🤖 NIRYO ROBOT WORKSHOP

Module 1: Variables

in NiryoStudio

High School Robotics Workshop • NiryoStudio Block Programming
Level: Beginner / Intermediate • Duration: ~45 min

🎯 Learning Goals

By the end of this workshop, you will be able to:

1. Understand what a variable is and why it's useful in programming.
2. Create, rename, and delete variables in NiryoStudio.
3. Use the **Set** and **Change** blocks to assign and update values.
4. Read a variable's value inside another block (e.g., a robot movement).
5. Apply variables to a real Niryo robot program.

📦 Section 1 — What is a Variable?

The Everyday Analogy

Imagine you have a labeled box in your room. You write "score" on the outside, and inside you put the number 10. Later, you can open the box, look at the number, change it, or use it however you like.

> 💡 **Definition**
> A variable is a named container that stores a value. The value can be a number, text, or anything else. You can read it, change it, or use it at any point in your program.

Variables are at the heart of almost every program ever written. Without them, a robot could not remember where it had placed an object, how many times it had repeated a movement, or what position it should move to next.

Why Use Variables in Robotics?

Think about a robot that has to pick up 4 items and place them in 4 different positions along a conveyor. Without a variable, you would have to write the position coordinates four separate times — and if you ever need to adjust the spacing, you would have to edit every single block by hand.

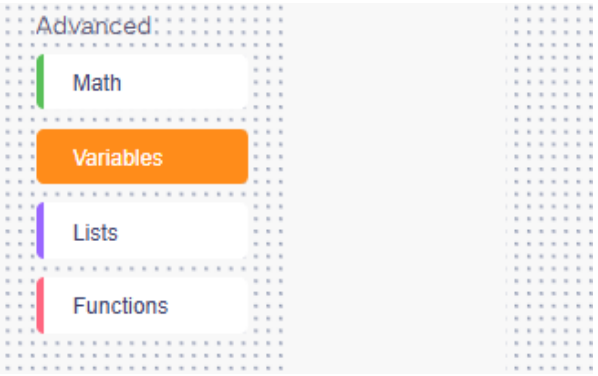
With a variable called y, you simply change y by a fixed offset each loop iteration. That is exactly what we will see in the real example at the end of this workshop!

Section 2 — The Variables Panel in NiryoStudio

Where to Find It

In NiryoStudio, open the block palette on the left side. Scroll down to the Advanced section. You will see four categories:

Math
Variables ← this is our focus today!
Lists
Functions



Advanced panel — unselected

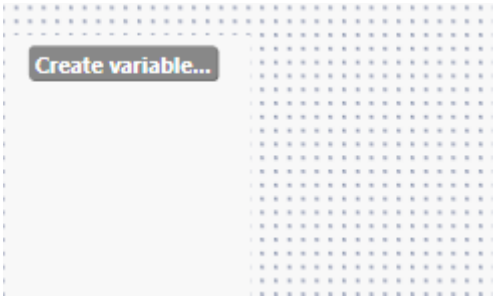
Click on Variables to open the Variables palette. At first, it looks almost empty:

The Variables palette before any variable is created

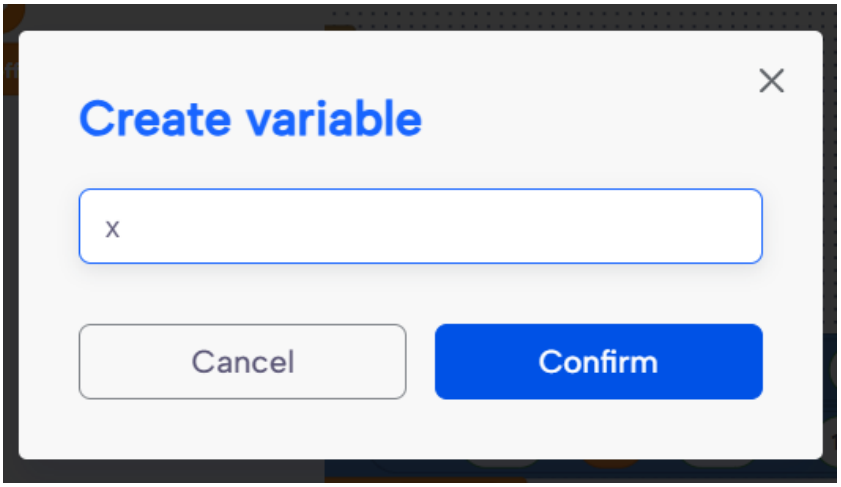
🔨 Section 3 — Creating a Variable

Step-by-Step

6. Open the Variables palette (see Section 2).
7. Click the grey 'Create variable...' button.
8. A dialog box appears — type the name of your variable (e.g., y) and click Confirm.
9. The three main variable blocks now appear in the palette!



The Create variable... button



The naming dialog — here the variable is called 'x'

> **📝 Naming Tips**
> Choose short, meaningful names. Use lowercase letters and avoid spaces (use underscores instead, e.g., y_offset). Variable names are case-sensitive: 'Y' and 'y' would be two different variables!

🧩 Section 4 — The Three Variable Blocks

Once a variable exists, NiryoStudio gives you three blocks to work with it. Let's look at each one carefully.

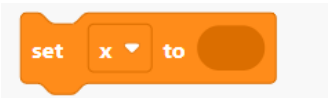
The three variable blocks: Set, Change, and the value reporter

Block 1 — Set [variable] to [value]

What it does

Assigns a specific value to your variable. Whatever was stored before is replaced by the new value. Think of it as writing a new number on a sticky note — the old one is erased.

Syntax in the block:



The dropdown (▼) lets you choose **which variable** to set, in case you have created several. The oval slot on the right accepts **any value** — a number or even the result of a math calculation.

🌍 Real Example

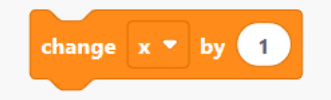
Set y to 0.2 will store the number 0.2 into the variable y. In our robot program, this is the starting Y position on the conveyor.

Block 2 — Change [variable] by [amount]

What it does

Adds a number to the current value of the variable. The variable is updated in place. If y was 0.2 and you change y by 0.1, then y becomes 0.3.

Syntax in the block:

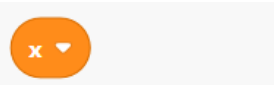


The dropdown selects the variable; the oval on the right is the **amount to add**. Use a **negative number** (e.g., -5) to subtract. This block is ideal inside loops because it updates the variable automatically on each pass.

🌍 Real Example

Here, instead of a fixed number, we pass in another variable called y_offset (which is set to -0.1). So after each loop iteration, y decreases by 0.1 — shifting the placement position along the conveyor.

Block 3 — The Value Reporter (the little oval)



What it does

This small orange oval block is not a standalone instruction — it is a value 'token' that you can slot into another block wherever a value is expected. It simply reads and returns the current value of the variable.

You can think of it like a reference card: wherever the block programming language needs a number or text, you can plug in this oval instead of typing a value manually. The robot will read the current value of the variable at that moment in time.

🌍 Real Example

In the 'Place from pose' block, the y coordinate slot contains the y oval instead of a fixed number. This means every time the block runs, it reads the current value of y — which changes on each loop pass thanks to the 'change' block!

✏️ Section 5 — Renaming and Deleting Variables

Right-Click Menu

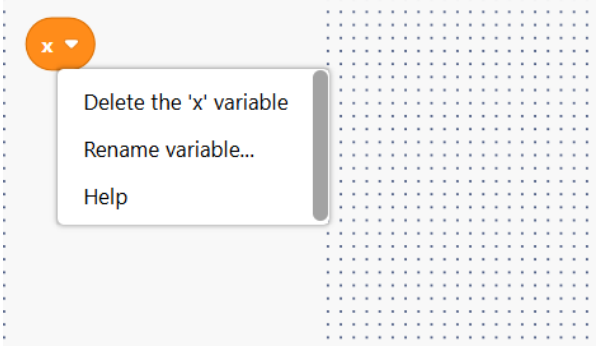
If you right-click (or click the dropdown arrow ▼) on a variable oval block already placed on the workspace, a context menu appears with three options:

Delete the 'x' variable

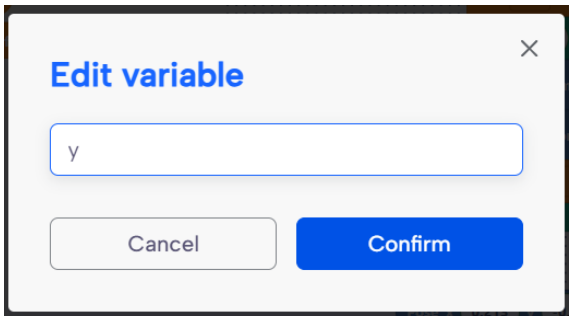
Permanently removes the variable and all blocks that reference it.

Rename variable…

Opens the same Edit variable dialog so you can give your variable a new name. All existing blocks using this variable are updated automatically.



The right-click context menu on a variable block

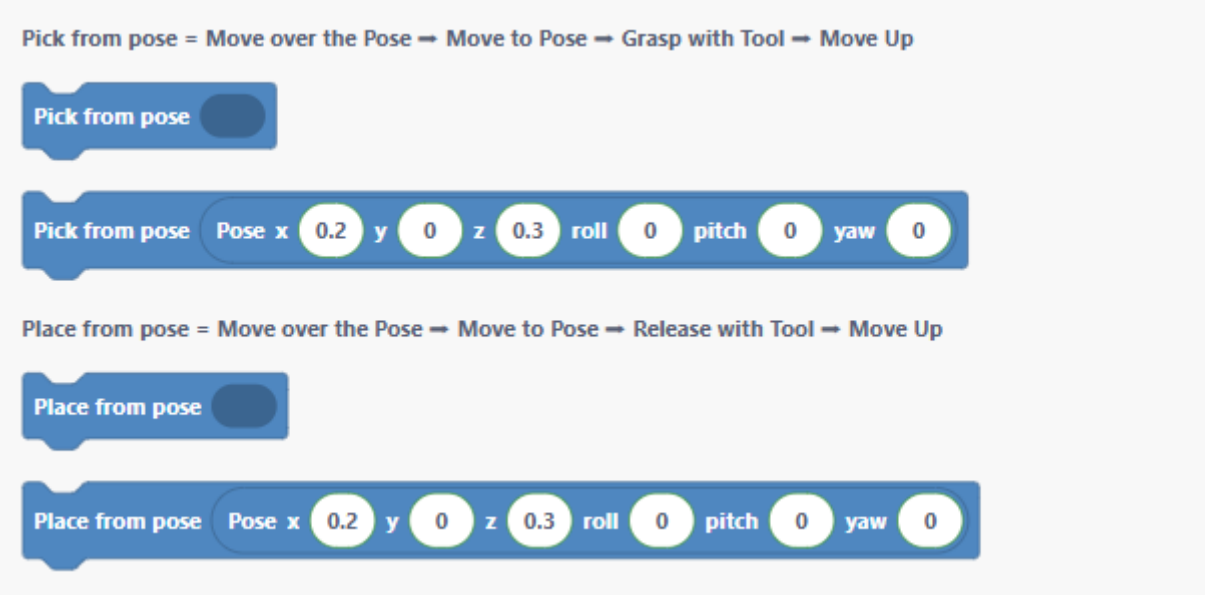


The “Rename variable” panel

> ⚠️ **Warning**
> Deleting a variable that is used inside loop blocks or movement blocks will also delete those blocks. Always double-check your program before deleting!
> You can undo by pressing CTRL-Z.

📖 Section 6 — Pick & Place Block Reference

For reference, here is a reminder of what the Pick from pose and Place from pose blocks do:



Pick from pose and Place from pose — with and without parameters

Pick from pose

Move over pose → Move to pose → Grasp with tool → Move up

Parameters: Pose x, y, z (position in meters), roll, pitch, yaw (orientation in radians).

Place from pose

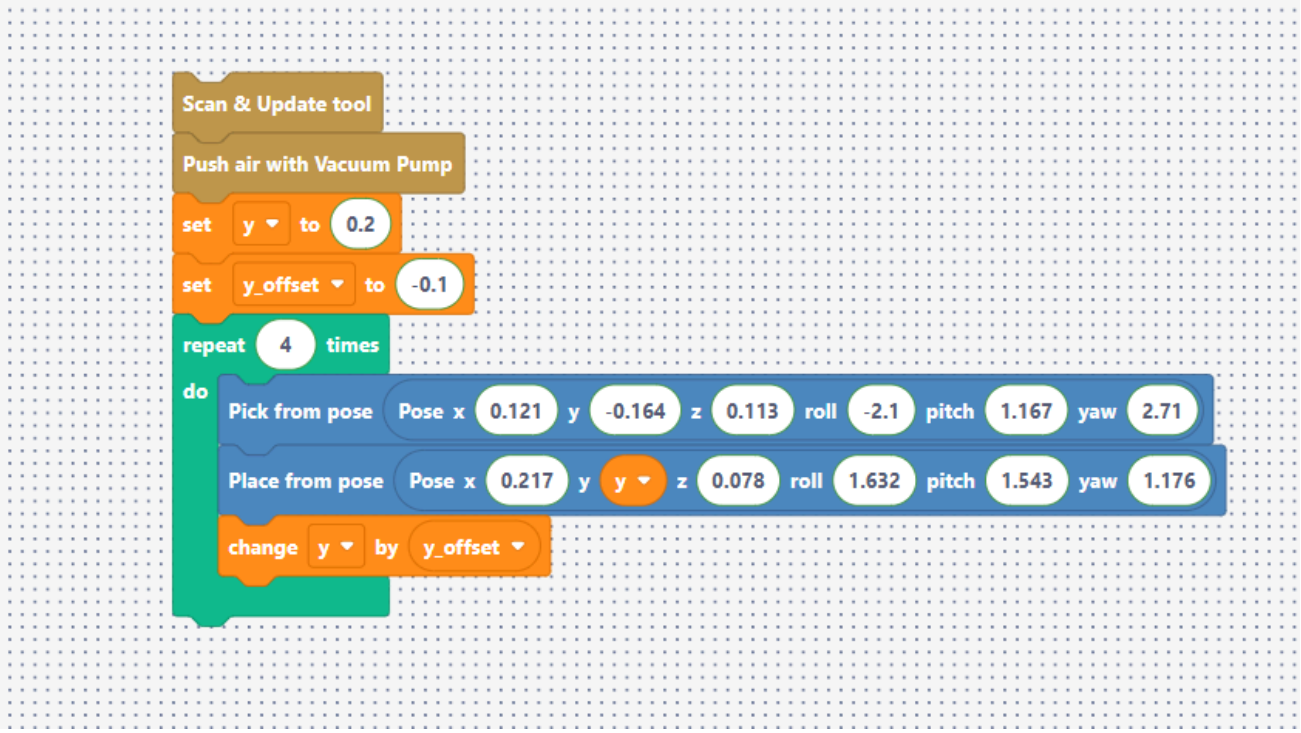
Move over pose → Move to pose → Release with tool → Move up

Parameters: Same as Pick. Here, the y slot uses the variable y instead of a fixed value.

🤖 Section 7 — Full Example: Robot conveyor Placement

What the Program Does

This is a real NiryoStudio program that uses variables to place objects on a conveyor at four different Y positions. Let's break it down step by step.



Complete program using variables y and y_offset in a repeat loop

Program Walkthrough

Step 1 — Scan & Update tool + Push air with Vacuum Pump

Before any movement, the robot prepares its tool (the vacuum pump) to be ready to pick up objects. This is standard setup code.

Step 2 — set y to 0.2

The variable y is initialized to 0.2. This is the starting Y coordinate (in meters) where the robot will place the first object on the conveyor.

Step 3 — set y_offset to -0.1

A second variable, y_offset, is set to -0.1. This is the gap between each placement position. A negative value means each position will be slightly lower (in Y) than the previous one.

Step 4 — repeat 4 times … do

The loop block runs everything inside it exactly 4 times — once for each object to be placed. Variables make this possible without copy-pasting movement blocks.

Step 5 (inside loop) — Pick from pose

The robot moves to a fixed pick-up position (x=0.121, y=-0.164, z=0.113…) and grabs an object. The pick position is the same every time, so no variable is needed here.

Step 6 (inside loop) — Place from pose (y slot = y variable!)

The robot moves to the placement position. Notice that the y coordinate slot contains the orange y oval — not a fixed number. This is the key moment where the variable is read and used.

Iteration breakdown

Iteration 1: y = 0.2 → place at y = 0.2 Iteration 2: y = 0.1 → place at y = 0.1 Iteration 3: y = 0.0 → place at y = 0.0 Iteration 4: y = -0.1 → place at y = -0.1

Step 7 (inside loop) — change y by y_offset

At the end of each loop pass, y is updated: y = y + y_offset = y + (-0.1) = y - 0.1. On the next iteration, the placement position has shifted by 10 cm along Y.

🧠 Key Insight

Without variables, you would need 4 separate 'Place from pose' blocks with y values of 0.2, 0.1, 0.0, and -0.1 hardcoded. With variables, one block handles all four placements dynamically — and changing the spacing is as simple as changing y_offset!