20060803

Exclusive software for autonomous robot KIROBO

Learn to Program a Robot with

IconWorks

MR-9132E
USER'S MANUAL I
=ALPHABET OF KIROBO=

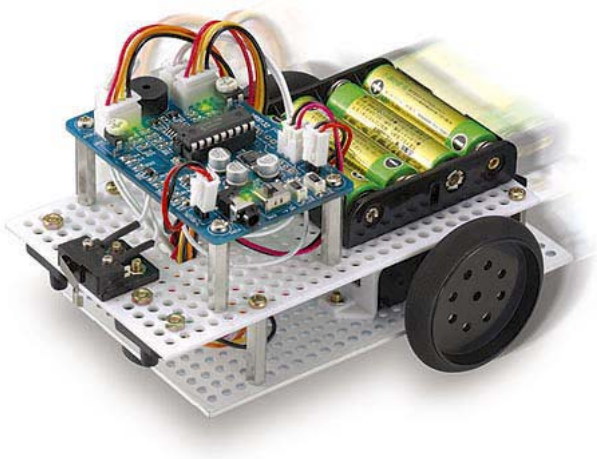
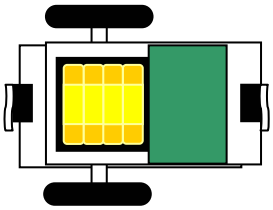


TABLE OF CONTENTS



I. INTRODUCTION

[1] NOTICE FOR GUARDIANS

PLEASE READ THE FOLLOWING WITH YOUR GUARDIAN BEFORE STARTING.

<Introduction>

IconWorks is dedicated software, designed for use with an autonomous robot kit "KIROBO", developed by EK Japan Co., Ltd. IconWorks and KIROBO have been developed in order to help beginners in autonomous robotics to experience the joy of programming and to develop their interest in science and technology.

<Use Restriction Policy>

- The IconWorks software is freely available but, its copyright is the exclusive property of EK Japan Co., Ltd. Re-use in any fashion whatsoever, such as reprinting, redistribution, sale, alteration or modification, is strictly prohibited.
- Changing or modifying all or part of the software into a readable form, by way of reverse assembling, decompiling, reverse engineering or any other way whatsoever, is prohibited.

<Disclaimer Policy>

- The user understands and assures that EK Japan makes no guarantee as to the accuracy, practicality and credibility of this manual and use of the software or the results arising there from. Furthermore, the user accepts that EK Japan makes no compensation for any damage incurred as a result of the installation or use of this software, which will be at the user's sole responsibility and liability.
- EK Japan undertakes no responsibility nor obligation to provide any kind of services such as (but not limited to) technical support, maintenance, improvement of this software.

<Please direct any inquiries you have to --->

EK Japan Co., Ltd.
2-19-30 Tofuro-Minami, Dazaifu City, Fukuoka, 818-0105, JAPAN
TEL: 81-92-923-8235 FAX: 81-92-923-8237
E-Mail : support@elekit.co.jp http://www.elekit.co.jp

● System Requirements

Use of IconWorks is possible only in the following environment.

Operating System (OS)	WindowsXP / WindowsVista
CPU	300 MHz or faster
Main Memory	128MB or more of RAM
Hard Disc free disc space	10MB or more of available hard disk space minimum
Display	800×600 SVGA display
Output	1 available headphone jack required to send a program using sound signals. (The program transfer cable is supplied in the KIROBO package.)

<Caution>

This software sends data using the headphone terminal. As such, there may be cases where data communication is not successful, particularly when the headphone signal output components of the PC are significantly degraded or damaged. If difficulties arise when transferring data, before proceeding further, please check the condition of your hardware. If you continue to have difficulties please test the data transfer operation using another PC.

<Notice to Guardians>

This software is intended for students of 10 years old and above at an elementary school on the presumption that he or she has a basic understanding of the operation of a PC. If not, supervision by a guardian is strongly recommended.

[2] SETTING UP "ICONWORKS"

●How to install IconWorks

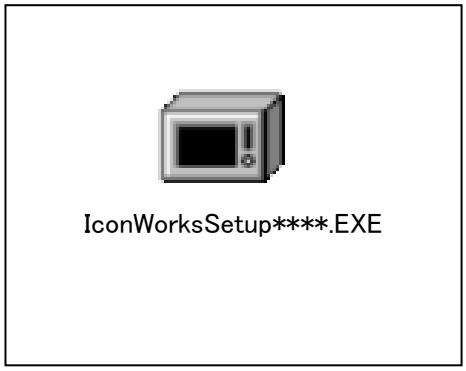
For WindowsXP

Download the IconWorks software in accordance with the instructions on EK JAPAN's homepage.

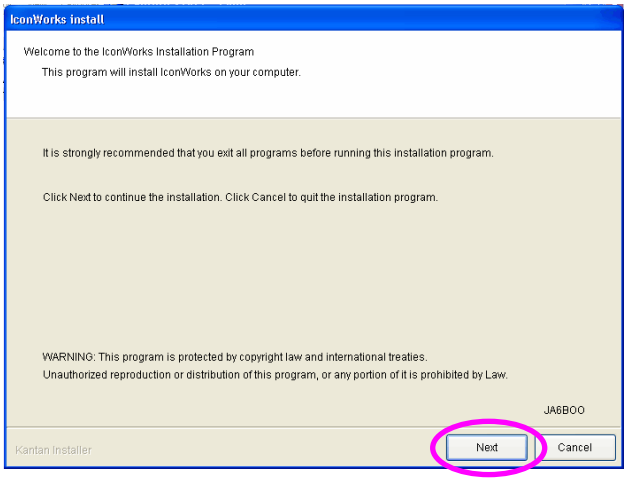
http://www.elekit.co.jp/english/download/index.php

1. Save the IconWorks software for "WindowsXP" from the download screen in an appropriate folder according to the instructions that appear on the screen. (The file named "IconWorksSetup_E" is saved.)
2. "IconWorksSetup****.EXE" is created in the specified folder.
3. Double-click the "IconWorksSetup****.EXE" icon to start the software installation process.
   * The numbers that come in the part of "****" change depending on the IconWorks version.
4. Restart the PC when "INSTALLATION COMPLETED" is displayed.

●Installation flow

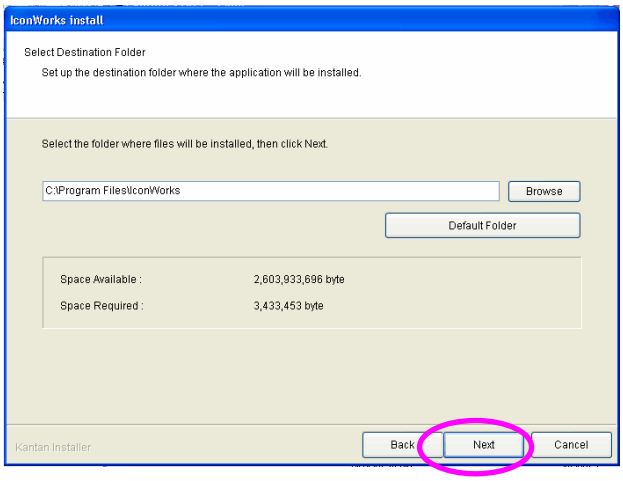


(1) IconWorks installation screen



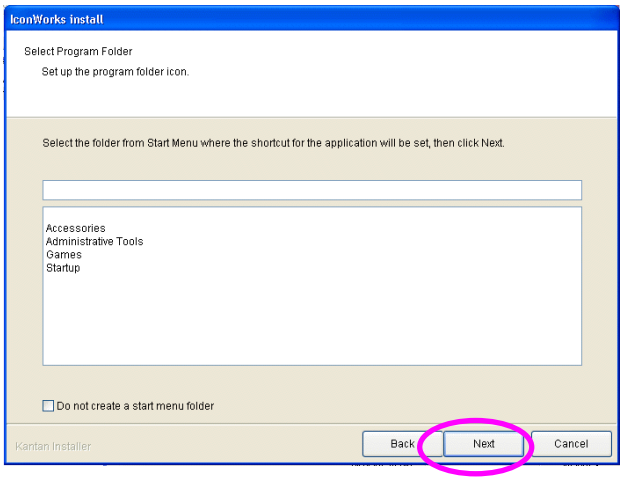
Click [Next].

(2) Specify the folder to install IconWorks



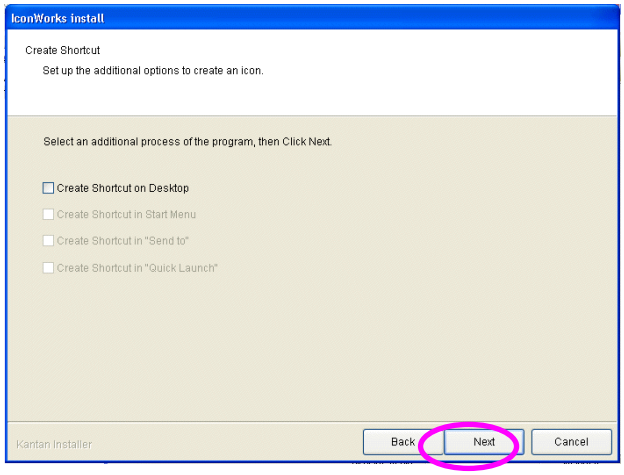
Click [Next].

(3) Select the program folder.



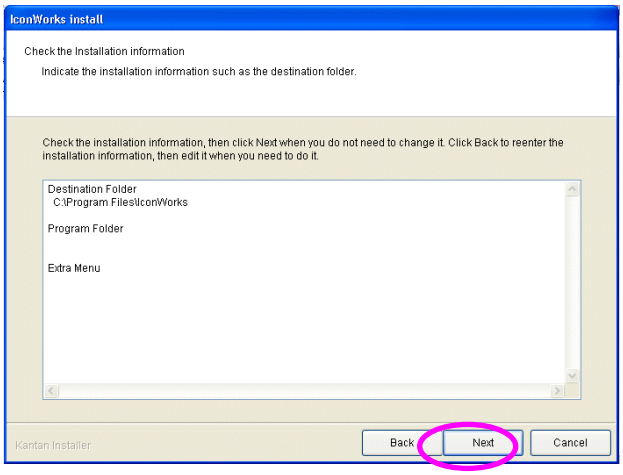
Click [Next].

(4) Set up the short cut.



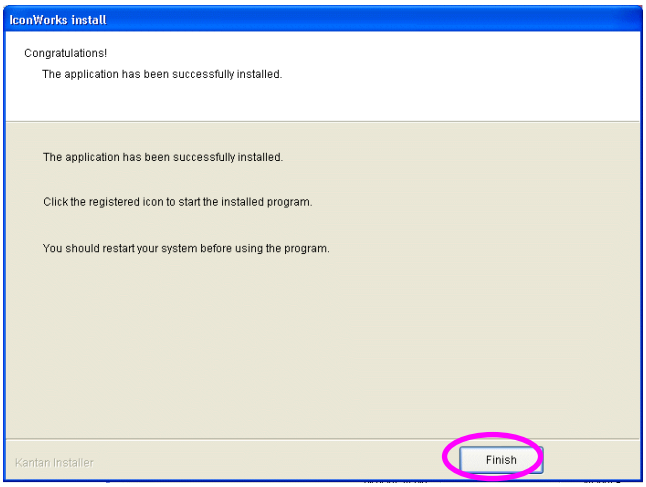
Click [Next].

(5) Check the installation detail.



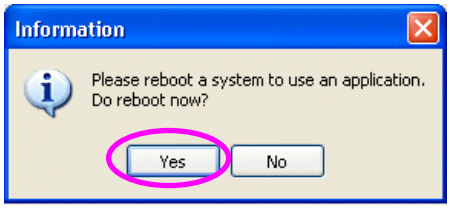
Click [Next].

(6) Installation is completed.



Click [Finish].

(7) Restart the PC.



Click [Yes] to restart the PC.

●Uninstalling IconWorks

1. Open the control panel and click "Add or Remove Programs".
2. Select IconWorks from the program list and click "Remove".
3. Click "OK" when the message "UNINSTALLATION COMPLETED" appears.

For WindowsVista

Download the IconWorks software in accordance with the instructions on EK JAPAN's homepage.

http://www.elekit.co.jp/english/download/index.php

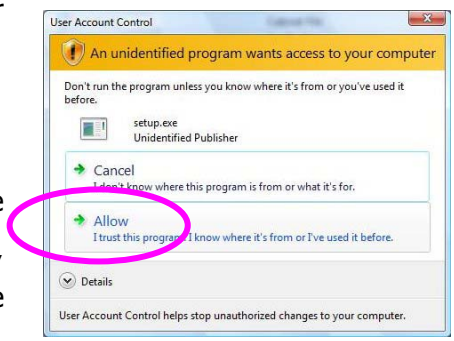
1. Save the IconWorks software for "WindowsVista" from the download screen in an appropriate folder according to the instructions that appear on the screen. (The file named "IconWorksVista***E" is saved.)
   * The numbers that come in the part of "***" change depending on the IconWorks version.
2. Double-click the downloaded file, and "setup.exe" is created in the folder.
3. Double-click "setup.exe", and follow the instructions that appear on the screen.

●Installation flow

Double-click the icon.

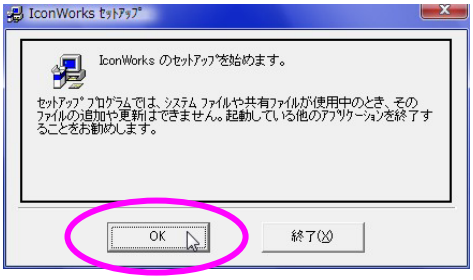
For WindowsVista, the message as shown on the right might appear after the above icon is double-clicked.
When this message appears, click [Allow].

Note: Depending on your PC environment the letters in the installation screens might not be displayed correctly. However, this is not a fatal error and the installation process can be continued.



The message that might be displayed

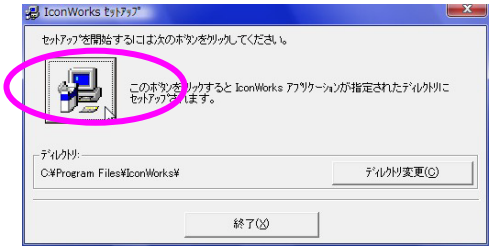
(1) IconWorks installation screen



Click [OK].

⇩

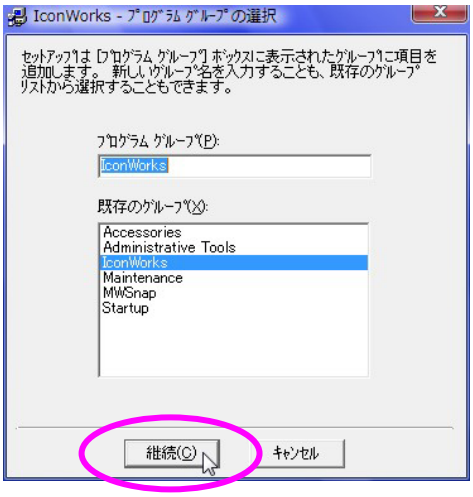
(2) Confirmation screen



Click the button.

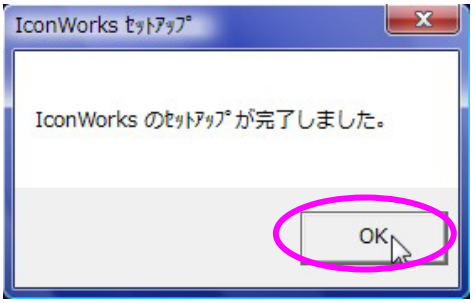
⇩

(3) Installing group



Click [Continue].

(4) Installation is completed.

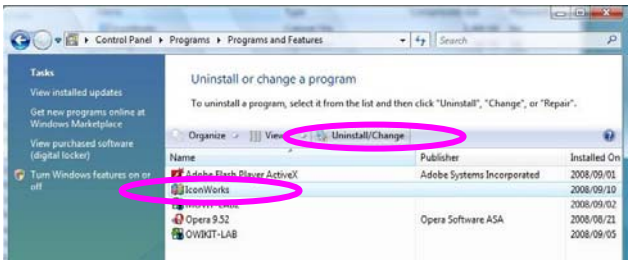


Click [OK].

●Uninstalling IconWorks

1. Open the control panel, and click "Programs", and "Uninstall a program".
2. Right-click IconWorks and click "Uninstall / Change".

Follow the instructions that appear on the screen, and click "Continue" and then "Yes".



[3] AUTONOMOUS ROBOT AND PROGRAMMING

What is an "autonomous robot"?

An autonomous robot gathers information about its surroundings, processes the information gathered through its sensors, makes judgments and sends commands to move a body.

An autonomous robot is able to carry out a "routine" or "pre-determined task" but also make judgments and actions, without external assistance or guidance, based on its environment and surroundings.

An autonomous robot can be defined by the following components:

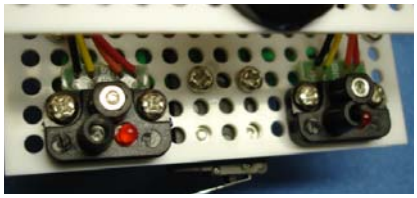
1) Sensor
2) Computer and program
3) Mechanism to move (motors, gears, etc)

1) Sensor (sensory equipment)

In order for a robot to gather information about its surroundings, it must have sensory equipment; a means to replicate the 5 senses (sense of sight, hearing, smell, taste, and touch) of a human. Various high-tech sensors, such as vision sensors, sound sensors, touch sensors, etc., have been developed and are being used more and more in our daily lives. Detailed below are the sensors used in KIROBO.

Sense of sight, camera, light sensor, etc

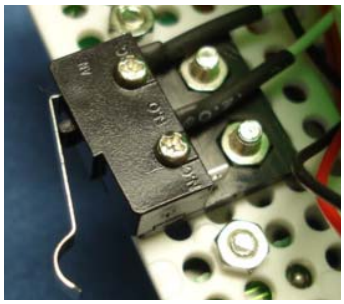
Most common small cameras around us are used as human "eyes", creating images on video monitors. There are many different types, some have 360 degree vision and in some cases 2 cameras are used to determine the exact location of an object. Recently, many sensors have been developed exclusively for robots. Some sensors do not see images but are able to detect the presence or absence of light. KIROBO has light sensors that can sense the presence of light emitting objects and detect contrast (black lines).



KIROBO's light sensors

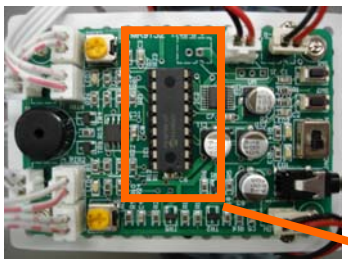
Sense of touch (skin), temperature sensor, touch sensor, pressure sensor, etc

These sensors can detect touch or temperature as human skin does. There are various types of sensor; ones that sense the electricity that runs through human body, those which allow electricity to flow when touched and ones that detect if it is touched when a pressure is applied to it, etc. . Touch sensors can be some of the most simple and are probably the most familiar sensors for us. They are also mounted on KIROBO and can be used to detect obstacles or serve for conditional judgment decisions when pressed.



KIROBO's touch sensor

2) Computer and program



KIROBO's microcomputer

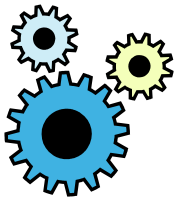
An autonomous robot is pre-programmed with what actions to take depending upon certain circumstances. The program is stored in the computers memory (RAM) in a language understandable for the robot and processed by the robot's brain (microcomputer). Both the memory (RAM) and the computer (microcomputer) are mounted on the main board (PCB).

To create series of actions for a robot, in a robot (machine) language, is called "programming" and the series of action commands created in the robot language is called "program". A completed program can be sent to the memory of the microcomputer which stores the action commands. The programming language varies depending on the types of computer used or the objectives of the work, ranging from a low-level language, easy for the computer to process, to a high-level language which is easy for human-being to comprehend.

A concept called "artificial intelligence" is available in some high-tech autonomous robots. Although an average robot can only do what it is told, those with "artificial intelligence" can learn from experience, and even apply such knowledge to future situations.

The action commands for KIROBO are indicated with Icons in the IconWorks software so that it is possible to create a program without knowing difficult programming languages. Therefore, with KIROBO, you can take the first step for mastering the basic concept of programming.

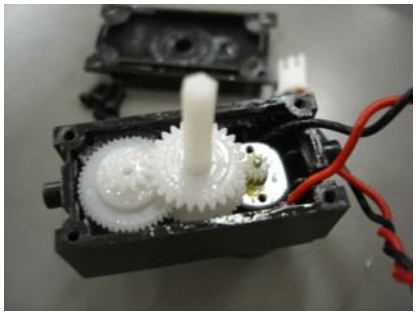
3) The mechanism of the motor and the gears



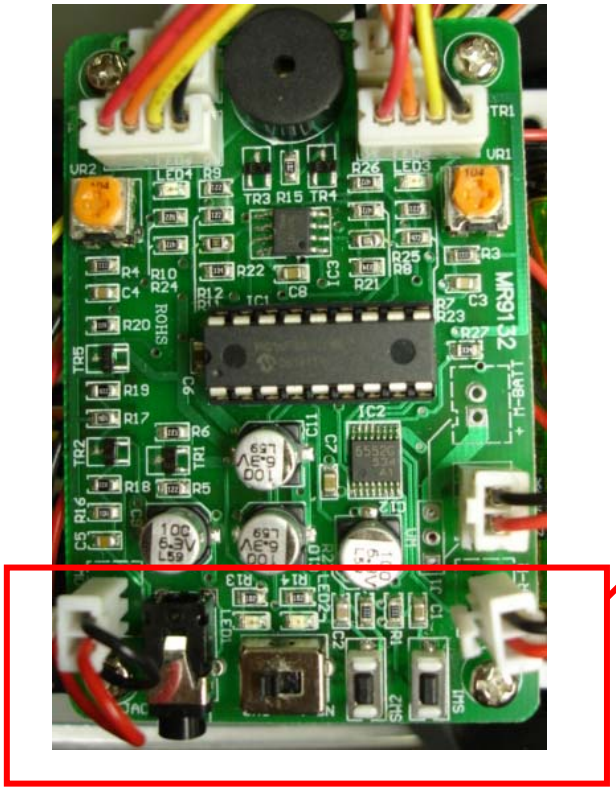
A mechanism like human muscles is necessary in order for a robot to move smoothly. As a human has muscles, a robot has motors and gears. Nowadays, various motors and gears, such as servo motors, hydraulic and pneumatic actuators etc, have been developed, which enable robots move very precisely just as humans can. Some of them can place an object in exactly the place designated and some can even go up stairs like a human by combining many moving parts. Almost all of them are controlled by a computer but some also have assistance from a human controller who may be at some remote location from the robot.

With KIROBO it is not possible to see how the motors and gears function. However, small motors and gears are installed in the 2 pre-assembled gear boxes of KIROBO.

* *Changes in the gear box and modification of the drive can be applied only to motors of the same specification.*



Inside the KIROBO geared motor.



KIROBO motherboard

The names and functions of each part on the motherboard are described on Page 6 of KIROBO USER'S MANUAL I.

The buttons used for program transfer and execution are located here.

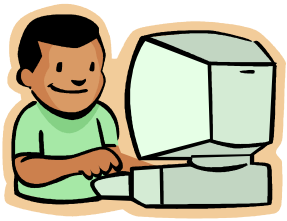
The motherboard is a PCB or Printed Circuit Board. It is a means of connecting the components of the robot. The CPU or Central Processing Unit which does the 'work' of the program, the RAM or Random Access Memory which stores the program and data before and after processing by the CPU.

[4] ABOUT ICONWORKS

● Simple and easy-to-learn

It is designed to help beginners to learn how to program an autonomous robot with the greatest ease.

To control an autonomous robot requires a high-level of knowledge. In IconWorks, however, a beginner can learn programming by manipulating icons which are interpreted as a command --- all you have to do is to place colorful icons, like when you play cards, in accordance with the type of a robot you want to make and how you want it to move.



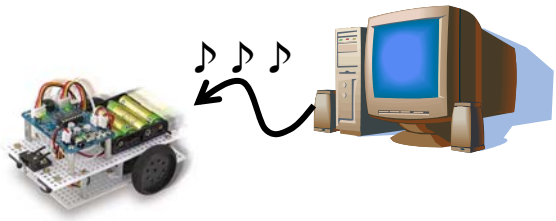
IconWorks has been designed for beginners to master the basics of programming; therefore, the functions contained in it are kept to a minimum. Nevertheless, there are an abundance of possibilities available for you to make the most of it. It will all depend on how and what you want to learn.

Remodeling a robot and challenging competitions such as robot dancing or line tracing will be a lot of fun too!

● About "sound communication"

Thanks to "sound communication", a completed program can be transferred easily to the robot.

Conventionally, it is necessary to buy a special communication cable or a piece of additional hardware depending on the PC.
In IconWorks, however, a program can be easily transferred using sound and the enclosed program transfer cable.



[5] USER SUPPORT INFORMATION

Any questions, suggestions or request for an update of the information, please send an inquiry to:

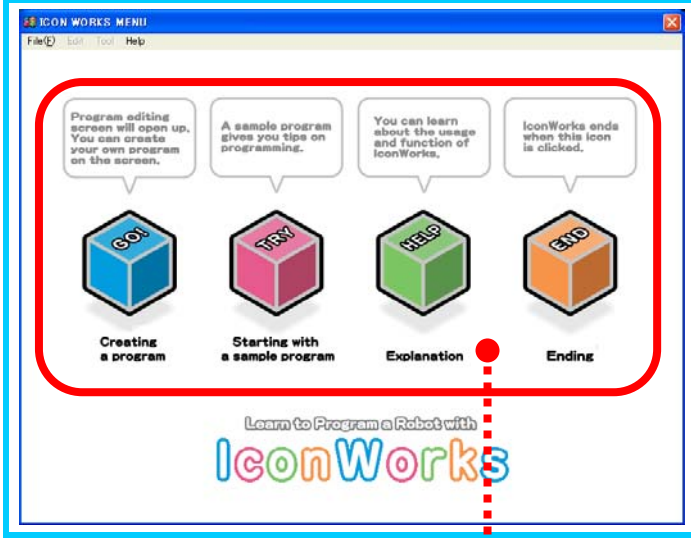
EK Japan Co., Ltd.
2-19-30 Tofuro-Minami, Dazaifu City, Fukuoka 818-0105, JAPAN
TEL : +81 – 92 – 923 – 8235 FAX : +81 – 92 – 923 – 8237
E-Mai l: support@elekit.co.jp http://www.elekit.co.jp

or our local distributor (Please ask us for contact information).

II. THE BASIC SCREEN AND ICONS

[1] STARTING AND ENDING ICONWORKS

Let's start IconWorks (the initial screen)



This is the IconWorks welcome screen.

Here is how to start it.

<Start>

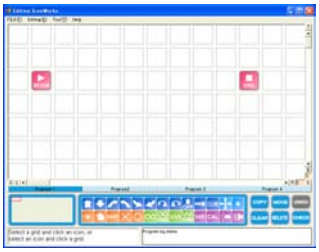
Or, double-click the short-cut icon on the desktop.

Icon 

LEFT-CLICKING AN ICON ON THE INITIAL SCREEN WILL TAKE YOU TO ITS ASSOCIATED SCREEN.

CREATING A PROGRAM: GO!.

PROGRAM EDITING SCREEN WILL OPEN UP. YOU CAN CREATE YOUR OWN PROGRAM ON THE SCREEN



STARTING WITH A SAMPLE PROGRAM: TRY

A SAMPLE PROGRAM GIVES YOU TIPS ON PROGRAMMING.

* Refer to the USER'S MANUAL II, SAMPLE PROGRAMS.

EXPLANATION: HELP

YOU CAN LEARN ABOUT THE USAGE AND FUNCTIONS OF ICONWORKS.

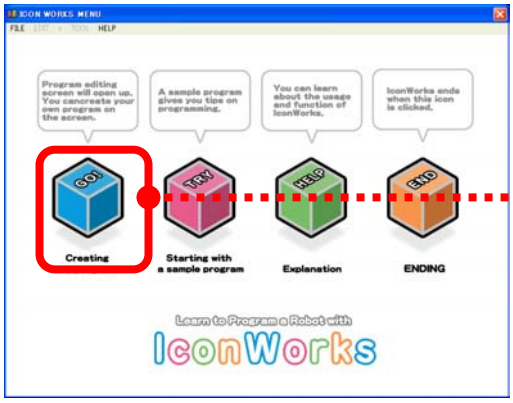
ENDING: END

ICONWORKS ENDS WHEN THIS ICON IS CLICKED.

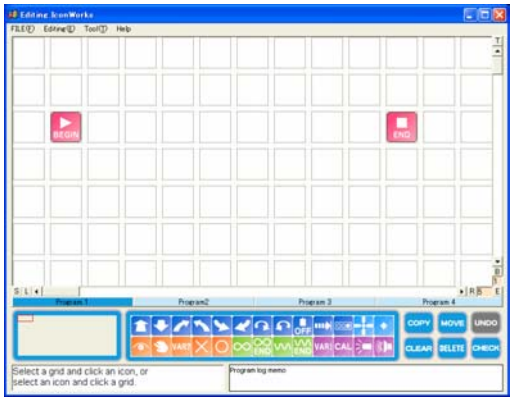
[2] LEARNING HOW TO USE THE PROGRAM EDITING SCREEN

TO START "CREATING A PROGRAM" ･･･THE EDITING SCREEN NEEDS TO BE ACTIVATED.

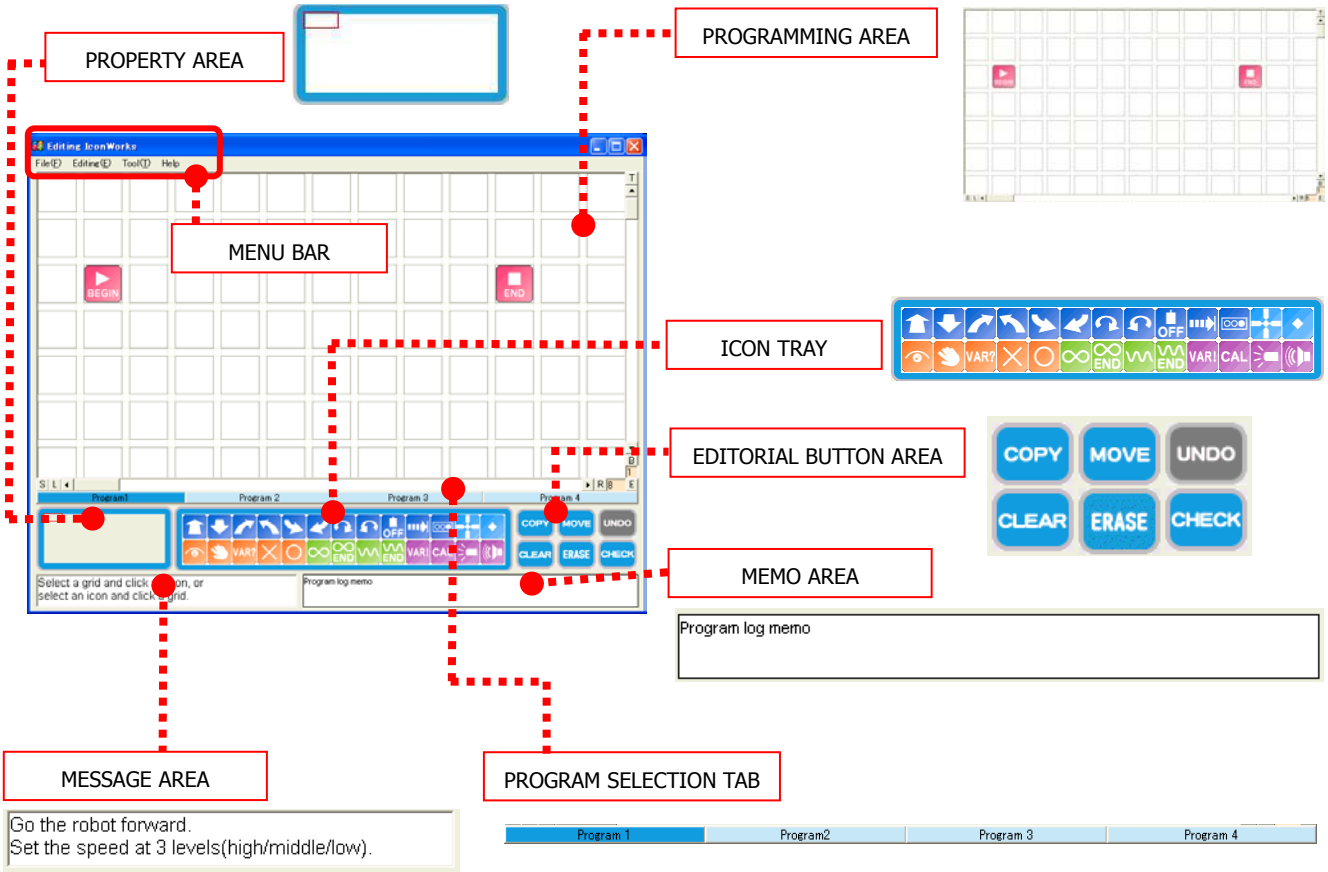
LEFT-CLICK THE "CREATING A PROGRAM" ICON.



THIS SCREEN WILL APPEAR.

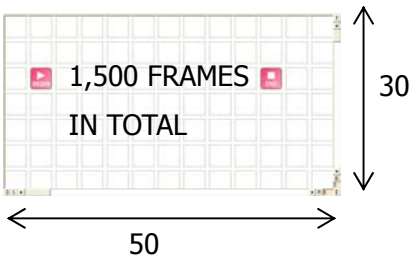


PROGRAM EDITOR SCREEN



INFORMATION ABOUT THE PROGRAM EDITOR SCREEN

PROGRAMMING AREA



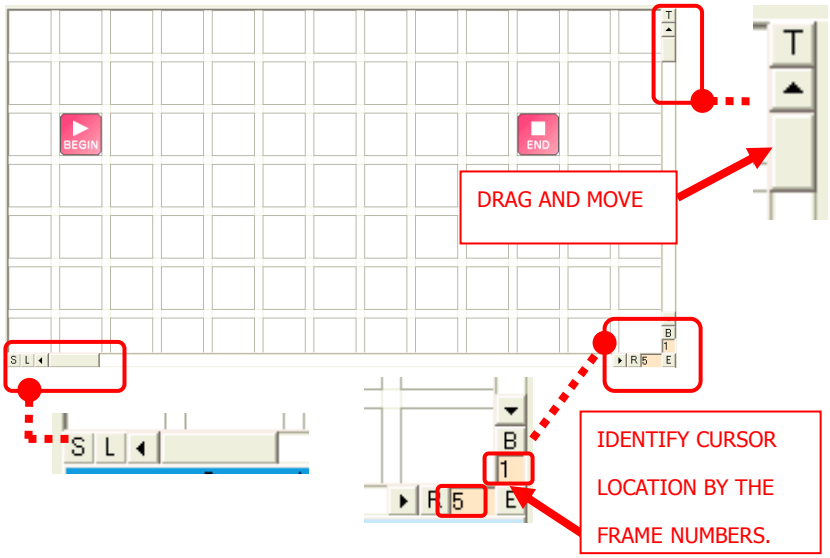
MAX. NUMBER OF ICONS: 100

This is the space available for creating a program.

A program can be created by arranging icons in this area.

In order to make placing icons easy in the programming area, grid lines are provided at constant distances.

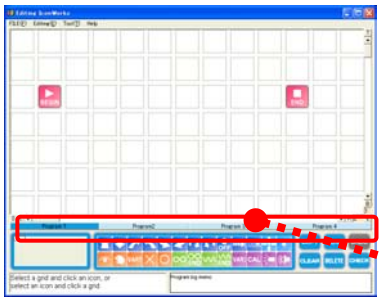
There are 1,500 frames in total, but the maximum number of icons that can be used for a program is 100.



TO DISPLAY HIDDEN AREAS, DRAG THE SCROLL BAR IN THE PROGRAMMING AREA (UP/DOWN, OR LEFT/RIGHT).

T	JUMP TO THE END OF THE SHEET IN THE DIRECTION OF THE ARROW.
▲	MOVE STEP BY STEP UP/DOWN IN THE DIRECTION OF THE ARROW.
S	JUMP TO THE TOP LEFT CORNER OF THE SHEET.
E	JUMP TO THE BOTTOM RIGHT CORNER OF THE SHEET.

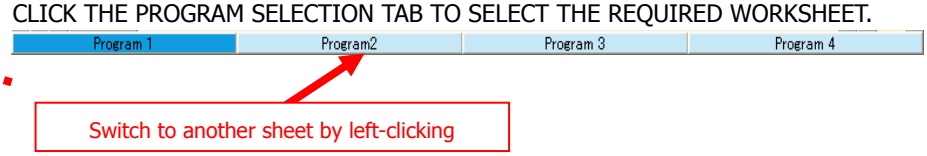
PROGRAM SELECTION TAB



THE PROGRAMMING AREA CONSISTS OF 4 SHEETS.

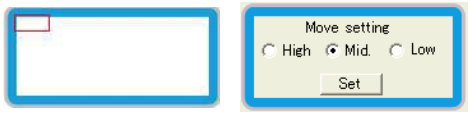
BY SWITCHING WORKSHEET TABS, 4 DIFFERENT PROGRAMS CAN BE EDITED CONCURRENTLY.

CLICK THE PROGRAM SELECTION TAB TO SELECT THE REQUIRED WORKSHEET.



In case of a saved program, its file name will be displayed on the associated tab.

PROPERTY AREA



* The displayed contents will change depending on the selected icon.

This area is to display the icon properties and to set the values of variables or conditions. When there is no information for set up in the icon, a "current area" frame will be displayed on the PROGRAMMING AREA.

The visible area being edited is displayed like this.

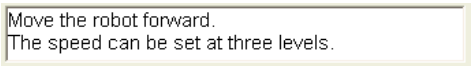
ICON TRAY

The command icons available for use in IconWorks are displayed in the icon tray. Place the mouse pointer over the selected icon and left-click.

EDITORIAL BUTTON AREA

These buttons can be used to edit and manage the icons already placed in the programming area.

MESSAGE AREA



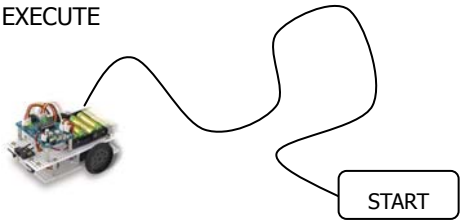
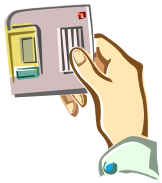
Messages explaining the icon's functions or set contents are displayed. Additionally, error messages and their causes are displayed if an error occurs during use.

MEMO BOOK AREA

This MEMO BOOK AREA is where you can write notes and save the programming record or any other information you want to keep. This information will be saved automatically when the program itself is saved.

[3] BASIC SOFTWARE OPERATION FLOW

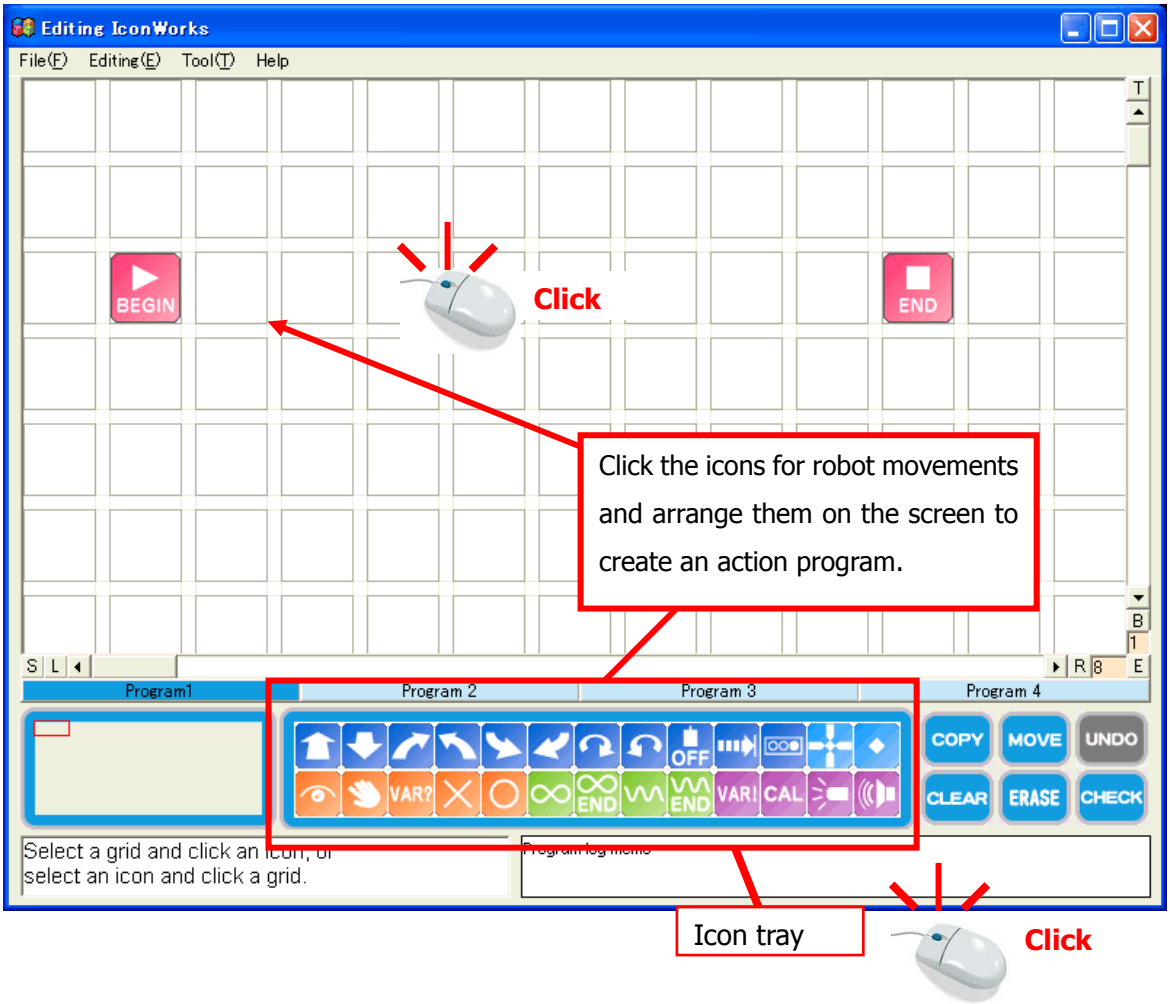
The flow from "programming" to " moving a robot" is described below.

OPEN 	Creates a new file or opens a file. Prepares a worksheet to write a program (PROGRAMMING AREA) by creating a new program or opening a program already available.
EDIT  	Creates and edits a program. Create a program by arranging and connecting icon(s), or edit the content of an existing program by reconnecting lines, erasing or copying.
HIGH-LEVEL EDIT  	Advanced programming Programming using sensors is a little more complex, but it is what you must learn if you want to be an "autonomous robot" expert. Here, you set or select parameters concerning when, where and how you want the robot to move.
DATA TRANSFER 	Sends the program to the robot. Uploads a program to the memory (microcomputer). Performs syntax-checking before transfer. Uses the program transfer cable (included) for sound communication.
EXECUTE 	Moves the robot. A transferred program can move the robot when executed. If the robot is not moving as you imagined, it may mean that the programming is incorrect. Check and adjust the program, and try again.
SAVE 	Saves a program so that it can be used again.

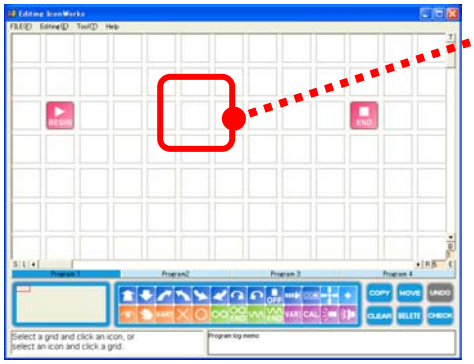
[4] BASIC OPERATION

Basic operation

Arrange icons on the screen using left-click and right-click accordingly.

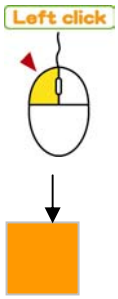


SELECTING THE GRID

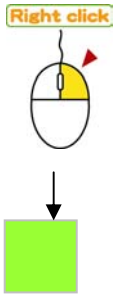


This frame will be used as a grid on which to place an icon.

Left-click the center of an empty grid.

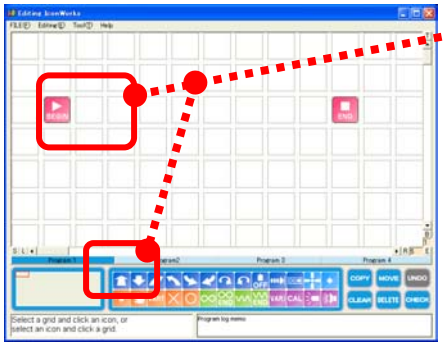


Or,



When the grid turns orange (flashing) or green, it is in a "SELECTED STATE".

PLACING ICONS



PUT A GRID IN THE SELECTED STATE.

			
Left-click the center of the grid.	The grid flashes an orange color	Left-click the icon on the icon tray.	An icon is arranged in the selected grid.

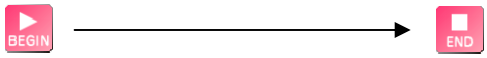
CONNECTING ICONS (CONNECTING PROGRAMS)

Connect icons with lines

The execution of a program starts with "BEGIN" towards "END" along the line.

If icons are not connected, the program is judged to be incomplete;

it therefore cannot be transferred to the robot.



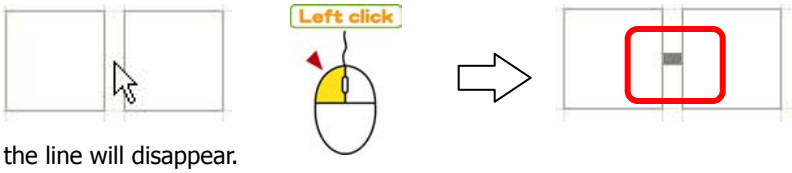
The program will be executed in the direction from BEGIN to END.

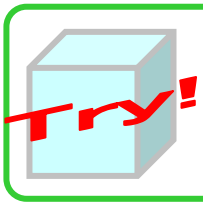
HOW TO DRAW A CONNECTING LINE

Put the cursor at the center between icons.

Left-click!

If you want to cancel connection, click again, then the line will disappear.



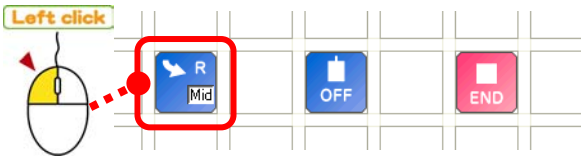


Let's review what you have learned up until now.

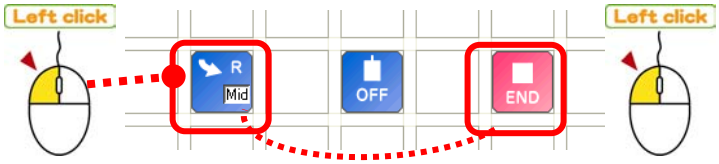
Place icons in any way you like and connect them with lines.

SELECTING ICONS (SELECTING A SINGLE ICON)

Select one



To select more than two, continue to left-click.

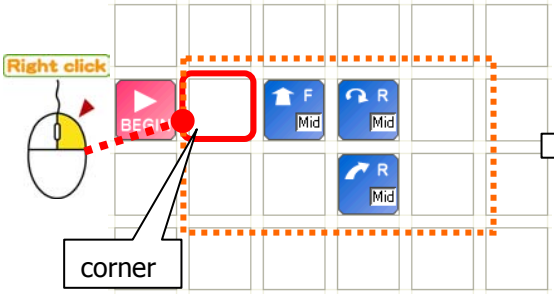


Left-click the icon placed in the programming area. The icon flashes and enters a "SELECTED STATE" condition.

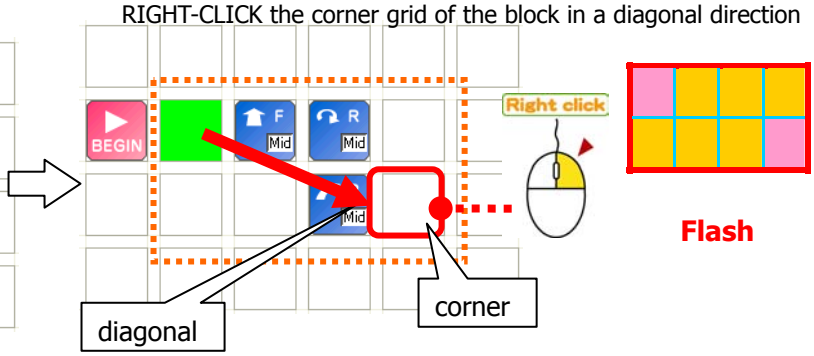
With one icon in a "SELECTED STATE", you can continue to select multiple icons by left-clicking.

SELECTING ICONS (SELECTING A BLOCK)

RIGHT-CLICK the corner grid of the block



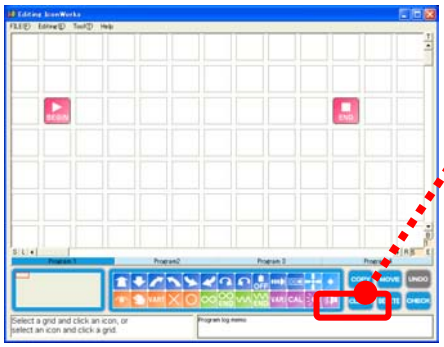
RIGHT-CLICK the corner grid of the block in a diagonal direction



This mode is for when you want to select an area in one action (Block Selection Mode).

In this mode, right-click the corner (or the top) of the block you want to select. Then, right-click the corner (or the bottom) on the diagonal line of the block you want to select. Then, the entire selected block area flashes orange and enters a "SELECTED STATE" condition.

CANCELLATION OF THE "SELECTED STATE" (flashing and lit-up conditions)

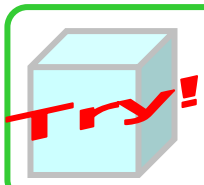


When you want to cancel the "SELECTED STATE".

Left-click "CLEAR" in the EDITORIAL BUTTON AREA.

Or, left-click once again on the flashing grid or icon.

When the "DESIGNATED BLOCK" mode is selected, cancellation is possible by "CLEAR" in the EDITORIAL BUTTON AREA.

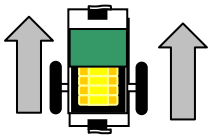
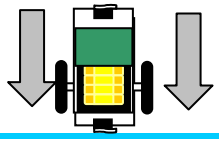
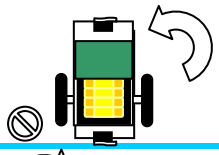
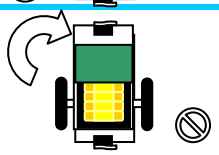
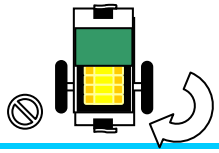
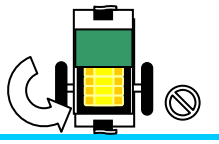
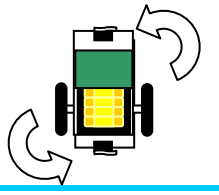


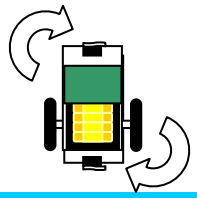
Let's review what you have learned so far.
Put the arranged icons in "SELECTED STATE" and then cancel them.

[5] ICON LIST

***Please refer to Page 32-38 for a detailed explanation of the functions.**

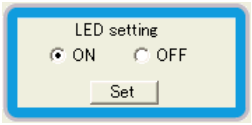
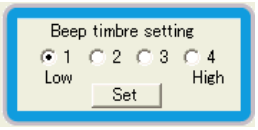
* Icon designs on the ICON TRAY are simplified.

	ICON NAME	FEATURES	PROPERTY	EXPLANATION OF PROPERTY
THE BEGINNING AND ENDING ICONS				
	BEGIN Beginning	This is the start point of a program. The program always begins with this icon (Execution always commences here.)	PROPERTY : NONE These icons are pre-arranged on the sheet. Deletion and copying is not possible but, it is possible to move them. There are no BEGIN and END icons on the ICON TRAY.	
	END Ending	This is the terminus of a program. The program ends with this icon.		
MOVEMENT ICONS				
	ICON NAME	FEATURES	ROBOT'S MOVEMENT	PROPERTY	EXPLANATION
	FORWARD	The robot moves forward.			The speed can be set at three levels (high, medium and low).
	BACKWARD	The robot moves backward.			
	PIVOT-TURN LEFT	The robot turns left forwards.			
	PIVOT-TURN RIGHT	The robot turns right forwards.			
	PIVOT-TURN LEFT BACKWARD	The robot turns left backward.			
	PIVOT-TURN RIGHT BACKWARD	The robot turns right backward.			
	SPIN-TURN COUNTER -CLOCKWISE (CCW)	The robot rotates CCW.			

R 00	PIVOT-TURN CLOCKWISE (CW)	The robot rotates CW.		Move setting High Mid. Low Set	The speed can be set at three levels (high, medium and low).
OFF	MOTOR OFF	It turns off the motor and halts movements such as forward or backward.		NONE	---
00	BRAKE	It applies a brake by short-circuiting the robot's motor. Use this when you want to make a sharp stop.		Brake setting (0.1-10) Time 0.5 Sec. Set	It sets the length of time a brake is applied and the motor stops. Can be set within the range of 0.1 - 10 seconds.
PROGRAM CONTROL ICONS					
00	WAIT	It waits for the specified time or waiting condition before moving onto the next command.		Wait setting (0.1-10) Time 0.5 Sec. Set	Input the length of waiting time. Can be set within the range of 0.1 - 10 seconds.
	LOOP START	It is the start of the LOOP command (endless repeat).		NONE	---
END	LOOP END	It is the terminal of the LOOP (endless repeat).		NONE	---
00	REPEAT	It repeats the program between REPEAT and REPEAT-END for a specified number of times. Must always be used in a pair with REPEAT END. Up to 7 multiple REPEATs are possible.		Repeat setting (0-255) Rep 1 Times Set	Input the number of repeats by numbers. Can be set within the range of 1-255.
END	REPEAT END	It is the terminal of the REPEAT.		NONE	---
00	LIGHT BRANCH	The optical sensor branches the program by On or Off.		Light sensor branch setting sensor1 sensor2 Set	Select Light Sensor 1 or 2.

	TOUCH BRANCH	The touch sensor branches the program by On or Off.	Touch sensor branch setting Sensor1 Sensor2 Set	Select Touch Sensor 1 or 2.
	VAR? Variable Branch	The program is branched off, depending whether or not the variable value is above the specified value. Variable branch (0-255) A >= 1 or Nu Set	Designate a variable for comparison or enter a specific value. Input any value which can be between 0-255.	
	YES/NO	If the conditions to a branch satisfy the criteria, the program branches to YES and, if not, to NO. Must always be connected after one of the conditional branch icons.	NONE	---
	MERGE	The branched-off programs must always merge.	NONE	---
	NOP (No Operation)	It takes no action for itself. Is used as a joint or spacer for isolated icons.	NONE	---

	ICON NAME	FEATURE	PROPERTY	EXPLANATION
VARIABLES CONTROL ICONS				
	VAR! Set Variable	It memorizes the specified variable value. Variable setting (0-255) A = 0 or Nu Set	Set a designated variable value or enter a specific value. Input any value which can be between 0-255.	
	CAL Calculate variables	Add, subtract, multiple or divide the present variable with the specified value or variable. The calculation result makes a new variable value for this icon. Calculation setting (0-255) A + 1 or Nu Set	Select the calculated variable and input value which can be used for calculation box. Input any value which can be between 0-255.	
OTHER ICONS				

	LED Set Led	Turns on and off the LED.		Select ON if you want to turn on the LED and OFF if off.
	BEEP Set Beep	A beep sound continues for 0.2 seconds.		Can be set at four levels between low and high-pitched tones.

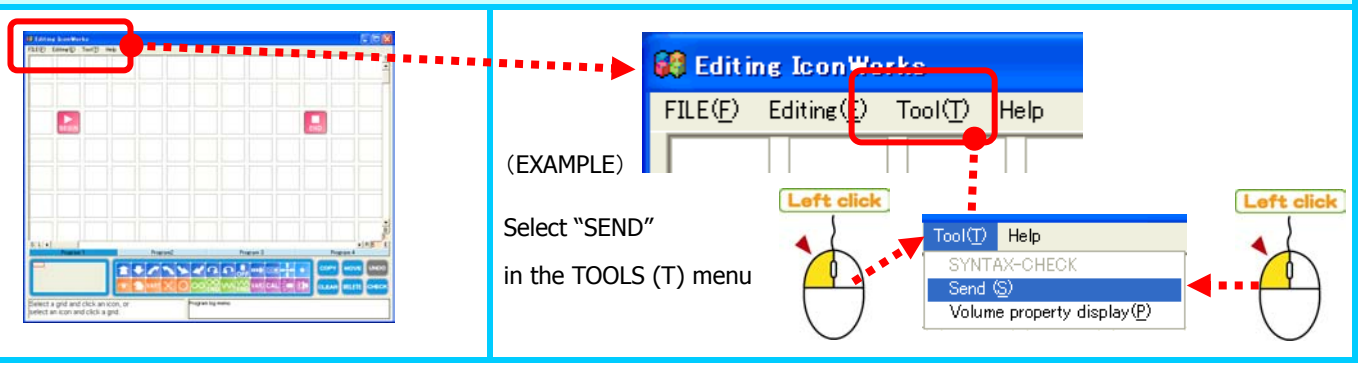
EDITORIAL BUTTONS			Reference page
	MOVE	Moves a selected icon to the designated grid.	29
	COPY	Copies a selected icon onto the designated grid.	29
	ERASE	Deletes a selected icon.	29
	CLEAR	Cancels the icon "selected state" on the edit-screen.	30
	UNDO	Returns the edit-screen to the situation before the last edit. (Undo an edit) This command cannot be used when it is displayed in a gray color.	30
 	SYNTAX-CHECK NEXT	Performs a syntax-check to verify whether or not the current program is correct. The syntax-check must be done three consecutive times. The button changes to "SEND" when all the three checks are positive.	30
	SEND	Sends a completed program to the robot. Becomes active when all syntax-checks are successful. Does not appear if a syntax-check is not successful.	31

III. BASIC FUNCTIONS

[1] MENU

The MENU BAR contains the commands that are available in IconWorks.

MENU BAR/SUB MENU FUNCTIONS



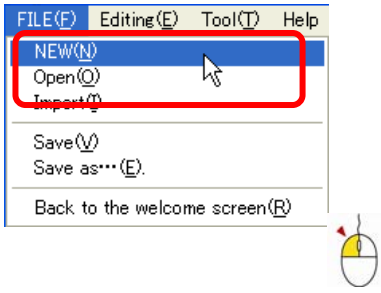
OVERVIEW OF THE MENUS

MENU	SUB MENU	CONTENTS	REFERENCE PAGE
FILE	NEW	Opens a new program area onto the present sheet.	26
	OPEN	Opens a saved file into the current sheet, deleting previous data.	26
	IMPORT	Imports a saved program into a selected area on the current sheet, previous data can be conserved if instructions are correctly followed.	27
	SAVE	Saves a program without changing the file name.	27
	SAVE AS …	Used when you want to save a newly created program or a current one but in a different name.	28
	RETURN	Ends a program edit screen and returns to the initial screen.	28
EDIT	ERASE	Deletes a selected icon.	29
	COPY	Copies a selected icon on to the specified grid.	29
	MOVE	Moves a selected icon to the specified grid.	29
	UNDO	Returns an edit-display to the previous one (cancels the last edit action).	30
	CLEAR	Cancels the selection of the icon/s in the edit-display.	30
TOOL	SYNTAX-CHECK	Verifies the data to confirm the program is complete. The program cannot be sent if incomplete. The user must perform a syntax-check before sending.	30
	SEND	Sends a created program to the robot (Only visible after syntax-check).	31
	VOLUME PROPERTY DISPLAY	Displays the sound control panel to allow modification of the settings to ensure sounds are emitted from the PC.	31

FILE

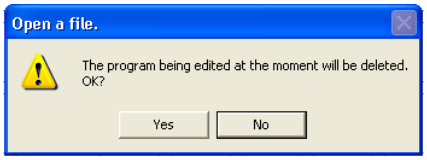
NEW (N) …Opens a new blank program area on the active sheet.

1) "FILE(F)" =>"NEW"(N)

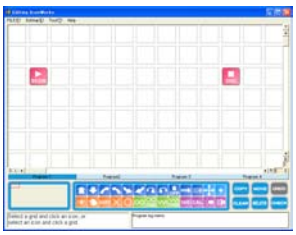


2) When opening a new file on the sheet, the following question is asked; "Any program being edited in the current program area will be overwritten. OK?"

Press (Y) after checking there is no risk to your data.

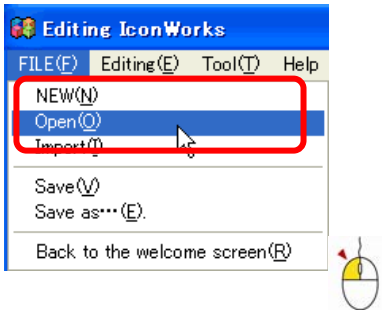


A new sheet will appear.

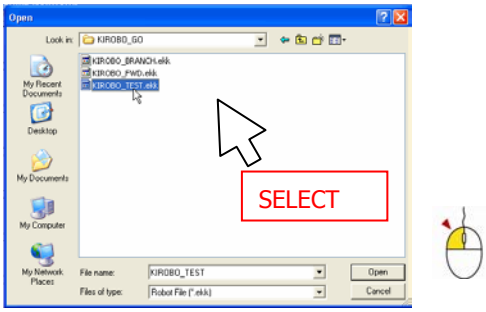


OPEN(O) …Opens a saved file into the active sheet.

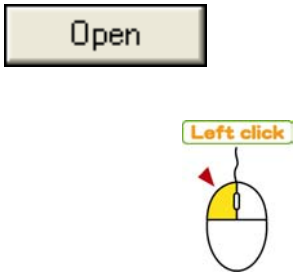
1) "FILE(F)" => "OPEN"(O)



2) Select the file on the selection screen.

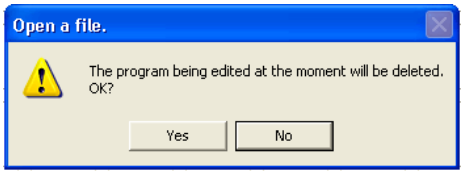


3) OPEN (O).



When opening a new file on the sheet being used, a confirmation box appears.

Select "Yes(Y)" if it is safe to overwrite the existing data or program.

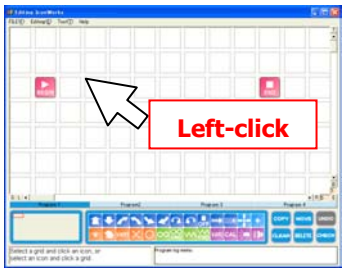


IMPORT(I): You can load a saved program into the active program area...

In IconWorks you cannot make a copy between worksheets like you can in the case of the Microsoft's EXCEL software.

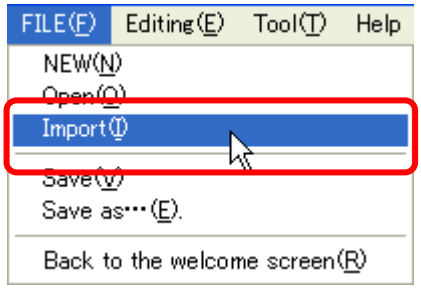
Instead, by saving the program once and importing into the active programming area, you can make a copy.

1) Left-click the area into which you want to insert the program.

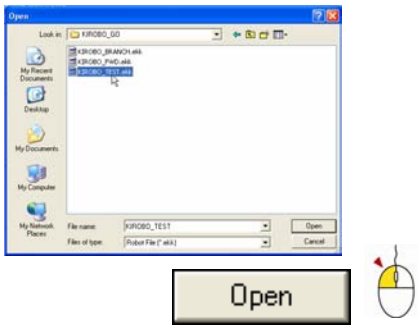


*Always click first!

2) "FILE" => "INSERT"(I)



3) Select "FILE" on the file selection screen and click "OPEN"(O)



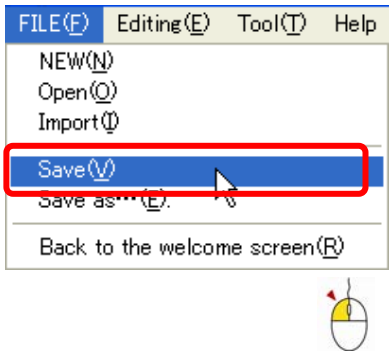
4) The imported program will be attached down and to the right, automatically filling the insertion area.

(Note)

The imported program cannot overwrite the currently created program when it overlaps.

SAVE(V)...The program being edited will be saved without changing the file name.

"FILE"(F) => "SAVE"(V)



* Saves the file which is currently being edited using its current file name.

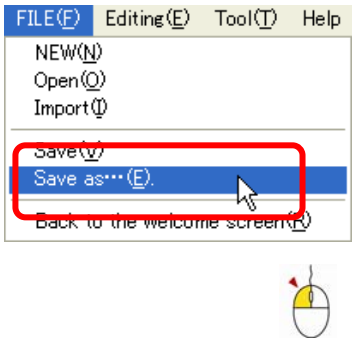
* A file which has never been saved cannot be overwritten.

  In such a case, use "SAVE AS" (E) command and enter a file name.

* The following message will appear in the case of an unsaved file.

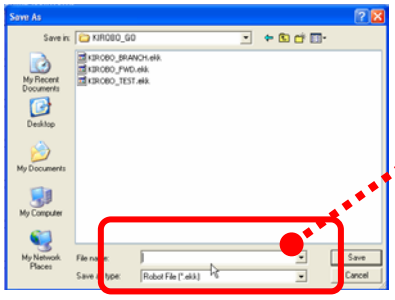
SAVE AS(E)…when you want to save a program for the first time or using a different name

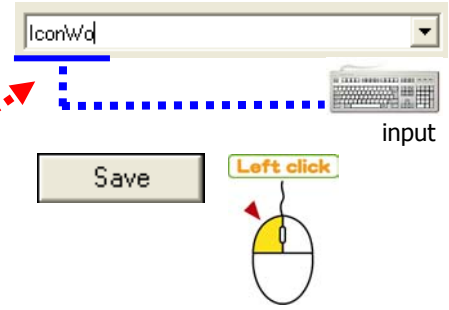
1) "FILE"(F) => "SAVE AS"(E)



2) Input the file name into the window which appears and press "SAVE"(S).

* An extension is automatically added; there is no need to enter it.

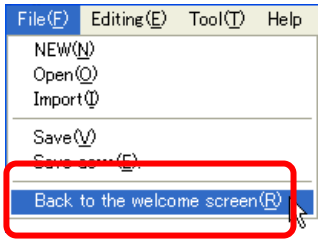




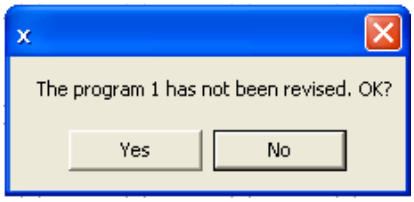
RETURN TO THE WELCOME SCREEN ··· Exit the program editing screen and return to the welcome screen.

When you want to finish editing the program, please follow the following procedure, or click the "CLOSE" button on the window.

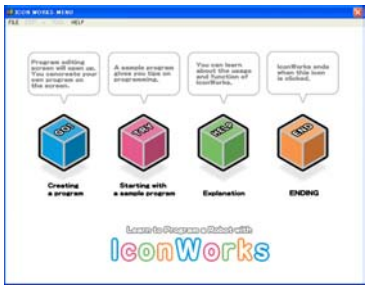
"FILE"(F)=>"RETURN TO THE WELCOME SCREEN"(R)



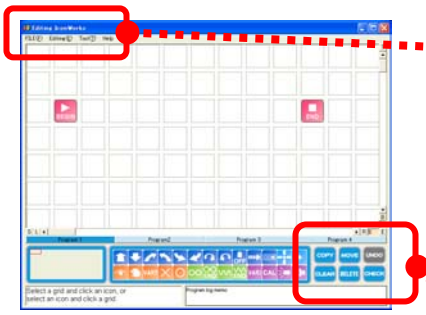
If there is an unsaved program, this message will be displayed. Please take the required actions if you wish to save it.



You will then return to this screen.

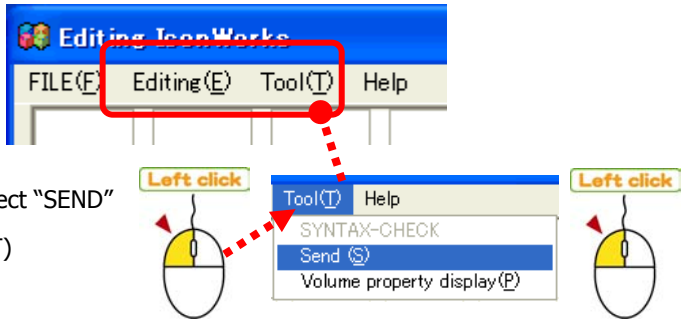


EDITORIAL BUTTON FUNCTIONS



(EXAMPLE)

When you select "SEND" in the TOOL(T)

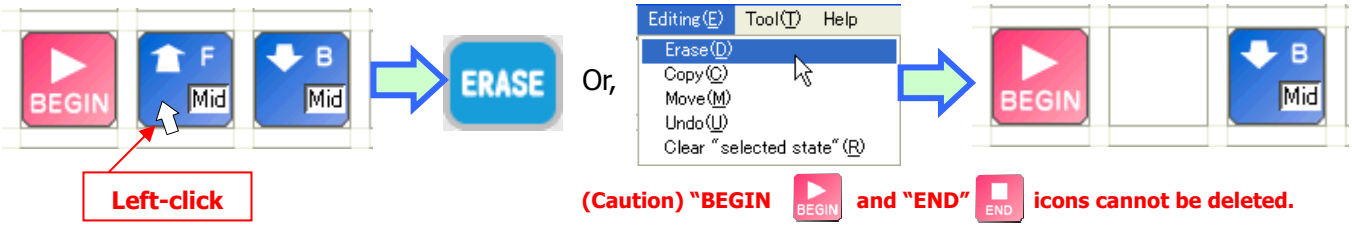


EDITING

ERASE … Delete the selected icon.

Put the icon you want to erase into the "SELECTED STATE" (blinking). (*Refer to Page 18.)

Click "EDIT" =>"ERASE". Or, Left-click "ERASE" in the "EDIT" tray.



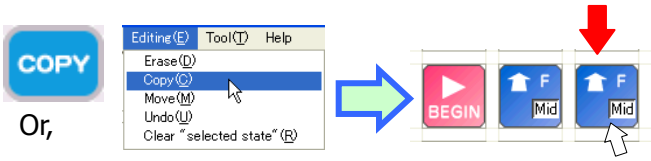
(Caution) "BEGIN **and "END"** **icons cannot be deleted.**

COPY … Copy the selected icon into the designated grid.

Put the icon you want to copy in the "SELECTED STATE".

Right-click the grid to copy to.

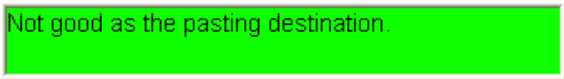
"EDIT"=>"COPY" or Left-click "COPY" in the "EDIT" tray



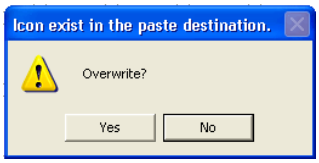
(Caution)

"BEGIN"  **or "END"**  **icons cannot be copied on.**

If you try to copy where it is not permitted, the following message will appear. Re-select the copy location and try the copy operation again.



* When there is already an icon in the designated area, the message "OVER-WRITE?" will appear.

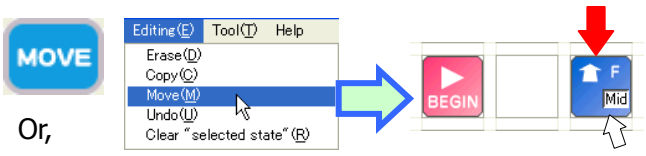


MOVE … Move the icon to the designated grid.

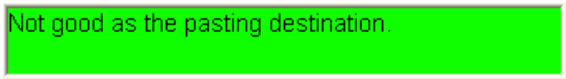
Put the icon you want to move into the "SELECTED STATE".

Right-click the grid to move to.

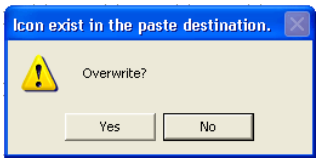
"EDIT"=>"MOVE" or left-click "MOVE" in the "EDIT" tray

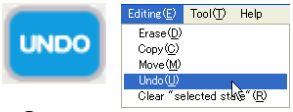


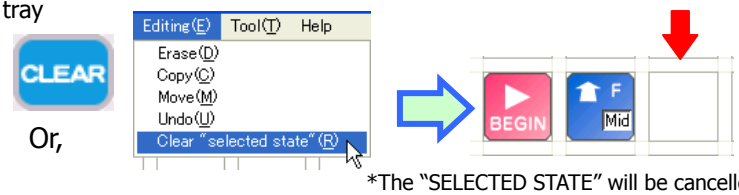
If you try to move where it is not permitted, the following message will appear. Re-select the move location and try the move operation again.

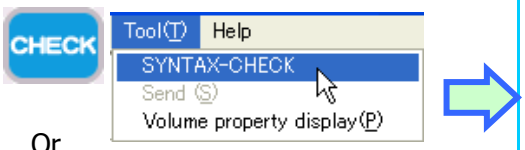
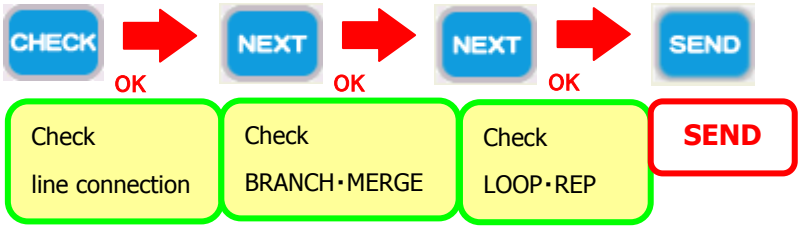


* When there is already an icon in the designated area, the message "OVERWRITE?" will appear.



UNDO … Returning to the preceding "EDIT" screen	
Sometimes you cannot use the "UNDO" command, in which case a gray icon is displayed.  *When this is displayed, you cannot return.	"EDIT" =>"UNDO LAST ACTION", or left-click "UNDO" in the "EDIT" tray  Or, the preceding screen

CLEAR … Clear the selection… cancel the selection of icons on the "EDIT" screen.	
When the icons or grids are in the "SELECTED STATE" (blinking) 	"EDIT"=>" CLEAR SELECTION", or left-click "CLEAR"(R) in the "EDIT" tray  Or, *The "SELECTED STATE" will be cancelled.

TOOL	
SYNTAX-CHECK … The program can be transferred to the robot when complete. The program cannot be sent if it is incomplete. A syntax-check is always performed on the program before sending.	
"EDIT" => "SYNTAX CHECK", or left-click "CHECK" in the "EDIT" tray when a program has been created.  Or,	The SYNTAX CHECK is done in three consecutive steps. It changes to "SEND" only when all the steps have been completed sucessfully. 

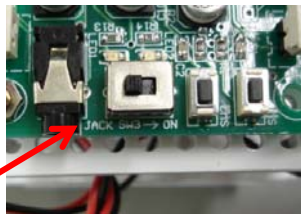
Check line connection	Confirms if the lines are correctly connected and identifies any connection error by flashing.	
	MESSAGE EXAMPLE	1 BEGIN icon 1 END icon The flashing icons are not connected. 1 BEGIN icon 1 END icon The icon connections are OK.
Check BRANCH・MERGE	Checks to make sure branching and merging sequences are correct.	
	MESSAGE EXAMPLE	BRANCH is OK.MERGE is OK. OK. Click the button again and go on to NEXT.
Check LOOP・REP	Checks if repeat icons such as LOOP and REP are in a pair.	
	MESSAGE EXAMPLE	Error in LOOP number 2,Error in REP number 1, REP and REP END not in pairs. LOOP is OK.REP is OK. OK. Send the program.
Warning indicators when errors are found.	If the syntax is not correct, the incorrect icon and related icon(s) flash and the content of the error is displayed in the message area. Confirm the reason for the error and make the required corrections… then re-run the SYNTAX CHECK.	

[2] PROGRAM TRANSFER

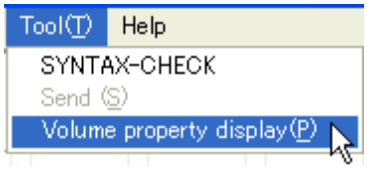
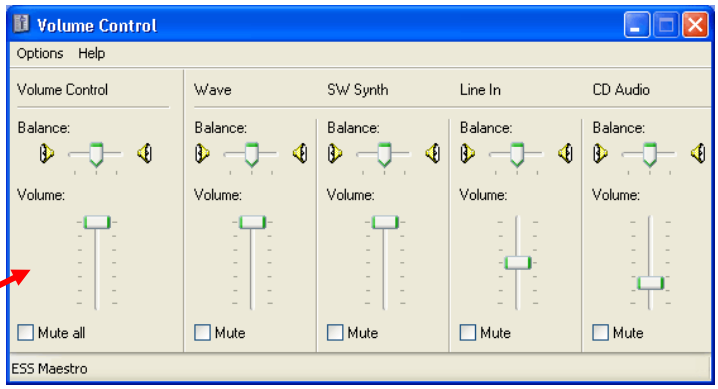
When the SYNTAX-CHECK is completed and the "SEND" button is displayed, confirm if the cable is connected and send the program data to the robot.

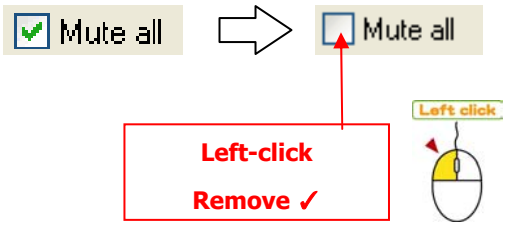
Below are the preparations needed for the data transfer.

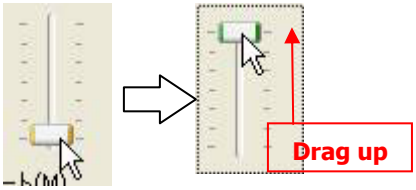
* Preparing for the transfer

SEND …Insert the program transfer cable to send the program from PC to the robot.		
	ON THE PC SIDE	ON THE KIROBO SIDE
What is needed for for the transfer?	A program transfer cable (bundled with the kit). 	KIROBO 
INSERTING THE CABLE Insert the program transfer cable into the headphone jack(plug) of the PC and the jack(plug) on the robot side.	This is the sign of an headphone terminal.  	The jack (plug) on the motherboard 

* Preparing for the sound communication

ADJUSTMENT OF VOLUME PROPERTIES …IconWorks uses sound for data transfer. Check if sound can be generated from the PC.	
Data cannot be transferred if the volume is 0 or set mute, or the WAVE is 0 or set mute. Check and adjust the volume condition in the volume property panel on the PC. It is recommended to set the master volume and WAVE volume to be above the center in the slide bar.	
VOLUME PROPERTY DISPLAY … Displays the screen for adjusting the volume on the PC before sending the program.	
"TOOL"(T) =>"VOLUME PROPERTY DISPLAY"(P)  Volume control panel will appear.	





If the slider is low, drag it up.

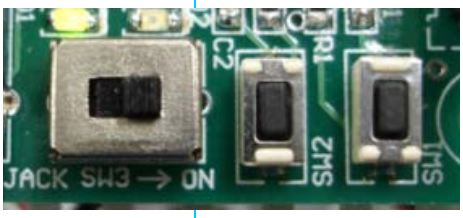
Close the window of PROPERTY when confirmation or change is done.

PREPARING THE ROBOT TO RECEIVE THE PROGRAM

Make sure once again that the cable is firmly inserted and then transfer using the following procedure.

1) Switch on the robot power supply.

2) Press Switch 2.



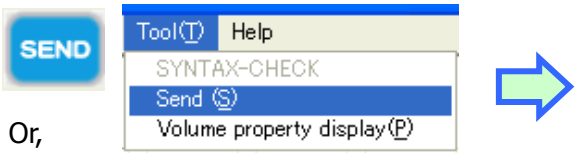
3) Confirm the robot's LED2 turns ON and is in the "STAND-BY" mode.

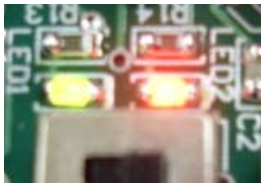
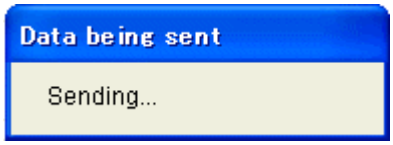
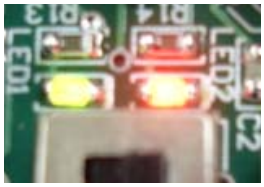
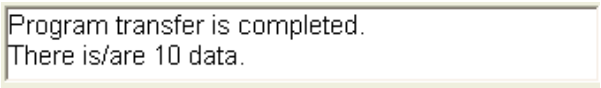
STARTING THE TRANSFER…Transfer the program you created to the robot.

When the SYNTAX-CHECK is complete, the "NEXT" button changes to "SEND". The program is ready for transfer. (You can click "SEND" from "TOOLS" in the "MENU BAR".)

When the SYNTAX-CHECK is incomplete, "SEND" will not appear.

"TOOLS"=> "SEND", or left-click "SEND" in the "EDIT" tray



ON THE KIROBO SIDE	ON THE PC SIDE
5) The LED2 flashes quickly during data transfer. 	Shows "Sending" 
6) When the transfer has been completed successfully, a sound "Pi-Pi-" is emitted. 	The following message appears when the transfer has been completed. 

CHECK THE FOLLOWING WHEN THE TRANSFER FAILS.

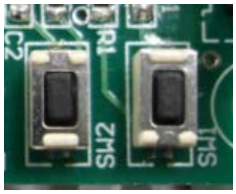
- Whent the transfer fails, an error sound "Piiiiiiiiii" is emitted and the LED2 starts flashing.
  In this case, check the volume property and connection of the program transfer cable again.

[3] PROGRAM EXECUTION

When the program is transferred successfully, move the robot.

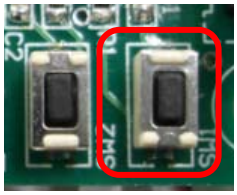
EXECUTE THE PROGRAM

1) When the transfer has been successfully completed, press SW1 again to execute the sent program.

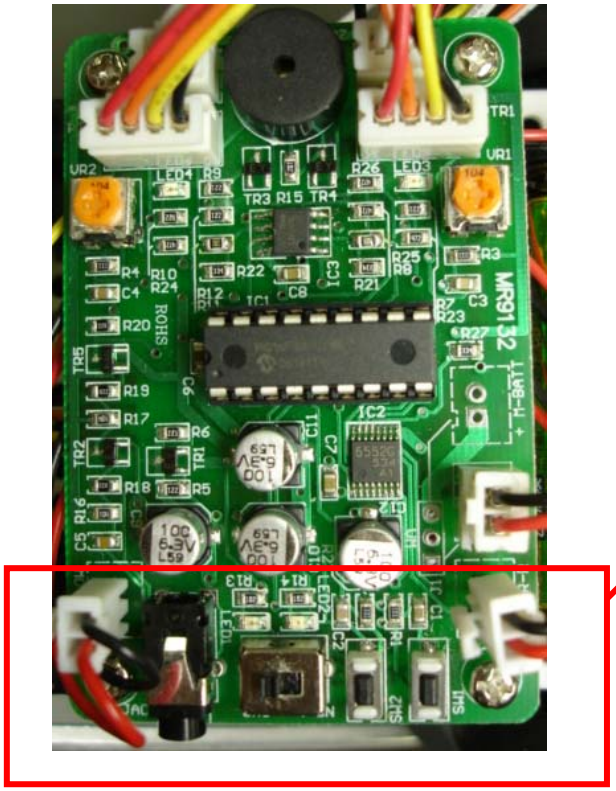


STOPPING THE ROBOT

1) LED2 flashes while the program is executed. You can stop the robot by pressing SW1 during program execution.



2) Depending on the program created, the robot may continue to operate even if the execution of the program is complete. In this case the robot continues even if LED2 is no longer flashing. To stop the robot, simply turn off the power.



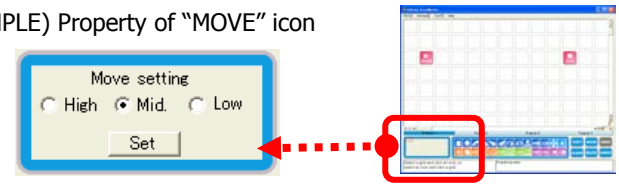
KIROBO motherboard

The names and functions of each part on the motherboard are described on Page 6 of KIROBO USER'S MANUAL I.

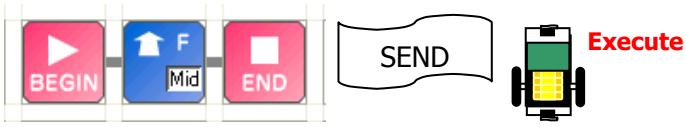
The buttons used for program transfer and execution are located here.

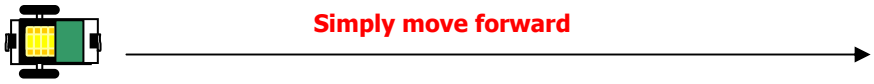
IV. BASIC OPERATION PRACTICE

[1] THE ICON FUNCTIONS AND PROPERTY SETTING PRACTICE

SETTING A PROPERTY	
There are some icons which allow the setting of additional properties. Left-click the desired icon and the current setting will be displayed in the property area. Enter or select your desired value and confirm by "SET".	(EXAMPLE) Property of "MOVE" icon 

Let's practice the basic movements.
Write the program as shown on the right-hand side, transfer and execute it on the robot.



 How did the robot move?	 Normally, the robot should move forward without stopping. The command "FORWARD" was made, but there was no command "STOP". It may seem as if it should stop because of the "END" icon; however, it is only the program which ends, but the command "FORWARD" has not been stopped.

<KIROBO: Forward or Straight ?>

When KIROBO is instructed to move forward it may not travel in a straight line, the following section explains why.
KIROBO is built with two motors to enable it to maneuver. These motors, although similar, are not exactly identical. Small differences in the manufacturing process can result in a range of slightly different characteristics. This range, called a 'tolerance', affects the speed of rotation in the motors. The target spindle speed maybe 3000rpm but it can vary between motors and be different depending on the direction of rotation. When one motor spins faster than the other it makes the wheel turn faster. This makes one side travel further in the same time causing KIROBO to follow a curved path. Other factors may affect the severity of the deviation, including the assembly of KIROBO'S chassis.
Given these factors there will almost certainly be some element of curvature in the forward or reverse paths taken by KIROBO. To compensate for this behavior we suggest that you write a program to correct KIROBO'S direction.
More complex robots compensate for these factors in a number of ways, including using 'paired' motors (motors which have identical characteristics), measuring the motor speed and adjusting the power while in use and GPS, LASER guides etc connected to steering systems. See the program included in the manual which guides KIROBO towards a light and develop your understanding of these techniques.

How can the robot be made to stop?

Execute a "STOP" command. There are 2 command icons for "STOP", but we will use "MOTOR OFF" first.

After the program is created, transfer and execute it.

	MOTOR OFF Turns off the motor power and ends the "FORWARD" or BACKWARD" movement commands.	PROPERTY : NONE

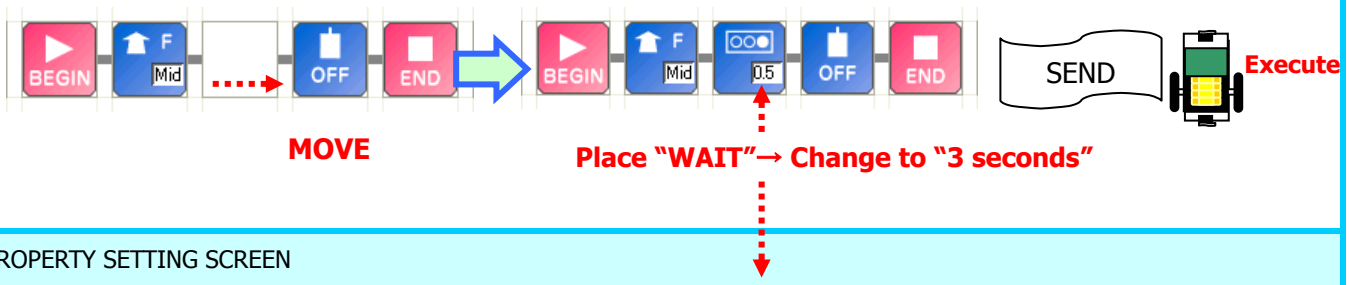
 How did the robot move this time?	 This time the robot should not move. However, the program has been executed (since LED2 flashes once). This is because the commands "FORWARD" and "MOTOR OFF" are processed instantaneously. You need a command icon to preserve the "FORWARD" status for a while.

For this, we use the "WAIT" icon.

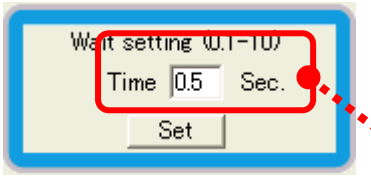
This icon continues the previous command for the specified time.

Preserve the condition before "WAIT"

Please prepare the following program, send and execute.

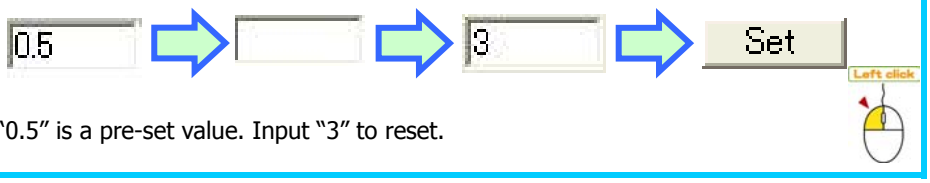


PROPERTY SETTING SCREEN



Input the length of waiting time. Can be set within the range of 0.1 - 10 seconds.

This is how you change.

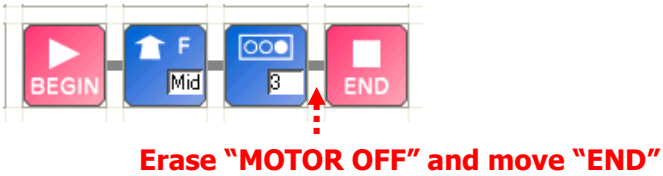


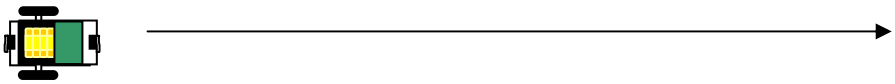
"0.5" is a pre-set value. Input "3" to reset.

 How did the robot move?	 The robot should move forward and stop about 3 seconds later. Let's try to remake this program but, this time not including "MOTOR OFF".

Prepare the following program, send and execute.

Erase "MOTOR OFF" and move "END".

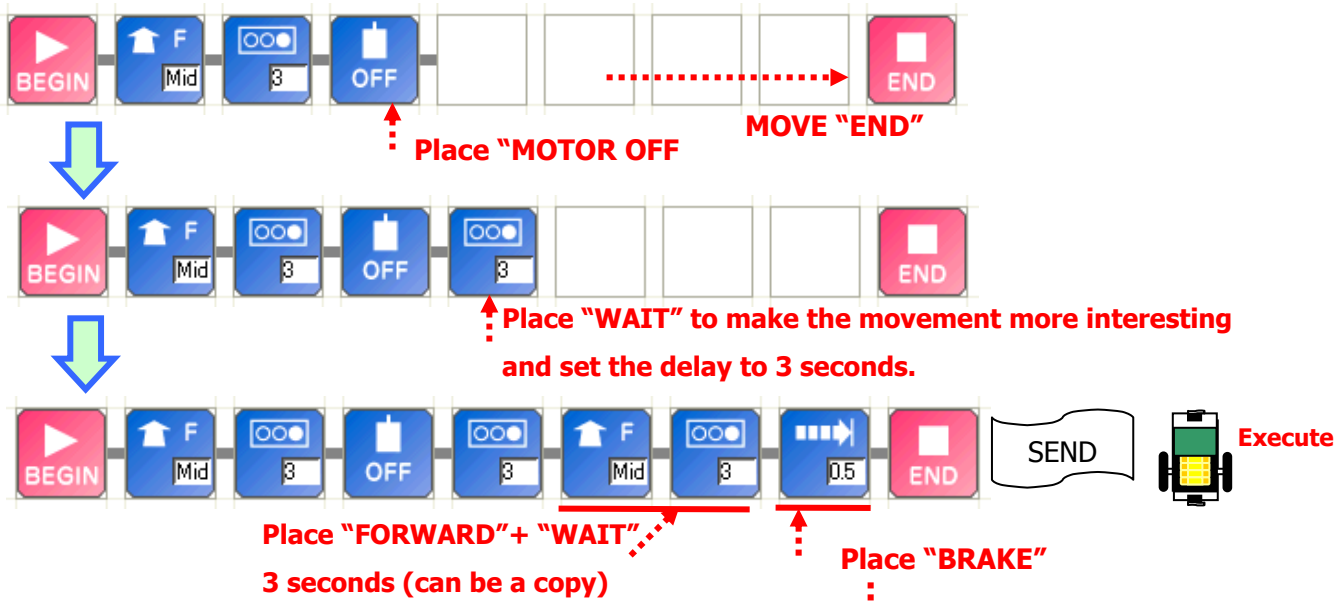


 How did the robot move?	 The robot should have only moved forward as it did in the first trial. This is because, like when only the "FORWARD" icon was placed, "FORWARD" is kept ON but there has been no OFF command to instruct the robot to "STOP". So, please remember to place the "MOTOR OFF" icon after movement commands to make sure the action is terminated.

As explained, there are 2 icons by which you can stop movements. What are the differences between them?

Compare the commands. (1) "MOTOR OFF" (2) "BRAKE"

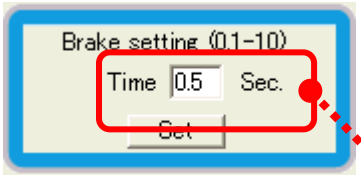
Make the following program, send and execute.



BRAKE

Applies a brake by short-circuiting the robot's motor. Use it when you want to make a sharp stop.

PROPERTY SETTING SCREEN

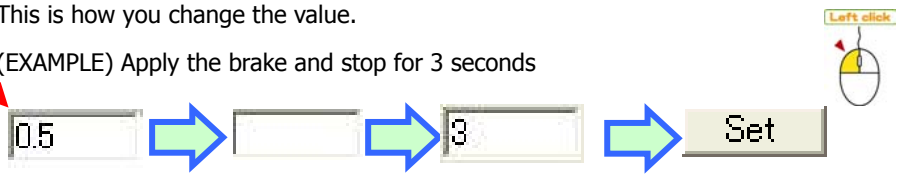


It sets the length of time a brake is applied and the motor stops.

Can be set within the range of 0.1 - 10 seconds.

This is how you change the value.

(EXAMPLE) Apply the brake and stop for 3 seconds



How did the robot move this time?

"FORWARD" 3 seconds → Stop by "MOTOR OFF" – "FORWARD" 3 seconds → Stop by "BRAKE"

Have you noticed how the robot stops differently between the movement 1 and movement 2?

In the first run, it is only that the power supply to the motor has been stopped, so the robot moves a little bit by inertia. However, in the second run, the motor is forced to stop therefore the robot makes a sharp stop.

Can you tell the difference? Select the icon you use in accordance with how you want the robot to move.

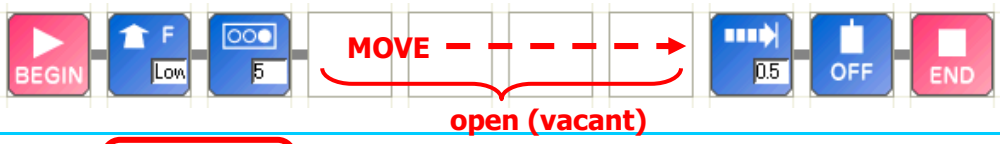
Practice the property setting of "MOVE" icons.

LET'S CREATE THE FOLLOWING PROGRAM, REVIEW WHAT YOU HAVE LEARNED SO FAR AND PLACE THE ICONS.

Open a sample program Step1-1.

* Refer to USER'S MANUAL I, Page 4.

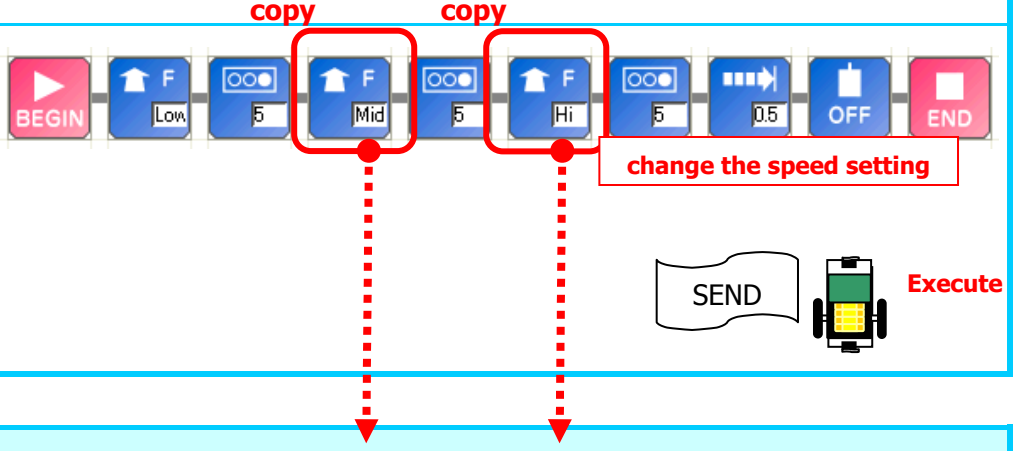
Use "MOVE" and create 4 open (vacant) frame areas.



Use "COPY" and layout the icons as illustrated on the right-hand side.

Use "MOVE INFORMATION" in the "PROPERTY" window. Edit the icons to have the properties "low", "medium" and "high" as shown.

When completed, send and execute the program.



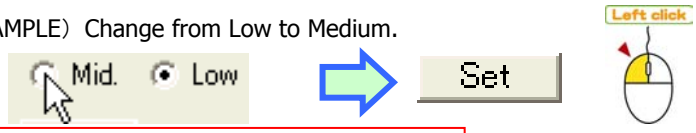
PROPERTY SETTING SCREEN

The speed can be set at 3 levels; High, Medium and Low.

Only 1 speed can be set per icon.

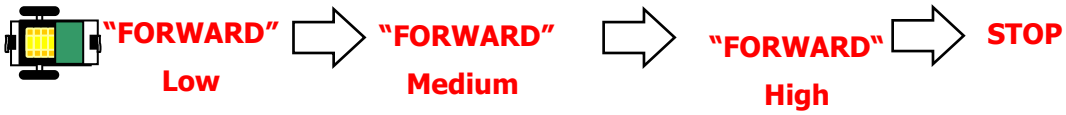
How to change the setting

(EXAMPLE) Change from Low to Medium.



Left-click the radio button of "Low"

How did the robot move?



In this program, the speed changes every 5 seconds, from low to medium, then to high.

Did you notice the speed change?

You may wish to edit these properties in accordance with how you want the robot to move.

If you want to repeat the same movement, how can it be done? Now, we will learn about using "REPEAT" icon.

To start with, please prepare the following program, send and execute.

"START"-"REPEAT"(once)-"FORWARD"-"WAIT"(1second)-"BEEP" (timbre 3)]-"MOTOR OFF"-"END"

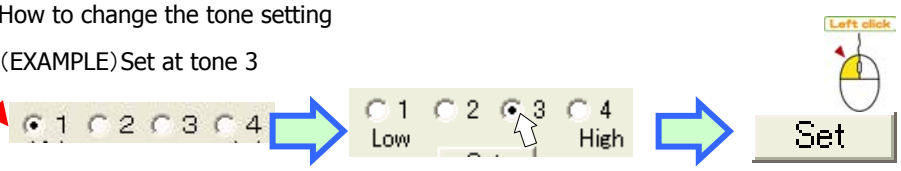
	BEEP	The BEEP sound continues for 0.2 seconds/icon.

PROPERTY SETTING SCREEN

Tones can be set at 4 levels, from low to high.

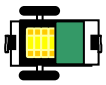
How to change the tone setting

(EXAMPLE) Set at tone 3



Pre-set at 1. Change to 3.

How did the robot move?

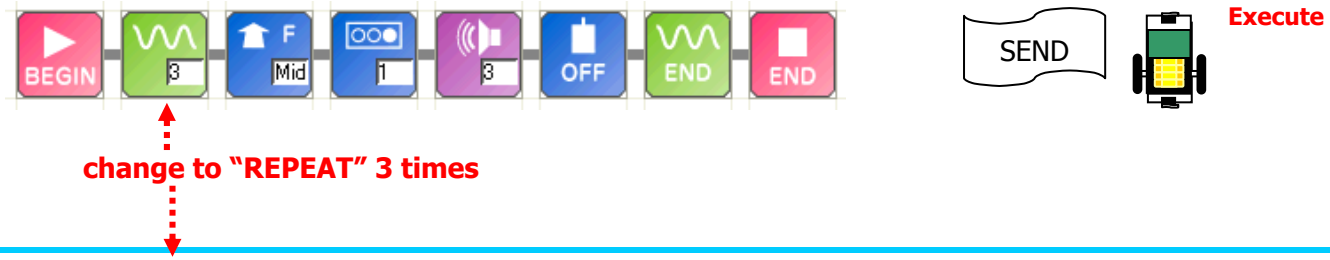


"FORWARD" 1 second → **BEEP** → **STOP**

Moves forward for a second, beeps and stops.

Please prepare the following program, send and execute.

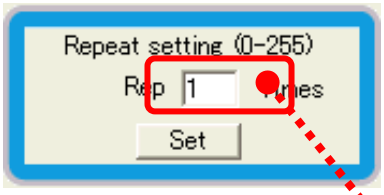
"BEGIN"-"REPEAT"(3 times)-"FORWARD"-"WAIT"(1second)-"BEEP" (timbre 3)]-"MOTOR OFF"-"END"



See next page for how to set the property.

	The program repeats between REPEAT and REPEAT-END for a specified number of times. It must always be used in a pair with REPEAT END. Up to 7 multiple REPEATs are possible.		

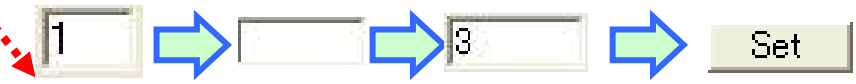
PROPERTY SETTING SCREEN



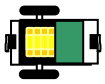
Enter the number of repeats using numbers. Can be set within the range of 1-255.
*Only real numbers can be used (no fractions).

How to change the setting

(EXAMPLE)Repeat the movement from  to  3 times.



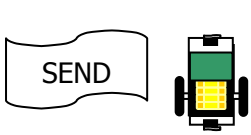
The pre-set value is 1. Change it to 3.

 How did the robot move?	 **"FORWARD" 1 second** ⇨ **"BEEP"** ⇨ **"STOP"** Repeat "Move forward for a second - beep – stop" 3 times and stop. **Repeats 3 times the same sequence of movement**

If you want to just continue repeating a program endlessly, how can it be done?

Create a program as follows, send and execute it.

"BEGIN" – "LOOP"- "FORWARD" – "WAIT" (1 second) – "BEEP" (timbre 3) – "MOTOR OFF" – "LOOP END" – "END"



	LOOP: Infinitely repeat the program between "LOOP" and "LOOP-END". Always in a pair with "LOOP END". The multiple loop command can be used up to 7 times.		LOOP END: The terminal of the "LOOP"

 How did the robot move?	 **"FORWARD" 1 second** ⇨ **"BEEP"** ⇨ **"STOP"** Endlessly repeats "FORWARD 1 second – BEEP – STOP". **Continues this sequence until SW1 is pressed.**

Example of a program to adjust the running direction of KIROBO (to enable KIROBO run straight)

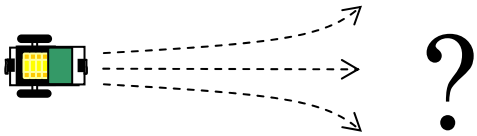
KIROBO has 2 separate geared motors to rotate the tires on the right and left sides. Even though 2 motors look the same, the motor specification is not always exactly the same, which causes KIROBO to run more or less curved.

Therefore, we recommend users to develop a program with the exclusive software Icon Works for correction of running direction of KIROBO so that it runs straight.

The following program is an example. Please try by yourself how it goes and also challenge developing other programs of your own.

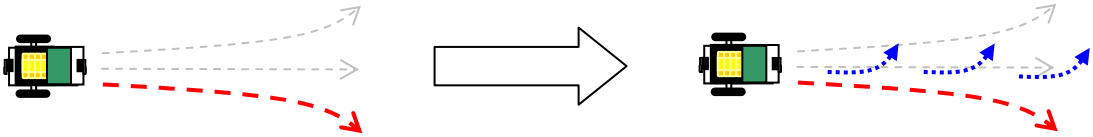
<1> Check how your KIROBO runs!

First, let's see how straight your KIROBO runs!



In which direction does it run? Check by using a program indicated below.

(EX) When KIROBO runs gradually to right



Change the program to correct the running course to left little by little.

<Program Example>

[Explanation]

① Add TURN LEFT icon before FORWARD icon.

* When your KIROBO gradually runs to the left, then add TURN RIGHT icon instead of TURN LEFT icon.

② Repeat FORWARD movement as shown above by setting the number to repeat in accordance with the running direction of your KIROBO. Adjustment is also possible by changing the turning speed.

③ Insert the above program in the main program to the location where you want your KIROBO run straight.

A program to correct the running direction depends all upon how your KIROBO moves in default. So, please refer to the above program and the explanation and make a program appropriate for your KIROBO.

[2] PROGRAM AND FLOW CHART

A PROGRAM THAT BRANCHES

When making a program, it is often recommended to “draw a flow chart”. What is a flowchart?

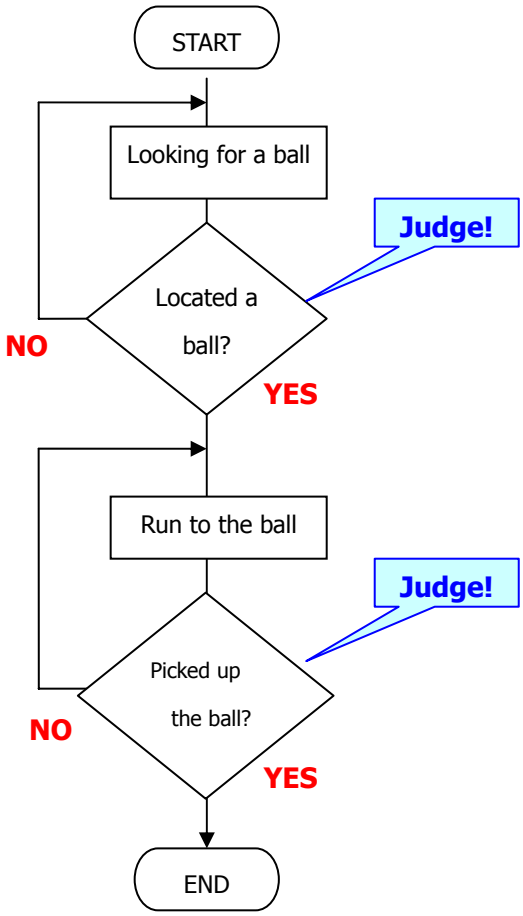
A flow chart is used to clearly express a series of procedures when designing or correcting a program. The flow chart is helpful to explain your ideas or organize your thoughts. There are various ways to describe the flow chart. In this manual, the symbols shown on the right-hand side will be used for the sake of simplicity.
When you draw a flow chart, these symbols will be used in combination along the program flow.

	terminal	Attached to both the beginning and end of the program.
	process	Corresponds to each of the computer processing actions.
	judge/ decide	Decides which course to select depending on a condition
	combine	Shows the exit or an entrance to another place on the flow chart.
↓	←	Shows the direction of the processing flow.

Let us illustrate by the flow chart how KIROBO moves.
"KIROBO is looking for a soccer ball. If it finds a ball, it runs to the ball and picks it up.”
The above can be translated into the flow chart on the right.

When an autonomous robot moves, it detects its environment using the sensors and judges what action it should take next.
In KIROBO’s case, it will be like this.
In response to the question "Have you located a ball?”
=> In case of "Yes", "It runs to the ball".
=> In case of "No", "It looks for the ball".
So, the next action depends on the result of a given condition.
Likewise, the program must be made for the robot to make a judgment depending on the condition, such as "Do xxx in case of yyy and zzz if not".

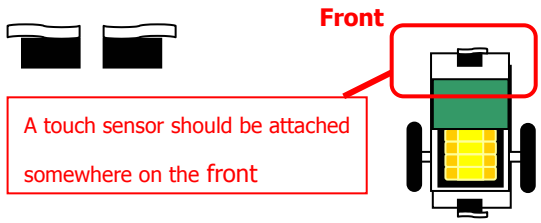
Use this flow chart when you create a program or check the created program.



[3] PROGRAM THAT BRANCHES --- TOUCH SENSOR

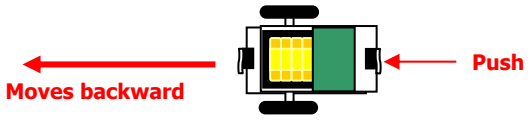
Let's learn about conditional branching with a simple program.

For this, touch sensors must be installed on the front of the robot.



We will prepare a branching program, using touch sensors, which repeats :

- the robot moves backward at a slow speed for a second and then the brake is applied for 0.5 seconds when pushed
- while the sensor is not pushed, it continues to move forward.



Moves backward at a low speed when pushed

Moves forward at a low speed when not pushed

<Hardware side>

There are a variety of touch sensors, however, KIROBO uses a micro-switch as its touch sensor. It lets the electricity pass through when pushed. If you push a touch sensor which is not connected to a power supply, you will hear a clicking sound. This clicking sound indicates the switching ON or OFF of the touch sensor.

When connected to the power source and pushed, electricity flows and turns the touch sensor ON. When not pressed, it is in the "OFF" condition because no electricity can flow through it.

<Program>

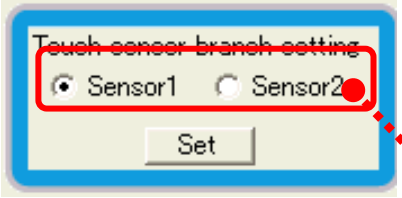
It branches at either ON or OFF.

Let's place the touch sensor icon and then set a property.

The property must be set so as to sense either of two touch sensors used in KIROBO.

Touch Sensor Icon

PROPERTY



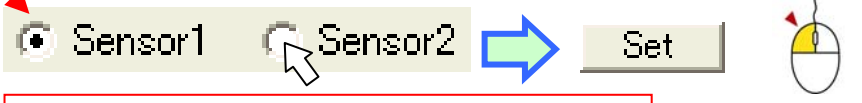
Indicates either 1 or 2

Select Touch sensor 1 or 2.

Select and confirm the touch sensor of the assembled robot.

This is how to change which sensor is being monitored.

(EXAMPLE) Change from "Sensor 1" to "Sensor 2"



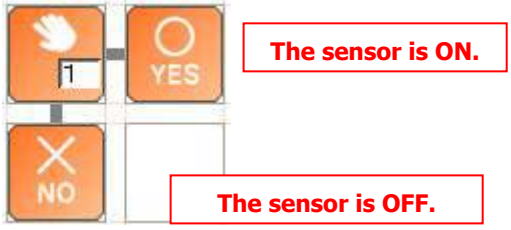
Left-click the radio button of sensor2

*** Either "YES" or "NO" icon must be set immediately after this icon.**

This icon asks whether the touch sensor is "ON" or "OFF".

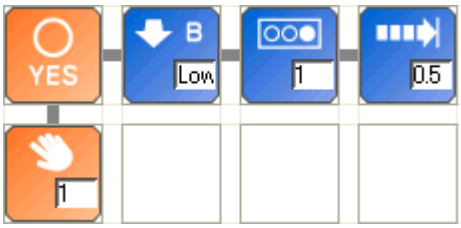
When the sensor is "ON", place "YES", and the program follows this path when the sensor is pushed.

When the sensor is "OFF", place "NO", and the program follows this path when the sensor is not pushed.



LET'S TRY PLACING THE OTHER COMMANDS WHILE YOUR MEMORY IS FRESH.

The robot "moves backward at a slow speed for a second and a brake is applied for 0.5 seconds when pushed".



1) When pushed, we follow "YES" because electricity flows through the touch sensor and becomes ON.
2) For "moves backward at a slow speed for 1 second", place a backward icon and change the property to low speed, then place "WAIT" and change the time setting to 1 second.
3) Place BRAKE icon and keep the condition without changing the property, and
4) Connect all the icons up using lines.

"moves forward when not pushed"

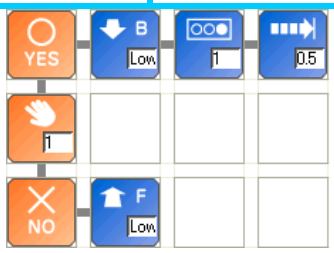
1) When not pushed, place NO because electricity isn't flowing through the touch sensor.
2) Change the "FORWARD" icon to the low speed.
3) Connect all the icons up using lines.

BRANCHING PROGRAM REGULATIONS

The icon name : MERGE

The branching program must be always returned to one flow. This icon makes the flow merge.

Let's learn how to use the "MERGE" icon…



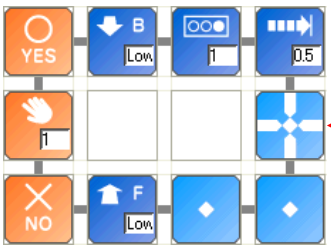
However, the number of the icons in the "NO" path isn't enough to make the flow merge.

In this case, we need to add icons to increase the path length to get them connected...

There are not enough icons for MERGE.

The icon name : NOP (meaning "No Operation")

This icon helps icons, which are distant, join together. It does not perform any operation or action itself and it has no properties. It is only a spacer.



Fill up the vacant area with "NOP" as shown and make the program.

Return the branched programs to one flow by "MERGE".

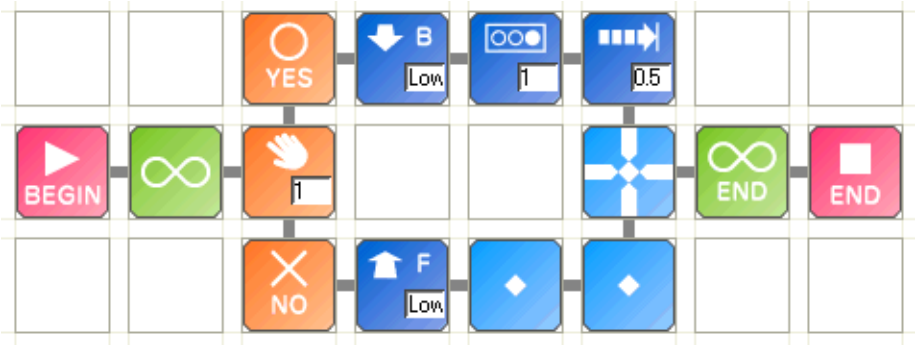
MERGE

Now, let's place the "repeat endlessly" icon.

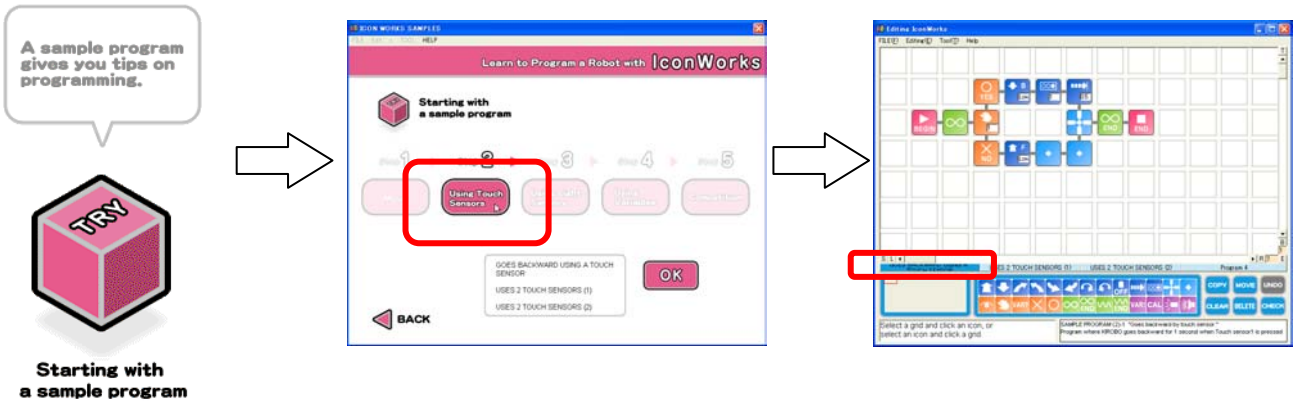
Do you recall which icon makes the robot move continuously until the power is turned off?

It is "LOOP"...

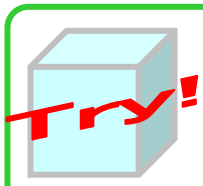
The completed program is as shown below. Send it to the robot and confirm how it works.



The same program is available from the initial screen "Starting from a sample program"--- "using step2 touch sensor"-"reversing with the touch sensor".



* Detailed guidance about this sample program is given on Page 4 of KIROBO USER'S MANUAL II, SAMPLE PROGRAM.



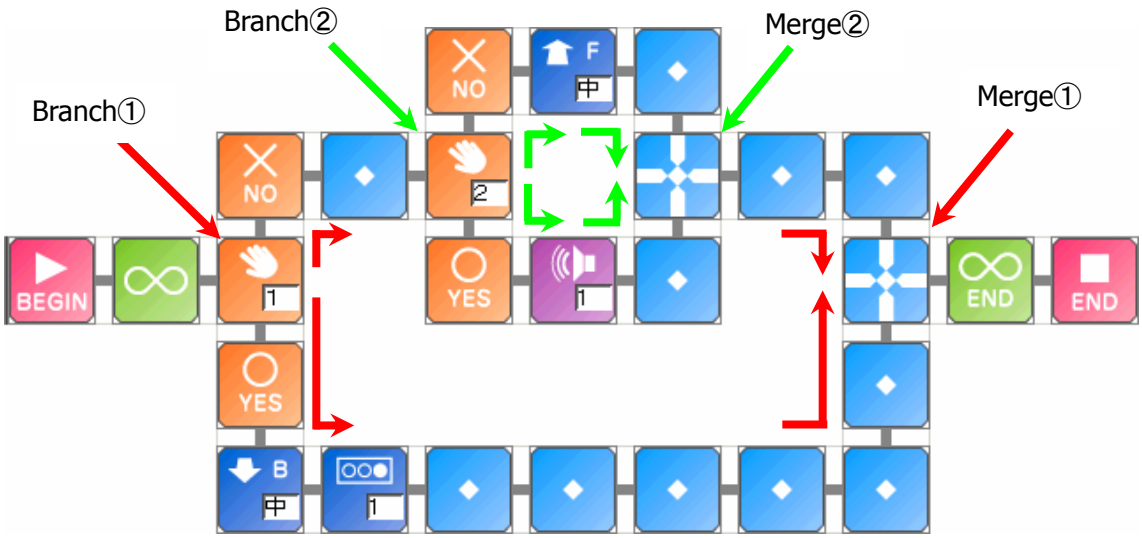
Make a program where the robot makes a high-speed right-turn for a second when the touch sensor1 is ON and generates the BEEP SOUND1 when the touch sensor2 is ON but it does nothing when neither touch sensor is ON.

Let's learn the rules for Branch and Merge!

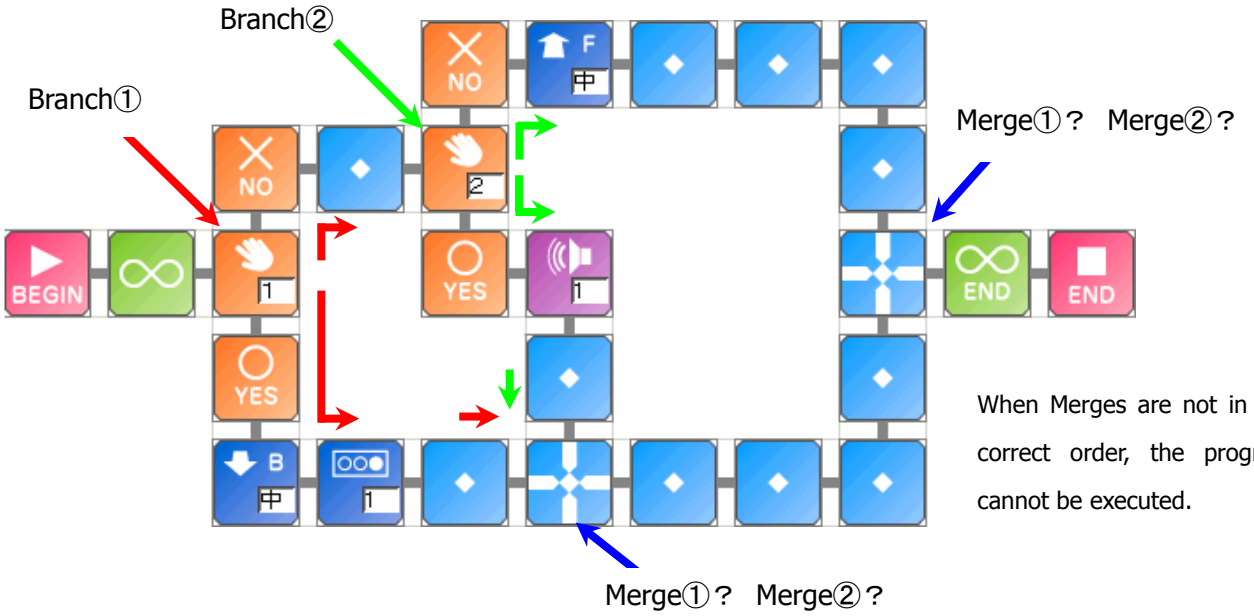
A branched program must always be merged into one line. When a program is branched for several times, make sure to merge two branched lines that are branched most recently first.

We recommend you, when you make a program with branches, to make it look visually simple so that you will know which branched lines need to be merged.

● Example of a correct program with 2 pairs of Branch and Merge



● Example of an **incorrect program** with 2 pairs of Branch and Merge



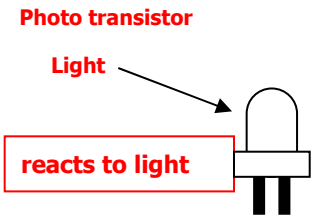
When Merges are not in the correct order, the program cannot be executed.

[4] PROGRAM THAT BRANCHES --- LIGHT SENSOR

As well as the touch sensors, there are 2 light sensors mounted on KIRORO. Let's study a program using a light sensor.

<Hardware>

From the variety of light (optical) sensors available, the one used on KIROBO is a "photo-transistor". When it finds (senses) light, an electric signal flows, this informs the microcomputer that it has found a light.



A light sensor is susceptible to the influence of surrounding light; therefore, its sensitivity must be adjusted according to the surrounding conditions by turning the resistor type adjuster on the motherboard.

<Program>

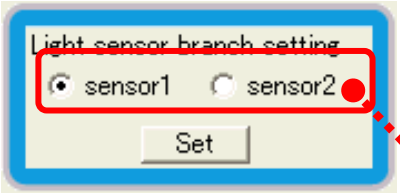
Program branches off if a light sensor has found a light (ON) or not (OFF).

Let's place the light sensor icon and set its property.

Setting the property means which of the 2 light sensors on KIROBO you want to monitor.

Light sensor icon

PROPERTY



Select either Light Sensor 1 or 2.

This is how you change the setting.

(EXAMPLE) Change from "Sensor 1" to "Sensor 2"

Indicates either 1 or 2

Left-click the radio button of sensor 2

*** After this icon, the icons of "YES" or "NO" must always be added.**

This icon is asking whether the light sensor is "YES" (ON) or "NO" (OFF).

If the sensor has found a light, "ON", then the program follows "YES".

If no light has been found, "OFF", then the program will follow "NO".

The sensor is ON.

The sensor is OFF.

Now, we will make a program which branches off using a light sensor.

In the program, the robot generates the beep sound 3 when the light sensor reacts while it repeats "doing nothing" endlessly when the light sensor does not react.

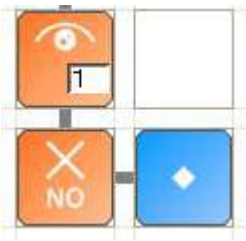
LET'S PLACE ICONS. MAKE SURE YOU REVIEW WHAT YOU HAVE LEARNED SO FAR.

"generates beep sound 3 when the light sensor reacts"

1) When it senses light, an electric signal flows from the light sensor. Therefore, we place "YES".
2) Place a BEEP icon and set at Timbre 3.

"doing nothing when the light sensor does not react"

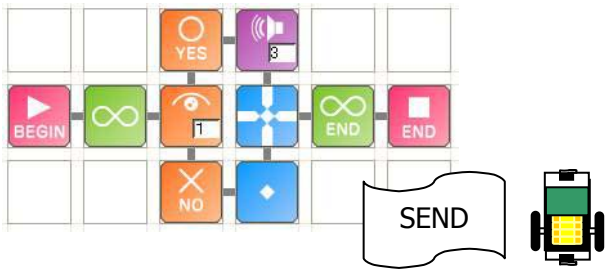
1) When it does not sense light, there is no electric signal flowing from the light sensor. Therefore, we place "NO".
2) Then, to do nothing, we place NOP icon for connection to other icons.



"Repeating endlessly"

1) Bring back the branched program to the main stream.
2) Place LOOP icon to repeat until the power is switched off.

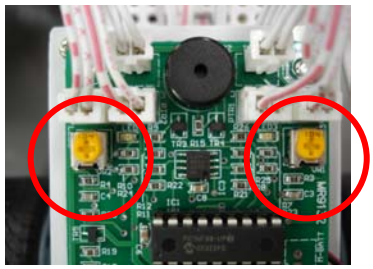
When completed, send the program to the robot.



Set the sensitivity adjustment of the light sensor at MIN.

"LED?" disappears when the sensitivity is low and the sensor is not reacting.

* Please handle with care.



Slowly adjust the volume towards the MAX direction.

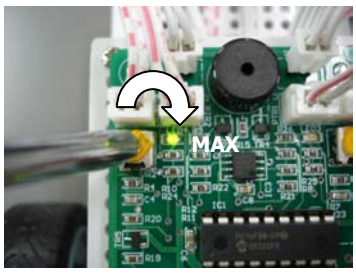
KIROBO beeps as soon as the sensor starts to react.

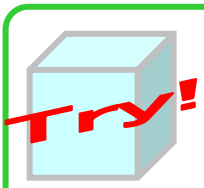
"LED?" also flashes when the sensor is reacting.

Since the amount of light varies depending on the environment,

please make adjustments accordingly,

(a bright environment – lower sensitivity, a dark environment – higher sensitivity).





Let's make a program where the robot beeps when the surroundings get dark.
*Go to "STARTING FROM A SAMPLE PROGRAM"

<Useful Shortcuts for the Operation of IconWorks>
Once you get used to operating IconWorks, try some more advanced techniques to program more speedily and with greater ease!

Move an icon quickly

To move an icon quickly, select the icon you want to move. Move the cursor to the grid where you want to move the icon to. Then, press the Ctrl (control) button.

Copy an icon quickly

To copy an icon quickly, have the icon you want to copy in the selected state. Move the cursor to the grid where you want to copy the icon to. Then, press Shift (shift) button.

Connect icons quickly

To connect 2 icons distant from each other, select all the grids (use area select) between the 2 icons, then click [NOP]. The [NOP] icons appear in all the grids in the selected area. You can also select any icons in-between as 'Action' icons in the selected grids remain as they are. [NOP] icons are allotted to vacant grids only.

Clear the selected state quickly

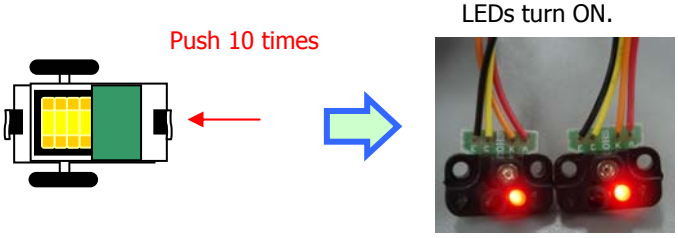
To clear the selected status when icons are in the selected status or an area is selected in block and the selected icons are flashing, press Esc (escape) button.

[5] PROGRAM USING VARIABLES

There are many ways of using variables depending on the program contents, and only a part of them are explained below. Let's try a simple program using variables that

To start with, make the below program.

Make a program as below;
The buzzer sound 1 is emitted every time Touch sensor 1 is pushed, and LED turns ON when Touch sensor is pushed 10 times.



As you have already learned, to create the above program, a conditional branching icon of "Push 10 times?" needs to be added to the program with a conditional branching of "Touch (1) is pressed?".

Both light sensors and touch sensors judge the situation by ON or OFF only.
However, any value in the range from 0 to 255 can be set for a variable icon,
and thus the situation can be judged according to the set value.

As described above, a value that can be arbitrarily set according to need or based on a calculation is called "Variable".
Variables in mathematics and that in programming do not mean exactly the same.
However, in this software IconWorks, a value that can be set according to need is called "Variable".
In IconWorks, a variable can be set according to the need, by a calculation, or by counting.

Variable icon

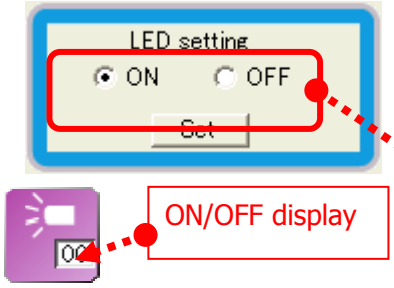
In addition, a variable in IconWorks judges the situation by an inequality sign ">=" (equal or more).
A variable is indicated by an alphabet from A to H, and the situation is judged by a number or another variable.
For example, in the case of "Press 10 times", the program questions as "Has Touch sensor1 been pressed more than 10 times?" (A>=10?), and proceeds to either "YES" or "NO".

How a LED turns ON is explained below.

LED
LED turns ON/OFF.

The red lights of the light sensors are the LEDs of the LED icons.

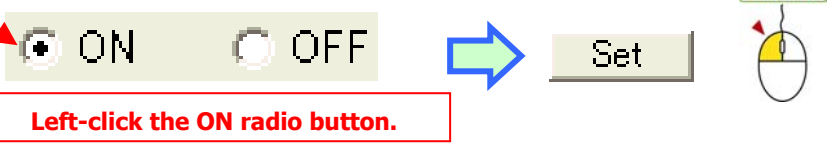
PROPERTY SETTING SCREEN



Select ON to turn ON the LED, and OFF to turn it OFF.

How to change the setting

(EX) Change from "OFF" to "ON".



Left-click the ON radio button.

RECALL THE OPERATION YOU HAVE DONE SO FAR, AND ARRANGE THE ICONS.

The buzzer sound 1 is emitted every time Touch sensor 1 is pushed.

1) Check if Touch sensor is pressed.

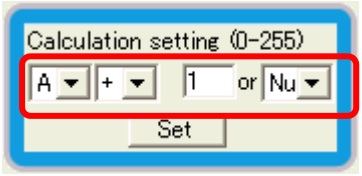
2) +1 is counted every time Touch sensor1 is pushed (turns ON).

Icon name: VARIABLE CALCULATION

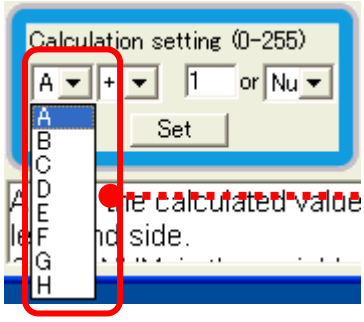
Add, subtract, multiple or divide the present variable with the specified value or variable.

The calculation result makes a new variable value for this icon.

PROPERTY



Select a variable to which a value found by the calculation is input from the selection box, and input a value in the value box to be used for calculation. A value is selectable in the range from 0-255. Or, select a value to be used for calculation from the selection box on the right-hand side.



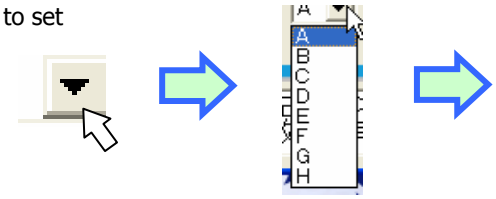
Up to 8 variables, A to H, can be set.

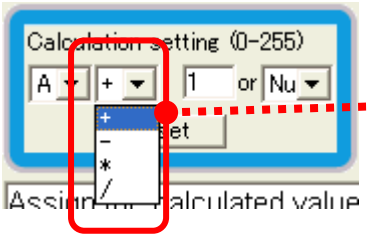
Here, for example, it is set to A.

In this case, calculate as below.

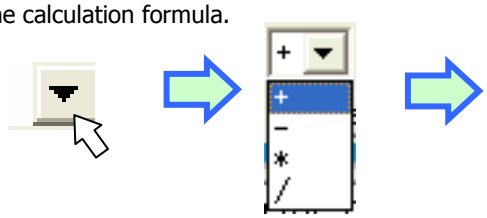
[Every time the program is repeated, +1 is counted for variable A.

How to set

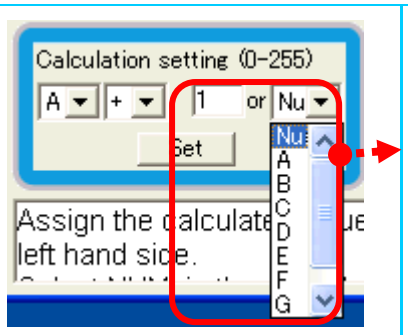




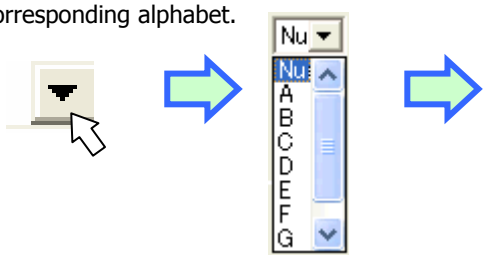
Set the calculation formula.



When you finish inputting in all the columns, left-click [SET].



When you input a number, select [NUMBER] from the selection box on the right-hand side. When you use another variable for calcuation, select a corresponding alphabet.



When you finish inputting in all the columns, left-click [SET].

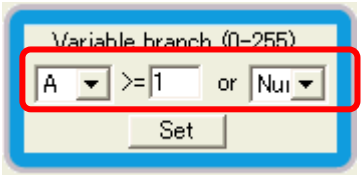
[When Touch sensor1 is pushed 10 times]

3) Check if Touch sensor1 is pushed 10 times.

Icon name: Variable branching

A program branches depending on the condition whether the variable value is more than a specified value or not.

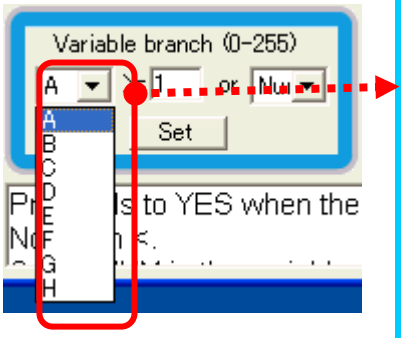
Property



Specify a variable to compare, or input a value. A value is selectable in the range from 0-255.

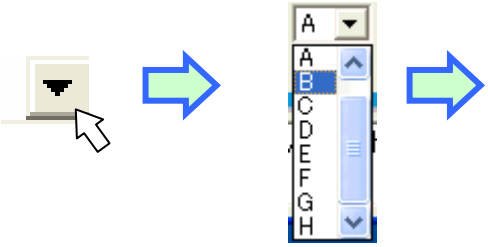
Up to 8 variables, A to H, can be set.

Here, for example, it is set to A.

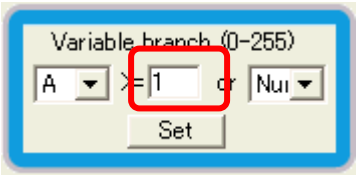


How to set

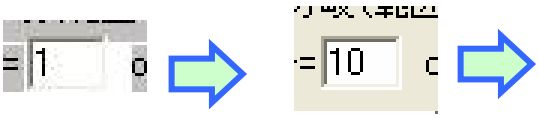
Select a variable for comparison from the selection box on the left-hand side.

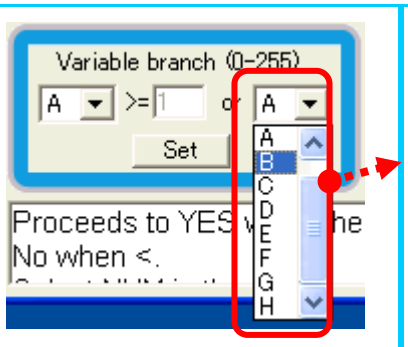


When all the columns are input, left-click [SET].

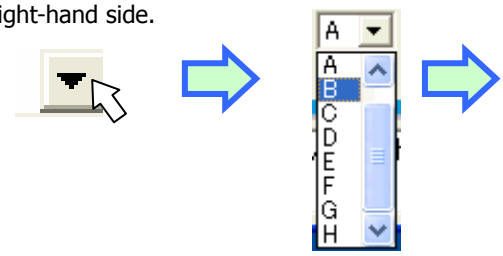


Input a value to compare.





When another variable is used for comparison, select a corresponding alphabet from the selection box on the right-hand side.



When all the columns are input, left-click [SET].

LET'S SUM UP!

"The buzzer sound 1 is emitted every time Touch sensor1 is pushed"

1) When Touch sensor1 is pushed, the program proceeds to YES direction and check if it is pressed more than 10 times.

2) When Touch sensor1 is not pressed, the program goes to the branch icon to check if it is pressed again.

"The LED turns ON when Touch sensor1 is pushed more than 10 times."

1) When Touch sensor1 is pushed more than 10 times, the program proceeds to YES, and LED turns ON.

2) When Touch sensor1 is pushed more than 10 times, the program proceeds to NO, and the buzzer sound 1 is emitted.

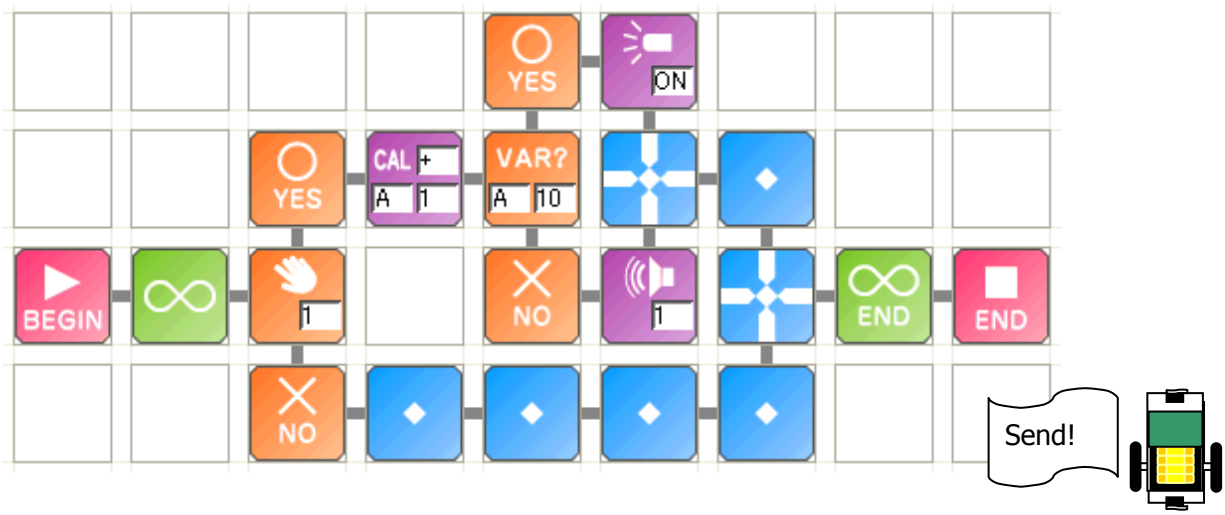
3) The programs that branched after "VAR" icon merge.

"Repeat endlessly"

1) Place "MERGE" icon to bring the programs branched after "TOUCH" icon back to the main stream.

2) Place "LOOP" and "LOOP END" icons to repeat a program until the robot power is turned OFF.

When the program is completed, send it to the robot.



In this program, the buzzer sound 1 is emitted also when Touch sensor1 is kept pressed when the program between "LOOP" and "LOOP END" is repeated 10 times, as it is regarded that Touch sensor1 is pressed more than 10 times.

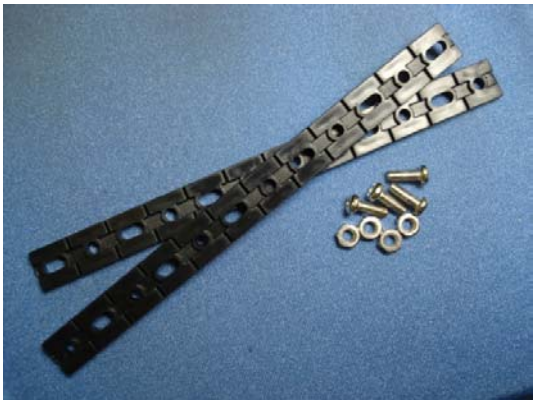
Keep pressing Touch sensor1 and check if the above phenomenon occurs.

[6] MODIFICATION

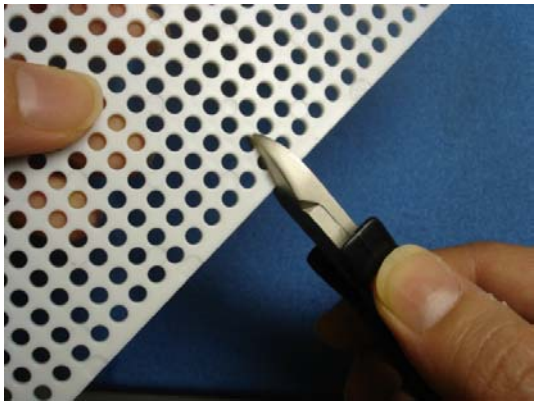
Now you have learned the basics of programming! So let's modify the robot and create your own KIROBO!

In the KIROBO package, various parts for modification are included. Use these parts to decorate the robot or change sensor positions.

The white panels can be painted. Use paints for plastic models and make your KIROBO look cool!

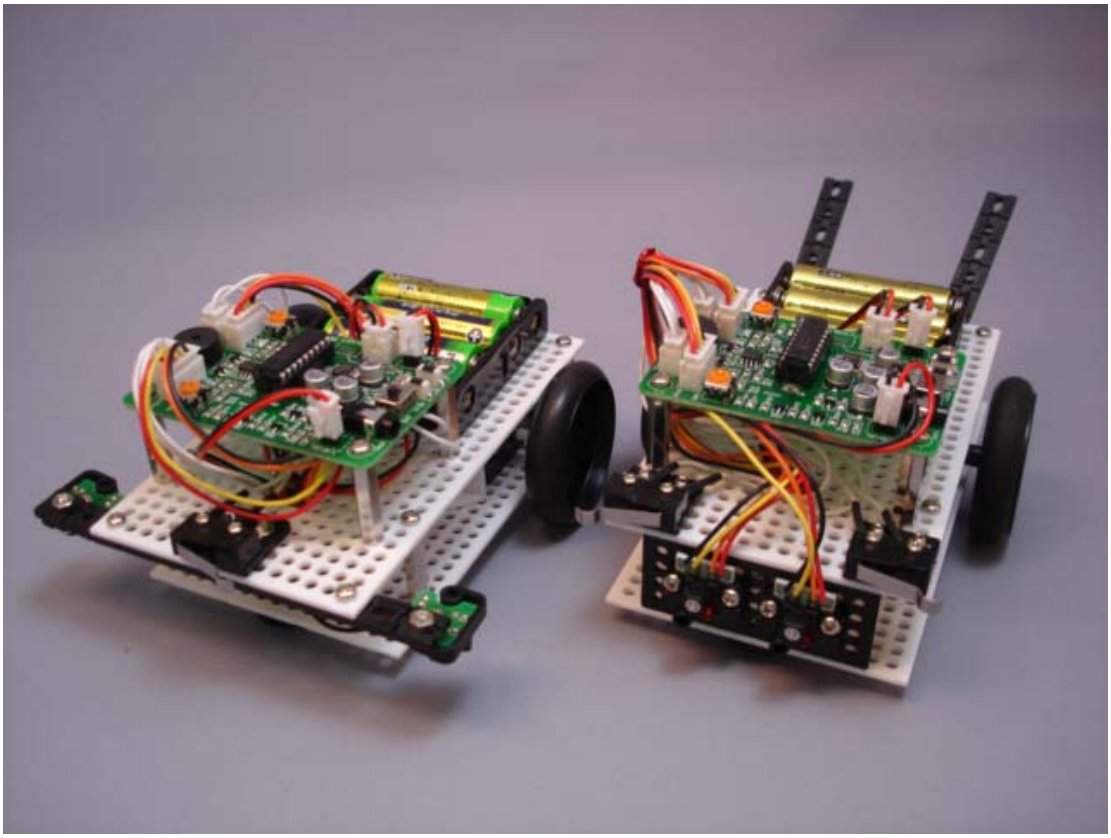


The parts can easily be installed with screws and nuts



The white plate can easily be cut by a diagonal cutter according to your need.

** Modification examples details will be introduced in coming Advanced version manual.*



V. TROUBLESHOOTING: Q & A

Q No icon is flashing even though a connection error occurs.

A When a connection error occurs even though there is no icon flashing on the visible area of the screen, scroll the screen and check the other part of the programming area where you could not see on the screen. Make sure there is no irrelevant icon in the programming area.

Q KIROBO does not go straight.

A When KIROBO is instructed to move forward it may not travel in a straight line, the following section explains why.
KIROBO is built with two motors to enable it to maneuver. These motors, although similar, are not exactly identical. Small differences in the manufacturing process can result in a range of slightly different characteristics. This range, called a 'tolerance', affects the speed of rotation in the motors. The target spindle speed maybe 3000rpm but it can vary between motors and be different depending on the direction of rotation. When one motor spins faster than the other it makes the wheel turn faster. This makes one side travel further in the same time causing KIROBO to follow a curved path. Other factors may affect the severity of the deviation, including the assembly of KIROBO'S chassis.
Given these factors there will almost certainly be some element of curvature in the forward or reverse paths taken by KIROBO. To compensate for this behavior we suggest that you write a program to correct KIROBO'S direction.
More complex robots compensate for these factors in a number of ways, including using 'paired' motors (motors which have identical characteristics), measuring the motor speed and adjusting the power while in use and GPS, LASER guides etc connected to steering systems. See the program included in the manual which guides KIROBO towards a light and develop your understanding of these techniques.

Q The sensor is sensing all the time.

A The sensors of KIROB react to ambient light, such as the light of a flashlight. Therefore, even though the sensitivity is adjusted to the lowest, the sensors might keep sensing the light all the time when KIROBO operates in a bright place, such as a window side where a lot of sunshine comes in or a room with a bright light. Adjust the brightness of the room where KIROBO operates by closing curtains or darkening up the light in the room.

Q [Error #480] occurs.

A The PC memory is running out. The software requires 128MB or more memory.
When this error occurs even though the PC memory is 128MB or more, it is possible software that is active concurrently with IconWorks is using the memory. Therefore, shut down the software other than IconWorks.

Q LED2 (red) starts flashing right after the power SW turns ON.

A This phenomenon occurs when the power SW is pushed with Touch sensor1 or 2 pressed. This indicates it is in a default adjustment mode, and there is no problem in the function. When SW1 or SW2 is pushed in this status, the default condition might be changed. In such case, turn OFF the power SW, and turn ON again when the touch sensor is not pressed.

Q A program cannot be transferred to KIROBO successfully, or it cannot be transferred from a specific PC.

A There is a sound output unit embedded in a PC (called a built-in sound board or a sound card), and some of them tend to emit larger noise than the others, which is considered to be one of the cause of unsuccessful program transfer. In such case, the problem could be solved by using a commercially available "USB audio conversion cable".

* "USB audio conversion cable" is usually available at PC shops.

Such "USB audio conversion cable" can be considered as an external sound circuit, and a KIROBO program can be sent through this external sound circuit, without passing through the embedded sound card. Therefore the noise upon program transfer can be reduced to enable successful program transfer.

End of the document