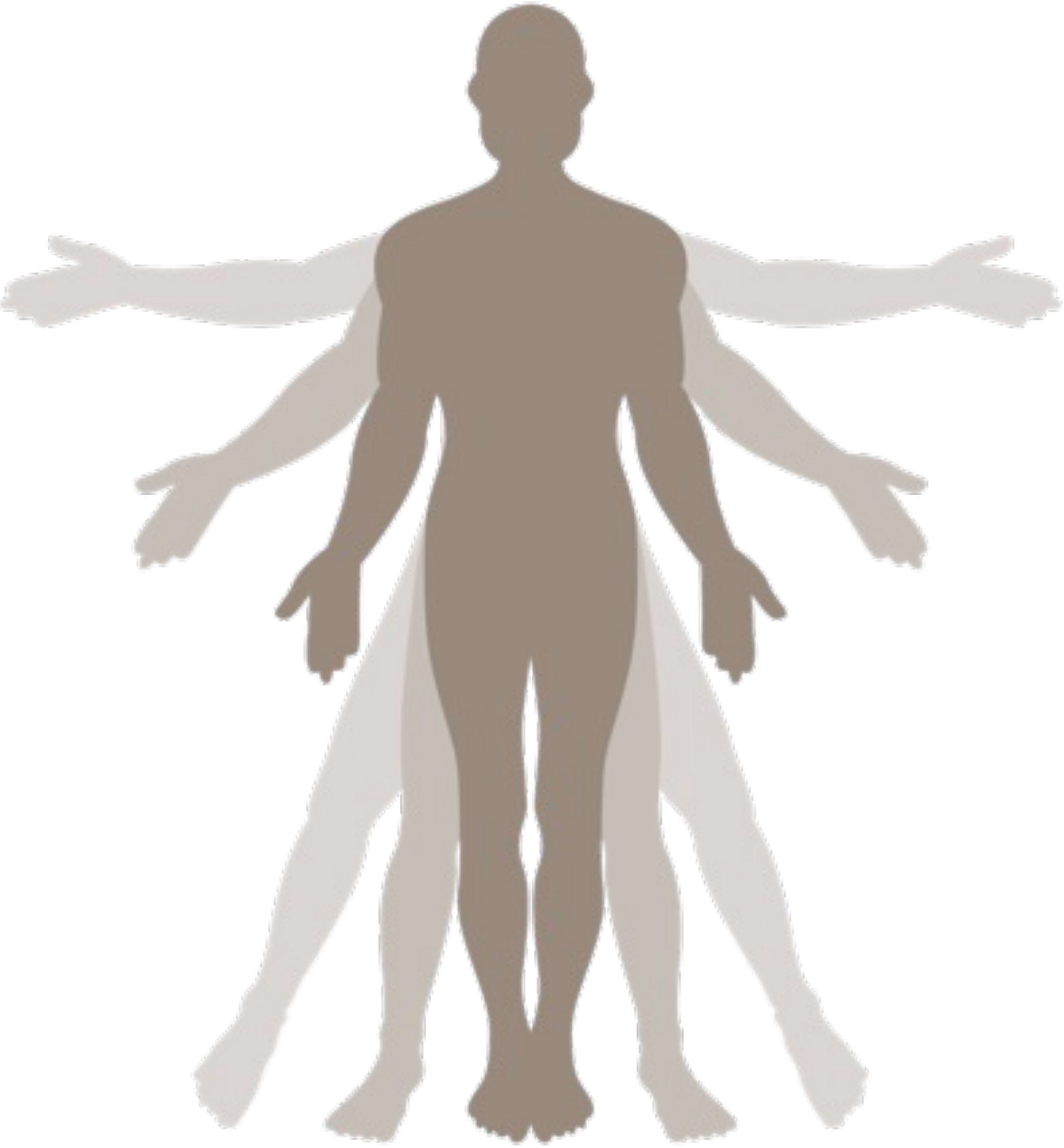
technocamps

Inspiring | Creative | Fun

Ysbrydoledig | Creadigol | Hwyl

# Kinect2Scratch Workbook



i.t.wales www.itwales.com

Swansea University Prifysgol Abertawe

Prifysgol Aberystwyth University

Prifysgol Bangor University

University of South Wales Prifysgol De Cymru

ESF

Llywodraeth Cymru Welsh Government

Ewrop & Chymru: Buddsoddi yn eich dyfodol

Cronfa Gymdeithasol Ewrop

Europe & Wales: Investing in your future

European Social Fund

# Kinect2Scratch

Scratch is a "drag and drop" programming environment created by MIT. It contains colour co-ordinated code blocks that allow a user to build up instructions for a game or animation, without being concerned with any confusing syntax or complicated error output. Scratch is a perfect environment for beginners.

When initially using Scratch to create games or animations, controls such as the keyboard arrow keys or mouse pointer movements are implemented to control the movement of Sprites - however there is alternative hardware to use, such as the Xbox Kinect visual sensor.

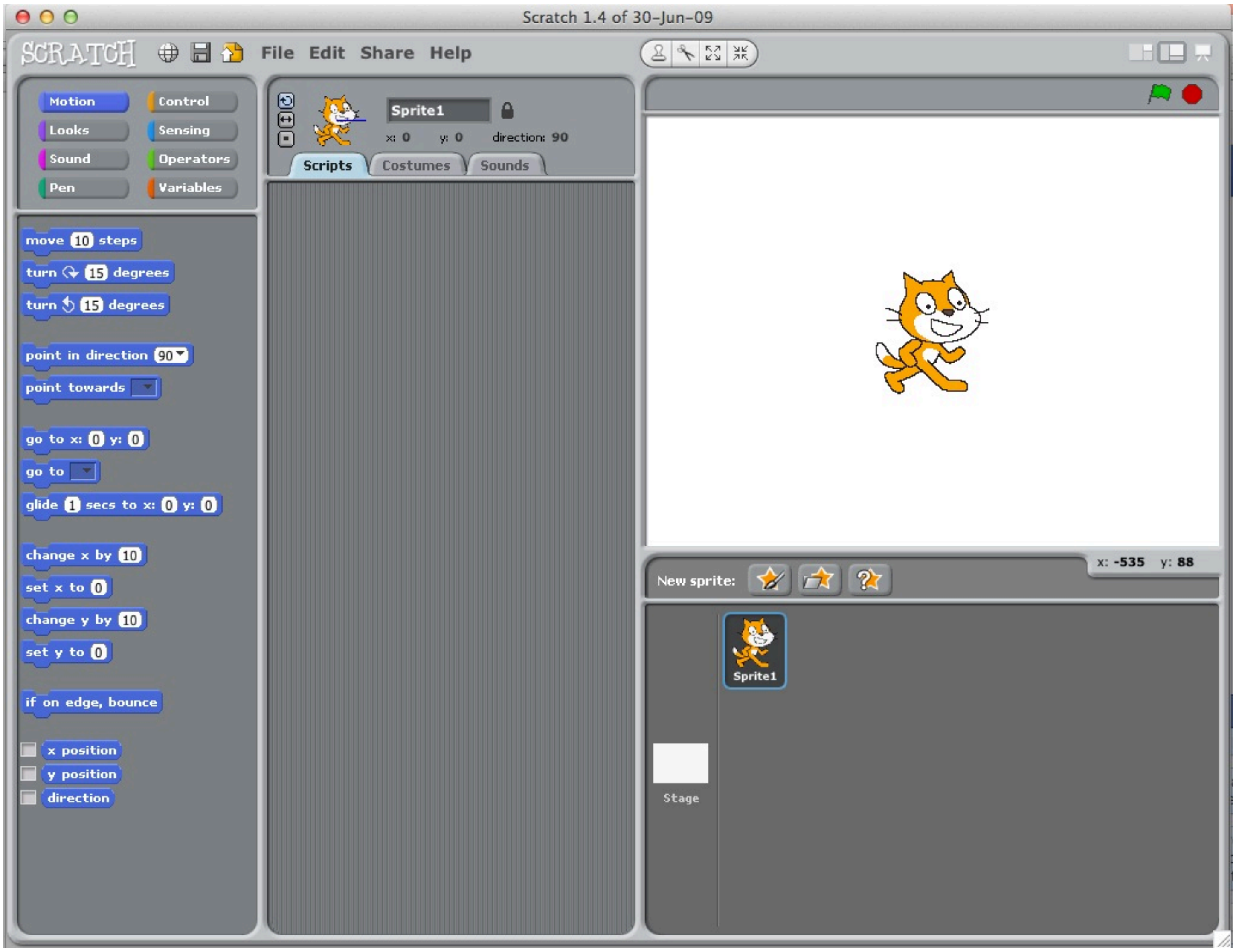
Kinect2Scratch is an application that allows Scratch to utilise the Microsoft Kinect as visual input, a Scratch project can then be controlled using a person's motion. The Microsoft Kinect sensor is a horizontal piece of hardware that uses web-cam style technology to enable users to control and interact with the Xbox console, or in this case the Scratch interface on a computer. This motion controller can be implemented replacing the need for a keyboard or mouse.

## 1) Setting up Kinect2Scratch

- Plug in the Kinect sensor to the mains and to the Laptop/PC via USB connection and turn on.
- Download Kinect2Scratch from http://scratch.saorog.com/blog.php/index.php/download-kinect2scratch/.
- Download the Microsoft Kinect SDK.
- Open Kinect2Scratch and press "Launch Kinect". You will see 2 video boxes that appear, on the left the red version shows your body has been detected.
- Press "Connect to Scratch", open Scratch up on your Computer.
- Go on the "Sensing" category, find the code that says "*Slider* sensor value". Right click and select "enable remote sensor connections". Click on "slider" on this code block. A drop down menu should appear with the new Kinect components added.

# Kinect2Scratch

A reminder of the Scratch interface, see the image and description below:



Scratch has 5 main sections, firstly starting from the top left you have a list of categories to choose from. As you select each one you can see different instructions appear in the box directly below (colour co-ordinated). These can be used to create your game by dragging them across to the right where the scripts can be used for your project. On the right of the screen, you can see the white box with the Scratch cat inside, this is your project. As you begin developing your project, you can play and stop it using the green flag and red circle buttons above. There is one last section below the project screen, this is your library. As you begin adding more sprites, you can see them all being stored in your project library here.

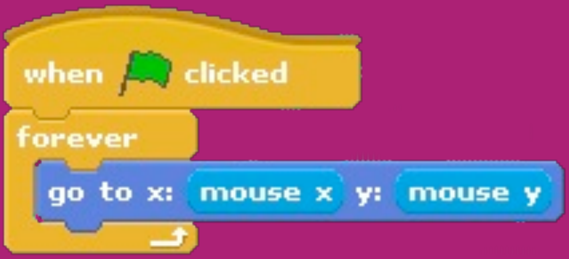
**Note:**

**"Sensing" is an important code category during the development of games for Kinect2Scratch, detecting certain movements of body parts via the Kinect sensor.**

## 2) Basic Movement

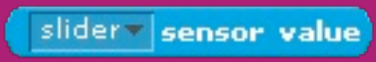
To begin, let's add a basic sprite, moving according to the movements of your right hand. Let's start by creating a simple forever loop, updating constantly the position of sprite according to the data input about where your right hand is located.

Firstly, let's code this according to mouse movements, then edit the code according to the motion of your right hand using the Kinect sensor:



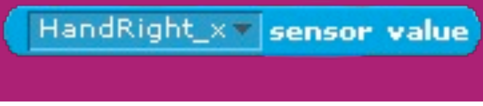
```
when ⚑ clicked
forever
    go to x: mouse x y: mouse y
```

After testing this and seeing how it works, you need to replace "Mouse x" and "Mouse y". Go onto "Sensing" and at the bottom of the category there is a code block that looks similar to below:



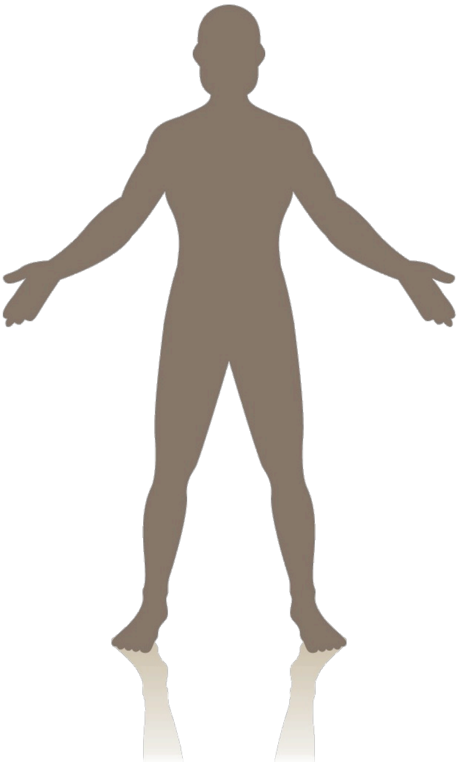
```
slider ▾ sensor value
```

If you click on "slider" you will see a large list of options to sense various parts of the body, for instance you can sense the right arm, left hand, the head, left or right ankles etc. For this example use the Right hand to control your sprite. Mouse x and Mouse y need to be replaced by the following code block, changing "HandRight_x" to "HandRight_y" for the "go to" y co-ordinate:



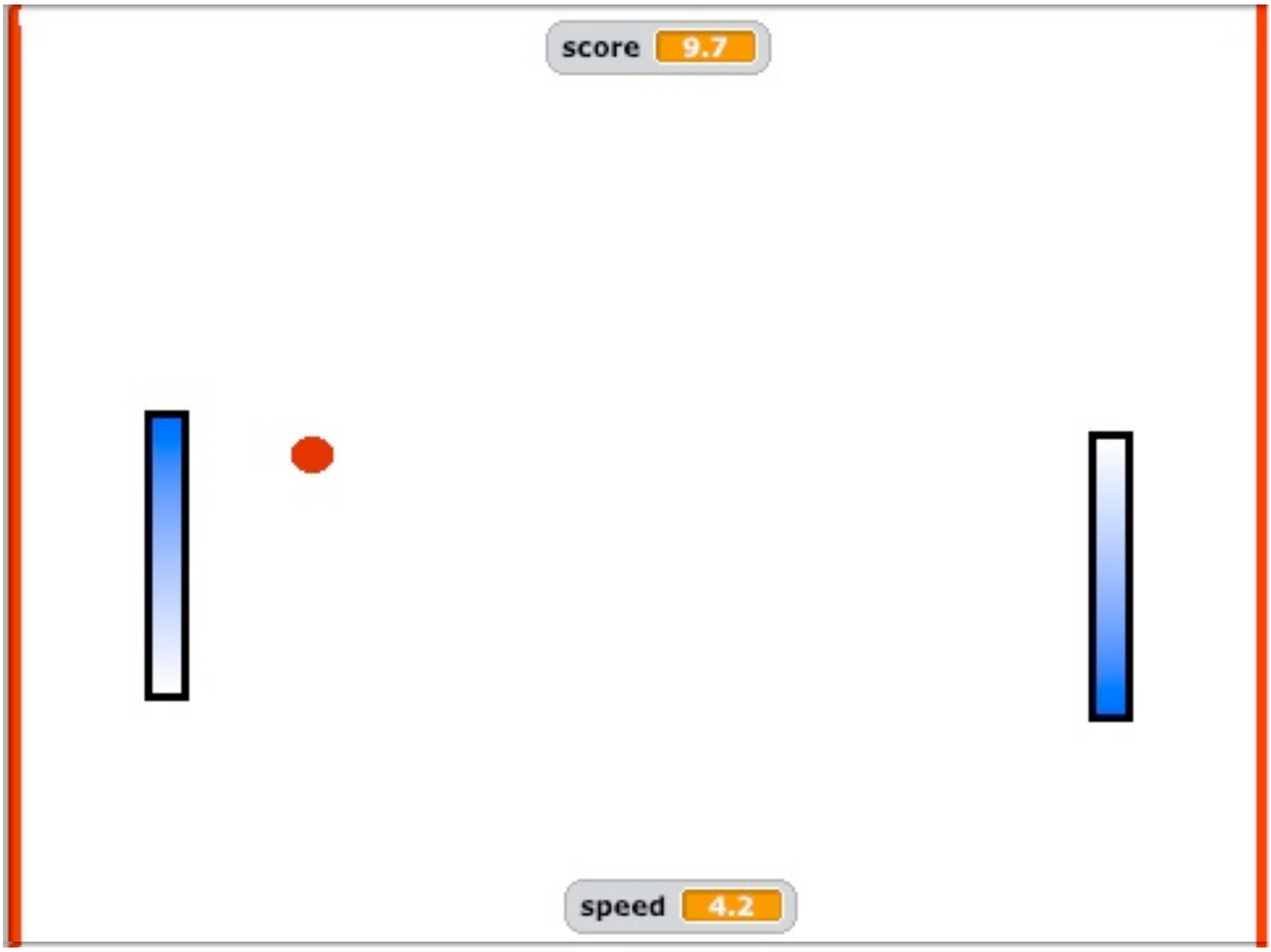
```
HandRight_x ▾ sensor value
```

After testing this and seeing how the Kinect sensor senses your position and can differentiate between the different parts of your body and movements, have a look at how clear the Kinect can sense you from different locations in the room or from different depths and see how quickly it can respond to quick movements. This assessment of the Kinect's capabilities will allow easy use of the Kinect later on with more complex game design and development, that way when it comes to designing your own game you are already aware of any limitations the Kinect may have.



# Kinect2Scratch

Now that you are familiar with the Scratch interface, the Kinect sensor and the Kinect2Scratch code blocks that are added to the Scratch interface can be used to begin developing your own computer game. You will be making a Paddle Game. This game is a 1 player game, however 2 players can be configured onto the Kinect2Scratch if you wish later on. The player must use their right hand and their left hand to control the movement of 2 separate paddles. These paddles must bounce a moving ball backwards and forwards accumulating a score.



### 3) Paddle Sprites

Create your own paddle sprite using the first out of three buttons next to the "New Sprite:" text beneath the game window. This will open up a window that looks similar to the "Paint" software used to draw/edit images on Windows. You can use this to draw your first paddle. This needs to be copied and pasted, so you have 2 separate sprites that look identical. This means they will be able to move independently. To copy, right-click the paddle sprite and press "Duplicate". The second sprite will automatically appear in the sprite library.

The initial instruction to give your first paddle is to tell it where its default location is on the game for each time the game is reset ("when green flag is pressed"). Set this using x and y co-ordinates and the "go to" command in the motion category. Now that you have set the starting position, the next task is to update the position of the sprite according to the motion of body parts detected through the Kinect visual sensor.

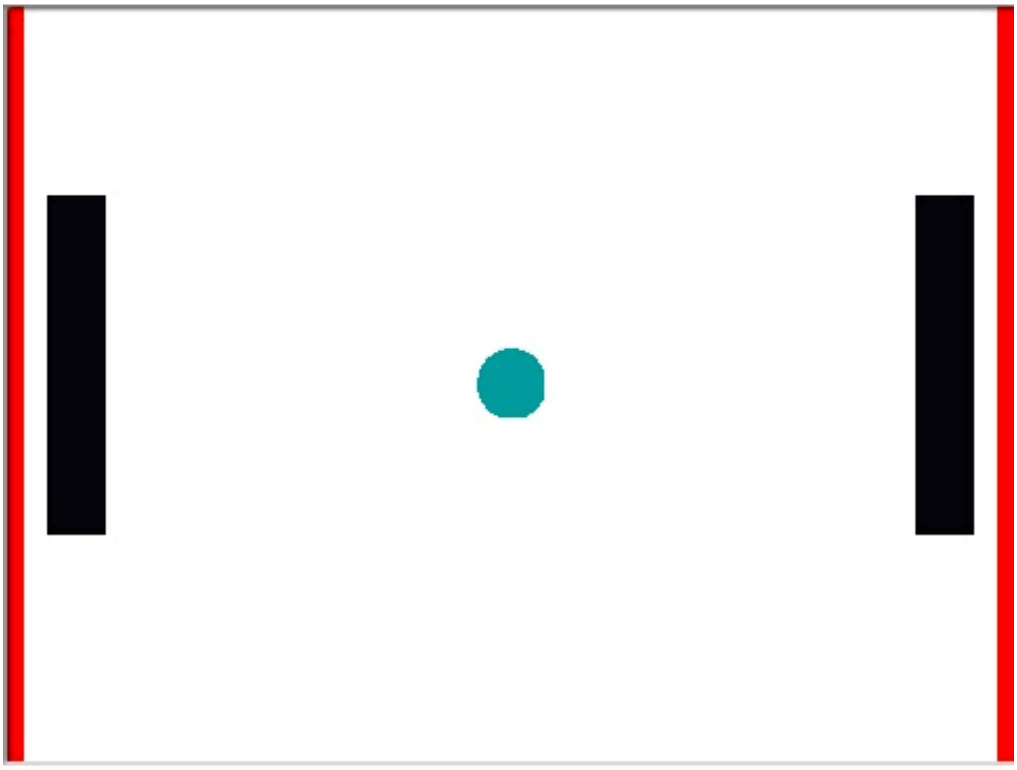
### 4) Paddle Movement

Program the paddle to move according to the movements of your hand. If it is the left paddle you are programming, code it to follow your left hand and if you are programming the right paddle to begin with, program it to follow your right hand. The paddle must only go up and down, so must stay on the same position on the x-axis. just the y-axis should change.

For a reminder of which code blocks to use to create a loop which regularly updates the position of the paddle according to hand gestures, take a look back at Task 2 on page 3.

### 5) Other Paddle

Program your second paddle, set the starting position for when the green flag is pressed using x and y co-ordinates. Then, program the other paddle to move according to the movements of your other hand.



The next sprite to implement onto your game is the ball, it is an important part of this game as it needs to move freely left and right, but also react if it collides with one of the paddles, bouncing off in the opposite direction. To start, create a ball sprite and add it to the screen. The first code to apply is setting the default position of x = 0 and y = 0 which is the center of the game window.

### 6) Ball movement

After the ball is set to its default position for when the game is reset (0, 0), the next task is to implement movement. Use the following instructions to implement code telling it to how to move:

```
“When green flag is clicked, go to starting position. Then, wait
  for a chosen amount of seconds before pointing randomly in a
          direction between 45 degrees and 135 degrees.

 Move forward a set amount of steps and if touching the edge of
  the game bounce off. This needs to be repeated over and over
                     throughout the game.”
```

Even though you are only telling the ball to move, e.g. 5 steps, as this instruction is being repeated over and over it will actually continuously move.

## Try this

**Another way to look at this is to learn at a dance move, like the Moon walk. The moonwalk is a famous Michael Jackson walk where he looked like he was gliding backwards across the stage when performing. The moonwalk can actually be broken down into 4 simple steps below, then this is repeated over and over to have the gliding effect:**

1. **Lift your left heel up. Your right foot needs to be flat on the ground.**
2. **Keep your left foot in place and slide your right foot behind you, keeping it flat on the ground.**
3. **Lift your right heel. Your left foot needs to be flat on the ground instead.**
4. **Keep your right foot in place and slide your left foot behind you, keeping it flat on the ground.**

**Try this out. You will see the benefits of learning 4 steps and repeating, rather than going left, right, left, right, left, right, left etc.**

So far, both Paddles should move up and down and the ball should move independently, the ball should also successfully bounces if it comes in contact with the edges of the game. But, the ball does not yet know how to respond if it collides with one of the paddles. The aim is for the ball to bounce off the paddles and accumulate a score.

## 7) Ball reflection off paddles

The code for the ball can be expanded further to contain 2 if statements. Use the following instructions to guide you to implement code telling the ball how to bounce off the paddles when contact is made:

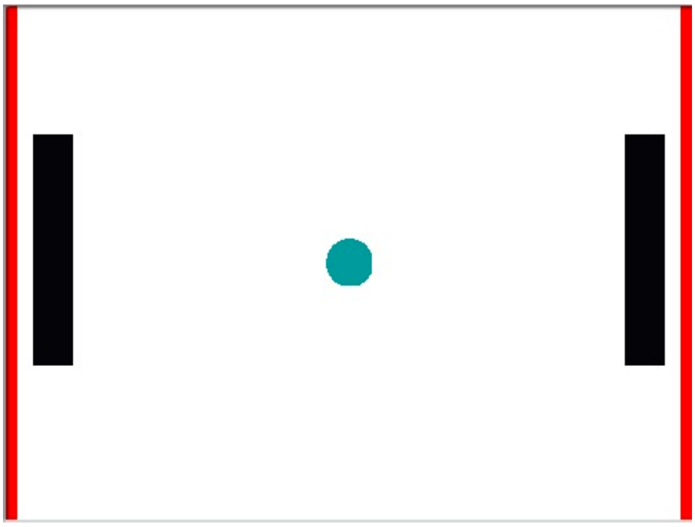
```
“If the ball touches the left paddle, point in a random
direction between 45 and 135 degrees and move forward.

If the ball touches the right paddle, point in a random direction
between -45 and -135 degrees and move forward.”
```

As before, even though you are only telling the ball to move, for example, 5 steps, as this instruction is being repeated over and over it will actually continuously move.

The background at the moment is plain white, the ball can recognise the edges and bounce off but you need to draw 2 vertical borders on the left and right hand side of the game, behind the paddles that will either deduct a score or end the game if the ball comes into contact with them. To do this you can draw the 2 borders on the stage which is the background image for the game.

## 8) Background

Select the “Stage” icon in the Sprite library beneath the game. Select the “Backgrounds” tab. Here you can either import an image for your background or you can draw one on the Paint window like you did for the Paddles. Draw a thick, red line on the left and right hand side of the game.



### 9) Ball red line detection

The code for the ball can be expanded further to contain another if statement. Use the following instructions to guide you to implement code telling the ball how to respond when contact is made with the red lines sandwiching the game window:

```
“If the ball touches the colour red, wait a few seconds then
tell the ball to go back to the center of the game and randomly
choose which direction to start moving in again between 45 and
135 degrees.”
```

The game could be expanded even further, for a competitive element a score can be introduced. See if you can implement a new “score” variable. The score value can be increased each time a paddle touches the ball, then when the ball touches the red, the score could either be deducted or even go back to 0.

**Don’t forget to add some code in your game telling your score variable to reset to 0 when the green flag is pressed (reset button).**

Another way to increase the competition factor of the game is to gradually increase difficult as the game progresses. There are several options to this as the you could create a timer (another variable that in a forever loop waits one second then adds 1 point to the variable) and as the timer hits certain values throughout the game, the speed of the ball or the paddles could increase. Alternatively, as the score builds up, the game could also respond by shortening the size of the paddles making it more tricky to be precise and hit the moving ball or you could decrease the size of the ball, or even introduce several new balls to hit!

### 10) Lets try it ourselves

Now that you are familiar with the scratch interface, the Kinect sensor and the capabilities of both combined, design a game to create on Scratch using Kinect2Scratch. Consider what sprites you will use, how they will move or be controlled. What part of the body will you use to control it? Will you re-configure the Kinect2Scratch and have a 2 player game?

Implement your game, carefully using the “Control” category to begin blocks of code in the Sprite scripts. See if you can use conditional statements and loops (if and forever).

technocamps

www.technocamps.com

@Technocamps

Find us on Facebook