Week 10

A new game: Archery

One of the class members had the idea for this game — you have to throw balls at a target, and you get more points the nearer the middle of the target your ball lands. To choose how hard to throw the ball, the player holds down the 'space' key. When they let go of 'space', the ball is thrown. The longer they hold down 'space', the faster (and so further) the ball goes.

Draw the target

I used a set of four circles, one 'on top of' another. You could make more or less than this, but too many will make the scripts more complicated later. You can of course use any shape as long as it has regions of different colours. Or you could find an image online.

Shrink or grow the sprite so it's about ⅓ as high as the stage, and put it near the top of the stage, centred left/right.

Ball

It's easiest to just pick one from the library. If you have time at the end and want to draw your own, you can. Adjust its size until it will fit in the bullseye of your target.

Scripts for the ball

For the rest of this worksheet, **all scripts belong to the ball**.

General idea of how the game will work

The player will get five shots. For each shot, the game will:

- Put the ball in its starting position at the bottom of the stage;
- Get the throwing speed from the player using the 'hold down space' idea;
- Throw the ball up the stage towards the target;
- Give the player some points, depending on where the ball landed.

Create a variable to track the speed

Just from this list of jobs, we know we will need a variable to track the ball's speed.

Make a 'for all sprites' variable called 'speed'.

While we're working on the game, leave the 'show on screen' box ticked so we can check things are behaving OK.

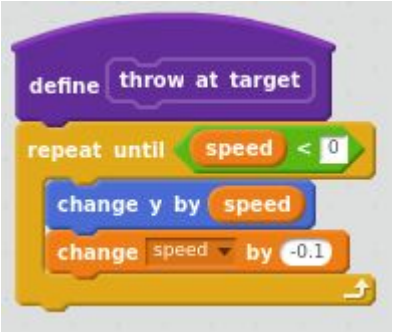
Throw the ball up the stage

We'll need to adjust this a bit to get it to work sensibly, so we'll set up scripts to make it easy to test and play with.

Custom 'throw at target' block

As we've done before in our SpaceChem project, go to the 'more blocks' section, and click 'Make a Block' to create your own block with a good name. I called it '*throw at target*'.

We want the ball to go faster the bigger the 'speed' variable number is. Also, the ball should slow down as it goes up the screen. So 'speed' will go down a tiny bit after each step. The throw is over when the speed is zero. **But!** The way computers work with numbers, sometimes you don't get exactly zero, so we'll play safe and stop when the speed gets below zero.



Put this together: ***Add this definition* for 'throw at target'**:

Throw ball by pressing 't' (for 'test') key

We want to play with different speed changes, and different starting speeds, so we'll make a testing script to make this easy.

***Add this script* to the ball:**

Test it!

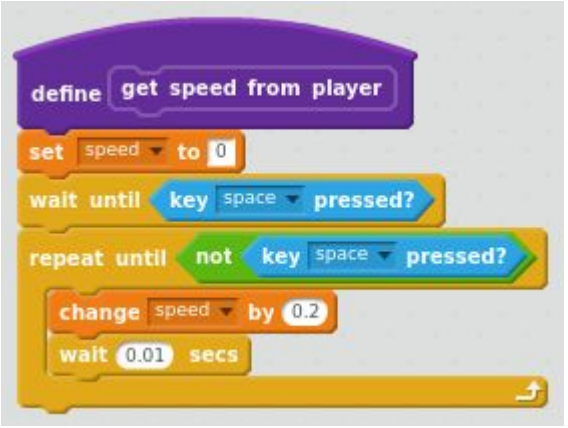
Press 't' and you should see the ball get thrown up the stage.

Adjust the '−160' and '6' in the test script, and the '−0.1' in the '*throw at target*' definition, until you like how high the ball starts, the speed the ball moves, and how it slows down.

Work out which numbers for 'speed' make sense — how big does 'speed' have to be so the ball just reaches the top of the stage?

Get speed from player

We want to let the player set the starting speed of the ball. To keep our program easy to read, make a new custom block '*get speed from player*'. This block needs to: Wait for player to press space; start 'speed' off at zero; increase 'speed' variable while player holds 'space' down; finish when player lets go of 'space'.



***Add this definition* for our new block:**

The very short 'wait' is so the speed doesn't go up so quickly that it's impossible to control.

Use in our test script

We can update our test script: Instead of just using '6' for the speed, we can get the speed from the player using our new block.

It will be handy to have a pause between two stages so we can check what 'speed' is.

***Change* the test script**:

Test it!

Press 't', then press and hold space. When you let go, the ball should get thrown up the screen faster the longer you held 'space' down.

Stop player throwing ball too hard

Playing around, you'll see you can hold 'space' down for a long enough time that the speed of the ball is stupidly high. We want to stop the player throwing the ball 'too hard'. Experiment until you find what is the highest sensible speed. For my numbers it was '8'. We want to change the '*get speed from player*' block so it stops if the speed reaches that number.

***Change* the 'get speed from player' definition**: In the 'repeat until', instead of

use

This means 'keep increasing the speed until *either* the speed has gone over 8, *or* the player has let go of the 'space' key'.

Find what the player scored

In the game, there are two ideas of ‘score’ we need:

- What score did the player get with this ball?
- What is the player’s total score so far in the game?

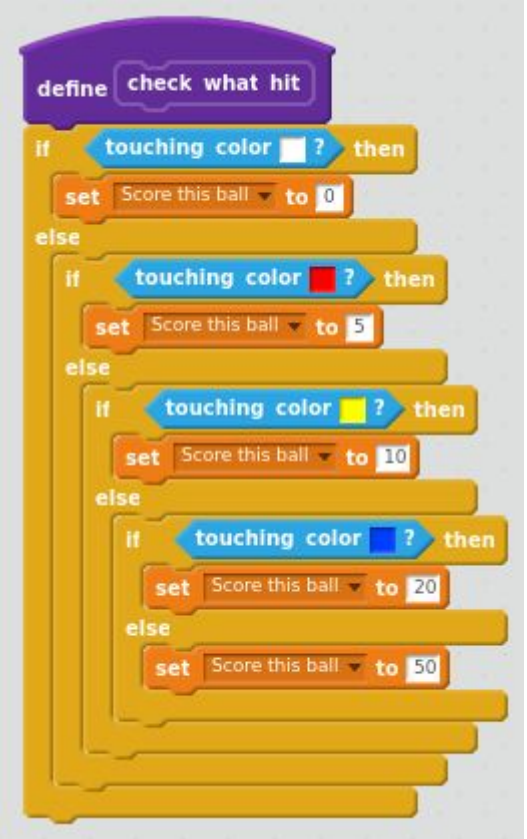
Make two ‘for all sprites’ variables: ‘Score’ and ‘Score this ball’. Move their displays around on the stage, so they’re not in the way.

Test what colour the ball landed on

To make the game harder, if the ball lands part on one colour and part on another, you get the *lower* score of the two. So we test the lowest-scoring colour first. The idea is to ask:

- Is the ball touching the background? If so, the player gets no points. Otherwise:
- Is the ball touching the outside ring of the target? If so, the player gets 5 points. Otherwise: … etc.
- For the bullseye, once we’ve checked all other possibilities, this must be what the player hit.

This is a bit fiddly, but you’ll end up with something like this:



***Add a custom block* ‘check what hit’ with this definition**:

All the colours in the tests here are the ones in your target. Choose some scores you think are fair.

Now we will use this custom block, which needs to happen after the ball has been thrown up the screen at the target.

***Add a block* to the end of your test script**:

Test it!

You should see the ‘Score this ball’ variable measure the right score for where the ball lands.

Add on this ball's score to total score

Once we know how many points the player got from this ball, we add those points onto the player's total score for the game so far.

***Add these blocks* to the end of the 'check what hit' definition**:

(We also show/hide the 'score this ball' variable to show the player their score for the ball.)

Make 'do one shot' custom block

Your 'test' script is now doing exactly what one shot of the game is.

***Make a new custom block* 'do one shot'**, and use your test script as its definition:

(You can throw away the 'when 't' key pressed' hat block.)

One game is five shots

The only thing left is to make the game give the player five shots at the target. To keep track of how many shots the player has left, we will need another variable.

Make a 'for all sprites' variable called 'shots left'. Adjust where it is shown on the stage.

To put the last bit of the game together: At the start of the game, we need to set the score to zero, and the number of shots left to five. Then we just 'do one shot' five times, pausing between each one.

***Add* this 'green flag' script**:

Try it!

The game should be finished. See how well you can score!

Challenges

Make the target move up or down a random amount just before each shot.

Make the target glide left and right.

If the ball touches two colours, give the player the *higher* score.

Add a ‘speed meter’ which shows how hard the player will throw the ball, but not in a way which makes it too easy to tell where the ball will land.

Key points

Realistic movement with slowing down.

Breaking problem down into small pieces with sensible names.

Making each piece one at a time and testing as you go along.

Sequence of tests in the right order to make the game harder.