SCRATCH PROJECT
Introduction

On 20 July 1969, Neil Armstrong took the first steps on the moon. As we celebrate the 50th anniversary of this incredible achievement, we have a chance to reflect on the difficulty and audacity of every part of the Moon missions. In this project, we're going to use Scratch to make a side-scrolling Moon Buggy game.

What you'll need:

Hardware

A computer capable of running Scratch 3

Software

Scratch 3 (either online or offline)

Step 1: Set up the surface

✔️ Activity Checklist

☐ Open the starter project at https://scratch.mit.edu/projects/293834812/editor/.

☐ First, we need to set up the Surface sprite so that it moves while the buggy stays still, simulating motion. Click on the Surface sprite and add the following code:
If you run your program now, nothing much will happen. We are telling our surface to move with `scrollx` but we aren’t changing `scrollx`.

This will continue running all of the code within the `forever` loop.

Next we will add code within our `forever` loop to move our surface:

```
set x to scrollx
```

We will use the variable `scrollx` to keep track of how far our buggy has moved, and move our surface accordingly.

**Step 2: Drive the buggy**

If you run your program now, nothing much will happen. We are telling our surface to move with `scrollx` but we aren’t changing `scrollx`.

**Activity Checklist**

- Click on the Buggy sprite and add the following code:

```
when [space] clicked

if then
```

In addition to the loop, we have added an `if` statement. The code inside this block will only run if the condition after the `if` is met.

- Add the `key pressed` block as your if condition. Change the key from `space` to `right arrow` by clicking the dropdown arrow next to `space`.
Add code to change \texttt{scrollx} by \texttt{-2} to make the Surface sprite move whenever the \texttt{right arrow} is down:

![Code block](image1)

Run your program. The Surface should move to the left, which makes the Buggy look like it's moving to the right. What happens when you get to the end of the surface? If you try and start again, the Surface stays in the same place. A game isn't much good if you can only play it once!

We'll add code to reset the Surface back to the centre when we press the \texttt{green flag}:

![Code block](image2)
Challenge: Reversing Buggy

Currently, the buggy only goes forward, can you make it go backwards, too?

Step 3: Make the buggy go to the surface

✔️ Activity Checklist

- We're going to create our own block that will make the buggy always go to the top of the Surface sprite. Go to My Blocks and click Make a Block. Name your block go to surface and click the Run without screen refresh checkbox.

Normally, when we have a loop, it will run once through the loop on every frame of the game. Checking the Run without screen refresh box will run through the entire block without needing to wait for the next frame.

- When you press OK, a define go to surface block will appear on your screen. Add a repeat until block beneath this:
We want to repeat the code inside this block until we're not touching the Surface sprite anymore:

Every time we go around the loop, we want to go up by one pixel:

This will 'push' the buggy upwards whenever it's touching the Surface sprite, until it's no longer touching that sprite.

We've finished writing our `go to surface` block, but we're not using it yet. We need to put it into our main loop:

Run your program now. Your Buggy will go up on the uphill parts, but it won't go down on the downhill parts.

To make your Buggy go down the hills, we will add a gravity effect to it. Add a `change y` block to make your buggy slowly fall down if it's in the air:
At the moment, we don't travel very far on the moon before we fall of the edge! Let's make a lunar experience a bit more expansive.

**Activity Checklist**

- Click on the Surface sprite and separate the `when green flag clicked` block from the rest of the code:

- Under the `when green flag clicked` add code to set up the initial conditions:
Add a loop that will create a clone for each costume of the sprite:

![Code block](image)

On the code that you detached, add `when I start as clone` hat block:

![Code block](image)

Because we have multiple costumes that make up the surface of our moon, we need to move them across the screen with our `scrollx`, but each costume needs offset by 480 pixels from the previous one (the width of the costume):

![Code block](image)

If we run our program now, the surface sprites won't appear, because we've previously told Scratch to hide them. We don't just want to show all of the sprites, because Scratch doesn't let our sprites go all the way off the screen. Instead we can tell Scratch that if the actual x coordinate of the sprite is where we think it should be, then show it, otherwise, hide it:
Tip: If you don't want to rewrite the code for the position of the surface, you can duplicate it by right clicking on the block and pressing duplicate:

If you run your program now, your first surface sprite will start off the screen to the right, you can fix this by clicking on the Buggy sprite and changing the initial value of scrollx to -480:

When clicked:
- set scrollx to -480
- forever
- go to surface
- change y by -1
- if key right arrow pressed? then
  - change scrollx by -2
**Step 5: Jumping buggy**

We want to make our buggy jump to avoid obstacles!

- **Activity Checklist**
  - Add the following code to make your buggy jump up by 100 pixels when the space bar is pressed:
    ```
    when space key pressed
    change y by 100
    ```
  - Run your program now. We're making our buggy jump, but it isn't very smooth.
  - Change your code to the following to make the jump a bit smoother:
    ```
    when space key pressed
    repeat 20
    change y by 5
    ```

**Step 6: Jumping the chasms and getting to the finish**

As you drive along, you'll see there are some chasms. We don't want to fall down a bottomless pit, so let's add code to make us jump them to get to the finish.

- **Activity Checklist**
  - Firstly, we need to tell the game that we've lost if we fall into the chasm. In this case, we can use the `touching edge` block to check if we're touching the edge, and if we are, it's game over. At the end of your Buggy's forever loop, add the following code:
    ```
    if touching edge then
    say "OH NO!" for 2 seconds
    stop all
    ```
  - Run your program now. If you fall into a chasm, it should end your game.
We can now lose the game, but we also want to win when we reach the goal. Add the following code to the end of your Buggy's forever loop to finish the game when we reach the goal:

![Code block showing an if statement and a say block]

Notice that because we're using a negative number for `scrollx`, we need to use the `<` block.

**Challenge: Fix the buggy buggy**

When your game restarts after losing the game, your buggy sprite is still touching the edge, causing you to lose straight away! Can you position the buggy when the game restarts so it’s not touching the edge?

**Challenge: Add a timer**

It's fun to be able to compare your best time with your friends. Can you add a timer to measure how long it takes you to win the game?

**Advanced challenge: Improved jumping**

Right now, when you jump, there's nothing to stop you jumping again or changing direction mid-way through your jump. On the moon, you can't do this! Can you update your program so that your buggy will only jump once, and can't jump again or change direction until it has landed?

_Hint:_ Setting a variable inside the repeat until loop of your go-to-surface block will help you keep track of whether or not you are touching the ground.