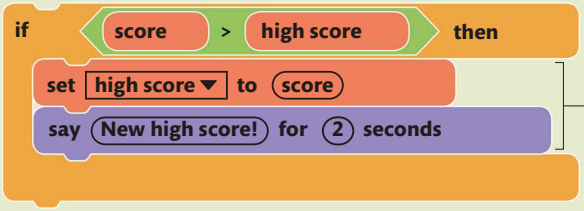
Making decisions

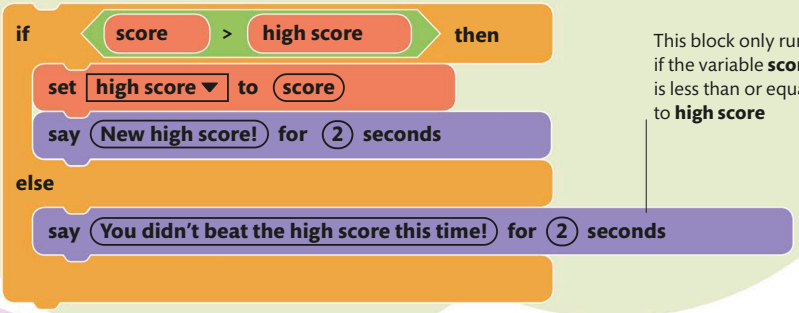
Decisions in Scratch can be made using the **if-then** block or the **if-then-else** block. The required Boolean expression is placed in the diamond-shaped hole. If the expression is true, the blocks inside the **if** block's bracket will run; otherwise, they are ignored.

Make a high score recorder
This script checks whether the player's score is more than the high score. If it is, the **high score** variable is changed to the player's score.



These blocks only run if the variable **score** is more than **high score**

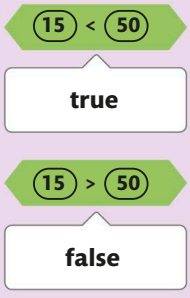
Using the if-then-else block
The **if-then-else** block can be used to add instructions that are run when the Boolean expression is false. This extends the example above with a message if the player's score is not more than the high score.



This block only runs if the variable **score** is less than or equal to **high score**

Boolean expressions

Used for making decisions in programs, a Boolean expression returns a value that is either "true" or "false". For example, the **< operator** block checks whether the number on the left is less than the one on its right.



Logic and decisions

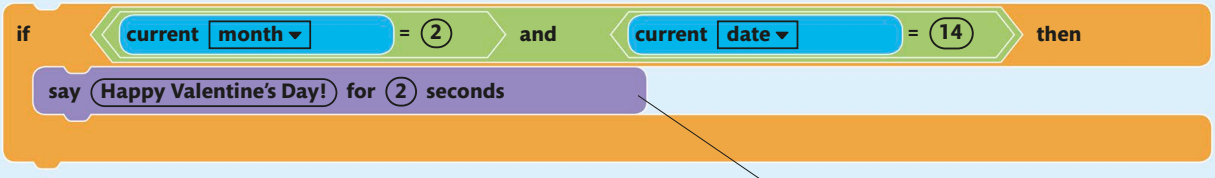
Programs can be made more flexible and useful if they are coded to make decisions about what to do next. They can use variables to control which instruction to run and when to run them.

Combining expressions

Boolean expressions can be combined to make decisions based on more than one factor. Here are some examples using the **current year** Sensing block. It has a menu in it to change the year to other time periods.

Using the and block

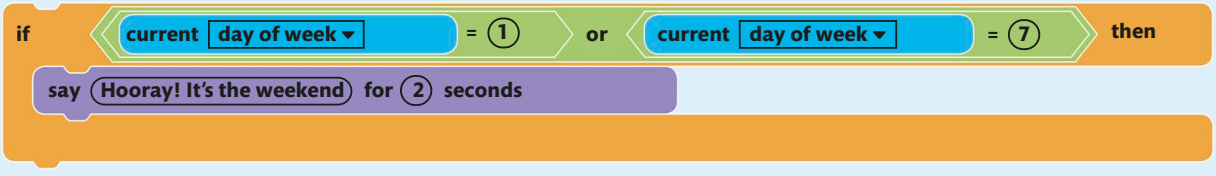
The **and** block has spaces for two Boolean expressions, and it checks whether both expressions are true. In this example, the program checks the month and date to give a special Valentine's Day message.



Find this block under the Looks section of the Blocks Palette

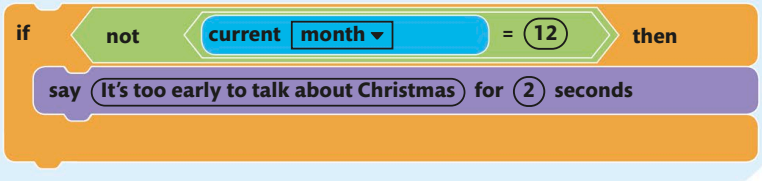
Using the or block

The **or** block checks whether either expression is true. In Scratch, the week starts with day **1** being Sunday. This example checks the day and displays a message if it is Saturday or Sunday.



Using the not block

The **not** block can be used to run instructions if an expression is not true. This example displays a message if the month is not December.



USING BOOLEANS

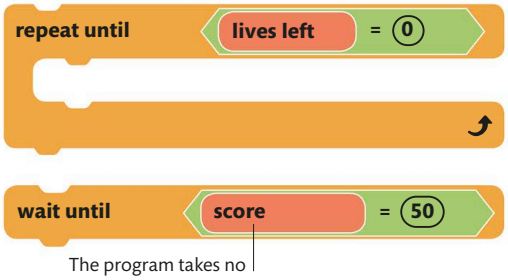
Boolean expressions can be used in other ways as well. Clicking on them in the Code Area allows users to see their results while programming. They can also be combined with **repeat** and **wait** blocks to make a program repeat or pause until something changes.

Using Booleans with loops

The **repeat until** block repeats one or more instructions until an expression is true. In this case, the loop will keep going until the player has no lives left.

Waiting using a Boolean

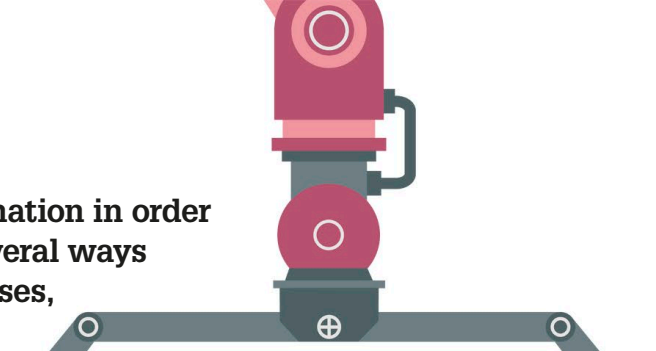
Using a **wait until** block will pause a script until an expression is true. However, it is often better to use a broadcast (see pp.48–49).



The program takes no action until the score is **50**

Input

Sometimes programs need to receive information in order to deliver a result, or output. Scratch has several ways of getting input, including through key presses, by sensing on-screen interactions, and by asking users to type in information.



Types of input

There are a lot of different ways in which information can be entered into a computer system for processing. In Scratch, input to a script is usually in the shape of a rounded block that contains some information or a pointed block that contains a Boolean expression (see p.44).

User interactions
Users can type information into the computer or use the keyboard or mouse to provide movement or positioning instructions.

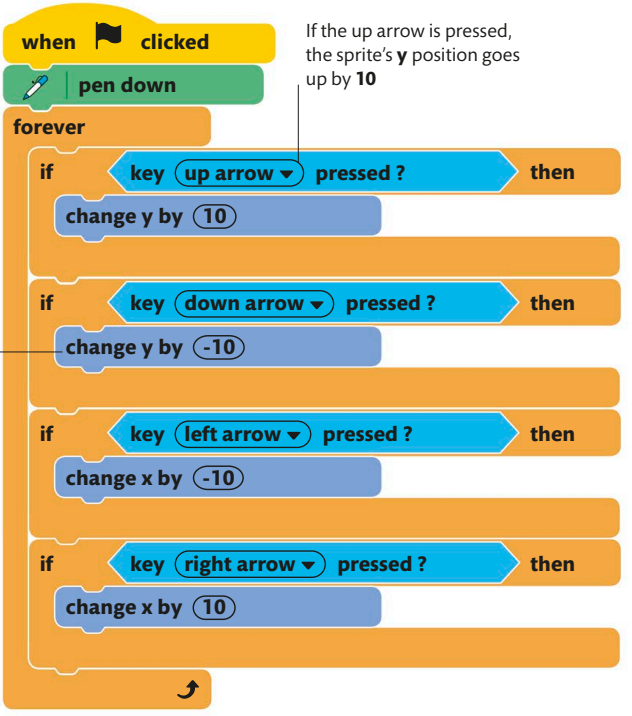
External information
Scratch can detect the username of the logged-in user and the current date and time, and can get translations from Google Translate using an extension.

Sensor input
Some computer systems can sense the outside world. In Scratch, it is possible to detect the loudness of sounds and video camera movements. Sprites can detect each other.

Information for processing
Sometimes programs are given some information to work with. For example, a program might have a list of items to add up for a shopping checkout.

Moving under keyboard control

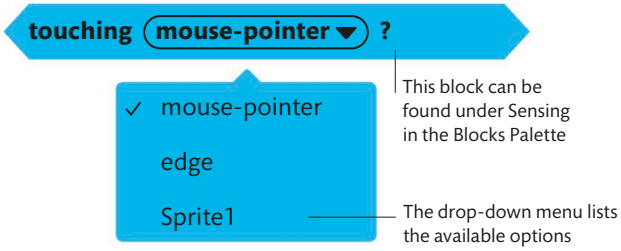
In many games and other programs, the user presses keys on the keyboard to move objects or make things happen. Scratch has an Events block that starts a script when a key is pressed. However, for smoother movement, a script can use a loop that continuously checks for key presses.



Movement script
This script illustrates how to move a sprite under keyboard control using the arrow keys. It uses the pen to draw a line as it goes, so it can also be used as a simple art program.

Collision detection

Video games often need to detect when two objects touch each other, which can require tricky calculations. However, Scratch has this capability built in. The **touching mouse-pointer** block has a drop-down menu that can be used to detect if a sprite is touching the mouse-pointer, the edge of the screen, or another sprite.

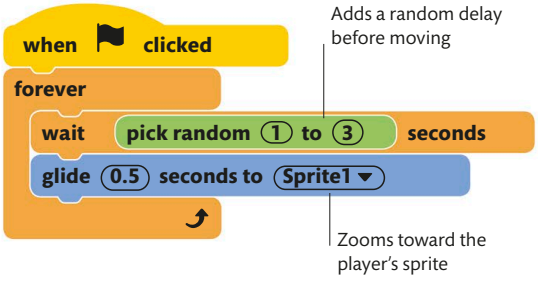


Making the Dodge Ball game

Here is a simple game to demonstrate collision detection in Scratch. Start a new project and add the keyboard control script from the previous page to the Cat sprite.

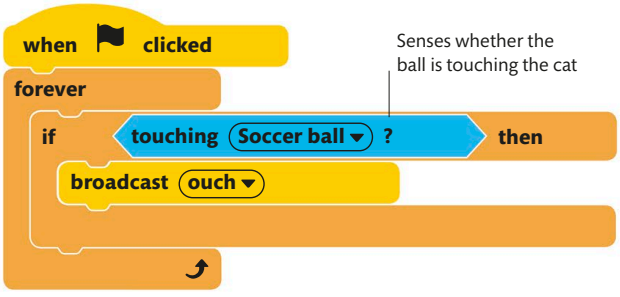
1 Add the Soccer Ball sprite

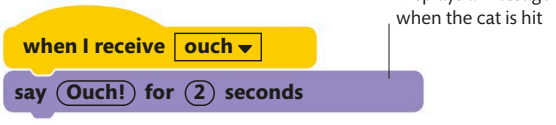
Select the Soccer Ball sprite from the Sprite library and add it to the project. Click on the sprite in the Sprite List and give it this script:



2 Add the cat's code

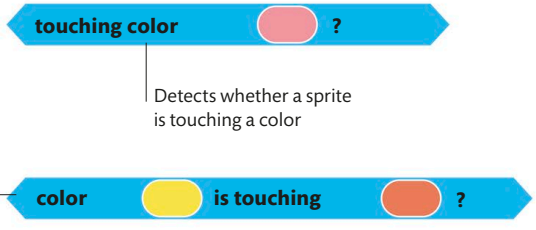
Add these two scripts to the Cat sprite. The first script checks whether the ball has hit the cat. If so, it uses a broadcast (see pp.48–49) to trigger the second script to show a message.





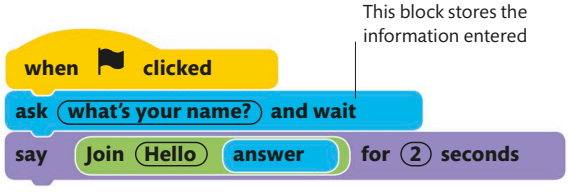
Colliding with colors

Scratch also has the ability to detect whether a sprite is touching a particular color. This can be used, for example, to detect when a ball has crossed a goal line by making the line a certain color that does not appear anywhere else on the screen.



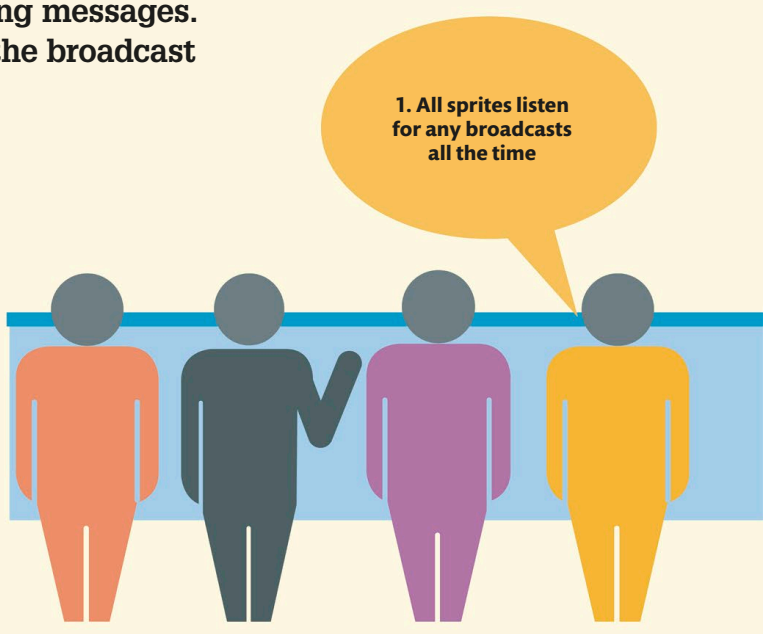
Text input from users

The **ask** block can be used to get users to type in information. The question inside the block appears in the sprite's speech bubble. Whatever the user types goes into an **answer** block. This program greets the user by name.



Sending messages

One of the ways that programs or parts of a program can interact with each other is by sending messages. Scratch has a dedicated Events block—the broadcast block—for this purpose.



Understanding broadcasts

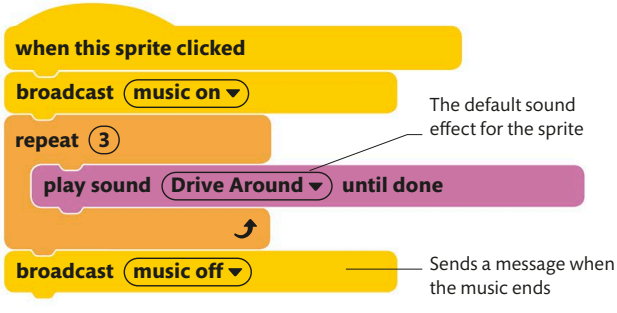
Broadcast blocks make it possible to send a message from a script that can be seen by all other scripts for all other sprites in a program. Scripts can be set to start when they receive the broadcast message either for the same sprite (see p.47) or for different sprites. The **when I receive** block is triggered in response to an incoming message, while the **broadcast** block allows sprites to send messages to other sprites.

Using broadcasts

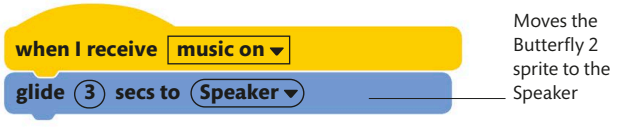
In Scratch, a single broadcast can trigger multiple sprites to run their scripts. In the example below, when the Speaker sprite is clicked, it starts playing music and also broadcasts a message. This message then triggers the other sprites. When the music ends, another message is sent, which makes the Ballerina sprite stop dancing.

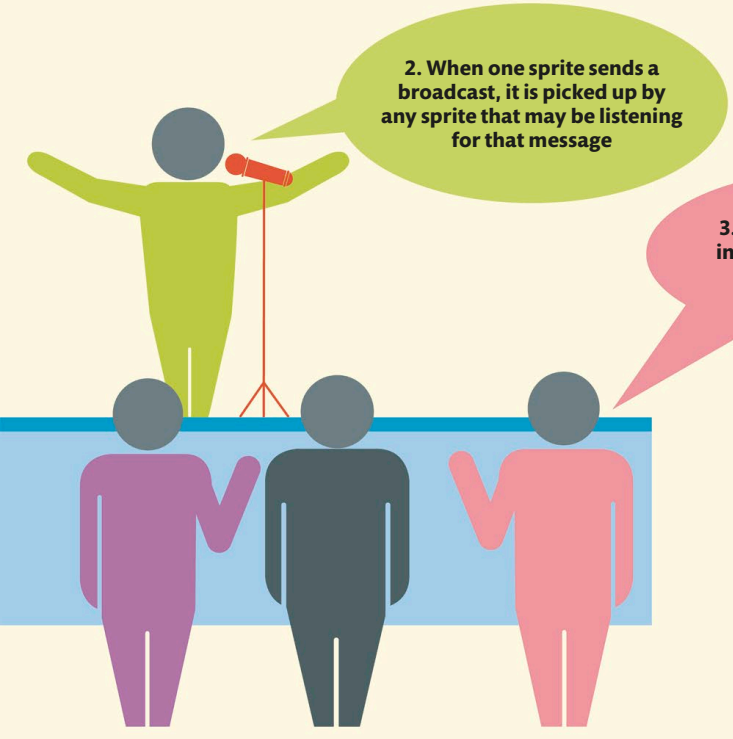
1 **Add new sprites**
Start a new project and delete the default Cat sprite. Choose the Ballerina, Butterfly 2, and Speaker sprites from the library. Click the Choose a Sprite icon and use the Search box in the Sprite library to find them.

2 **Send the broadcast**
Click Speaker in the Sprite List and add the following script to it. Find the broadcast blocks in the Events section of the Blocks Palette and click the menu to enter a new broadcast message.



3 **Trigger the Butterfly 2 sprite**
Next, click and drag the Butterfly 2 sprite on the Stage to move it away from the speaker. Add this script to Butterfly 2. When it receives the message indicating the music has started, the butterfly will fly toward the speaker.



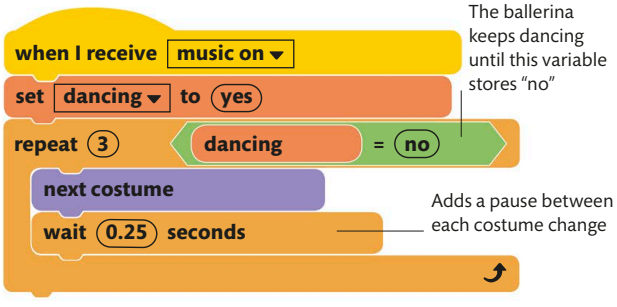


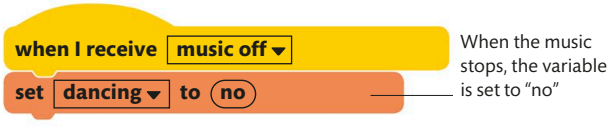
SENSIBLE MESSAGES

The default name for a broadcast message in Scratch is **message1**, but this can be renamed by the user. To make a program easier to understand, it is advisable to change the message to something relevant. The menu in the broadcast blocks can be used to choose a new message name.

4 Trigger the Ballerina sprite

Select the Ballerina sprite in the Sprite List and add these two scripts. A variable is used to store whether the ballerina should be dancing or not. When the music begins, this variable is set to "yes" and the ballerina starts dancing. The dance moves repeat until the dancing variable is set to "no". This happens when the Speaker broadcasts the "music off" message. The outcome is that the ballerina starts dancing when the music starts and stops when it ends.





WHY BROADCAST?

Broadcasts can be used for several purposes in Scratch programs. Here are some of the most popular uses of broadcasting:

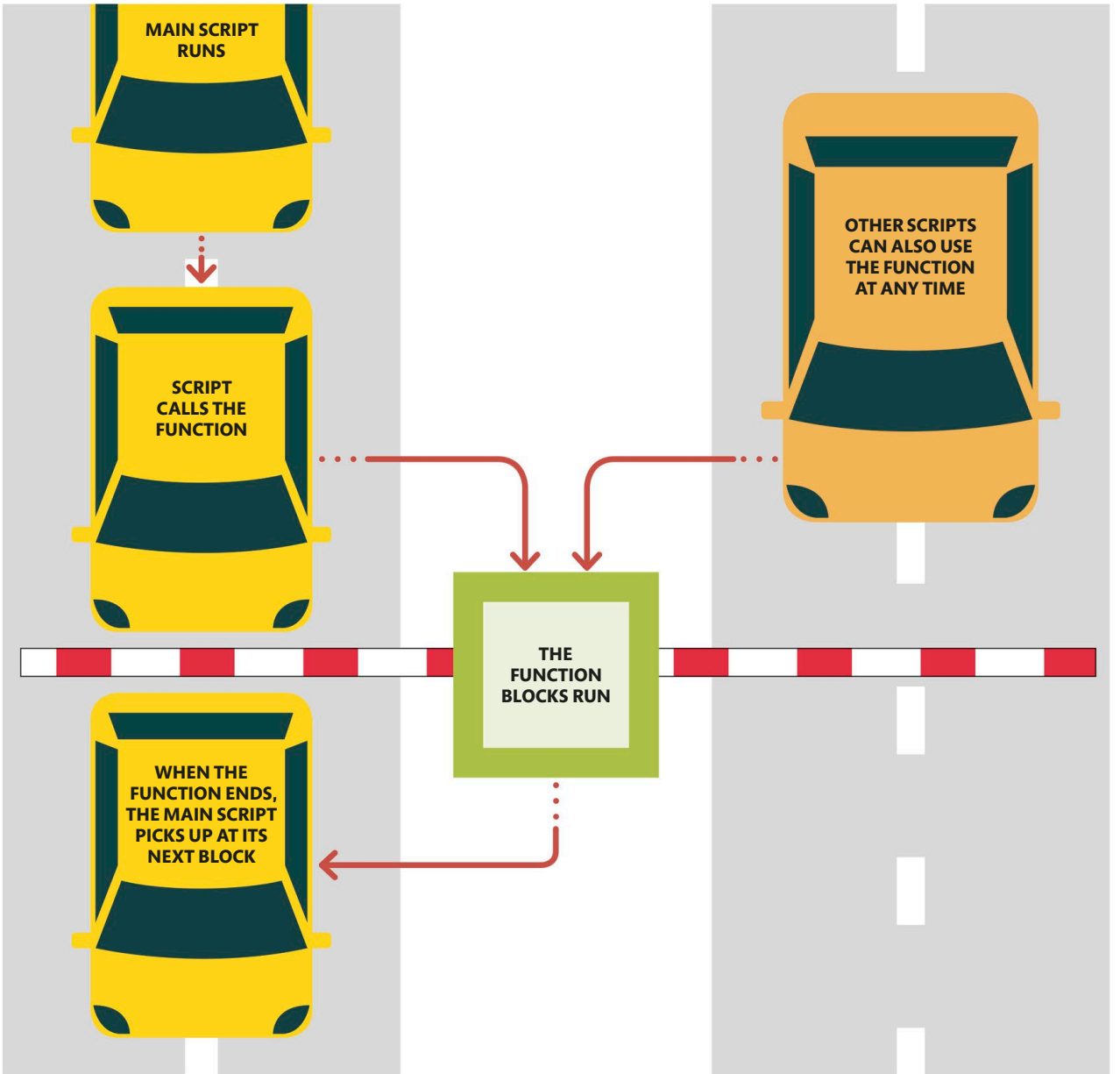
- **Synchronization:** Broadcasts can be used to trigger several scripts across several sprites to start at the same time so that they can be synchronized as a group.
- **Making other sprites move:** Though a sprite can only move itself, it can also tell other sprites when it is time for them to move. For example, clicking the Speaker sprite also triggers the other sprites to move.
- **Enforcing a sequence:** It is possible to make sure scripts run in the right order by using broadcasts to trigger them. The **broadcast and wait** block sends a message, but the script does not continue until every script that receives the message is finished.

Using functions

A function is part of a program that performs a particular task and can be reused. Functions make code easier to read, write, and test. In Scratch, each block is a function, and users can define new blocks.

How the program flows

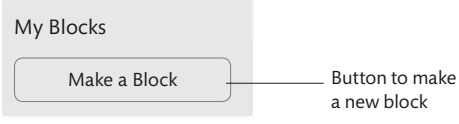
When a script is run, Scratch carries out one instruction block at a time, from top to bottom. When the instruction is a function, Scratch remembers its place in the script and switches to run the instructions in the function. When the function ends, Scratch picks up the main script where it left off. Functions can be used by multiple scripts and can accept information for processing.



Define your own blocks

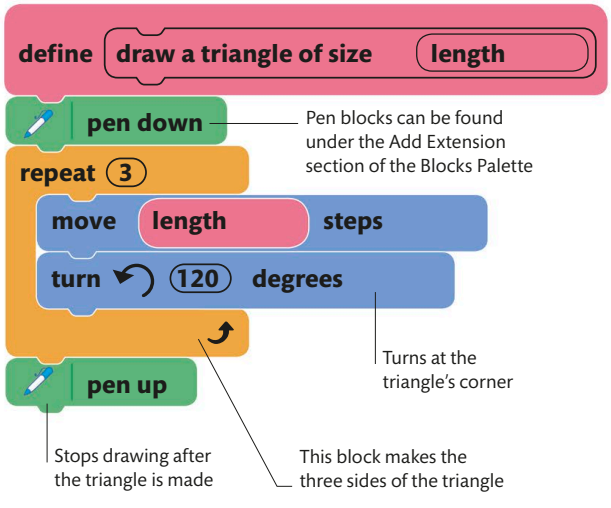
To avoid repeating chunks of code multiple times, Scratch allows users to create their own blocks. Each new block can be made up of several instructions. The example below illustrates how to create a function to draw a triangle and then use it to draw triangles of three different sizes, stacked on top of each other. The end result looks like a fir tree.

1 Make a new block
Go to the My Blocks section of the Blocks Palette and select the Make a Block button. Name the block "draw a triangle of size".

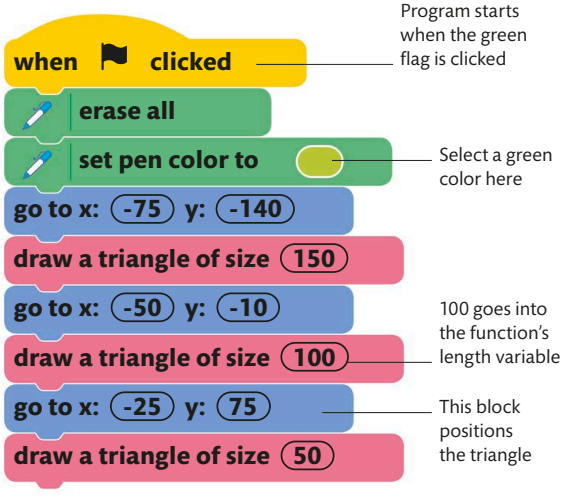


2 Add and name an input
Select the option to add a text input. A new input area will appear in the block. Name this "length" and click OK.

3 Define your script
Add the following instructions to the **define** block. The number the function receives goes into the **length** block. To use it in the script, drag this variable block from the **define** block into the **move 10 steps** block.



4 Using the new block
The new block can now be used in a program. Add this script to the Code Area, and when the green flag is clicked, Scratch will draw three triangles.



WHY USE FUNCTIONS ?

Nearly all programming languages use functions in some way. Here are some advantages of using functions:

- Once a function is written, it can be reused in other programs.
- When each function has a meaningful name, programs are easier to read.
- When functions are reused, programs tend to be shorter.
- It is easier to write and test many small functions rather than one large program.

The example above would be longer, more complicated, and harder to understand if each instance of "draw a triangle" were replaced by all of the blocks in the function.

Travel translator

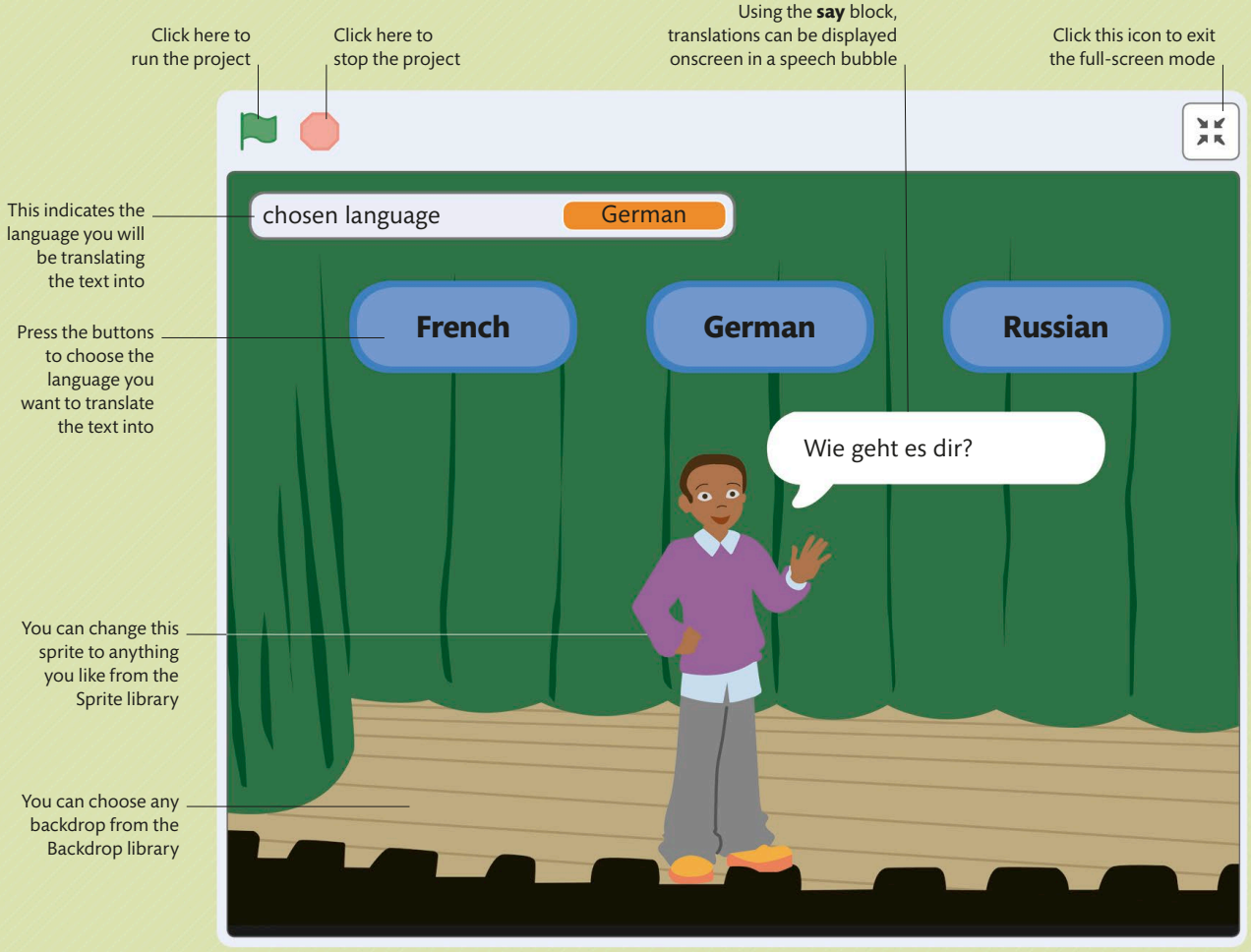
Travelers like to use translation apps to help them communicate in foreign languages. Using Scratch's extension blocks, this project will create a simple text translator. You can use it to translate any text into dozens of different languages. This can be the perfect app for your next vacation.

How the app works

To use this app, users first need to select a language they want to translate the text into. The program then prompts the user to type in the phrase to be translated. Once the user enters a phrase, the app displays the translated text in the chosen language on the screen.

Translating languages

This project uses the Scratch's Translate extension to convert one language into another. The blocks in this extension use the Google Translate API for the translations, so make sure you are connected to the internet when using the app.



YOU WILL LEARN

- How to add and code sprites
- How to use the Paint Editor to change a costume
- How to add Scratch extensions to a project

Time: 15–20 mins

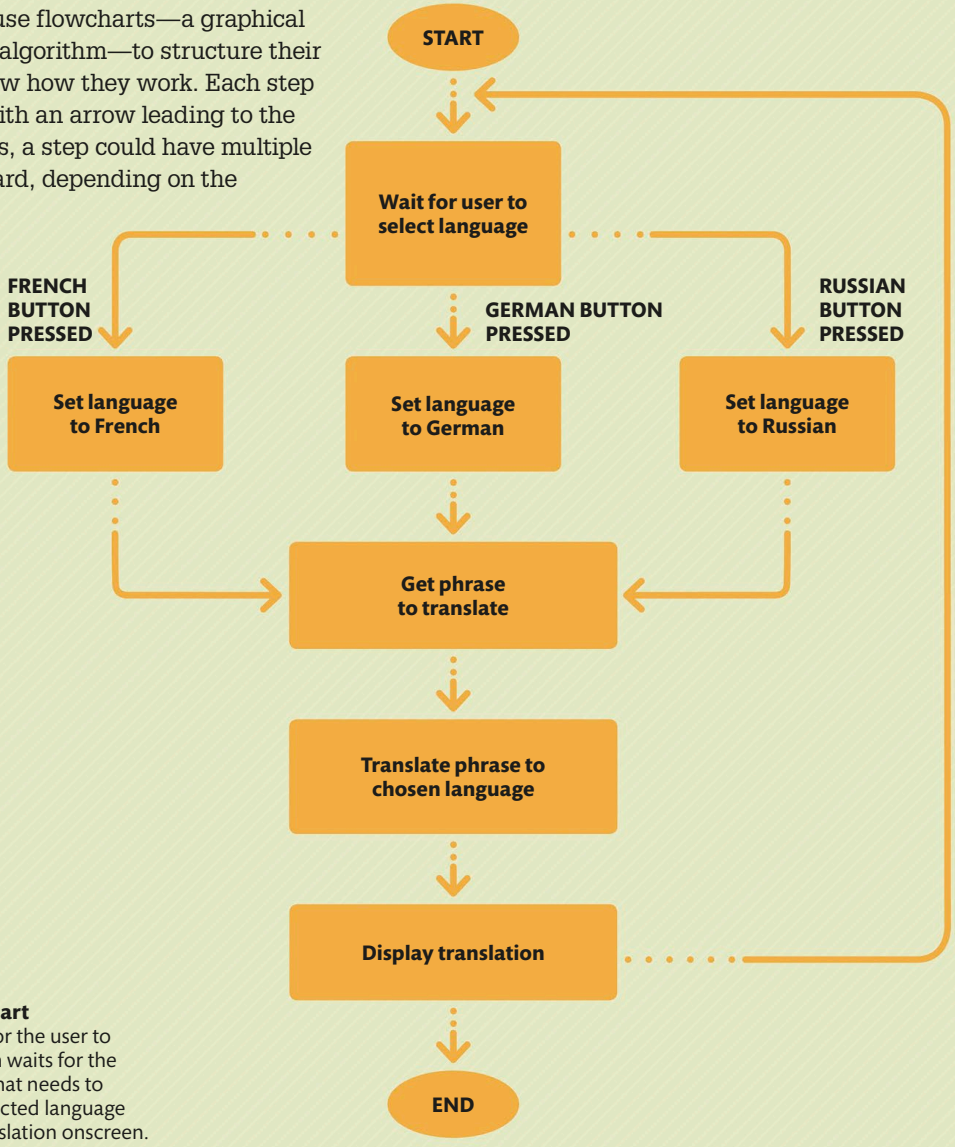
Difficulty level

WHERE THIS IS USED

Good translation apps need to be accurate with their translations. The code used in this project can be reused to translate from a list of languages available in Scratch. You can also experiment by adding blocks that speak the translation out loud so you don't need to worry about your pronunciation.

Program design

Programmers often use flowcharts—a graphical representation of an algorithm—to structure their programs and to show how they work. Each step is shown in a box, with an arrow leading to the next step. Sometimes, a step could have multiple arrows leading onward, depending on the answer to that step.



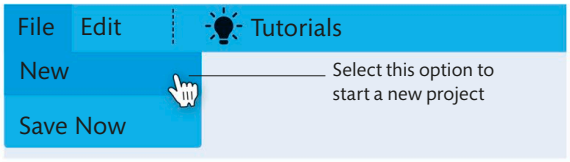
Travel translator flowchart
First, this program waits for the user to choose a language. It then waits for the user to enter the phrase that needs to be translated into the selected language before displaying the translation onscreen.

1 Setting the scene

To begin coding this project, you first need to create an account on Scratch by clicking on "Join Scratch". Next, start a new project and add the sprite and backdrop required to create the app. You can then build the code by joining together colored blocks.

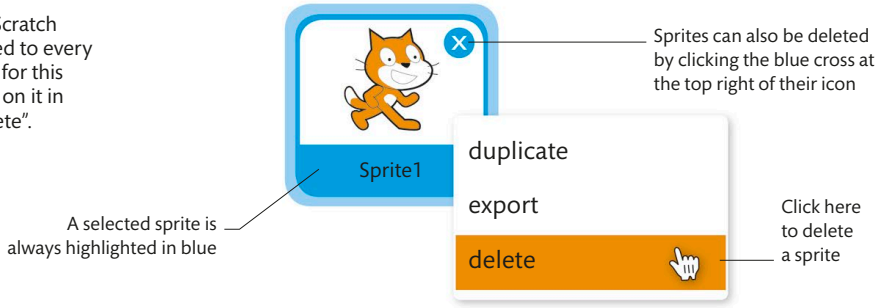
1.1 START A BLANK PROJECT

Let us start with a fresh, blank project. Choose New from the File menu at the top left of the Scratch interface. This creates a brand-new project.



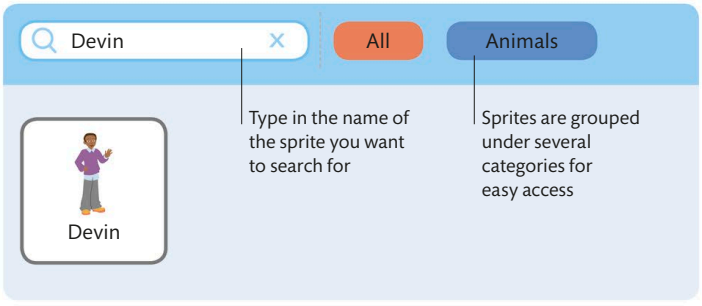
1.2 DELETE THE CAT

Scratch's mascot is the Scratch cat. The cat is automatically added to every new project. You will not need it for this app, so delete it by right-clicking on it in the Sprite List and choosing "delete".

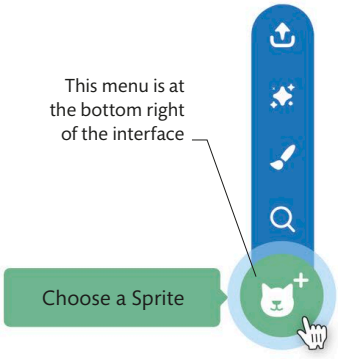


1.3 ADD A NEW SPRITE

Scratch comes preloaded with a lot of sprites that you can use in your projects. These include characters and other objects. For this project, add a sprite named Devin. At the bottom right of the Sprite List, select the Choose a Sprite icon to open the Sprite library. You can scroll through the sprites or type "Devin" in the search bar to locate it quickly. Select the sprite to add it to the project. You can also create your own sprite (see pp.32–33).



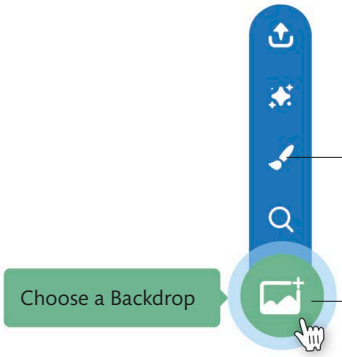
SPRITE LIBRARY



SPRITE LIST

1.4 CHANGE THE BACKDROP

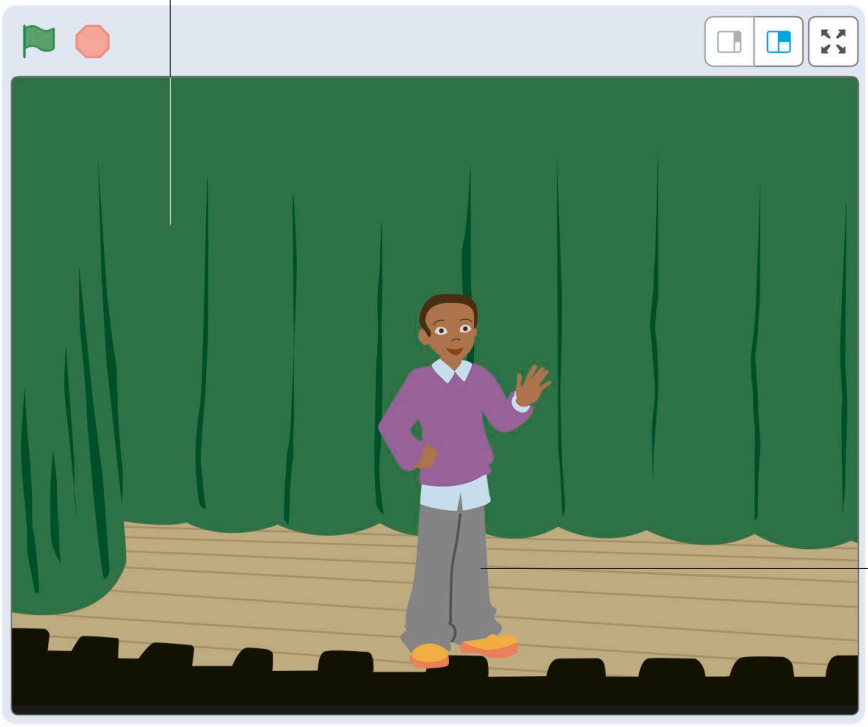
It is now time to set the stage. Scratch uses backdrops to set the scene for each project. Like the Sprite library, the Backdrop library comes with a collection of backdrops you can use. You can even create backdrops if you like. Click the Choose a Backdrop icon at the bottom right of the screen. Type "Theater2" to search for the backdrop, then select it to add it to the project.



Create your own backdrop using the Paint Editor

Select this icon to open the Backdrop library

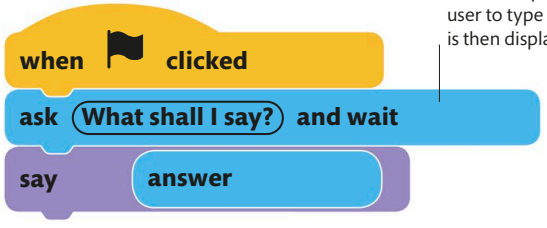
All the action happens in the Stage area, which serves as a miniature screen



Now it looks like Devin is standing on a stage. You can reposition the sprite by dragging it around the Stage

1.5 A SIMPLE PROGRAM

Next, add some code. Click on Devin in the Sprite List and add these blocks by dragging and dropping them from the Blocks Palette on the left-hand side of the interface. The blocks are organized by color, making them easy to find.



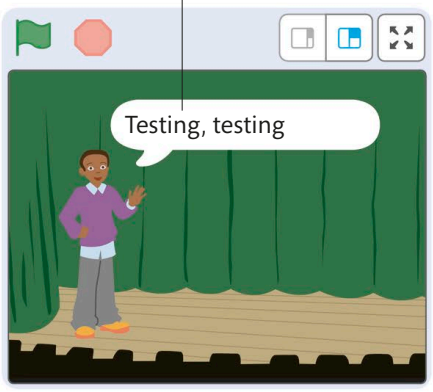
When the project starts, it asks the user to type a message. This message is then displayed on the screen

1.6 TRY IT OUT

Now run the code and see what happens. Click the green flag on the top left corner of the Stage to start the code. The red stop sign next to it will stop the code.

Type in a phrase and press the check icon to test the project

Devin will say whatever you have typed in



2 Adding a language

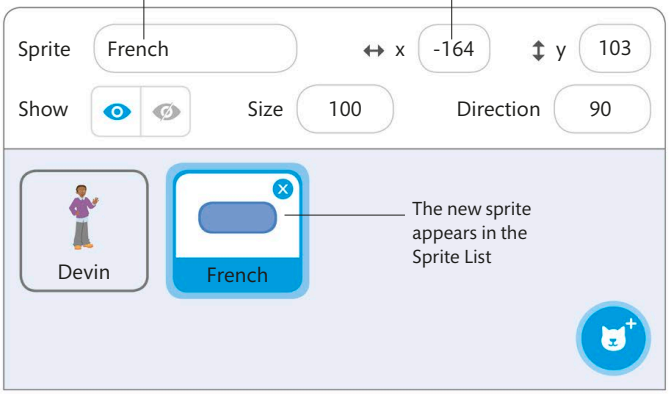
At the moment, Devin can only speak in English—the default language. To proceed with the program, you need to add another language that Devin can translate the text into. The next few steps will help you create a button for the language and will then add the Translate blocks to begin translating.

2.1 ADD A BUTTON

Start by adding a button to select a different language. Click the Choose a Sprite icon in the Sprite List and find the sprite called "Button2". Select it to add it to the project, then drag it to the top left of the Stage. Rename the sprite by clicking the text box in the information panel and typing the word "French".

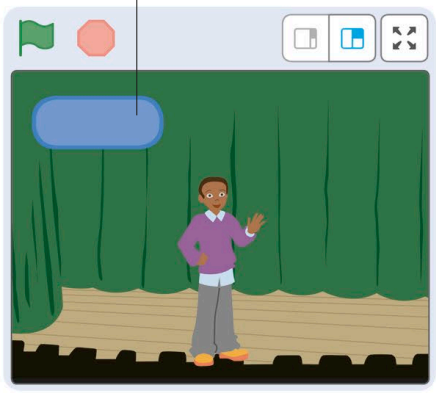
Type here to rename the sprite so you know which language it is for

This gives the position of a sprite on the Stage



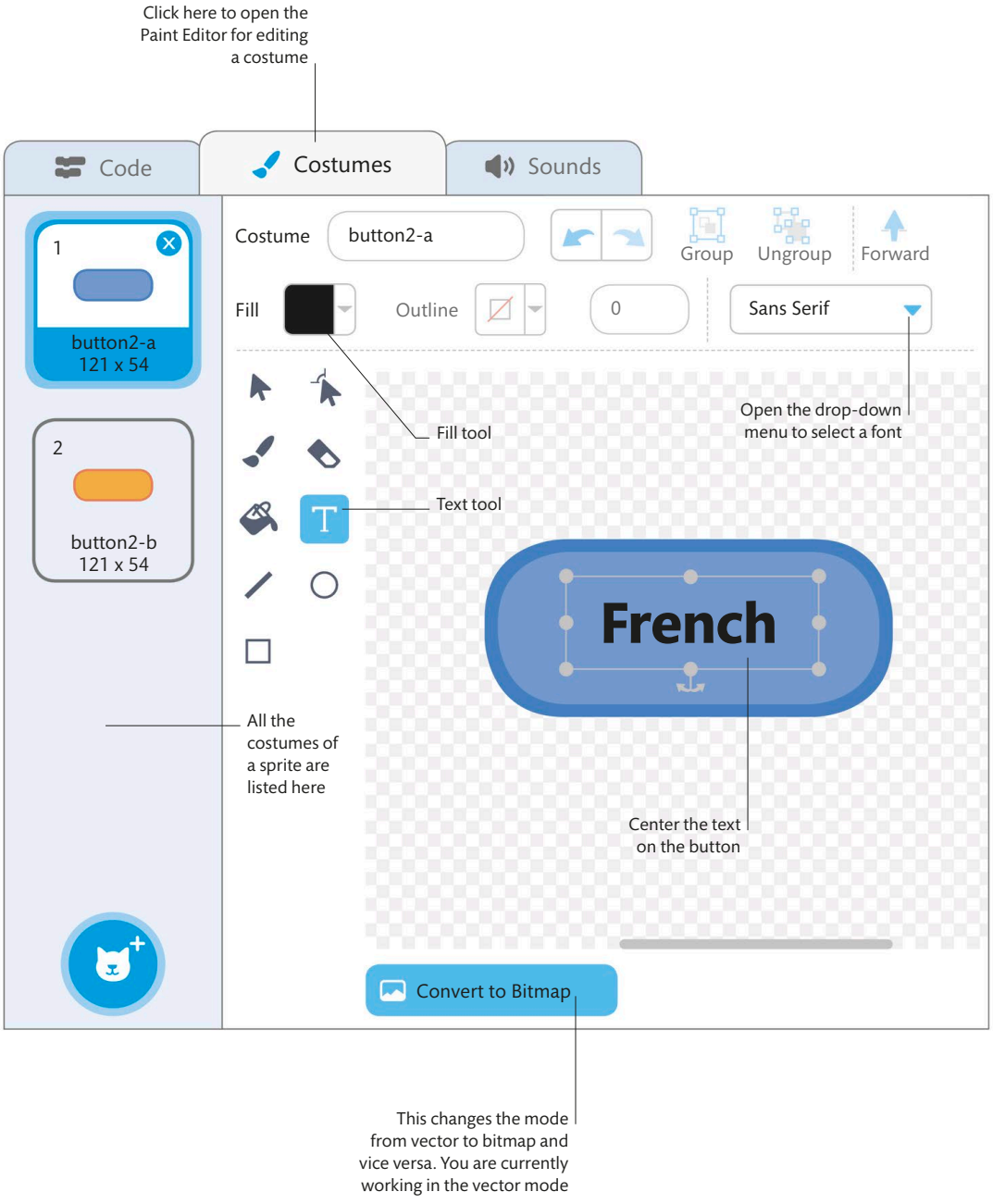
The new sprite appears in the Sprite List

Drag the button to a corner of the Stage



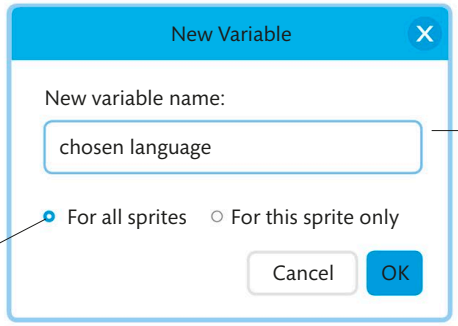
2.2 EDIT THE BUTTON SPRITE'S COSTUME

You can change the way a sprite looks by changing its costume. Select the Button2 sprite to modify it, then click on the Costumes tab at the top left of the interface. This will open Scratch's Paint Editor. You can use this editor to draw your own costumes for the sprites or edit the selected ones. The Text tool lets you add text to an image. Select the Text tool icon and click inside the button, then type "French". This creates a label for the button. You can change the font using the drop-down menu or the color of the text using the Fill option, if you want.



2.3 CREATE A VARIABLE

You can now add some more code to the project. Select the French sprite and then click on the Code tab at the top left of the interface. Go to Variables in the Blocks Palette and click the Make a Variable button. A dialog box will pop up to create a new variable. Name this variable "chosen language" and select OK.



This creates a new variable that will be used to store the name of the language you want to translate the text into

Make sure this option is selected so Devin can use this variable

2.4 SET CHOSEN LANGUAGE TO FRENCH

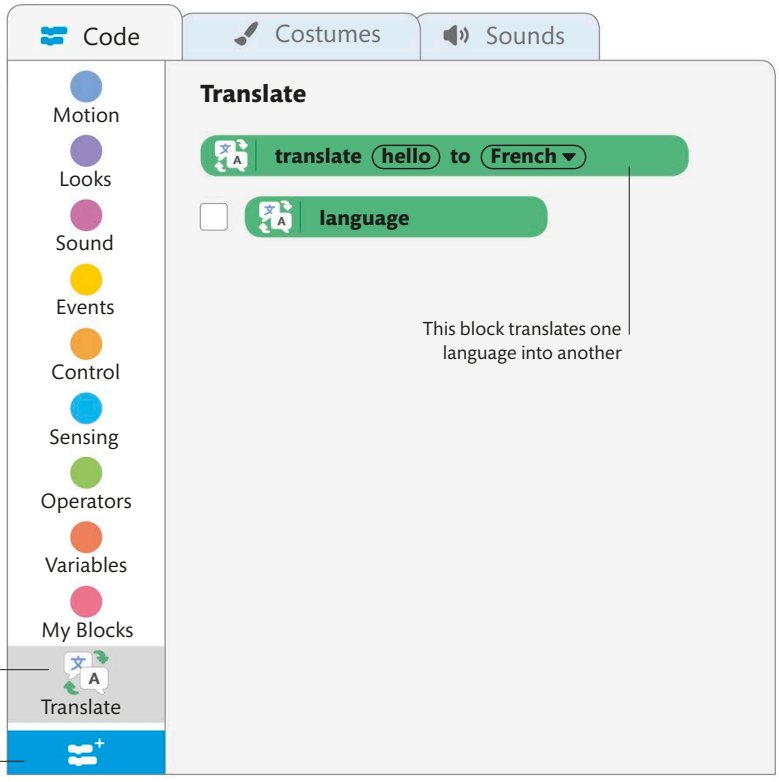
Next, add these blocks of code to the French sprite. When the user clicks on the sprite, it will set the chosen language variable to French. This will help keep track of the language you want to translate the text into later.

Type in the name of the language here

You can find these blocks in the yellow and orange sections of the Blocks Palette

2.5 ADD AN EXTENSION BLOCK

It is now time to start translating. You will need to add some extra blocks to do this. Select the Add Extension button at the bottom of the Blocks Palette. You will see a selection of extra extensions that you can add to your projects. Choose the extension called "Translate". This will add some extra blocks to your palette in a section called Translate.



This block translates one language into another

Every new section is added at the bottom of the Blocks Palette

Click here to open the Scratch extensions

EXTENSION BLOCKS

Scratch's extension blocks allow projects to communicate with hardware or software outside of the Scratch environment. Selecting an extension will add more blocks to the Blocks Palette for you to use.

Music
Use these extension blocks to make music using a variety of instruments and drum sound effects.

Pen
This extension enables a sprite to draw across the screen like a pen. It could be used to create a painting app.

Video sensing
These blocks let you connect Scratch to your webcam. They can be used to detect movement in front of the camera.

Text to speech
Use this extension to make your projects talk. These blocks use Amazon Web Services (an online tool) to read text out loud.

Translate
These blocks let you translate text into a lot of different languages—this is the extension you are using in the current project.

Makey Makey
This lets you connect everyday objects to your computer. These blocks allow you to use connected objects to control your games.

micro:bit
The micro:bit is a palm-sized gadget that you can control with the code blocks in this extension.

LEGO® BOOST
These blocks are designed to make it easy for children (or people new to coding) to build a set of interactive robots.

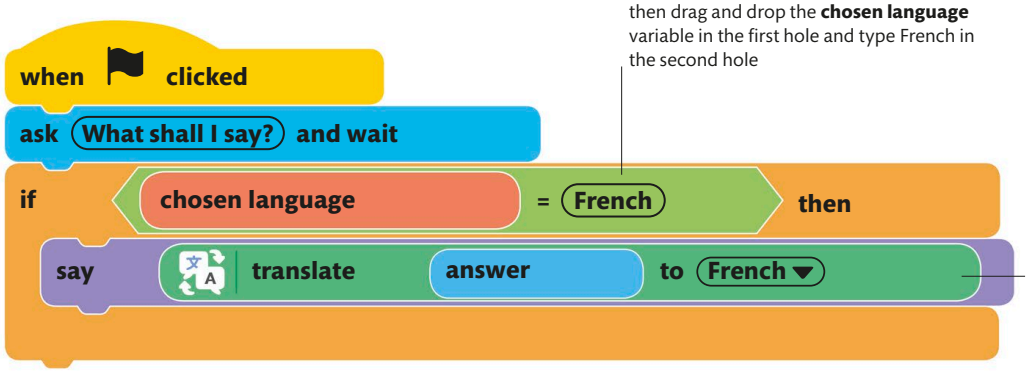
LEGO® Education WeDo 2.0
The WeDo extension blocks are used to control simple robotic projects built with LEGO blocks.

LEGO® MINDSTORMS™ EV3
These extension blocks are used to make projects that control more advanced robots built with LEGO blocks.

GoDirect Force & Acceleration
This extension lets you use an external sensor to record forces and acceleration and send the information to your Scratch project.

2.6 UPDATE DEVIN'S CODE

You can now use the new Translate blocks. Click on Devin and change the code to look like this. If the chosen language is set to French, then Devin will translate the text into French.

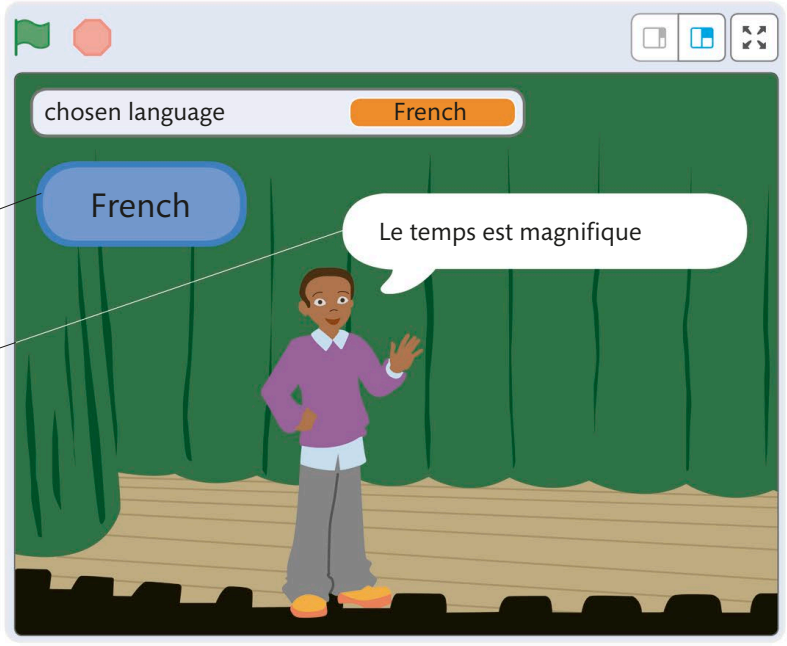


Find this block in the Operators section, then drag and drop the **chosen language** variable in the first hole and type French in the second hole

Update the purple block with this Translate block

2.7 TRY IT OUT

Test the code. Press the French button, then click the green flag to run the project. Next, type a phrase you want to translate and click the check icon to enter it.



Reposition the button to avoid overlapping with other elements on the Stage

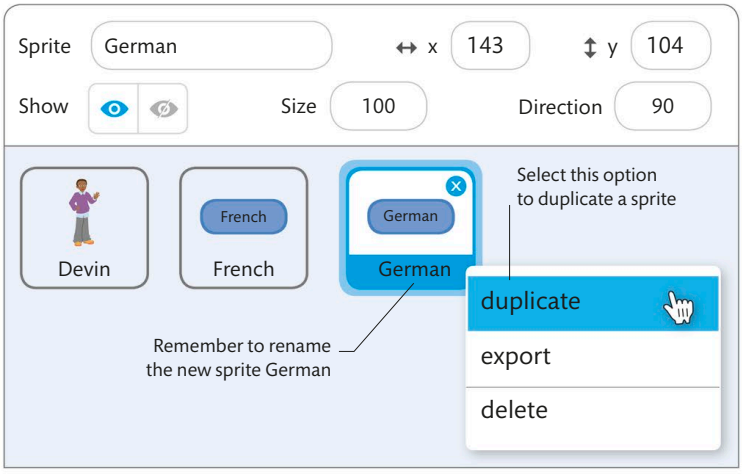
Devin will give the French translation of the text you enter. Here you can see the French translation of "The weather is beautiful"

3 Adding more languages

You can increase the complexity of the app and make it more useful by adding more languages to the project. Start by creating buttons for each new language and then adding the Translate blocks, just like before. You can add as many languages as you like.

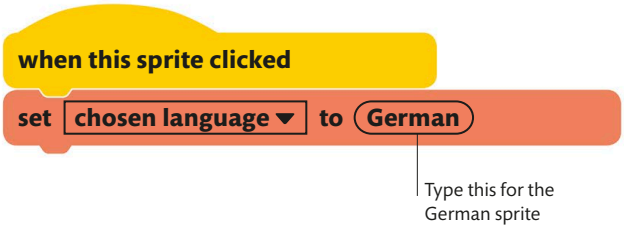
3.1 CREATE MORE BUTTONS

In this step, you will make a few more buttons so you can translate into other languages. Scratch makes it easy to do this. Right click on the French sprite in the Sprite List and choose "duplicate". This creates a copy of the sprite and all the code associated with it. Create two duplicates and rename them German and Russian. You can choose other languages if you like.



3.2 EDIT THE NEW BUTTONS

You now need to make some changes to the code for each of these new sprites. Edit the code for the German and Russian buttons to look like this.

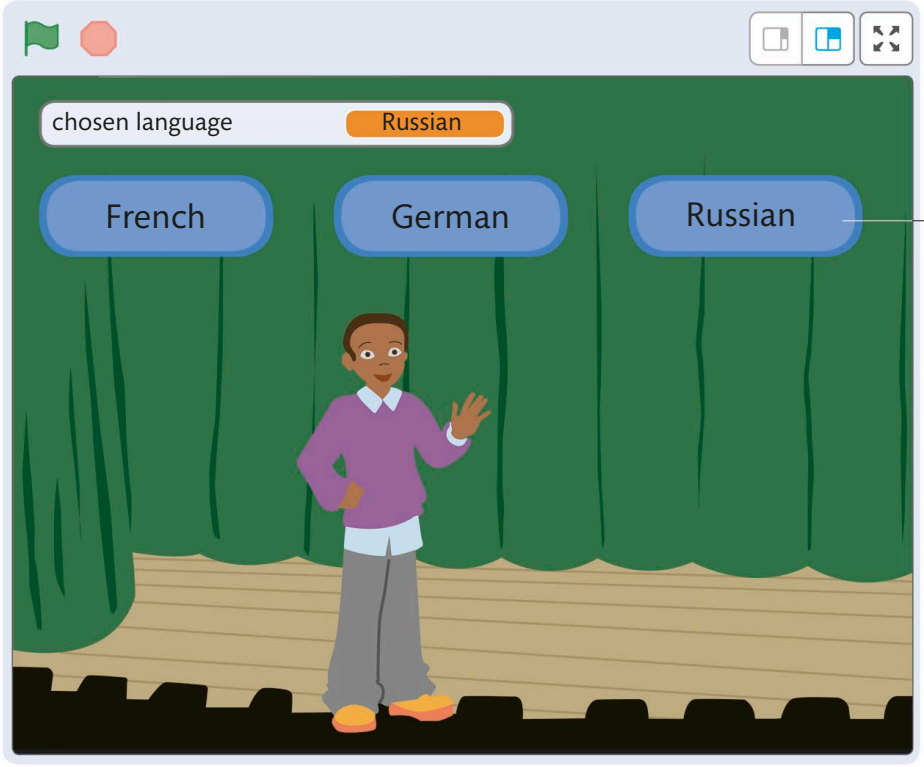


Type this for the German sprite

Type this for the Russian sprite

3.3 UPDATE THE COSTUMES

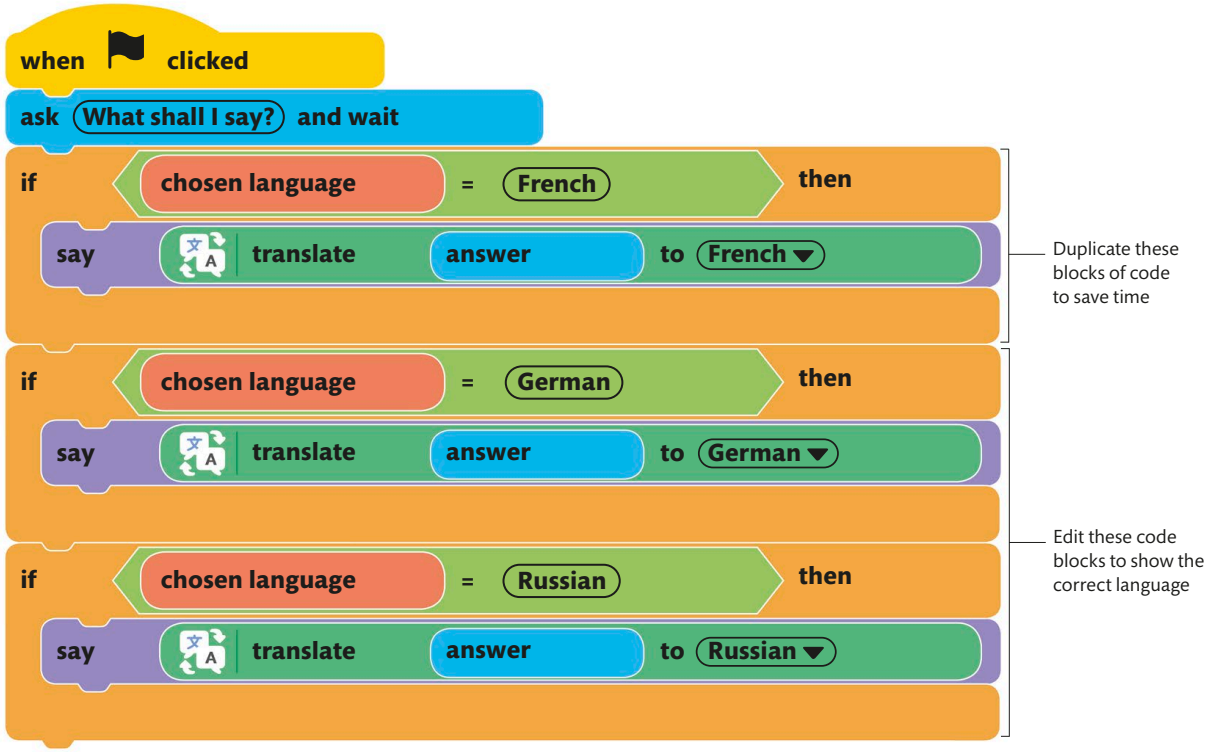
Next, you will also need to edit the costumes for these new sprites so their label reads German and Russian, and not French. Remember to click on the Costumes tab to open the Paint Editor and then use the Text tool. Make sure you are in the vector mode.



Drag the buttons to the top of the Stage and align with the button for French

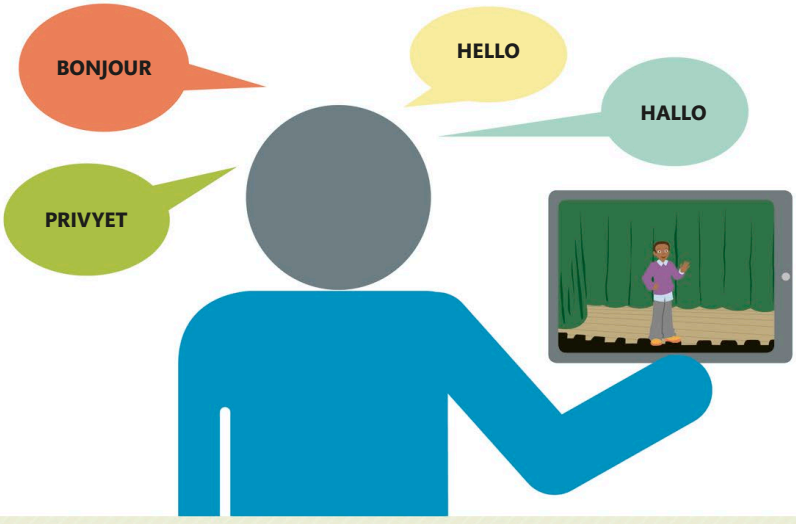
3.4 UPDATE DEVIN'S CODE

Finally, click on Devin in the Sprite List and edit the code to get the correct translations. You can duplicate the original code by right-clicking on it and selecting "duplicate", then just make the edits so it matches the code shown here.



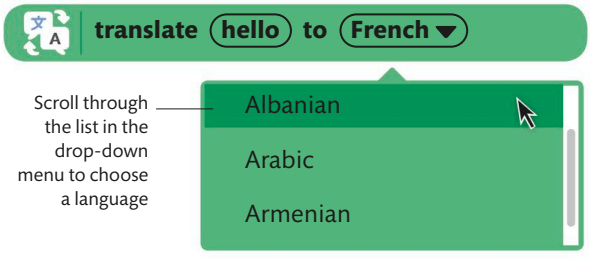
3.5 TRANSLATE NOW

Congratulations! You have now successfully created your first app. Just click on the green flag and start translating.

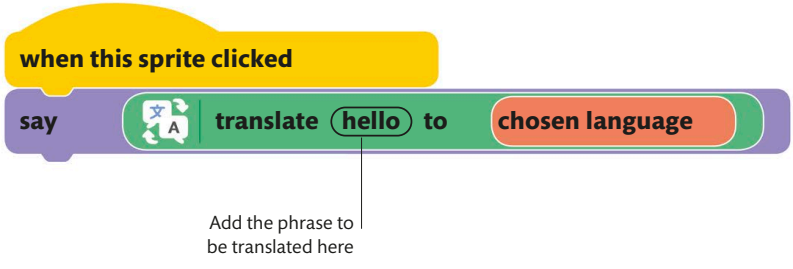


Hacks and tweaks

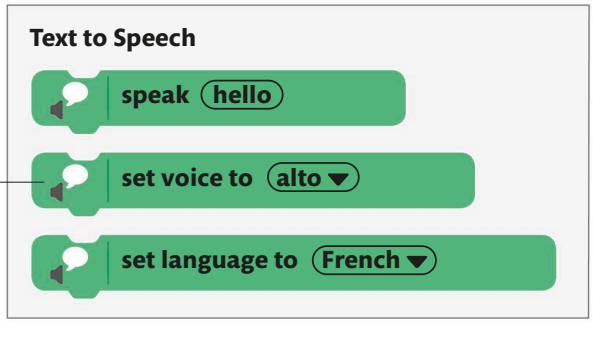
Multilingual
Create additional buttons and code blocks so you can translate more languages. Scratch's Translate blocks allow you to choose from dozens of different languages. Which ones would you like to add?



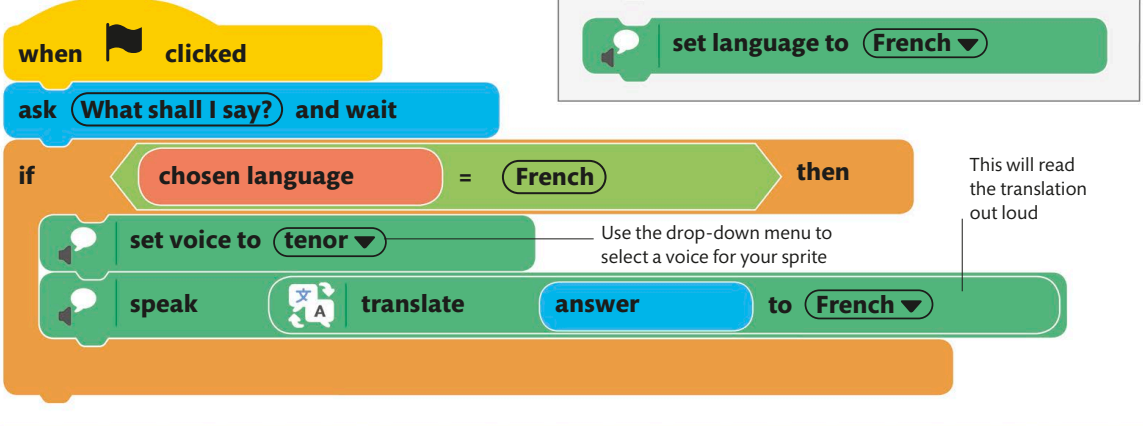
Common phrases
"Hello", "How are you?", "How much is this?"—these common phrases are useful all over the world. Can you adapt your code and use the Translate blocks to see the translations for these useful phrases without having to type them in? You might want to add some dedicated buttons to do this.



Speak it
There is another Scratch extension called "Text to Speech" that can be used to read text out loud. Tweak your code so that the phrases are read out loud. You can then listen and learn how to pronounce the words.



Scratch has five different voices that you can choose from



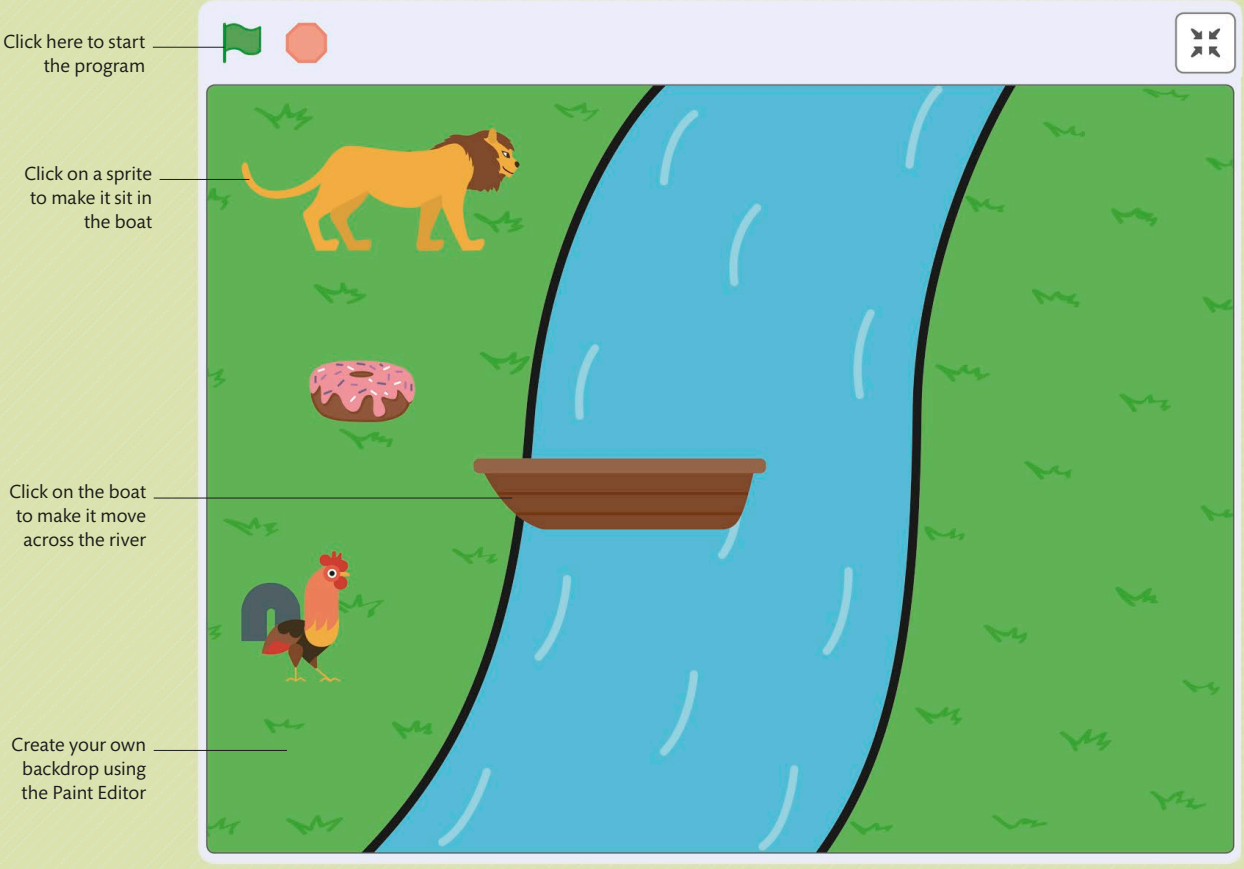
Brain teaser

Brain teasers are a great way to stimulate your brain and help develop logical thinking and cognitive skills. This project uses loops and Scratch's Operators blocks to create a complex brain teaser. The program checks the code each time a sprite moves.

What is the brain teaser?

The aim of the brain teaser is to transport a lion, a donut, and a rooster from one side of a river to the other. You can only fit one sprite in the boat at a time. However, if left unattended together, the lion will eat the rooster, and the rooster will eat the donut. The challenge is to work out the logic and get everything over to the other side of the river safely.

Complex logic
The complexity in the brain teaser arises from the restrictions on what sprites can be transported at the same time or what sprites may be safely left together.



YOU WILL LEARN

- How to use the Paint Editor to create backdrops and sprites
- How to create a simulation
- How to add complex logic to a project

Time: 20–25 mins

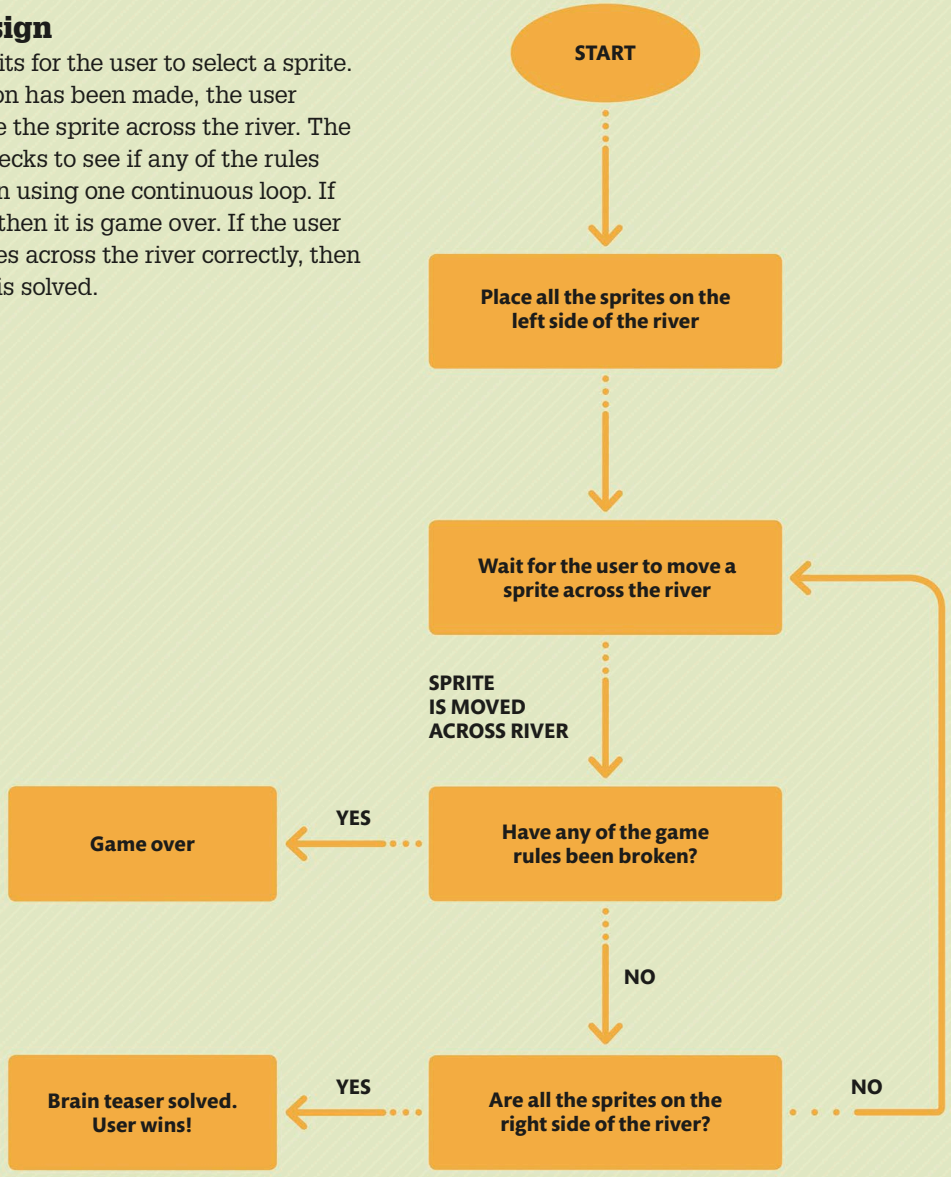
Difficulty level

WHERE THIS IS USED

Computer programs can simulate real-world problems and situations. By using code, it is possible to investigate and test different ways of solving a problem, often much more quickly than it would be to test it in the real world.

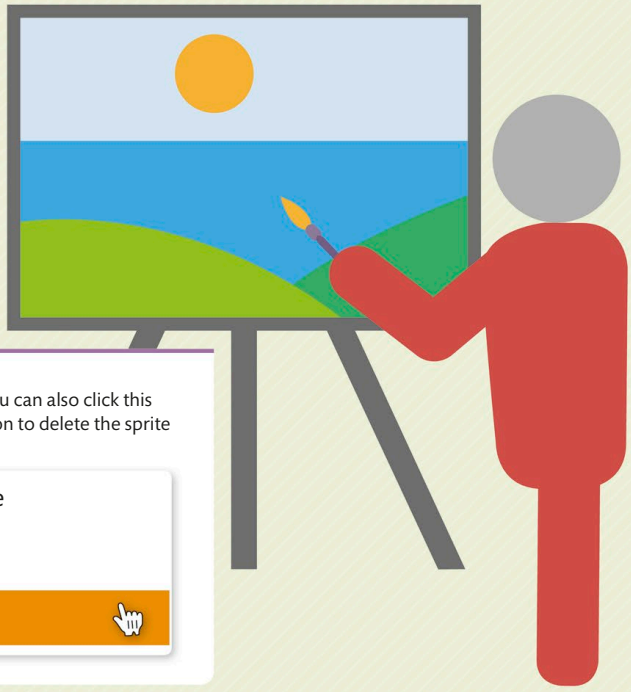
Program design

The program waits for the user to select a sprite. Once the selection has been made, the user attempts to move the sprite across the river. The program then checks to see if any of the rules have been broken using one continuous loop. If a rule is broken, then it is game over. If the user gets all the sprites across the river correctly, then the brain teaser is solved.



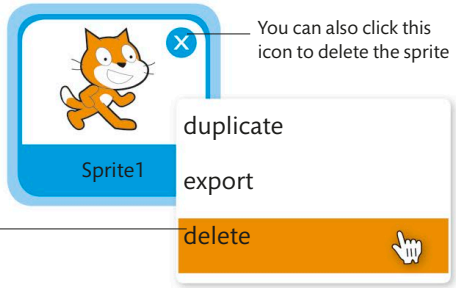
1 Getting started

Starting a project usually involves picking sprites and backdrops from the Scratch library. In this project, however, you will create your own backdrop and even a sprite using the Paint Editor. You will then add some code to make the sprite move across the screen.



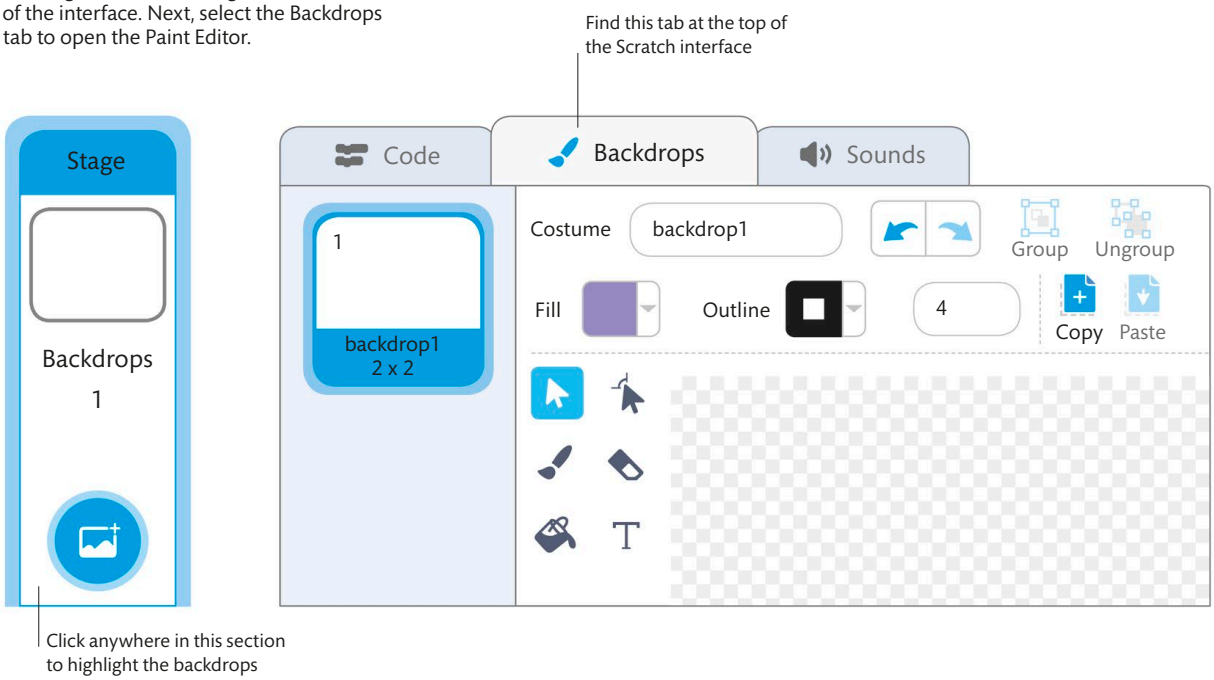
1.1 START A NEW PROJECT

Create a new project and delete the default Cat sprite. Remember, you can do this by right-clicking on the sprite in the Sprite List and choosing "delete".



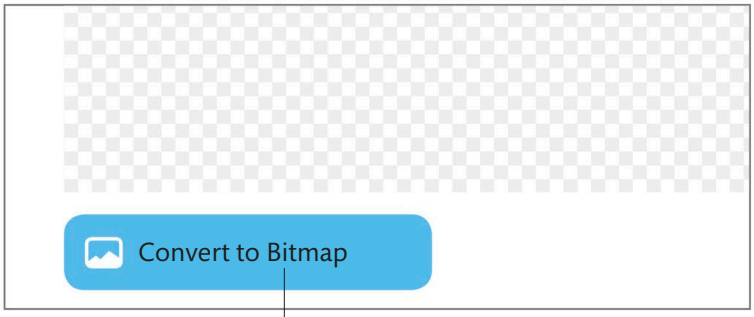
1.2 DESIGN THE BACKDROP

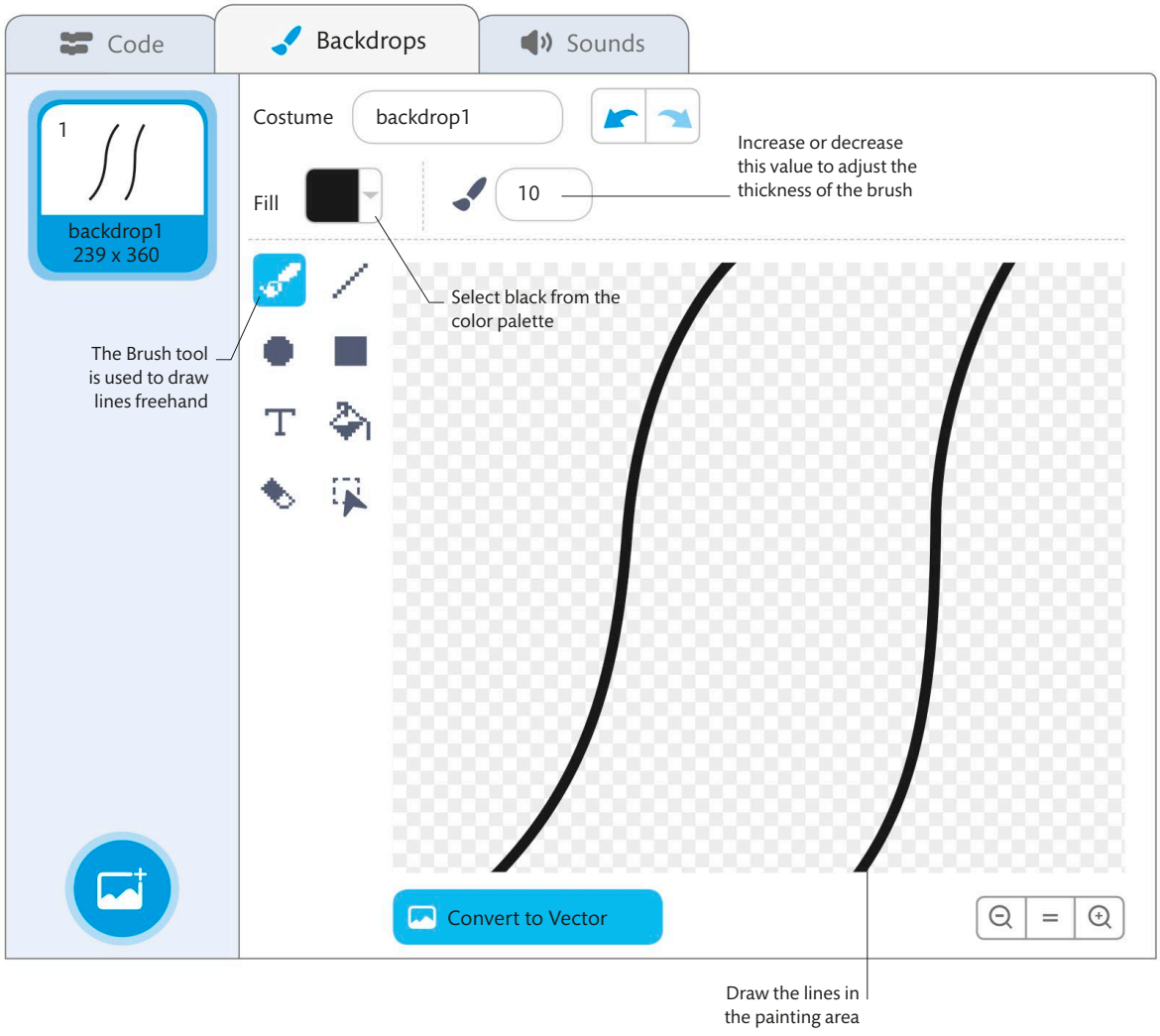
Now create a background for the brain teaser. Click on Backdrops under the Stage section at the right hand side of the interface. Next, select the Backdrops tab to open the Paint Editor.



1.3 START DRAWING

You need to draw a backdrop that has a river in the middle and two grassy banks on each side. First, select the Convert to Bitmap button, then use the Brush tool to draw the edges of the river. Make sure there are no gaps in the lines; they need to go from the very bottom to the very top. It does not matter if the lines are not straight.

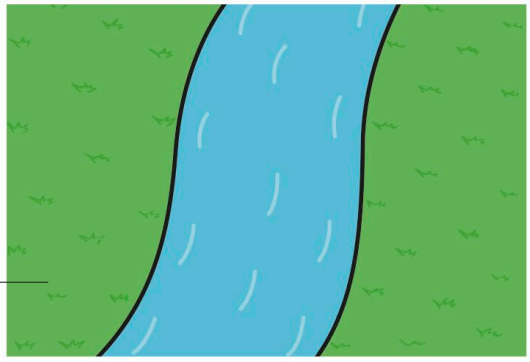




1.4 FILL IT IN

Next, use the Fill tool to color each section of the backdrop. Select the Fill icon (it looks like a paint pot), then choose a color from the Fill menu at the top left of the Paint Editor. Just click on a section to fill it.

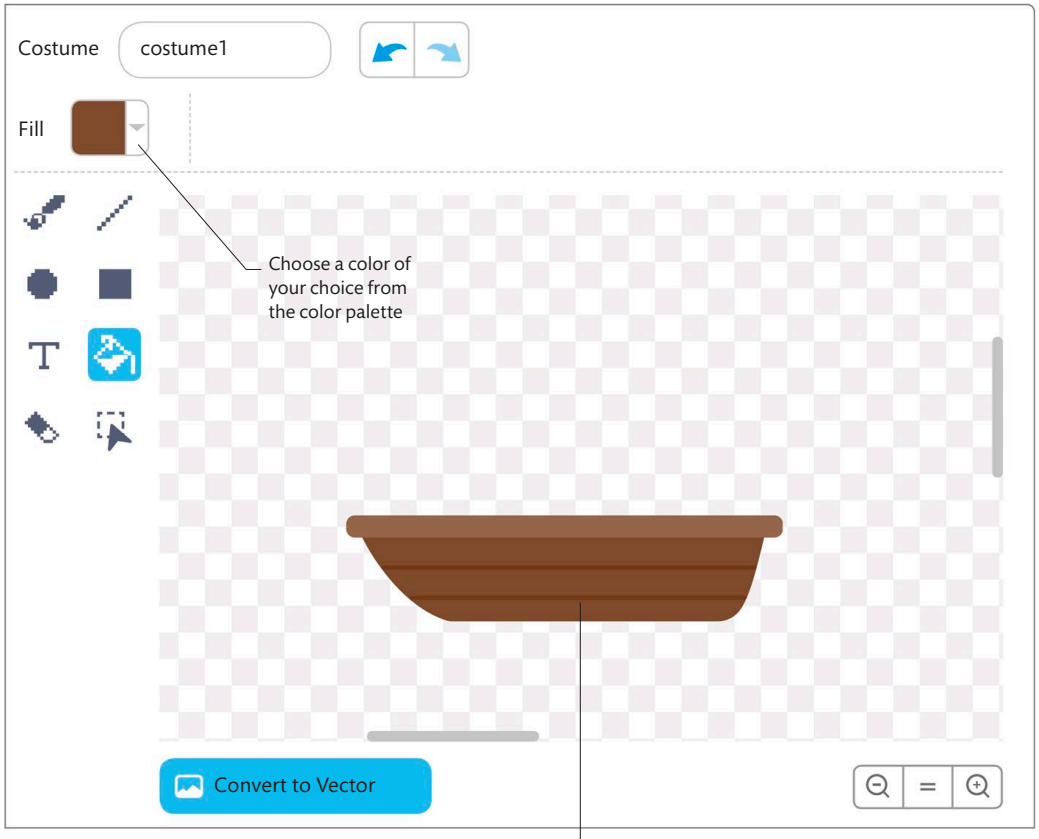
FILL TOOL



Make the grassy sections green and pick a shade of blue to fill in the river. You can add more details if you want the backdrop to look more realistic

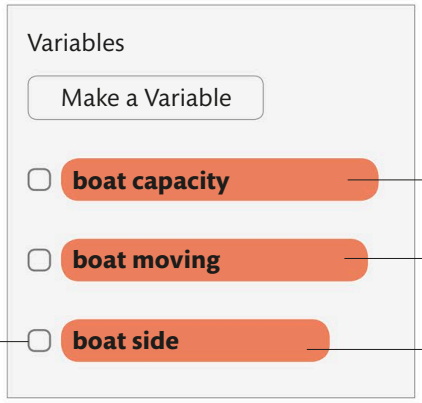
1.5 MAKE THE BOAT

The stage is ready, so let's add the first sprite. Scratch does not have a boat sprite, so you need to create it using the Paint Editor. Click on Choose a Sprite and select the paintbrush from the Sprite menu. This will open the Paint Editor. Select the bitmap mode and use the tools to create a boat. Name this sprite "Boat" and size it correctly. You can also choose the Bowl sprite from the Sprite library, if you like. Just make sure to change its name to "Boat" and its size to "200" in the information panel.



1.6 CREATE SOME VARIABLES

Now you are ready to write the code. Click on the Code tab on the top left of the interface. Make sure the Boat sprite is selected. Then, in the Variables section, click on the Make a Variable button to create three new variables. Name them "boat side", "boat capacity", and "boat moving".



This will be true or false depending on the presence of an object in the boat

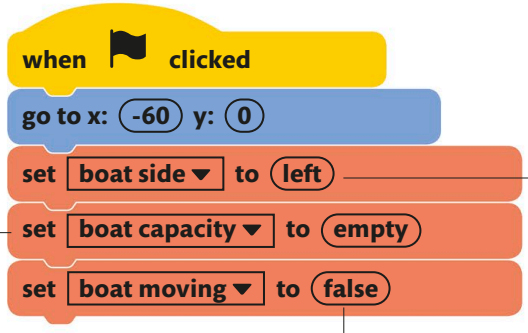
This will be true when the boat is currently moving across the river

Make sure these boxes are unchecked to avoid making these variables appear on the Stage

This will be used to track which side of the river the boat is on

1.7 GET READY TO SAIL

Next, add these code blocks to the Boat sprite. When the project starts, this will place the empty boat at the left side of the river. You might need to adjust the x and y values so that the boat sits correctly on the backdrop you created. You can drag the boat to place it where you want and then copy the x and y position from the sprite information panel into the script.

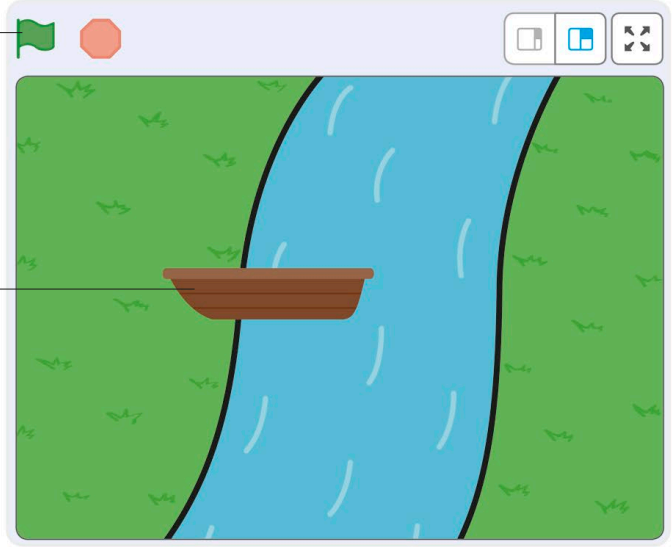


The boat starts on the left side of the river

The boat is empty when the project starts

The boat is currently not moving

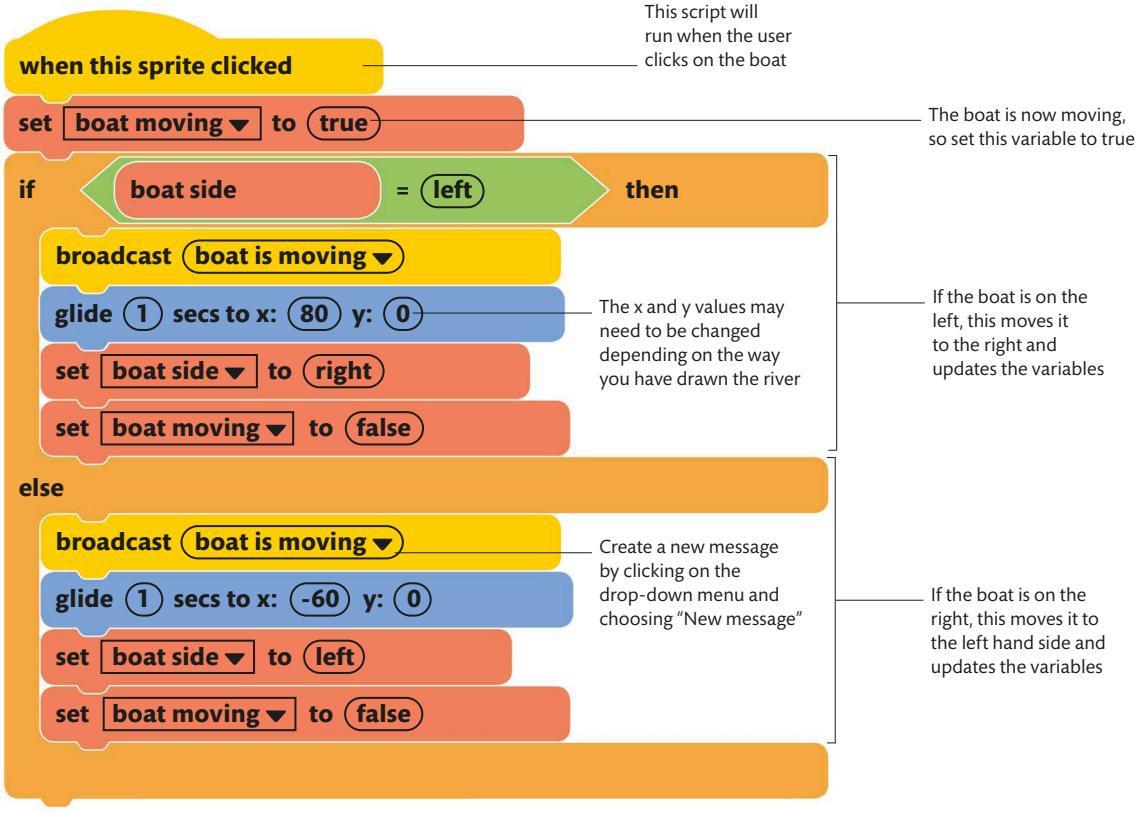
Press the green flag to run the project

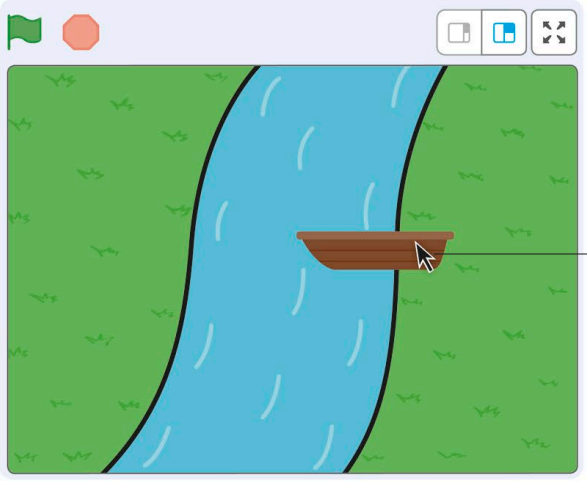


The boat should be sitting on the left side of the river

1.8 START SAILING

Now add this code to make the boat move to the other side of the river when you click on it. Try it out by running the project and clicking on the boat. The boat should automatically move to the other side of the river.





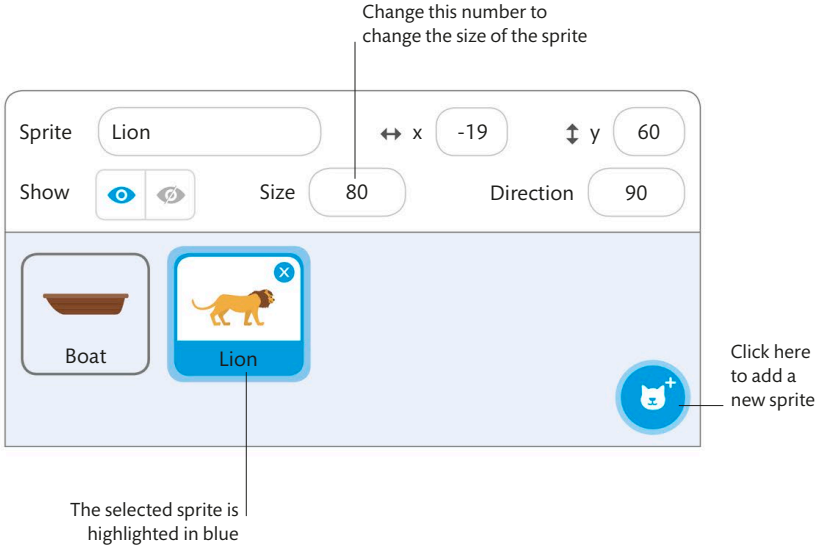
Click on the boat to move it across the river. Click on it again to move it back to its original position

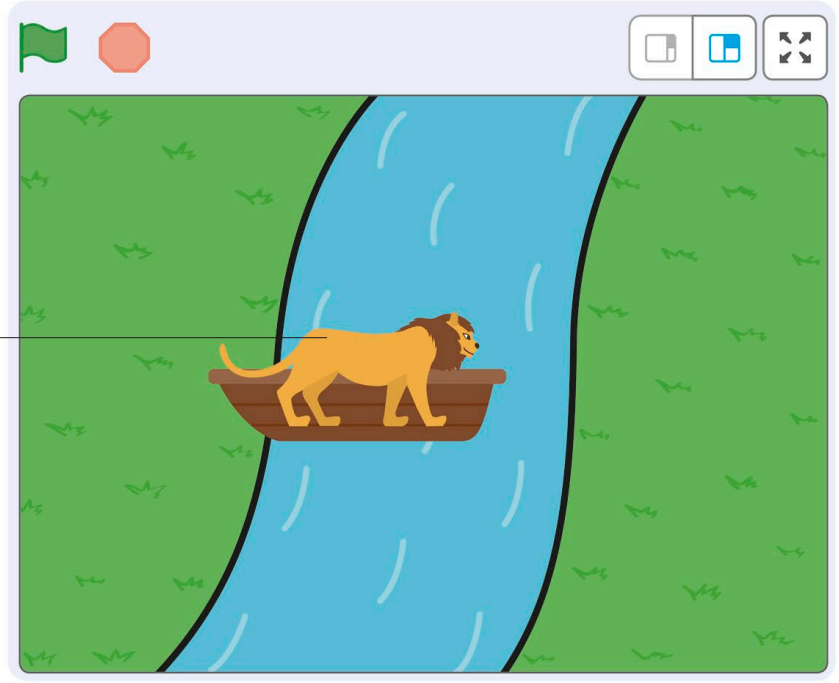
2 Add a new sprite

The boat is ready to set sail. Now you need to add another character to the brain teaser. Add the next sprite, then program it to move along with the boat.

2.1 ADD THE LION

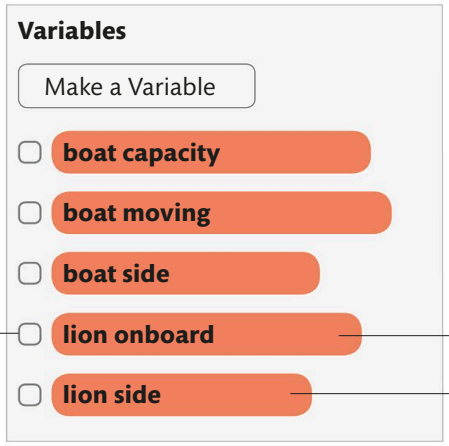
Go to the Sprite library and look for the Lion sprite. Select it to add it to the project, then change its size to 80 in the sprite information panel.





2.2 CREATE THE VARIABLES

With the Lion sprite still highlighted, go to the Variables section of the Blocks Palette and create two new variables for this sprite. Call them "lion side" and "lion onboard". If you need to rename or delete a variable, right-click or Ctrl + click on it.



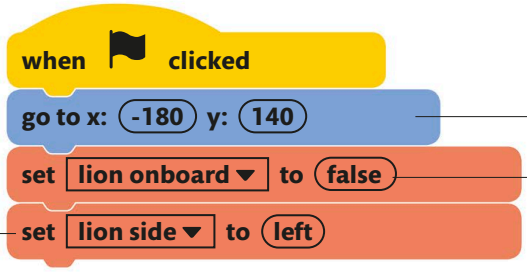
Make sure these boxes are unchecked

This will be true if the lion is on the boat

This will be used to track which side of the river the lion is on

2.3 PLACE THE LION

Next, add these code blocks to the Lion sprite to position it on the left-hand side of the river when the project starts. Remember, you might need to adjust the x and y values to suit your backdrop.



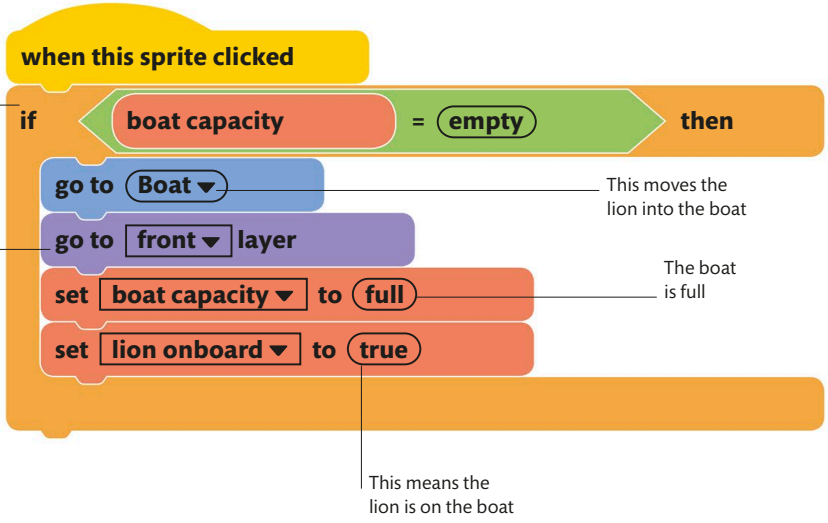
Places the lion at the top left of the Stage when the brain teaser starts

This means the lion is not on the boat

Puts the lion on the left-hand side of the river

2.4 GET ON THE BOAT

When the user clicks the lion, it must move onto the boat. Add this code to do that. Click on the green flag to try it out.



These blocks are run only if the boat is empty. This prevents more than one object from being inside the boat at the same time

This moves the lion into the boat

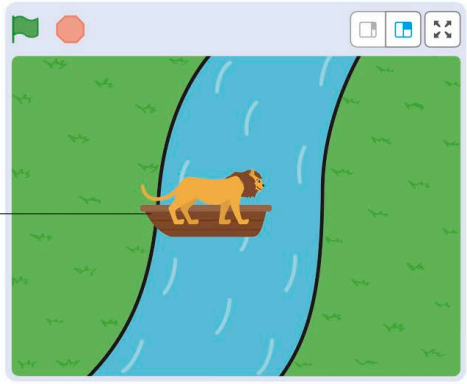
Find this block in the Looks section of the Blocks Palette. It ensures that the Lion sprite stays in front of the boat

The boat is full

This means the lion is on the boat

2.5 MOVE THE LION WITH THE BOAT

When the boat moves across the river, the lion needs to move with it. This will make it seem like the lion is sailing to the other side of the river. Add the following code to the Lion sprite to do this, then run the code. See if you can move the lion across the river in the boat and then back again.



Click on the lion to move it along with the boat

These blocks are run when the user clicks on the Boat sprite

```
When I receive [boat is moving]
if <(lion onboard) = (true)> then
    repeat until <(boat moving) = (false)>
        go to (Boat)
        go to [front] layer
    if <(boat side) = (right)> then
        go to x: (165) y: (99)
        set [lion side] to (right)
    else
        go to x: (-180) y: (140)
        set [lion side] to (left)
    set [boat capacity] to (empty)
    set [lion onboard] to (false)
```

This loop runs only if the lion is on the boat

The lion moves along with the boat until the boat stops

If the boat ends up on the right, this moves the lion to the grassy area on the right-hand side of the river and updates the variables

If the boat ends up on the left, this moves the lion to the grassy area on the left-hand side of the river and updates the variables

Ensures the boat is empty

This means the lion is no longer on the boat

3 Add more sprites

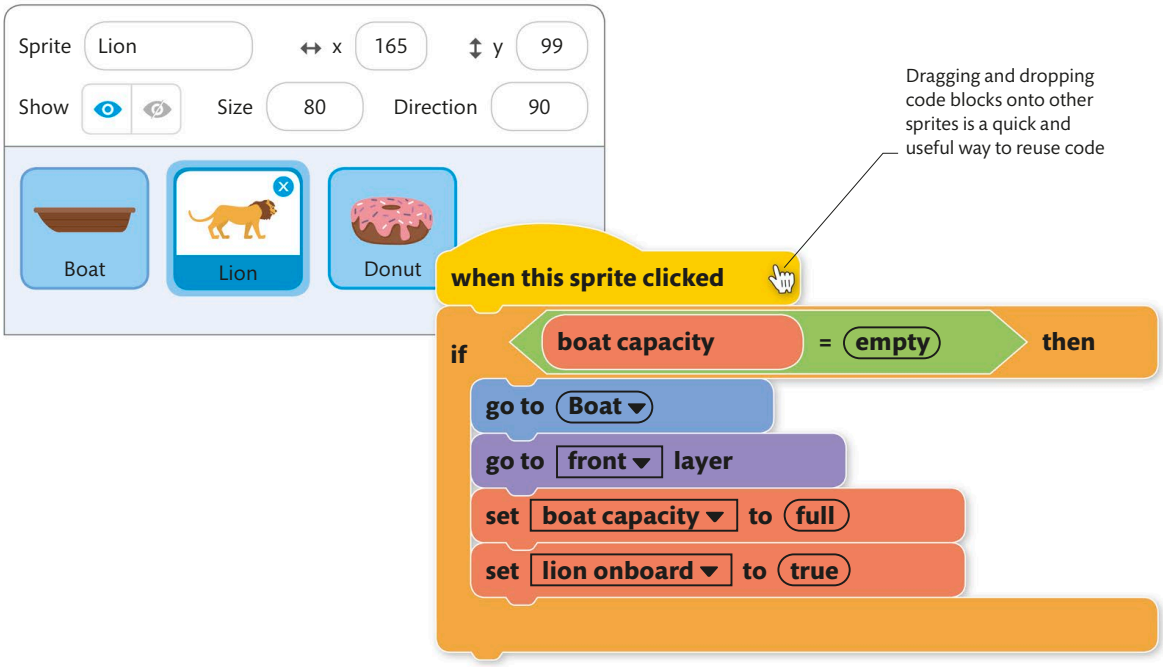
The next step is to add more characters to increase the complexity of the project. You need to code the new sprites in exactly the same way as the lion in the previous steps. You can then add some rules that will constantly check if the correct logic has been applied to solve the brain teaser.

3.1 ADD A DONUT

Go to the Sprite library and look for the Donut sprite. Select it to add it to the project. It is a big sprite, so change its size to 50 in the information panel.

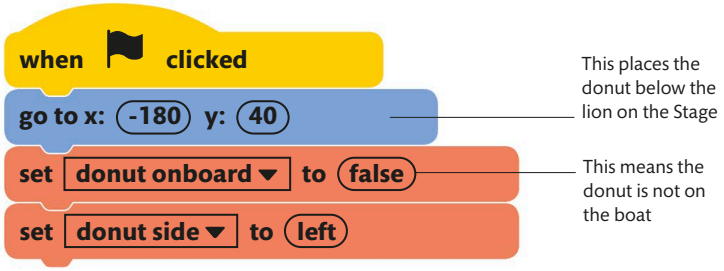
3.2 COPY CODE FROM THE LION

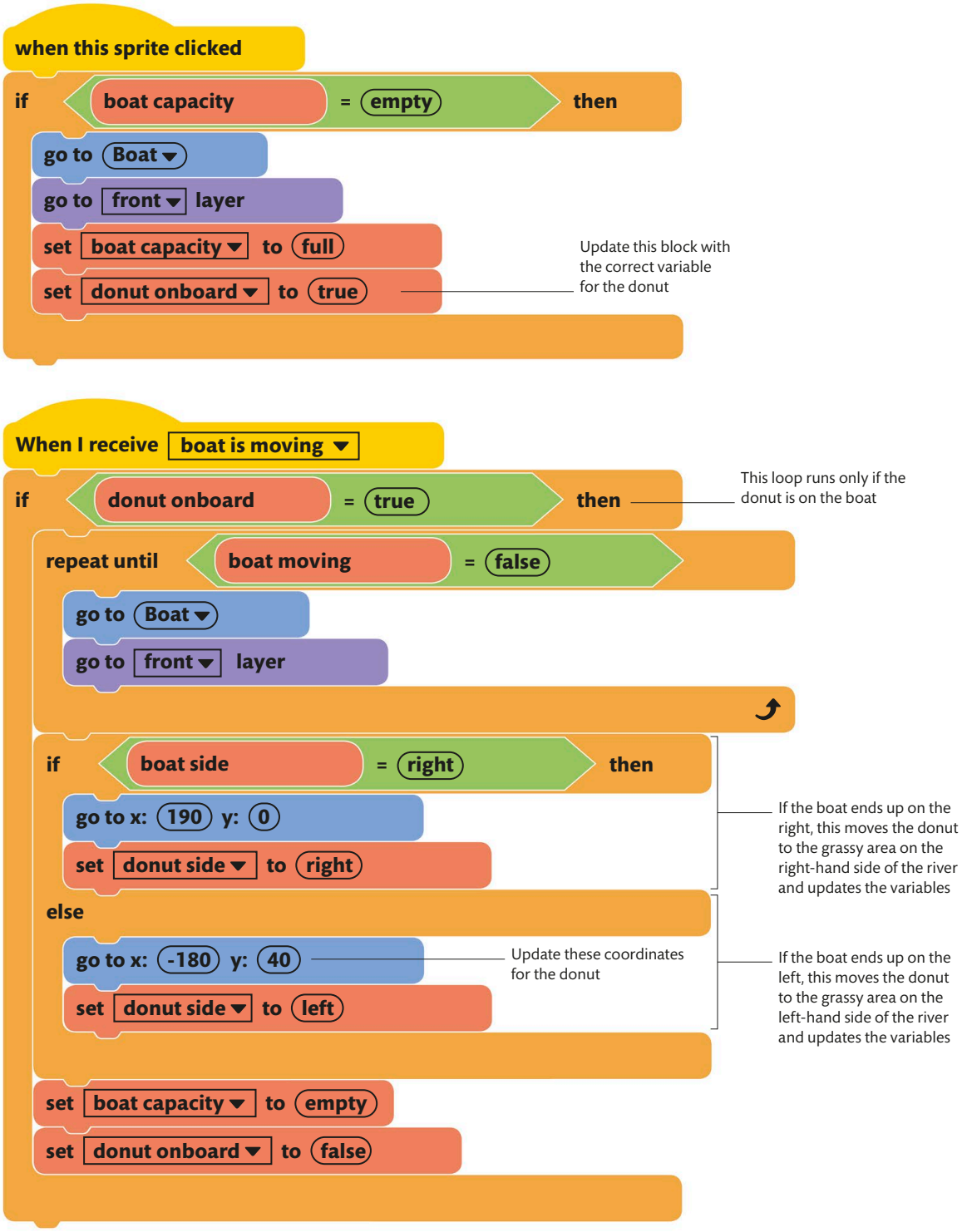
The code for the donut is very similar to the code for the lion. Luckily, Scratch makes it easy to reuse code. Click on the Lion sprite and find the blocks of code you made in steps 2.3, 2.4, and 2.5. Drag and drop all of those blocks onto the Donut sprite in the Sprite List. This will create a copy of all the blocks for the Donut sprite. The blocks may get copied on top of each other, but you can right click in the Code Area and select Clean up Blocks to set them in order.



3.3 UPDATE THE CODE

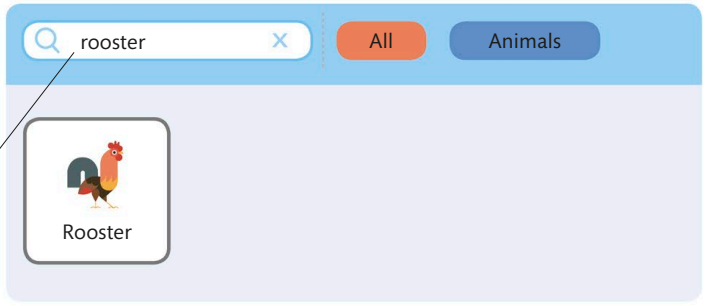
Now select the Donut sprite. You will see the blocks you just copied across. Update the code to make it work for the donut. First, create two new variables, **donut side** and **donut onboard**, then edit the code blocks to look like this. Make sure you uncheck the boxes next to the new variables.




when this sprite clicked
if boat capacity = empty then
go to Boat
go to front layer
set boat capacity to full
set donut onboard to true
Update this block with the correct variable for the donut
When I receive boat is moving
if donut onboard = true then
This loop runs only if the donut is on the boat
repeat until boat moving = false
go to Boat
go to front layer
if boat side = right then
go to x: 190 y: 0
set donut side to right
If the boat ends up on the right, this moves the donut to the grassy area on the right-hand side of the river and updates the variables
else
go to x: -180 y: 40
Update these coordinates for the donut
set donut side to left
If the boat ends up on the left, this moves the donut to the grassy area on the left-hand side of the river and updates the variables
set boat capacity to empty
set donut onboard to false

3.4 ADD THE ROOSTER

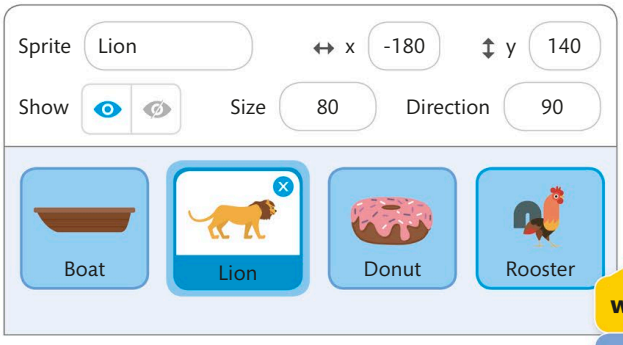
Now add the final sprite to the program. Go to the Sprite library and look for the "Rooster" sprite. Select it and make sure you reduce its size to 50 in the sprite information panel.



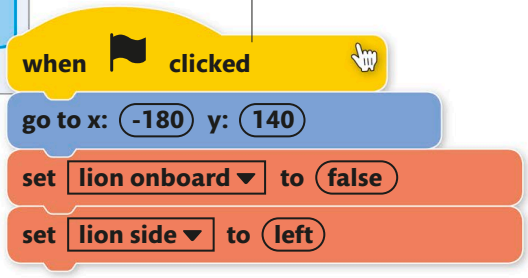
3.5 COPY CODE FROM THE LION

Next, add code to the rooster. Just like you did for the Donut, drag and drop all the code blocks from the Lion sprite onto the Rooster in the Sprite List.

This will make a copy of all the blocks. Right click anywhere in the Code Area and select "Clean up Blocks" to view the code blocks in an order.

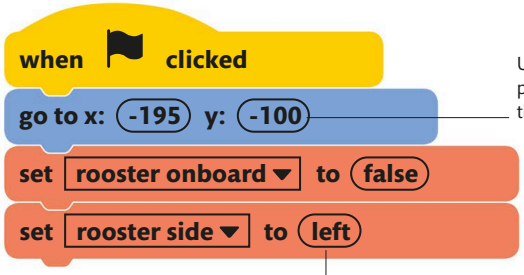


Drag and drop all the blocks of code from the Lion sprite, just like you did with the donut



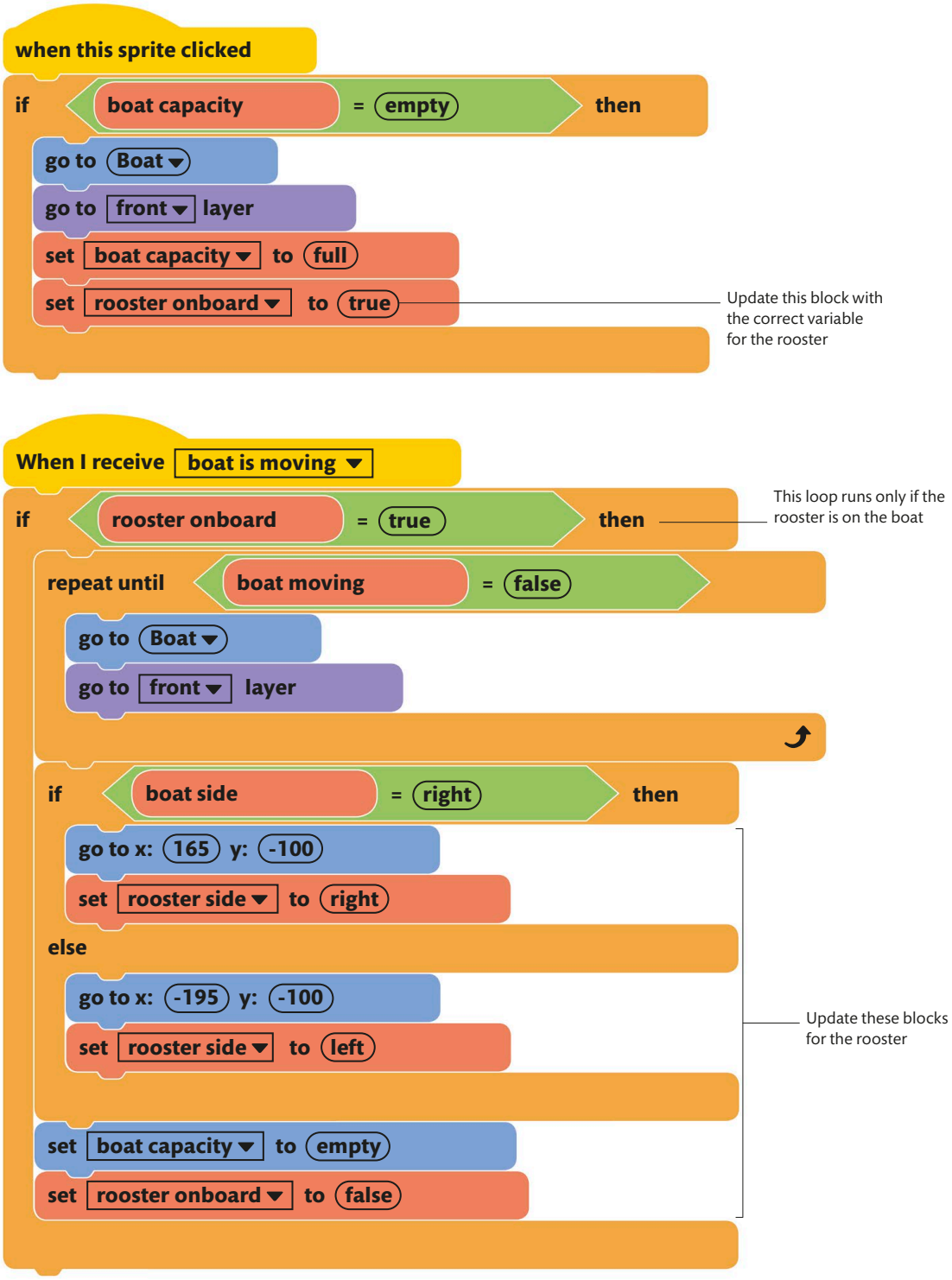
3.6 UPDATE THE CODE

Now you will need to edit the code you just copied. Remember to create two new variables—"rooster side" and "rooster onboard"—and make sure you uncheck the boxes next to the variables. The edited code for the Rooster sprite should look like this.



Update the coordinates to place the rooster below the donut on the Stage

This puts the rooster on the left-hand side of the river


when this sprite clicked
if boat capacity = empty then
go to Boat
go to front layer
set boat capacity to full
set rooster onboard to true
Update this block with the correct variable for the rooster
When I receive boat is moving
if rooster onboard = true then
This loop runs only if the rooster is on the boat
repeat until boat moving = false
go to Boat
go to front layer
if boat side = right then
go to x: 165 y: -100
set rooster side to right
else
go to x: -195 y: -100
set rooster side to left
Update these blocks for the rooster
set boat capacity to empty
set rooster onboard to false

3.7 ADD THE RULES

At this point, you should be able to move the objects back and forth across the river. Now it is time to introduce the rules of the brain teaser. Add these code blocks to the Boat sprite. They will check the sides to see if any of the rules are being broken or if the user has successfully solved the brain teaser.

Click on the drop-down menu and choose "New message" to create this message

```
when I receive [check sides]
if <lion side = right> and <donut side = right> and <rooster side>
    say [Win: Everything is on the right side of the river]
if <lion side = left> and <donut side = right> and <rooster side>
    say [Lose: The lion eats the rooster]
if <lion side = right> and <donut side = left> and <rooster side>
    say [Lose: The lion eats the rooster]
if <lion side = right> and <donut side = left> and <rooster side>
    say [Lose: The rooster eats the donut]
if <lion side = left> and <donut side = right> and <rooster side>
    say [Lose: The rooster eats the donut]
```

If the lion and rooster are left alone on the left-hand side, then it is game over

If the lion and rooster are left alone on the right-hand side, then it is game over

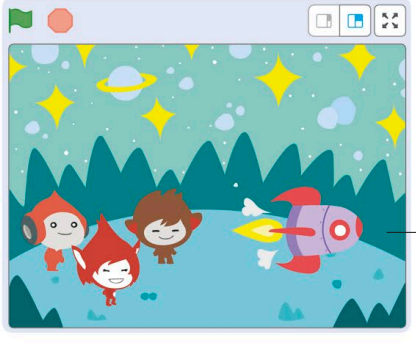
If the rooster and donut are left alone on the left-hand side, it will be against the rules

If the rooster has been left alone with the donut on the right-hand side, it will be against the rules

Hacks and tweaks

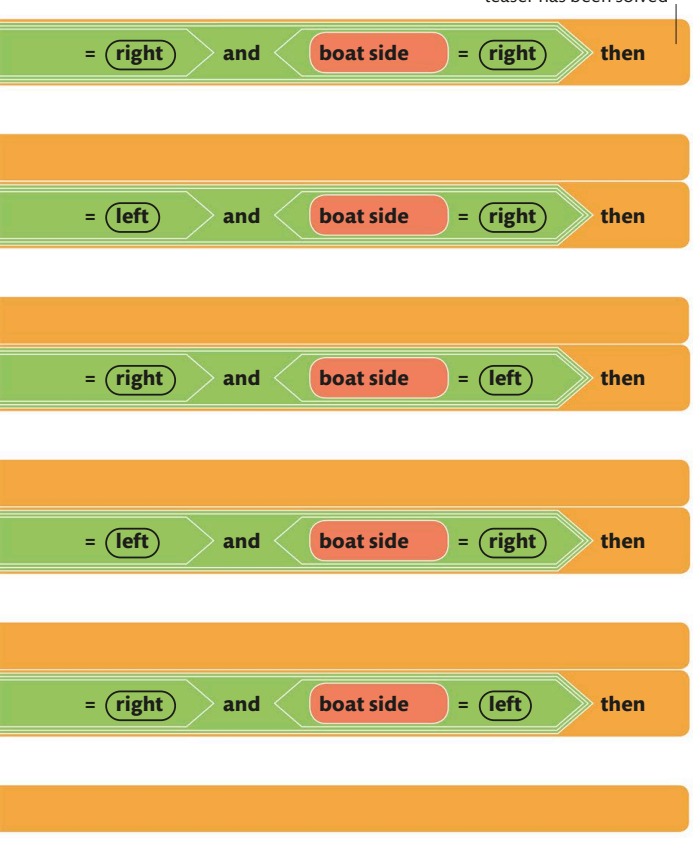
Design your own scenario

Use the Paint Editor to change the sprites' costumes and the project's backdrop to create a whole new scenario for this brain teaser. Maybe it could be set in space, and you need to transport aliens from one space station to another, but you cannot leave certain types of aliens together. Use the Sprite and Backdrop library to come up with more ideas.



Experiment with other sprites and backdrops to create different scenarios

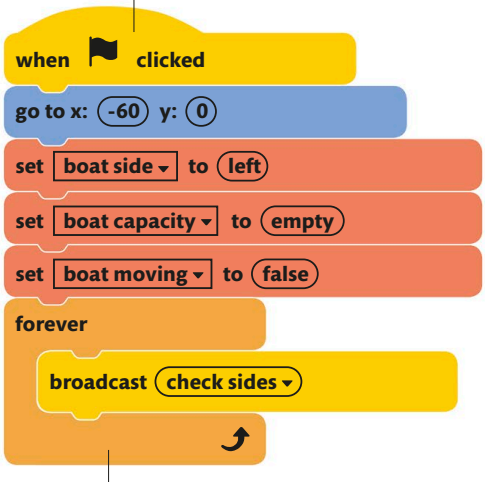
If every sprite is on the right-hand side of the river, then the brain teaser has been solved



3.8 ENFORCE THE RULES

Finally, these rules need to be checked at all times. Just update the blocks of code you added in step 1.7 with a few new blocks, then run the code and test the logic.

Find these blocks of code in the Boat sprite



The forever loop will constantly check the sides to see if any of the rules have been broken

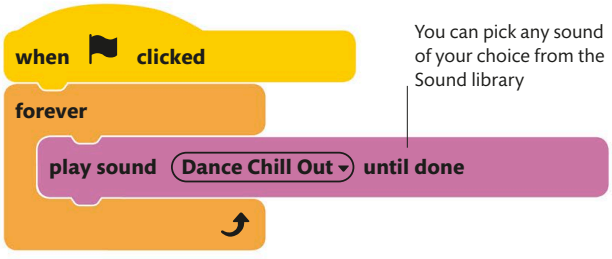
Count the moves

Can you add a variable that counts how many "moves" the player has made so far? You will need to add a new variable called **moves** and set it to increase by one every time someone clicks on the boat.

Add this block to the code created in step 1.8 to increase the number of moves

Background music

Many brain teasers and puzzle games have simple background music to help the player focus. To add music, select the Backdrops icon at the bottom right of the screen, then click the Sounds tab. Go to the Choose a sound icon at the bottom left and look for "Dance Chill Out", then add this code to make the sound play forever.



You can pick any sound of your choice from the Sound library

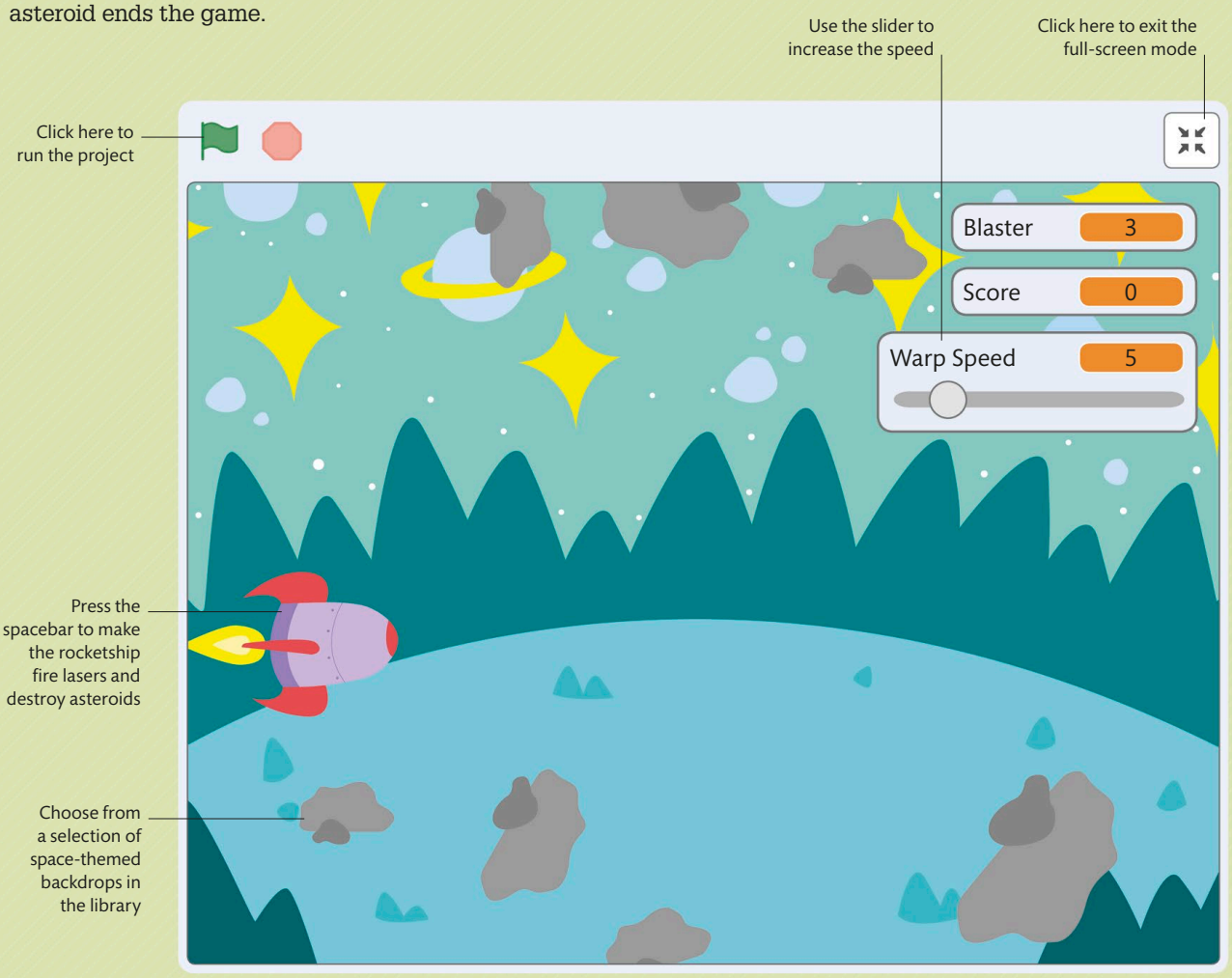
Asteroid dodge

In this Scratch project, you will create a side-scrolling game with animated sprites. This is a great way to get started with game development. The finished game will test your concentration and the speed of your reflexes.

How the game works

The game lets a player use the up and down arrow keys to navigate a rocketship around asteroids. The "Warp Speed" slider controls the speed of the game, and the rate at which asteroids appear increases as the game progresses. Any contact between the rocketship and an asteroid ends the game.

Moving obstacles
The project creates an illusion of motion by moving the obstacles along the x-axis and making them appear at random intervals.



YOU WILL LEARN

- How to create a side-scrolling game
- How to use loops to create continuous game play
- How to create a game that increases in difficulty as it progresses

Time: 20–25 mins

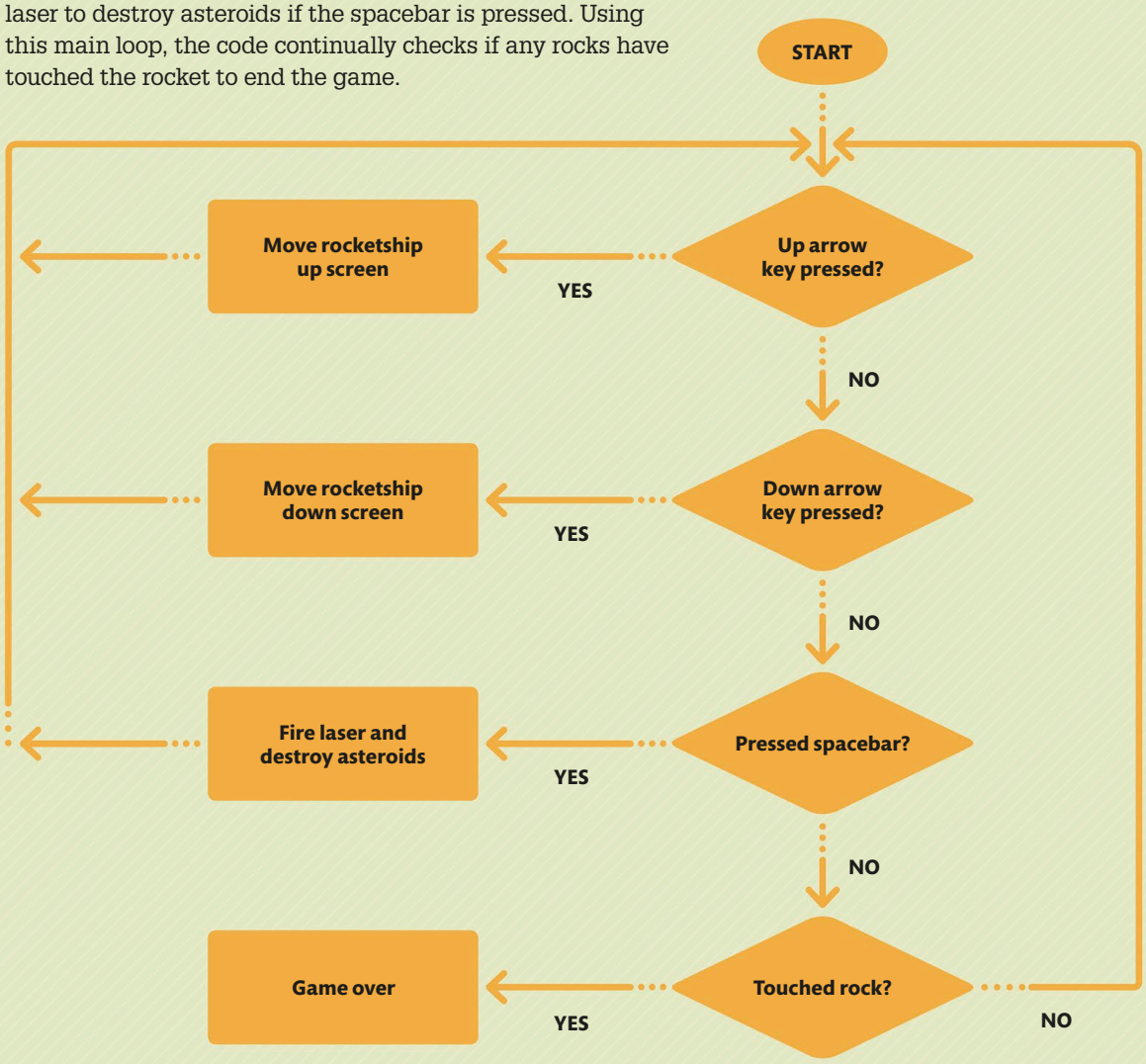
Difficulty level

WHERE THIS IS USED

In this game, the background and other objects move across the screen to make it seem like the player is moving. This popular approach is called a side-scrolling game and can be adapted for racing or shooting games.

Program design

The program uses one main loop to check which key is being pressed to move the rocket up or down the screen. It fires a laser to destroy asteroids if the spacebar is pressed. Using this main loop, the code continually checks if any rocks have touched the rocket to end the game.

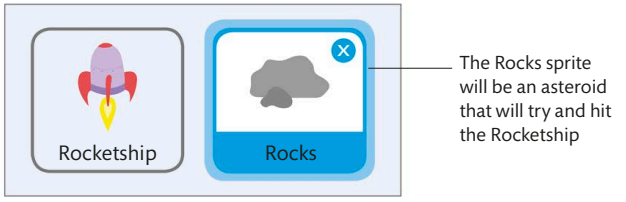


1 Prepare for launch

This project requires a few basic elements to get started. The sprites and backdrop will create the space setting for the game, and variables will add functionality.

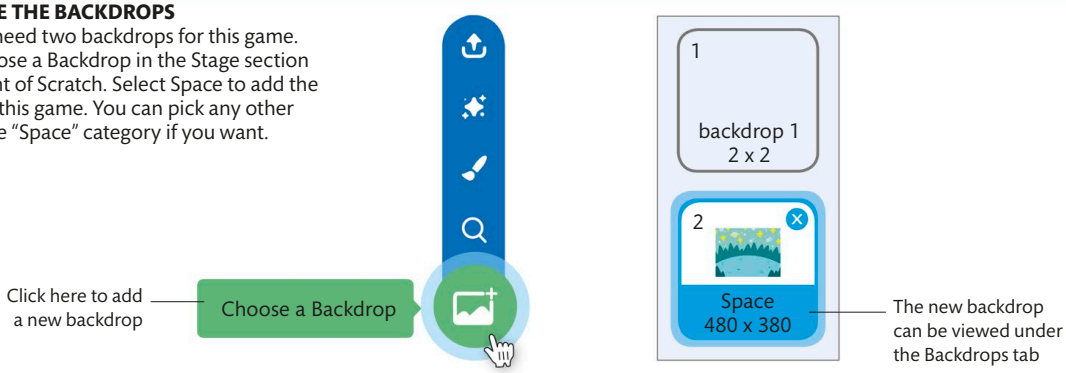
1.1 ADD SPRITES

Start a new project and delete the default Cat sprite. Add the two new sprites required for this game: Rocketship and Rocks. You can find them in the Sprite library.



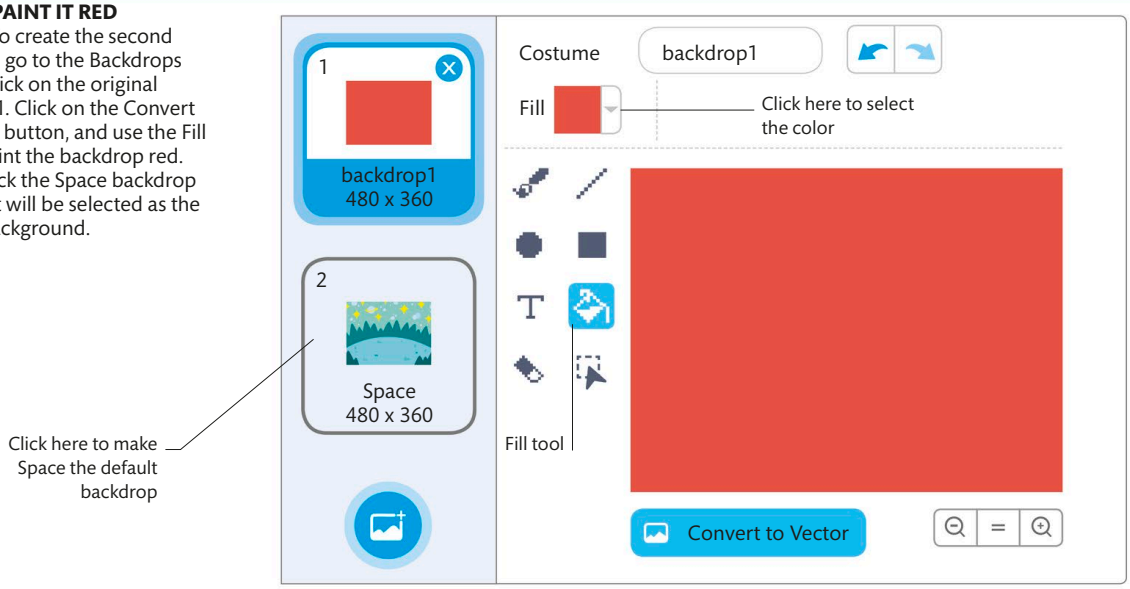
1.2 PREPARE THE BACKDROPS

You will need two backdrops for this game. First, click on Choose a Backdrop in the Stage section at the bottom right of Scratch. Select Space to add the first backdrop for this game. You can pick any other backdrop from the "Space" category if you want.



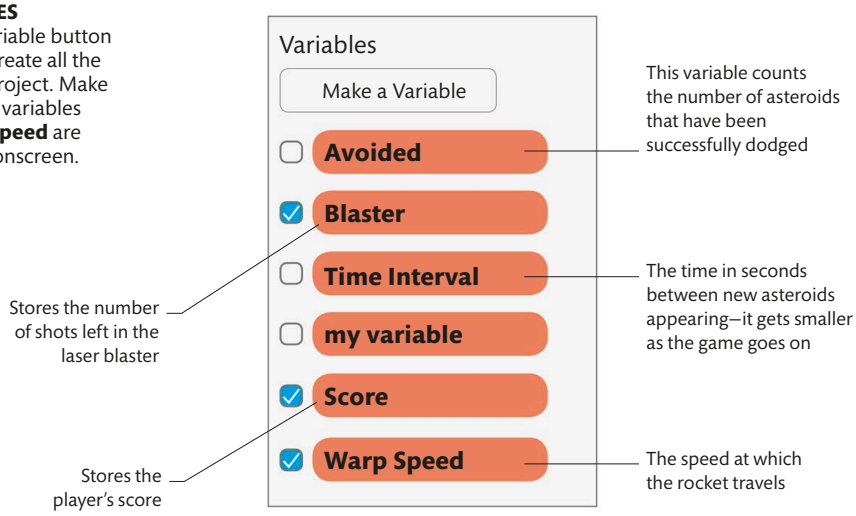
1.3 PAINT IT RED

To create the second backdrop, go to the Backdrops tab and click on the original backdrop1. Click on the Convert to Bitmap button, and use the Fill tool to paint the backdrop red. Finally, click the Space backdrop again so it will be selected as the default background.



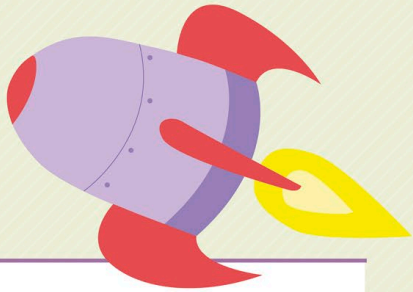
1.4 CREATE VARIABLES

Use the Make a Variable button in the Variables section to create all the variables required for this project. Make sure the checkboxes for the variables **Blaster**, **Score**, and **Warp Speed** are checked so that they show onscreen.



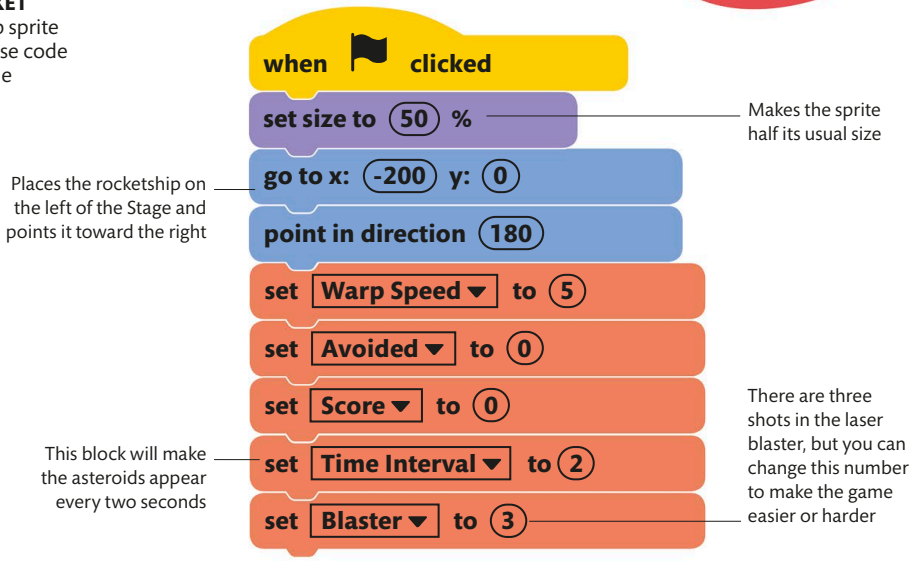
2 Code the rocketship

Now that the basic elements required for the project are ready, you can begin coding. Start by adding code for the Rocketship sprite. The next few steps will add steering controls for the rocket, make it move through space, and add a blaster gun for firing the laser.



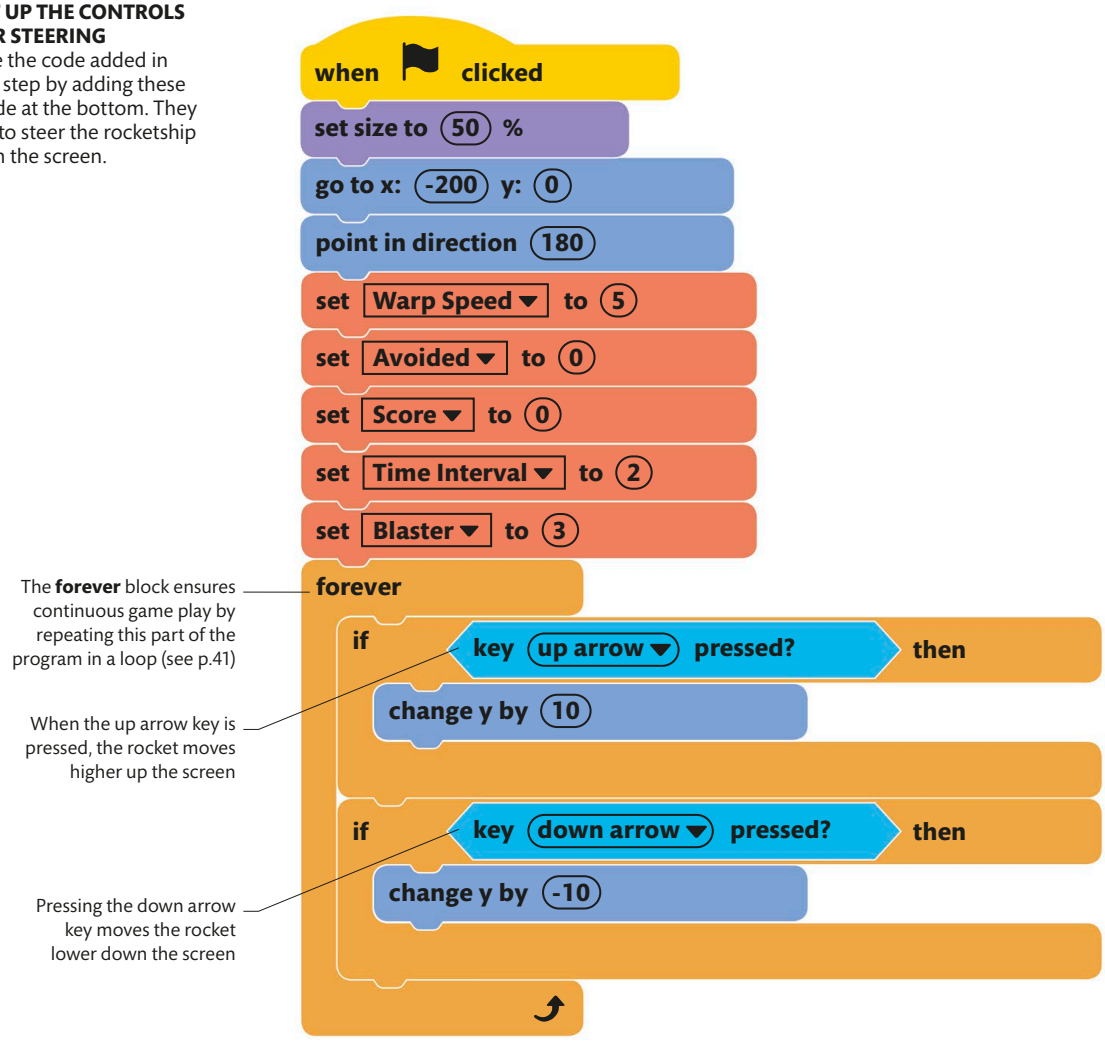
2.1 PREPARE THE ROCKET

Select the Rocketship sprite in the Sprite List, then add these code blocks for it. This will set up the rocketship for the game.



2.2 SET UP THE CONTROLS FOR STEERING

Next, update the code added in the previous step by adding these blocks of code at the bottom. They will be used to steer the rocketship up and down the screen.

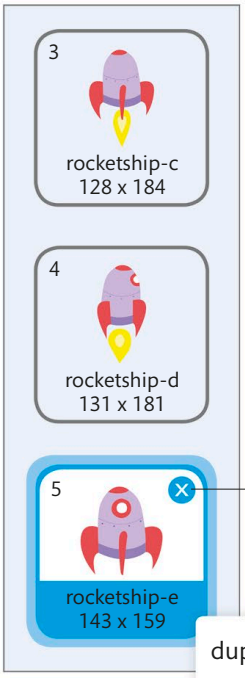


2.3 TEST FLIGHT

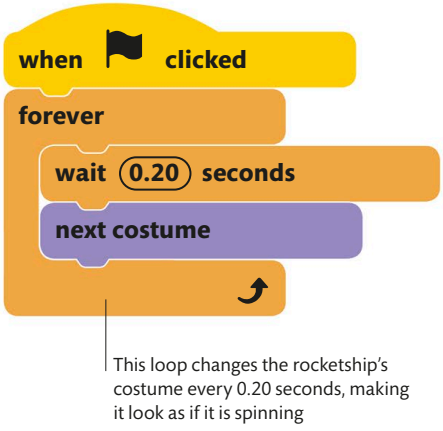
Click on the green flag above the Stage to run the code and test the rocket. See if you can make the rocketship move up and down the screen using the arrow keys.

2.4 ANIMATE THE ROCKET

Without the flame, it may look like the rocket has stopped. To avoid this, click on the Costumes tab and then right-click on costume number five and choose delete. Next, add the code given below to make the rocket appear as if it is soaring through space. Test the code before going on to the next step.



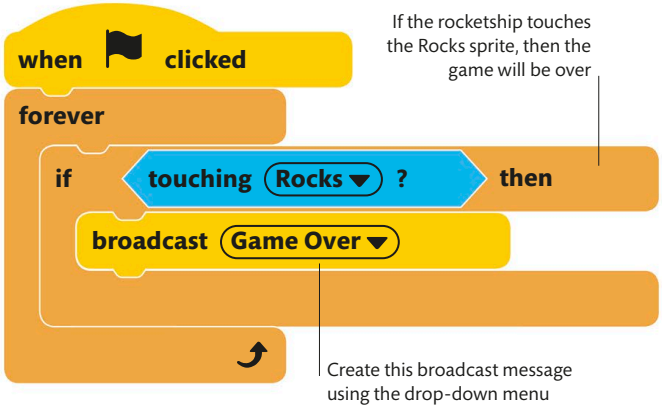
You can also delete a costume by clicking on the small blue cross



This loop changes the rocketship's costume every 0.20 seconds, making it look as if it is spinning

2.5 AVOID THE ASTEROIDS

The next step is to add in some logic. Add these blocks to the Rocketship sprite. This will tell the rocketship what to do if it touches the asteroids that will be added later.

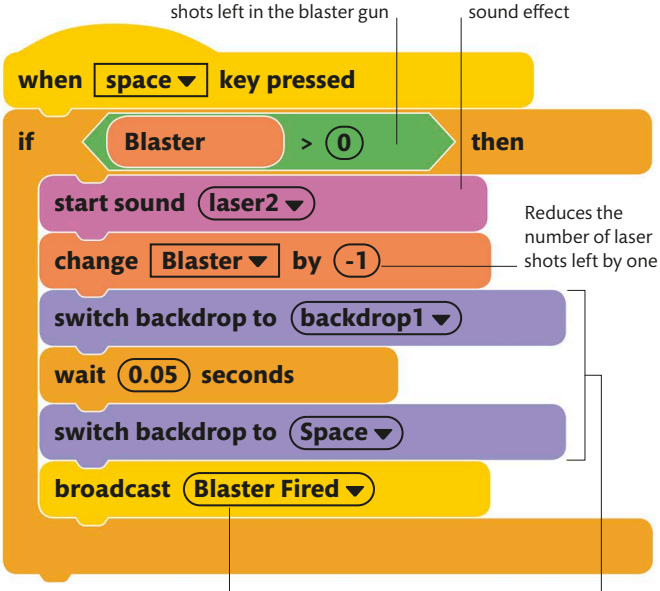


If the rocketship touches the Rocks sprite, then the game will be over

Create this broadcast message using the drop-down menu

2.6 PREPARE THE BLASTER GUN

Now you will write the code that controls the blaster gun. When the space key is pressed, if there are shots left in the blaster, then the laser fires. To achieve this effect, the background quickly flashes to the red backdrop, which makes it look like the laser has been shot. Add this code to the Rocketship sprite and test the code.



Checks if there are shots left in the blaster gun

Plays a laser blaster sound effect

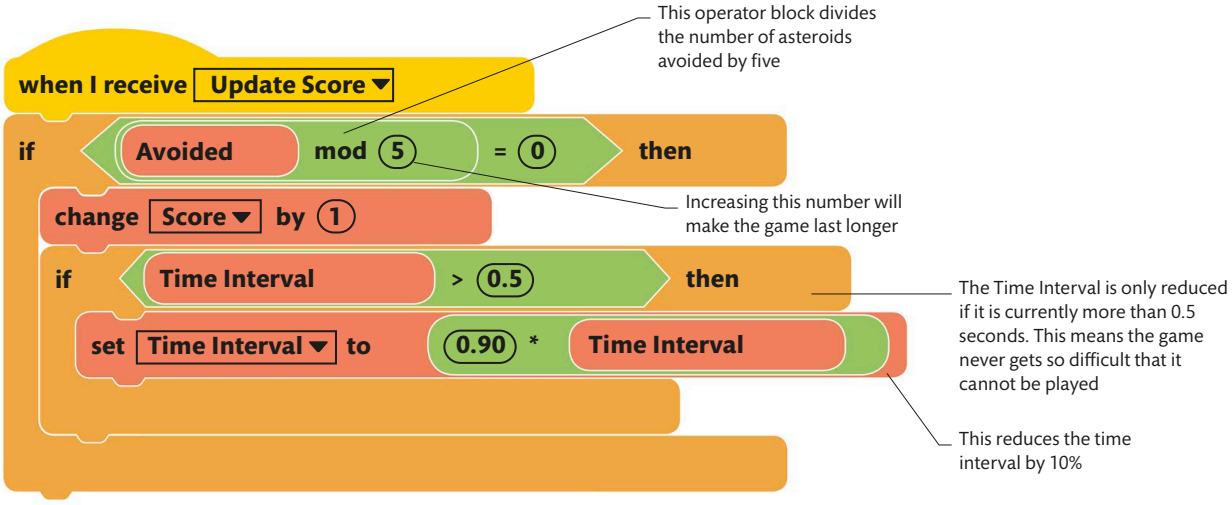
Reduces the number of laser shots left by one

Create a new broadcast message called Blaster Fired

Quickly changes the background to red, then back to Space, making it look like a shot has been fired

2.7 UPDATE THE SCORE

Now add these blocks of code. This will increase the score by one each time five asteroids are avoided. It reduces the Time Interval—this means new asteroids will appear more often.

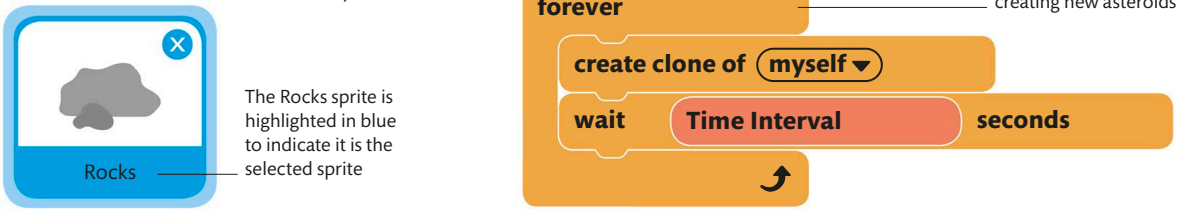


3 Code the asteroids

Once the rocketship is prepared, you need to program the asteroids. The code in the next few steps will make a stream of asteroids move across the screen, making them fly across space. An explosion will also be added to indicate the asteroids being hit by the blaster gun.

3.1 CREATE THE ASTEROIDS

To write the code for the asteroids, click on the Rocks sprite and add this code. The forever loop means that once the game is started, asteroids keep getting created.



3.2 MAKE THE ASTEROIDS MOVE

To create the illusion of the rocketship moving, the rocks will move across the screen. Once they reach the left side, they vanish. The random blocks are used so that each asteroid starts at different positions at the right side of the screen. That way, the player cannot guess where the next one will appear, making the game more challenging. Add this code to the Rocks sprite to make this happen.

```
when I start as a clone
set size to [pick random (20) to (50)] %
forever
    show
    go to x: (200) y: [pick random (-240) to (240)]
    forever
        turn (15) degrees
        change x by [(-1) * (Warp Speed)]
        if [(x position) < (-240)] then
            change [Avoided] by (1)
            broadcast [Update Score]
            delete this clone
```

The size of the asteroid is chosen randomly

Makes the clone visible on the Stage while the original Rocks sprite stays hidden

Places the asteroid at the right side of the screen at a random position

Makes the asteroids spin as they move through space

If the asteroid reaches the left side of the screen, increases the number of avoided asteroids by one

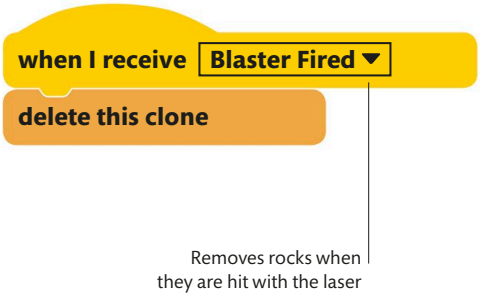
Removes the asteroid

The rocks will appear on the right edge of the screen at a random y position and move toward the left

Rocks will be of random sizes

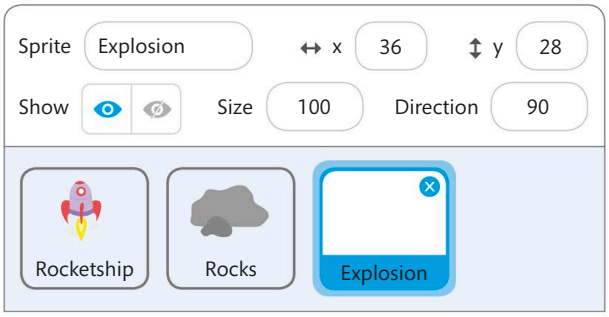
3.3 REMOVE ASTEROIDS

Now it is time to add some code to fire the blaster gun. When this program is run, it will destroy the asteroids when the blaster is fired. Add these blocks to the Rocks sprite and then test it out. Remember that you only have three shots.



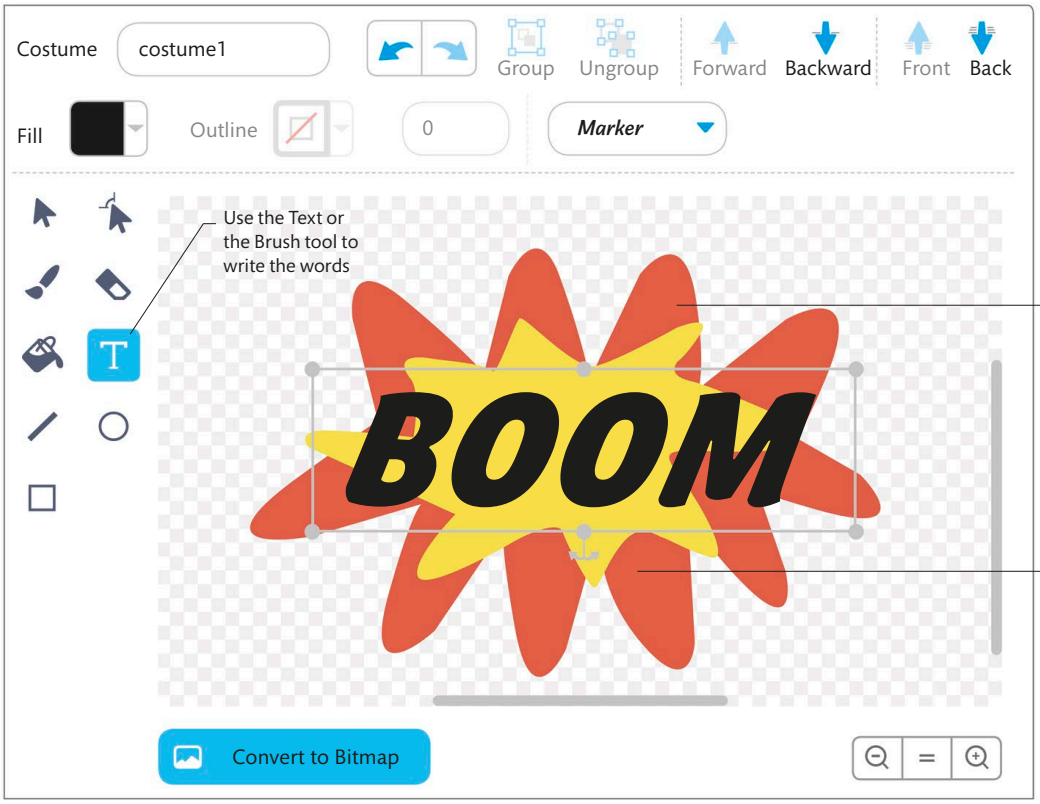
3.4 CREATE AN EXPLOSION SPRITE

Next, add one more sprite to create an explosion when the asteroids hit the rocketship. Choose "Paint" from the Choose a Sprite menu at the bottom right of the Sprite List and name the sprite "Explosion".



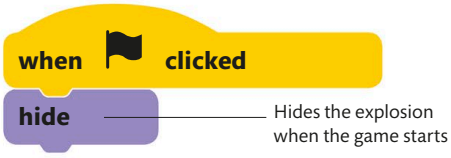
3.5 PAINT THE EXPLOSION

Use the Paint Editor to draw a large fireball effect. You can use the Brush, Fill, and Text tools to create a large, colorful explosion.



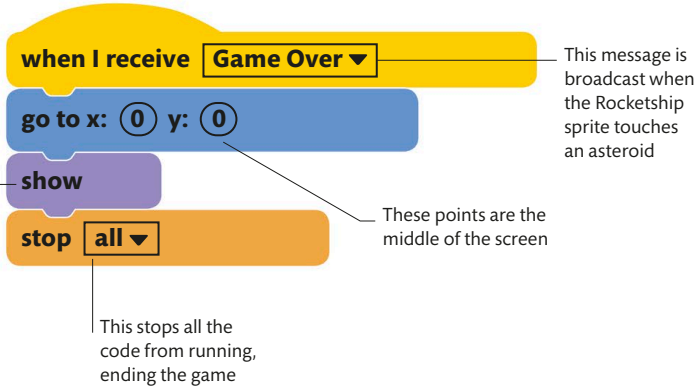
3.6 HIDE EXPLOSION

When the game starts, you do not want the explosion to be visible on the screen. Add this code to the Explosion sprite to hide it.



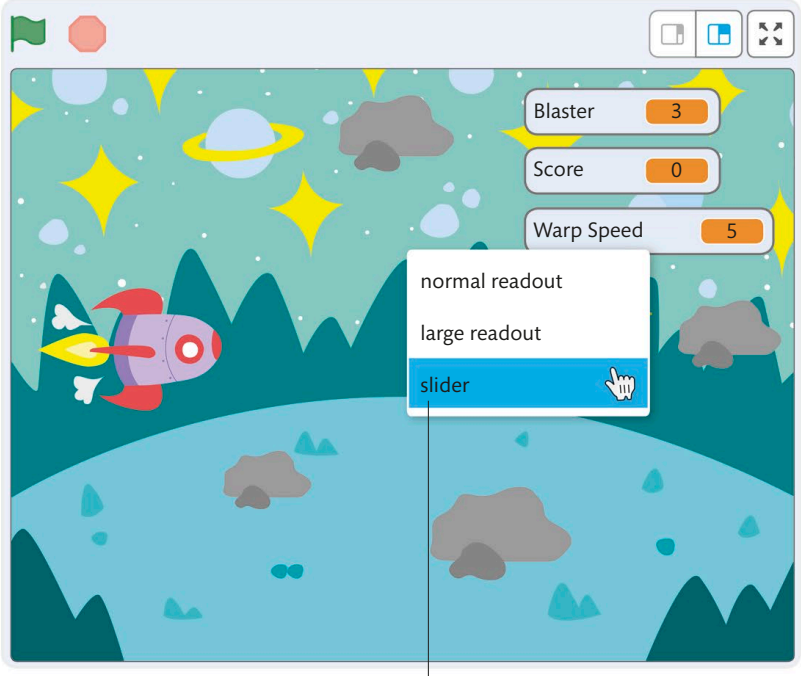
3.7 GAME OVER

Next, add these blocks of code to the Explosion sprite. This will make the explosion appear in the middle of the screen when the Game Over message is broadcast and then stops the game.



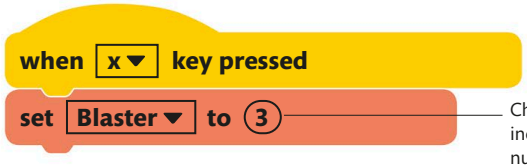
3.8 WARP SPEED SLIDER

One final touch is to right-click on the **Warp Speed** variable at the top right of the screen and choose "slider". This means that the player can now adjust the speed of the game by moving the slider left and right. The game is now ready to play. See how far you can guide the rocketship, and do not forget to use the blaster when you need to. You can even adjust the Warp Speed slider and see how fast you can go.



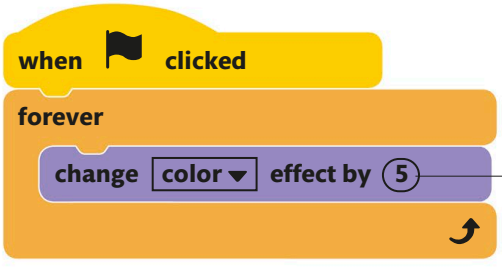
Hacks and tweaks

Add a cheat code to refill your blaster gun
Adding your own cheat codes is a fun way to personalize a project. Add this code to the Rocketship sprite so that when you press the x key, the blaster is refilled with three more shots. You can also try to create a sprite that appears every 20 asteroids and increases the number of blaster shots by one if it touches the spaceship.



Change this number to increase or decrease the number of shots to be refilled

Deep space spectrum
Add this code to the backdrop and it will make the background cycle through a spectrum of colors, creating a fantastic intergalactic lightshow.



Increasing this number will make the colors flash quickly

Run the code to see the changing colors of the backdrop

Unidentified flying objects
You can easily add other objects for the rocketship to dodge. Just add a new costume to the Rocks sprite and then amend the code as shown here. This will make dogs fly through space.