Step by Step Part 1:

The Scratch window/ coding environment

These are all the types of 'code' you can use

Read all of these speech bubbles - they contain helpful hints for you

This is the 'stage' where your game will be played

Click the green flag to run your code

This is a 'sprite' or character in your game. Sprites wear 'costumes'

Drag the code blocks you want into this space. When you run the code, your sprites follow your instructions exactly

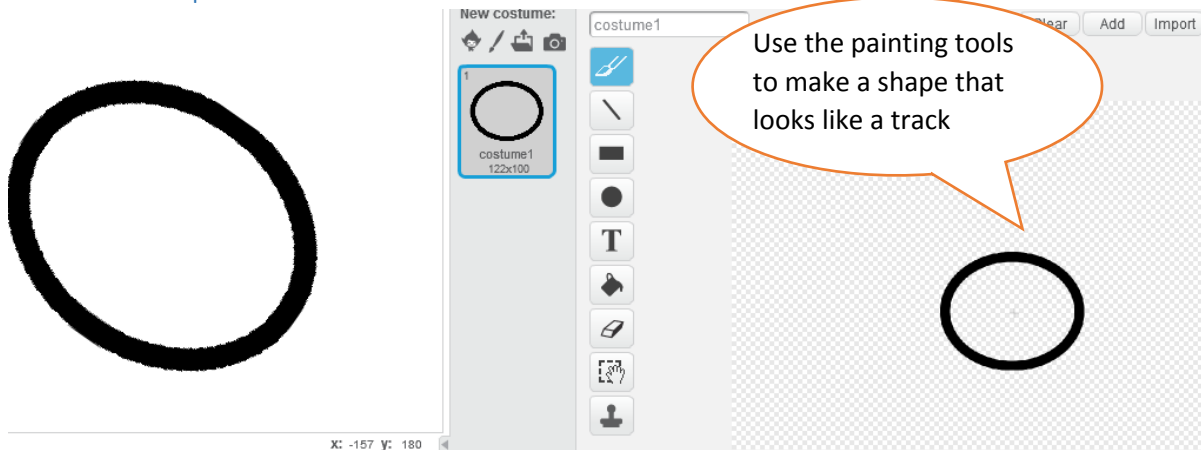You can add or paint a new sprite from here

Make a sprite move and bounce off the edges of the stage

Use your mouse to find and drag blocks with the right colours to add this code for Sprite1. Then run the code using the green flag
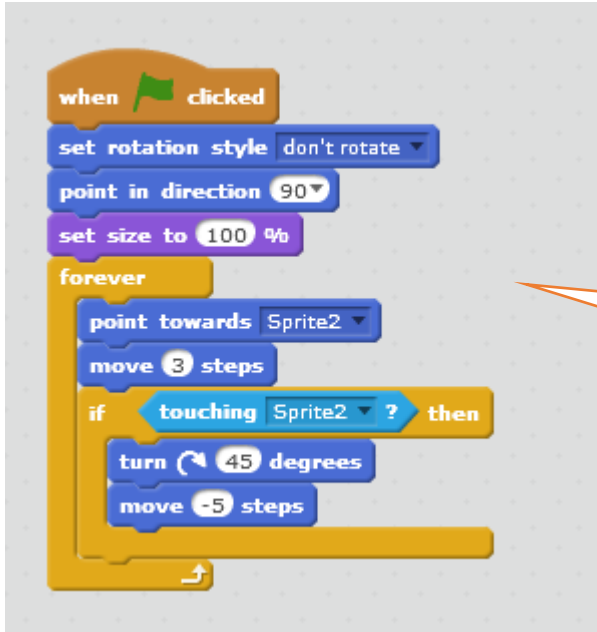
Paint a new sprite

Use the painting tools to make a shape that looks like a track

Turn the page over >>>

Change the code for Sprite1 (the cat) so that it follows the track

```
when [green flag] clicked
set rotation style [don't rotate ▼]
point in direction (90▼)
set size to (100) %
forever
    point towards [Sprite2 ▼]
    move (3) steps
    if < touching [Sprite2 ▼] ? > then
        turn ↻ (45) degrees
        move (-5) steps
```

To remove old code you don't want, pull the block out using your mouse. Then click with your right mouse button to delete it fully

As you drag in the new code, look at all the other options available in Scratch for code blocks you can use

Press the green flag when finished, to run your code

Save your project in a folder

Use File>Save from the Scratch menu, just above the stage. Pick a folder (e.g. your desktop) and give your file a name

Enjoy it for a few minutes, then move on

Watch the animation, show it to someone, or playa round with it for a few minutes. Then you're ready for your next project.

Grab the next handout, which helps you to build a game where a cat plays fetch with a ball thrown by a friendly alien.
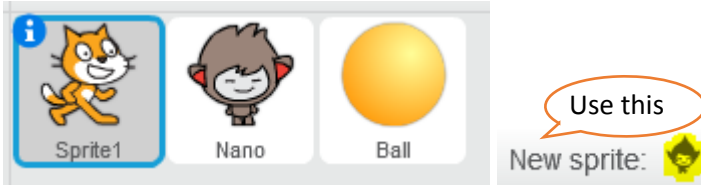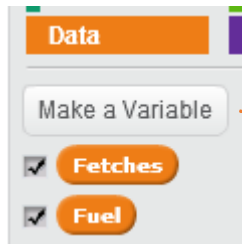
Fuel [66]

Fetches [1]
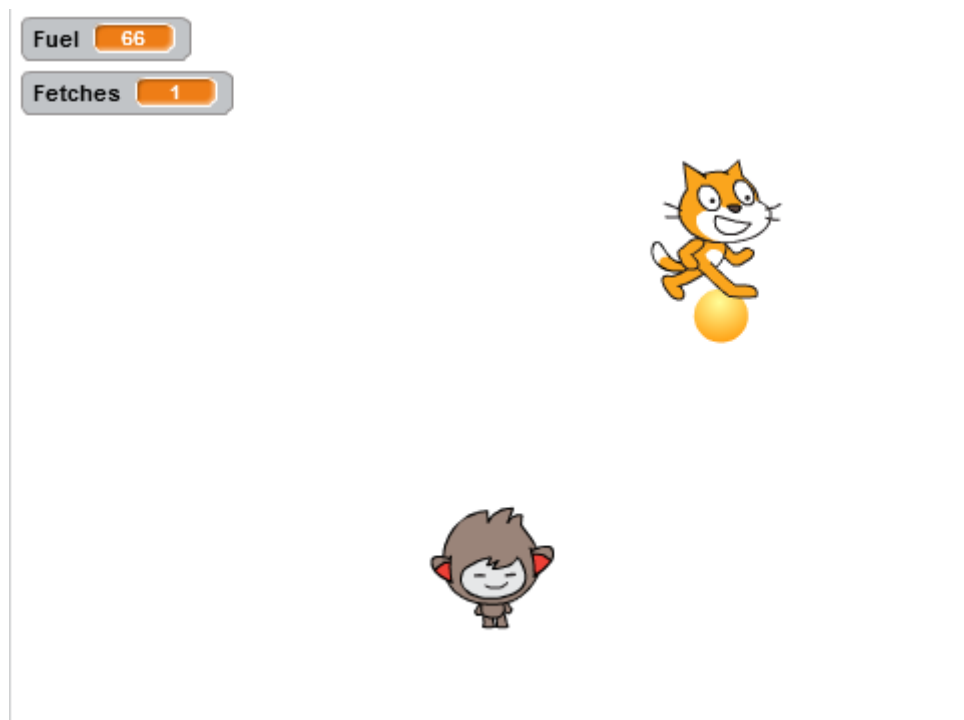
Step by Step Part 2:

Start a new project, in which a cat plays fetch with a ball thrown by a friendly alien.

Use File>New

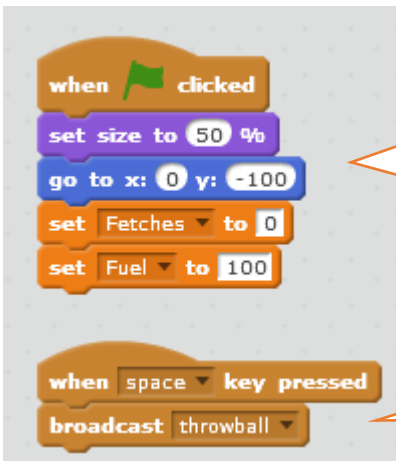### Add new sprites from the Library

Use this

New sprite:

Get ready to track the score in your game

Create two 'variables' that will store the score and the amount of fuel remaining before its 'game over'

Add these code blocks for Nano, so that he can send the command 'throw ball'

This block gets the game started correctly. Coders call this 'initializing'. It's a really good idea

This code listens for the space key to be pressed by you, then sends a command to all sprites

Add these code blocks for Ball, so that it responds to the throw ball command

This code makes it look like Nano throws the ball

This code makes the direction of the ball a surprise every time – a 'random number'

Turn the page over >>>

```
forever
  if  touching Sprite1 ? then
    say Well done, Spot! for 0.5 secs
    change Fetches by 1
    stop all
  else
    move 5 steps
    change Fuel by -1
    if on edge, bounce

  if  Fuel < 0  then
    set Fuel to 0
    say Game Over for 2 secs
    stop all
```
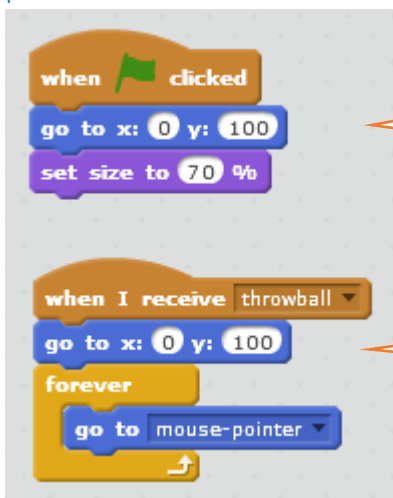
This code checks to see if Sprite1, the cat, has caught the ball. Every 'fetch' increases the score

Otherwise, the Cat's fuel drops. If the ball hits the edge of the screen, it bounces back

If the fuel level gets too low (<0), it's game over

Add these code blocks for the cat, Sprite1, so that he can fetch the ball by following your mouse pointer

```
when [flag] clicked
go to x: 0 y: 100
set size to 70 %


when I receive throwball
go to x: 0 y: 100
forever
  go to mouse-pointer
```

This code sets up the cat correctly at the start of the game ('initializing')

This code makes the cat follow your mouse pointer

Play the game and see how many fetches you can get before your fuel runs out.

Press the green flag, then hit the spacebar to throw and move your mouse to fetch.

Save the game

You already know how to do that.

Change the game

Try pulling out some code blocks to see what effect that has on the game.

Change the game to make it more fun to play – e.g. make it harder to score, make the ball move faster and come up with your own ideas.

Move onto another project when you are ready (grab a handout or start your own project).
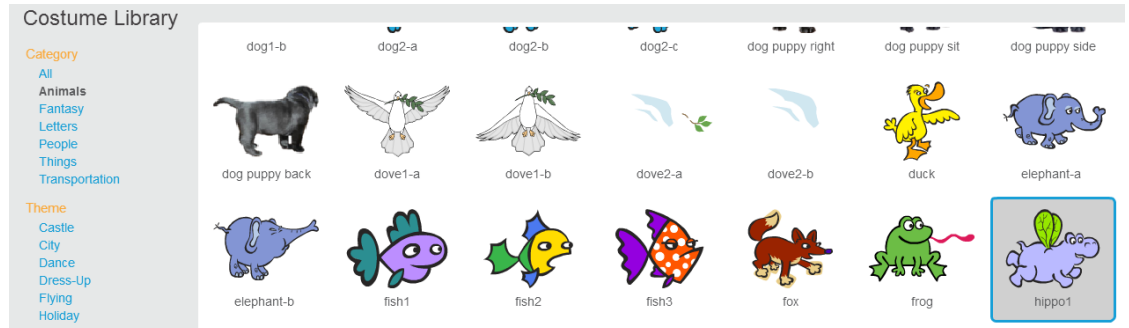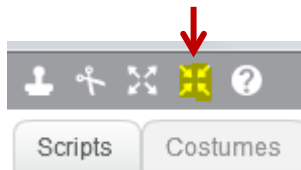
Hippo Climbdown

In this **advanced** project, you will make a platform game, in which the sprite can move around, jump and climb, in a way that looks like what happens in the real world, especially including 'gravity'.
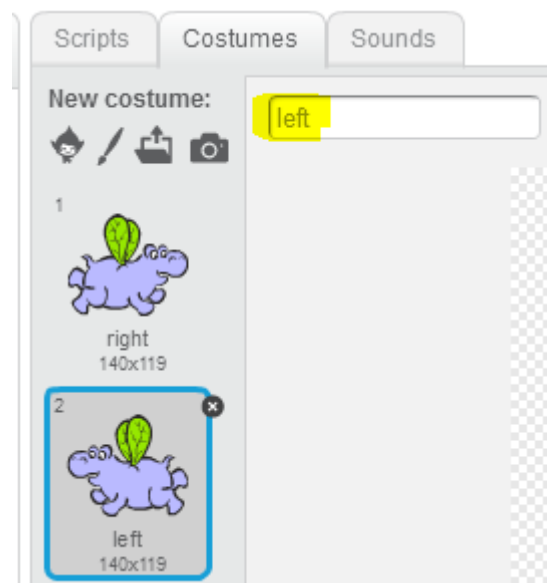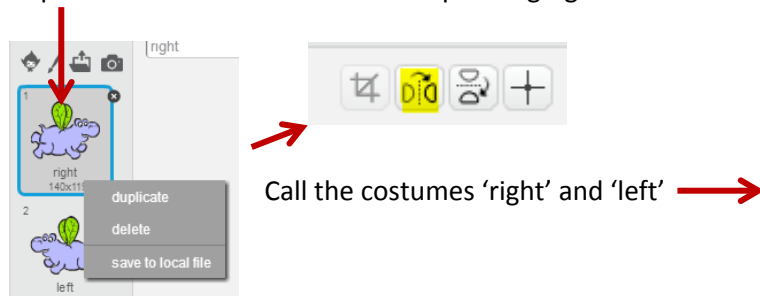
Get a costume from the library: I used a hippo
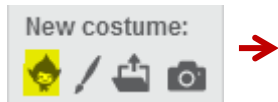
I made the hippo a bit smaller with this tool:

Duplicate the costume into versions pointing right and left

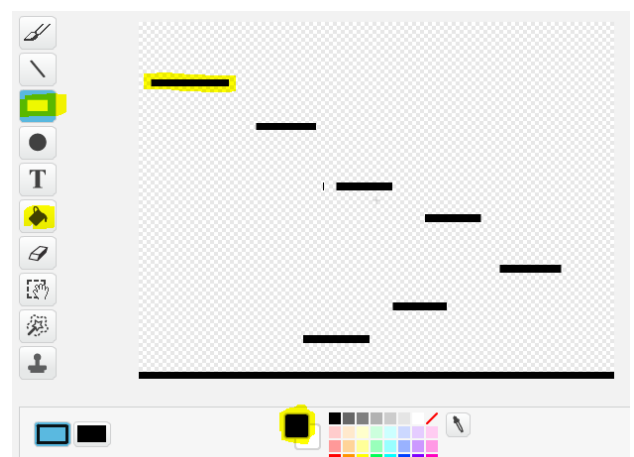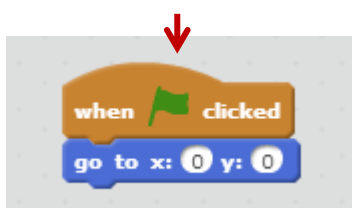Call the costumes 'right' and 'left'

Paint a new sprite:

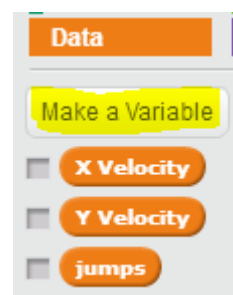This sprite should take up the full size of the stage.

Draw some rectangular 'steps' coloured black at various places on the screen, and a long thin rectangle right at the bottom that represents the ground. e.g.
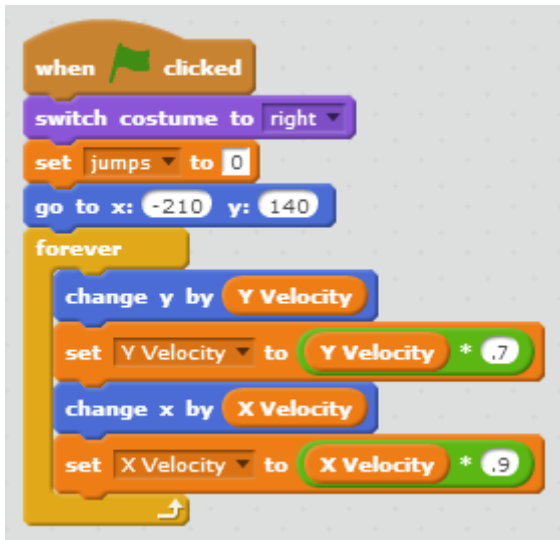
Use this code ('Script') to get the new sprite to always be in the right position on the stage:

Set up three 'Data' variables like

this:

Enter this code for your first sprite (not the steps):

Enter this code so the first sprite can move left and right:

```
when [green flag] clicked
switch costume to right
set jumps to 0
go to x: -210 y: 140
forever
    change y by Y Velocity
    set Y Velocity to (Y Velocity * .7)
    change x by X Velocity
    set X Velocity to (X Velocity * .9)
```

```
when [left arrow] key pressed
switch costume to left
set X Velocity to -5
```

```
when [right arrow] key pressed
switch costume to right
set X Velocity to 5
```
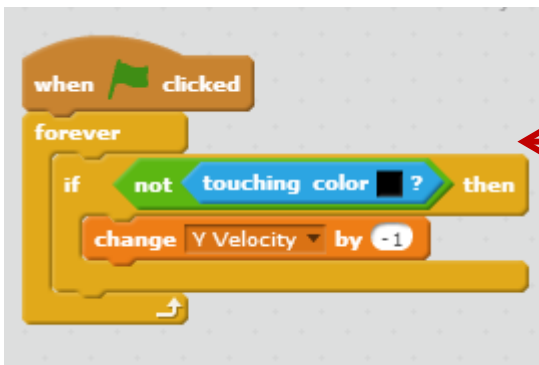
Now play the game.

You should see that your sprite can move when you press the left and right arrow keys.
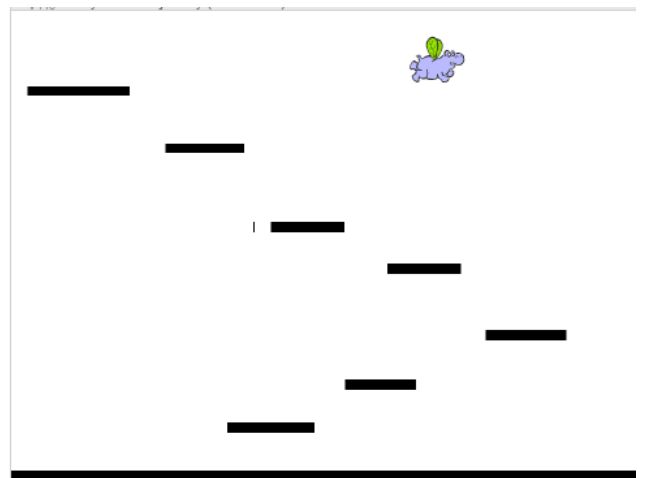
He slows down a little while after you press the keys – that's like friction or air resistance in the real world.

He does not fall through the air yet, just sits there: that is not like the real world.

To make him 'fall' properly, you need to add gravity to the game:

```
when [green flag] clicked
forever
    if < not <touching color [black] ?> > then
        change Y Velocity by -1
```

He doesn't quite land correctly on each step, until you add:

```
when [green flag] clicked
forever
    if <touching color [black] ?> then
        set Y Velocity to 0
```

Most video games that look like this involve climbing up as well as climbing down. To enable the hippo to jump, add:

```
when [green flag] clicked
forever
    if <key up arrow pressed?> then
        set Y Velocity to 7
```
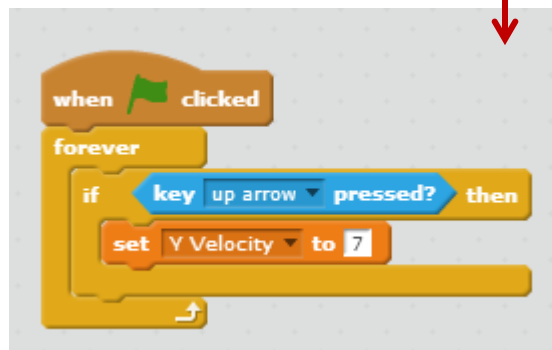
Now you should be able to both descend and ascend using the arrow keys.

Extra:

Change the numbers for jumping, falling down (gravity) and slowing down to see how they affect the game.

Sometimes the top of the hippo sticks to one of the steps above him. How could you fix the game so that doesn't happen?

Improve the game with your own ideas. Show your finished game to the kids near you and to a mentor. Upload your game to the web (username: coderdojobray; password: brayhead).

Muffin shower: control game through webcam

In this **<u>advanced</u>** project, you will make a game that uses your computer's webcam to see whether YOU are moving. If you move enough, you get a high score.

Change the costume of a sprite to a muffin.

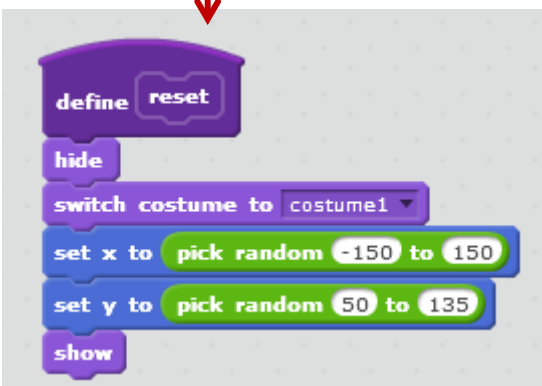Make a second costume for that sprite that reads "+1"
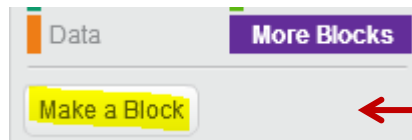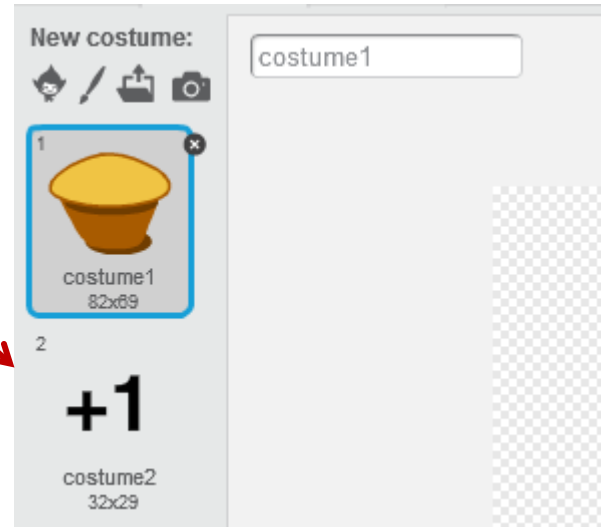
Make your own Scratch 'block' to make the muffin appear at random locations on your screen: blocks like this are re-usable and save you duplicating your code. They also make your code look neater and easier to understand.

Call the block 'reset':

Then add this code to the block:

When finished, test the new block by double-clicking it with your left mouse button. You should see a muffin appearing on the stage each time. That's what this block does.

Add a variable to keep track of the score:
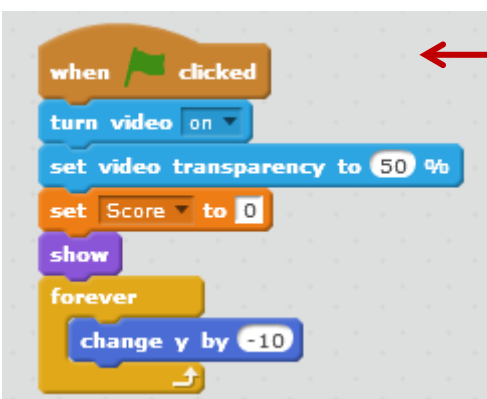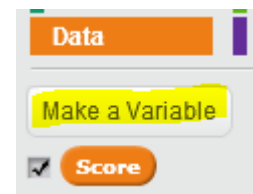
Add this code to turn on your webcam so that you can use your movements (e.g. your head and hands) to interact with the game:

When you press the green flag now, you should see your own reflection on the screen, a bit hazy: (the muffin will fall to the bottom of the screen)
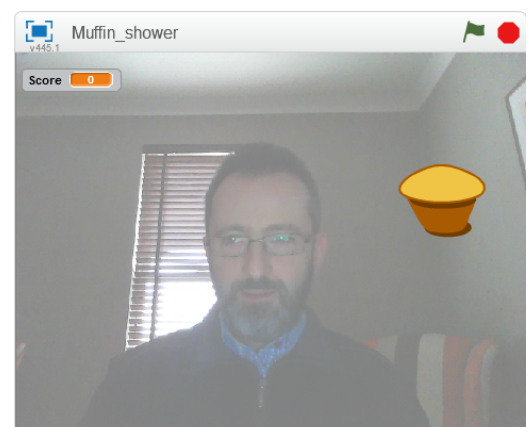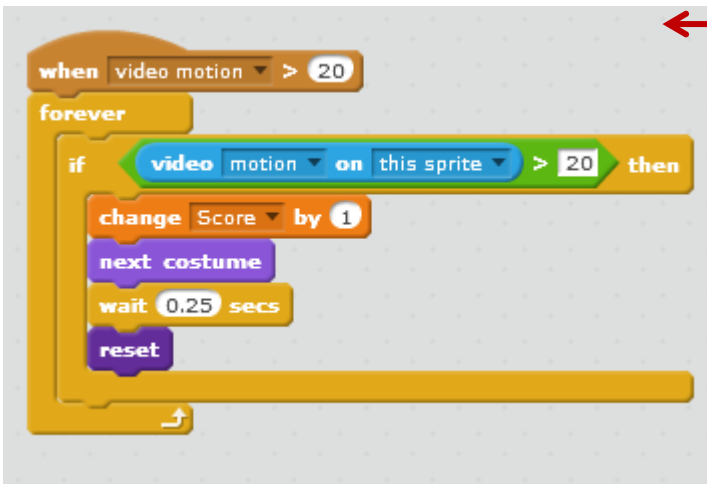
Now add this code to check for motion on your webcam:

When motion occurs (e.g. you move), the game will run 'reset' and muffins will fall from the top of the screen.

To get a muffin, point your finger or your hand at it, so that your movement appears on screen, and keep the movement going until you get a score. Then another muffin will appear:
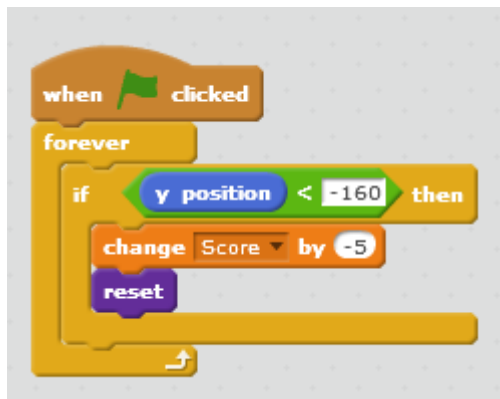
This code penalizes you if any muffins make it to the bottom of the screen; it also makes those 'lost muffins' disappear and a new one falls:

Extra:

Experiment with the numbers used to detect motion.

Try standing back from your computer and using motion with your arms to get muffins. Make sure you do this safely and nobody gets hurt!

Now use your own ideas to make the game better.

Show your finished game to the kids near you and to a mentor.

Upload your game to the web (username: coderdojobray; password: brayhead).

To print more copies, use the Dropbox folder link at http://scratch.coderdojobray.com

Use File>Save to save your work:

CoderDojo Bray – Advanced: Start making a game with a scrolling background

This sets you up to finish a scrolling game in your second session.

Make three **sprites** as shown: the first two are full-size background pictures that will move. The **costumes** for these should be the same. You can use **Import** in the costume painting screen to grab a saved backgr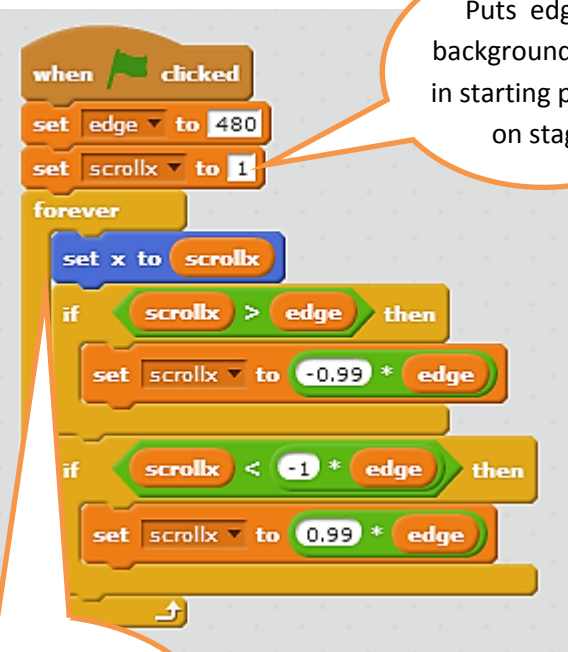ound image that will work for these. (You can create that image using **New backdrop** for the **stage**, then **saving that to a local file**). When you have finished this page, the cat should be able to walk forever in both directions.

Both background sprites have width 480

Code for the sprites: For **Sprite 1**:

Puts edge of background image in starting position on stage

```
when [flag] clicked
set edge to 480
set scrollx to 1
forever
  set x to scrollx
  if scrollx > edge then
    set scrollx to -0.99 * edge
  if scrollx < -1 * edge then
    set scrollx to 0.99 * edge
```

When the cat code changes scrollx, the background sprite moves

For **Sprite 2**:

```
when [flag] clicked
forever
  if scrollx < 0 then
    set x to edge + scrollx
  if scrollx > 0 then
    set x to scrollx - edge
```

This sprite fills the space left when the other sprite moves

For the **Cat**:

```
when left arrow key pressed
switch costume to cat1-a2
change scrollx by 10

when right arrow key pressed
switch costume to cat1-a
change scrollx by -10
```

When the cat code changes scrollx, the background sprite moves

CoderDojo Bray – Advanced - Finish a game with a scrolling background, part 2:

This is where your scrolling game is heading. Make the scrolling background (part 1) first. Then do this part. Apples fall randomly from the sky. Your character needs to collect green apples and avoid red ones. Play it when finished. Then improve it!

Make two additional apple sprites, one green apple and one red.

Add some variables:

Add more code for the Cat:

We want this sprite to be visible at all times, not hidden by others

Small apples work best

Fuel keeps the cat moving

Apples fall slowly at the start

It's game over when the fuel is gone

To enter code for a Sprite, make sure you have clicked it here

Collect

Fuel 3   Score 0

Get Green Apples only

X: 188 Y: 180

Sprites   New sprite:

Stage 1 backdrop   Sprite1   Sprite2   Cat1   Red   Green

Data

Make a Variable

☑ Fuel
☑ Score
☐ apple2x
☐ applex
☐ edge
☐ scrollx
☐ speed

when 🚩 clicked
go to front
say Get Green Apples only for 2 secs

when 🚩 clicked
set Fuel to 3
set Score to 0
set speed to 5
forever
  if Fuel < 0 then
    say Game Over for 1 secs
    set speed to 0
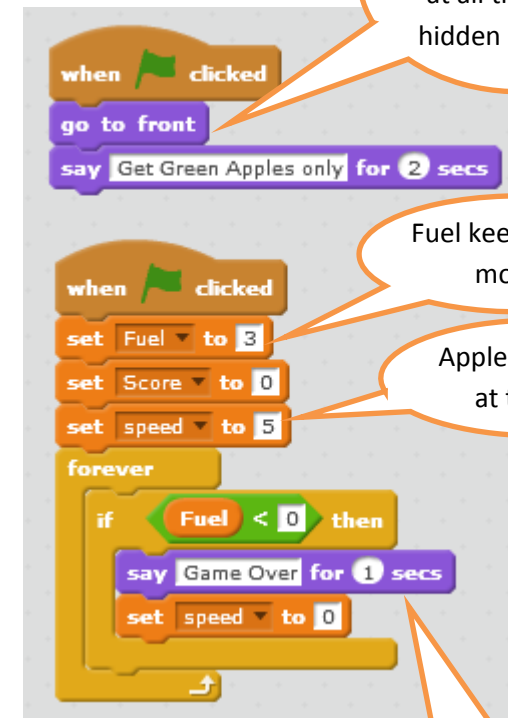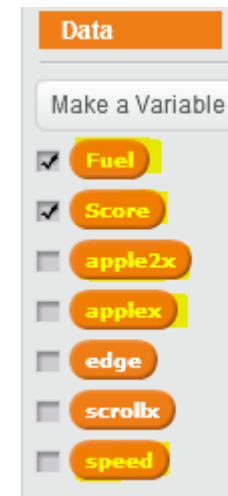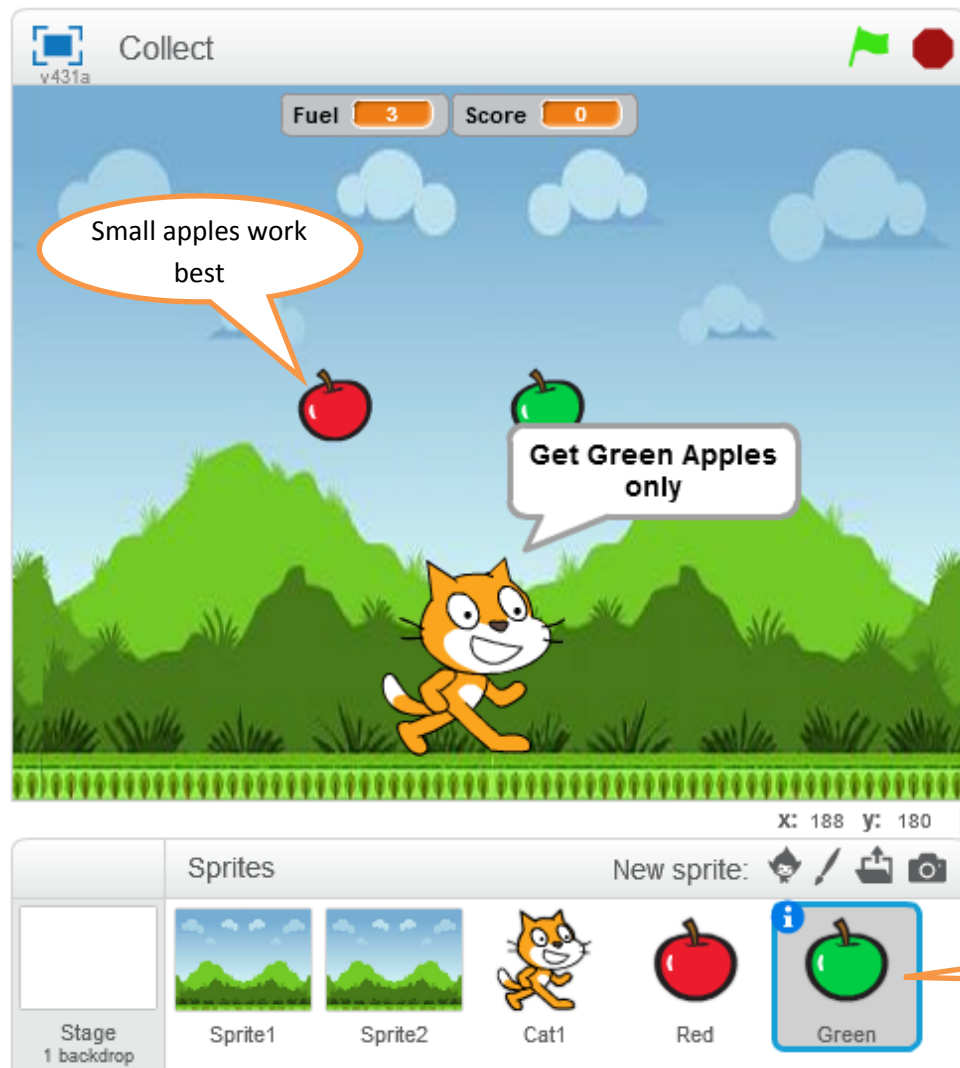
This script runs when it receives a message

Add code for Red Apple:

And for Green Apple: (similar, minor changes)

Apples fall from random x positions

Applex stores the initial x position

Apple2x stores the initial x position of the green apple

```
when I receive RedApple
hide
go to x: pick random -240 to 240 y: 180
set applex to x position
show
forever
    change y by -1 * speed
    set x to applex + scrollx
    if y position < -150 then
        hide
        broadcast RedApple
    if x position > 240 then
        hide
        broadcast RedApple
    if x position < -240 then
        hide
        broadcast RedApple
    if touching Cat1 ? then
        change Fuel by -2
        change speed by 2
        hide
        broadcast RedApple
```

```
when [flag] clicked
go to front
broadcast RedApple
```

```
when I receive GreenApple
hide
go to x: pick random -240 to 240 y: 180
set apple2x to x position
show
forever
    change y by -1 * speed
    set x to apple2x + scrollx
    if y position < -150 then
        hide
        broadcast GreenApple
    if x position > 240 then
        hide
        broadcast GreenApple
    if x position < -240 then
        hide
        broadcast GreenApple
    if touching Cat1 ? then
        change Fuel by 1
        change Score by 1
        hide
        broadcast GreenApple
```

```
when [flag] clicked
go to front
broadcast GreenApple
```
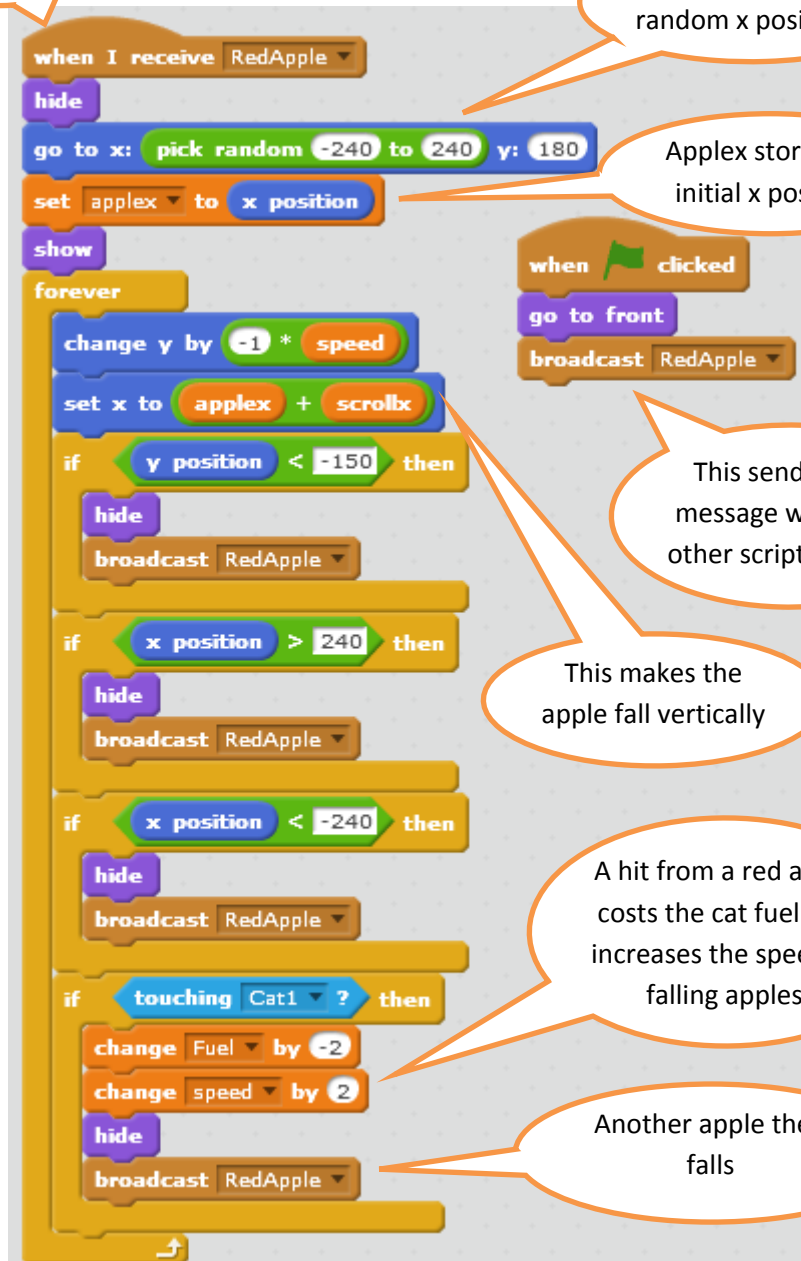
This sends out a message which the other script receives

This makes the apple fall vertically

A hit from a red apple costs the cat fuel and increases the speed of falling apples

Another apple then falls

A hit from a green apple boosts the cat's fuel and score
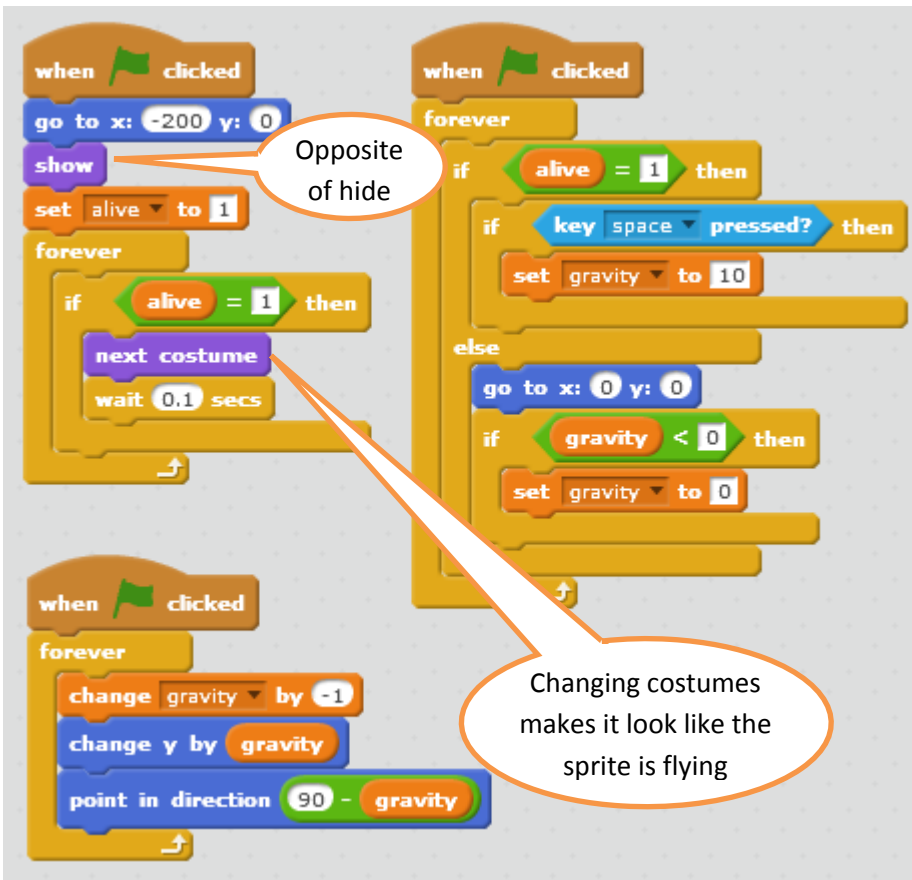
These speech bubbles contain helpful hints for you.

To print more copies, use the Dropbox folder link at http://scratch.coderdojobray.com

CoderDojo Bray 2015: Flappy Bird in Scratch

To play the game you use the space bar to make Flappy gain height. You must get him to fly through the gaps in the pillars. The highest score wins.

Create 2 sprites:

- **Flappy** - a bird with flapping wings. Use the butterfly available in Scratch if you like. Use two costumes for flapping wings.
- **Pillar** – Draw this and make several copies as costumes (e.g. 5), then use the eraser to make one space in each pillar.
- Enter the code below for Flappy (left) and the pillars (right); add the variables too. Play. Improve the game.

Like this

Add these variables:

Opposite of hide

Ticking box shows variable on stage

Changing costumes makes it look like the sprite is flying

Detects collision

Left edge of stage

The next pillar shape is a random choice