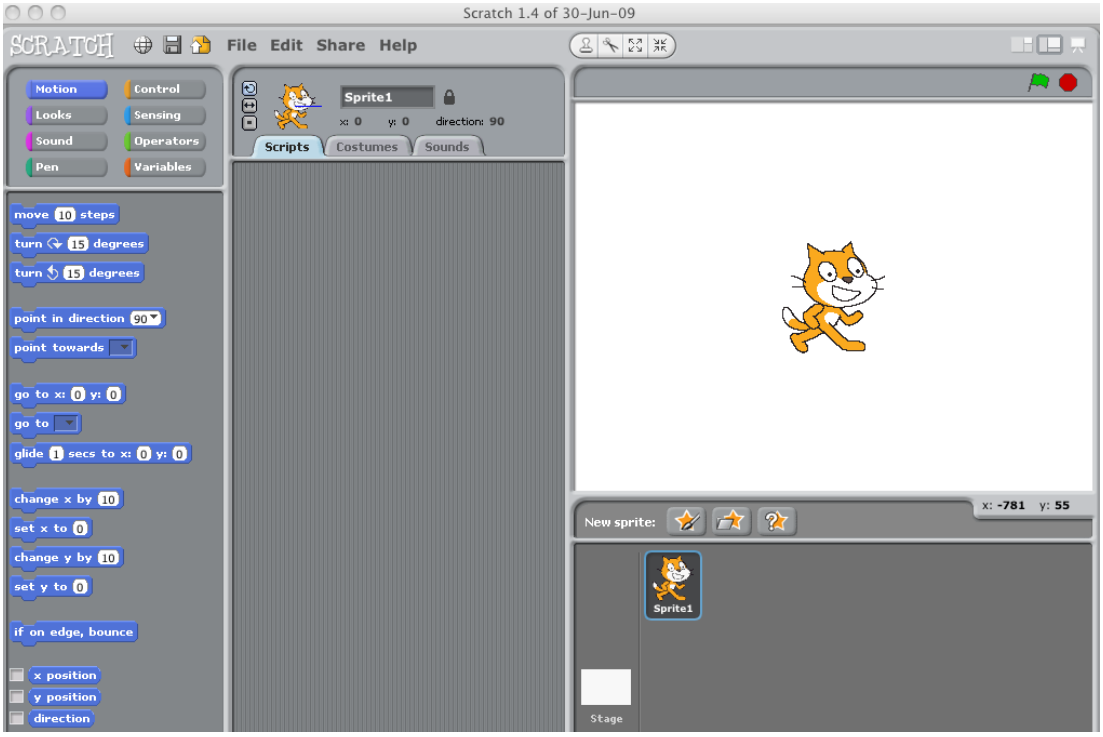
However please note that *a Sprite cannot do anything by itself.*

A Sprite's actions comes from the scripts in the ***Script*** (or *programme code*) Window or area. These scripts are the instructions or commands that tell the sprite what to do. You drag these individual pieces of code from the Blocks Palette into the Scripts area. These blocks then fit together like a jigsaw puzzle to create the instructions.

So let us enter Scratch and program the Sprite to talk and to move around the screen!

1. Open **Scratch**
a. Go to the folder where you copied "Scratch."
b. Double click on the Scratch icon.

2. You will see the opening screen.



Take a few minutes to familiarise yourself with the main features of this screen or what is referred to as the *Home Page*.

Notice that the small version of the cat is highlighted in *blue* signifying that it is the *active* component.

Placing Text
By placing blocks in the Script area, the Sprite will tell us his name.

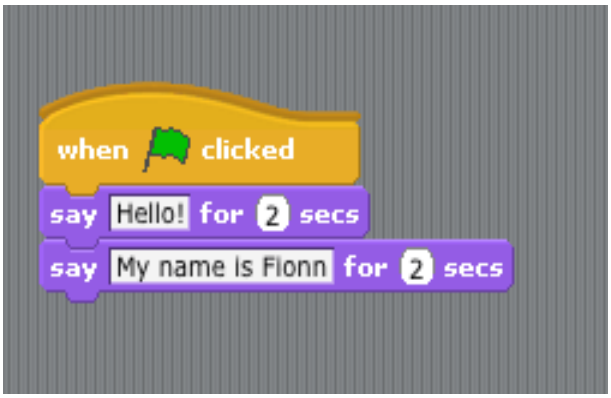
First go to the ***Commands Folders*** located at the top left side of the Scratch screen.

Go to the ***Control*** folder.

Place the following block in the Script area:

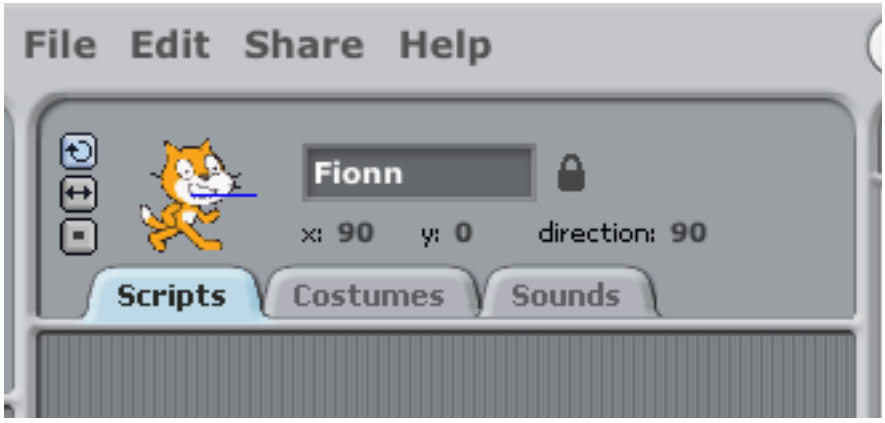
This will mean that once the ***Green flag*** above the Stage is clicked, the Sprite will follow the commands that are placed in the Script palette.

Go to the **Looks** section located in the panel at the top left hand corner of the screen.
Select the blocks below (beginning with '*say*').
Type in the text *Hello!* and *My name is Fionn* before placing both in the Scripts area.

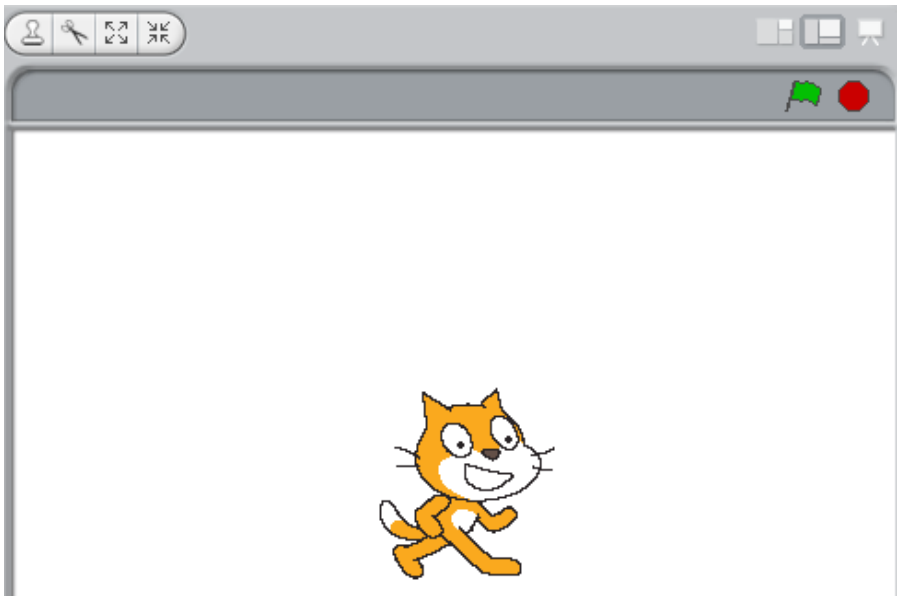


Make sure the blocks *click* together

Type in the text *Fionn* in the relevant box in section located above the *Scripts Palette*



Click the *Green Flag* at the top left hand side of the computer screen and see what follows:

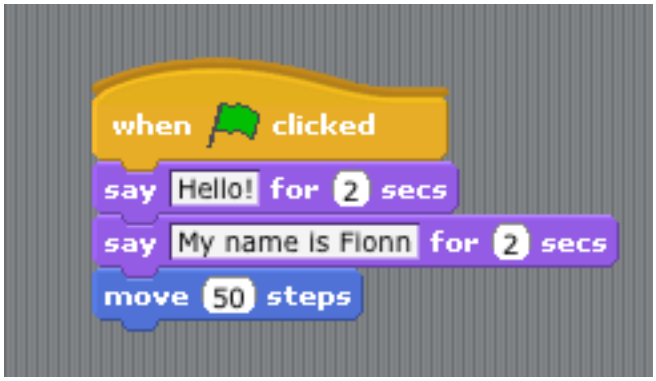


Now we need to get the cat moving.

Go to the *Motion* folder

Select the *Move* block and change the number of steps to 50

Attach this block to the rest of the blocks in the scripts area and start the program.



To increase movement of the Sprite and to give the impression of walking, first go to the *Control* folder and select the *Wait* block

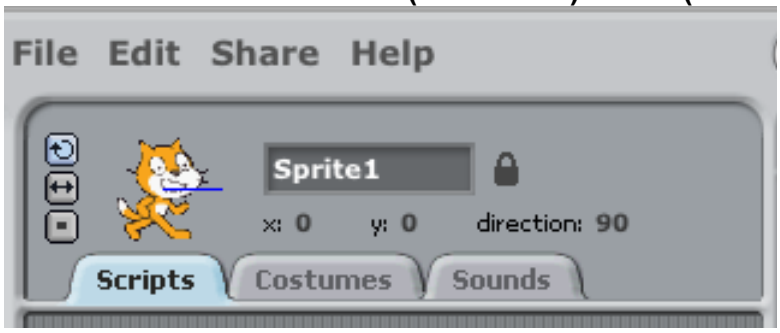
The place this block in the Script area with the addition of some extra Move blocks as follows:

However, we now have a problem with the Sprite.

As you may have noticed, if we keep using this script, the cat will keeping moving until he almost disappears off the screen.

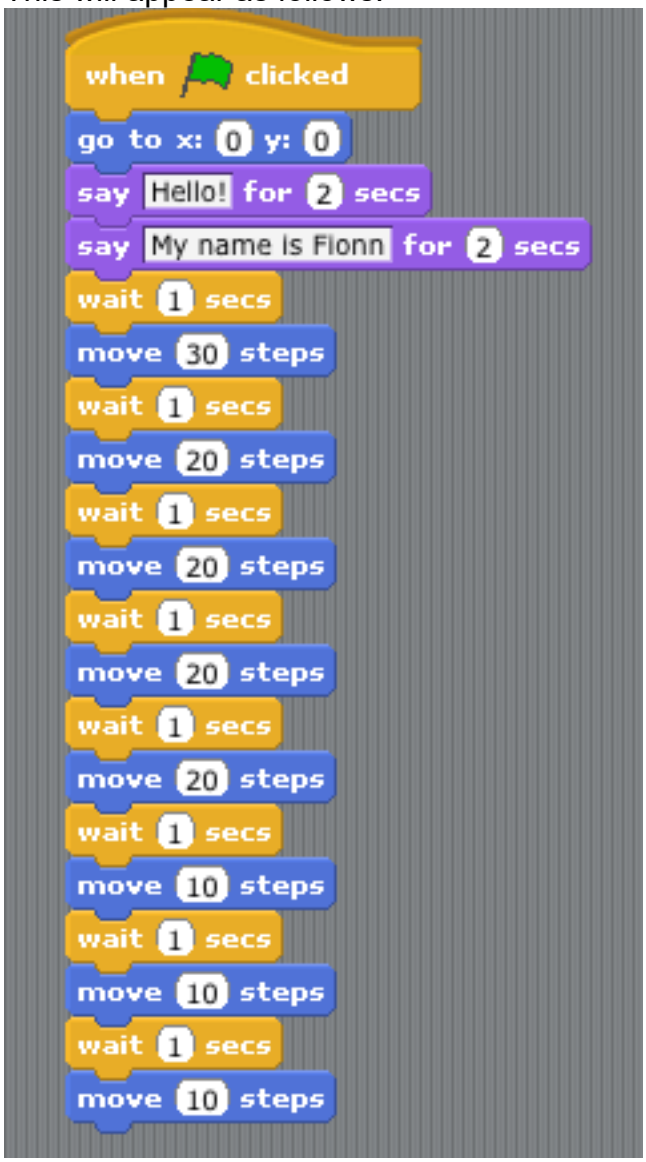
So we have to put in an extra command that will bring him back to the centre of the screen, on every occasion that we use this set of instructions.

The screen is divided into **X (horizontal)** and **Y (vertical) coordinates**.



Hence we can place a piece of code or block at the beginning of the set of commands that will instruct the Cat to move back to the centre every time we select the Green flag.

This will appear as follows:



Test the effectiveness of this new piece of code by using the mouse to position the *Sprite* towards the bottom or top of the screen before clicking on the *Green Flag* icon.

Different methods other than a Green Flag can be used to start a Script.

For instance, the Space Bar or Arrow Keys.

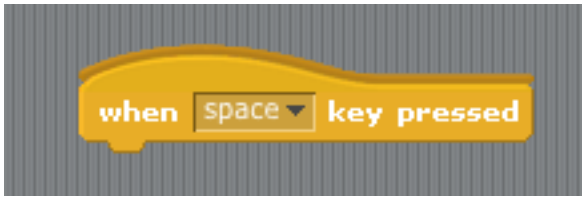
So replace

with

in the Script.

Now *click the Space Bar* on the computer keyboard to start the programme.

Replace



with

And start the programme by *clicking on the Cat*

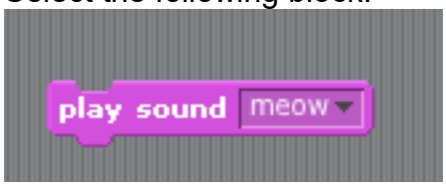
Revert back to the Green Flag block

Lesson 2 - Placing Sounds in a Script

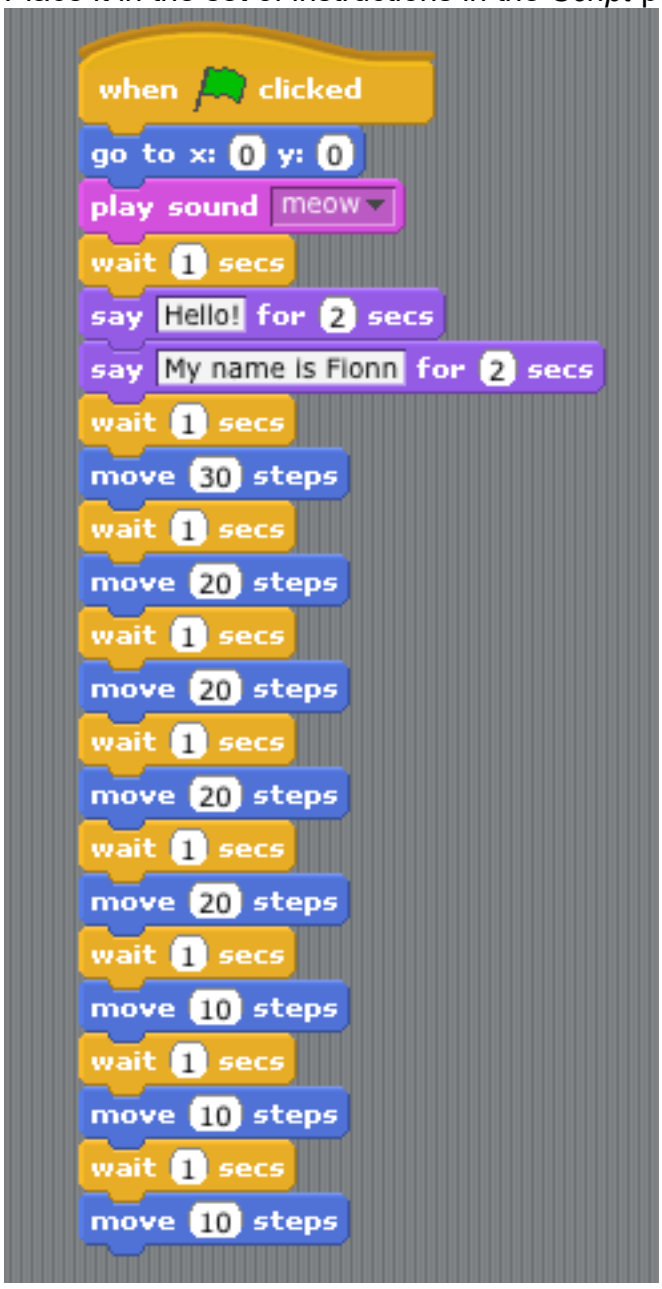
Lets Hear the Cat Purr!

Go to the *Sounds* folder in the *Commands* section

Select the following block:



Place it in the set of instructions in the *Script* palette



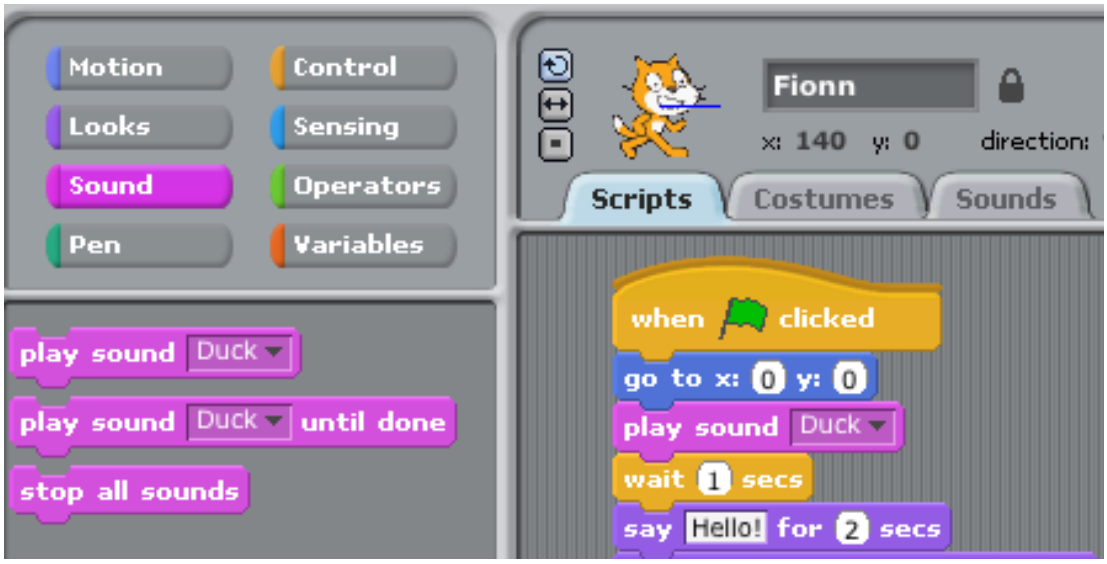
Play the programme.

Now replace the cat meow in the Script with other animal noises

This can be done by going to the *Sound* section located above the Scripts palette.

Then chose *IMPORT*, pick your chosen sound and then click on *OK*.

Then click on the *Sprite icon* before placing the new sound in your Script



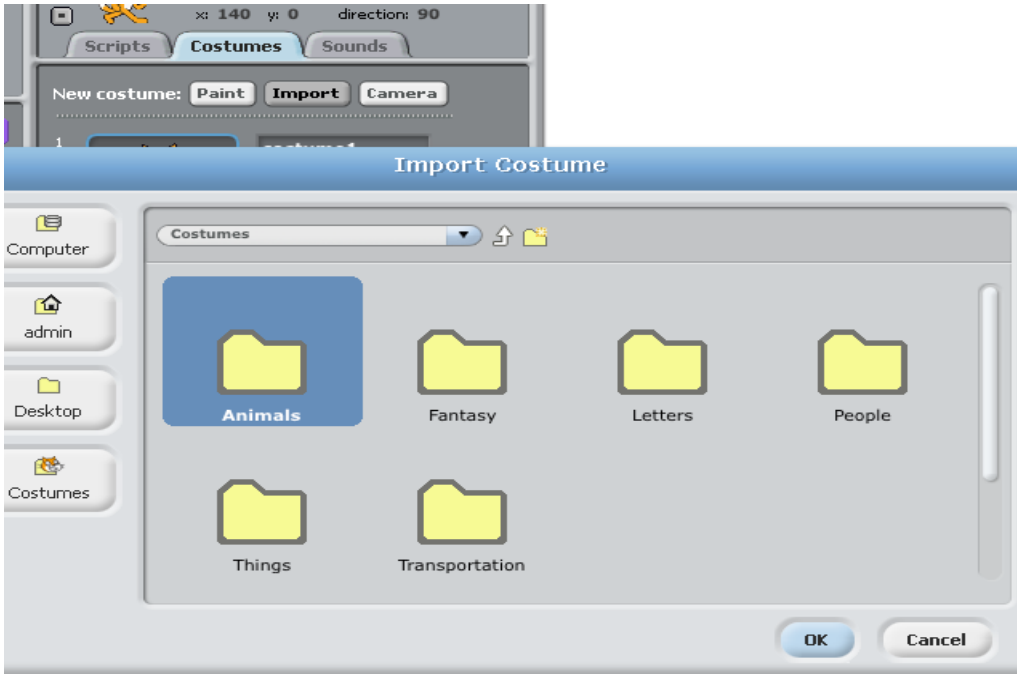
Change Sound back to Meow.

Lesson 3 - Making the Sprite Walk Better

Let us now make the Cat more realistic in his movements by getting him to move his legs when he walks!

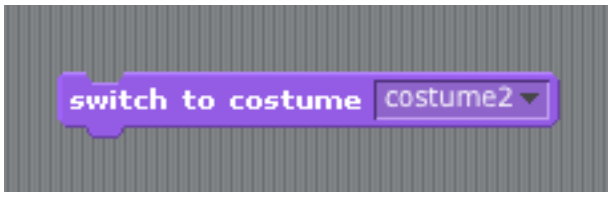
This can be done by changing the physical look of an individual Sprite under the section labelled *Costumes.*

Go to *Costumes* and use the option *Import* to add on a second image or *Costume* of the same *Sprite* from the *Animal folder* in Scratch.



After the first *Move and Wait* Block located in the Script, place in a *'Switch to costume* block with *Costume 2'* taken from the *Looks* Folder.

The option *Costume 2* is chosen by clicking on the black triangle icon located to the left side of this block



Then for the next *Move and Wait* Blocks, place in a *Switch to costume* block *with Costume 1.*

Repeat this process all the way through the Script, thus alternatively *Costume 1* and *Costume 2*.

Play the Script by clicking on the *Green Flag* icon

Lesson 4 - Multi-Coloured Sprites

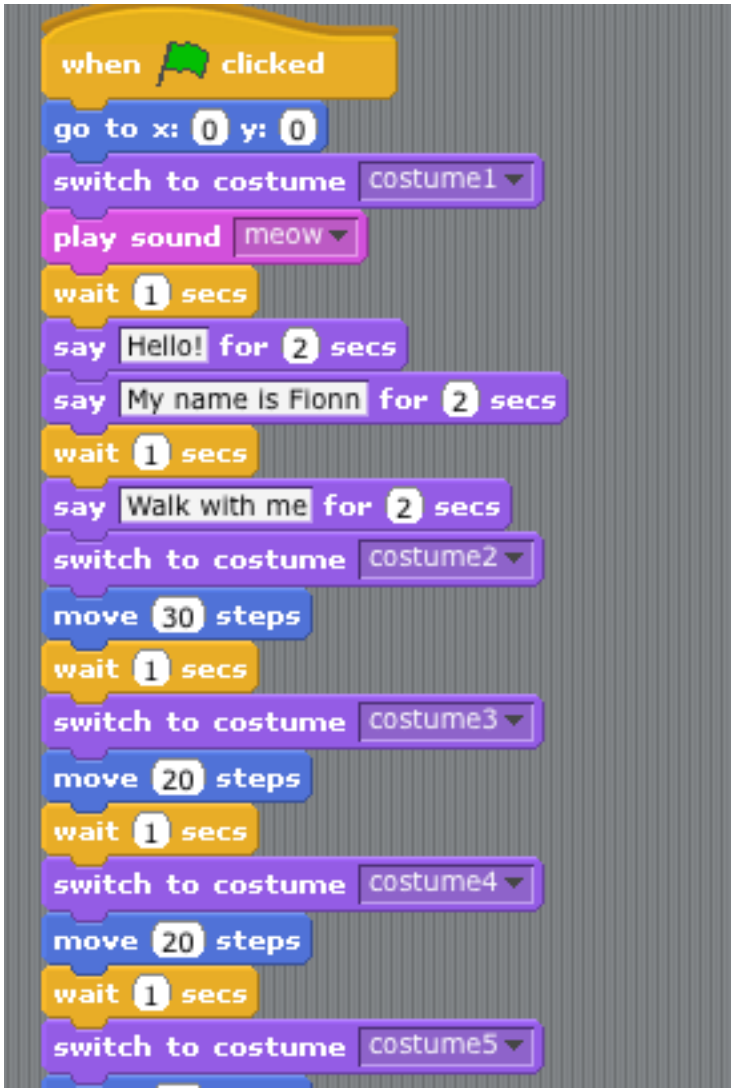
The Chameleon Cat! *Changing the Colour of the Sprite*

Let us change the colour of the cat whilst he walks.

To do this go to *Costumes* option and select *Copy* for *Costume 1*.
Do the same for *Costume 2.*

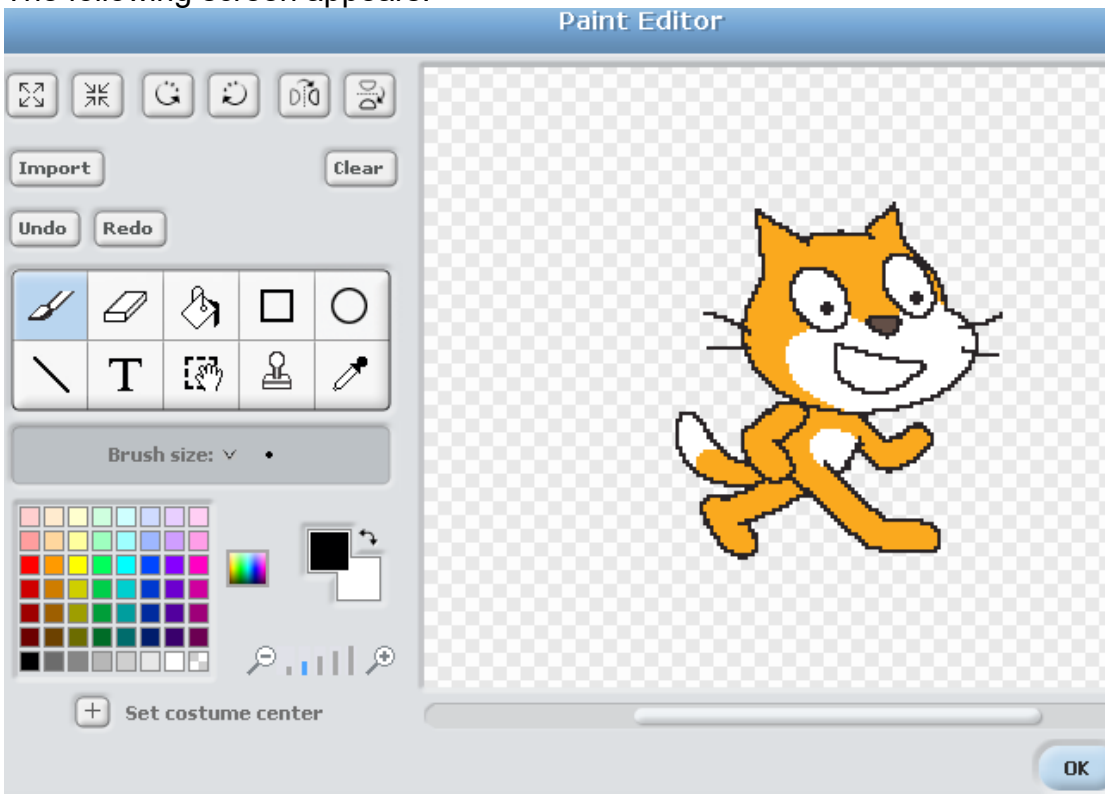
Repeat this process for both Costume 1 and 2 until you have 9 Costumes.

Then change the Costume numbers in the Script so that they appear chronologically.



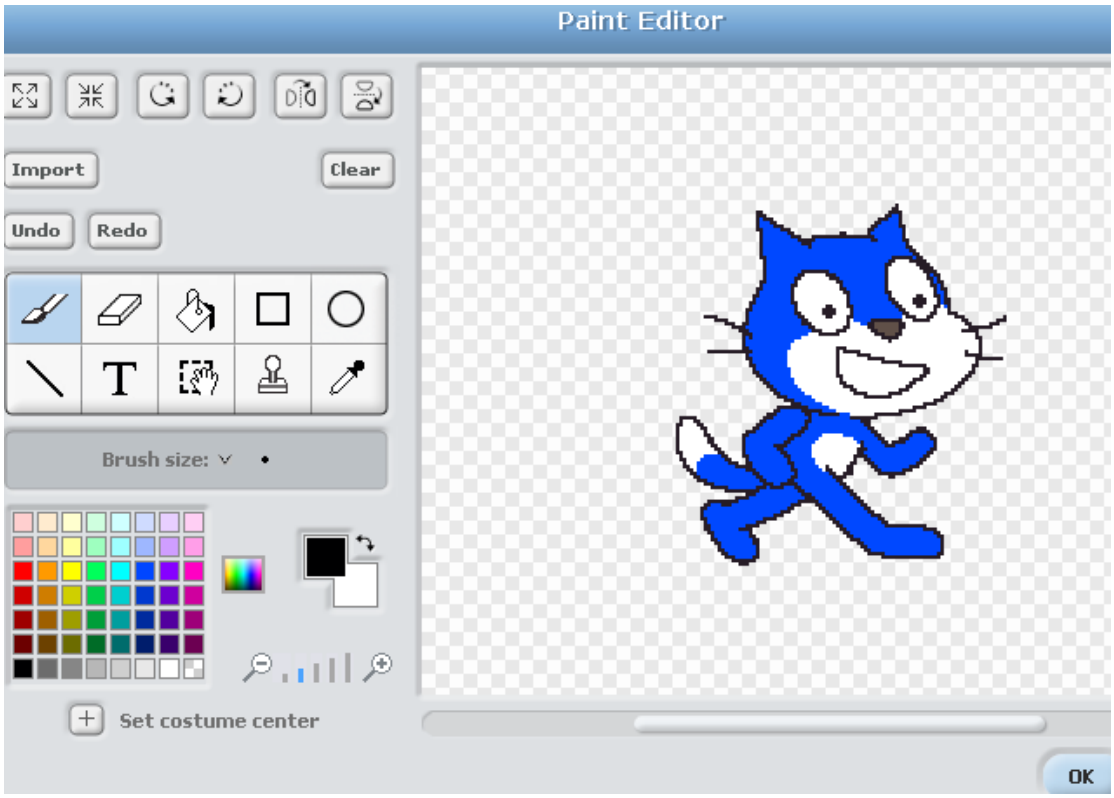
Then select the Edit option for Costume 3.

The following screen appears:



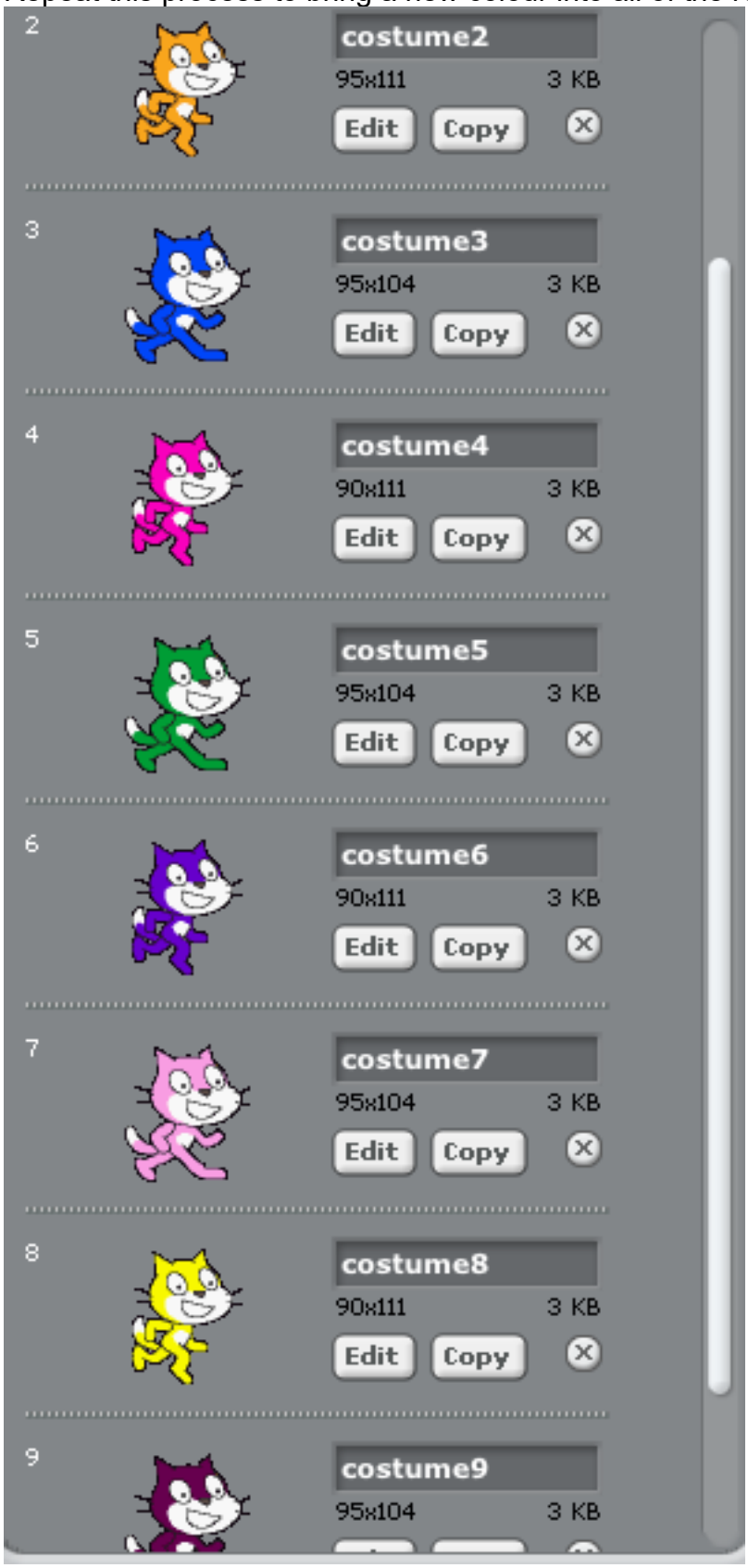
This is the ***Paint Editor***, which allows you to colour in existing Sprites or paint and draw new ones.

In our case, choose a different colour from the palette box on the left and use the paint bucket icon to bring this new characteristic into the existing Sprite on the screen.



When finished, press okay.

Repeat this process to bring a new colour into all of the remaining costumes



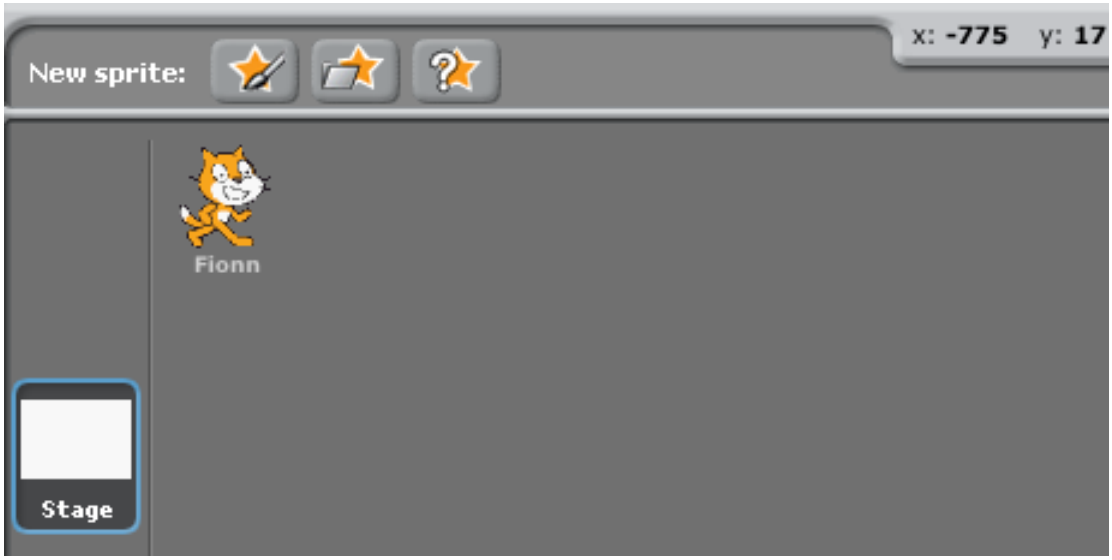
Play the Script by clicking on the Green Flag icon and watch the cat of many colours walk!

Lesson 5 - Changing the Background (Stage)

At present, we are using a blank background for the cat.
So let us bring some excitement into his life by having the cat walk around a new landscape.

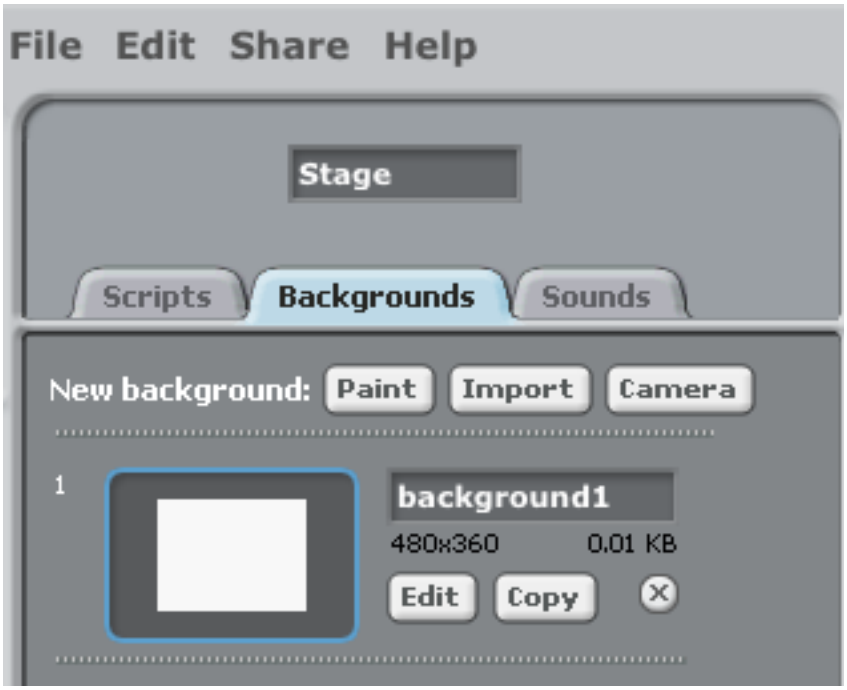
First go to the **panel** at the bottom right hand side of the screen.

Fionn the cat is presently highlighted meaning, as mentioned previously, that the *Sprite* is the *action* element in the *Script*.



So *click* on *Stage*. The Stage icon is now highlighted in *blue* meaning that it has now become the *active* zone which the Scratch operator can now changes.

Return to the *Script*.



Click on *Import.*

Go to the folder labelled *Indoors* and select *Bedroom 1.*

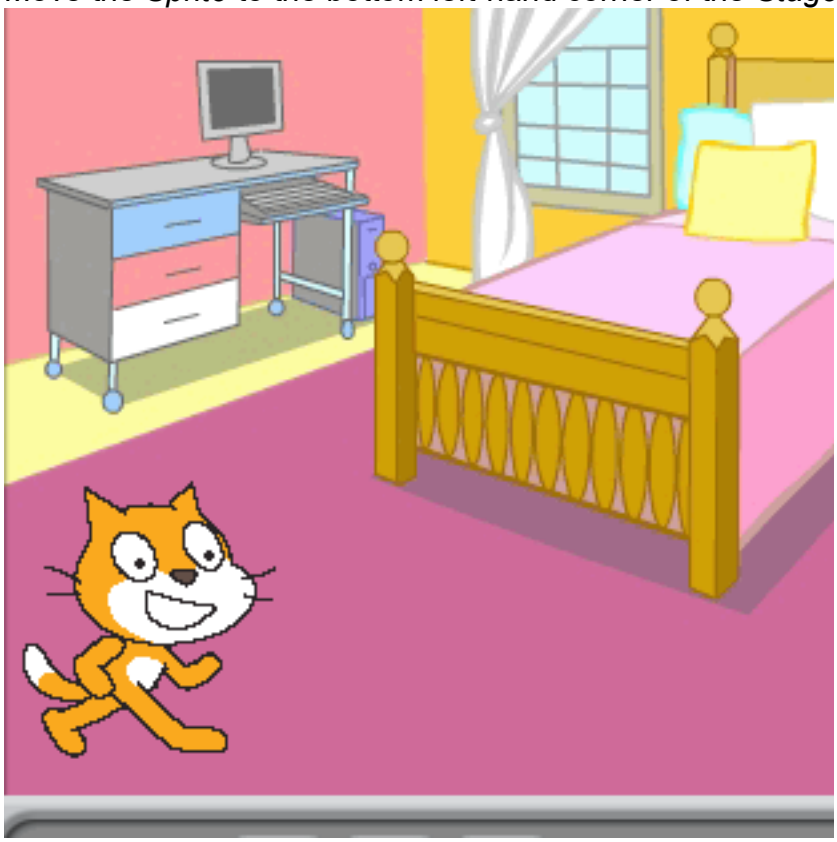
Click on the *Green Flag* icon to start the programme.

But we now have got a problem, as the cat looks like as if he is walking on air! So what do you think that we have to do to the *Script* in order to make the cat move across the floor of the bedroom?

We have to put in some extra command code that will ensure that the cat walks along the floor.

First step is to make the *Sprite* (cat) the *active* element instead of the *Background*.
So *double Click* on the *Background* icon to move back to the *Script* of the *Sprite*.

Move the *Sprite* to the bottom left-hand corner of the *Stage* screen.



GEOMETRY

Computer Images are made up of Pixels. They can be identified using X & Y Coordinates.

A computer screen or a picture is made up of basic units known as **Pixels**. It is an abbreviation of two words, *picture* and *element*.
Pixels are the smallest units of color on a computer display or in a computer image that can be controlled or programmed.
X, Y coordinates are respectively the **horizontal** (X) and **vertical** (Y) addresses of any pixel or addressable point on a computer display screen.
The *x* coordinate is a given number of pixels along the horizontal axis of a display starting from the pixel (pixel 0) in the centre of the screen. The *y* coordinate is a given number of pixels along the vertical axis of a display starting from the pixel (pixel 0) in the middle screen. Together, the *x* and *y* coordinates locate any specific pixel location on the screen.
X & Y coordinates are part of the branch of **mathematics** known as **Geometry** which is concerned with questions of relative position of figures, shape, size, and the properties of space.

Memorise or write down the X and Y coordinates that are shown.
Go to the Motion folder of the Command panel.

Find the ***Go to the X: Y:*** block

Move this block into the *Script* area and position it directly under the first (*Green Flag*) block in the set of programme instructions.

Fill in the X & Y coordinates that you had just previously written down.

Click the Green Flag icon to start the programme.

Question: What do we have to do to get the cat to jump onto the bed?

The operator of course must instruct the Sprite to move onto the bed by placing, in the correct spot in the programme, a Motion block that includes the correct X and Y coordinates.

So first start the programme.
Look at the location where the cat stops and the programme ends

Move the cat onto the bed

Memorise or write down the X and Y coordinates shown

Go to the Motion folder of the Command panel.

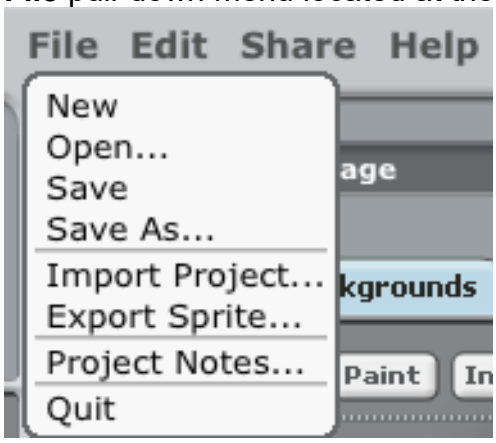
Find the **Glide 1 secs to X: Y:**

Move this block into the Script Area and position it in the correct location in the set of programme instructions (probably at the end).

Fill in the X & Y coordinates that you had just previously written down.

Click the *Green Flag* icon to start the programme.

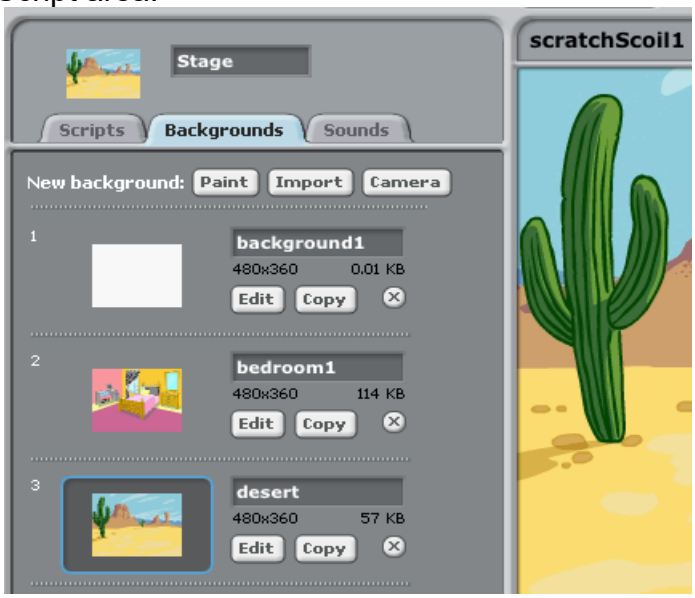
Save your programme file with the name *Scoil1* by selecting **Save As** in the **File** pull-down menu located at the top of the Scratch screen.



Exercise:
Place some further blocks of code in the programme that will allow the cat to jump off the bed and walk a few steps.
Furthermore, ensure that the cat changes colour for each step of the remainder of his walk.

Exercise: Replace the background with a new image from the Background folder (See folder icon below).

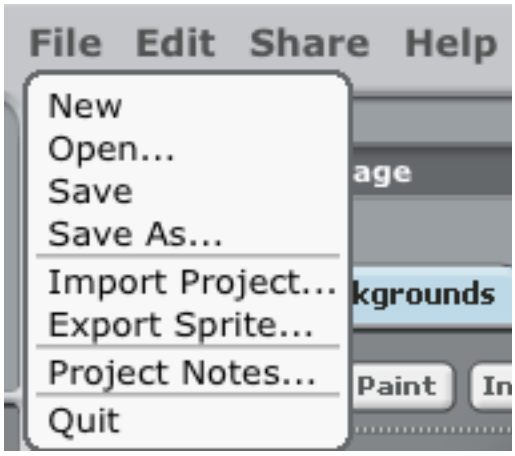
Notice that the new background appears in the list of backgrounds in the Script area.



To revert to a previous or new background, just bring your cursor onto the appropriate background image and click.
The new image is highlighted and it becomes the background that appears in the *Screen Stage*.

Then re-do the instructions so that the *Sprite* jumps on and off some object that is in your selected picture.
Note: the user has to click on the *Sprite* image of the Cat before it is highlighted (in blue) allowing instructions to be added.

Save your new programme file with the name *Scoil2* by selecting **Save As** in the **File** pull-down menu located at the top of the Scratch screen.

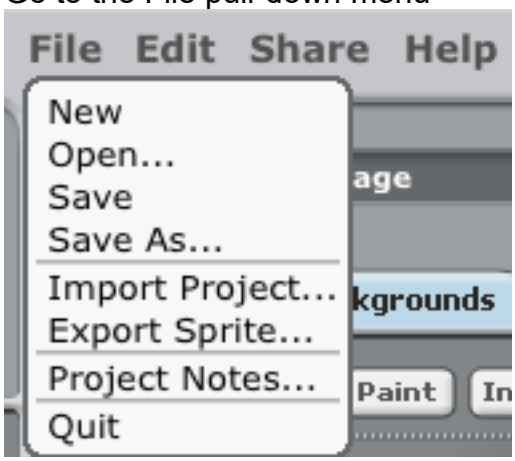


Exercise
Walking along the Road
Draw a new Sprite e.g. a boy or girl
Draw different costumes (versions)
Draw a street scene (background)
Write a programme to have the boy or girl walk along the street or road

Lesson 6 - Barking Dog Chases Cat!

Let us have a barking dog chase after *Fionn* our lovable cat!

Go to the File pull-down menu



and choose *Open*.

Then click on Select **Scoil1** to open your previous Scratch file

Go to the *New Sprite* tool bar located at the bottom of the *Stage* Screen and select the middle icon

Then go to the *Animal* folder and select the first running dog image (dog1-a) as shown below

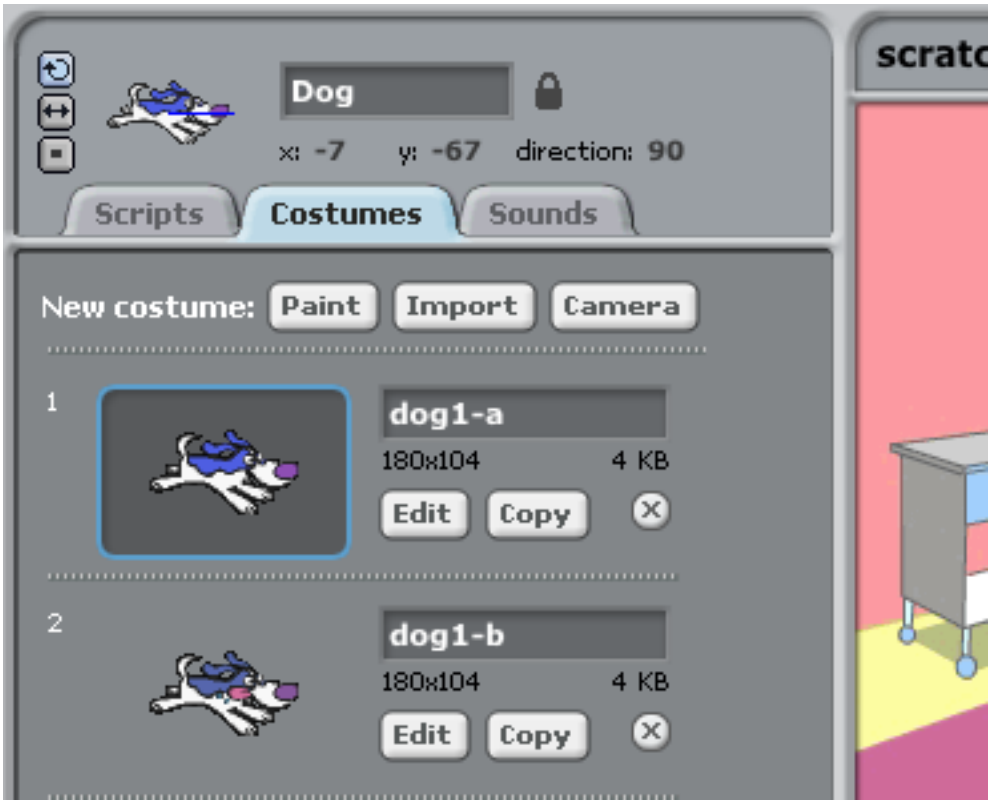
Notice that the Dog icon is now highlighted in blue outline meaning it is the Sprite that is *active*.

We now have to build a script or programme to operate the Dog (aka 'Blueback') allowing him to run and to talk.

So first let us select another version of the dog that will, in combination with the existing dog sprite, give the impression of movement.

This can be done by going to the section labelled *Costumes.*

Use the option *Import* to upload the second image (dog 1-b) or of the same Sprite from the Animal folder in Scratch.


Dog
x: -7
y: -67
direction: 90
Scripts
Costumes
Sounds
New costume:
Paint
Import
Camera
1
dog1-a
180x104
4 KB
Edit
Copy
2
dog1-b
180x104
4 KB
Edit
Copy

Building a Script to operate the Dog
Go to the *Commands* Folders located at the top left side of the Scratch screen.

Go to the *Control* folder

Place the following block in the Script area:

This will mean that once the *Green flag* above the *Stage* is clicked, the Sprite will follow the commands that are placed in the Script palette.

However remember that we now have two Sprites, namely the cat and the dog.
We want to have the dog *only appear on stage* (screen) *after the cat has jumped on the bed*.
Hence we must have him *hide* when the programme begins, and only appear (*show*) in the scene at a certain location (X & Y) after a certain amount of time has elapsed (*wait*).

The following Commands in a combination sequence will allow the operator to undertake this task:

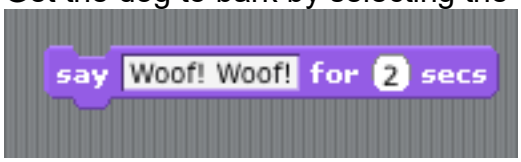
Show and *Hide* blocks in the *Looks* folder

Wait bock in the Control folder

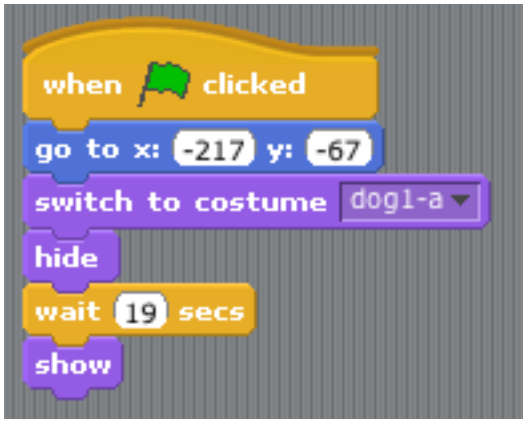
&

X & Y coordinates block in the *Motion* folder

Get the dog to bark by selecting the *Say* block in the *Looks* Folder



To get the Sprite to appear as if it is moving, we first have to place a *Switch to costume* command block towards the beginning of the script so that the first version of the dog, namely *dog1-a* <u>always</u> appears once the programme starts.



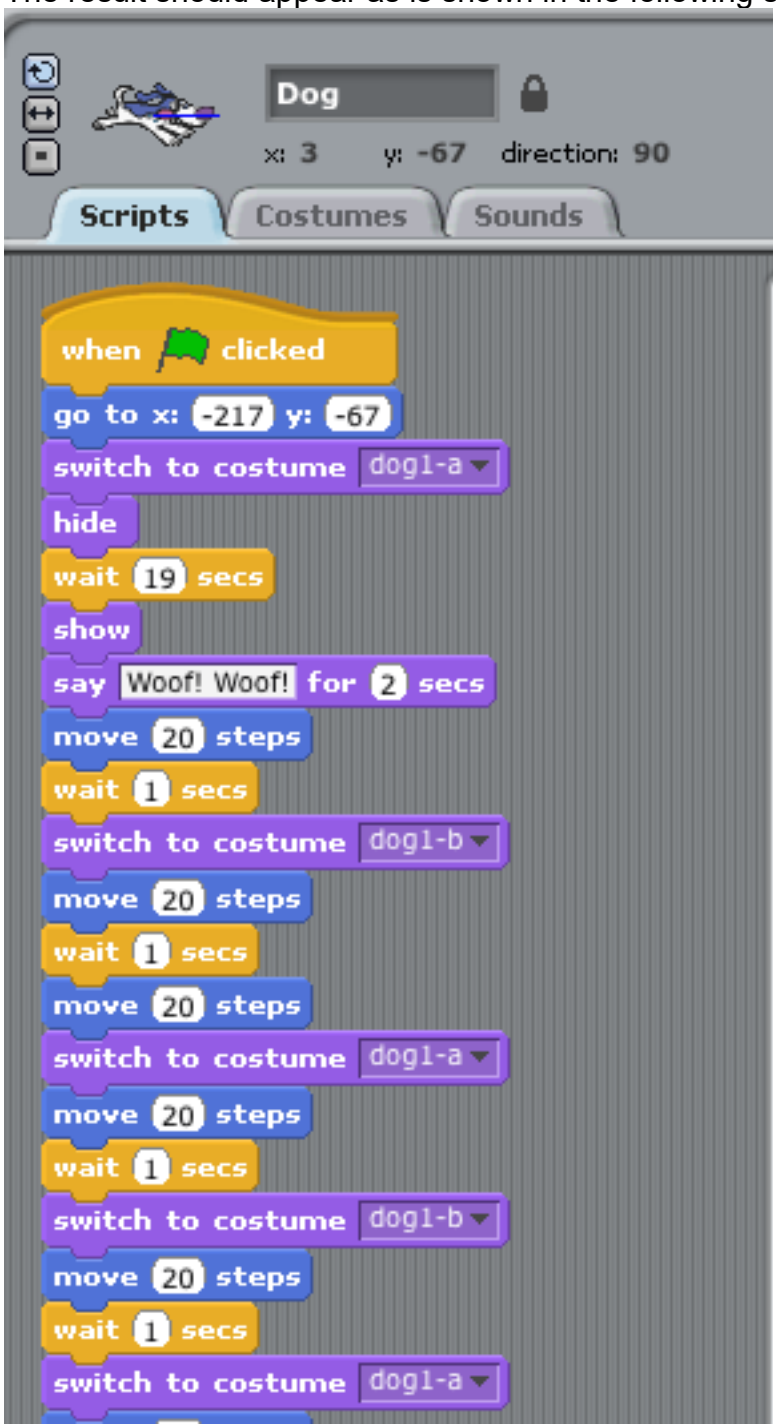
After the first *Move* (in this case got to x ___ & y___ block) and *Wait* block located in the Script, place in a '*Switch to costume'* block with *dog1-b* taken from the *Looks* Folder.

The option *dog1-b* is chosen by clicking on the *black triangle* icon located to the right side of this block

Then for the next *Move* and *Wait* blocks, place in a *Switch to costume* block with *dog1-a*.

Repeat this process all the way through the Script, thus alternating in sequence between *dog1-a* and *dog1-b*.

The result should appear as is shown in the following screen:

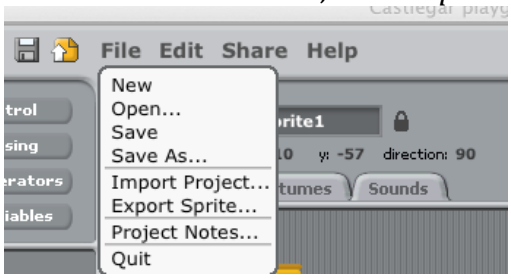


Exercise
Shark chasing Little Fish
Draw a Sprite of a Shark
Draw different costumes (versions)
Draw a Sprite of a Fish
Make the fish appear small
Draw different costumes (versions)
Draw an underwater scene (background)
Write a programme to have the shark chase the fish and eat it!

Lesson 7 - Changing/Enhancing Existing Projects
a) The Playground

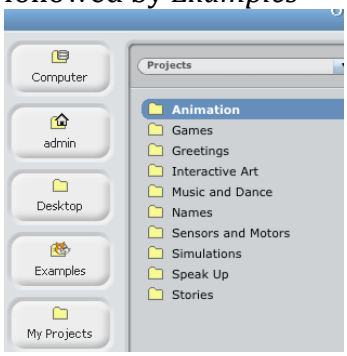
This lesson allows students to change and enhance the code of existing projects, which is a central tenet of the Scratch ethos.

In the Scratch interface, select *Open* in the *File* drop-down commands

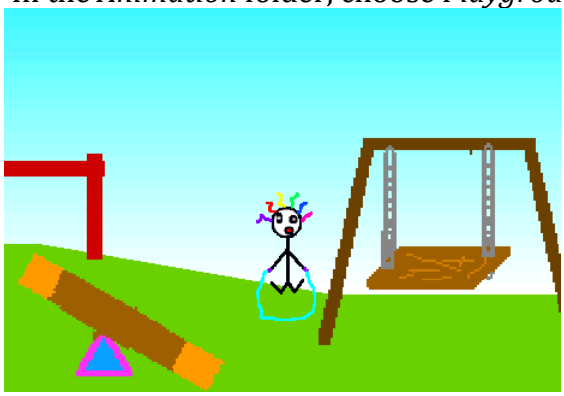


.

followed by *Examples*

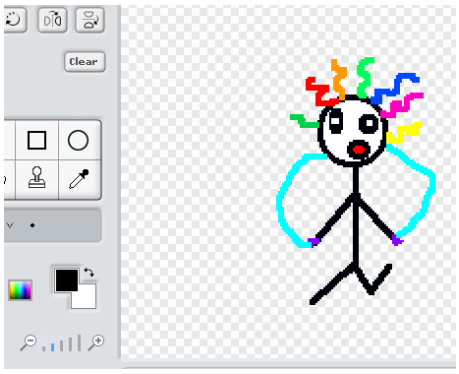


In the *Animation* folder, choose *Playground*



Exercise

1. Change the *speed* of the *See-Saw* and the *Swing*
2. Draw a boy or girl skipping. These characters can be simple *matchstick* people
3. Draw in two children positioned at each end of the See-Saw by changing the existing See-Saw sprite



b) The Trampoline

In the *Animation* folder, choose *Trampoline*

Exercise
1. Integrate into the programme the two extra costumes versions of Jodi (hitting1 & ball1) that are not active.

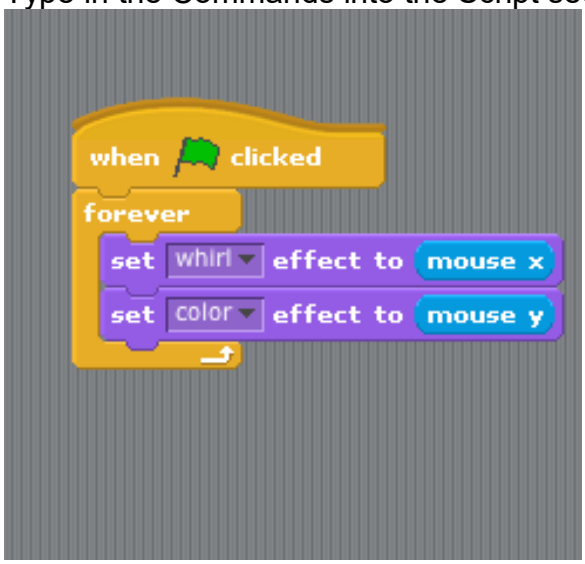
Lesson 8 - The Whirling Sprite

*This lesson introduces students to the **Sensing** folder in the **Commands** section to change the Sprite's physical appearance by movements of the mouse (or keypad).*

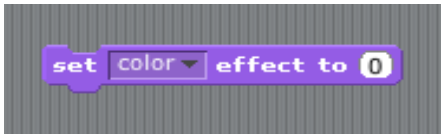
Select a *Sprite*, best one with a multi-colour body
e.g.

Go to the *Script* section of the Sprite

Type in the Commands into the Script section as shown on the screen below:



The block below is located in the Looks folder



Goto to the Sensing folder to find the *Mouse X* and *Mouse Y* 

blocks which you drop onto the Set commands as show in the piece of code above.

Start the programme by clicking the Green flag on the top hand corner of the screen.

Because of the instructions that you have typed in, moving the mouse left and right along the X axis, the Sprite will change shape.
If you move the mouse long the Y axis, the Sprite will change colour.

Exercise
Experiment with other effects by clicking the black arrow in the '*set... effect to* and choose other options besides *Whirl* and *Colour*.

Lesson 9 - Dancing Sprite

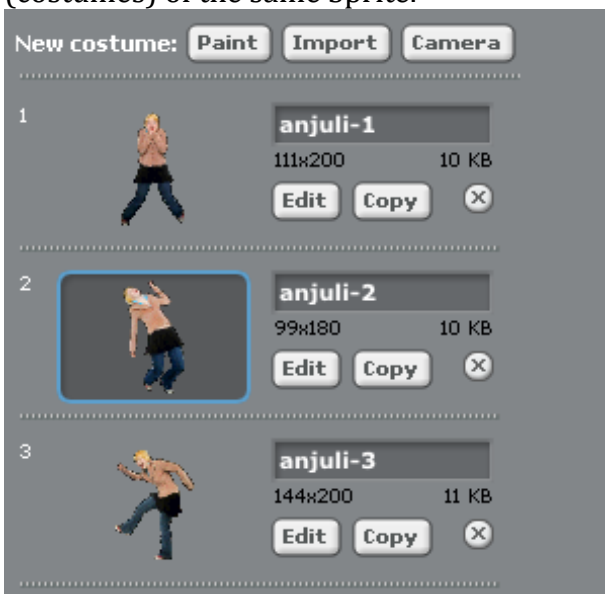
In this lesson, students import a new background and learn how to animate your sprite and create a music loop

First, click on the *Stage* icon located under the Stage section of the Scratch screen.

Go to *Import* and choose an appropriate *Background* for the dancing sequence you are about to make.

Then choose a new Sprite that has a choice of different positions.

Next, go to the *Costume* section and import a number of different versions (costumes) of the same Sprite.



Now let's animate!

Go to *Script* box and select from the Control folder:

Followed by the *Forever* command
*(This command will let the dance programme run continuously until you hit the red circle **Stop** icon located at the top right side of the Stage screen)*

Next select the *Wait* command which you should reduce in duration from 1 second (secs) to 0.5 seconds as otherwise the Sprite will be dancing too slowly.

Go to *Looks* folder, select *Next Costume* and place within the *Forever* command block.

Run the program

In order to add sound, go to *Sound* folder, click on Sounds box and select *Hip-Hop* in Music Loops.

In the Sprite section, add the following:

Run the program.

Exercise
Start a new programme with a different background, different music and three dancing sprites.
Draw a few Irish traditional dancers in different poses, locate some appropriate music and make your very own Riverdance scene!

Lesson 10 - Drawing Shapes

In this lesson students learn how to draw different geometric shapes using a series of different scripts.

First we are going to create a Script to draw a **Square**

This time we do not need a Sprite. But as all of the Starch commands cannot function without a Sprite, we need to hide it.

So use the following commands in the Script to make the Sprite disappear:

In order to draw in Scratch, we have to go to the *Pen* folder where we select a pen size and colour.

We also have to give instructions for the pen a) to start 

and b) to finish 

drawing.
It is also important to clear previous drawings from the stage area and to recommence the drawing process on a blank canvas one the Green Flag is selected.

So use the command 

Adding on '*wait 1 secs*' block will allow the viewer to appreciate better both the commencement and the actual drawing of a new object.

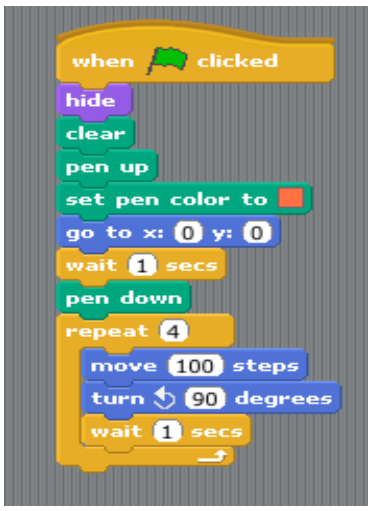
To ensure that the Square is of sufficient dimensions for the viewer, use a sizeable number of *steps* from the Motion folder e.g. 100 steps.

So how do we code in commands in order to make the four lines form a box?

First we use the *Repeat* command
So, in the example of the Square it is **Repeat 4** times

Please note also that for each geometric shape, the angle is proportionate to the number of sides i.e. 360 degrees divided by the number of sides.
For instance, a square is 360 divided by four = 90;
a triangle is 360 divided by three = 120 and
a circle is 360 degrees divided by 360 = 1

So we have to *turn* the *lines* using the *degree* option in the Motion commands



We can also change the colours for each drawing by picking the random option in the *Operators* Command folder. The spectrum of colours go from 1 to 360. Hence choose a high random range e.g. 1-200 (See image below) for colour variation.

Now how do you draw a **triangle**?

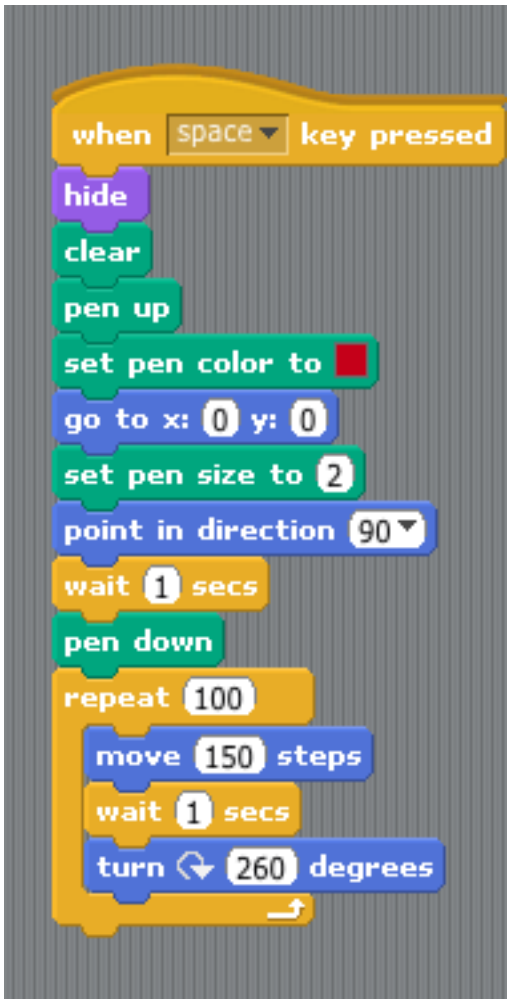
How many sides in a triangle?
What would the angle (degree) settings be?

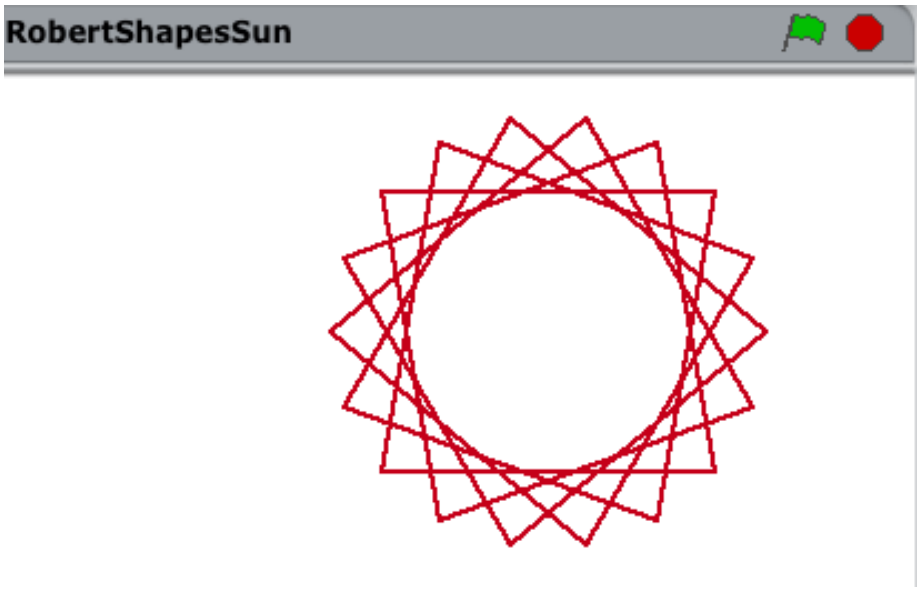
How do you form a **circle**?
How many turns (degrees) in a circle?
So turn *one degree* at a time. Do not use *'Wait 1 secs'* block as it will take too long to form the circular shape.

Exercise

1. Draw a *Pentagon*
2. Write a programme that draws three different shapes that appear at different times at different locations on the Stage.

Experiment with different angles and 'repeats' in the programme.
For instance, select Robert's code below and admire the results

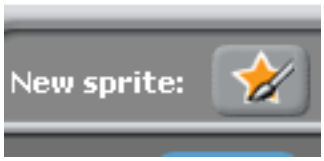




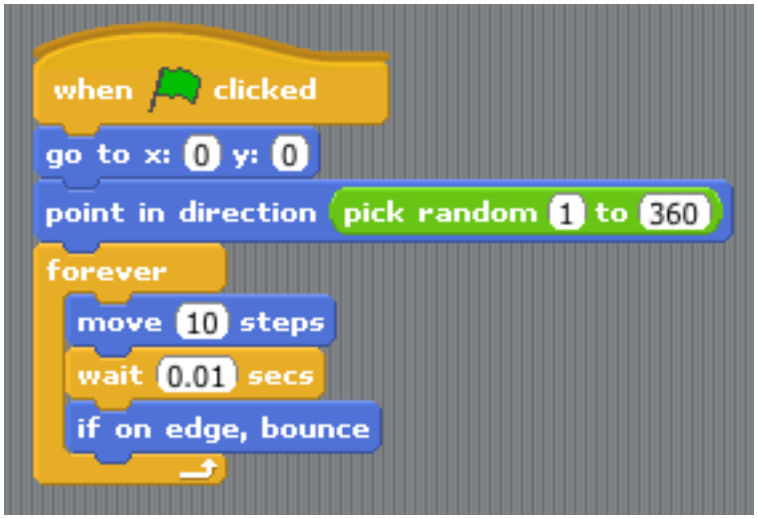
Lesson 11 - Bouncing Ball

In this lesson, students become familiar with ***Degrees*** *and* ***Variables of Space.***

First draw a Ball using the *Paint New Sprite* option

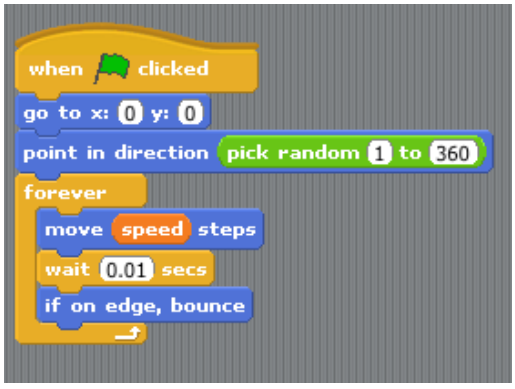


Then start writing a Script as follows:



Experiment with the speed of the ball by *increasing* and *decreasing* the *number* in the steps box

We can also control the speed of the ball during a game by using the **Variable Control Box.**



Select *Make a Variable* and Type in Speed in the option *Variable Name*

A *Speed icon* appears on the Stage

Left click on the icon and select *Slider* option which allows the user to adjust the speed of the ball when in motion.

Generating a *Comet* tail on the Bouncing Ball

Exercise

1. Write a programme with a number of bouncing balls
2. Write a marine-themed programme that has a number of different bouncing sea creatures such as jellyfish and sharks.

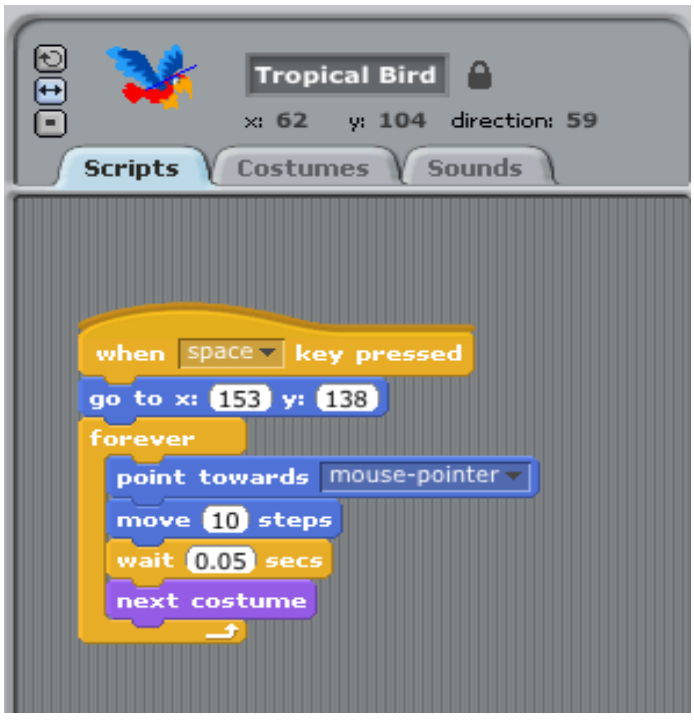
Lesson 12 – Cursor-controlled Sprites

In this lesson, students become familiar with programming Sprites to follow the movements of the cursor

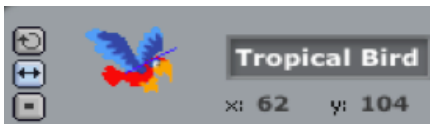
Tropical Bird on the Beach

Select a beach scene and a two costume sprite from the Scratch folder libraries

*Utilise the **Mouse Tracker** option located in the Motion box command **Point towards** and write up the following script:*



To stop the bird flying upside down, use the middle option in the **Rotation** section:



Exercise:
Draw your own flying *Sprite* with two costume changes as well as your own *Background*,
Create a programme with the Sprite controlled by the *Mouse-pointer* code.

Lesson 13 - Building A Game

In this lesson, students learn how to build their own game based around two Sprites

Demon Chaser

First students should be informed of the importance of **planning** out their proposed game with pencil/pen and paper**.**
They should understand the need to define the purpose or the game, how this is to be achieved and the role of each of the game elements such as the different Sprites and the content**.**
The written design is then ßused to build the game

However the tutor should first provide an example of a game that could then be used as a starting point or template for the students' own creations.

Sample Game **Guideline** of ***'Demon Chaser'***

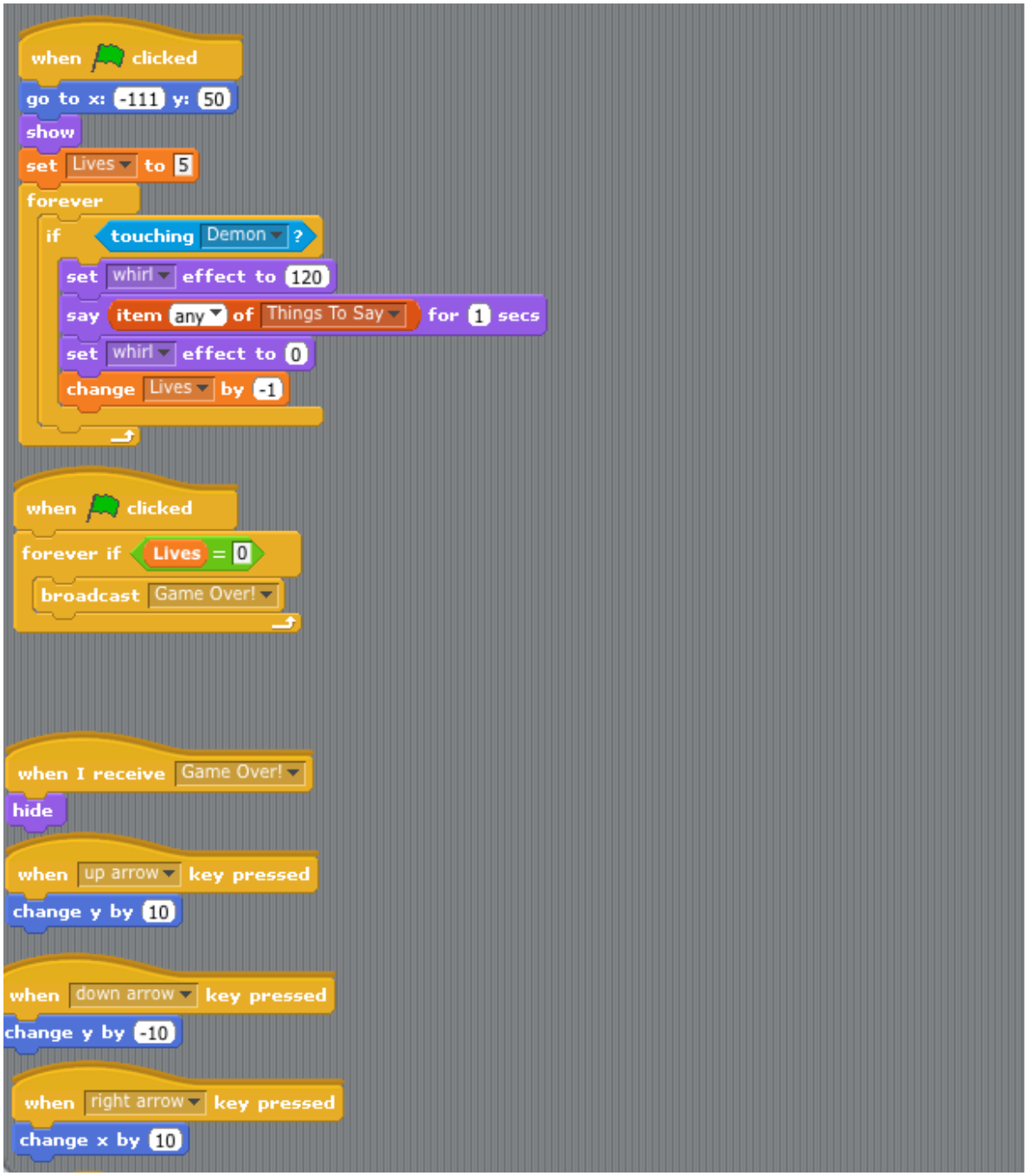
- *How many characters (Two Sprites to start with)*
- *Role of characters (Good Sprite, Bad Demon)*
- *Operations of characters (one sprite's movement controlled by the four 'arrow keys'; the other character to move randomly)*
- *What background should I use?*

Game Play

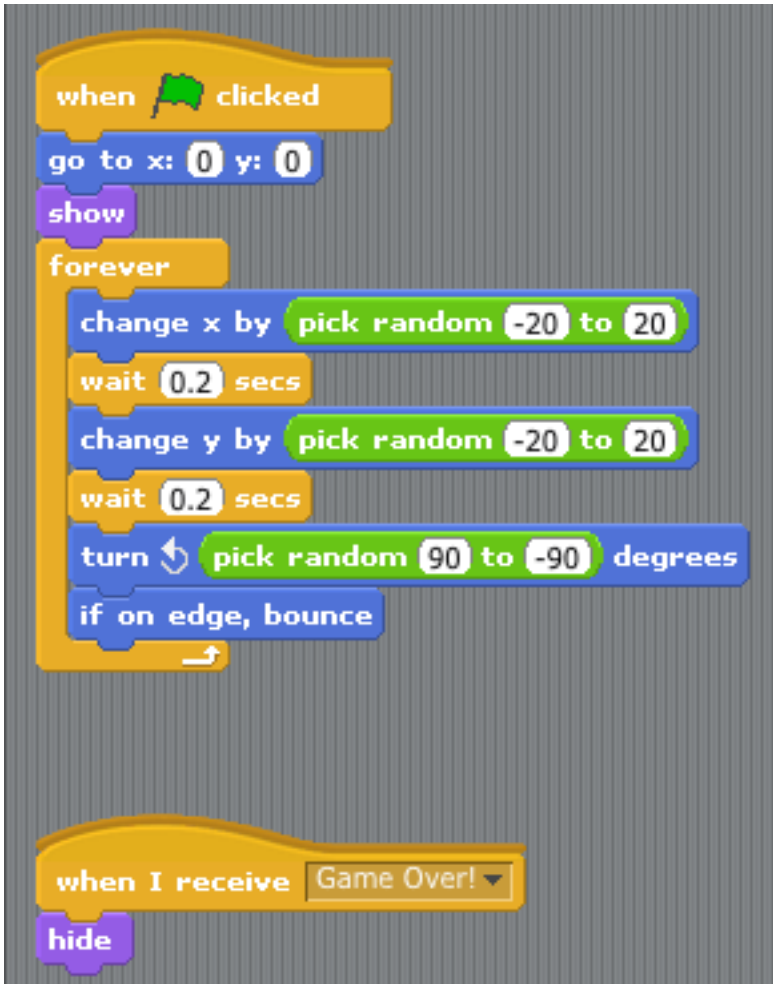
- *Good Sprite moves by arrow keys but has to keep moving*
- *Demon moves randomly so that the Good Sprite does not know where he will appear*
- *The Good Sprite Starts with 5 lives*
- *If the Demon touches the Good Sprite , the Good Sprite loses a life*
- *If the Good Sprite lives become equal to Zero, then the game stops and a Game Over Background appears*
- *When the Game restarts the background is reset and the lives are initialised back to 5*

***Improvements**: After completion of the above guidelines, get the students to make some adjustments that they feel might improve the game*
Explain the content of the Code for all 3 elements, namely *Sprite 1, Demon* and *Background.*
Particularly important are the *Sensing, Variable* and *Background* folders

Code (Instructions) For Sprite 1



Code for Demon



Code for Background

Exercise

Get the students to create their own game for the next class using sprites drawn by themselves.

Each group will explain and demonstrate their own project creations to the full class.