How to create (and extend) the Library Hunter game in Scratch

This guide assumes you're aware of the basics of how to use Scratch. We recommend you look through the Scratch cheat sheet and familiarise yourself with the Scratch interface before starting.

For further information about the platform, see our Scratch resources on our Coding Skills Guide.

Library Hunter game

The idea of the game is a simple point and click game, in which the player is trying to rid the empty library of something that shouldn't be there. Each click on the sprite earns the player a point, and there is a timer showing how long until the game ends.

See the example version in Scratch.

The game can be customised to add more sprites or functionality, but this guide will show you how to set up the basic game yourself.

Overview

These steps are the main elements of the process. You can follow these to try creating the game, or follow the more detailed walkthrough that explains which code blocks are needed.

1. Log in to Scratch https://scratch.mit.edu/ and create a new project.
2. Add a backdrop of your choice and your choice of sprite for the player to 'catch'. Use a site such as Pixabay or Unsplash
3. Create score and time variables and ensure they are displayed on the stage.
4. Select the backdrop and add code to set the score and time variables to 0, then to decrease the time by 1 every second.
5. Still in the backdrop, add a sound that plays once the timer has reached 0, then stop 'all'.
6. Now, select your sprite. The sprite needs two sections of code.
    a. One that runs when the green flag is clicked, and ensures that the sprite 'forever' goes to a random position, appears, waits a second, then disappears, then waits a random amount of time between 0.25 and 2 seconds.
    b. One that runs when the sprite is clicked, to change the score by one and then hide the sprite.
7. Finally, make sure your sprite is hidden by default, then test out your game!

If you've got your game working, check out the suggestions for extending the game further using your new Scratch skills. If not, use the following walkthrough to try and diagnose why your game isn't working as expected.

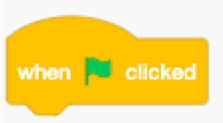
Walkthrough

If you'd like a detailed guide of the steps to create the game, follow these instructions, which go through every step of creating the game.

If you're not sure, 'See inside' the example version to look at what the code and set up should look like. Don't forget to refer to the cheat sheet if you're not sure about Scratch terminology.

1. Sign up or log in to scratch.mit.edu and click 'Create' to start a new project.
2. Rename your project with a suitable name (e.g. Library Hunter).
3. Next you'll need to add a backdrop of the library. Click on the 'Choose a backdrop' button on the bottom right of the screen, then click 'Upload backdrop'. You can then select an image you have downloaded that you are free to use (Pixabay, or Unsplash, for example).
4. Next, the game needs a sprite, aka the thing you'll be 'hunting'. Click on the 'Choose a sprite' button, then the search icon 'Choose a sprite' option will allow you to choose from the built-in sprites. Or you could remove the background from an image and upload your own sprite!
5. Now the stage is set, it is time to start on the code. The first thing needed is to set up the score and timer. These need to be variables, which are ways of storing a value in code by giving it a name.
    a. In the 'Code' pane on the left hand side, click on the **Variables** option.
    b. Click 'Make a Variable'.

    c. Give it the name 'score' and make sure it is set to 'For all sprites', then click OK.
    d. Do the same again, but give the variable the name 'time'.
    e. You should see both of these on the stage on the right hand side of the screen. You can drag them to reposition them if you want.
6. Now, you need to add code to the backdrop to make the game's mechanisms work. The game needs to have a timer that counts down until the end of the game, and then the game needs to stop.
    a. In the bottom right hand corner of the screen, you'll see a small version of the backdrop, under the Stage heading. Click on it to select it.
    b. You can now drag and drop code blocks into the central part of the screen. All code blocks need to start with an Event, so the computer knows when to run the code. In this case, we need the timer to start when the green flag button is clicked, as that starts the game.

    ```
    when 🏁 clicked
    ```

    c. Whenever the game starts, the variables need to be set to the right value, or a player might get the previous player's score, or not be able to play the game again. Drag the 'Set [variable] to 0' under the green flag Event block you've already added, making sure they click together. Click on the variable name part of the block to change it so it is setting the 'score' variable to 0.

d. Drag another Set variable block underneath, and change it to set the 'time' variable to 10.

e. Next, the code needs to repeat reducing the time by 1 each second, until the time is 0. Drag a repeat until block (from the Control section) underneath your other blocks.

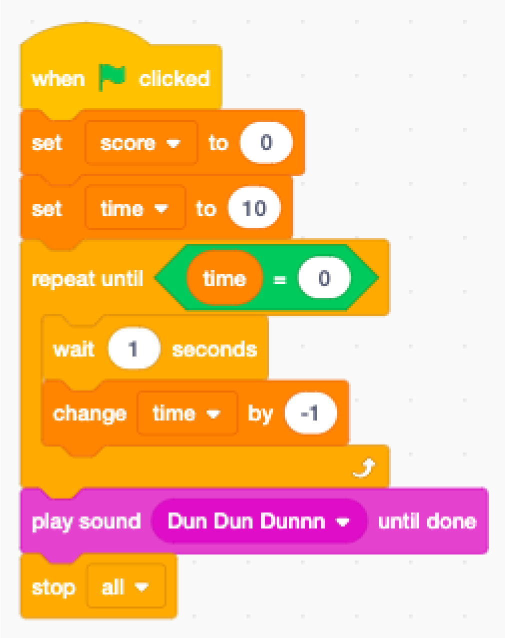
f. Time to set what the code repeats until. In the Operators section, you'll find a block for [blank] = 50, which allows you to tell the computer to stop the code when the time is 0. You can't just type 'time' however; you need to go to the Variables section and drag the orange 'time' block into the space so it knows to use an existing variable.

☑ time

g. Inside the repeat until loop, you need to tell the computer to change the time and then wait a second, as you would if counting. You'll need to use a 'wait X seconds' block and then a 'change [variable] by [number]' block. As there isn't a decrease option, you'll need to remember to type '-1' as the number to change it by.

h. Underneath the repeat block, add a sound to play to signal the end of the game, by going to the Sound section and using the 'Play sound until done' block. After this, add a 'Stop all' block from the Control section so the game stops at the end.

i. The backdrop code is done! It should look like this:



```
when 🏴 clicked
set score ▾ to 0
set time ▾ to 10
repeat until time = 0
    wait 1 seconds
    change time ▾ by -1
play sound Dun Dun Dunnn ▾ until done
stop all ▾
```

7. Now, the sprite needs to have some code so the computer knows to make it appear randomly on the screen for the user to click on.

a. Firstly, click on your sprite in the bottom right of the screen to select it. The central code screen should now be blank, as there's no blocks attached to the sprite yet, only the backdrop.

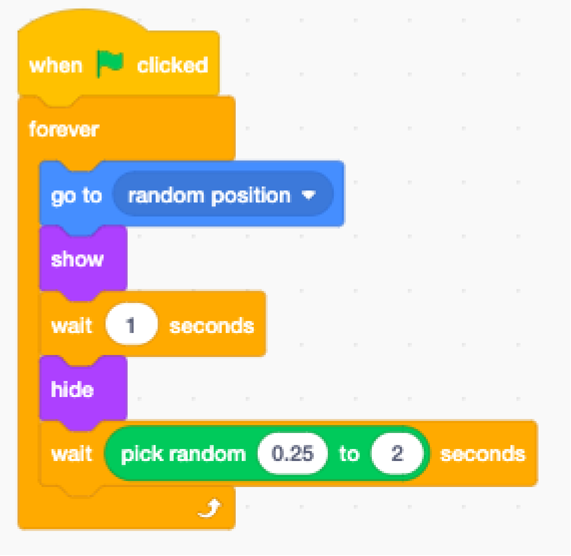
b. The sprite needs code to happen at two different times - when the green flag is clicked, the sprite needs to appear and disappear randomly, and then when the user clicks the sprite, they need to get a point and then the sprite disappear again. Start by dragging a 'When green flag clicked' block into the central code pane.

c. The sprite needs to keep moving 'forever', as in until the game ends, so drag a 'forever' block underneath the first block until they click together.

d. Inside your 'forever' block, you need to create the following process (try having a go, or see the code in the next step):

    i. Go to random position
    ii. Show sprite
    iii. Wait 1 second
    iv. Hide sprite
    v. Wait a random number of seconds between 0.25 and 2

e. Your first section of code for the sprite should look like the following (note the colours tell you which section they come from):



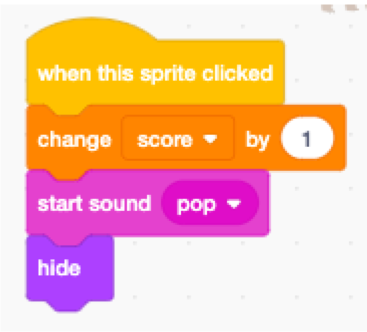
```
when 🏁 clicked
forever
    go to random position ▾
    show
    wait 1 seconds
    hide
    wait pick random 0.25 to 2 seconds
```

f. The second section of code needed will start when the sprite is clicked, which is a different Event block. Drag this into the central pane, somewhere where it won't be confused with the other section of code.

g. Next, you'll need the 'score' variable to change when the sprite is clicked, using the same 'Change [variable] by' block that you used earlier to change the timer. Make sure to set the right variable and make it change by 1.

h. Next, you can add a sound to play once the sprite is clicked, using the 'Play sound [sound]' block, and then make sure to add the hide block to the end, so the user can't click on the sprite at the same position again.

    i. You should now have a second section that looks like this:



```
when this sprite clicked
change score by 1
start sound pop
hide
```

8. Finally, check the Sprite information in the bottom right hand corner of the screen and make sure that the sprite is hidden (using the Show option) by default, as the sprite should only appear when the code says it should.
9. Now test your game and see if it works! If it isn't working as expected, check you've used the right code blocks in the right order, and that they are all connected to each other.

Ways to extend the game

Now you've got started with Scratch, you might want to try out more things. You could start a brand new project, or you might want to try and extend the game. Here are some ideas for how you could change the game:

- Adjust the numbers for the amount of time the sprite appears and disappears for, to make the game harder or easier.
- Add a second sprite, that the user can get more points for clicking on, and make it appear less often.
- Add a second backdrop and find a way to make the game change backdrop, or add a second level that takes place in the second backdrop.
- Add a start screen with music before the user starts playing - you'll need to change the Event blocks used for the existing code to ensure that the game only begins when the user wants it to.

Here an example of an extended game with a title screen and this game with an extra level.

Have a go at trying to extend your game yourself. If you need help developing your extended idea, see this detailed walkthrough on how to add a second 'level'.