Pong!

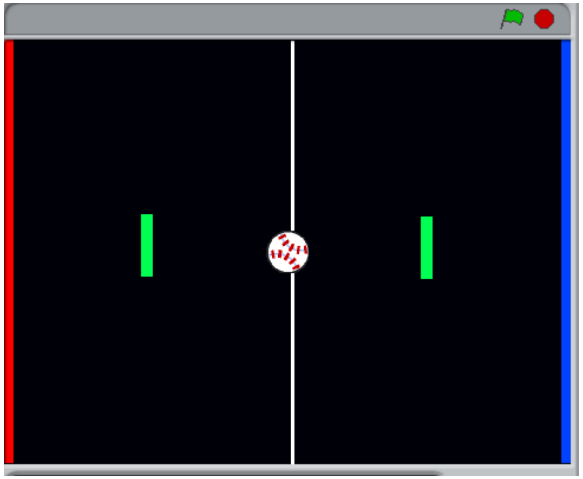
“The oldest commercially available game in history”

Resources created from the video tutorials provided by David Phillips on http://www.teach-ict.com

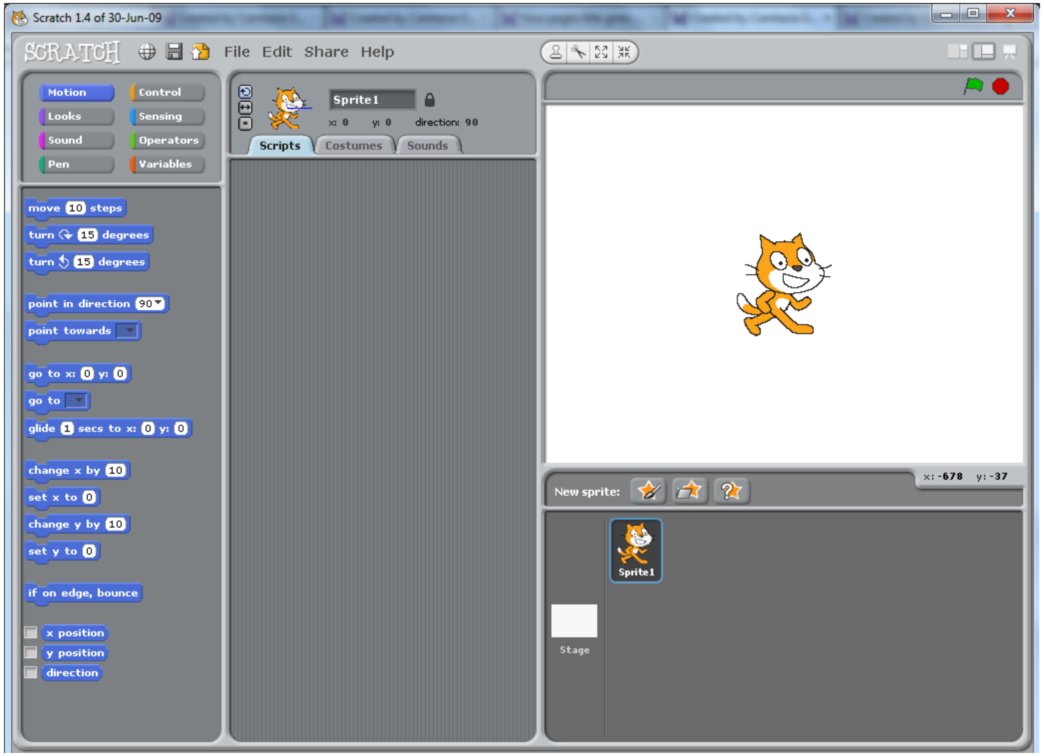
Stage 1

- Before you start to script the game you need to create the assets.

- **<u>Stage 1</u>** will help you to create your background image, score zones, paddles (bats) and ball.



Getting Started – Open SCRATCH

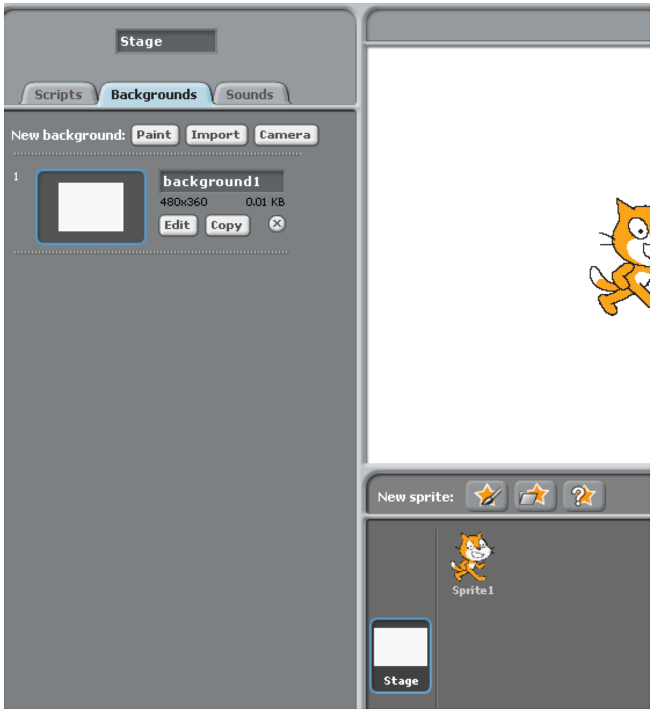


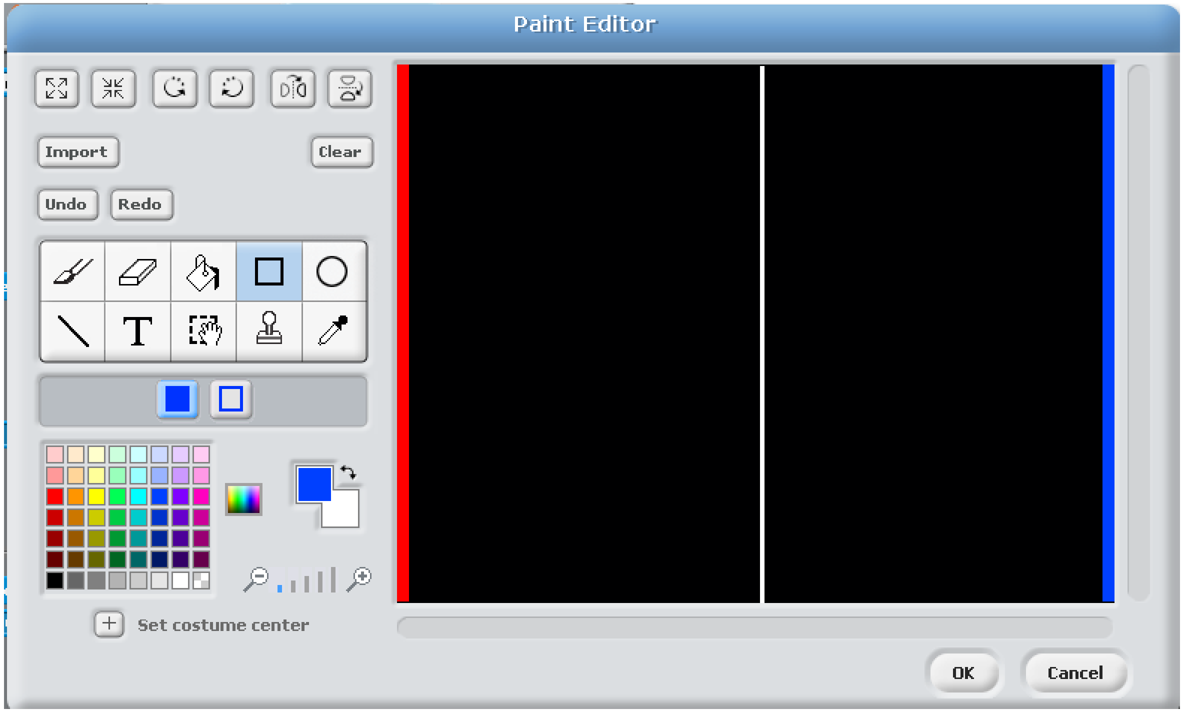
1. Save As.
2. My Home Drive.
3. SCRATCH PROJECTS.
4. Pong.

QUICK QUESTION:

Why are you saving your file BEFORE you have changed anything?

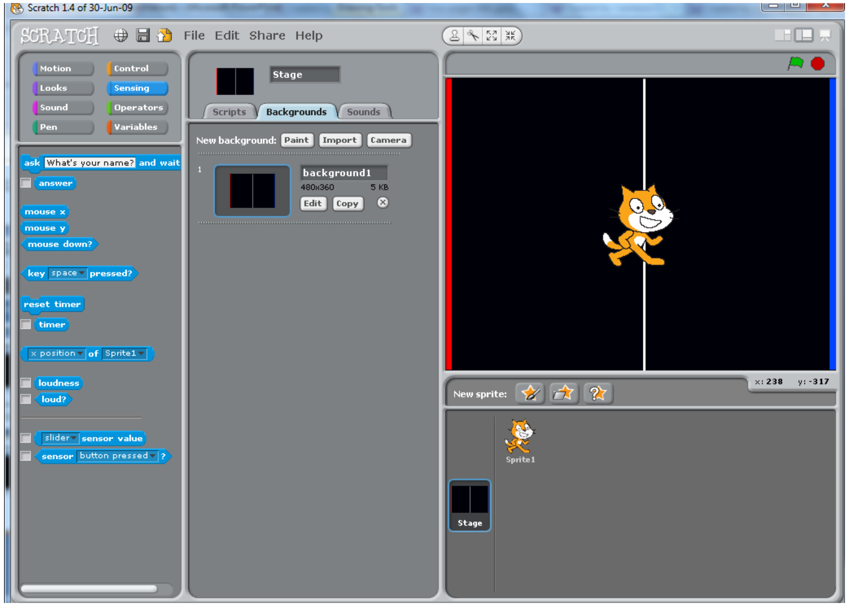
Creating the background

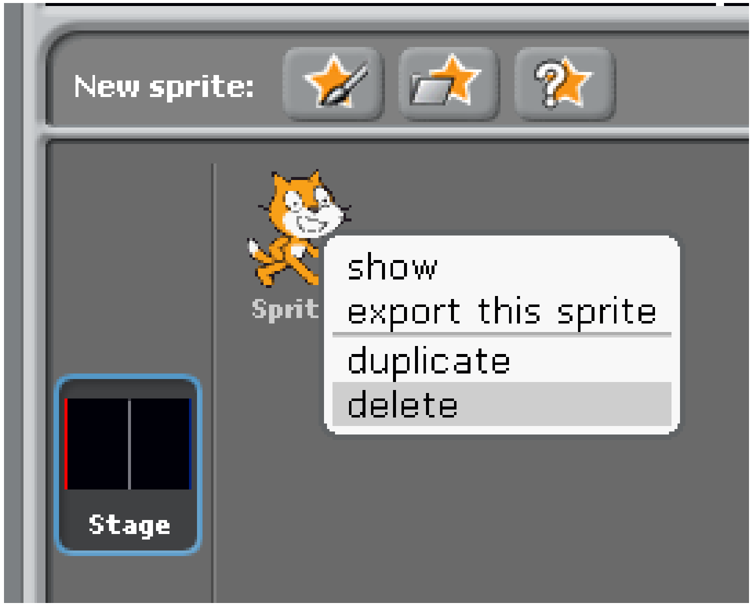




1. You need to select **stage** and **background.**
2. Use the **paint bucket** to colour in the background.
3. Use the **line tool** to create the centre line.
4. Use the **rectangle tool** to create the score zones in two contrasting colours – I have used red and blue in the example.

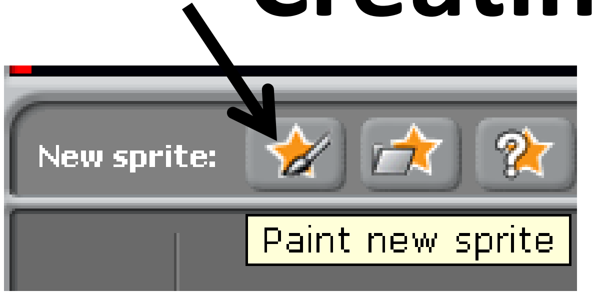
Creating the paddles part 1

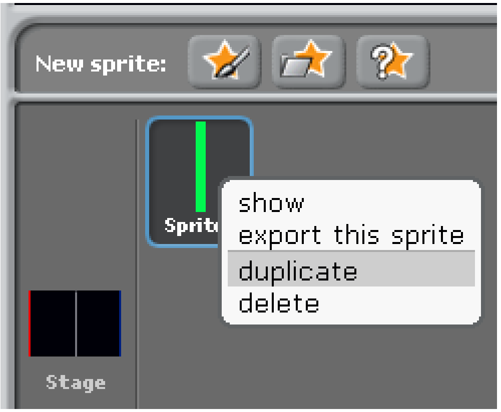


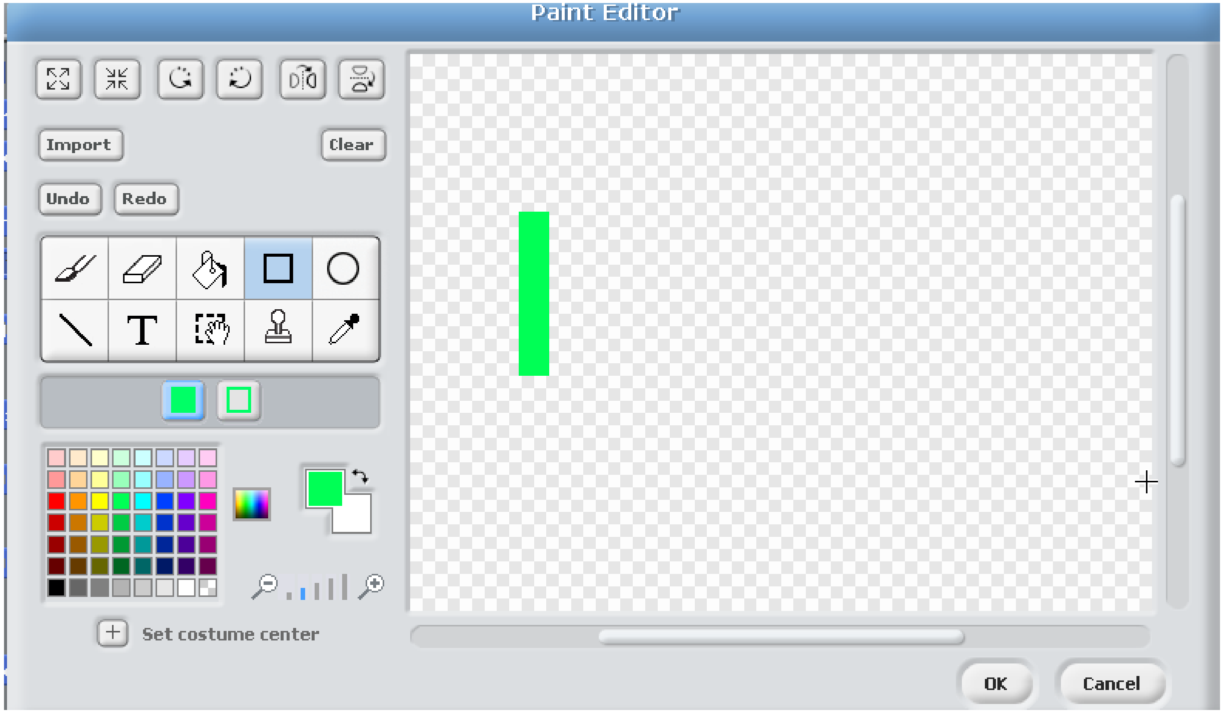


1. You need to **delete the cat** sprite.
2. **Right click** and select **delete.**

Creating the paddles part 2

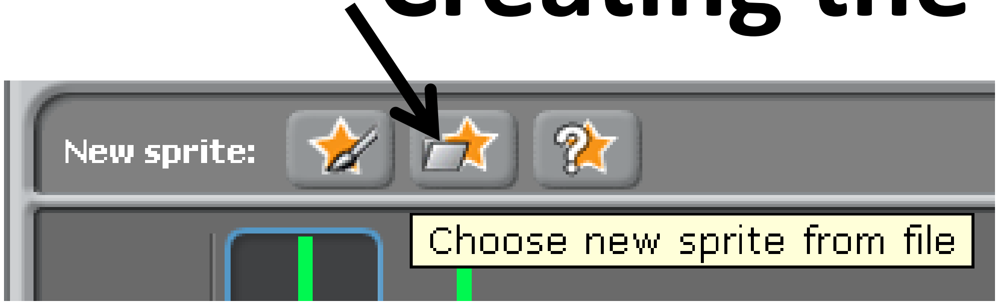


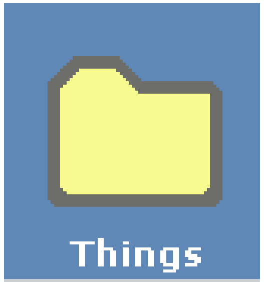


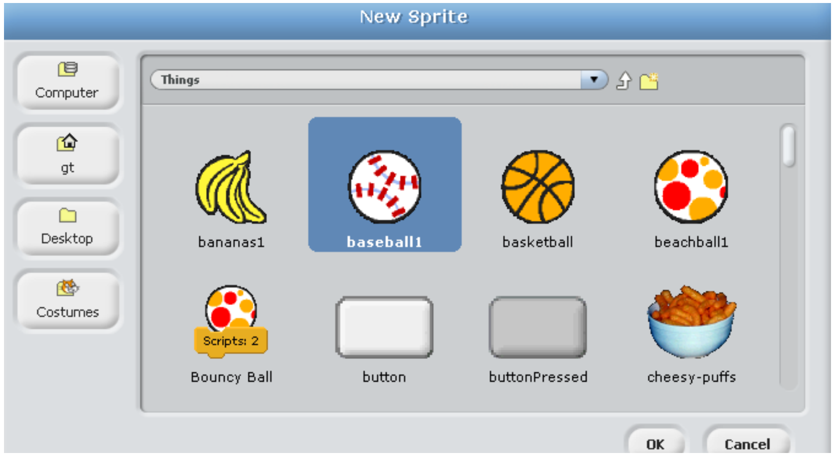


1. Select **paint new sprite.**
2. Use the **rectangle tool** to create a suitable paddle.
3. Select OK.
4. Right click to duplicate the paddle for player #2.

Creating the ball part 1

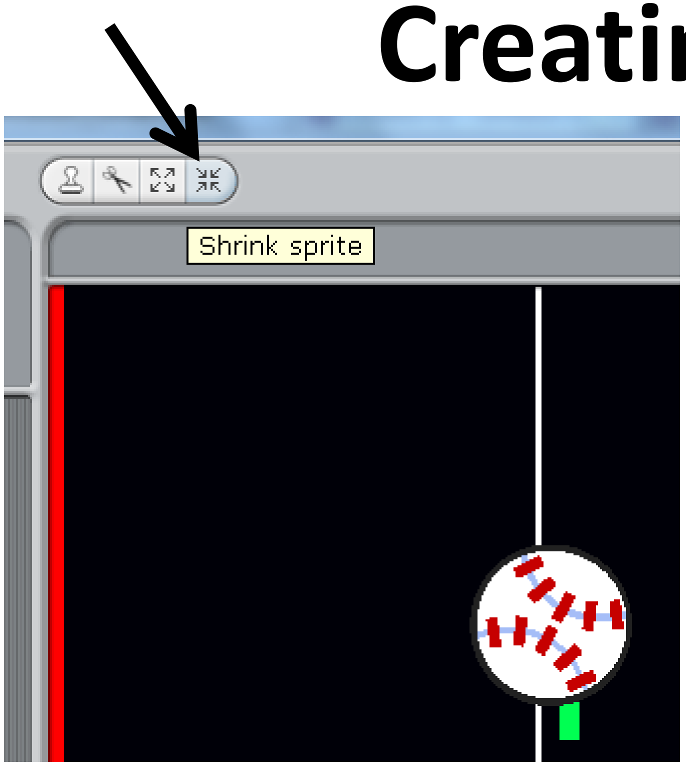


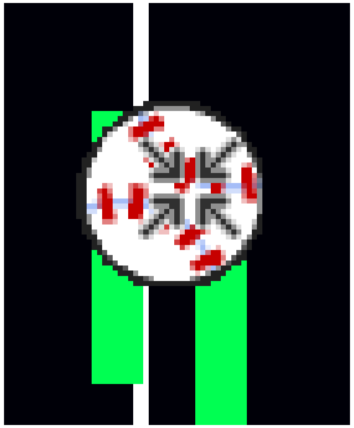


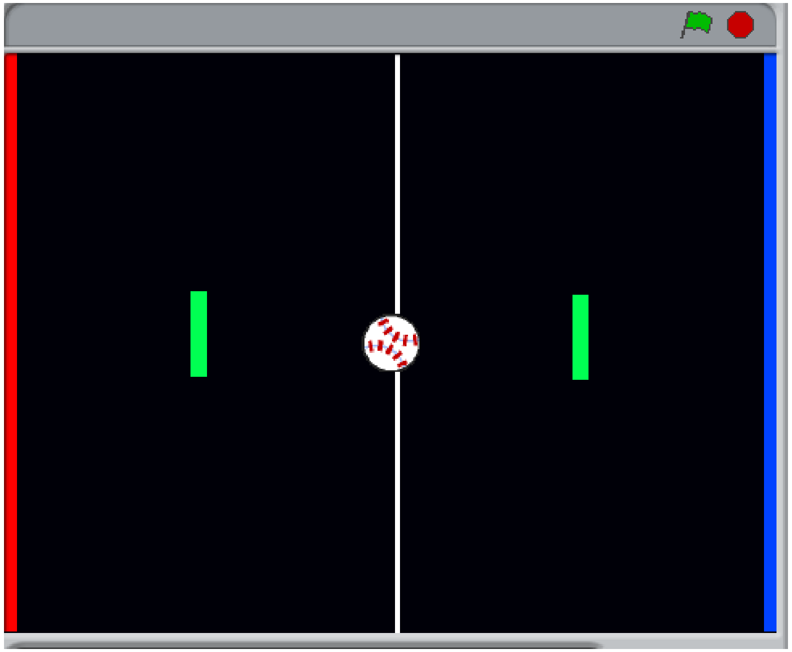


1. Select “Chose new sprite from file”.
2. Select the “Things” folder.
3. Search for an appropriate ball sprite to use.
4. Press OK.

Creating the ball part 2







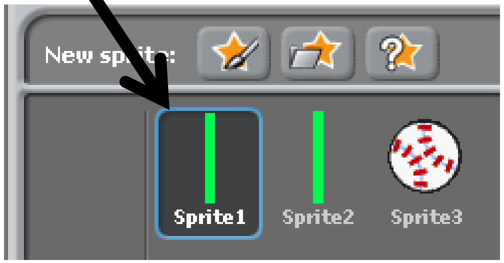
1. You need to shrink the ball using the **shrink sprite** tool.
2. Hover over the ball sprite and **click your mouse** to shrink it to an appropriate size for your game.
3. Then, re-arrange the assets to the “start position”.

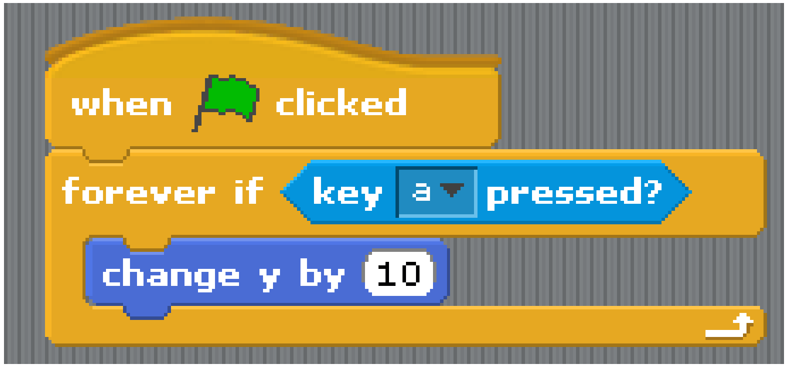
Stage 2

- Getting the paddles to move up and down.

- **Stage 2** will help you to make the player 1 and player 2 paddle controls to move them up and down on the Y axis.

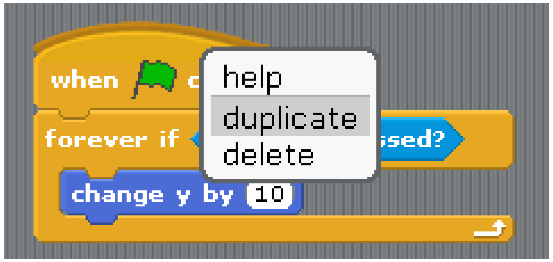
Making the paddles move – Player #1



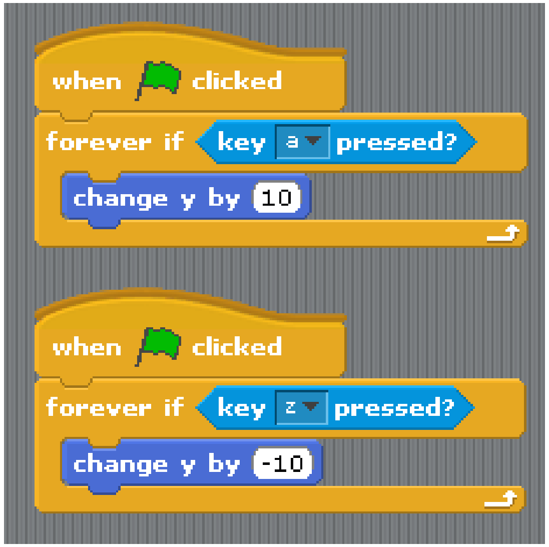


```
when 🏁 clicked
forever if key a pressed?
    change y by 10
```

1. Select the first paddle(**sprite1**).
2. Create the **script** shown above.
3. Right click to duplicate the script, then change the key to "z" and "-10" to move the paddle down.



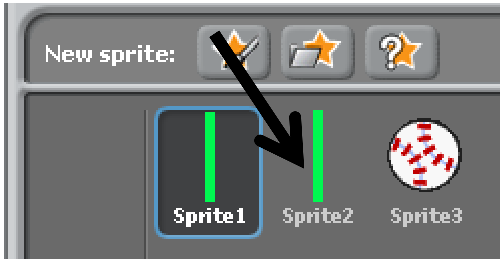
```
when 🏁 c
forever if       ssed?
    change y by 10
```

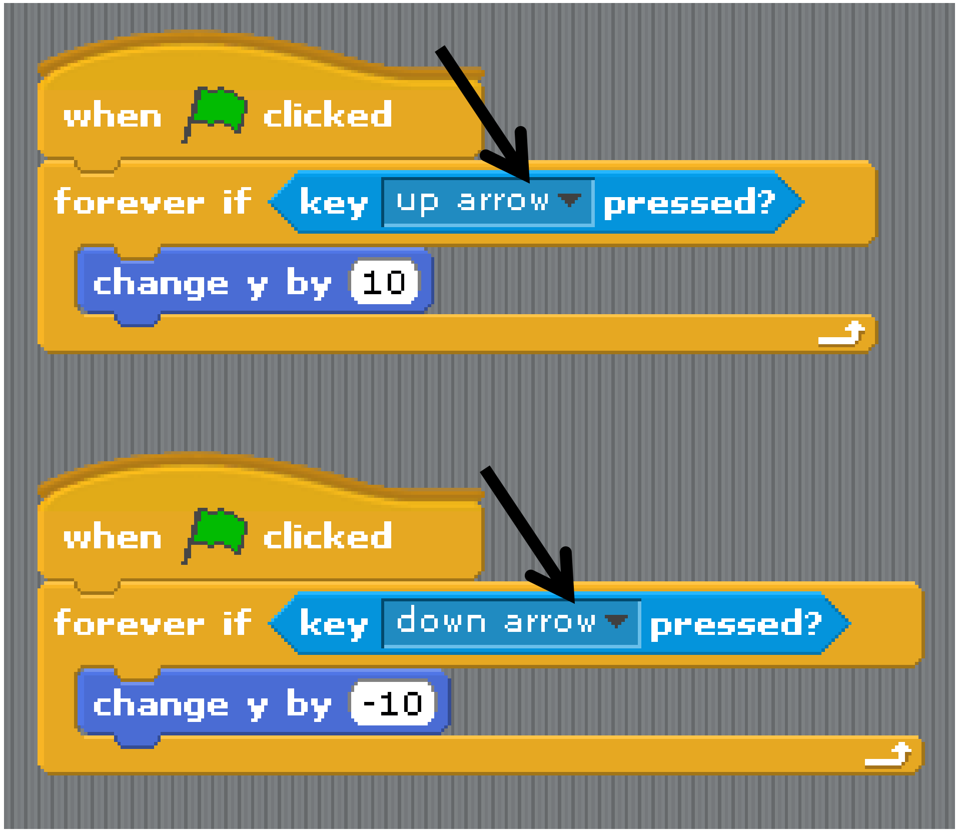


```
when 🏁 clicked
forever if key a pressed?
    change y by 10

when 🏁 clicked
forever if key z pressed?
    change y by -10
```

Making the paddles move – Player #2





```
when [green flag] clicked
forever if <key [up arrow] pressed?>
    change y by (10)

when [green flag] clicked
forever if <key [down arrow] pressed?>
    change y by (-10)
```

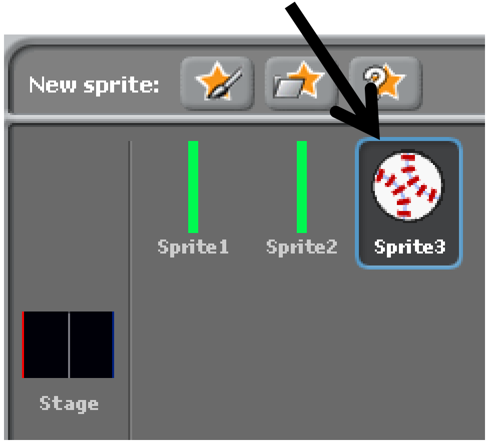
1. Select the first paddle (**sprite1**).
2. Drag the script over Sprite2 to copy the script for player #2.
3. Select the second paddle (**sprite 2**), right click to duplicate the script, then change the keys to “up arrow” and “down arrow” and the values “10” and “-10” as per the previous script.

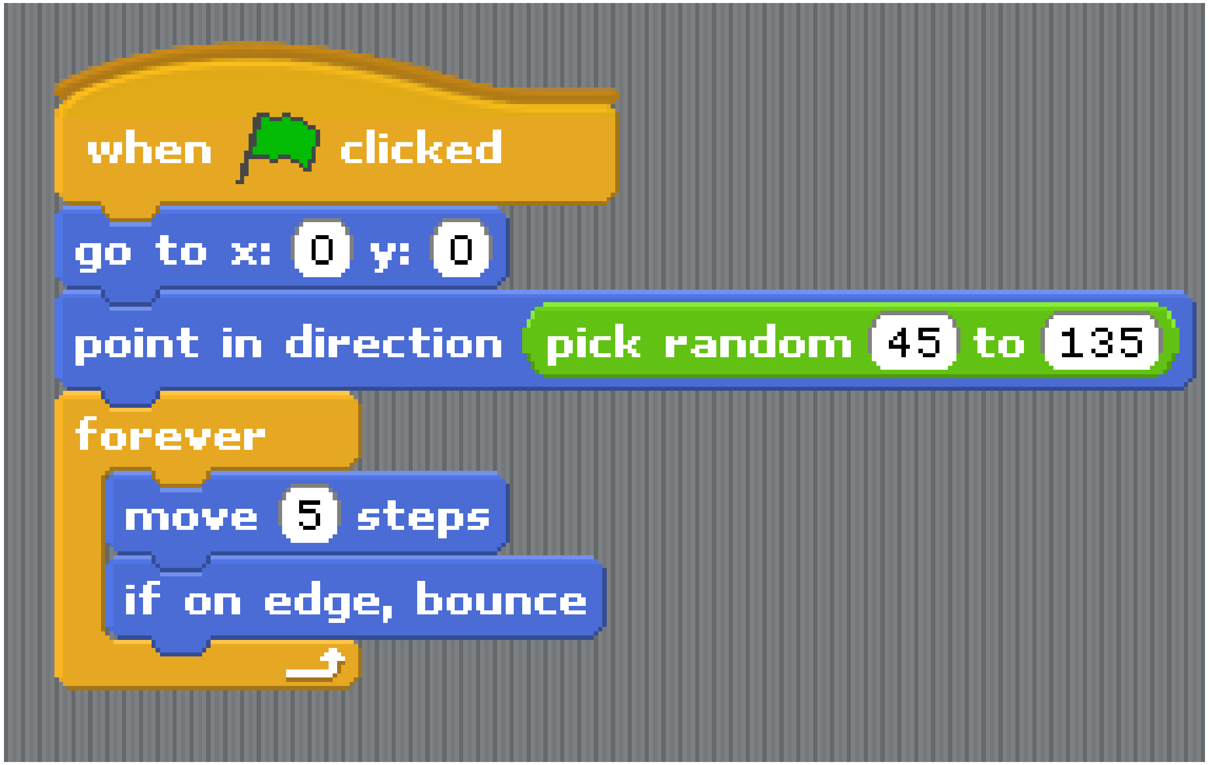
Stage 3

- Getting the ball to move in a “fairly random” manner.

- **Stage 3** will help you to make the ball bounce off the sides of the game area with a simple script.

Making the ball move





```
when [flag] clicked
go to x: 0 y: 0
point in direction (pick random 45 to 135)
forever
    move 5 steps
    if on edge, bounce
```

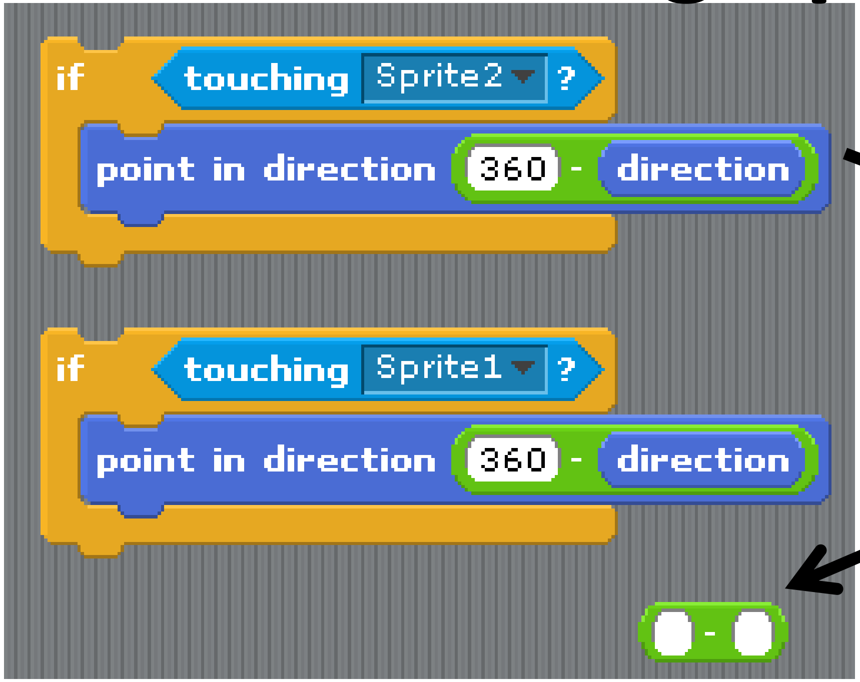
1. Select the ball (**sprite3**).
2. Create the **script** shown above.
3. **Test** the ball uses random directions by starting and stopping the game 2-3 times.

Stage 4

- Playing the game.

- **Stage 4** will help you bounce the ball off the paddles so that the game can be played.

Setting up the paddles

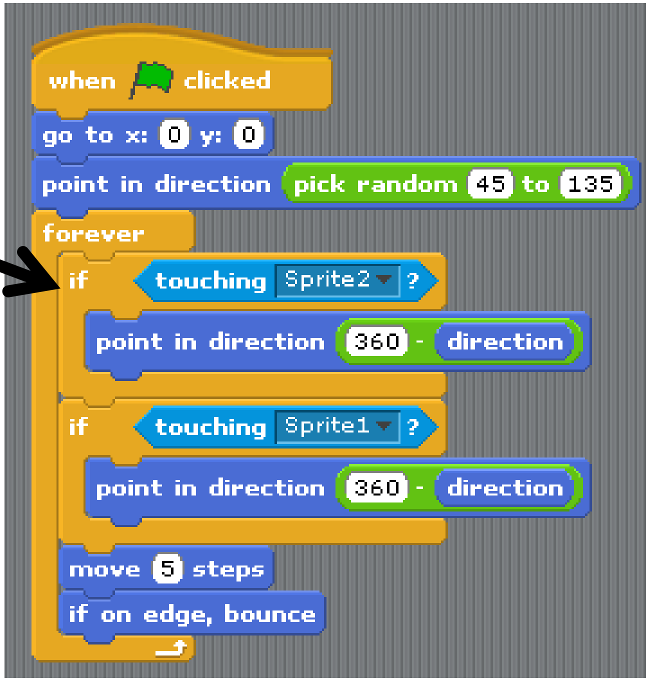


```
if <touching [Sprite2 v] ?>
  point in direction (360 - direction)

if <touching [Sprite1 v] ?>
  point in direction (360 - direction)

( ) - ( )
```

Use this operator to set up the direction instructions.



```
when [flag] clicked
go to x: 0 y: 0
point in direction (pick random 45 to 135)
forever
  if <touching [Sprite2 v] ?>
    point in direction (360 - direction)
  if <touching [Sprite1 v] ?>
    point in direction (360 - direction)
  move 5 steps
  if on edge, bounce
```

1. You need to create an **IF statement** for the behaviour of each sprite, within the ball sprite.
2. Select the ball (**sprite3**), create the statements above.
3. **Move** them into the correct place in the ball script.
4. Don’t forget to **test** that they work!

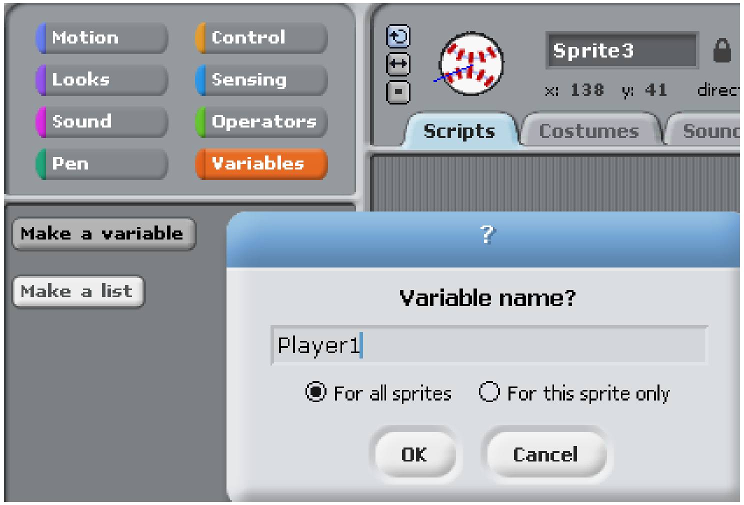
Stage 5

- The score board(s).

- **Stage 5** will help you create scores for your game, using the red and blue score zones that you set up at the very start.

- Remember blue = player 1 and red = player 2.

Setting up the score board part 1



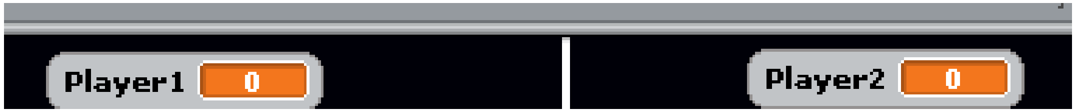
Sprite3

x: 138 y: 41 direc

Variable name?

Player1

● For all sprites ○ For this sprite only

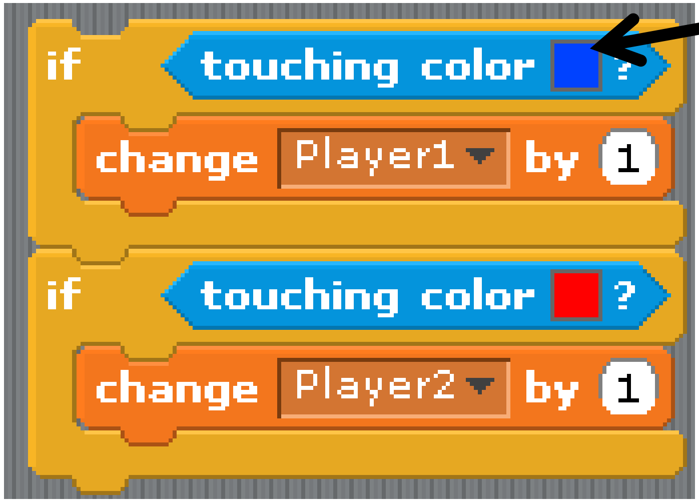


Player1 0

Player2 0

1. You need to create variables for each player.
2. Select the ball (**sprite3**).
3. Go to the variables options and select “make a variable”.
4. Name the variable “Player1”, then repeat for “Player2”.

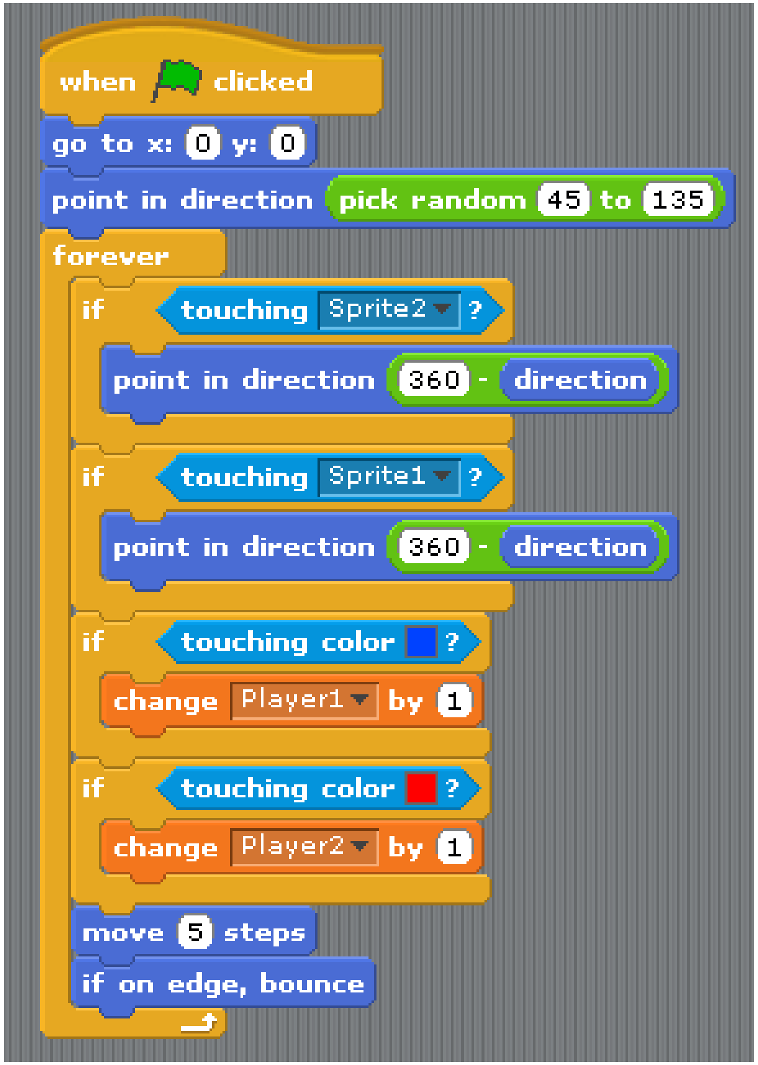
Setting up the score board part 2



```
if touching color 🟦 ?
    change Player1 ▼ by 1
if touching color 🟥 ?
    change Player2 ▼ by 1
```

To select the score zone colour click on the coloured square for the dropper to come up, then click the dropper on the score zone.

1. Create the if statements above to define the rules for scoring points.
2. Then drag them into the forever command below the other if statements.



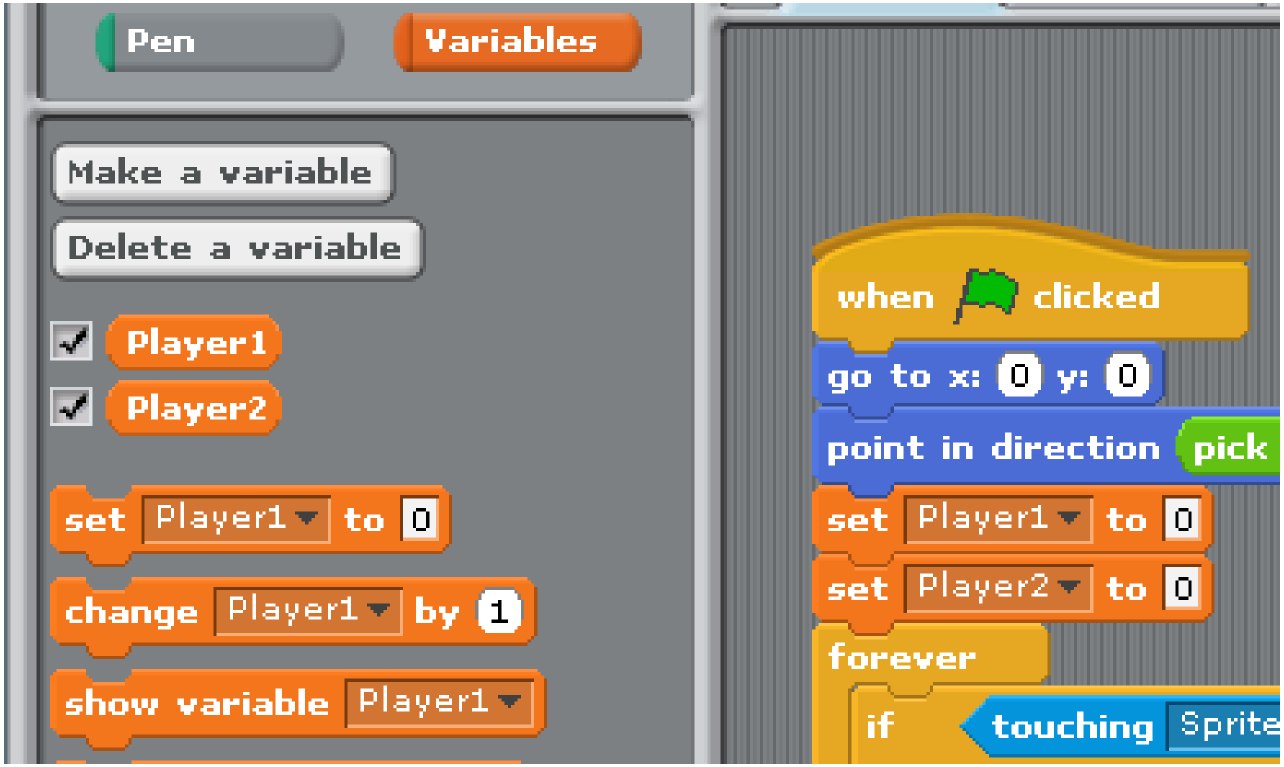
```
when ⚑ clicked
go to x: 0 y: 0
point in direction pick random 45 to 135
forever
    if touching Sprite2 ▼ ?
        point in direction 360 - direction
    if touching Sprite1 ▼ ?
        point in direction 360 - direction
    if touching color 🟦 ?
        change Player1 ▼ by 1
    if touching color 🟥 ?
        change Player2 ▼ by 1
    move 5 steps
    if on edge, bounce
```

Stage 6

- Improvements

- **Stage 6** will help you refine the scoring system and the starting position for the ball to make the game more accurate

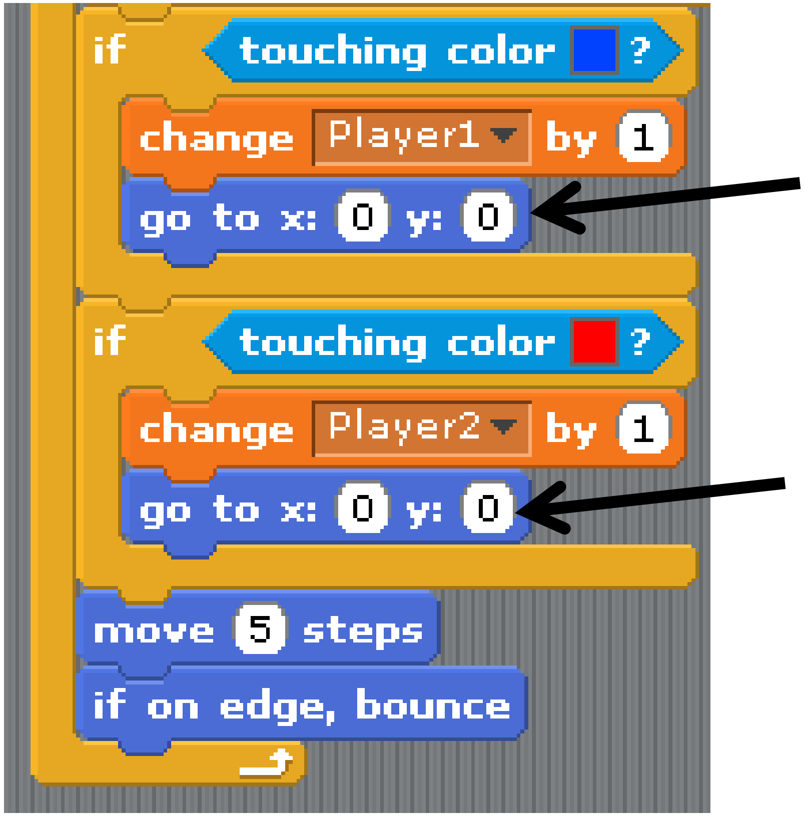
Improvement #1 Re-setting the scores



```
when 🏁 clicked
go to x: 0 y: 0
point in direction pick
set Player1 to 0
set Player2 to 0
forever
    if touching Sprite
```

1. Use the **set variable** command to re-set the scores to zero each time the game is stopped and started again.
2. Don’t forget to test it.

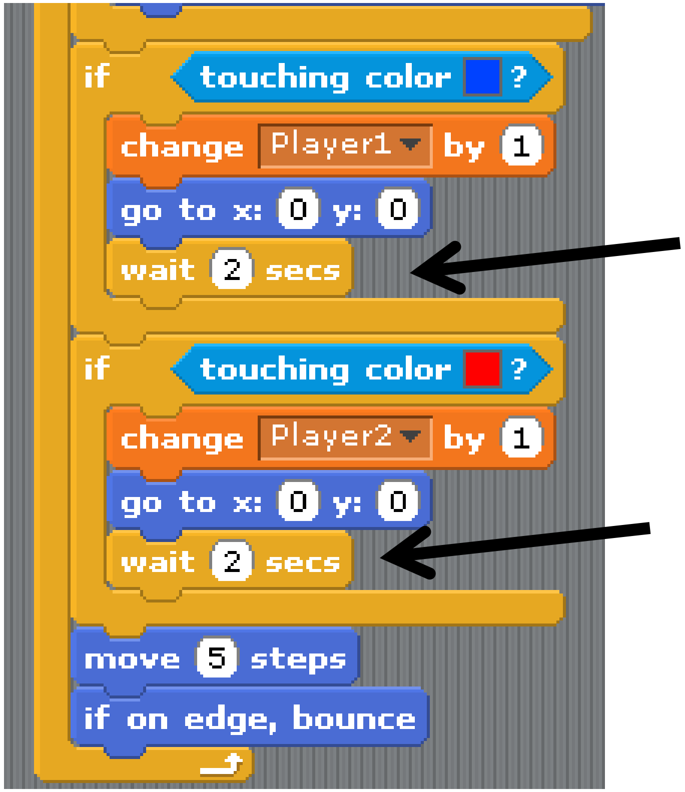
Improvement #2 Re-positioning the ball



```
    if touching color 🟦 ?
        change Player1 ▾ by 1
        go to x: 0 y: 0
    if touching color 🟥 ?
        change Player2 ▾ by 1
        go to x: 0 y: 0
    move 5 steps
    if on edge, bounce
```

1. After a point has been scored the ball should be returned to the centre of the screen.
2. Use the **go to** motion script adjustments above to create this.

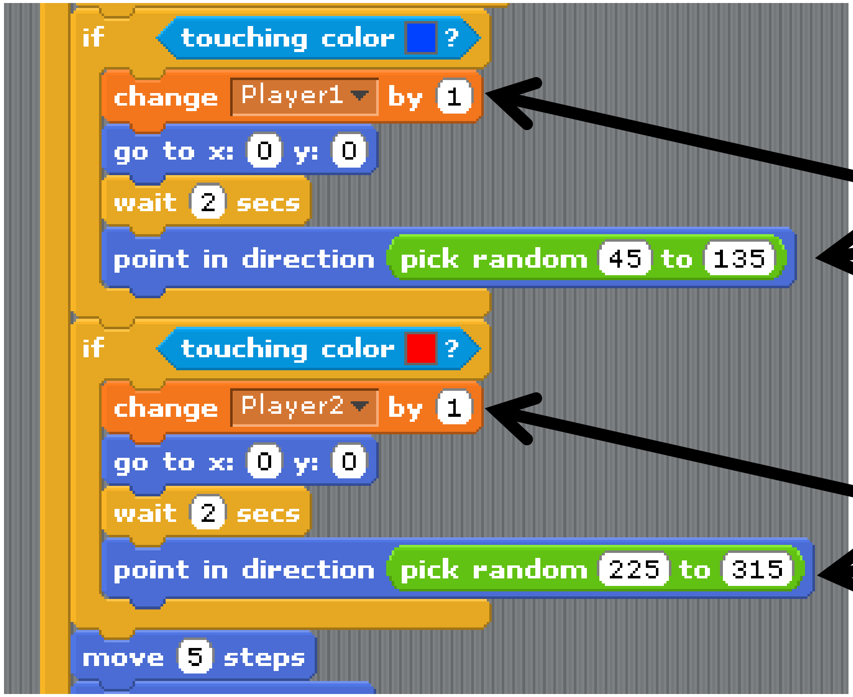
Improvement #3 Adding a pause



```
if <touching color 🟦 ?>
    change [Player1 ▾] by (1)
    go to x: (0) y: (0)
    wait (2) secs
if <touching color 🟥 ?>
    change [Player2 ▾] by (1)
    go to x: (0) y: (0)
    wait (2) secs
move (5) steps
if on edge, bounce
```

1. Adding a delay gives the players time to re-position their paddles for the next point.
2. Use the **wait** control script adjustments above to create this.

Improvement #4 Changing direction



```
if touching color 🟦 ?
    change Player1 by 1
    go to x: 0 y: 0
    wait 2 secs
    point in direction pick random 45 to 135
if touching color 🟥 ?
    change Player2 by 1
    go to x: 0 y: 0
    wait 2 secs
    point in direction pick random 225 to 315
move 5 steps
```

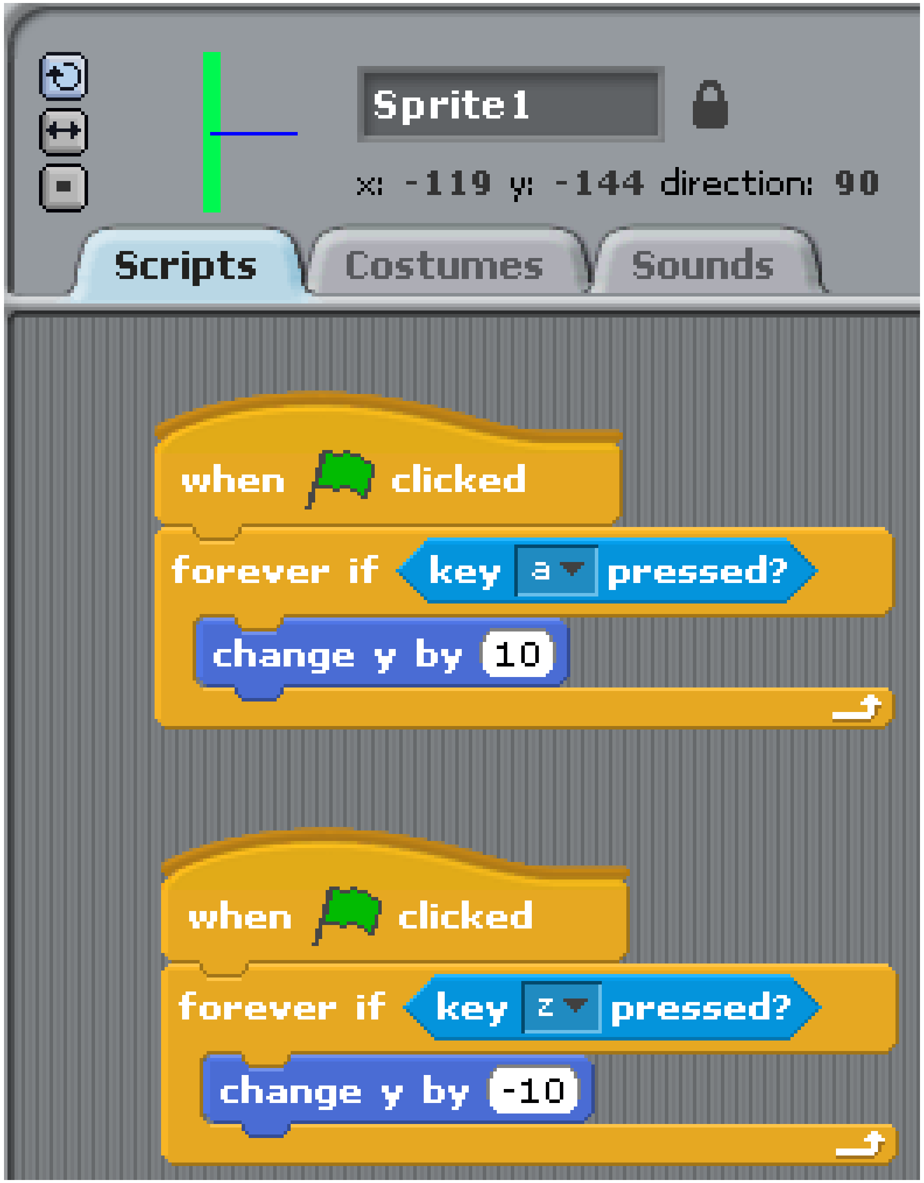
Check that you have set the direction for the correct player.

1. Adding a delay gives the players time to re-position their paddles for the next point.
2. Use the **point in direction** motion script and **pick random** operator.

Summary

- The scripts that you will use have been summarised here for you to check your work against.

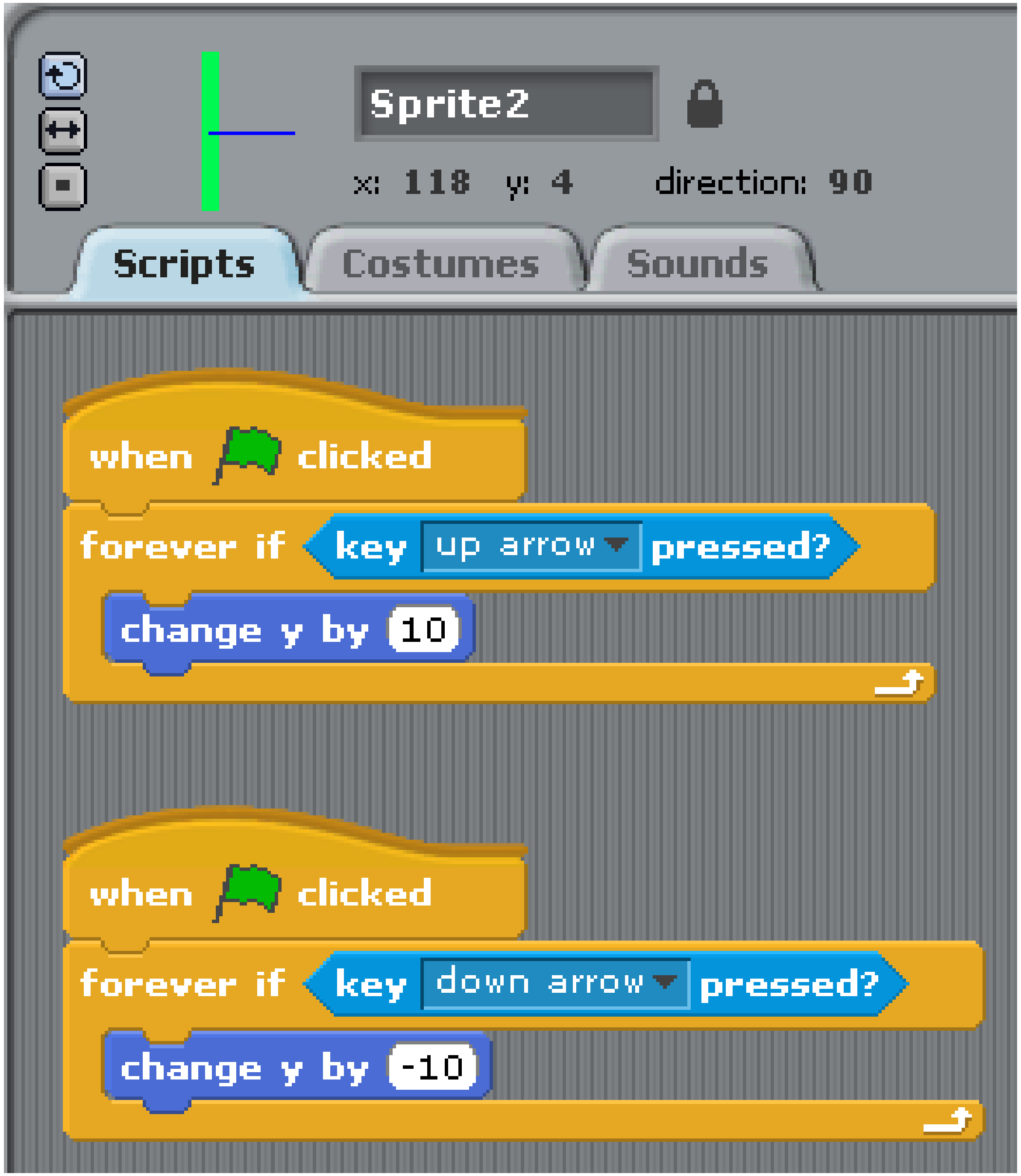
Sprite1



```
when [green flag] clicked
forever if <key [a] pressed?>
    change y by 10

when [green flag] clicked
forever if <key [z] pressed?>
    change y by -10
```

Sprite2



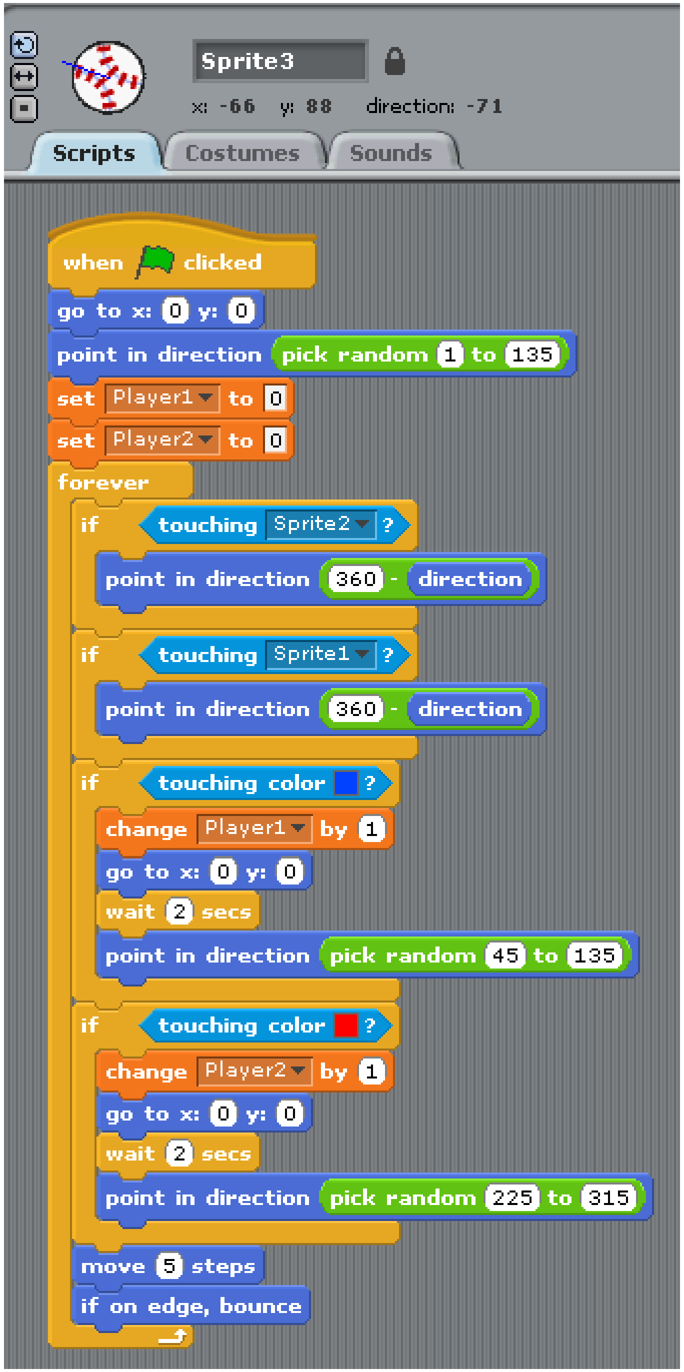
Sprite2

x: 118 y: 4 direction: 90

```
when [green flag] clicked
forever if <key [up arrow] pressed?>
    change y by (10)

when [green flag] clicked
forever if <key [down arrow] pressed?>
    change y by (-10)
```

Sprite3



Sprite3

x: -66 y: 88 direction: -71

```
when 🟩 clicked
go to x: 0 y: 0
point in direction (pick random 1 to 135)
set Player1 to 0
set Player2 to 0
forever
    if touching Sprite2 ?
        point in direction (360 - direction)
    if touching Sprite1 ?
        point in direction (360 - direction)
    if touching color 🟦 ?
        change Player1 by 1
        go to x: 0 y: 0
        wait 2 secs
        point in direction (pick random 45 to 135)
    if touching color 🟥 ?
        change Player2 by 1
        go to x: 0 y: 0
        wait 2 secs
        point in direction (pick random 225 to 315)
    move 5 steps
    if on edge, bounce
```

Why not try . . .

- 4 players in the game?
- Changing the sprites to be more interesting?
- Making the paddles shorter?
- Speeding up the ball?
- Making the score zone smaller, maybe add goal posts?
- Changing the colours/layout to make it more child friendly?
- Changing the layout/colours to make it more appealing to an older audience?
- Writing instructions for the technical user and end users?