Scrolling Platformer in Space

Click here for the starter project with the sprites and backgrounds but no code: https://scratch.mit.edu/projects/662209925/editor/

Now click the green Remix button to get your own copy to work from

We've got four sprites in the scrolling platformer:

Dot, the astronaut dog who has to travel along the platforms:

The **Guide** sprite, which is always hidden behind the Dot sprite:

The **Guard** sprite, which is also hidden behind the Dot sprite

And the **Platforms** sprite:

There are also two Backdrops which you can see if you click on the Stage icon and then choose the Backdrops tab:

The **Stars** backdrop, which is just lots of stars in outer space

And the **Game Over** backdrop, which has "Game Over" written on it.

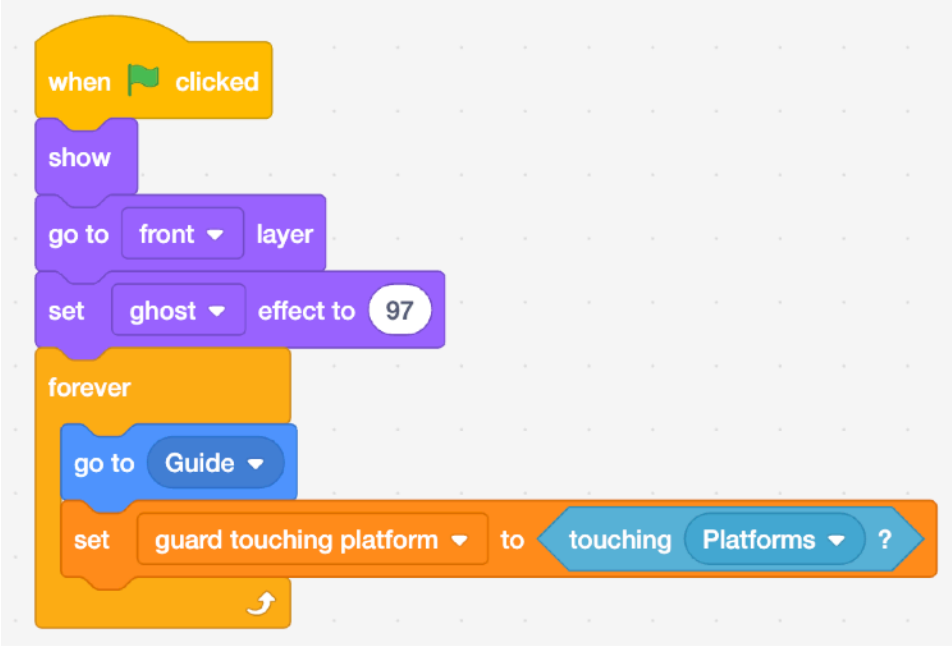
The Guard Sprite

Slightly oddly, we're going to start with one of the sprites that's not visible to players when the game is running: the **Guard** sprite which looks like a yellow rectangle in its icon.

The Guard sprite is there to make sure the Guide sprite and the Dot sprite are actually standing on top of the platforms, not moving "through" them.

To do this, first of all make a variable **for all sprites** called **guard touching platform**:

Now add this code to the sprite's code window.



Strangely, we start off with the **show** command.

This is because a sprite that's been hidden with the **hide** command can't be detected when it touches another sprite.

But this sprite is supposed to be invisible to the player! To do that we use the **set ghost effect** block.

This has the same effect as hiding the sprite, but we can still use the pale blue **touching** block to see if it touches something else.

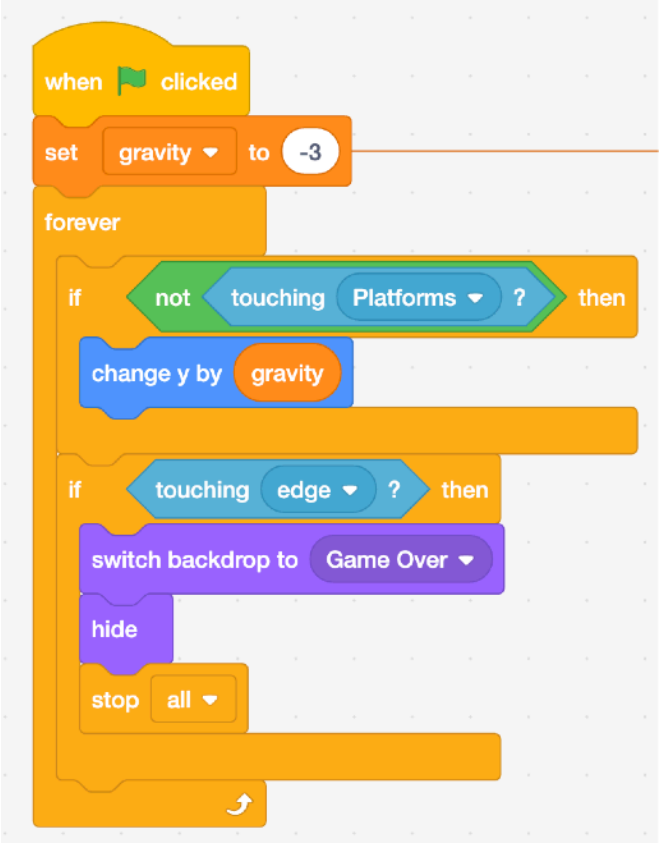
Inside the forever block we tell our sprite to go to the same position as the **Guide** sprite. And we update the **guard touching platform** variable to the value of **touching Platforms**. Because it's a bit smaller than the Guide block it shouldn't touch the platforms if the Guide is sitting on top of one.

The Guide Sprite

The **Guide** sprite is the one that's actually going to do most of the interacting with platforms. If you look at its costume you'll see that it's a grey square. The reason it's square is because it's useful to have a shape that can sit flat on the platforms. (The greyness is just accidental).

Gravity and edges

One of the most important features of a platformer game is **gravity**: if a sprite isn't standing on something solid, it should fall towards the ground. In this case we'll make it fall down the way, even though there isn't any "ground" in space.



First, make a new variable called **gravity** to store the value of gravity in our game in.

Now add the code on the right here to the **Guide** sprite

This sets the value of **gravity** to -3 then, for the rest of the time the game is running, it:

- Checks to see if the **Guide** sprite is touching the **Platforms** sprite (i.e. is it on one of the platforms). If not, it pulls the sprite down by changing y by **gravity**.
- Checks to see if the **Guide** sprite is touching the edge of the screen. If it is, it changes the backdrop to **Game Over** and stops everything in the game.

Moving left and right

Next, we want to deal with the **Guide** sprite moving left and right.

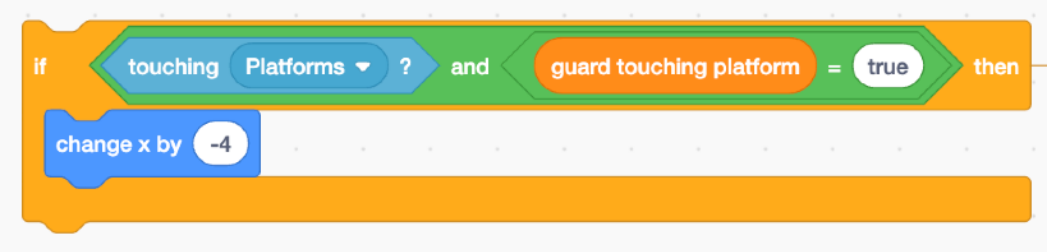
Add this code that sets up the starting position, then adds code that moves the **Guide** sprite to the right if the user presses the right arrow key.

(You'll see that it also broadcasts a message, this will trigger some code in the **Platforms** sprite that we'll add a bit further on.)

Next add this code just below the **if** block for the right arrow key and still inside the **forever** block.

This will allow the sprite to move left if it needs to.

Finally, below that and still inside the forever block, add this code which will push the **Guide** sprite off a platform if it's not sitting right on top of it.

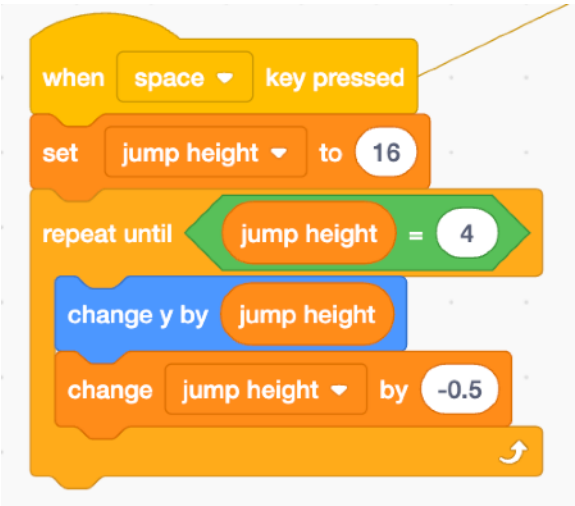


You can see that this checks to see what the value (set by the **Guard** sprite) of the **guard touching platform** variable is.

Jumping

The last bit of code we need to add to the Guide sprite (until we're almost finished) makes the sprite jump when the player presses the space key.

This code tries to give a "natural-looking" jump by having Dot move up several times, but slightly less each time. At the top of the jump she'll stop going up and start being pulled back down by **gravity**.



If you look at the picture of the Dot below you can see this pattern in the multicoloured path of her jump that's been drawn in to demonstrate.