[the academy_of_code]

Grade 2 - Unit 1&2 (Games)

[the academy_of_code]

Hungry Shark

Lets get Coding

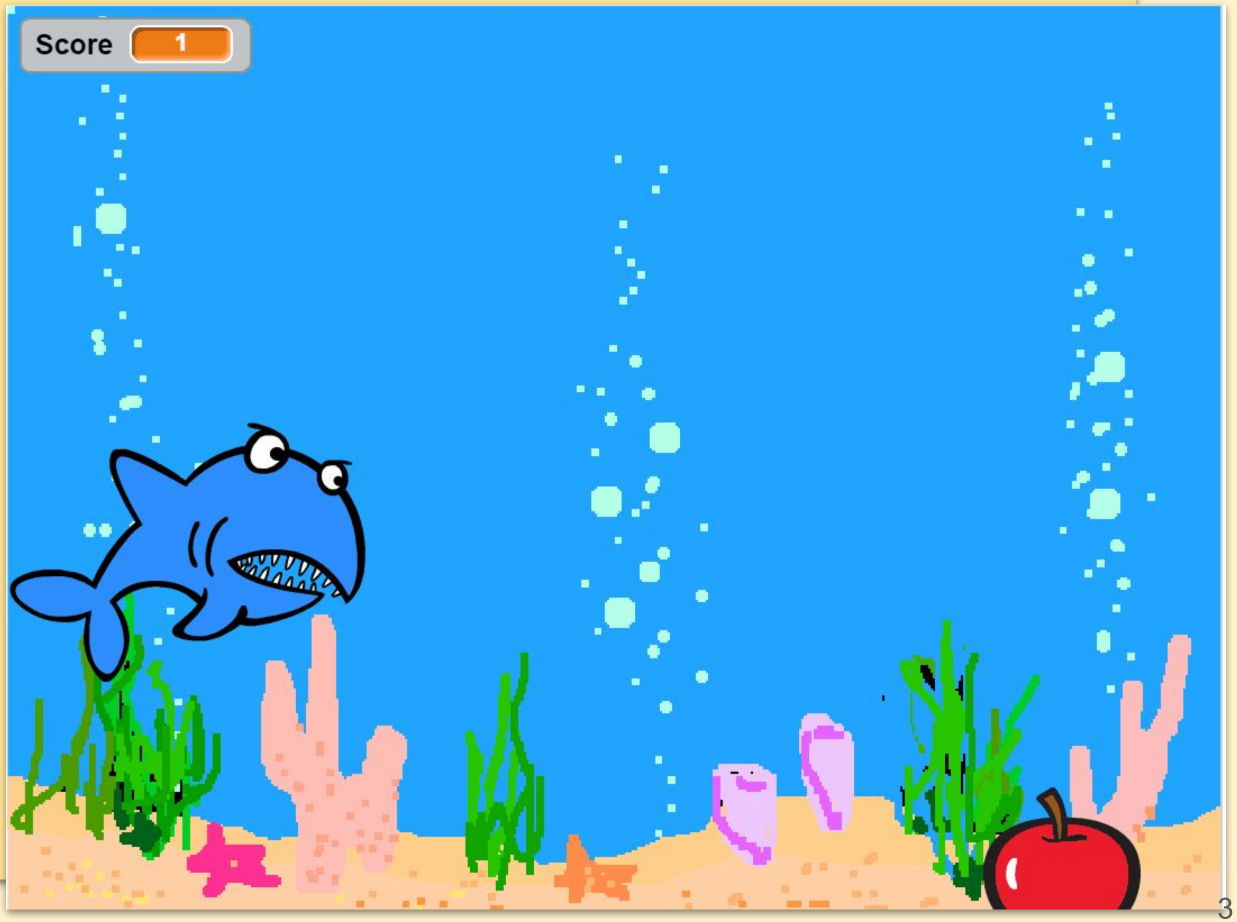
Ever since we were old enough to play hide and seek and tag, we've been hooked on gaming, but in the same way we needed to learn to walk and talk before we could make and play these games, we need to learn coding basics to make and play the Scratch games we're going to make.

We're now going to make our first game; ***Hungry Shark.***

How many sprites are in this game?

*What **variables** can you see?*

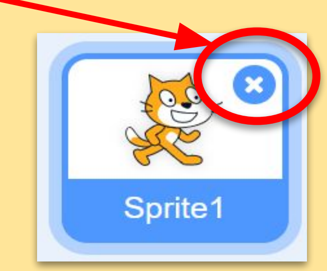
How do you think the game will work?

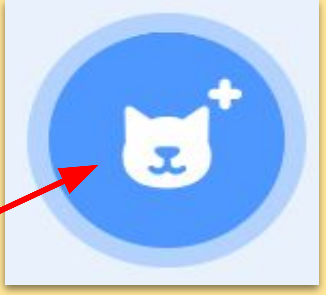


Lets get Coding

1 Name your project (For example: The Hungry Shark)

2 Click on the **'X'** icon to delete the **Cat** sprite giving us a fresh and empty stage ready for programming.

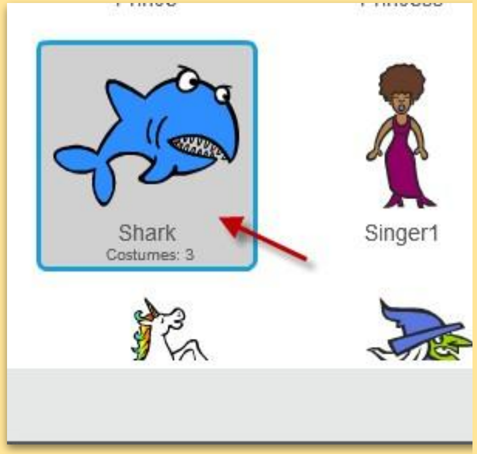


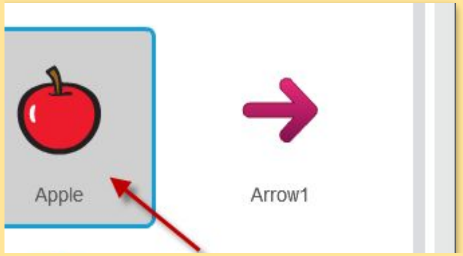


3 Click on **'Choose sprite from library'** and add 2 Sprites:

- **Shark**

- Something for the Shark to Eat





Your project should have two sprites now.

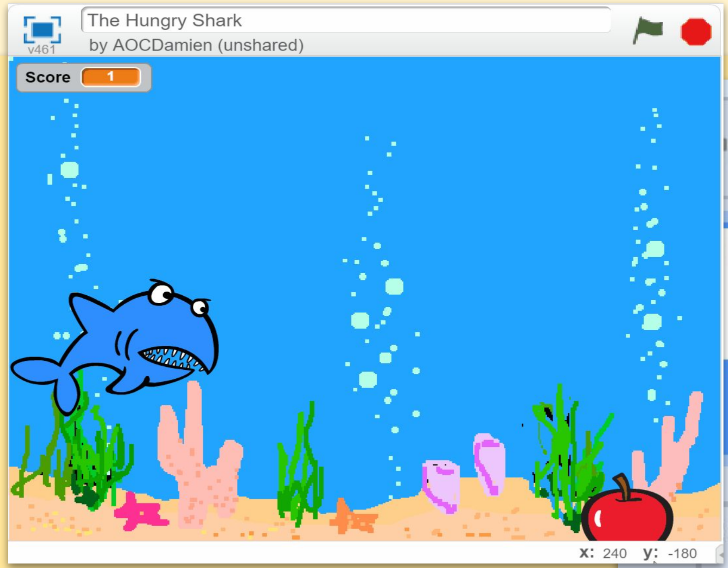
Pair Programme

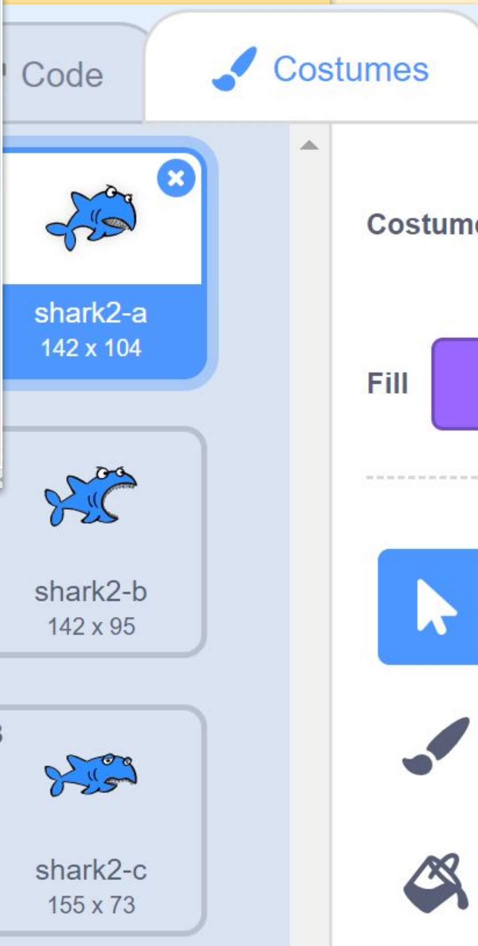
Help a Friend, Make a Friend!

Game Progress 18%

Costumes

*How many costumes does the **Shark** have?*
What are costumes?
How can we use them to make our game
More interesting and appealing to the user?

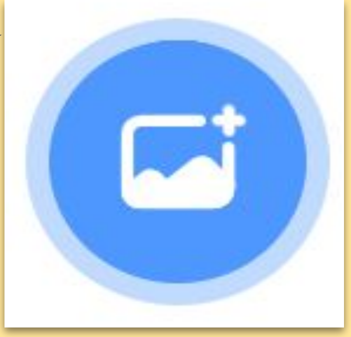




Which costumes do we need for our game?

Backdrop

Now click on **'Choose backdrop'** and add the **Underwater1,2 or 3** backdrop (or any backdrop of your choice) to the **Stage** of your project.

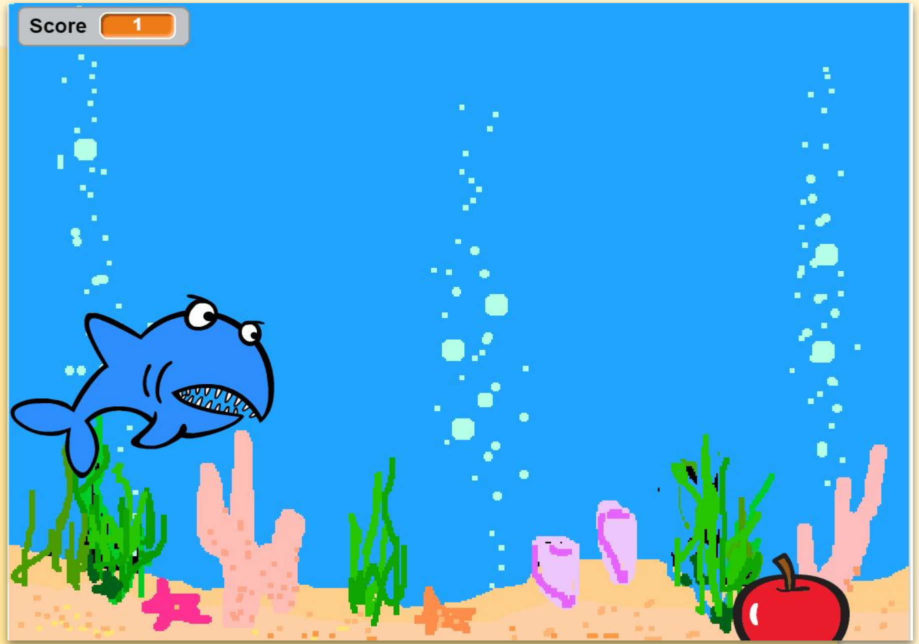


Pair Programme

Help a Friend,
Make a Friend!

Game Progress 32%

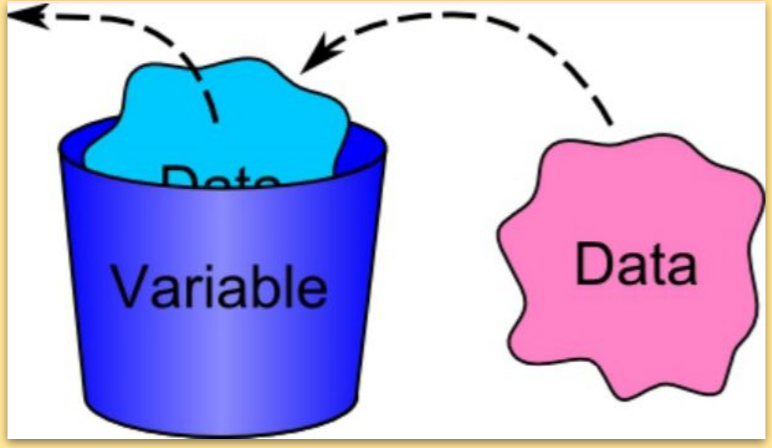
Your project should now look like this.



What does the word vary mean?

What is a variable?

A **variable** is a place to hold a number or letters so it can be used by the computer.

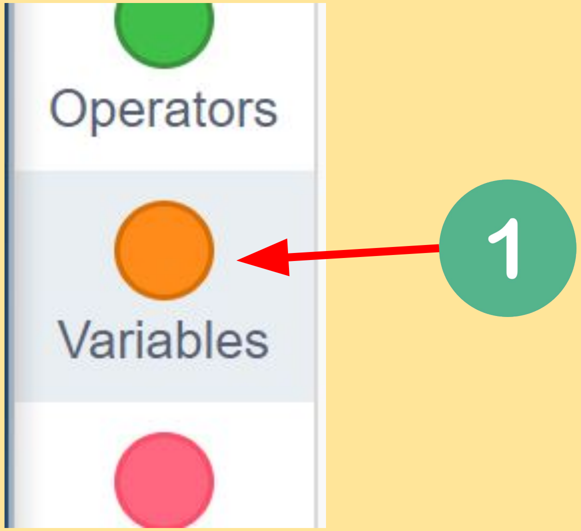


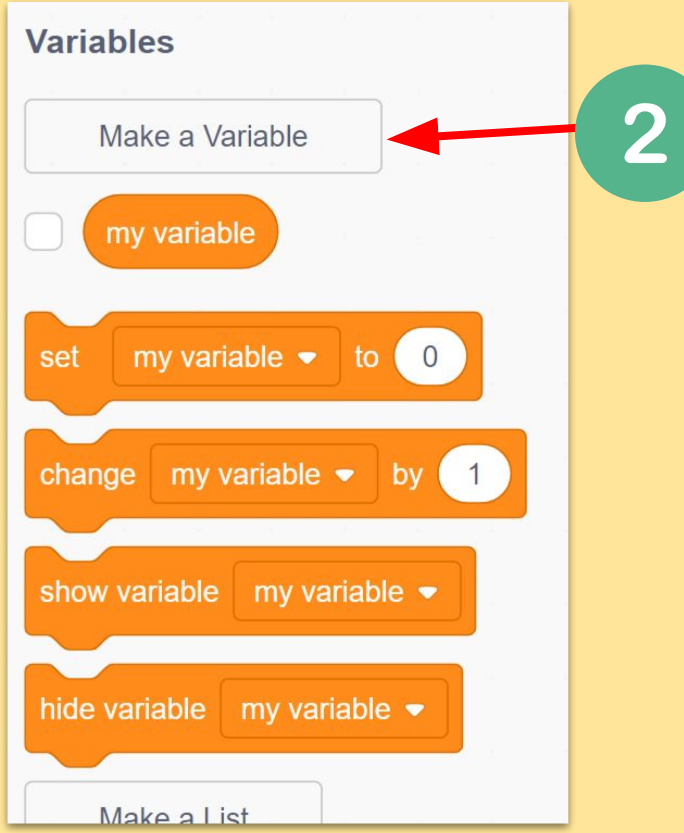
The value of the data which the variable is holding can be any value and it can be changed at any time. It is called a variable because its **value** can **vary**.

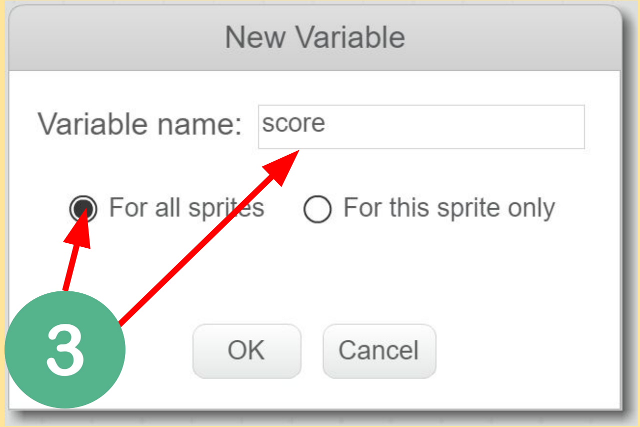
Our game could have many variables but we need a **score variable**.

Making a variable

Select the stage and in the **Data** section make a variable called *Score.* Make sure you make this for all sprites.

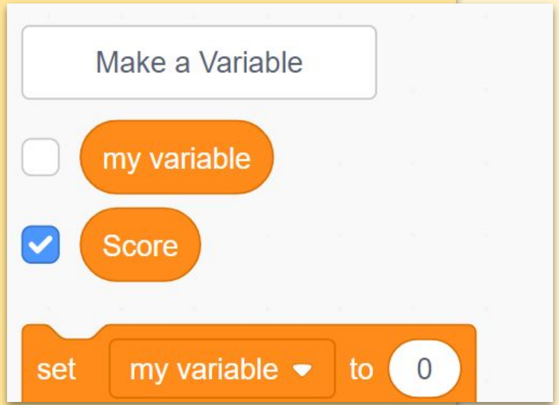



Variables
Make a Variable
2
□ my variable
set my variable ▾ to 0
change my variable ▾ by 1
show variable my variable ▾
hide variable my variable ▾
Make a List

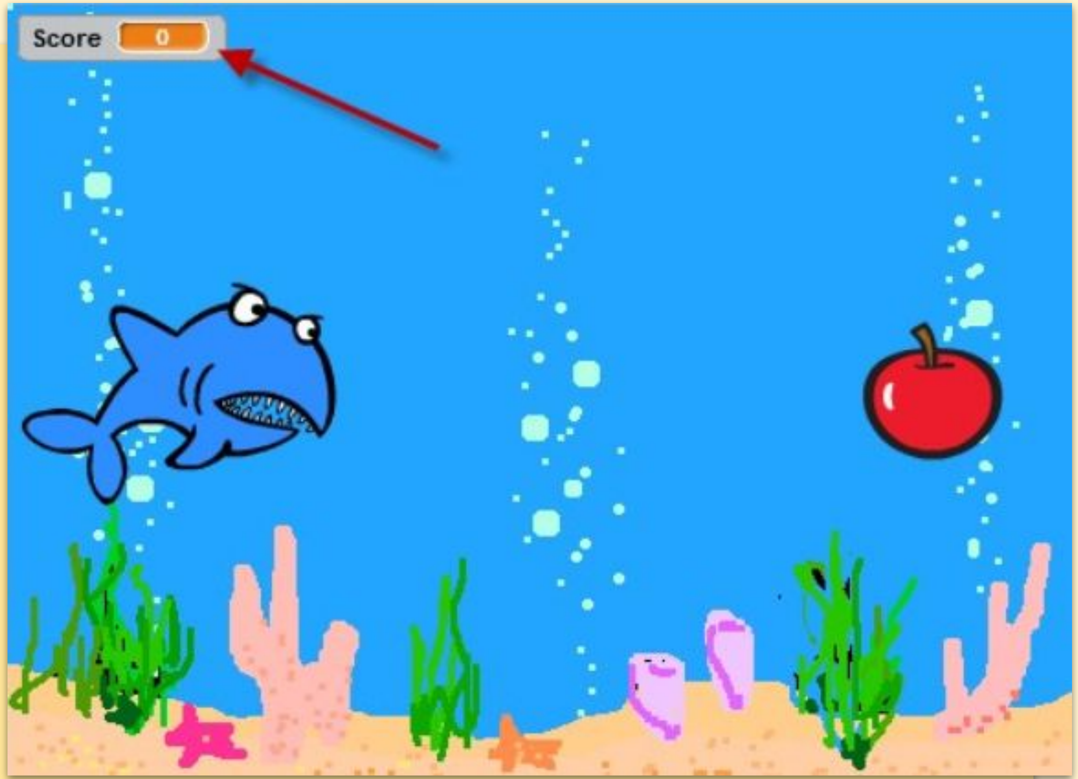

New Variable

Variable name: score

● For all sprites ○ For this sprite only



You should have now the *Score* variable displayed on your project, as shown in this image.

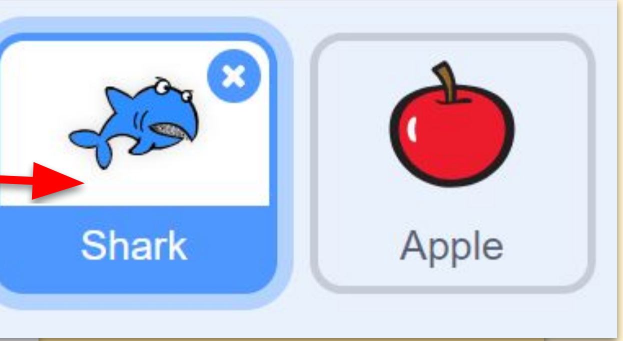


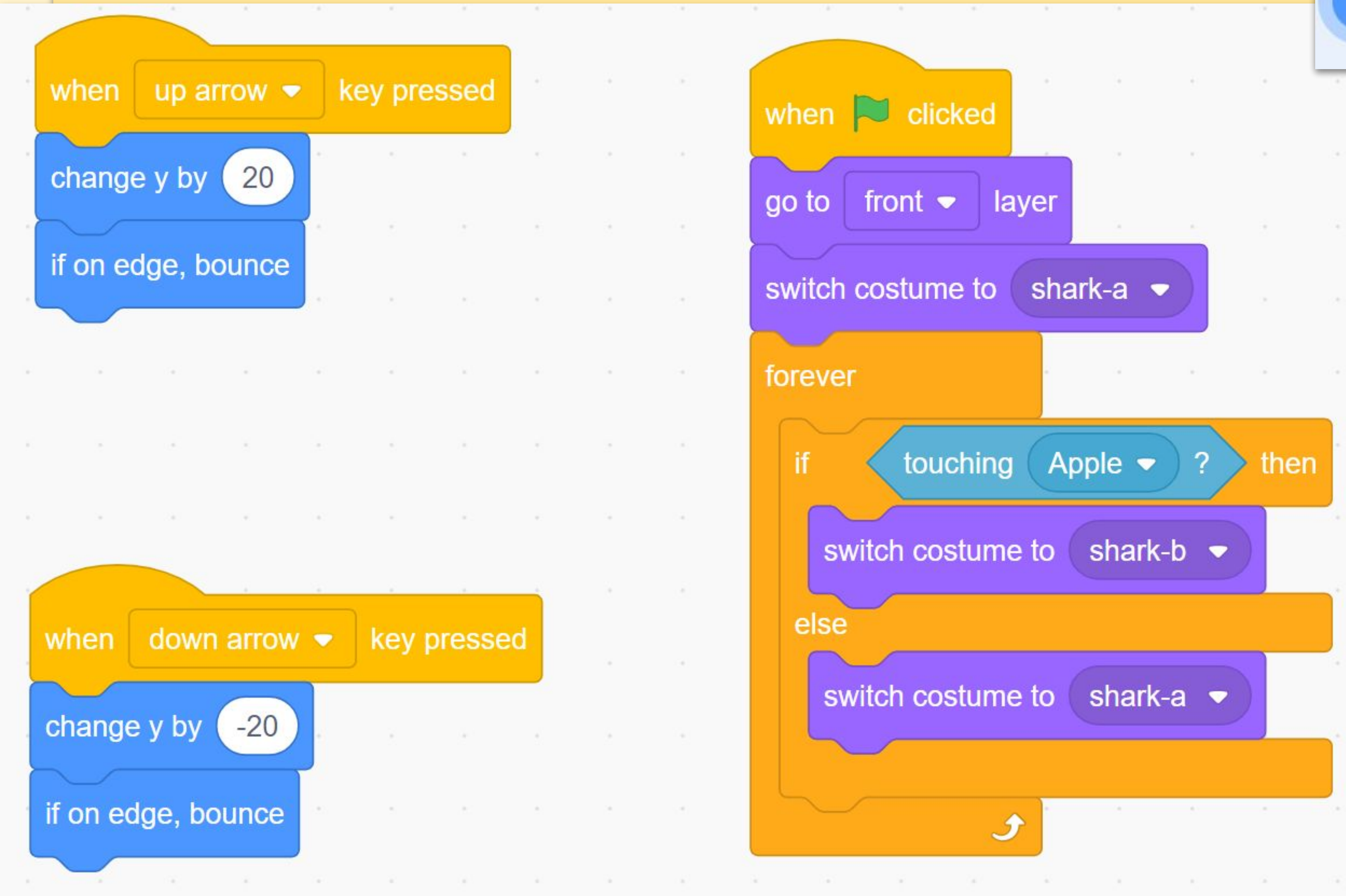
If you got this far, now is a good time to **pair programme** - helping others around you who might be stuck!

Don't go any further whenever you see this sign!

Game Progress 50%

Click on the **Shark** sprite to select it, and add the following script to the **Shark.**
Before you do, let's chat about what everything does.



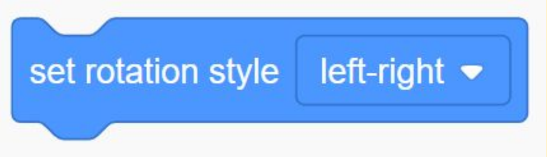


```
when up arrow ▾ key pressed
change y by 20
if on edge, bounce

when down arrow ▾ key pressed
change y by -20
if on edge, bounce

when 🏁 clicked
go to front ▾ layer
switch costume to shark-a ▾
forever
    if touching Apple ▾ ? then
        switch costume to shark-b ▾
    else
        switch costume to shark-a ▾
```

Pair Programme

Help a Friend,
Make a Friend!

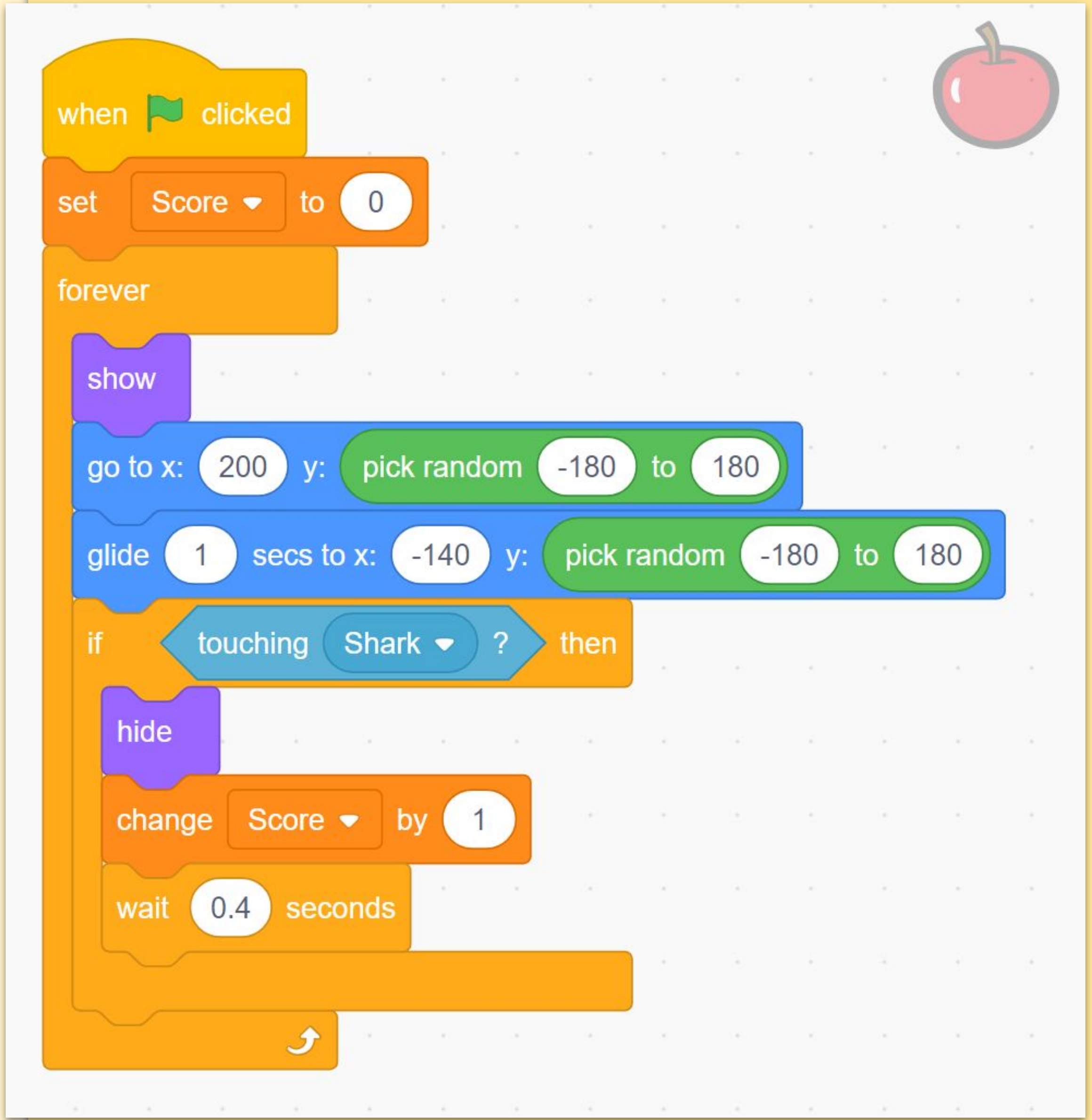
If your **Shark** is facing the wrong way, use this block



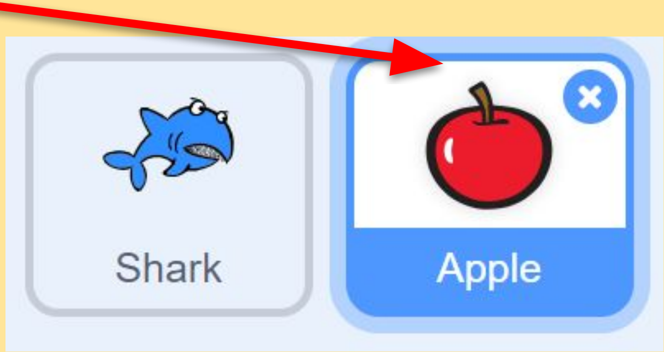
```
set rotation style left-right ▾
```

Game Progress 🟩 75%

Click on the **Apple** sprite to select it,
and add the following script to the **Apple.**



```
when 🏴 clicked
set Score ▾ to 0
forever
    show
    go to x: 200 y: pick random -180 to 180
    glide 1 secs to x: -140 y: pick random -180 to 180
    if touching Shark ▾ ? then
        hide
        change Score ▾ by 1
        wait 0.4 seconds
    end
end ↺
```



Pair Programme

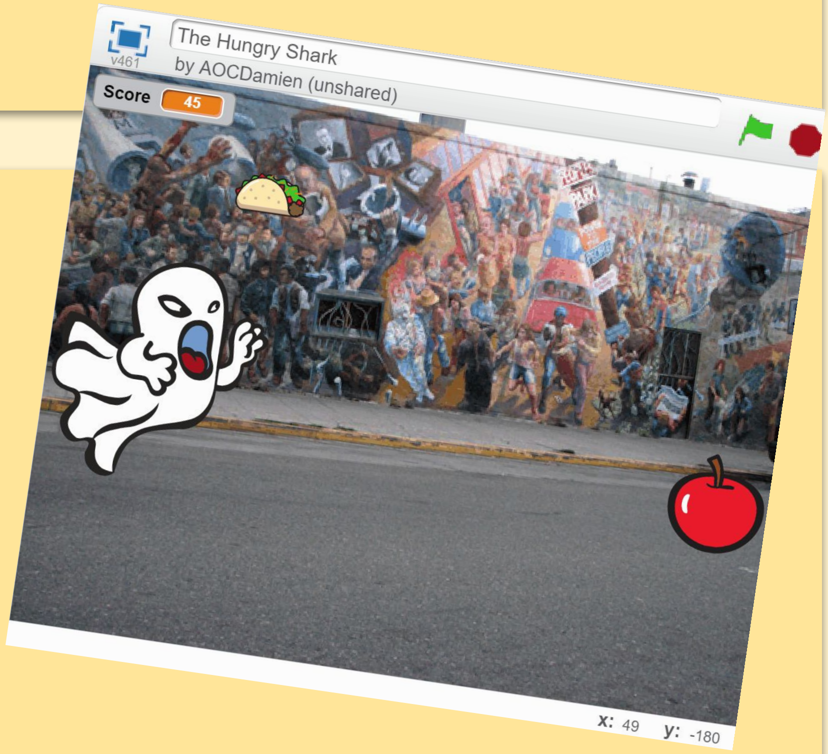
Help a Friend,
Make a Friend!

Game Progress 90%

Game Time!

Instructions

- Press the **Green Flag** to start.
- Use the **Up and Down arrow keys** on your keyboard to move the Shark and catch the Apple.
- Each time the shark “eats” an apple, the Score is increased by 1.
- A new apple **spawns** when the old one is eaten.



Challenges

Once you’ve finished Hungry Shark as per the example, it’s now time to customise:

- Try adding another food item that subtracts from the score.
- Add a feature that ends the game if you go below a certain score. You may need to broadcast here.
- Customise your game so that it has another player or enemy.
- Try having the background change every time the shark gets an apple.
- See if you can make the apple turn into a different food item (eg a burrito) when the shark eats it.

100%

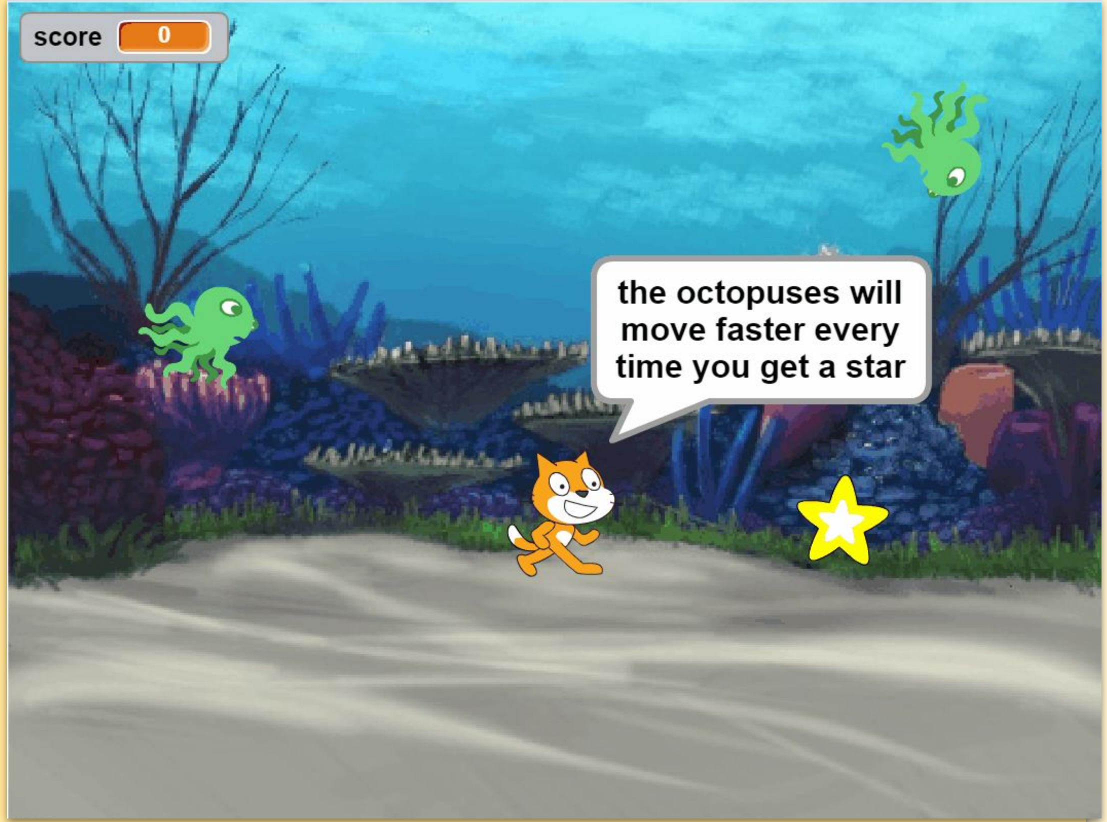
[the academy_of_code]

Starfish Hunter

Lesson 4 - Programming Games with Scratch II

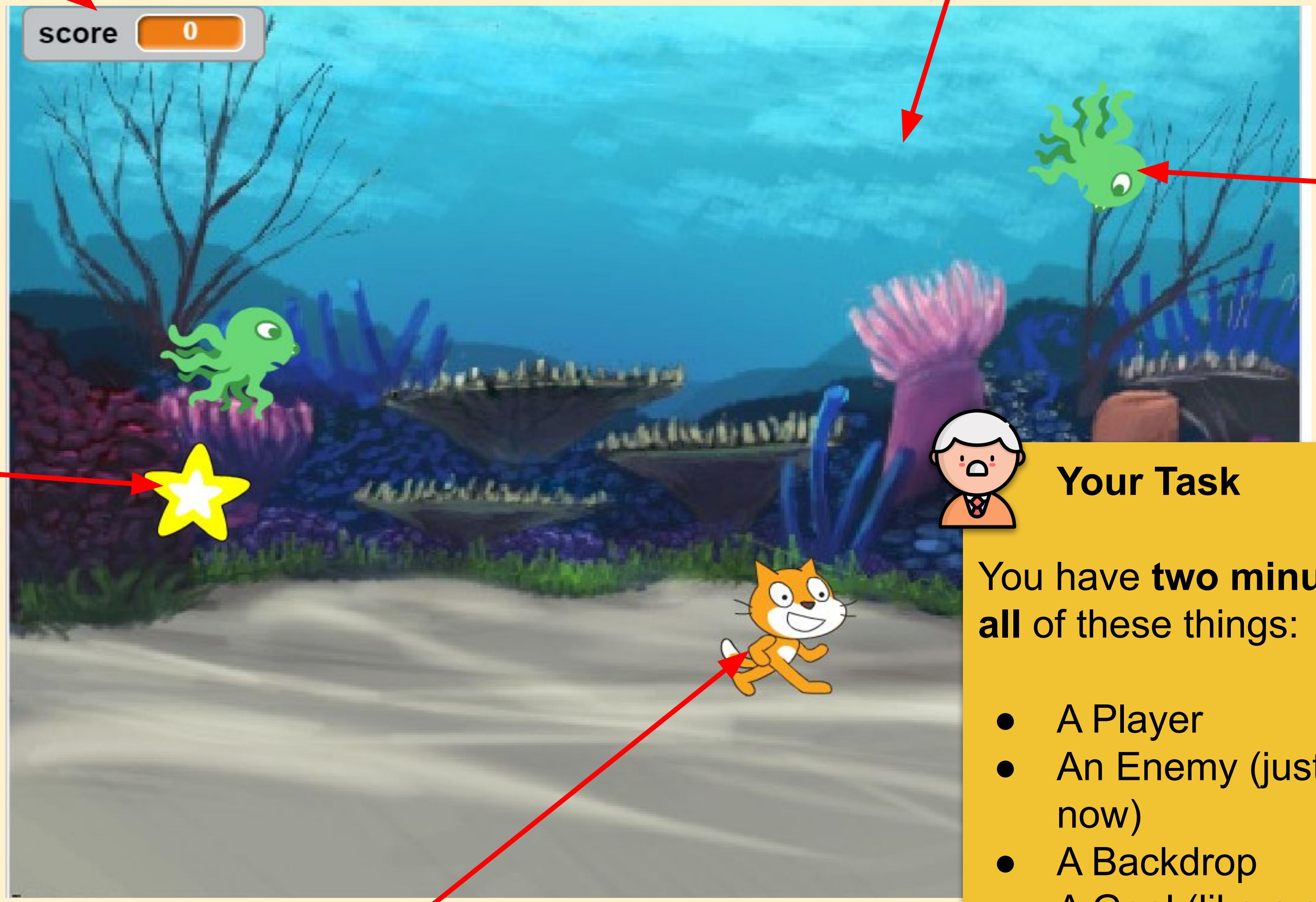
Learning Outcomes:

- Learning about the scratch IDE
- Learning how to make sprites follow the mouse
- Using Random
- Learning how to use the coding blocks in motion, looks, events, control



In our last project, we made **Hungry Shark** an interactive game that used the Up and down keys. It had a **player controlled sprite** and a **goal**.

Today we're going to make a game called **Starfish Hunter,** a game which has a player controlled sprite and a goal but also enemies. Unlike **Hungry Shark**, the player uses the **mouse** and not the keyboard to control the sprite.



Score Variable

A Backdrop

Enemies (something we must avoid!)

A Goal to collect and score points

Player Character (controlled by mouse)

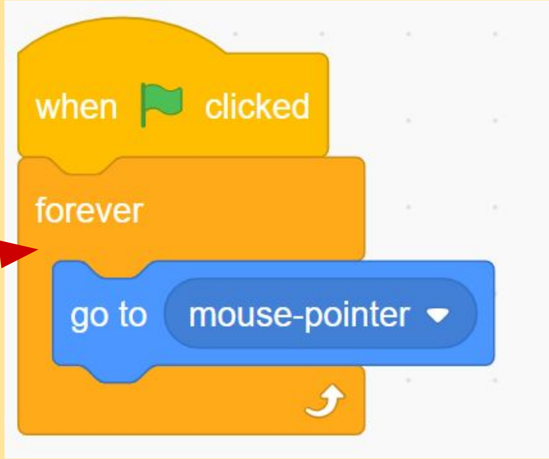
Your Task

You have **two minutes** to get **all** of these things:

- A Player
- An Enemy (just one for now)
- A Backdrop
- A Goal (like a star)
- A score variable

*"When the **green flag** is clicked,*
*make the sprite move **constantly (forever)***
in the direction of the mouse pointer"

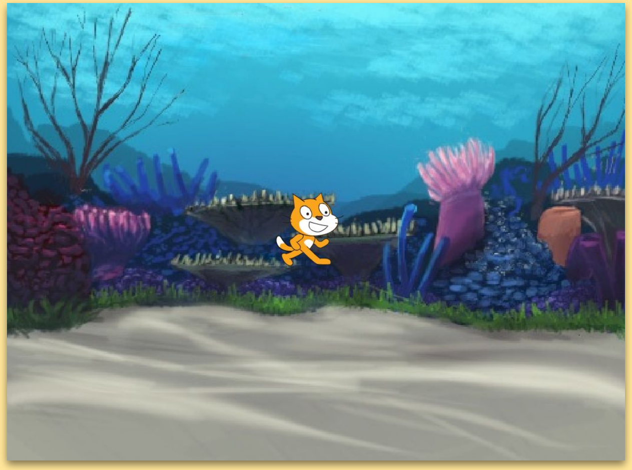
ENTIRE CAT CODE



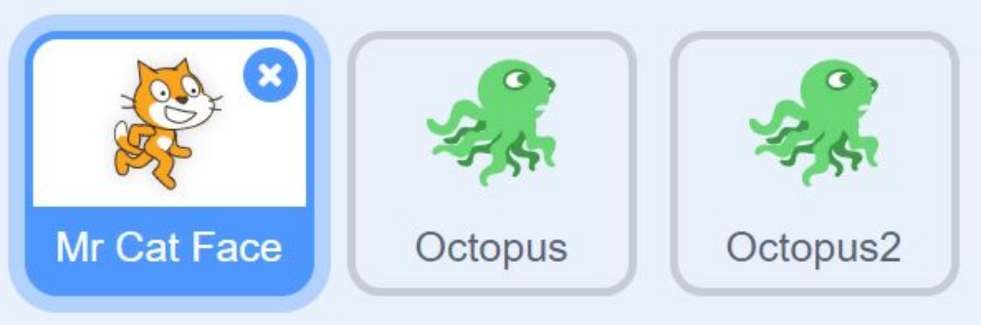
```
when [flag] clicked
forever
    go to mouse-pointer
```

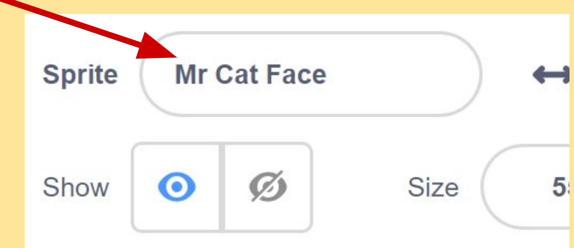
Next - Setting the scene

1 Pick an underwater or another scene for your game.



2 **Pick a player** and **two enemies** and **rename** them.

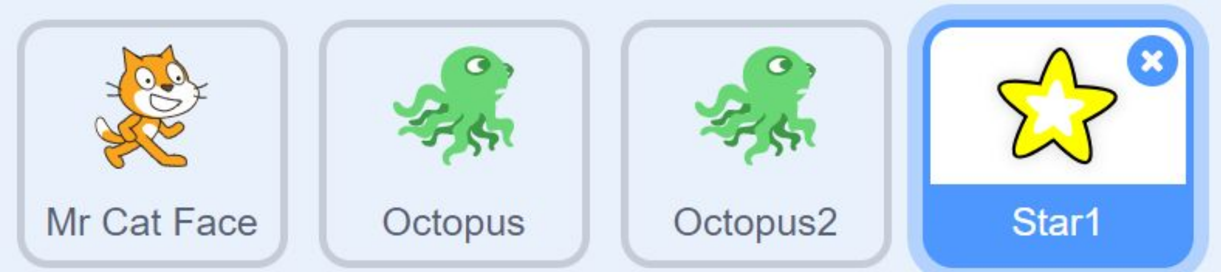


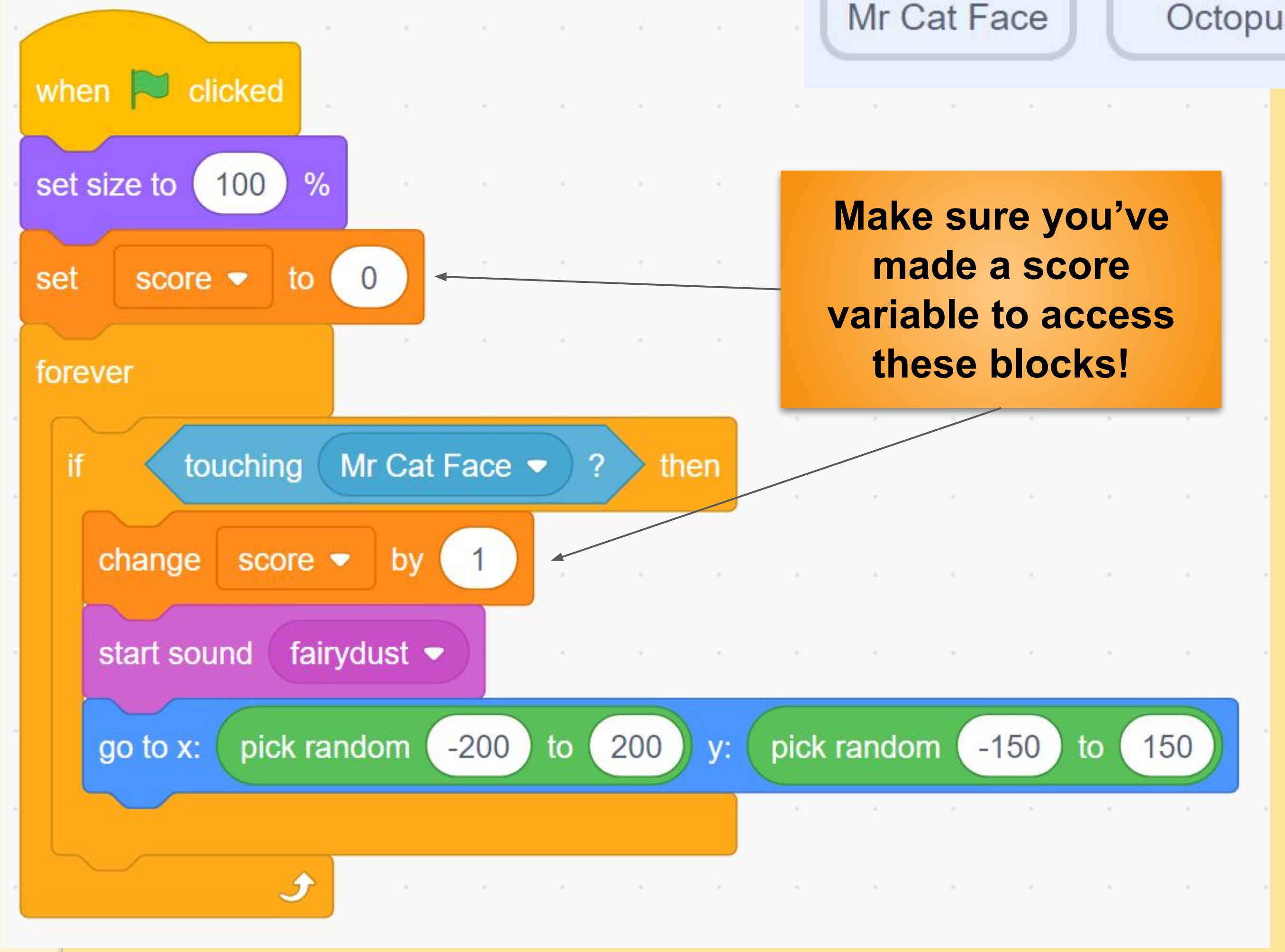


Sprite: Mr Cat Face

Game Progress 🟩 24%

Coding the Stars





```
when 🏁 clicked
set size to 100 %
set score to 0
forever
    if touching Mr Cat Face ? then
        change score by 1
        start sound fairydust
        go to x: pick random -200 to 200 y: pick random -150 to 150
```

Make sure you've made a score variable to access these blocks!

In *Starfish Hunter*, stars are treasure the player must collect. Every star collected adds to the **score variable**.

Add this script to the **goal/star**, but first, *what will it do*?

Game Progress 66%

Adding Collision Detection

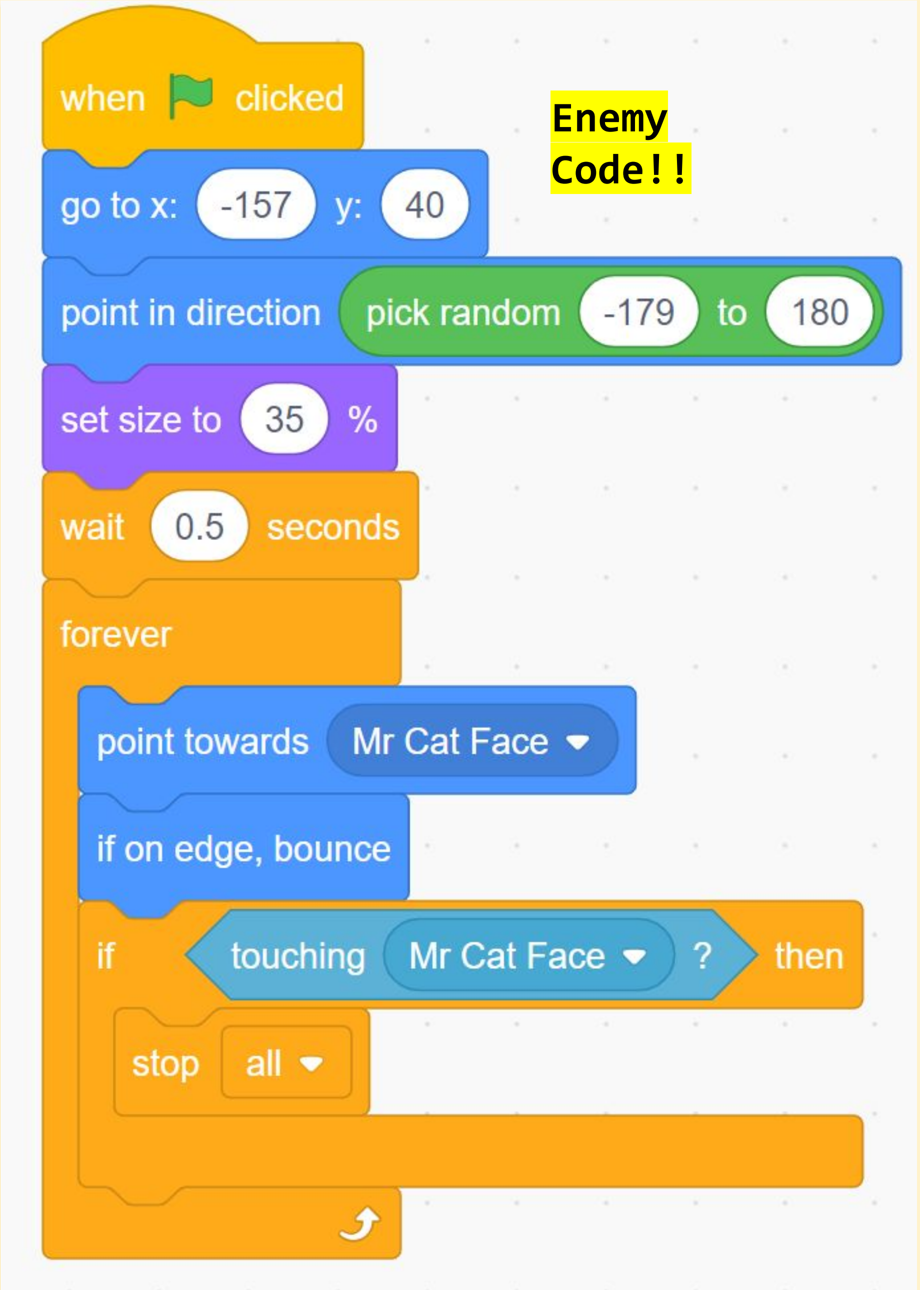
Your enemy’s code should look like this.

We need to add **Collision Detection** so that when the enemy hits the Player, the game will end. **Change** the above code so it looks like the one below.

“if then”

Every day we make decisions. If it’s raining, you use an umbrella. If it isn’t, then you don’t.

Computer programmes use **conditional statements** such as **“if then”. IF** the enemy touches the cat **THEN** stop all, if it isn’t, keep going.



```
when ⚑ clicked
go to x: -157 y: 40
point in direction (pick random -179 to 180)
set size to 35 %
wait 0.5 seconds
forever
    point towards Mr Cat Face ▾
    if on edge, bounce
    if < touching Mr Cat Face ▾ ? > then
        stop all ▾
```

Enemy Code!!

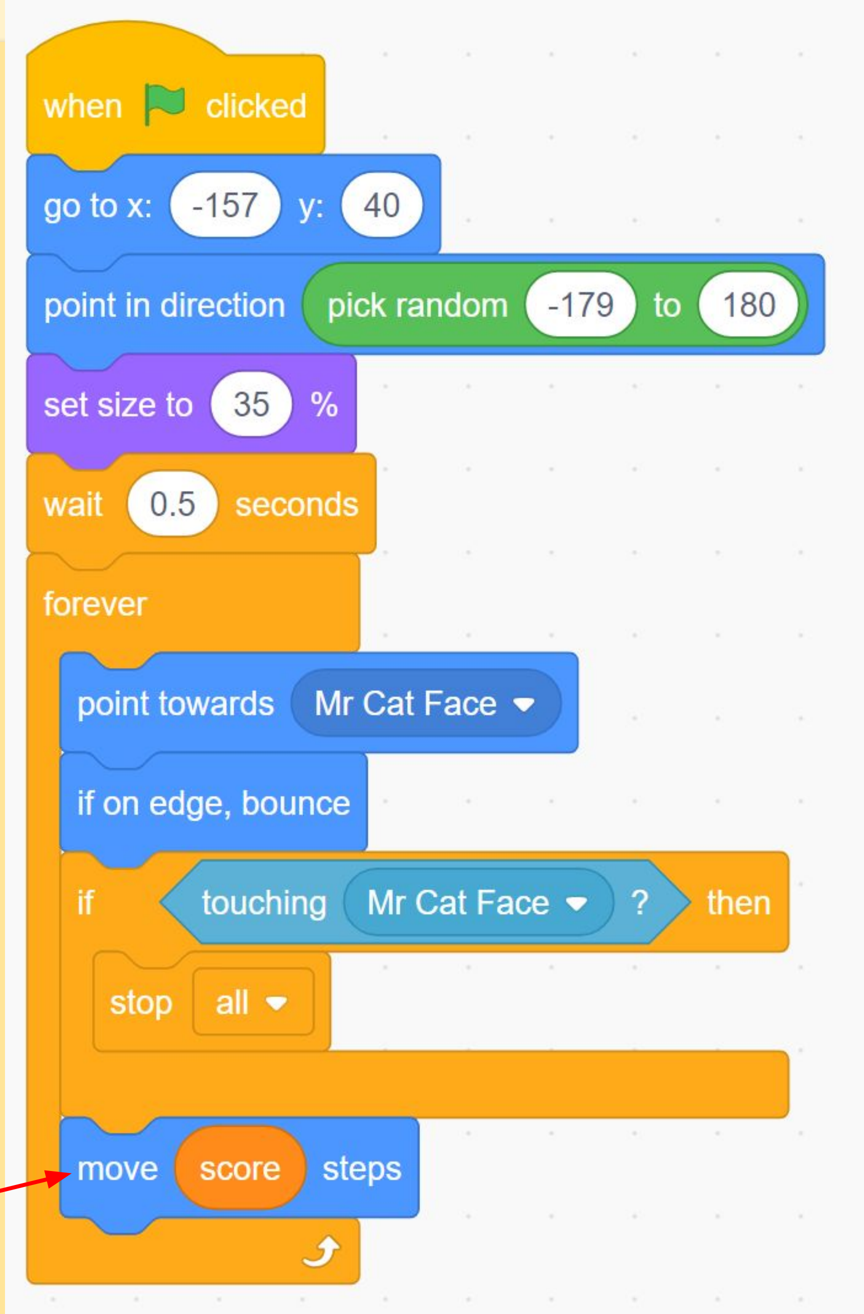
Game Progress 48%

Making The Enemies Move

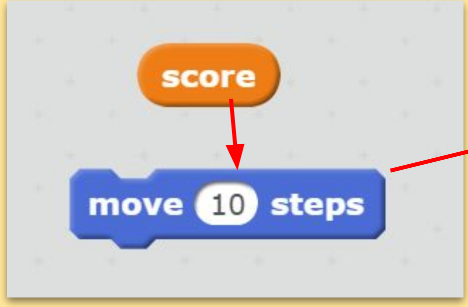
We can now test and change our game to make the enemy “better”, by this we mean to make it more challenging for the player.

We want our enemy sprites to get faster every time the player gets a starfish.

Find these blocks and add it to our **enemy** code.



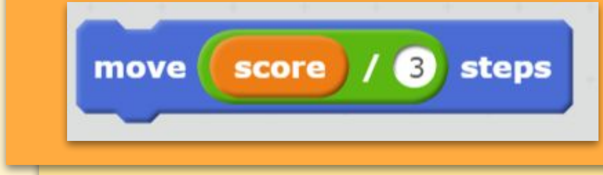
```
when 🏴 clicked
go to x: -157 y: 40
point in direction (pick random -179 to 180)
set size to 35 %
wait 0.5 seconds
forever
    point towards Mr Cat Face ▾
    if on edge, bounce
    if <touching Mr Cat Face ▾ ?> then
        stop all ▾
    move (score) steps
↺
```



```
(score)
move 10 steps
```

Expert Tip

To make the enemies go slower than the score, use these blocks



```
move ((score) / 3) steps
```

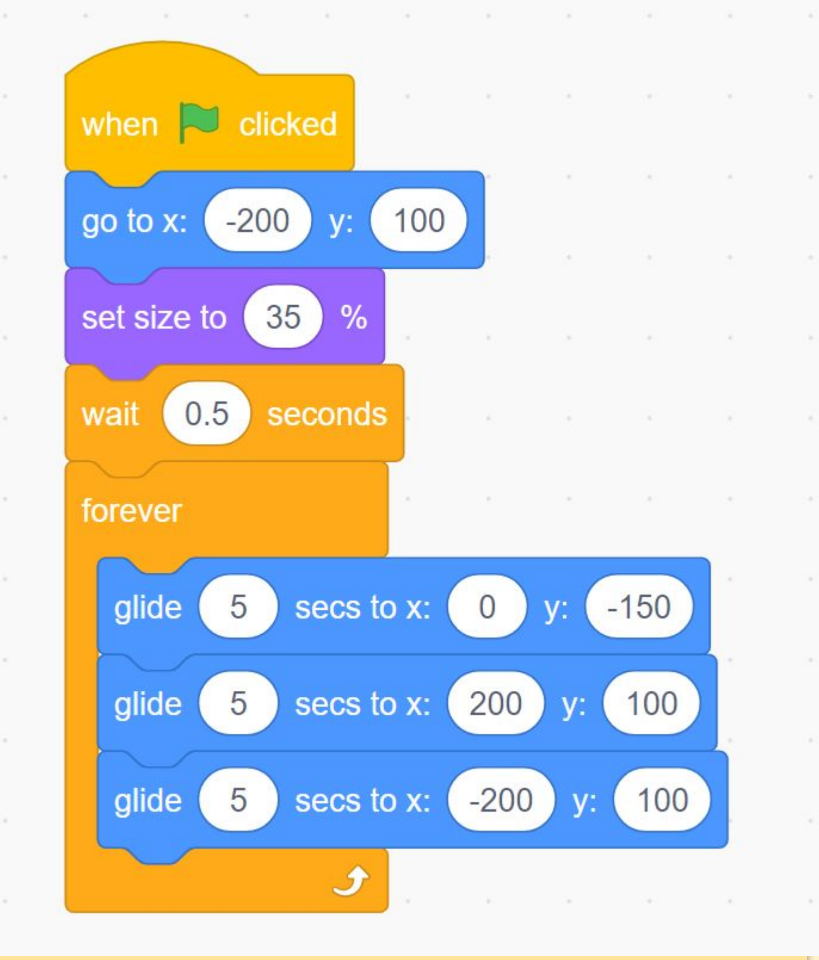
Game Progress 91%

Making a “Patrol” Octopus

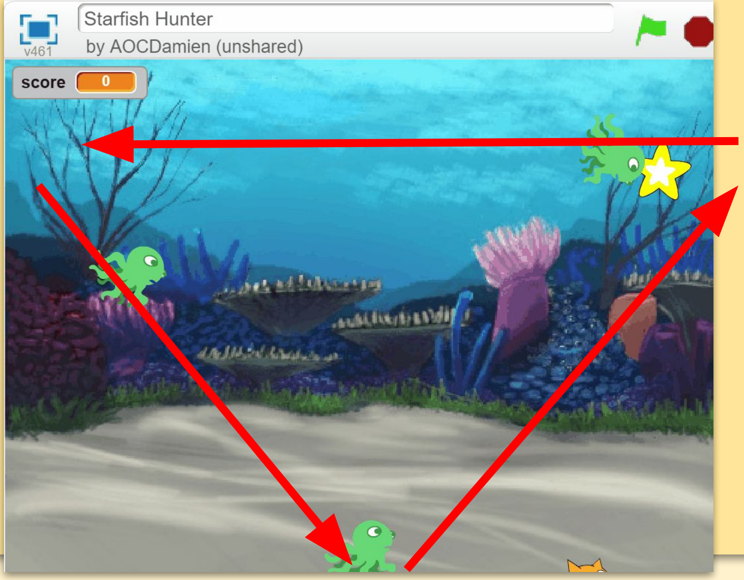
We’re going to have one or two enemies that follow the code similar from the previous pages, but let's make one that **doesn’t** go towards the mouse. It’s going to patrol back and forward.

Make a new enemy and give it a code.

These numbers will give the result below.

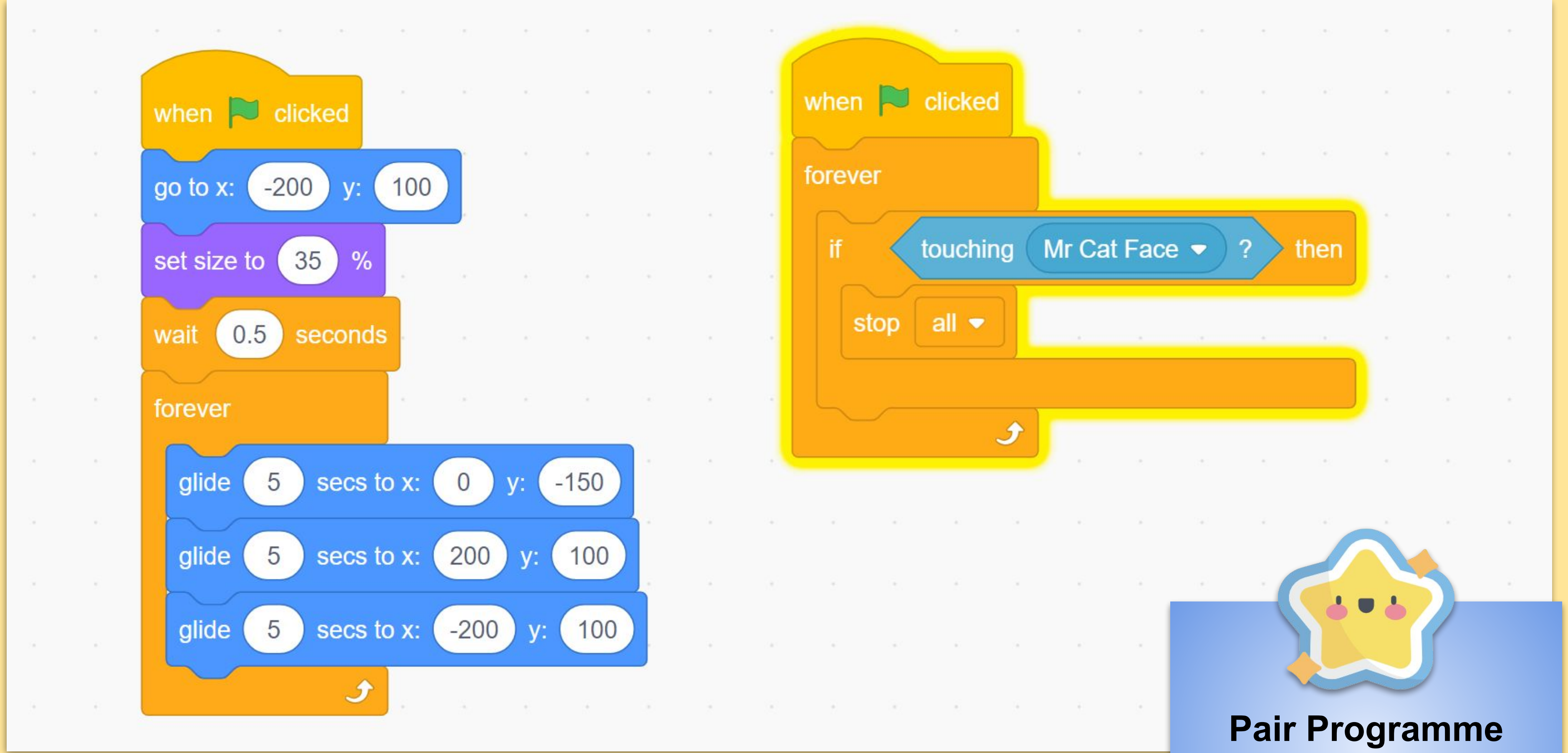


```
when 🏁 clicked
go to x: -200 y: 100
set size to 35 %
wait 0.5 seconds
forever
    glide 5 secs to x: 0 y: -150
    glide 5 secs to x: 200 y: 100
    glide 5 secs to x: -200 y: 100
↻
```



Finishing it off

To add **collision detection** to our final Patrol enemy, add the code below.



```
when 🏴 clicked
go to x: -200 y: 100
set size to 35 %
wait 0.5 seconds
forever
    glide 5 secs to x: 0 y: -150
    glide 5 secs to x: 200 y: 100
    glide 5 secs to x: -200 y: 100
↻
```

```
when 🏴 clicked
forever
    if touching Mr Cat Face ▾ ? then
        stop all ▾
↻
```

If you are finished, make sure to pair programme.
On big projects, coders work as part of a team.
Don’t leave anyone behind!

Pair Programme

Help a Friend,
Make a Friend!

Game Progress 🟩 100%

[the academy_of_code]

Snow Skatin’

Lesson 2 - Scratch - *Snow Skater*

Learning Outcomes

- Making *Snow Skater*
- *Customising your game*

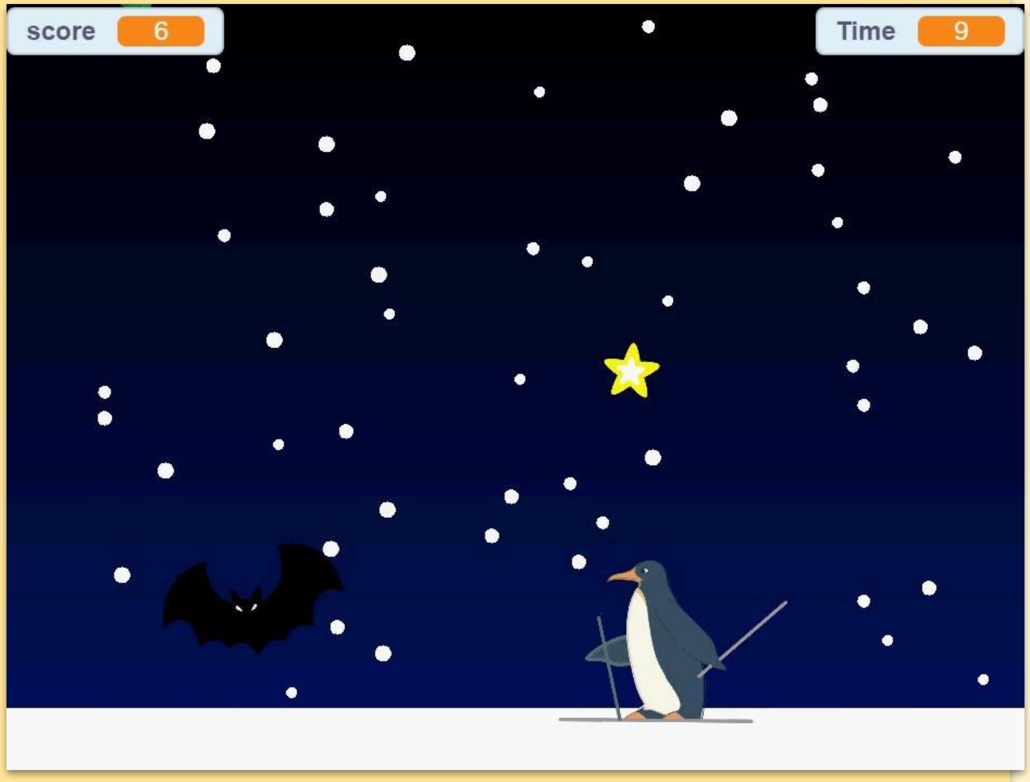
REMEMBER: Put up your hand. We love to help!

Snow Skater

In this lesson we're going to make a game which will use **keyboard input**.

How many sprites can you see in this?

How many variables can you see?

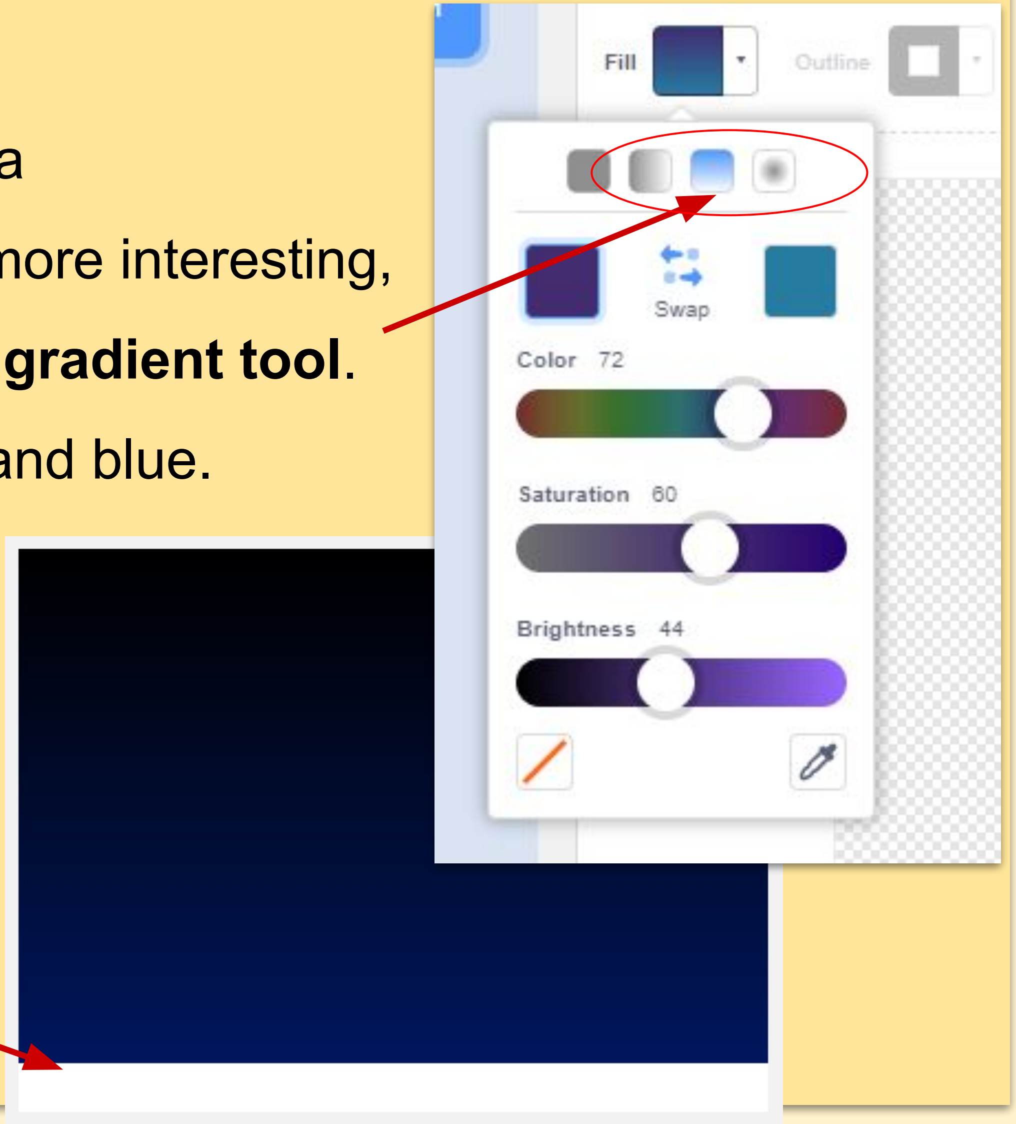


Making the Backdrop

1 Click on the Backdrop and then the Backdrops tab

2 Select the **Fill tool** and choose a background colour. To make things more interesting, you can blend two colours using the **gradient tool**. Choose two different colours, black and blue.

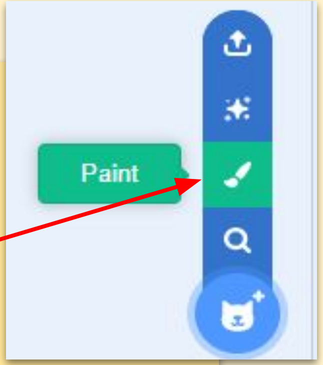
3 Now using the fill or rectangle tool, draw a white ice rink for your Player to skate on.


Fill
Outline
Swap
Color 72
Saturation 60
Brightness 44

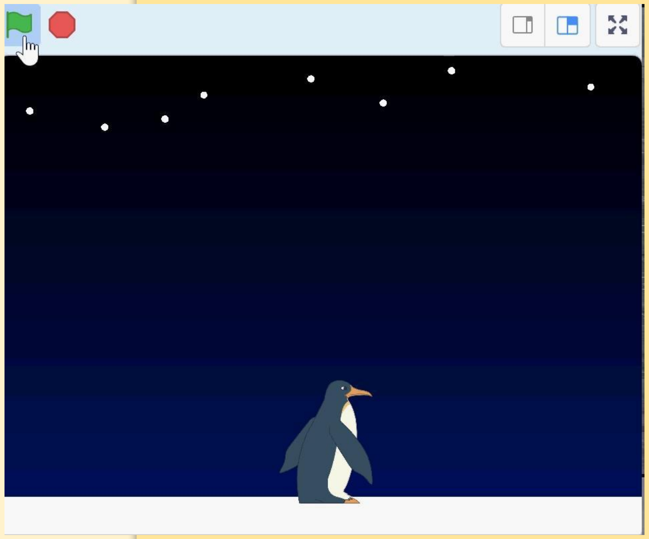
Game Progress 8%

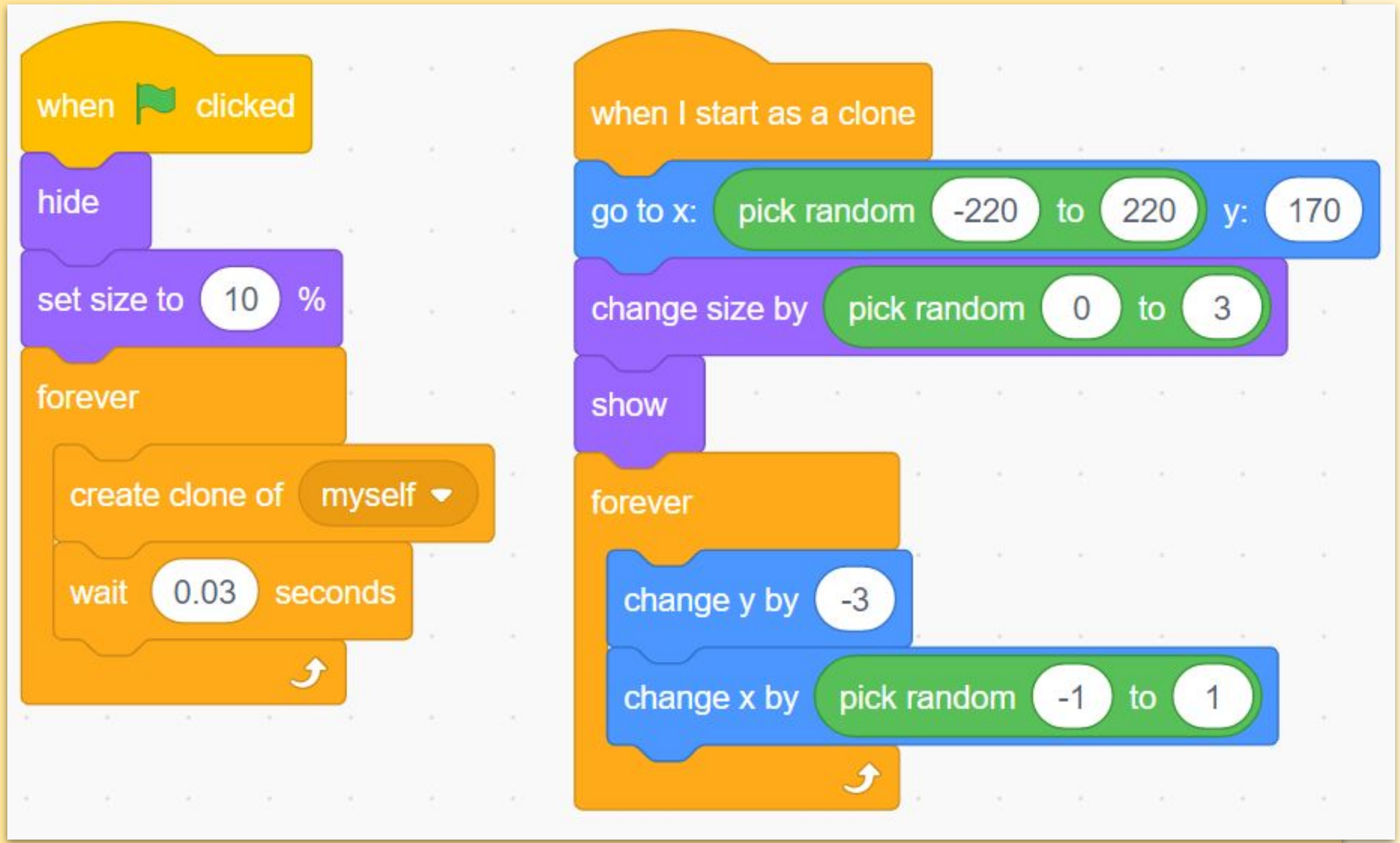
Cloning

We will make a snowflake sprite and **clone** it to give the effect of snowfall. Select the '*Paint New Sprite*' paintbrush.



Using the ellipse tool (circle), draw a **small** white circle in the middle.

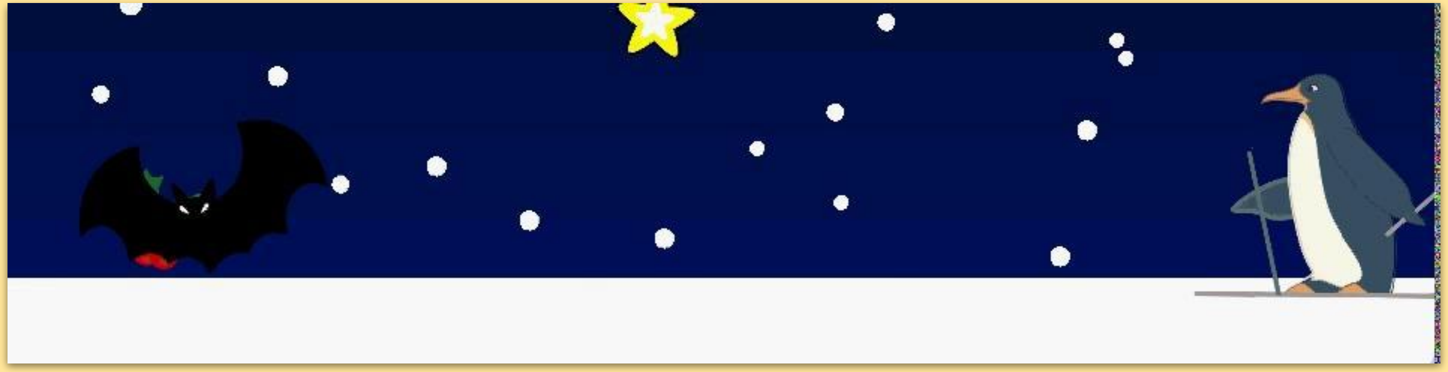


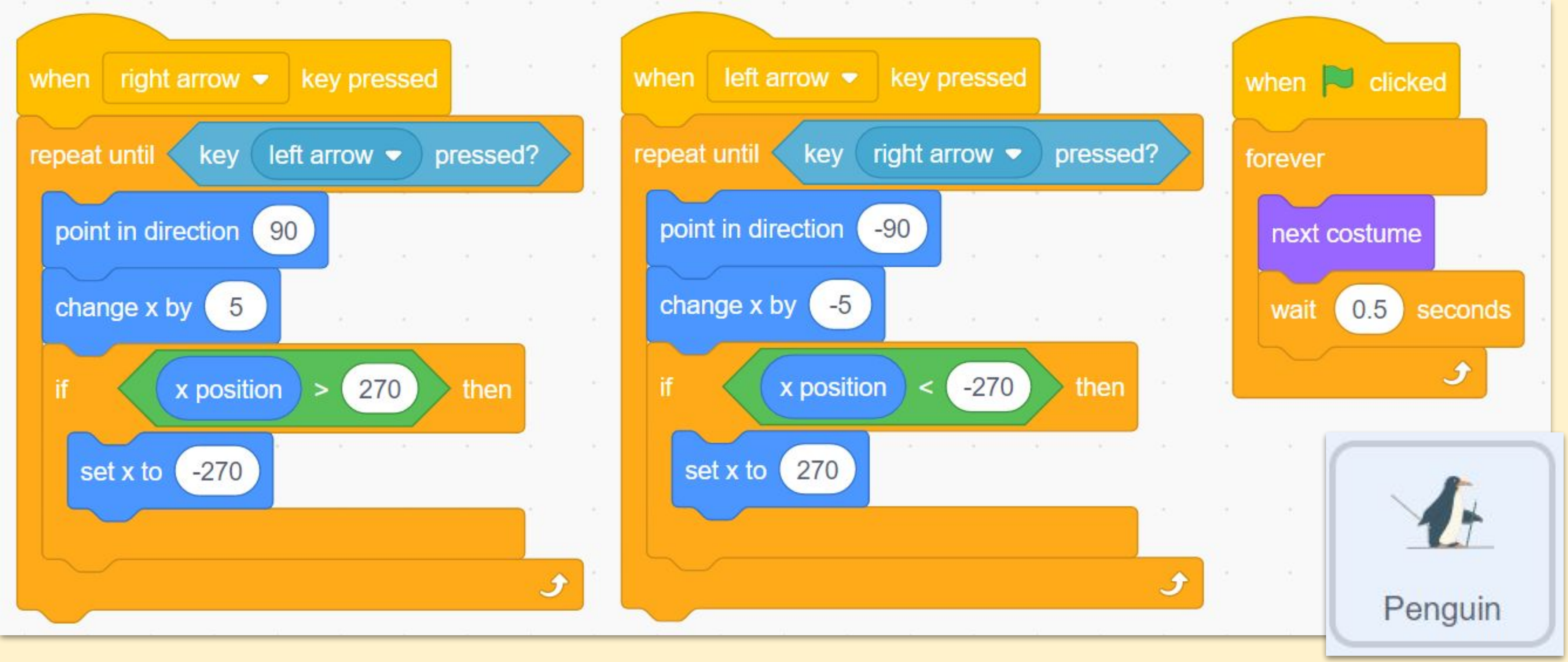


```
when 🏁 clicked
hide
set size to (10) %
forever
    create clone of (myself ▾)
    wait (0.03) seconds
```

```
when I start as a clone
go to x: (pick random (-220) to (220)) y: (170)
change size by (pick random (0) to (3))
show
forever
    change y by (-3)
    change x by (pick random (-1) to (1))
```

The Penguin’s Code





```
when right arrow ▾ key pressed
repeat until <key left arrow ▾ pressed?>
    point in direction (90)
    change x by (5)
    if <(x position) > (270)> then
        set x to (-270)
```

```
when left arrow ▾ key pressed
repeat until <key right arrow ▾ pressed?>
    point in direction (-90)
    change x by (-5)
    if <(x position) < (-270)> then
        set x to (270)
```

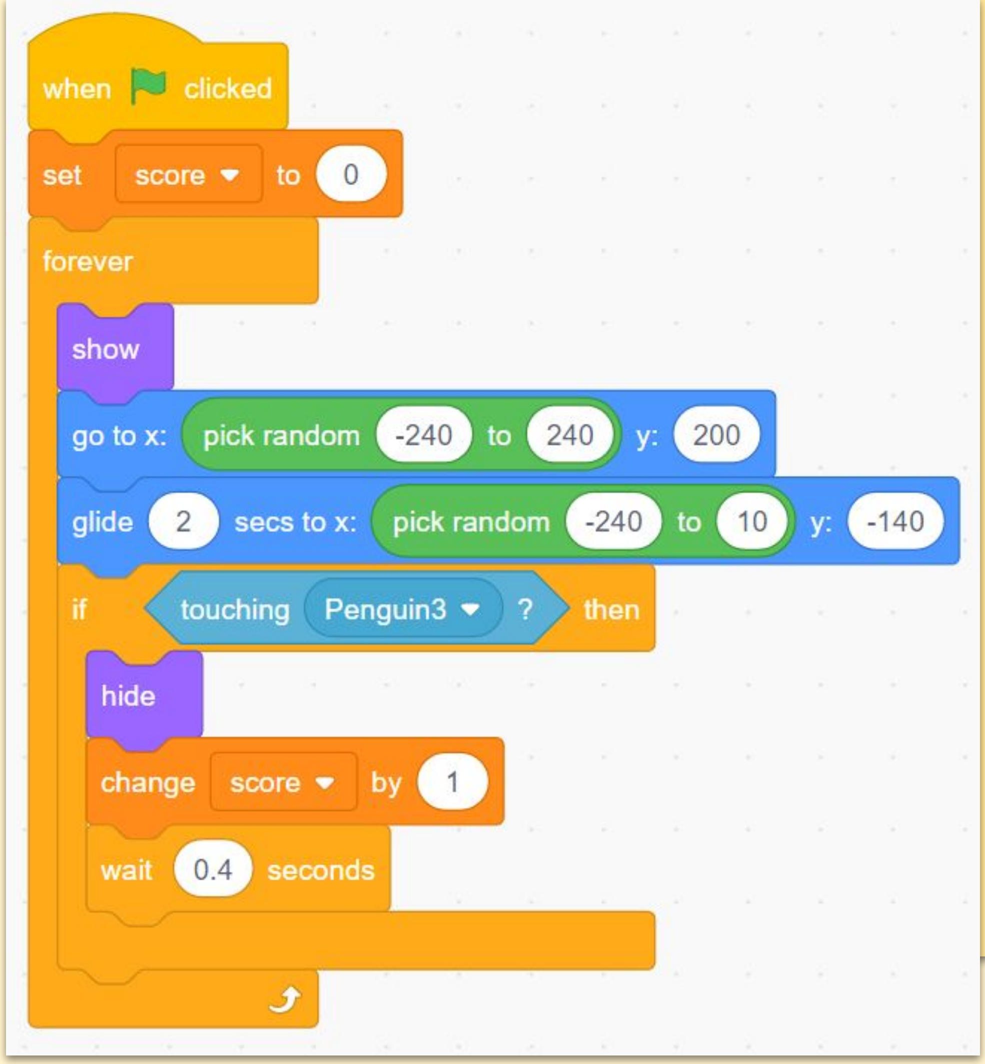
```
when 🏁 clicked
forever
    next costume
    wait (0.5) seconds
```

Coding the Goals (Stars/Holly)

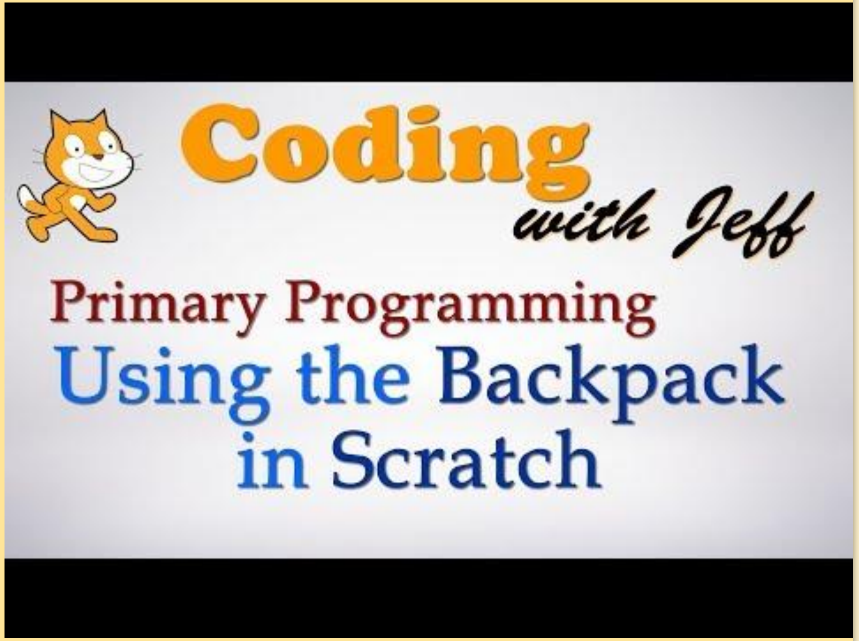
Have you seen a code like this before? Which game used a very similar code to this?

If you've found out which one, you can take the code from *that* game and put it in your **backpack**.

You'll need to make a **score variable**.



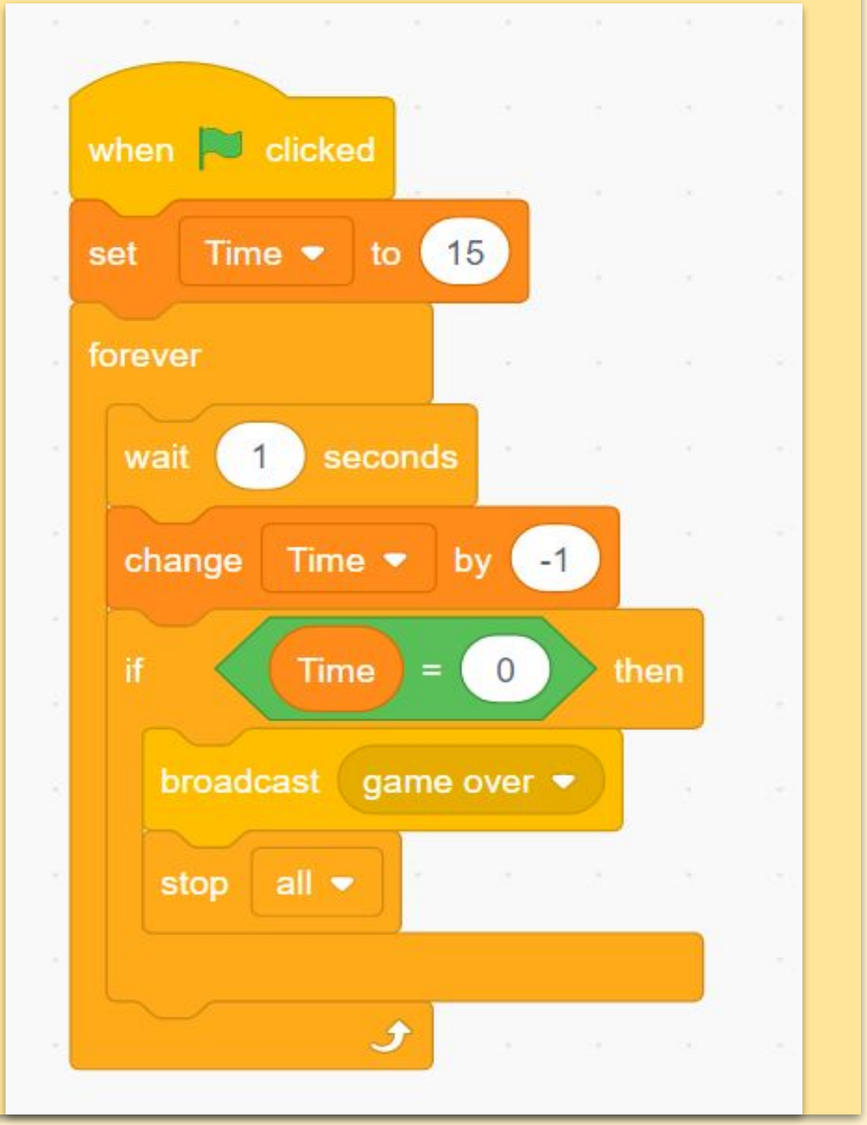
```
when ⚑ clicked
set score ▾ to 0
forever
    show
    go to x: pick random -240 to 240 y: 200
    glide 2 secs to x: pick random -240 to 10 y: -140
    if touching Penguin3 ▾ ? then
        hide
        change score ▾ by 1
        wait 0.4 seconds
    ↺
```



Game Progress 0%

Add a Time Variable

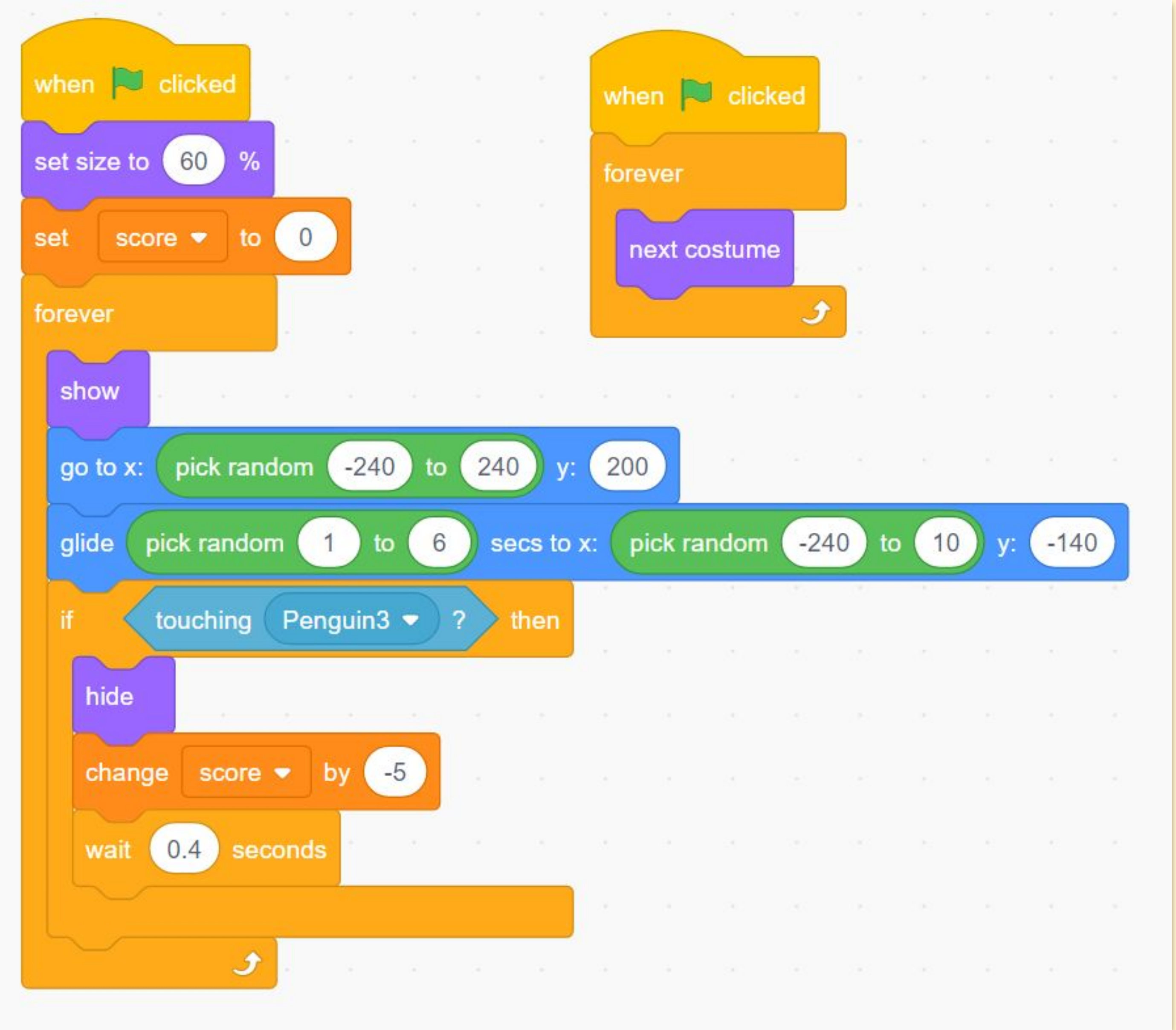
Make a **variable called time** (data section). Put the following code on the **stage** (although it could go anywhere and would work the same).



```
when 🏳 clicked
set Time ▾ to 15
forever
    wait 1 seconds
    change Time ▾ by -1
    if < (Time) = 0 > then
        broadcast game over ▾
        stop all ▾
↻
```

Coding the Enemy

Taking the Holy/Star code from your **backpack**, put it on a bat/enemy but change it so it looks like the code here.



```
when ⚑ clicked
set size to (60) %
set [score ▾] to (0)
forever
    show
    go to x: (pick random (-240) to (240)) y: (200)
    glide (pick random (1) to (6)) secs to x: (pick random (-240) to (10)) y: (-140)
    if <touching [Penguin3 ▾] ?> then
        hide
        change [score ▾] by (-5)
        wait (0.4) seconds
```

```
when ⚑ clicked
forever
    next costume
```

Game Progress 🟩 00%

[the academy_of_code]

‘Basic’ Maze Game

Lesson 5 - ‘Basic’ Maze Game

Learning Outcomes:

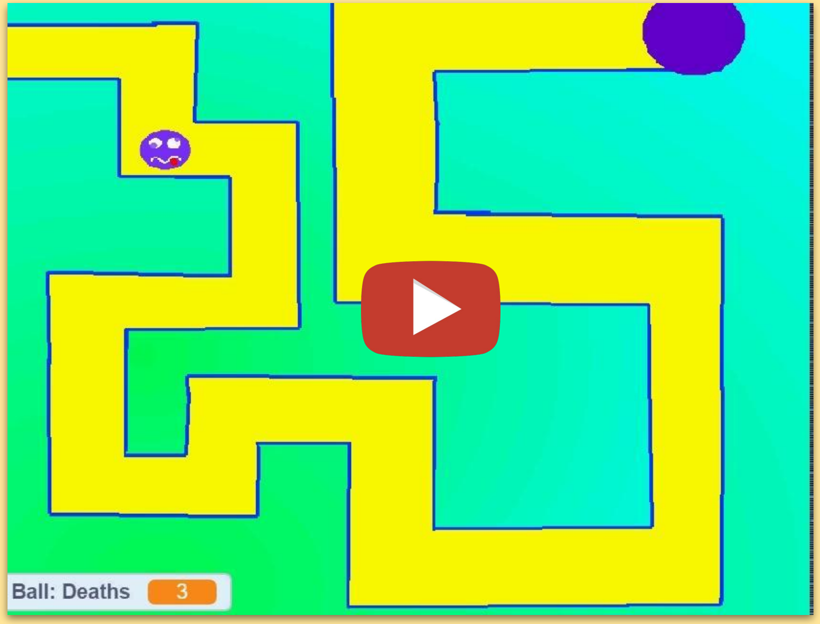
- To make a basic maze game based on a simple template
- Making multiple levels
- Using the ‘touching colour’ code block
- Having intro, direction and multiple level backdrops

The Final Product

Our plan for this was to make a **Scary maze game** where the player would direct the sprite around a maze for a few levels leading to a scary picture at the end, but you can chose to follow your own path!

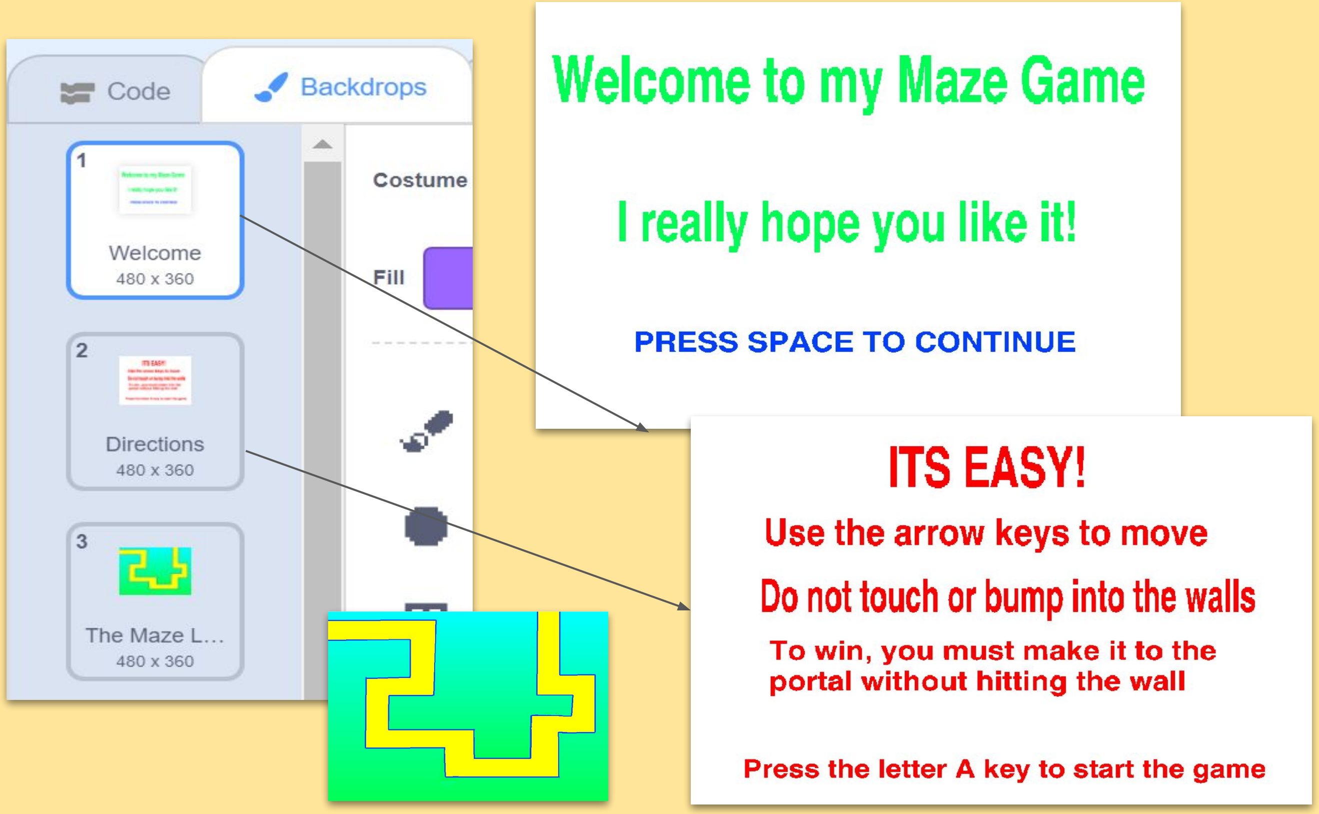
As with any good game, start it out easy and have it get harder as the game progresses.

AND REMEMBER: Make sure your game **works**, that it is **completable** and that it is **bug-free**, or else nobody will want to play it!



Making an Intro Screen and Starting Maze

1 Make a *Welcome*, *Directions* screen and Level one of your maze. You character will be controlled by the arrow keys and must be small enough to actually fit through your maze.



Welcome to my Maze Game

I really hope you like it!

PRESS SPACE TO CONTINUE

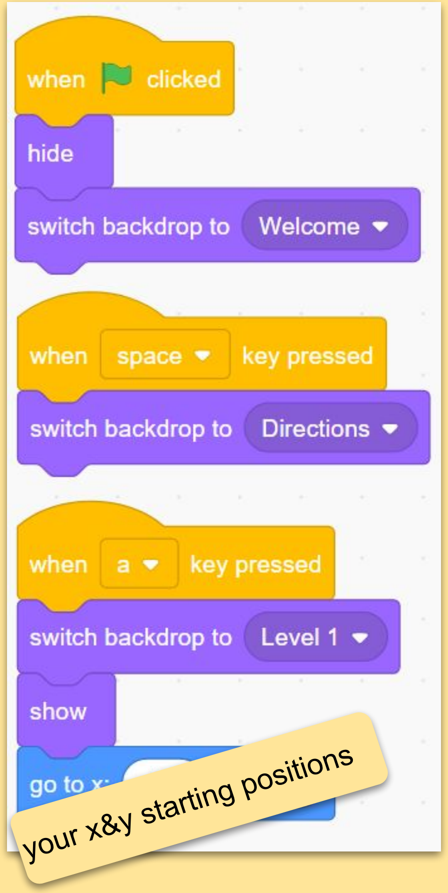
ITS EASY!

Use the arrow keys to move

Do not touch or bump into the walls

To win, you must make it to the portal without hitting the wall

Press the letter A key to start the game



```
when ⚑ clicked
hide
switch backdrop to (Welcome ▾)

when (space ▾) key pressed
switch backdrop to (Directions ▾)

when (a ▾) key pressed
switch backdrop to (Level 1 ▾)
show
go to x: (  ) y: (  )
```

your x&y starting positions

Pick or Design a Player Sprite

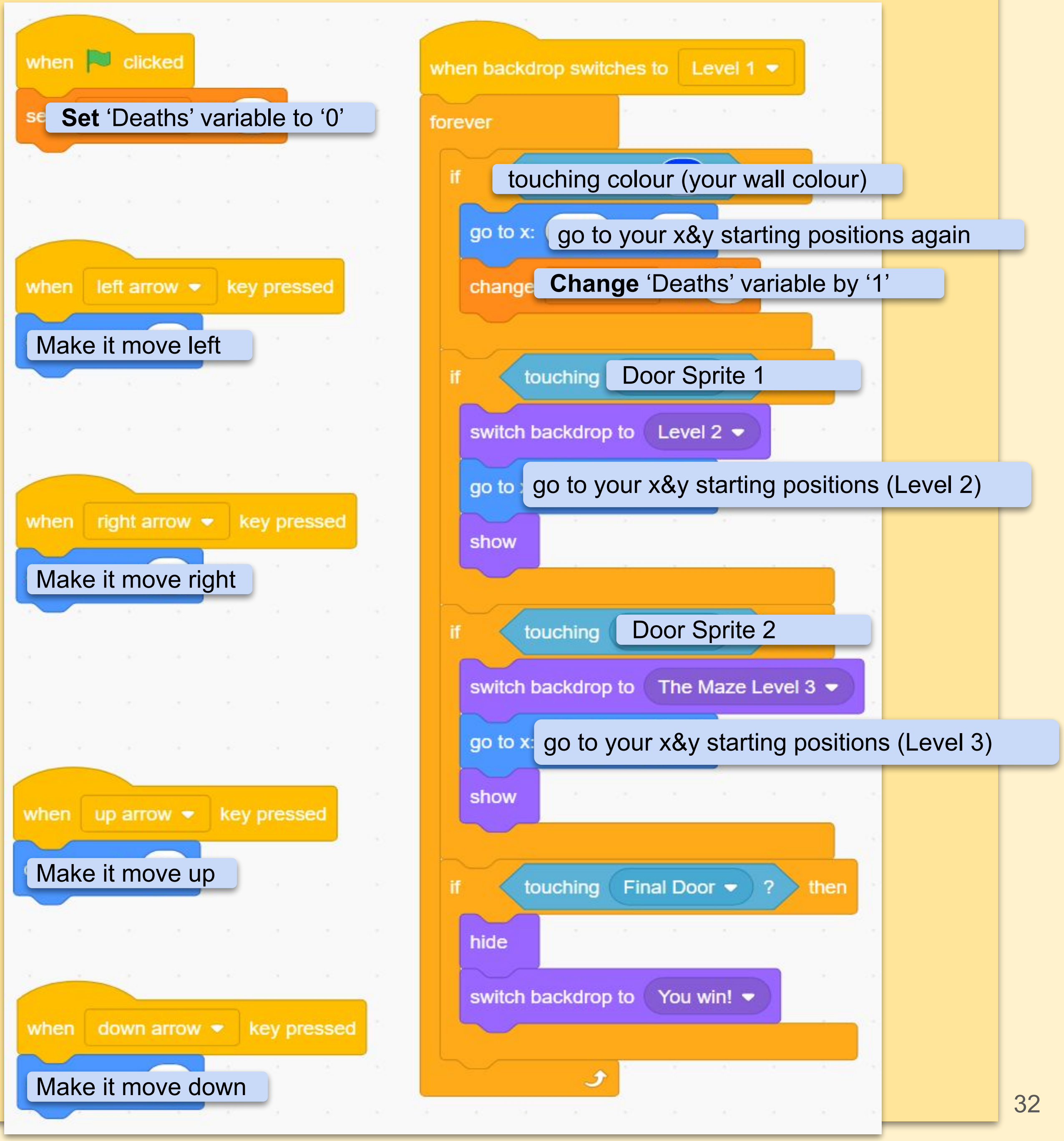
2 Once you've picked/ designed a player, add the code on the right

NB The numbers in '**go to x: ( ) y: ( )**' will be the part of your maze where the sprite starts from!

Getting things moving - More Player Code

We need to get the **player**'s sprite moving but also **set conditions** on what happens if it hits a wall or a portal (to go to the next level).

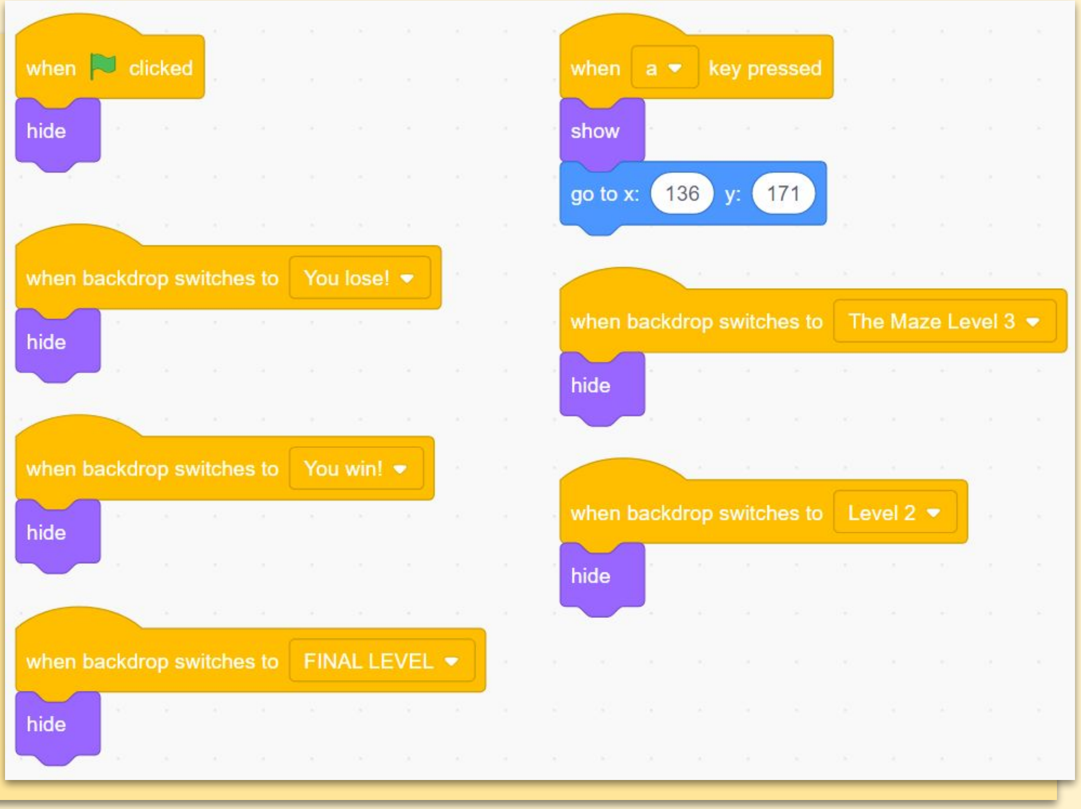
Try and figure out the pseudocode here on your **player sprite** to get things moving.



```
when ⚑ clicked
se Set 'Deaths' variable to '0'

when left arrow ▾ key pressed
Make it move left

when right arrow ▾ key pressed
Make it move right

when up arrow ▾ key pressed
Make it move up

when down arrow ▾ key pressed
Make it move down

when backdrop switches to Level 1 ▾
forever
    if touching colour (your wall colour)
        go to x: go to your x&y starting positions again
        change Change 'Deaths' variable by '1'
    if touching Door Sprite 1
        switch backdrop to Level 2 ▾
        go to go to your x&y starting positions (Level 2)
        show
    if touching Door Sprite 2
        switch backdrop to The Maze Level 3 ▾
        go to x: go to your x&y starting positions (Level 3)
        show
    if touching Final Door ▾ ? then
        hide
        switch backdrop to You win! ▾
```

Showing and hiding elements

Here is a sample code for a **door/ portal** which will appear at x:136 y:171 in Level One of our version of the game **(your numbers will be different)**. It will only be displayed in Level One as, in our example, when you press the a key, level one begins.



```
when 🏳 clicked
hide

when backdrop switches to [You lose! ▾]
hide

when backdrop switches to [You win! ▾]
hide

when backdrop switches to [FINAL LEVEL ▾]
hide

when [a ▾] key pressed
show
go to x: (136) y: (171)

when backdrop switches to [The Maze Level 3 ▾]
hide

when backdrop switches to [Level 2 ▾]
hide
```

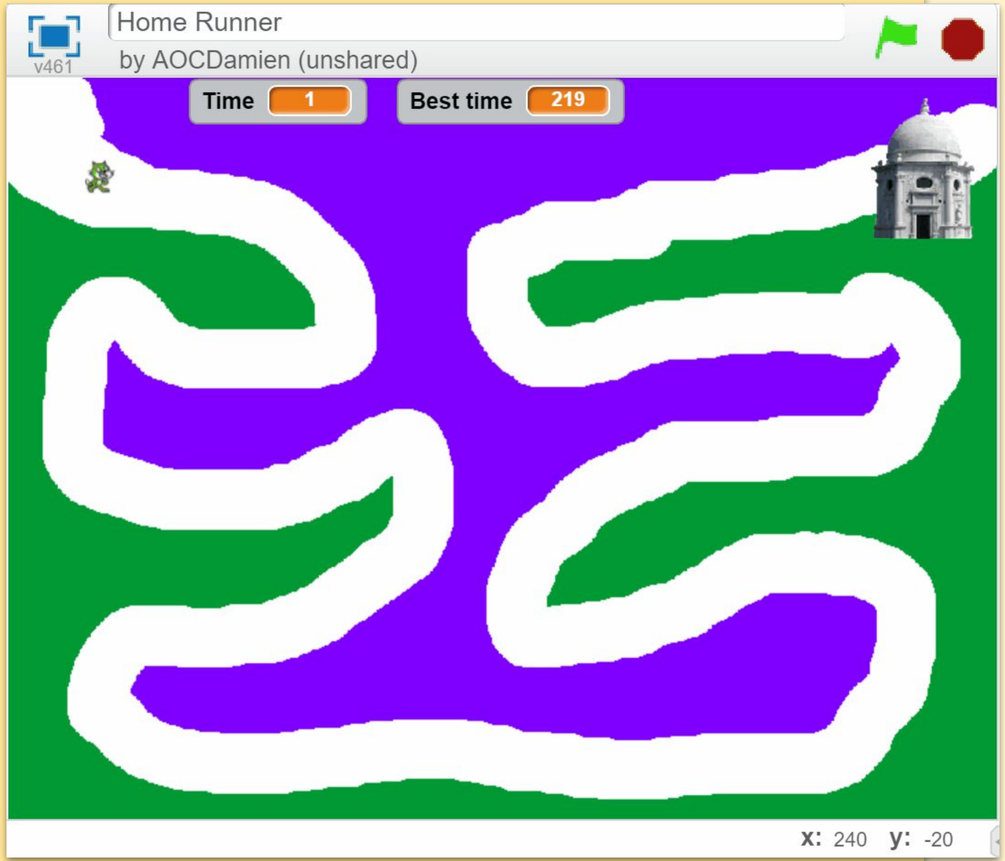
Before you say “I’m finished” have you:

- Added in more levels
- Added in a timer
- Considered having something for your player to collect for points
- Considered having some moving walls (the same colour as the walls) that reset our player sprite
- Added in a ‘next level’ or a ‘You lose’ screen
- Sanity-tested and debugged all levels and featured.

[the academy_of_code]

‘Advanced’ Maze Game

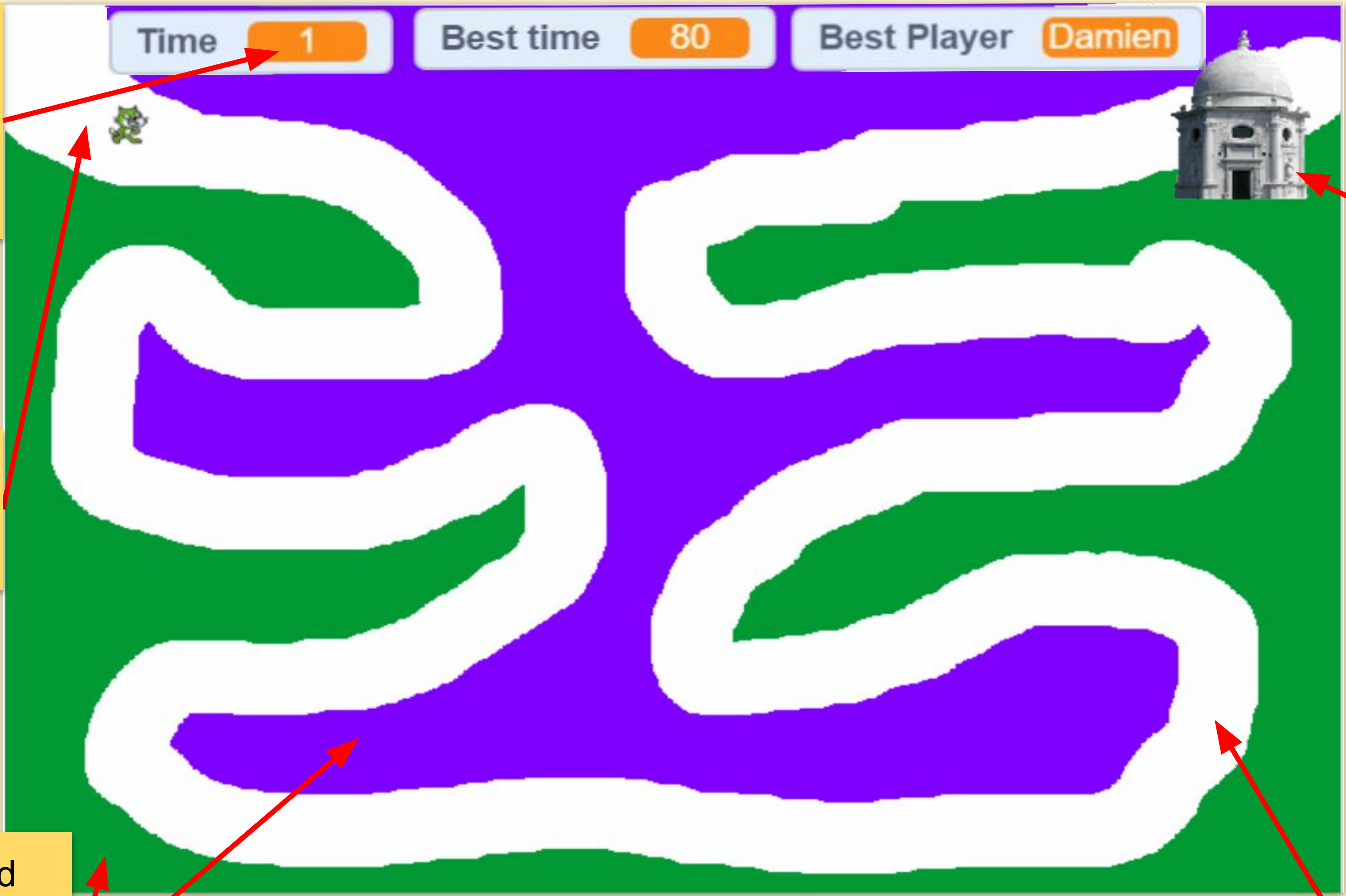
Home Runner *(Advanced Maze)*



Our last game! So far we have **Hungry Shark.** It uses the up and down arrows to control a snapping shark.

Today we're going to make a game called **Home Runner,** a game which has a player controlled sprite and a goal. Like **Hungry Shark,** it uses up and down arrows to control the **y axis** but also left and right arrows to control the **x axis.** The goal is to get the player sprite home in the quickest time possible.

Game Elements



Time Variable, best score and top player

Starting/ Respawn Point

Home. Get the sprite to here as quickly as possible

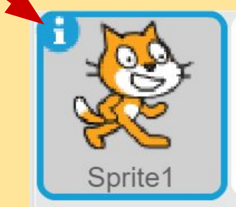
Water and grass form a maze. We could call this our enemy.

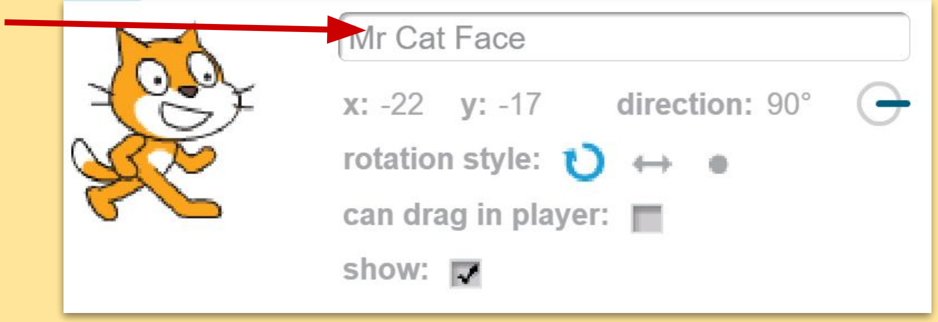
The track (where the sprite can walk)

Lets get Coding

1 Name your project (For example: *Home Runner*)

2 **Choose** and **rename** your sprite.





Mr Cat Face

x: -22 y: -17 direction: 90°

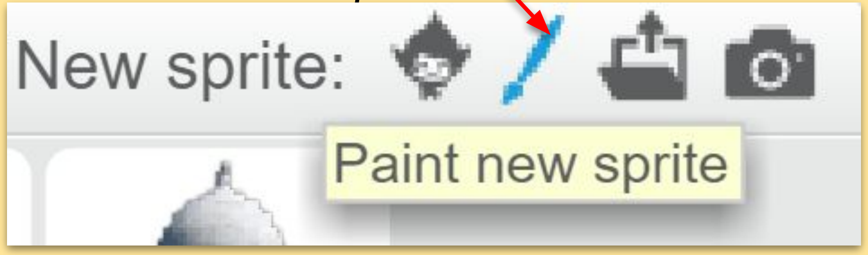
rotation style:

can drag in player: □

show: ☑

Unlike the others games we made, <u>**DON'T**</u> load a background image. Our background will actually be a custom sprite that doesn't move and takes up the entire window. Our project needs three sprites in total, **the player, the goal** (home) and **the map.**

3 Click on the paintbrush to *Paint a new sprite*



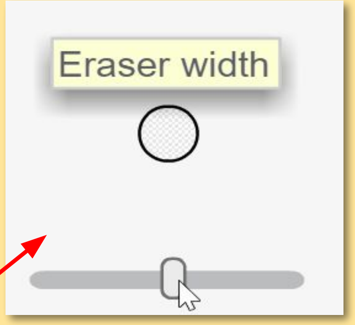
Making our Maze

1 Click the aint can

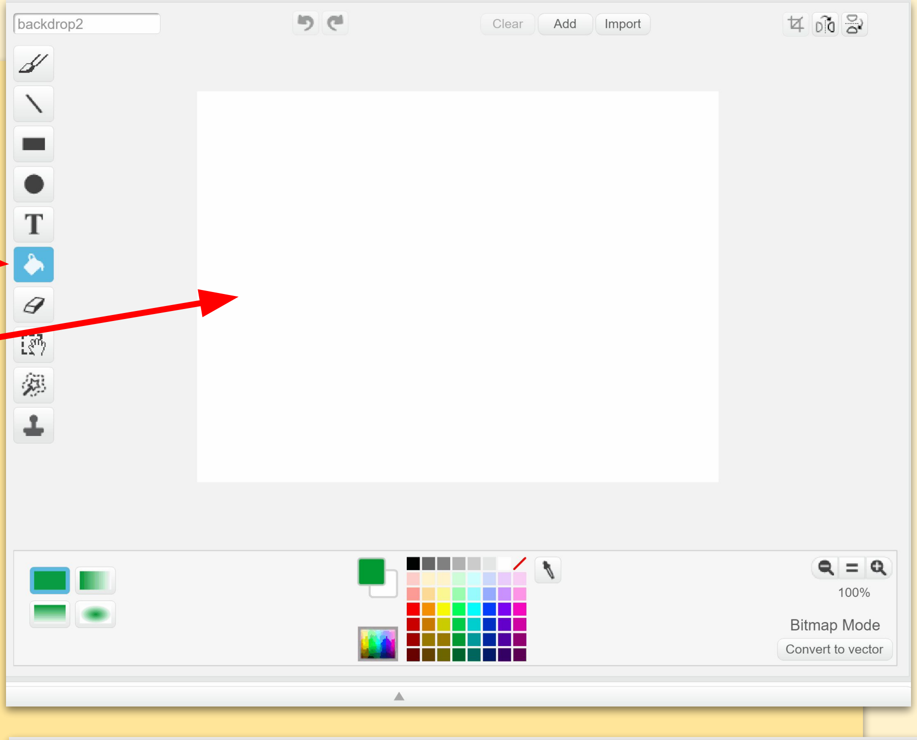
2 Paint the entire area green.

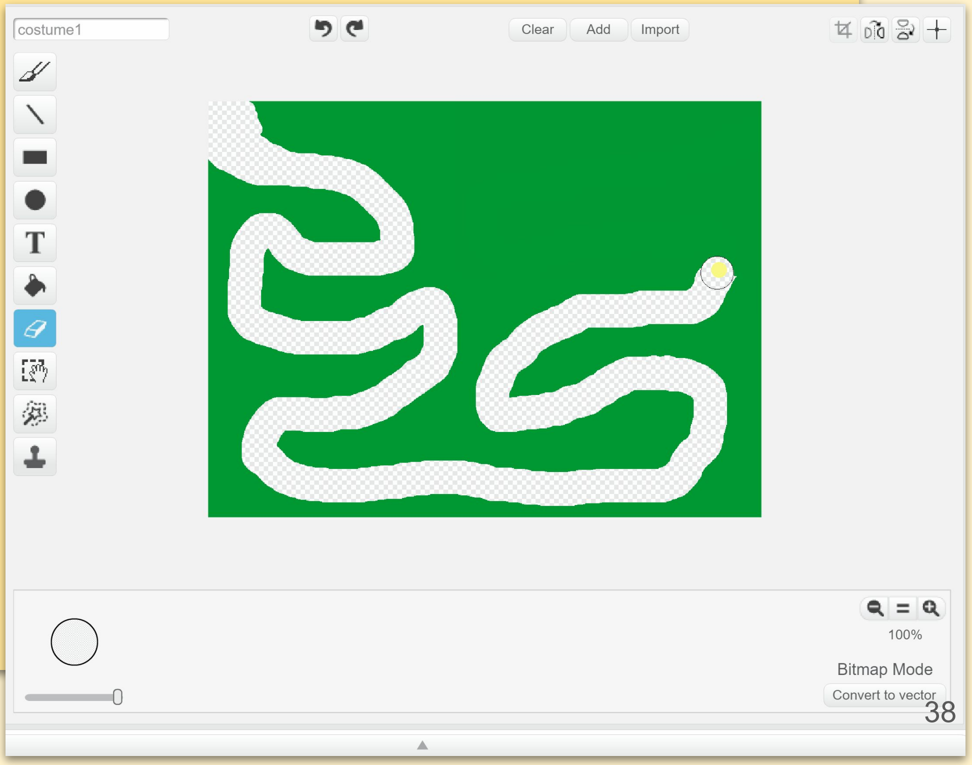
3 Select the Eraser

4 Increase the Eraser Width



5 Using the **Eraser Tool** carve out a maze wide enough for your sprite to fit through. Your maze should split the window into two sections, grass and water.

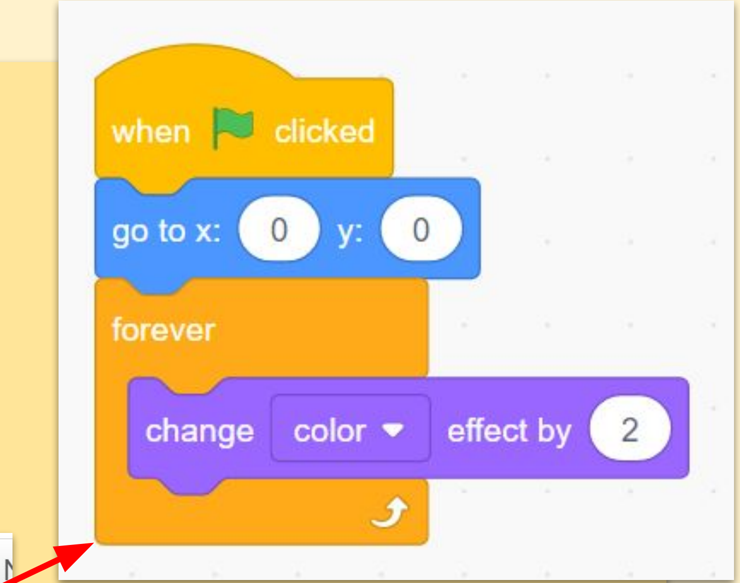




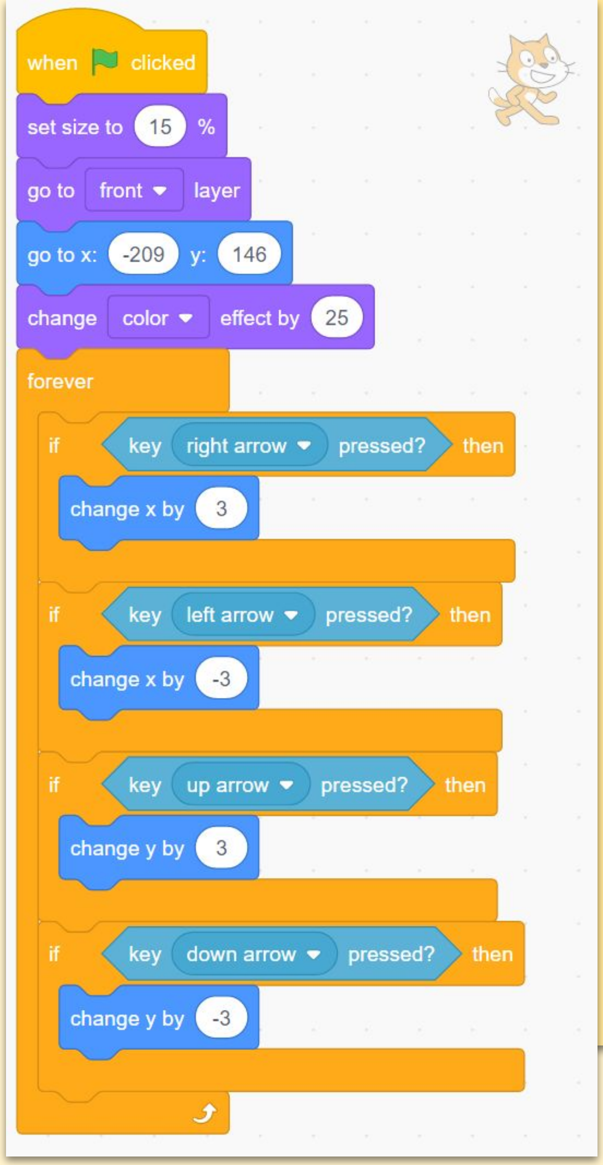
Game Progress 7%

Code for the Maze

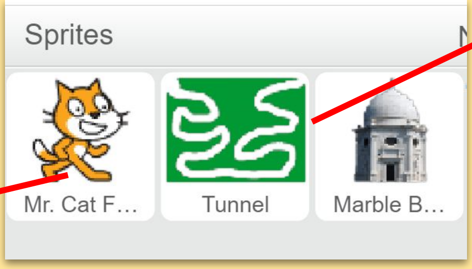
Our maze doesn’t actually need any code to function properly, but if you like, this code will change the colour in a psychedelic way.



```
when 🏴 clicked
go to x: 0 y: 0
forever
    change color ▾ effect by 2
↺
```



```
when 🏴 clicked
set size to 15 %
go to front ▾ layer
go to x: -209 y: 146
change color ▾ effect by 25
forever
    if key right arrow ▾ pressed? then
        change x by 3
    if key left arrow ▾ pressed? then
        change x by -3
    if key up arrow ▾ pressed? then
        change y by 3
    if key down arrow ▾ pressed? then
        change y by -3
↺
```



Coding the Player (cat)

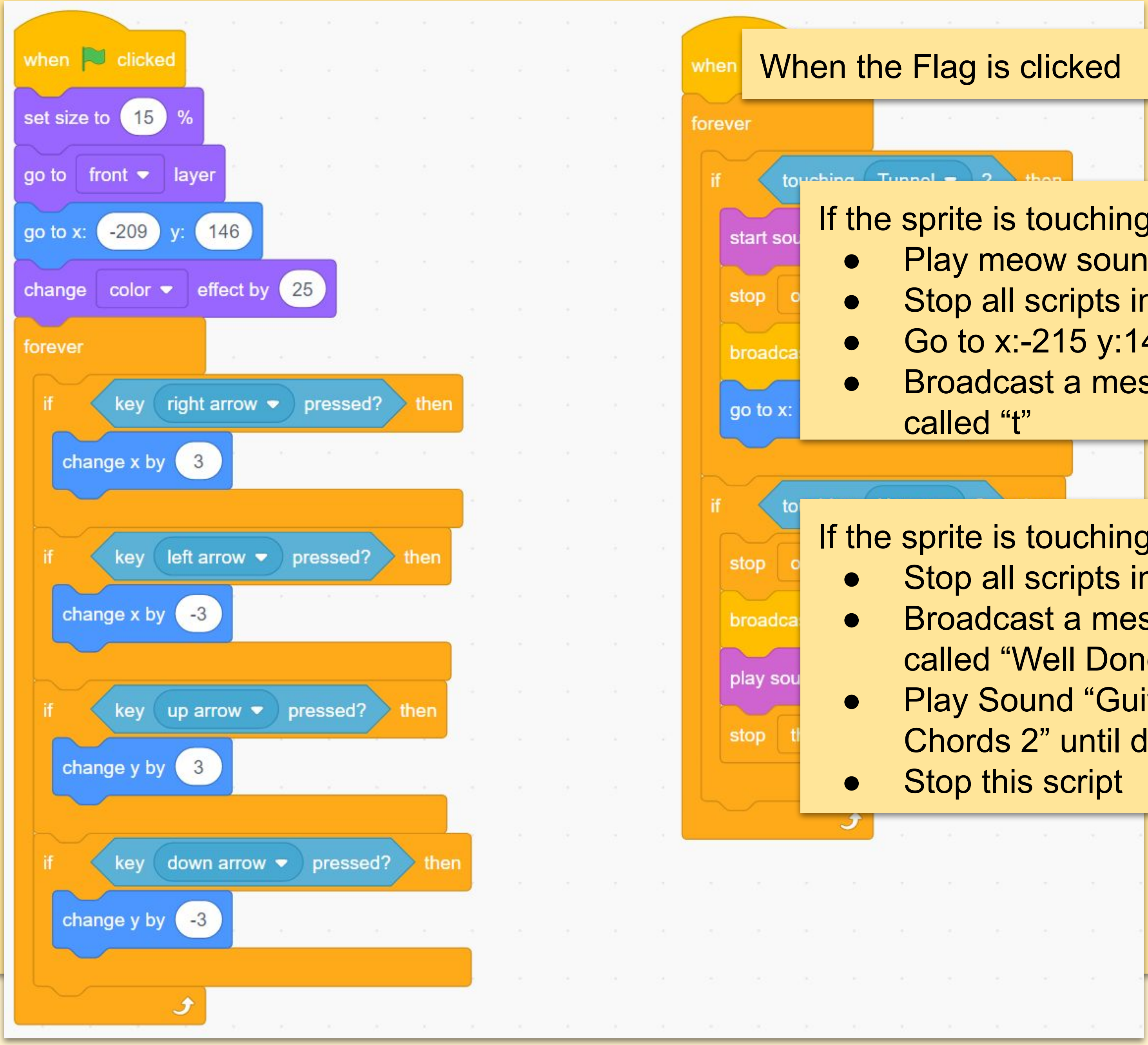
The code on the left is the starting code to get the cat moving. Make sure and help those around you if you get it finished early.

Pair Programme

Help a Friend, Make a Friend!

Game Progress 5%

More Player Code - Crack the Pseudocode



```
when 🏁 clicked
set size to 15 %
go to front ▾ layer
go to x: -209 y: 146
change color ▾ effect by 25
forever
    if key right arrow ▾ pressed? then
        change x by 3
    if key left arrow ▾ pressed? then
        change x by -3
    if key up arrow ▾ pressed? then
        change y by 3
    if key down arrow ▾ pressed? then
        change y by -3
```

When the Flag is clicked

If the sprite is touching tunnel

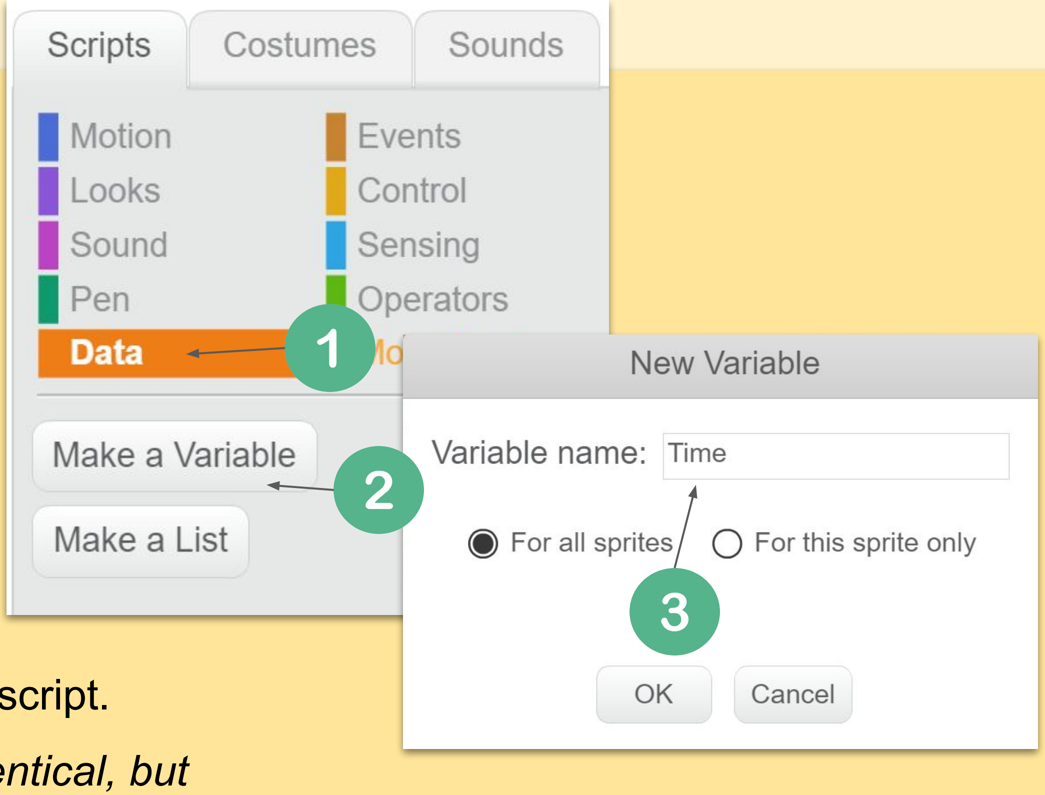
- Play meow sound
- Stop all scripts in sprite
- Go to x:-215 y:141
- Broadcast a message called “t”

If the sprite is touching Home

- Stop all scripts in sprite
- Broadcast a message called “Well Done”
- Play Sound “Guitar Chords 2” until done
- Stop this script

Against the Clock

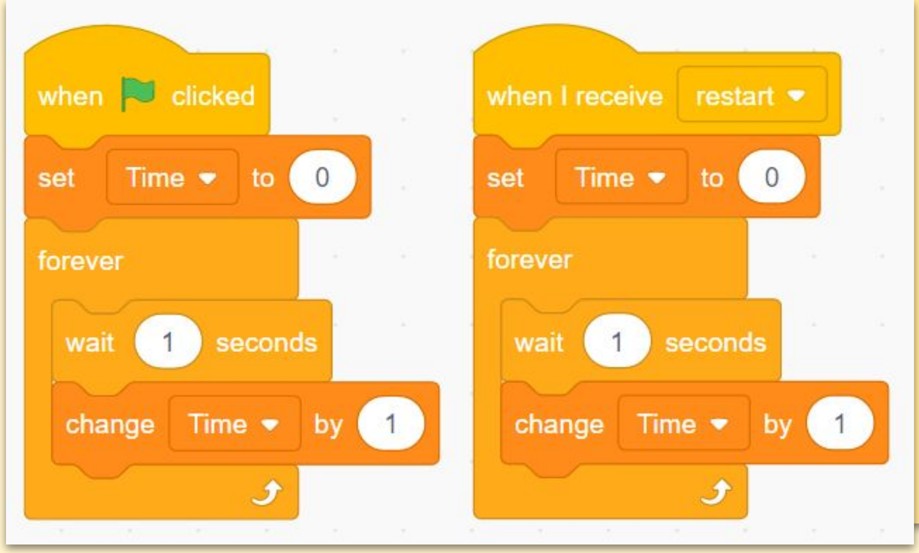
Home Runner needs a **time variable.** We need to make a variable called time but also to code this into our game.



Variable name: Time

● For all sprites ○ For this sprite only

Add these code blocks to the **player** script. *What will they do? They're almost identical, but why do we need both of these?*



```
when 🏁 clicked
set Time ▾ to 0
forever
    wait 1 seconds
    change Time ▾ by 1
```

```
when I receive restart ▾
set Time ▾ to 0
forever
    wait 1 seconds
    change Time ▾ by 1
```

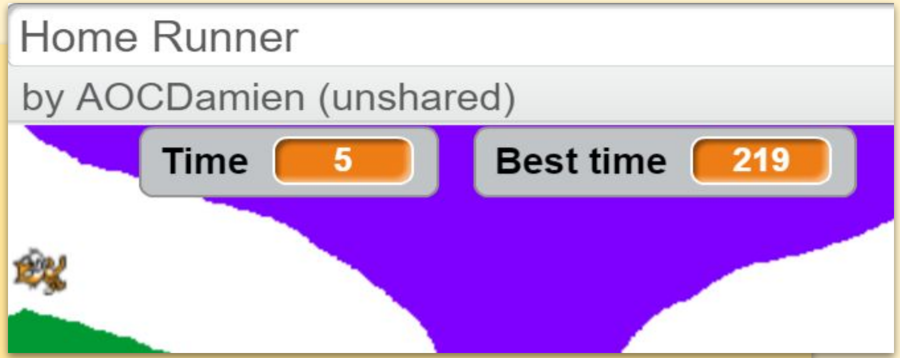
Expert Tip

A **broadcast** is a message that is sent through the Scratch programme. It allows sprites to 'talk' to each other. One sprite will broadcast a set of commands and another will receive them.

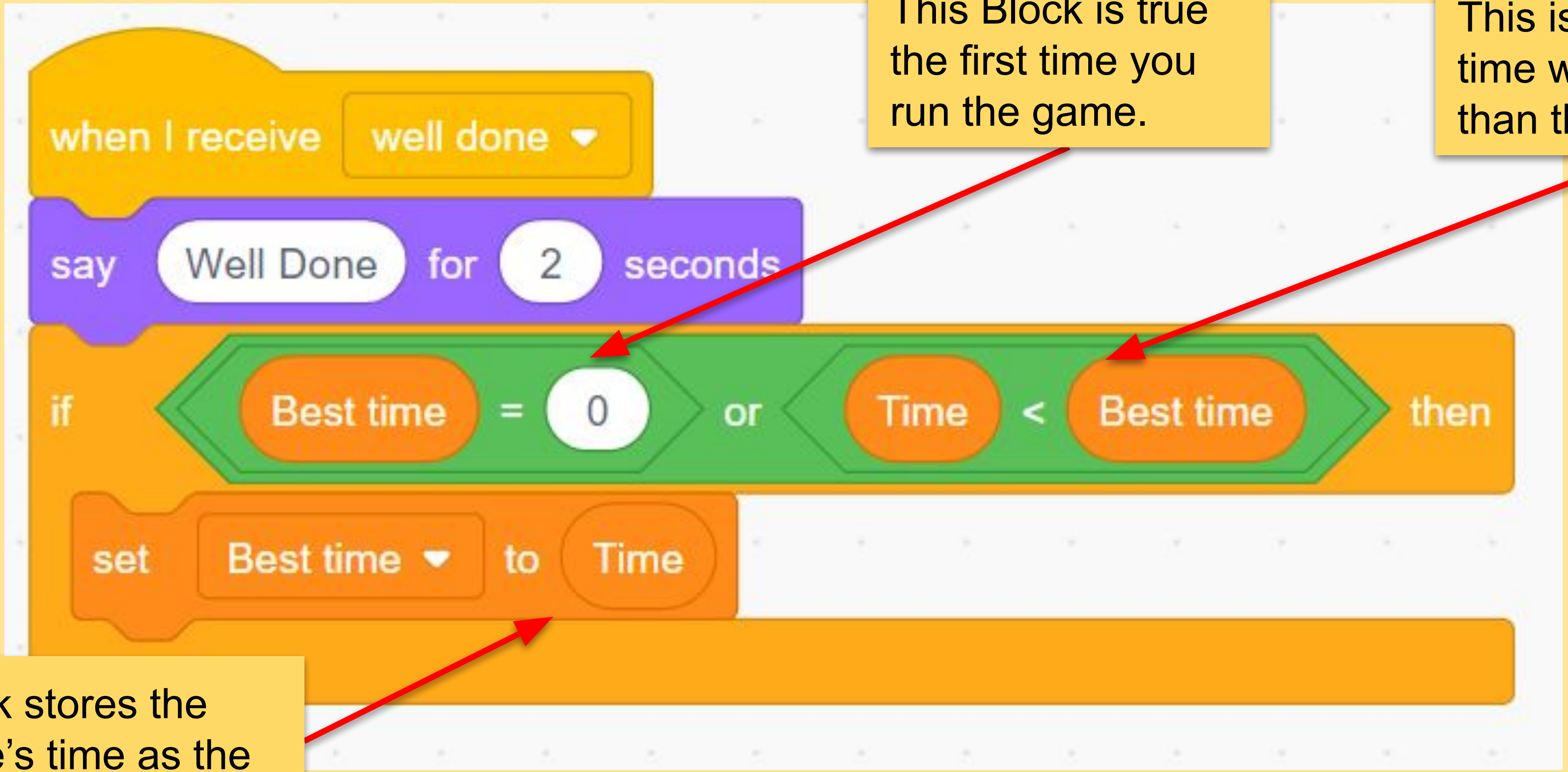
Game Progress 8%

Keeping Track of Time

We’re going to make this game more competitive by adding a **Best Time** feature.


Home Runner
by AOCDamien (unshared)

Make a Variable called *Best Time* and drag it next to the *Time* display on the stage. Then make the following code. *But what does it do?*


```
when I receive [well done ▾]
say (Well Done) for (2) seconds
if <<(Best time) = (0)> or <(Time) < (Best time)>> then
    set [Best time ▾] to (Time)
```

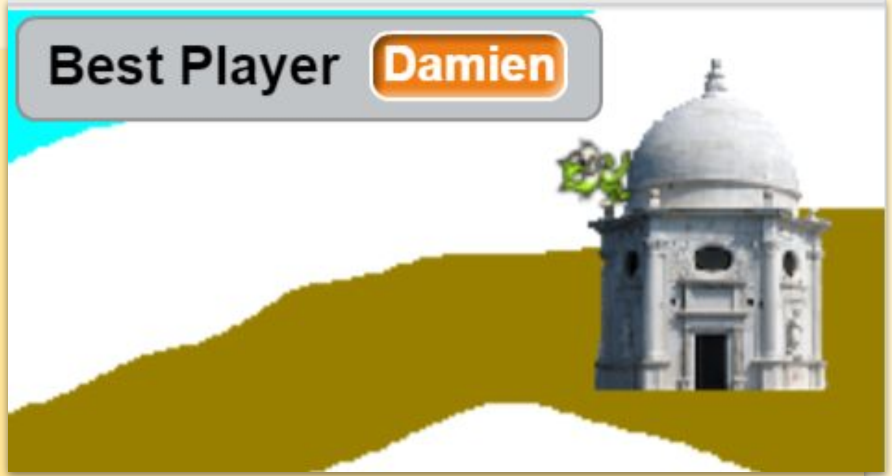
This Block is true the first time you run the game.

This is True if your time was quicker than the old record

This block stores the last game’s time as the new best time.

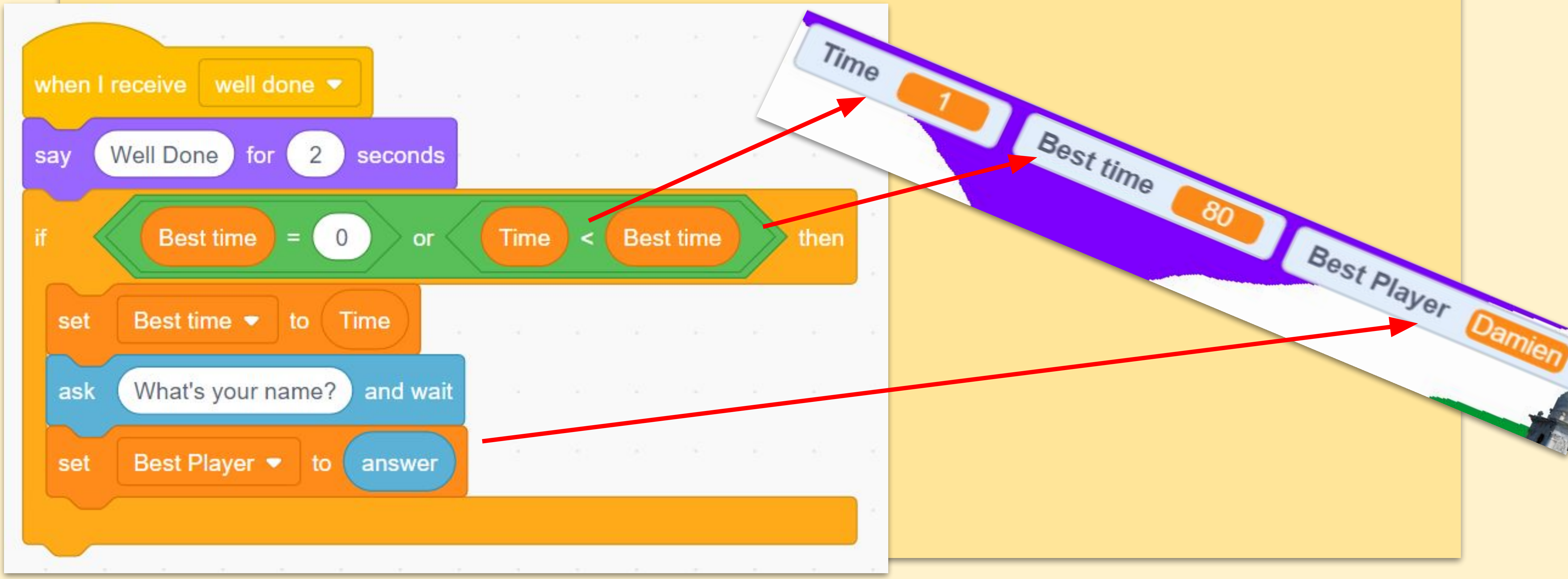
Game Progress 5%

Keeping a Leaderboard



Last but not least, you can add a *Best Player* feature to keep track of high scores.

Make a final Variable called *Best Player* and drag it next to the *Time* and *Best Score* display on the stage. Then add the following code.



```
when I receive [well done ▾]
say [Well Done] for (2) seconds
if <<(Best time) = (0)> or <(Time) < (Best time)>> then
    set [Best time ▾] to (Time)
    ask [What's your name?] and wait
    set [Best Player ▾] to (answer)
end
```

Game Progress 00%