Lets create characters that respond to the keyboard

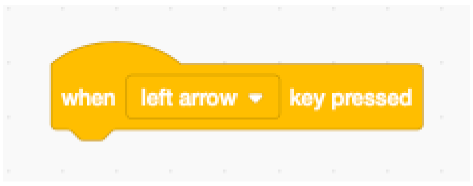
SLQ Wiki Fabrication Lab 2026/01/09 15:05

Lets create characters that respond to the keyboard

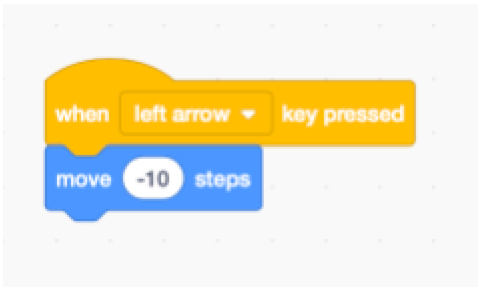
Creating characters that move based on keys pressed on the keyboard is simple. In this tutorial, we are going to move the Scratch cat left, right and jump when the arrow keys or the spacebar is pressed.

Moving Scratch the Cat Left

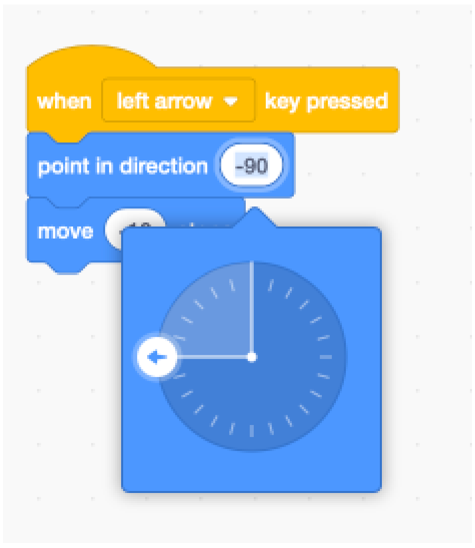
To begin, we are going to drag the *When space key is pressed* hat from the *Events* palette and change the key from *space* to *left arrow*.



Next lets connect a *move X steps* block from the *Motion* palette under the *When left arrow key is pressed* hat, change the 10 steps to a -10 steps to change our character from moving right to moving left, and finally press the Left Arrow key on the keyboard to see if it works.



Well the Scratch Cat is moving left, but backwards. Lets fix it by first adding the *point in direction* block between the event hat and the move steps block, then change the direction to -90 to point left.



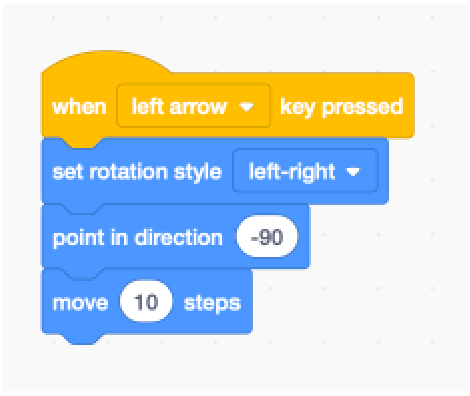
If you press the Left Arrow key now, you will see that the Scratch cat will:

- Point the correct direction, left but upside down!
- Will now move right instead of left!

This is because by default, the *point in direction* block rotates the sprite in a circle, not left-right and now that the sprite is facing to the left, the move -10 steps tells the sprite to move backwards from the direction it is facing.

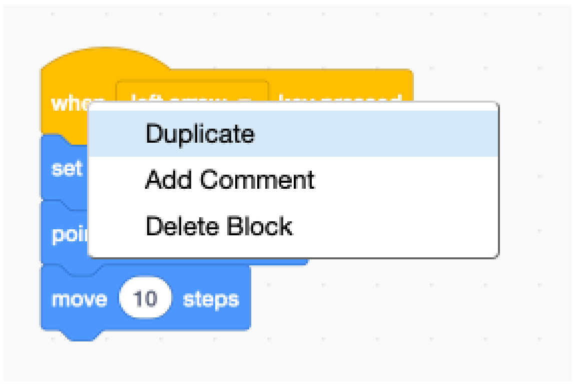
Lets fix this, first change the *move -10 steps* block to *move 10 steps*. This fixes the move left problem.

Next we need to let scratch know that when we use the *point in direction* block, we want it to act in a left-right style, so lets add a *set rotation style left-right* block from the *Motion* palette above the *point in direction* block.



Moving Scratch the Cat Right

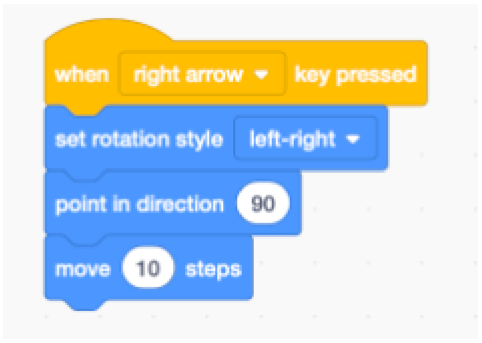
With the moving left code done, we can now create the moving right. As the code is similar to the moving left stack, we can right click on the *When left key is pressed* hat and select duplicate.



Next, we need to change a few variables on this new stack:

- The *left arrow* needs to change to *right arrow*
- The *point in direction -90* needs to change to *point in direction 90*

And we are done here!

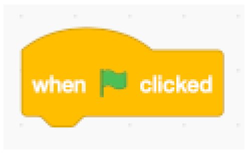


Resetting the Scratch Cat

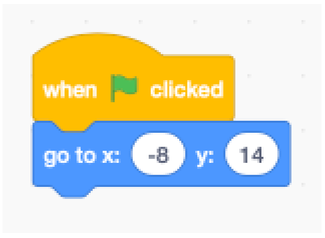
Before we continue to the jumping code, we need to reset the sprite to its starting location whenever we press the *Green Flag*.

On the stage, move the Scratch Cat to where you would like to be whenever the *Green Flag* is pressed. Typically this is in the centre of the stage.

Now grab a *When Green Flag clicked* hat from the *Events* palette and place it in the programming area.

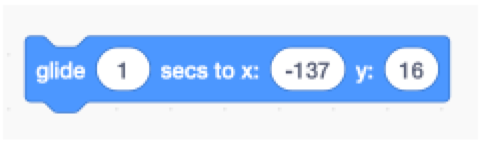


Next grab a *go to x / y* block and connect it under the *When Green Flag clicked* hat. The *go to* block will be preset to the current position of the Scratch Cat. This is why we moved it around on the stage before grabbing the *go to* block.



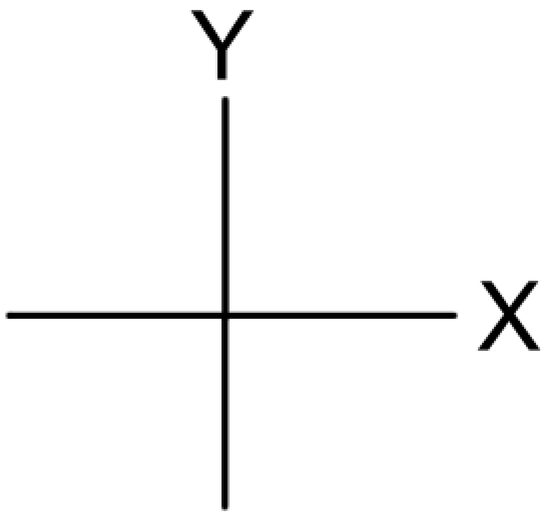
Jump Cat Jump!

Onto the tricky part, to make the Scratch Cat jump, we will need a *glide 1 secs to x / y* block from the *Motion* palette. Here we use a *glide* block instead of a *go to* block as it 'glides' or 'slides' instead of 'jumping' or 'teleporting' to the position.



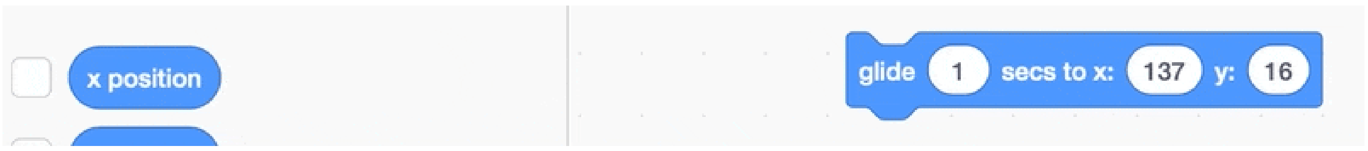
The X and Y on your block maybe different to this example. That's ok, we are going to change them in a second.

First we need to know that whenever we are talking about the X position (or X axis) of a sprite, we are talking the horizontal or left-right position. The Y position (or Y axis) is the vertical or up-down position.



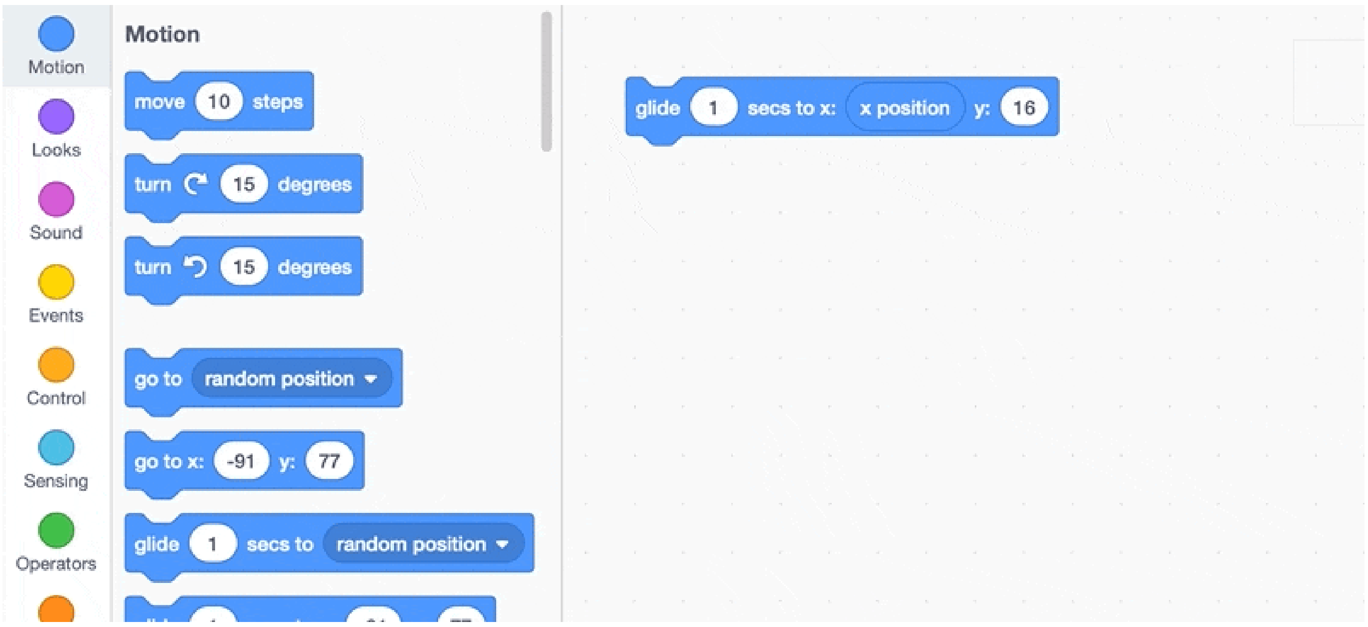
Now, the *glide* block needs to know the X and Y position to move our Scratch Cat to, but our Scratch Cat can move left or right, along the X axis, so we don't know what the X position of the cat will be, it could be anywhere depending on how many times the player has pressed the left or right arrows. So what do we do?

We can use a variable block. Under the *Motion* palette, find the *x position* and move it into the X field of the *glide* block. We have now told that we want to move the X position of the Scratch Cat to the current X position of the Scratch Cat! So the Scratch Cat won't actually move left or right when we jump!

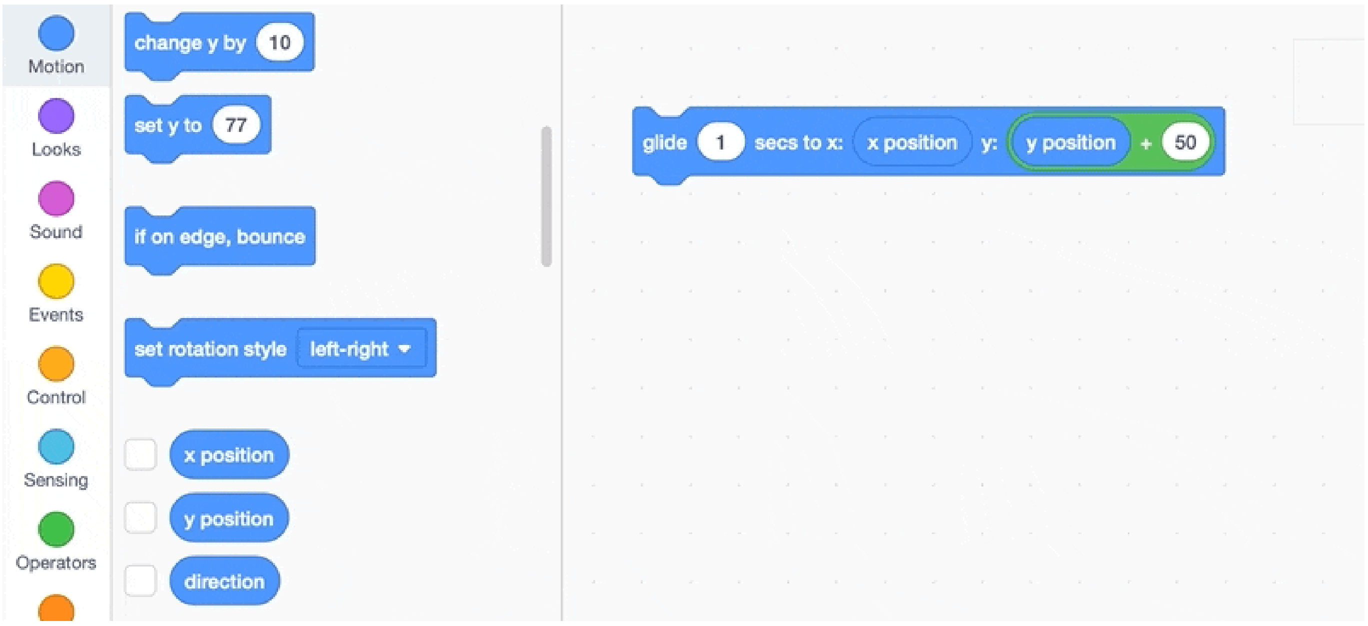


Next we need to glide the Scratch Cat upwards along the Y position. To do this we are going to use some Math and an operator block.

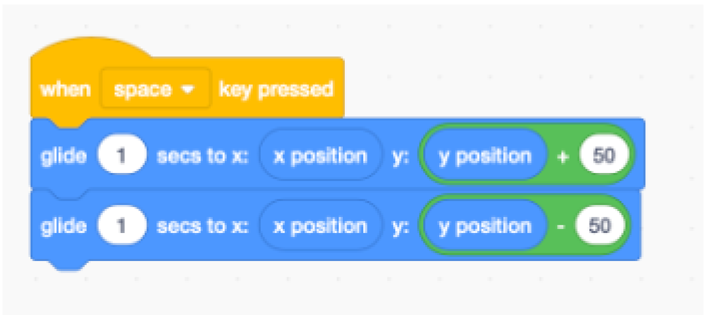
Grab a *X* + *X* block under the *Operators* palette and place it in the Y field of the *glide* block. Next we want to the *y position* block under the *Motion* palette into the first field of the *X* + *X* block, and finally we are going to type the number 10 in the second field of the *X* + *X* block.



Now that the Scratch Cat is able to jump up, we need it to fall back down. We can do this by duplicating the first glide block and changing the *X* + *X* block to a *X* - *X* block.



and now to finish it off, lets grab a *when space key is pressed* hat from the *Events* palette and connect our glide blocks underneath.



```
when [space] key pressed
glide (1) secs to x: (x position) y: ((y position) + (50))
glide (1) secs to x: (x position) y: ((y position) - (50))
```

Awesome!

You can view the final Scratch project over on the Scratch Website