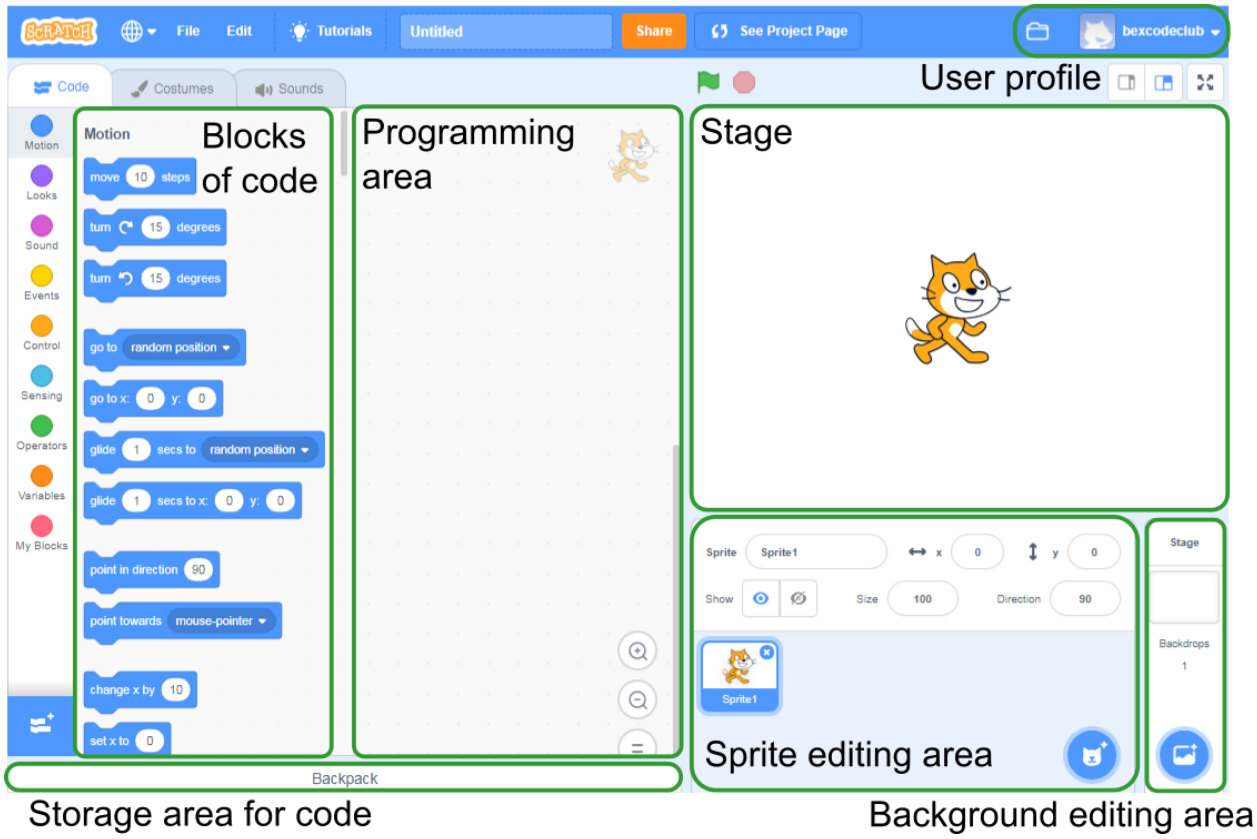
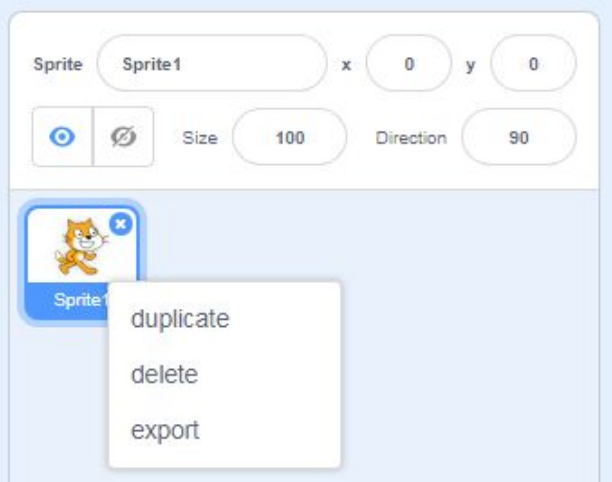
Te Reo Māori Chatbot

SESSION 1 – STARTING OUT

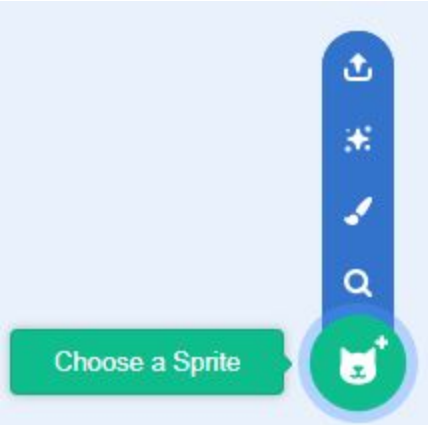
Head to scratch.mit.edu and create a new project. The cat is called a 'sprite.' This is a little character that we will will be able to control with our code. The coding blocks are on the left and they can be dragged and dropped into the central programming area.



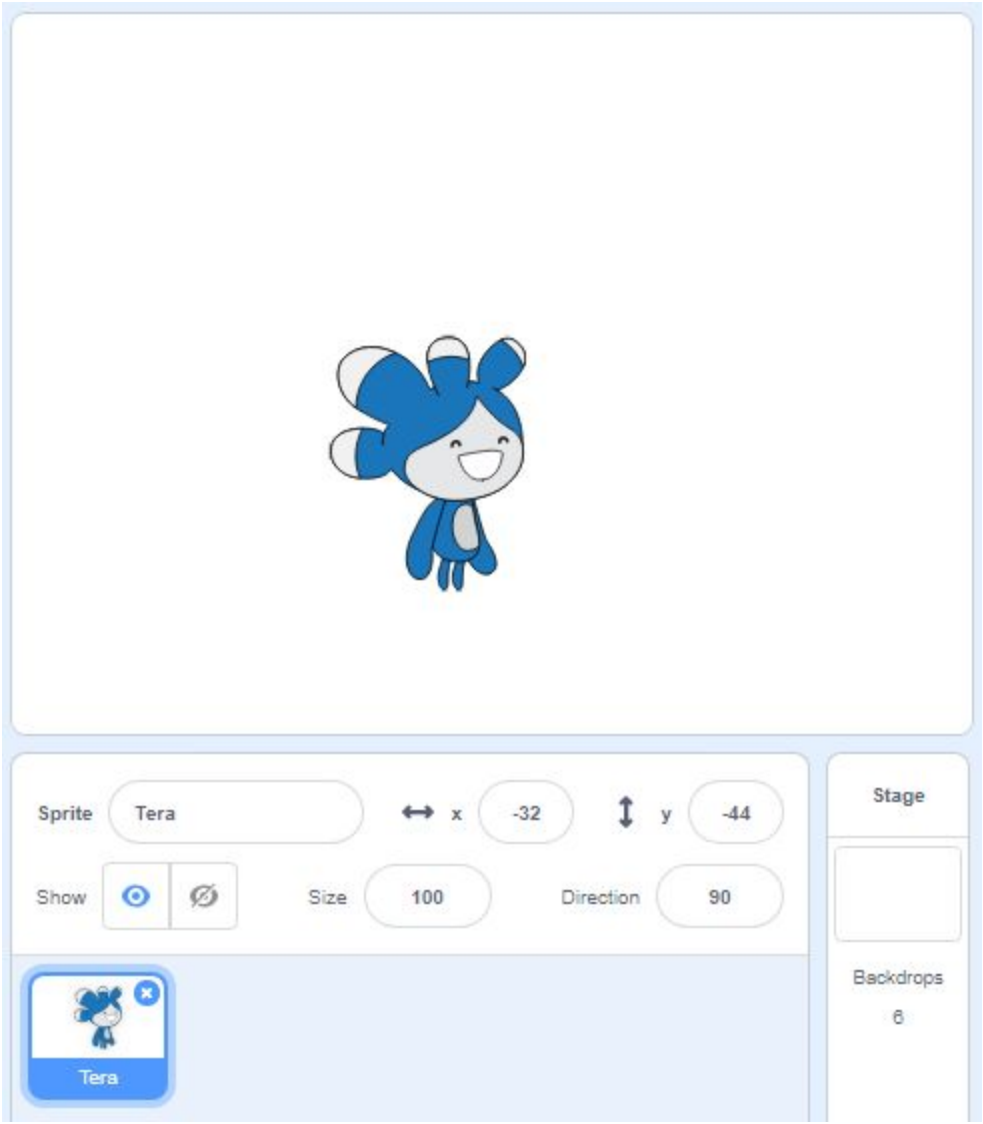
The first thing we want to do is change the sprite. We do this by right clicking and selecting delete.



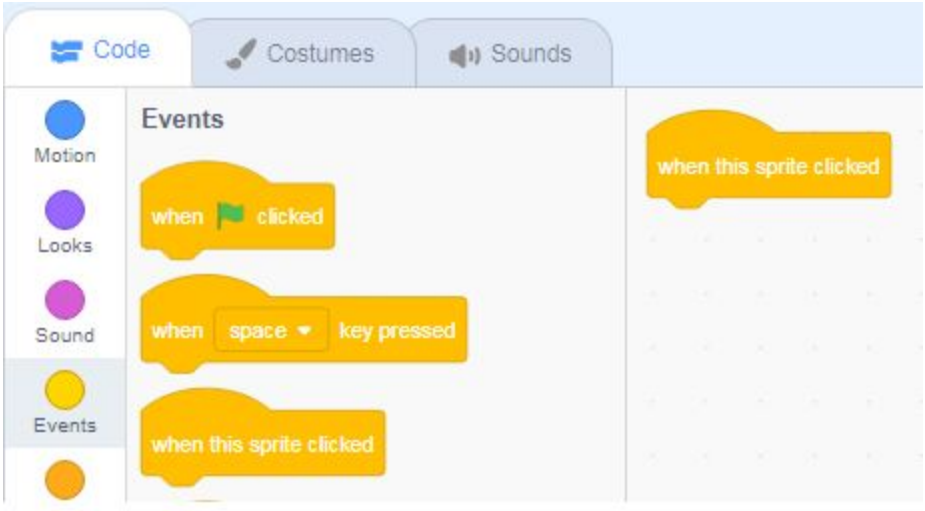
To add a new sprite, we click the sprite icon to open the sprite library.



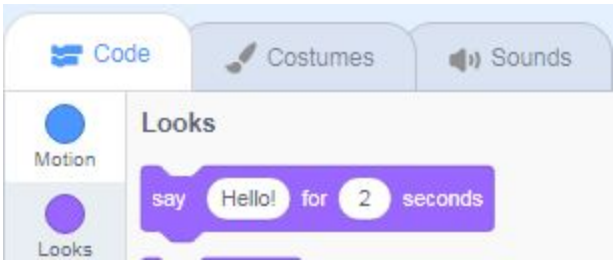
We can filter the sprite types using the categories along the top. Click on the sprite you want to use, and click OK.



We're going to write a program that makes the sprite talk to us. We need to use a block to signify when our program will start. These can be found in **Events**. From **Events**, find the 'when this sprite clicked' block. Drag and drop the block into the grey programming area.



Now click on **Looks**, and find the 'say ____ for __ secs' block.



Drag and drop the block under the 'when this sprite clicked' block. A grey shadow will appear under the 'when this sprite clicked' block when the 'say ____ for __ secs' block is close enough to snap together. Double click on the text and change it to say "Let's count to five!"

Congratulations! You have now written your first program! To test it, click on the sprite. A speech bubble should appear above your sprite.

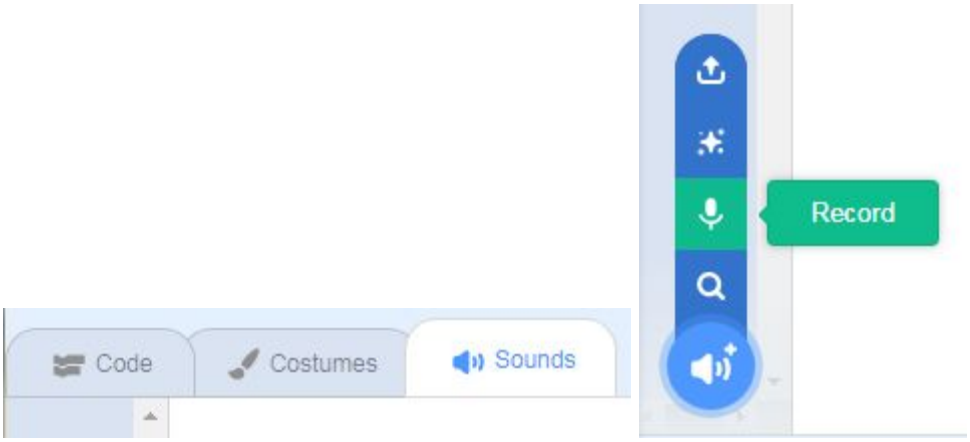
Not working? If you didn't see a speech bubble, you will need to look for the problem. This is called debugging. Have the two blocks snapped together like two lego blocks? If not, try moving the say block closer to the event block so that they snap together.

Sequencing

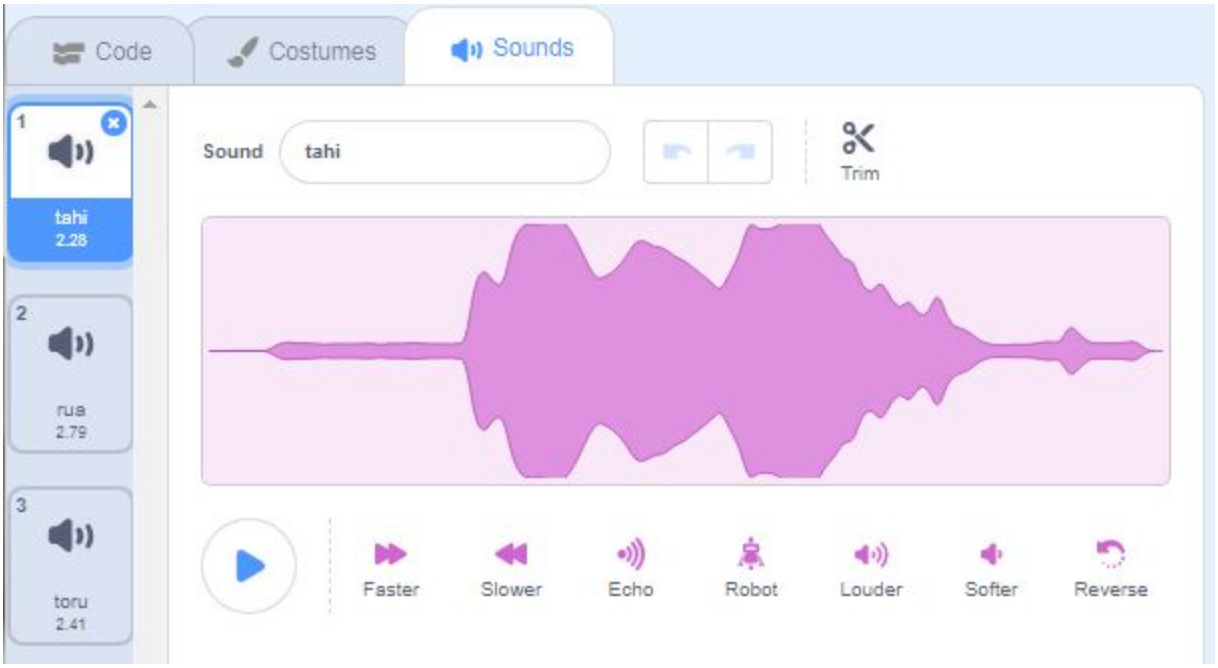
We can add more detail to our program by adding more blocks. Drag five more 'say ____ for __ secs' block under your 'When this sprite clicked' block. Change the text to the numbers one to five in Te Reo Māori, and the number of seconds that they are displayed for.

The computer doesn't know which order the numbers should go in. Instead it reads the order that you have put them in. Try rearranging the blocks. Does that change the order that the sprite counts in?

We can also use Scratch to record and play sounds. Head into the Sounds tab and click the microphone icon.



Rename the sound to tahi, and click on the round record button to record yourself saying tahi.



Now click on **Sounds** in the Scripts tab. Drag the 'start sound ____' block out and add it to your code above the 'say tahi for __ secs' block. Use the small drop-down arrow to select your sound, tahi.

To delete blocks of code in Scratch, you can either right click on the block you want to remove and click delete, or drag and drop the blocks back into the centre code bar.

Now when you click your sprite, it should say tahi before it has displayed the tahi speech bubble. Record and add the other four numbers to the program.

Extra: Sprites can have different costumes that changes their appearance. Check out your Sprite's costumes in the costume tab. Using blocks from Looks, can you make it look like your Sprite it talking? Hint: you might need to also use a wait block from Control!

Session 2 – Adding Detail

We are now going to start interacting with our sprite by getting it to ask us questions.

Questions and Answers

In Scratch, it is possible to ask the user something and to record their answer. The sprite will ask a question, and a text box will appear at the bottom of the display area where the user can type in an answer. This is then stored as 'answer' and can be used to direct the program.

From **Sensing** find the 'ask ____' block. Add it to your program so that it is the first block after the 'when this sprite clicked block'. Change the question to ask "Would you like to know how to count to five in Te Reo Māori?"

For the computer to be able to respond to the user's answer, we will need to use a conditional statement.

Conditional Statements

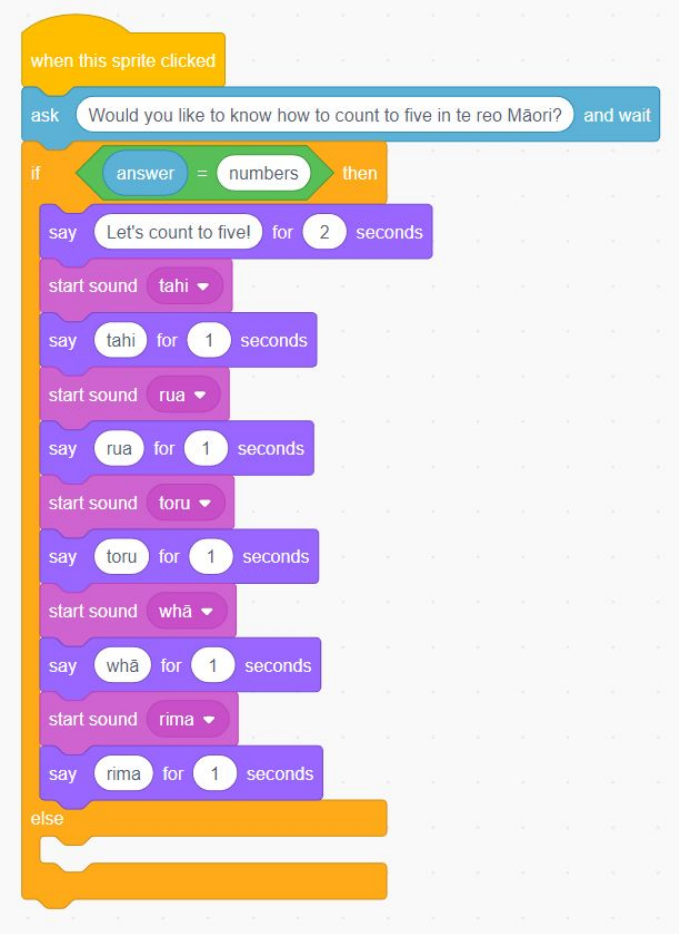
A conditional statement is a statement that a computer uses to help it make a decision.

We use conditional statements all the time, for example "if you pick up all your toys, then you can watch tv" or "if you eat your broccoli, then you can have an ice cream or else you'll have to have an apple".

We are going to create some code that causes something to happen depending on the answer given.

From **Control**, find the 'if ___ then else' block. Drag it into the programming area so that the code below the 'ask ____' block is inside the 'if' section. The block will automatically expand to contain your existing code.

We want the program to say the numbers if the user answered 'yes' to the question. Head into **Operations**, find the '__ = __' block and slot it into the space in the 'if ____ then else' block. From **Sensing**, find the 'answer' block. Drag it into the first space in the '__ = __' block and type "yes" into the second space.

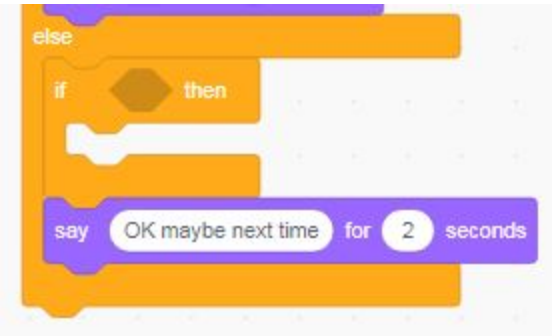


We now need to add in something that will happen if the user doesn't reply "yes". From **Looks**, find the 'say ____ for __ secs' block. Drag it into the 'else' section of the 'if ____ then else' block. Type "OK maybe next time" in the space.

Nested Statements

We can add a second answer option to program, but to do this, we are going to have to place a conditional statement inside another conditional statement. This is called a nested statement.

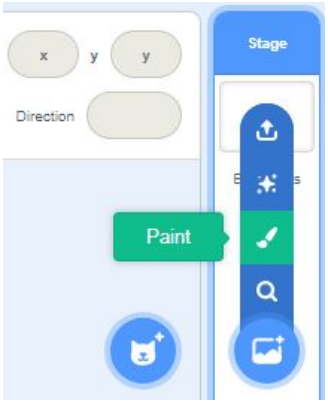
Drag another 'if ___ then' block from **Control**, and place it inside the 'else' section of your first 'if ____ then else' block. You may need to remove and replace your existing blocks to get them all in the right order.



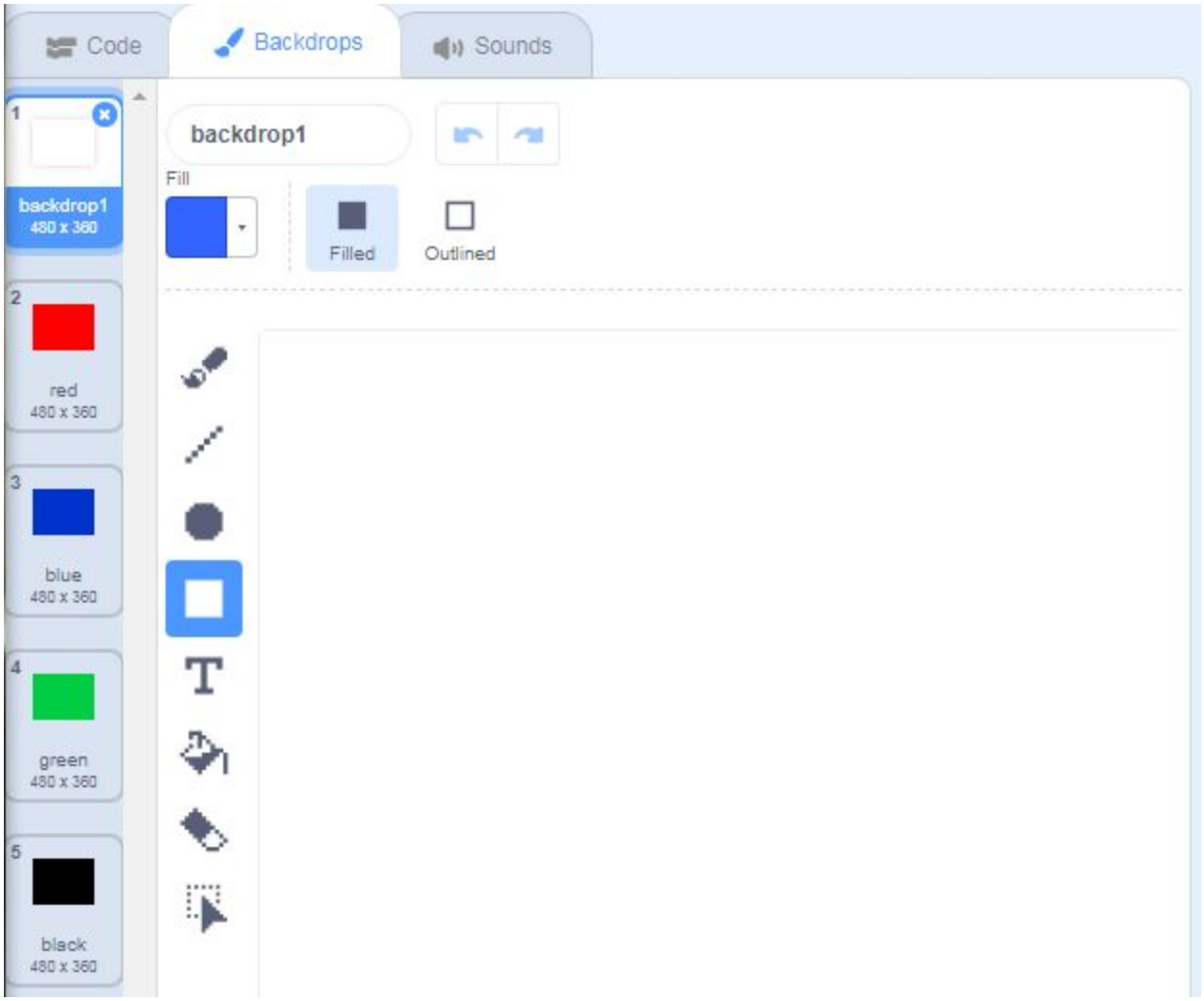
Change your question to ask if the user wants to learn about colours or numbers.

Change the condition in the first if statement so that it reads 'answer = numbers'. Create a condition in the second if statement that reads 'answer = colours'. From **Looks**, find the 'say ____ for __ secs' block and drag five blocks into your 'if ____ then' block. Add the names of the colours so that the sprite says the colours.

It is possible to change the colour of the stage. Hover over the new backdrop icon to bring up the menu.

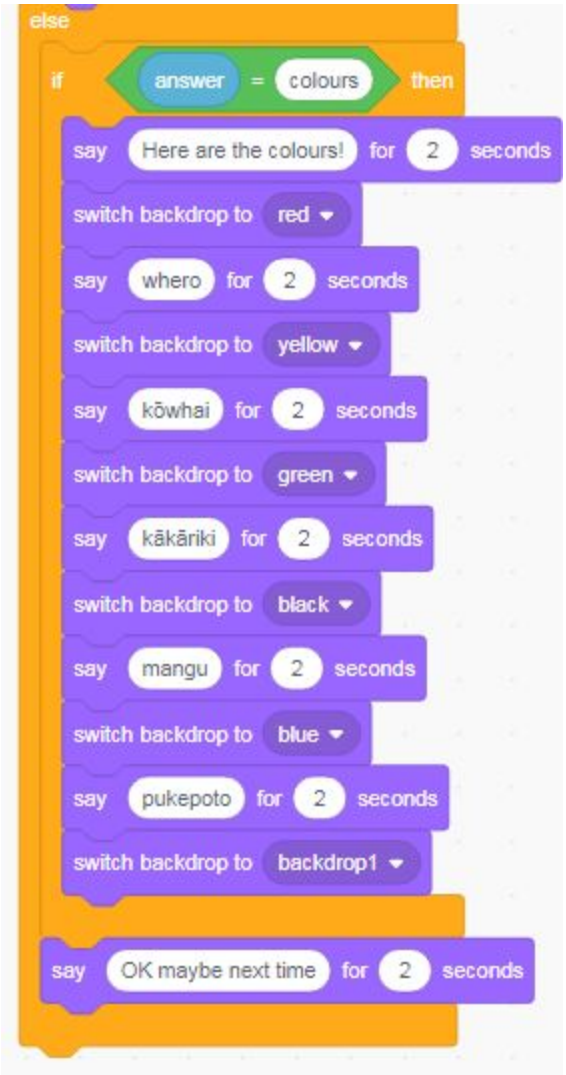


Click on the paintbrush icon to draw a new background. Use the fill colour to create coloured backgrounds. Make sure you rename each background to reflect its colour.



Click back on your sprite. From **Looks**, find the 'switch backdrop to ____' block. Add a block in between each of your 'say ____ for __ secs' block so that the colour of the background changes as the sprite says the name of the colour.

> Not working? Remember that the order of blocks is important as the computer can only process one instruction at a time.

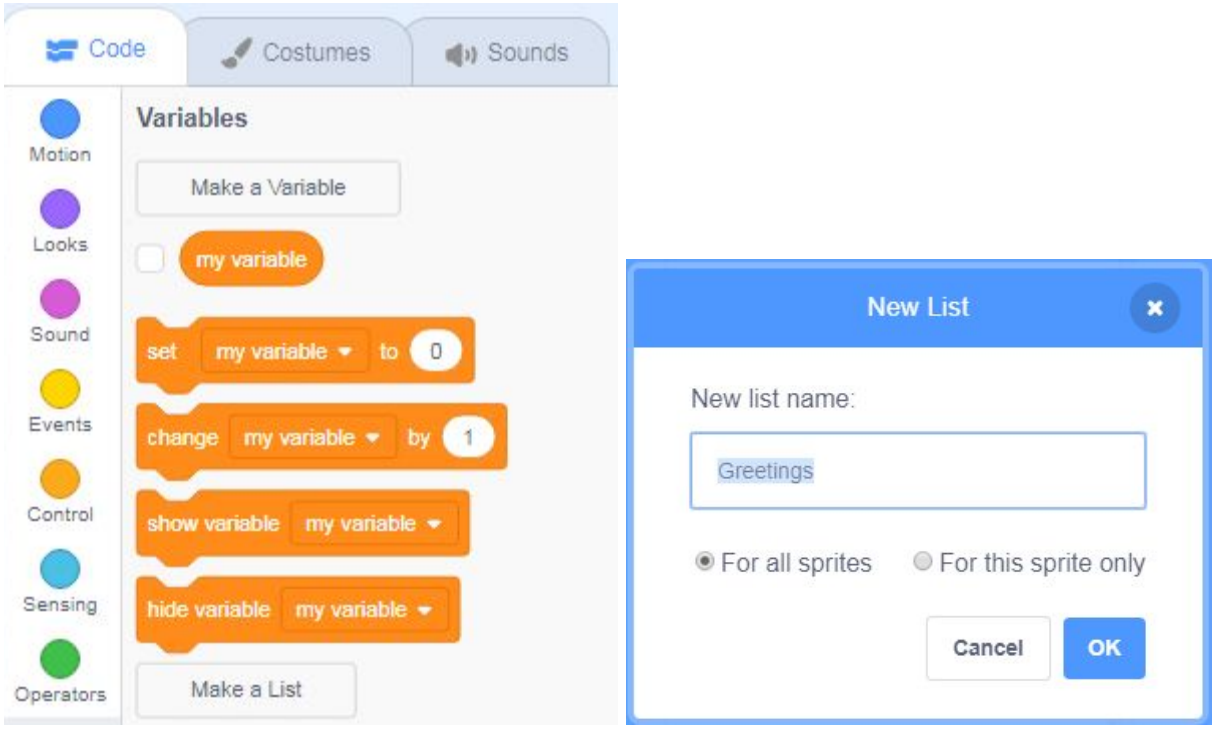

else
if answer = colours then
say Here are the colours! for 2 seconds
switch backdrop to red
say whero for 2 seconds
switch backdrop to yellow
say kōwhai for 2 seconds
switch backdrop to green
say kākāriki for 2 seconds
switch backdrop to black
say mangu for 2 seconds
switch backdrop to blue
say pukepoto for 2 seconds
switch backdrop to backdrop1
say OK maybe next time for 2 seconds

Session 3 – Building complexity

At the moment, our sprite can't say very much and is quite predictable. We can record a selection of greetings in a list and get our sprite to randomly choose which greeting to use.

Lists

A list (also known as an array) is a way of storing lots of pieces of information at once. In **Variables**, select 'Make a List' and call it greetings.



Create a new event by taking a 'when flag clicked' block from **Events**, and drag it into a space in the programming area. In **Variables**, find the 'add ____ to ____' block and drag it to under the 'when flag clicked' block. Change the text to say "Kia Ora" and select your list, greetings, from the drop down menu.

When you click the flag icon at the top of the stage, your list should now appear with Kia Ora as the first entry. Using more 'add ____ to ____' blocks, add more greetings to the list.

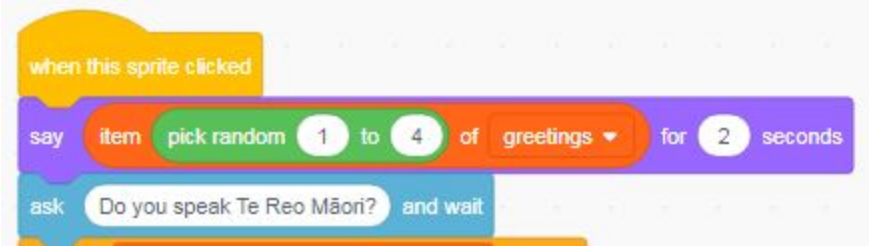
Now when you click the flag, multiple greetings should get added to your list.

We now need to get our sprite to pick one of the greetings to use. Find a 'say ____ for __ secs' block and add it to the top of your 'when this sprite clicked' event. From **Variables**, find the 'item __ of ____' and drag it into the text space of the 'say ____ for __ secs' block. Using the first drop down menu, you can choose a number. This refers to the item at that position in your list. Select a number and greetings from the second drop down menu. Now when you click the sprite, you it should say the greeting at the position you selected.

Got duplicates in your list? Try adding a 'delete ____ from ____' block to your 'when flag clicked' event to remove all the previous entries.

Random number generators

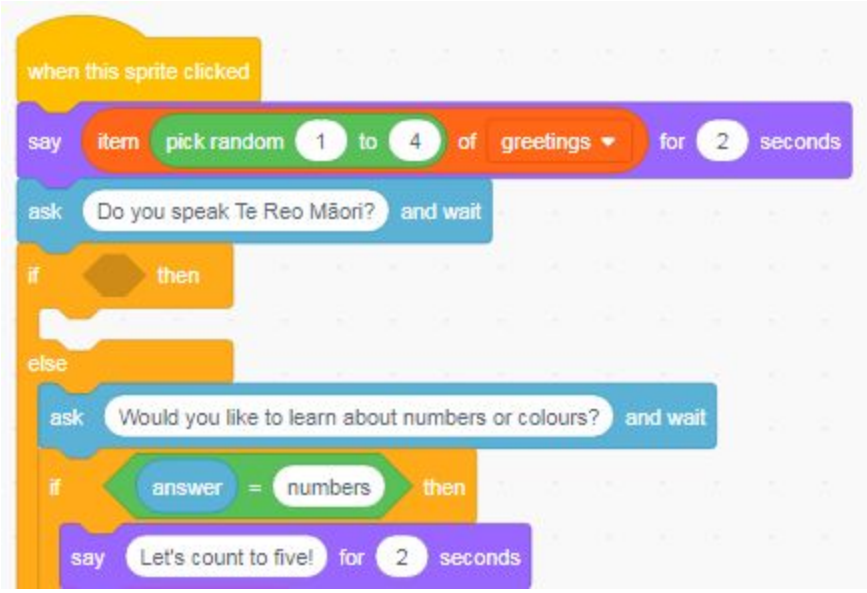
It is possible to ask the computer to choose a random number for you. In **Operations**, find the 'pick random __ to __' block and drag it into the space where the number is in your 'item __ of ____' block. Change the numbers to reflect the number of greetings you have in your list, eg if you have 4 greetings, change the numbers so that the block reads 'pick random 1 to 4'. Every time you now click the sprite, the computer will randomly select a number between 1 and 4, and then say the greeting at that position in your list.



We can also use lists to set a variety of different answers to a question. Under your first 'say ___ for __ secs' block, add a new 'ask ____ and wait' block. Ask the question "Do you speak Te Reo Māori?".

Create a new list called "saying yes" by selecting the 'make a list option' under Variables. Under 'when flag clicked', add in all the different ways you can think of that someone might reply yes. You might want to also include common spelling mistakes and typos!

We're going to use an 'if ____ then else' block to help us decide what to do with the answer to the question "Do you speak Te Reo Māori?". Drag the block and place it under the first ask so that the remaining code is inside the else section. Drag the second ask out from under the first ask and drop this large chuck of code into the 'else' section of this new if/else block.



In the if section, we want the sprite to respond to a yes answer. To do this, we can use a block that searches our whole list of ways to say yes. From **Variables**, find the '____ contains ____ ?' block and drag it into the space in the 'if ___ then else' block. Using the drop down arrow in the first space, select your saying yes list. In the second space, drag an 'answer' block from **Sensing**. Use a 'say ____ for __ secs' to respond to the user.

> Extra: Can you create a little conversation between the user and the Sprite in Te Reo Maori if the user answers yes?

Session 4 – Next Level

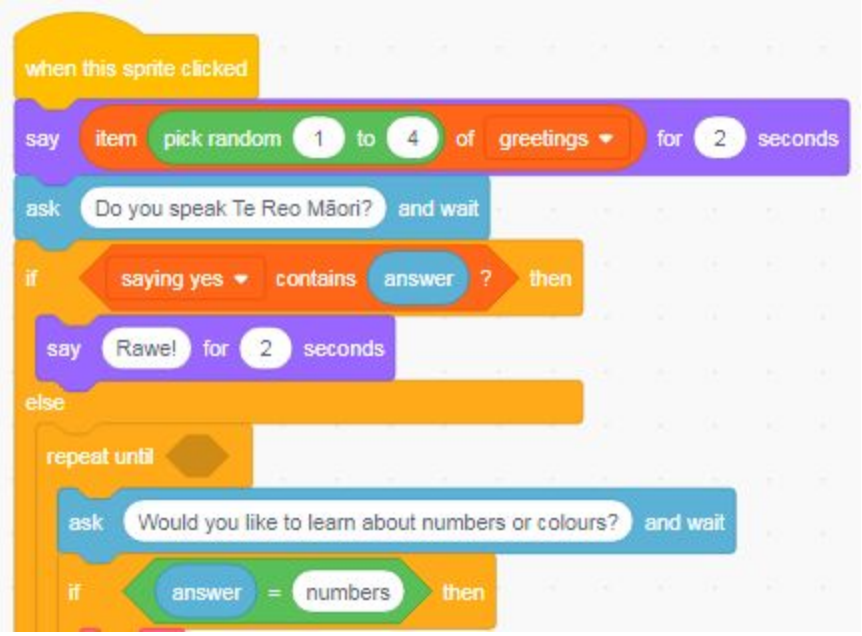
Repeats

With an program like a chat bot, it might sometimes be useful to be able to repeat a section of code until a particular answer is given (for example to allow people to retype an answer if they had a mistake the first time).

Scratch has a block that can be used to repeat section of code or a question until a recognisable answer is given (for example a correctly spelt option). In **Control**, there is a block called repeat. Other blocks can be put inside this block, and will repeat the instructions for the number of times specified or until a condition is satisfied. This is called a loop.

> A loop is a section of instructions that are repeated until they are told to stop. The signal to stop can vary, for example if something has repeated a certain number of time or if a certain condition has been met.

From **Control**, find the 'repeat until ____' block. We want to add it to our program so that it contains everything inside the first else section in our program. Click on and drag out the current contents of the 'else' section of first if/else block (from the 'ask Would you like to learn about numbers or colours' block down), and place it inside the 'repeat until' block. Replace all of this back in the 'else' section of the first if/else block.



We want to repeat asking our question until we have an answer we understand (either colours or numbers). For this, we will use a Boolean Operator.

Boolean operators

In **Operators**, there are six green blocks that are hexagonal in shape. These blocks (and every hexagonal block in Scratch) are Boolean Operators. These blocks give simple conditions that provide the computer information with which it can use to make a decision. They are commonly used with conditional statements like the 'if ____ then' block.

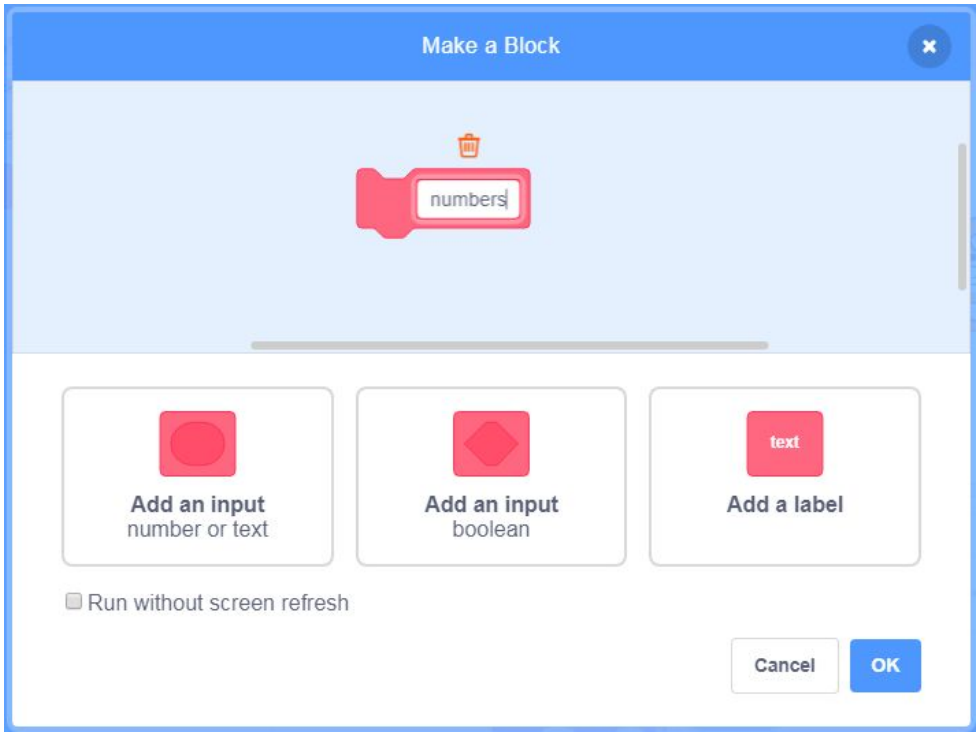
We've already used one Boolean operator, the equals sign. In this case, we are going to use 'or'. This lets us provide two conditions, and if one of them is true, our program can carry on. Drag the '__ or __' block into the space in the 'repeat until ____' block. Add an '__ = __' block either side of the 'or'. Create a statement that says "repeat until answer equals numbers or answer equals colours"

Try your program. What happens if you misspell numbers?

> Extra: Can you get your Sprite to respond if the user answers no?

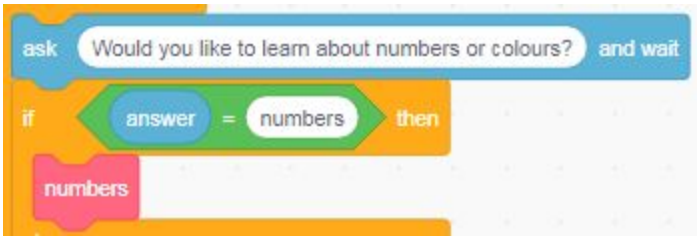
Make a Block

Currently, our programming area is looking messy. We can tidy it up and make it easier to read by using Scratch's **My Blocks** function. We can use this to define our own blocks. In **My Blocks**, click 'Make a Block', and call it 'numbers'.



A new block will appear in your programming area. This is a space for you to define your new numbers block. Drag the code that gets the sprite to count to five out of the conditional statement and put it under the define block. Drag the code that gets the sprite to count to five out of the conditional statement (if answer = numbers) and put it under the define block. If you click on the top piece of code directly under the 'if anwer = numbers' block it will automatically drag out the rest of the code within that part of the if/else block. Repeat the process to create a second new block for the colours code.

Make a Block creates a new Function. In programming, a function is a section of code that performs a specific task - In this case the task is saying the numbers or colours!



This could also be done with Broadcasting, a tool that can be used to pass messages between sprites and backdrops. This allows us to signal to a different part of our program when something has happened or to make something happen.

Congratulations! You have completed your project! Click the green flag to run the program and hit share to show your friends! Well done.