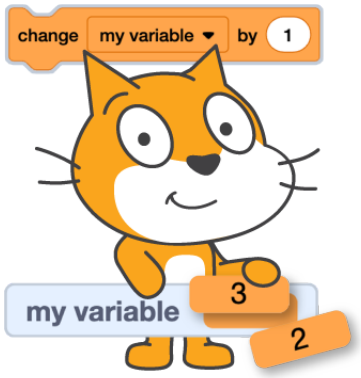
Variables and Lists

Have you used Variable or List blocks in a Scratch project? They can be a powerful tool when you are creating customized programs. Perhaps you have used a variable to store a game score*, but did you know a variable can hold numbers *or* text (also known as a "string")? And if you are storing a lot of custom information, using a list may be more efficient than creating multiple variables. Let's explore what variables and lists are, and see a number of ways they can be used.

**See our in-editor tutorials or coding cards for instructions on how to set up a basic score.*

What Is a Variable and a List?

You can think of a variable like a container that stores an assigned value/data/information. Variables can only contain one piece of information at a time, so as new information is placed inside the variable, the old information is replaced.



A list, on the other hand, can store multiple pieces of information, all ordered as separate items. A list is also called an "array" in other programming languages.

The benefit of storing information in either a variable or a list is that it can be recalled later in your program. For instance, you can:

- ask the program to check the current score of the game and use that to determine if a player has won or lost
- record a series of answers provided to questions and restate the information gathered later in the form of a sentence to customize a story
- let the user control the speed of a character's spin
- let the user control if music is on or off

The information stored can change the direction of a game or story, turn on and off actions, signal special messages, and more!

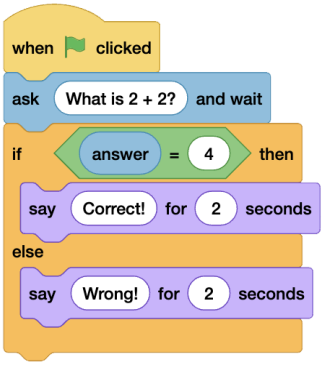
Scratch Reporter Blocks

Variables and lists hold information you can use in your program, but Scratch comes with some built-in reporter blocks that also store information. Unlike a stack block, which can be placed directly above or below another block, **reporter blocks go inside another block to serve as an input, hence their oval shape**. Here are some examples:

Ask a Question and Respond to the Answer

You might use the "ask" block to pose a question to a user. When the user types an answer into the dialogue box, it is stored in a reporter block called "answer."

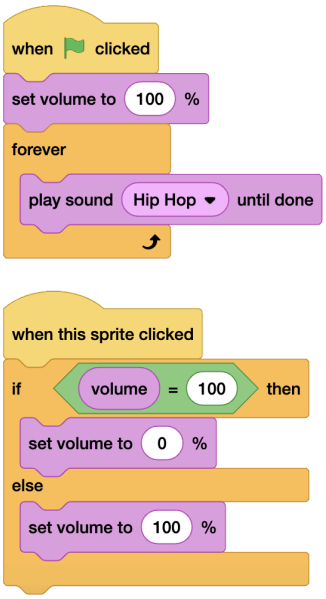
You can then use a conditional "if then" statement to show one response if their answer is correct and a different response if the answer is incorrect.



Adjust the Volume

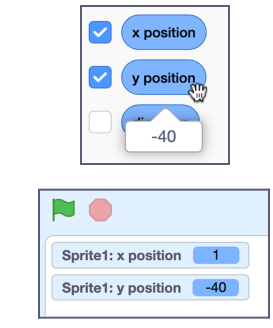
Another reporter block is "volume." It stores the number representing the current volume of the sprite, clone, or stage. You could write a program checking the volume of a sprite when the user clicks the sprite. Then, use a conditional "if then" statement so the script sets the volume to 0% or 100% (turning the sound on or off) depending on what that current number is.

(What do you think the difference is between the "volume" reporter block in the Sound category and the "loudness" reporter in the Sensing category? "Loudness" reports how loud the microphone input is. You could try using "loudness" to control a sprite. For instance, as the noise in the room gets louder, your project could have the sprite change costumes or you could program your sprite to move different distances based on the loudness level.)



Reporter Block Characteristics

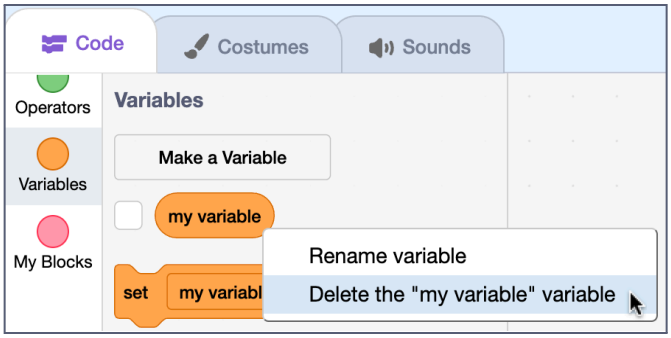
Look through the various block categories in the block palette and see if you can find other reporter blocks that store pieces of information you can use. You can click on a reporter block in the block palette or in the script area to see the piece of data it currently holds/the value it reports. You can also check a box next to many of these reporter blocks to display them on the stage via a stage monitor.

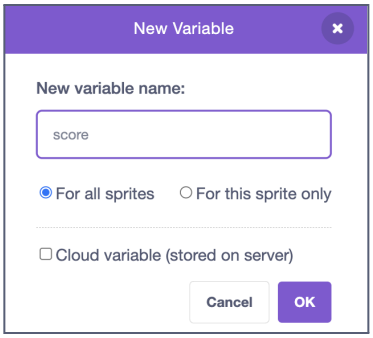


Create Your Own Variable

What if there isn't a built-in reporter block to store the information you need? In that case, you can create your own variable! Select the "Variables" category from the block palette. A generic variable called "my variable" is already provided.

You could rename it or delete it by right-clicking on it. Or click the **"Make a Variable" box** to create a new one. When the "New Variable" text box appears, type the name of the information you want to store in the text box for the name.



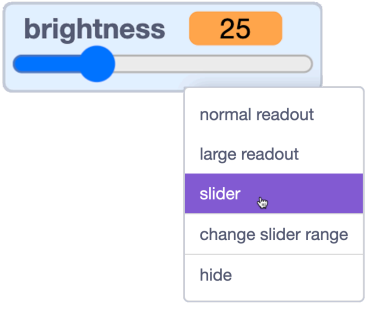


You can then **choose if all sprites will have access to the variable, or only the current one you've selected**. Many times, you'll probably choose to leave it as the default "for all sprites," so any sprite in your program can change the variable, reset the variable, or check the data the variable currently holds (for instance, an overall game score). However, there may be times where a variable only applies to one sprite. For example, when different sprites might have their own health meter in a game. If you opt to use Cloud Variables, they are stored "in the cloud" or on Scratch's servers so the values are preserved when projects are reloaded.

Notice the different types of blocks available in the Variables category. There are blocks to set or change the variable, for instance, and even a variable reporter block (that oval block) that can be used inside another block.

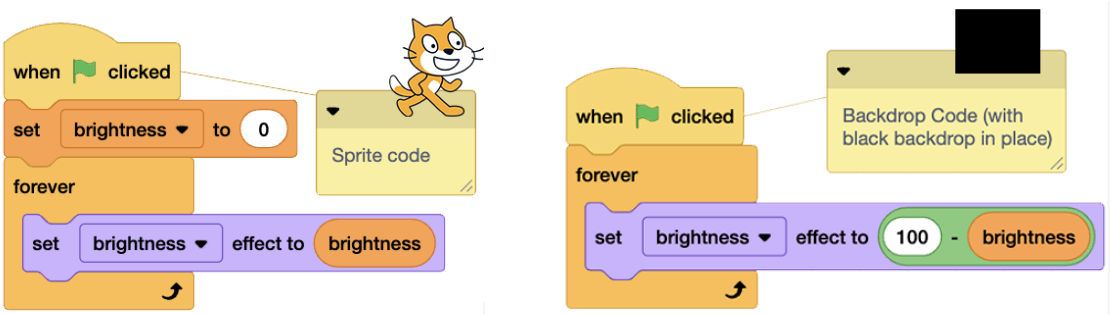
Using Variables

Let's create a project where the variable controls the brightness of the sprite. You'll see there is no reporter block for brightness available in the block palette. Create a variable called "brightness" and check the box next to your custom "brightness" variable in the block palette so it will be shown on the stage. Now, **right-click its stage monitor and you'll see you have a few options: "normal readout," "large readout," and "slider."** Let's make this a slider, so the user can control the sprite's brightness.



You could create a short script that says: When the green flag is clicked, the “brightness” variable should be set to 0. Then, the program should forever set the brightness effect to the number in the variable “brightness.” Click the green flag and adjust the slider to test!

And what if you want to have the background’s brightness be the opposite of the sprite’s? You could make a black box as your backdrop and then create a script that says the background should forever set the brightness effect to 100 minus what is in the “brightness” variable. Test again to see how this works. The brightness of the background and the sprite should be opposite. What else could you try?



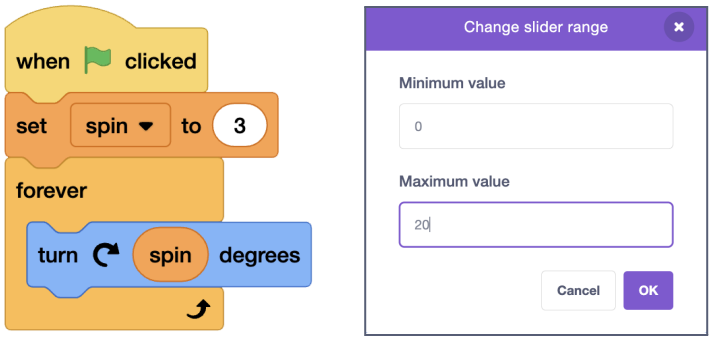
Slider Variables

Sliders can be handy for other things, too, like speed or volume.

If you want to set limits on the range of numbers available in the slider, you can right-click on the variable’s stage monitor to set a **“slider range” minimum and maximum**.

For instance, you could create a project with code like that below, where the slider variable controls the speed of the spin by setting the “spin” variable as the number of degrees to turn.

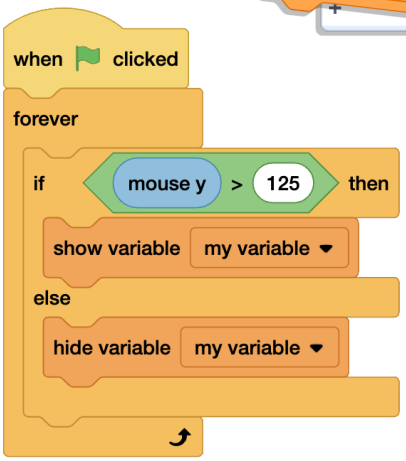
If it spins too fast, perhaps that makes you dizzy, so you may want to set a lower maximum number for the slider.



Show or Hide Variables or Lists

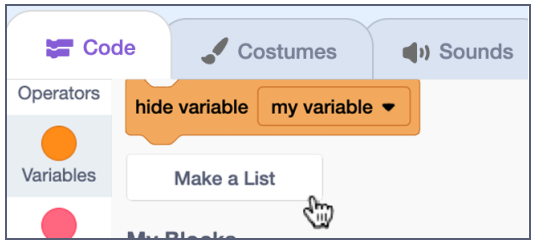
Use the blocks “show variable” and “hide variable” in your program if you want to control when the variable’s stage monitor is shown on the stage (for example, showing and then hiding when they reach a certain point in the program, like the ending screen). You can use similar blocks for showing and hiding lists.

You could also choose to show or hide the variables on the stage based on something like mouse placement, like in the example here, so they don’t clutter the screen until the user wants to make a change.



Create Your Own List

To create a list, select the “Variables” category from the block palette and click the **“Make a List” box**. No default list is present in a new project, so no list blocks will be visible until you create a list.



When the “New List” text box appears, type the name of the information you want to store in the text box for the name. You can then choose if all sprites will have access to the list, or only the current one you’ve selected. Once a list is created, a mix of stack blocks and reporter blocks will appear under the Variables category for you to use.

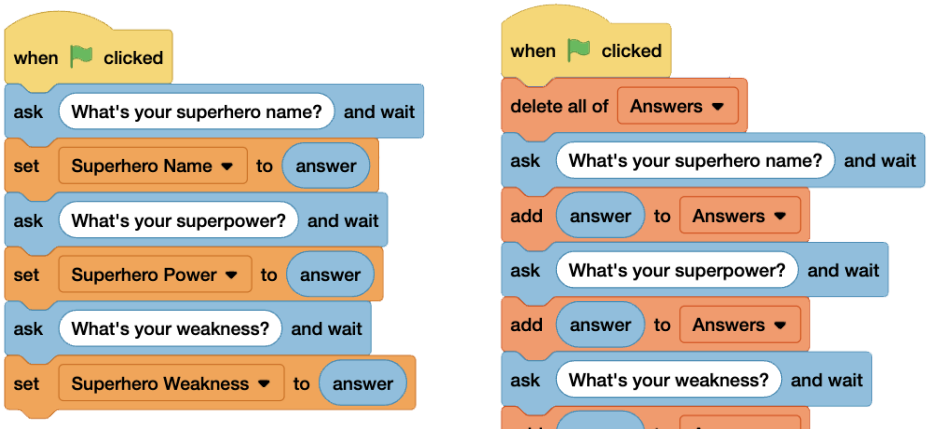
A list could be used to store multiple pieces of related information, or it could be used in place of creating multiple variables.

Pass Data Into Variables and Lists

You can pass information to a variable or list by clicking or moving a sprite, adjusting a slider, via code blocks, and more! You can also pass information from one reporter block, variable, or list to another. This could be helpful because variables and reporter blocks can only hold one piece of information at a time.

For example, say you want to create a project that collects the user’s answers to multiple questions. Each answer could be passed and stored in individual variables by using the “set [variable] to” block to pass the “answer” into a variable after each related question is asked. This requires setting up multiple variables.

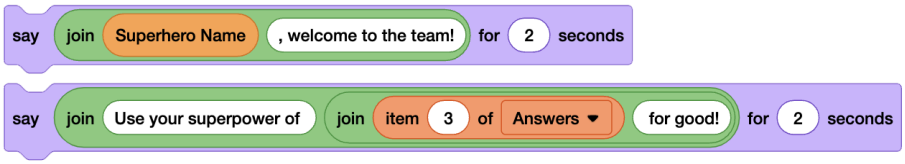
Or each answer could be added to a list by using the "add (thing) to (list)" block, where "thing" is the "answer" (see examples of passing data to variables vs passing to a list in the scripts below). One list could take the place of multiple variables. **With a list, new information is simply added to the list, rather than previous data being overwritten**. And the list gets longer and longer until you tell it to delete some, or all, of the information use the "delete" code blocks. If you are collecting a lot of data, storing it in a list might save you time.



Output Data from Variables and Lists

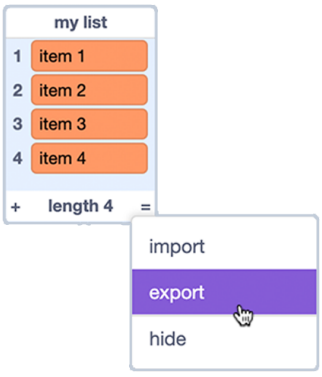
Information stored in a variable or list can then be recalled and used later. A list consists of numbers paired with items, so information from lists can be requested based on their placement within the list using the "item of (list)" reporter block.

For example, you could share the data in the form of a sentence using the "say" block to make a customized message for each user. Or try adding the variable and list inputs into other blocks to see the results!



Using Lists

You can check a box next to the list in the block palette to display it on the stage via a stage monitor. When visible on the stage, you can right-click on the stage monitor and choose to export data from the list to a file on your computer if you want to save it. Or you can import data into a list (from a .txt or .csv file), if you need to quickly add a lot of information at once. You can also click the plus (+) sign in the lower left



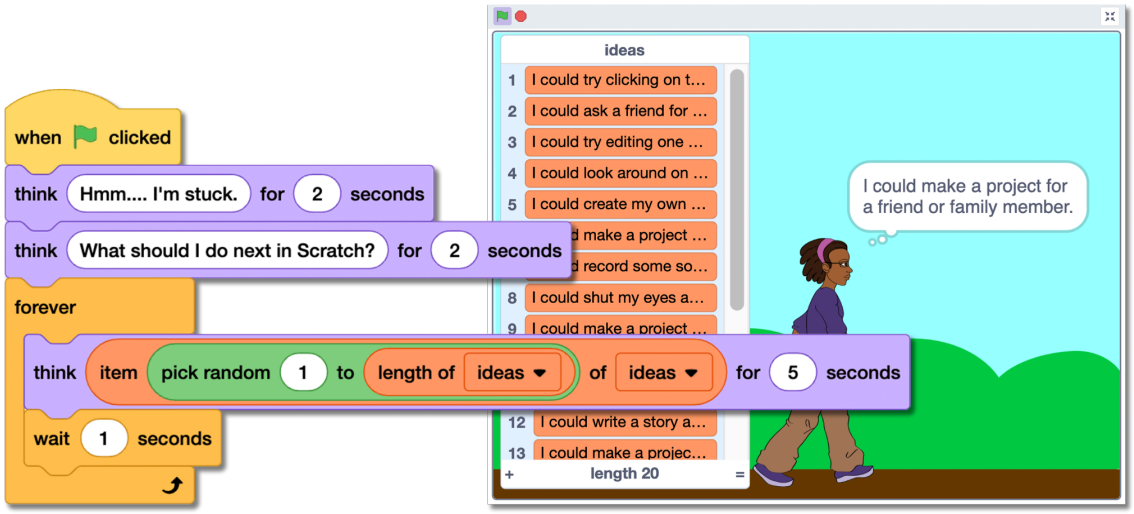
corner to add a new item to a list via the stage monitor, and click on the items to edit them by typing in the boxes of the stage monitor.

When you create a Scratch project and populate a list, **the information in the list will be retained until you tell the program to delete items from the list or delete the entire list**. The state of your project is preserved when you save it, meaning if there is information in your list, you do not need to tell the program to add those same items to your list each time it is run unless you have deleted them from the list and they need to be re-added.

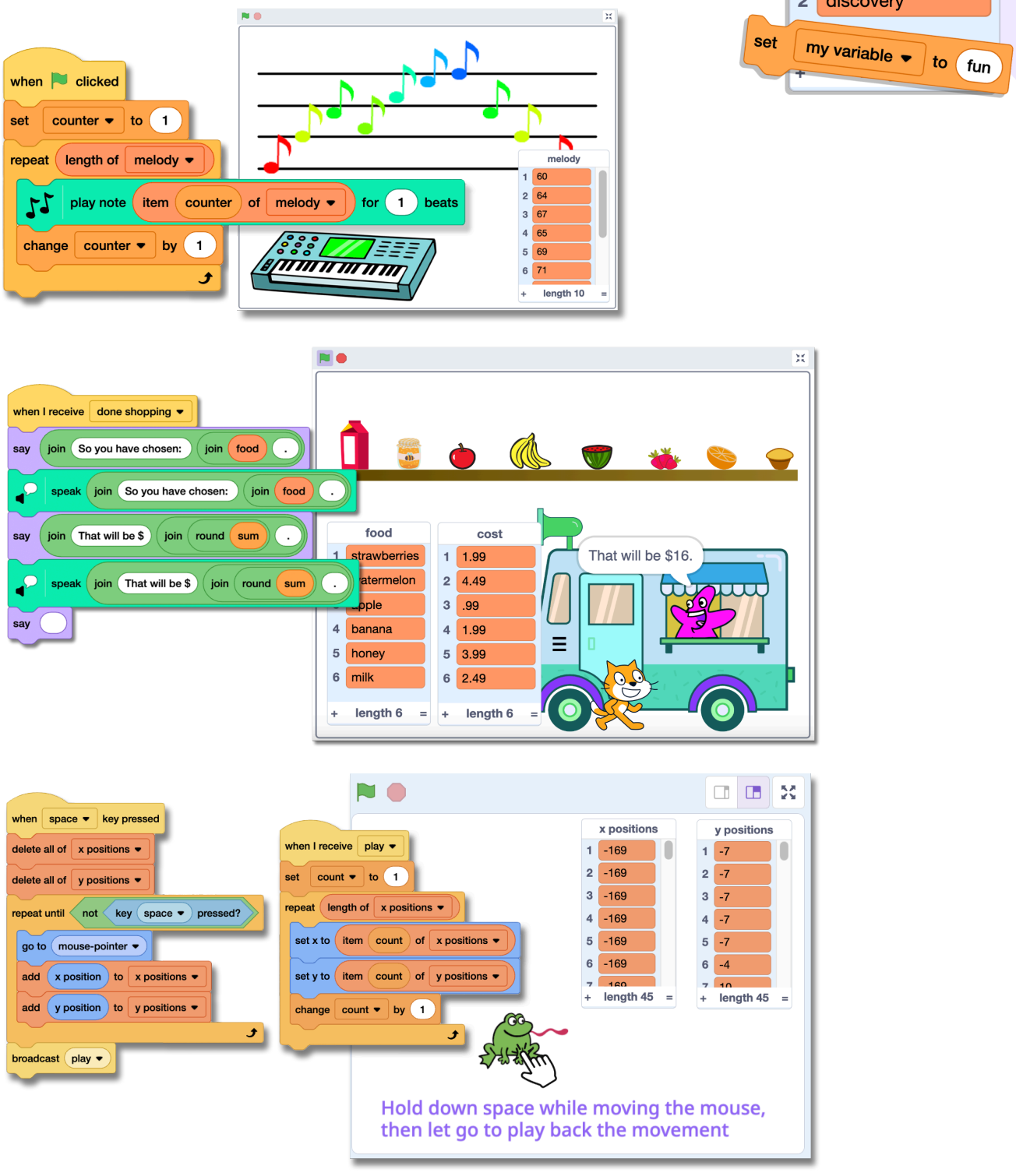
When a list has been populated, click on the list reporter blocks to see the value each holds.

There are a variety of simple and complex uses for lists, such as:

- using a predefined list to determine animation (like a melody project)
- performing calculations (like determining a sum of numbers in a list)
- storing inventories in games (to check what items have been collected or can be collected)
- encoding and decoding messages/strings (like an alphabet to binary code converter)
- recording multiple values like the movement of a mouse or sprite around the stage (like in a fruit slicer game or in a position recorder project)
- creating a system for users to type letters or display words/closed captions on the stage (like in a Mad Lib project)
- programming a randomized prompt idea generator (utilizing multiple lists full of adjectives or categories, for when you need ideas about what kinds of projects to create) or a simple version where the program picks a random item from a list of ideas



These are just a few of the possibilities. Look through all the blocks available for lists. What do you think each does? How might you use some of the blocks in a program to add, edit, or reset the data in a list? Experiment!



How will you use variables and lists to create your own customized program?

Tip: If you'd like to translate this guide, **click here to make a copy** of this Google doc.