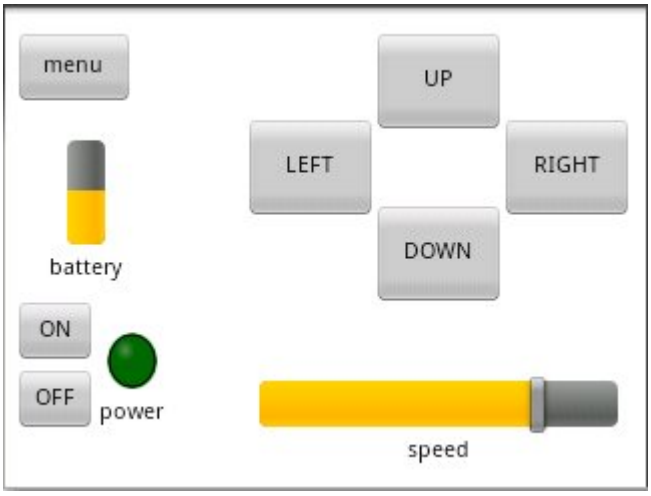
RoboRemo v2.5.0
User Manual



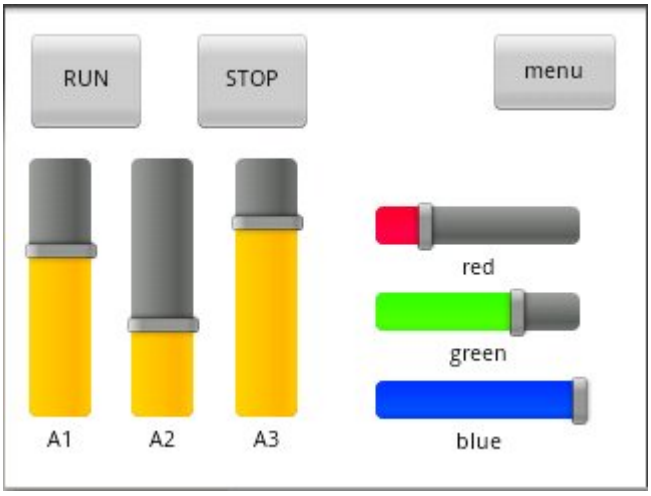


Table of Contents

General Description....3
Bluetooth / WiFi / Ethernet / USB modules....4
Available interface items....6
Building the interface....9
Common 3 buttons in item settings....10
Button settings....11
Local actions....12
Slider settings....13
Joystick settings....14
LED settings....14
LED examples....15
Level indicator settings....16
Text log settings....16
Accelerometer settings....17
Text field settings....17
Plot settings....18
Image settings....19
Touchpad settings....20
Kbd connector settings....21
Kbd connector examples....21
Heartbeat sender settings....22
Touch stopper settings....22
Vibrator settings....22
File sender settings....23
File receiver settings....24
Speaker settings....25
Printf() settings....26
Menu options....27
App. settings....28
Interface options and settings....30
New features and fixed bugs....31

General Description

RoboRemo is a **user-customizable** remote control application intended mainly for **electronics hobby projects**. RoboRemo can connect via **Bluetooth (RFCOMM / BLE)**, **Internet** or **WiFi (TCP, UDP)**, and **USB (CDC-ACM, FTDI, PL2303, CP210x, CH34x, etc.)**.

Disclaimer: Don't use RoboRemo for life support systems or any other situations where system failure may affect user or environmental safety. Please don't use RoboRemo in projects where high-level security is required.

To connect via **Bluetooth**, a remote device must contain a **Bluetooth to Serial adapter** like **BlueSMiRF, BTM-222, HC-05, HC-06, etc.** or **BLE Serial adapter like HM-10 CC2540** and a **microcontroller (MCU)** programmed to interpret commands from RoboRemo. Adapter boards / modules for **Ethernet** or **WiFi** are easily available on the market as well. Some examples for WiFi are **ESP8266** and **ESP32** boards.

For **USB connection**, the phone / tablet must have **USB API** and **USB OTG** support.

For **BLE connection**, the phone / tablet must have **BLE support**.

All the **commands** from RoboRemo are **text strings**, ending with **command ending** which is **LF** character **'\n'** (hex code 0x0A) by default, but **user can change it**. Example: a button configured to send “abc” when pressed, will send “abc\n” if command ending is '\n' or it will send “abcqwerty123” if command ending is “qwerty123”. The remote device’s MCU is interpreting the Bytes received from RoboRemo. It must use exactly the same command ending to **“know” where each command ends**.

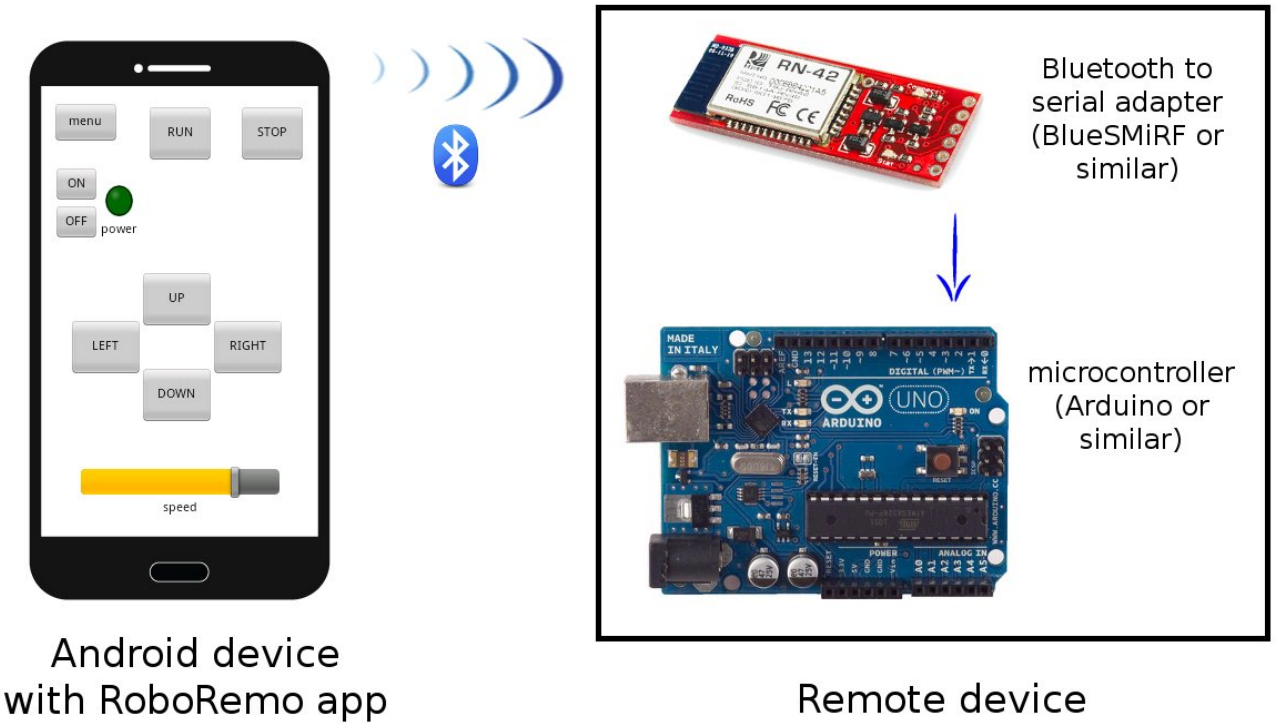
RoboRemo can also receive input commands for updating some interface items (example: a level indicator to display the battery level of the remote device or an LED to indicate some state). These **input commands also** must **end with command ending.** The **command ending** must be configured exactly the **same** in **both** app and MCU. Common mismatch is “**\n**” vs “**\r\n**”.

Command ending can also be set to **empty string,** which is **not recommended**, as each received character will be interpreted as separate command, which is probably not the expected behavior. Sending, however, will work as expected (no command ending → only the actual command string will be sent → a button configured to send “abc” will send “abc”).

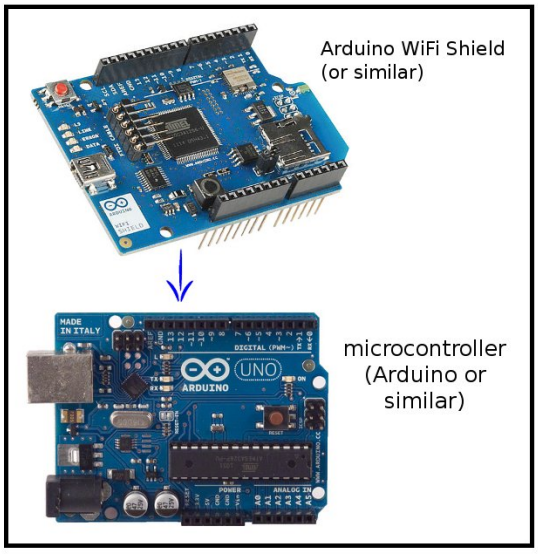
Bluetooth / WiFi / Ethernet / USB modules

RoboRemo was tested and works with: BlueSMiRF, HC-05, HC-06, BTM-222, HM-10, CC2540, ESP8266 (ESP-01), HUZZAH ESP, Arduino UNO, Arduino Mega (Arduino connected directly with USB cable + OTG adapter). Many other boards and modules should be compatible as well.

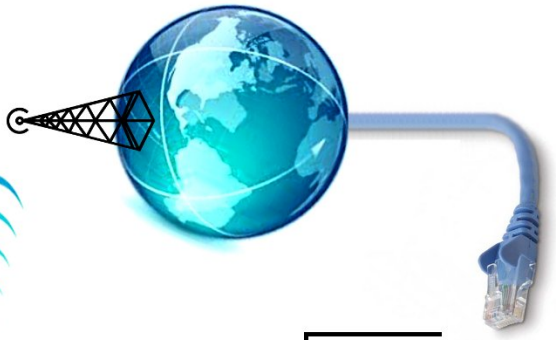
When using Bluetooth module and microcontroller, please **make sure** the microcontroller uses **same BaudRate** as the module. The default BaudRate for BlueSMiRF is usually **115200**, and for HC-05 / HC-06 is usually **9600**.

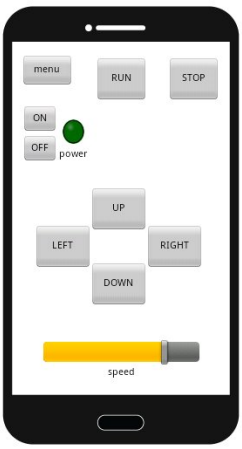


Android device
with RoboRemo app

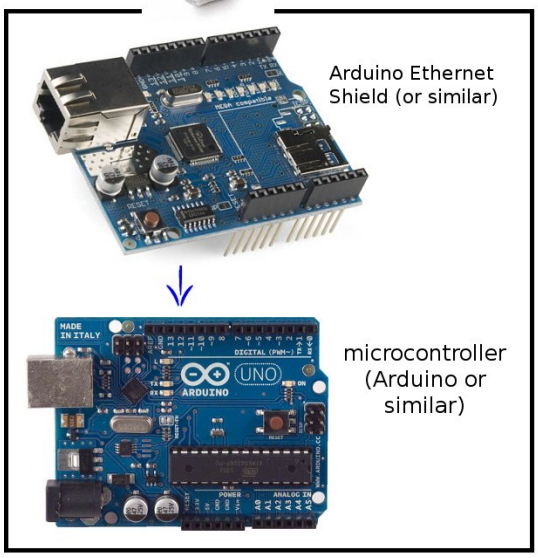


Remote device





Android device
with RoboRemo app



Remote device

Available interface items

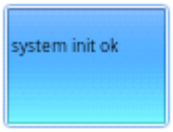
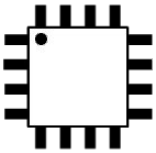
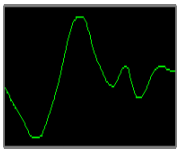
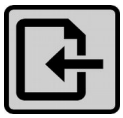
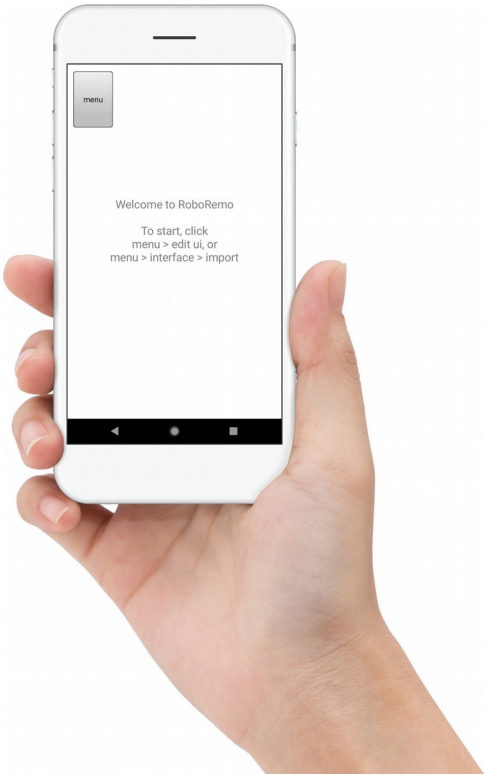
item name	photo	description
button		Buttons can be used for sending commands to the remote device. A button can be configured to send one command when pressed and other command when released.
slider		Sliders can be used to vary remote device parameters, for example volume control, or motor speed or light intensity.
joystick* (**new!**)		Joystick allows user to control 2 parameters at once. It is like a slider with 2 axis: X and Y. Instead of having one slider for X and one for Y, now both can be controlled with one finger. *available only in RoboRemo Full version.
led		LEDs can be used as indicators for remote device states.
level indicator		A level indicator item can be used to display data from remote sensors, for example temperature or battery level.
text log		A text log item is a log screen that can be used to display debug messages from the remote device.
accelerometer		The accelerometer can be used to periodically send the acceleration measured by the phone's accelerometer. Each of the 3 axes has its own configurable ID and other settings. The repeat (refresh) period for the accelerometer is also configurable.
text field	text	Text field item, to add some text to the interface. Note: to make an item be displayed on top of other items, copy the item, then remove it, then paste it.
plot		Plot item can be used to display some signals.

image*		Image item has many uses: - background image for the interface - set of images to show from (MCU sends an index that selects which image to display) - display image data (MCU sends the image data), can be used for live streaming from a camera that is attached to the MCU – requires fast-enough MCU and communication. *available only in RoboRemo Full version
touchpad		Touchpad item can be used to send the (x, y) position and motion of the finger.
kbd connector		Kbd connector can be used to send commands using the keyboard.
heartbeat sender		heartbeat sender can be used to periodically send some string (like a heartbeat signal). The remote device can watch for this signal to make sure it is still connected.
touch stopper		Normally the touch event triggers the nearest touchable item on the interface. This is the intended behavior that assures small items are not missed. This touch event propagation can be stopped by touch stoppers. A reset button, for example, can be surrounded with touch stoppers so the nearby touch events do not trigger it. Touch stoppers are visible only in the UI editor.
vibrator*		Vibrator item gives the remote device access to the phone's vibrator (MCU can send a command to make the phone vibrate). *available only in RoboRemo Full version.
file sender*		File sender item can be used to send files from the phone to the MCU. *available only in RoboRemo Full version.

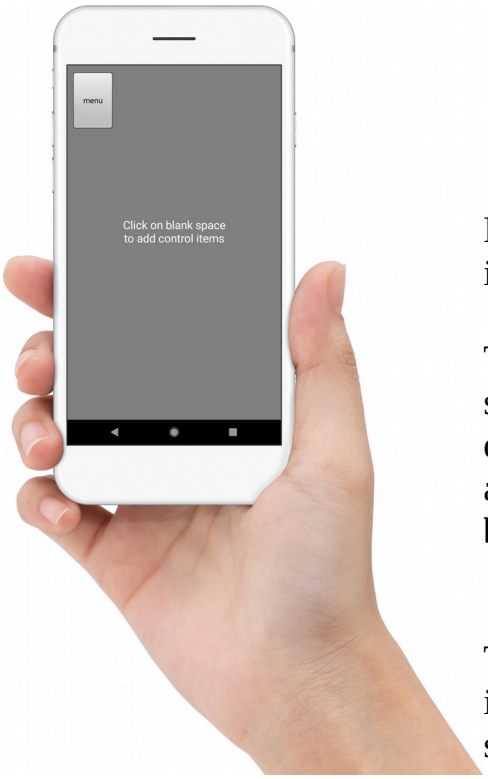
file receiver*		File receiver item can be used to receive files from the MCU and store them on the phone. *available only in RoboRemo Full version.
speaker*		Speaker item gives the remote device access to the phone's speaker (MCU can send a command to make the phone beep). *available only in RoboRemo Full version.
printf()		Gets the data from other items and sends a formatted packet. User can configure the format. Currently, the printf() item can get data from button, slider, joystick, accelerometer and heartbeat sender. See printf() settings for more details.

Building the interface

When first launching the RoboRemo app,
it will look like this picture on the right →

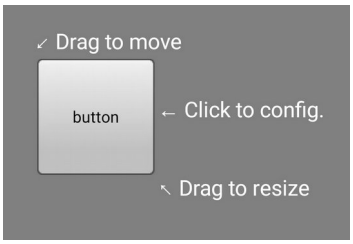


To edit the interface, click **menu → edit UI.**



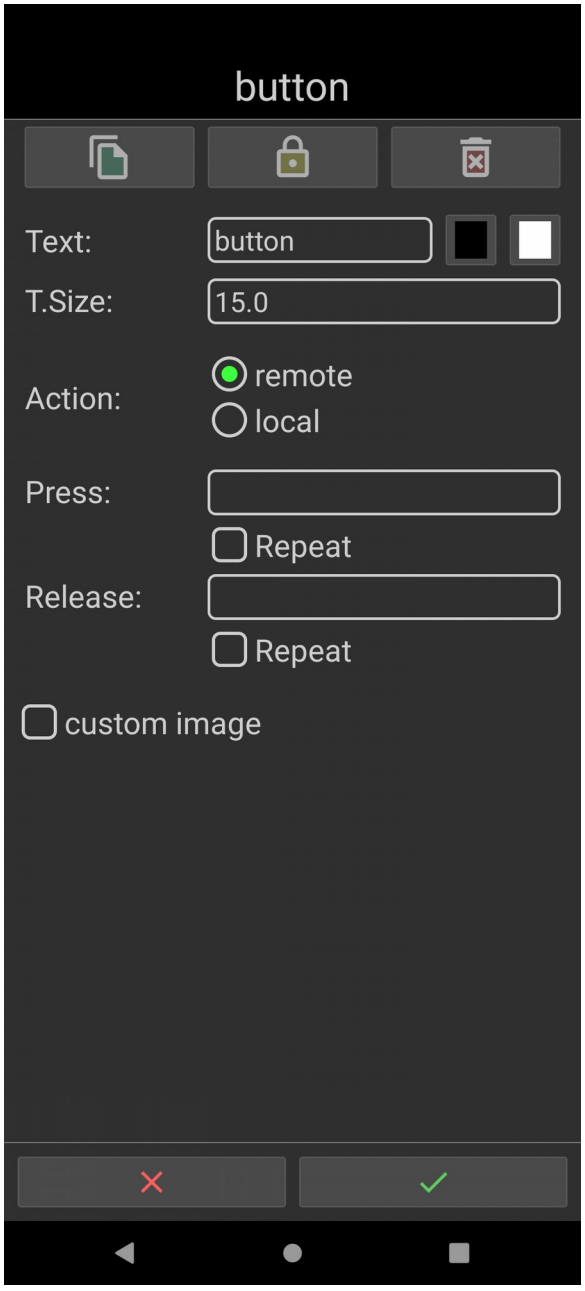
Now in edit mode, **click on blank** space to add an item, then choose the item type.

The item will appear on screen. It can be **moved** by dragging the **top left corner,** and **resized** by dragging the **bottom right corner**.



To **configure** the item settings, **click** on the desired item and the configuration window will appear on screen.

Common 3 buttons in item settings



← The first **3 buttons** are common for all items. They have the following functions:

Copy	Copy the item. Then the “add item” menu will have “paste” as first option, that is used to paste the copied item.
Lock edit	Lock the item from editing. It will not be editable anymore. Locking an item is useful when adding other items on top, or when there is small screen space between items. Items can be unlocked all at once from menu → interface → unlock edit all.
Remove	Remove the item from the interface.

Button settings

setting	description
Text	Text that appears on the button.
Text color	Text color when button is released.
Text color (pressed)	Text color when button is pressed.
T.Size	Font size for the text that appears on the button.
Action (remote / local)	Change button action type. Default is remote.
Press	Action to execute when the button is pressed. For remote action: String to send to the remote device. (RoboRemo will append the command ending to mark the end for each command). For local action: See the list of local actions.
Release	Action to execute when the button is released. For remote action: String to send to the remote device. (RoboRemo will append the command ending to mark the end for each command). For local action type: See the list of local actions.
Repeat	Repeat the press and / or release action(s) while the button is pressed. When user holds a button, it will first repeat after the delay value, then after the period.
custom image*	User can change the button image, can select different images for **pressed** button and **released** button. The selected images are stored in the interface file. The interface can be exported and imported on another device. Note: big image files may slow down the app. It is recommended to use images **less than 100KB**. *available only in RoboRemo Full version.

Local actions

Local action / example	description
“iface n” or “interface n”	select the nth interface. (n from 0 to 7)
“echo abc”	will simulate the receiving of “abc” command.
“send abc”	will send “abc” (followed by command ending) to the remote device.
“sendnce abc”	will send “abc” (without command ending).
“sendhex 4A4B4C”	will send “JKL” (without command ending).
“disconnect”	disconnect from the remote device
“connect rfcomm 11:22:33:44:55:66 1” or “connect rfcomm 112233445566 1”	connect via Bluetooth RFCOMM to the remote device with address 11:22:33:44:55:66 and port 1.
“connect rfcomm e 112233445566 1”	Also use encryption.
“connect rfcomm 112233445566 d”	Connect using the default UUID
“connect rfcomm 112233445566 f”	Connect using the first UUID
“connect ble 112233445566 0000ffe0-0000-1000-8000-00805f9b34fb 0000ffe1-0000-1000-8000-00805f9b34fb 0000ffe1-0000-1000-8000-00805f9b34fb 0 0”	Connect to BLE device with address 11:22:33:44:55:66. The 3 UUIDs are: service UUID, TX UUID, RX UUID. The last 2 parameters “0 0” will be used in the future. Since manually writing the UUIDs is not fun, it is recommended to use these steps: connect to the device using menu → connect, edit the interface, add a button, set local, set press action to “con” (without “”) and it will complete automatically.
“connect tcp 192.168.0.5:9876”	Connect via TCP/IP to the remote device with IP 192.168.0.5 and port 9876.
“connect udp 192.168.0.5:9876”	Connect via UDP. Packets will be sent to the port 9876 of the remote device with IP 192.168.0.5 Phone / tablet will listen for packets on port 9876.
“connect usb 115200”	Connect via USB using BaudRate 115200.
“con”	If there is an active connection and button is set to local, then setting the press or release action to “con” will **auto-complete** it with the string for the **current connection**. (No need to manually write the IP and port or the Bluetooth device address).

Note: Starting with version 1.9.1, local action can contain **multiple commands**, separated by command ending. For example:
- set command ending to “\n”
- add a text log
- add a button, set it local, set its press action to “**echo sending abc\nsend abc**”

Now when pressing the button, “sending abc” will appear in the text log and “abc” will be sent to the remote device.

Slider settings

setting	description
ID	Slider ID string. Example: ID = “s1”, command ending = “\n” and user moves the slider to the value 100 → it will send “s1 100\n” to the MCU (ID followed by space followed by value followed by command ending).
Label	Text string to appear under the slider. The slider’s current value can be used inside the label. The decimal count for the value inside the label is also configurable. Examples for slider with value 100: label “speed = #*0.1” will show “speed = 10.0” label “x = #*-5+10 cm” will show “x = -490 cm”
Label color	Color for the label text.
Dec.count	Decimal count for the **label** (how many digits after “.”)
T.Size	Font size for the label.
Output (int / float)	Output type for the slider value (integer / floating point number). For float output type, user can also set the decimal count and the step count.
min, max	Minimum and maximum values for the slider.
Dec.count	Decimal count for the **output value** (how many digits after “.”)
Steps	Number of slider positions, for the float output type.
send space after ID	Configure the slider to send space character after ID or not.
auto return to	Enable / disable auto-return of the slider handle when released, set the value to return to (integer number, or min/mid/max).
repeat, period	Enable / disable slider repeat (sending the value periodically even if not moved), set the repeat period.
Color (ARGB)	Slider fill color.
send when released / moved	Slider send mode (send only when released / while moved).

Joystick settings

setting	description
Label	Text string to appear under the joystick.
Label color	Color for the label text.
T.Size	Font size for the label text.
X/Y axis ID	ID string for the X/Y axis of the joystick. Example: X ID = “x”, Y ID = “y”, command ending = “\n” and user moves the joystick to (10, 20) → it will send “x 10\ny 20\n” (for each enabled axis: ID followed by space followed by value followed by command ending). Empty ID disables the corresponding axis. Non-empty ID makes other settings appear for configuring the corresponding axis.
Output (int / float)	Output type for the axis value (integer / floating point number).
min, max	Minimum and maximum values for the axis.
auto return to	Enable / disable auto-return of the joystick handle when released, set the value to return to (number, or min/mid/max).
Dec.count	Decimal count for the float **output value** (how many digits after “.”)
steps	Number of available values for that axis, for the float output type.
send space after ID	Configure the joystick to send space character after IDs or not.
repeat, period	Enable / disable joystick repeat (sending the values periodically even if not moved), set the repeat period.

LED settings

setting	description
ID	LED ID string.
Label	Text string to appear under the LED.
Label color	Color for the label text.
T.Size	Font size for the label.
ON cmd.	LED ON command, default is “1”.
OFF cmd.	LED OFF command, default is “0”.

ON / OFF Timeout	Timeout after which LED will turn OFF / ON if it does not receive ON / OFF command. Example: LED with ON timeout = 500ms, a remote device that sends the turn ON command every 250 ms -> the LED will stay ON as long as the connection is alive. The **LED will turn OFF** if the **connection** is **lost**.
Color (ARGB)	ON color for the LED. OFF color (darker) is computed automatically from this.

LED examples

LED ID	led	ledA	ledB
LED ON command	1	on	#>=10
LED OFF command	0	off	#<10
Interface command ending	\n	;	ok
To turn ON the LED, microcontroller must send	led 1\n	ledA on;	ledB 10ok or ledB 11ok etc.
To turn OFF the LED, microcontroller must send	led 0\n	ledA off;	ledB 9ok or ledB 8ok etc.

Level indicator settings

setting	description
ID	Level indicator ID string. Example: ID = “batt”, command ending = “\n”, MCU sending “batt 100\n” will set the level indicator to 100.
Label	Text string to appear under the level indicator. The level indicator’s current value can be used inside the label. The decimal count for the value inside the label is also configurable. Examples for level indicator with value 100: label “temp. = #*0.5-10 °C” will show “temp. = 40.0 °C” label “batt. #*1%” will show “batt. 100%”
Label color	Color for the label text.
Dec.count	Decimal count for the label (how many digits after “.”)
T.Size	Font size for the label.
min	minimum value, default is 0.
max	maximum value, default is 255.
Color (ARGB)	Level indicator fill color.

Text log settings

setting	description
ID	Text log ID string. Example: ID = “dbg”, command ending = “\n”, MCU sending “dbg qwerty\n” will append the message “qwerty” to the text log.
Label	Text string to appear under the text log.
Label color	Color for the label text.
Label T.Size	Font size for the label text.
T.Size	Font size for the text that appears inside the text log.
Log to file*	User can choose a file where to log the data that is received by this text log. For each received string, it will first append a prefix, then the string. Default prefix is “\n” (new line) so that each received string will be written to the file in a new line. User can change the prefix. Multiple items can log to the same file. *available only in RoboRemo Full version.

Accelerometer settings

setting	description
X/Y/Z axis ID	ID string for X/Y/Z axis. Accelerometer will periodically send the ID followed by a space character (or not), followed by the processed acceleration value for that axis. The value is processed as follows: The data from the accelerometer sensor is multiplied by gain, then it is **mapped** from **(-9.8, 9.8)** to **(min, max)**, then it is limited (or not) to [min, max]. Empty ID disables the corresponding axis. Non-empty ID makes other options appear as well for configuring the corresponding axis.
Output (int / float)	Output type for the axis value (integer / floating point number).
min, max	Minimum and maximum values for the axis.
Gain	Amplification factor for the axis value (for float output type).
Limit to [min, max]	Limit the axis to the minimum and maximum values.
Send space after ID	Configure the accelerometer to send space character after IDs or not.
Repeat period	Repeat (refresh) period for sending the accelerometer data. Default is 20ms.

Text field settings

setting	description
ID	ID string for the text field. Example: ID = “text1”, command ending = “\n”, MCU sending “text1 abc\n” will set the text to “abc”.
Text	Text to be displayed initially inside text field.
T.Size	Font size for the displayed text.
Color (ARGB) Foreground / Background	Foreground / Background color for the text. Format is ARGB_8888 (hex code). Examples: FF000000 = solid black, 80FF8000 = half transparent orange FFFFFFFF = solid white, 00FFFFFF = fully transparent

Hint: To make an **item** be displayed **on top of other items**, **copy** the item, then **remove** it, then **paste** it.

Plot settings

setting	description
ID	ID string for the plot. Example: ID = “x”, command ending = “\n”, MCU sending “x 100\n” will add a new sample with value 100 to the plot.
Label	Text string to appear under the plot. The plot's most recent value can also be used inside the label. The decimal count for the value inside the label is also configurable. Examples for plot with last value 100: label “temp. = #*0.5-10 °C” will show “temp. = 40.0 °C” To show “temp. = 40 °C”, set the decimal count to 0. label “batt. #*1%” will show “batt. 100%”
Label color	Color for the label text.
Dec.count	Decimal count for the label (how many digits after “.”)
T.Size	Font size for the label.
min	minimum value, default is 0.
max	maximum value, default is 255.
Length	Window length (number of samples in the plot window), default is 256. Maximum length is 10000.
Line width	Line width for the plot. Increase this value to increase visibility.
Line color	Line color for the plot.
Disp.mode scrolling / oscilloscope	Display mode for the plot. Scrolling (default) – plot scrolls from right to left, new data appears on the right. Oscilloscope – plot doesn't scroll. Data appears from left to right. When it reaches the end, it waits for trigger event, then starts again from the left.
Trigger rising / falling, Threshold	Trigger edge (rising / falling / both / none) and threshold level for oscilloscope display mode. If none edge selected, then the trigger is off, so it will not wait for trigger event. For scrolling display mode there is no trigger (The trigger settings are ignored).
Don't repaint	Disables UI repaint when new sample is received. (For increased performance, when multiple plots are updated at the same time, the last one can trigger the UI repaint for all).
log to file*	User can choose a file where to log the data that is received by this plot. For each received value, it will first append a prefix, then the value. Default prefix is “\n” (new line) so that each value will be written to the file in a new line. User can change the prefix. Multiple items can log to the same file. *available only in RoboRemo Full version.

Image settings

setting	description
ID	ID string for the image. Example: ID = “img”, image format = “auto”, command ending = “\n”, MCU sending “img [len]\n” followed by [len] Bytes of the image data will make that image appear inside the image item in RoboRemo. [len] must be the size in Bytes of the image data that follows. The “**auto**” format accepts **JPG / PNG / BMP** data.
Label	Text string to appear under the image.
Label color	Color for the label text.
T.Size	Font size for the label text.
Keep ratio	Sets if the image should keep the original w/h ratio, or should it fill the entire image item.
Filter	Sets if the image should be filtered or not (square pixels) when scaled.
Set as background	All other items will be displayed on top of the image. The image item is locked from editing. It can be unlocked from menu → interface → unlock edit all.
Received image format	Sets the image format. Default format: auto (JPG / PNG / BMP data). Raw formats: RGB_888, GRAY_8, GRAY_4, GRAY_2, GRAY_1. The pixels are **scanned** from **left to right**, lines from **top to bottom**. RGB_888 uses 3 bytes per pixel (the levels for red, green and blue). GRAY_8 uses 1 byte per pixel (has 256 gray levels). GRAY_4 uses 4 bits per pixel (1 byte for 2 pixels) and has 16 gray levels. GRAY_2 uses 2 bits per pixel (1 byte for 4 pixels) and has 4 gray levels. GRAY_1 uses 1 bit per pixel (1 byte for 8 pixels) and has 2 gray levels (0 is full black and 1 is full white)
Width / Height	Width and height (in pixels) for the received images for the raw image formats.
Length	Expected data length (in bytes) for the received image.
Imported images [+]	Import images from the phone’s file system. The selected images are stored in the interface file. The interface can be exported and imported on another device. Note: big image files may slow down the app. It is recommended to use images **less than 100KB**. By default, the first image (index 0) will be displayed. Other imported image can be displayed by sending [image ID] + space + “i” + space + [index] + [cmd. End] **Example:** ID = “**img**”, command ending = “**\n**”, MCU sends “**img i 2\n**” → the imported image with **index 2** will be displayed.

Touchpad settings

setting	description
IDs Press / Drag / Release	ID strings for the press / drag / release events. Example: press ID = “pressed”, command ending = “\n”, when the user touches the touchpad, it will send “pressed [x] [y]\n”, where [x] and [y] are the point coordinates. Ex: “pressed 316 244\n” Empty ID disables corresponding touchpad event.
Label	Text string to appear under the touchpad.
Label color	Color for the label text.
T.Size	Font size for the label text.
x min	X coordinate corresponding to the left border of the touchpad. Default is 0.
x max	X coordinate corresponding to the right border of the touchpad. Default is 640.
y min	Y coordinate corresponding to the top border of the touchpad. Default is 0.
y max	Y coordinate corresponding to the bottom border of the touchpad. Default is 480.

Kbd connector settings

setting	description
ID	ID string for the kbd connector. (see examples below)
Label	Text string to appear under the kbd connector..
Label color	Color for the label text.
T.Size	Font size for the label text.
Action (remote / local)	Set if the kbd connector should send the commands to the remote device, or should it send back to the interface. (see examples below)
Send mode (while typing / when press enter)	Sets if the kbd connector should send each character separately or should it wait for enter key. (see examples below)

Kbd connector examples

Assuming command ending is “\n”

ID	local / remote	send while typing / when press enter	user is typing	Result
“kbd”	remote	while typing	ab[enter]	Remote device receives: “kbd a\nkbd b\n kbd \n\n”
“kbd”	remote	when press enter	ab[enter]	Remote device receives: “kbd ab\n”
“” (empty string)	remote	while typing	ab[enter]cde	Remote device receives: “ab\ncde”
“” (empty string)	remote	when press enter	ab[enter]cde	Remote device receives: “ab\n” (and at next [enter] will receive “cde\n”)
“echo”	local	while typing	abc	Local interface receives “a\ nb\nc\n”
“echo”	local	at enter	abc[enter]	Local interface receives “abc\n”
“” (empty string)	local	at enter	echo abc[enter]	Local interface receives “abc\n”

Heartbeat sender settings

option	description
ID	String that will be sent periodically to the remote device.
Label	Text string to appear under the item.
Label color	Color for the label text.
T.Size	Font size for the label text.
Repeat period	Time interval at which the heartbeat signal should be sent. Default is 1000ms.

Touch stopper settings

Touch stopper does not have any special settings.

Vibrator settings

setting	description
ID	ID string for the vibrator. Example: ID = “vib”, command ending = “\n”, MCU sending “vib 200\n” will make it vibrate for 200 ms. MCU sending only “vib\n”, it will use the default duration that is set with “set duration”. Sending “vib 0\n” will make it stop.
Label	Text string to appear under the item.
Label color	Color for the label text.
T.Size	Font size for the label text.
Default duration	Default duration for the vibration. Default is 500ms.
OFF timeout	The timeout after which vibrator will turn ON if it does not receive OFF command. (OFF command means “vib 0\n” for a vibrator with id “vib” on an interface with command ending “\n”). Default is empty string (timeout disabled).

File sender settings

setting	description
ID	ID string for the file sender. Default ID is “**file**”. Example: **ID** = “**file**”, **command ending** = “**\n**”, user clicks the file sender item and chooses a file “**a.txt**” containing the data “**qwerty**” → the MCU will receive “**file 5 6\na.txtqwerty**”, where **5** is the **length** of the filename “**a.txt**” and **6** is the **length** of the file data “**qwerty**”. If the file sender is configured not to include the filename, in that case the MCU will receive “**file 6\nqwerty**”. Keep in mind that the lengths (5 and 6 in this example) are the number of bytes in UTF-8 encoding, so that “**abcd**” has 4 Bytes, “**файл**” has 8 Bytes, “**ação**” has 6 Bytes, “**ţuică**” has 7 Bytes, “**フォン**” has 9 Bytes.
Label	Text string to appear under the item.
Label color	Color for the label text.
T.Size	Font size for the label text.
Include file name	Include the file name when sending the file. (see example at ID description).
Block size	Block size in Bytes. File will be sent in blocks of this size.
Block delay	Delay (in ms) the app. will wait before sending the next block (to give time for the microcontroller to process the received data).
Get next block cmd.	“Get next block” command. If non-empty, the app. will wait for this command before sending the next block. The microcontroller can send this when it finished processing the current block and it is ready for receiving the next one. Example: if file sender ID = “file” and GNB cmd. = “gnb” and interface cmd. ending = “\n” (without “”), in order to receive the next block, the microcontroller must send “file gnb\n” (without “”).
Cancel cmd.	Command used to cancel the file sending process. If the microcontroller does not want to receive more data from the file, it can send this command, and the app. will stop sending (after it finishes sending the current block). Example: if file sender ID = “file” and cancel cmd. = “cancel” and interface cmd. ending = “\n” (without “”), in order to cancel the file sending process, the microcontroller must send “file cancel\n” (without “”).

File receiver settings

setting	description
ID	ID string for the file receiver. Default ID is “**file**”. Here is an example for **ID** = “**file**” and **command ending** = “**\n**”: Microcontroller sends the string “**file 5 6\n**”. App will interpret the next 11 Bytes as following: first 5 Bytes will be the file name (in UTF-8), next 6 Bytes will be the file data. After receiving the 11 Bytes, the app will generate the file using the file name and data provided, and will save it to the storage folder. If the file with that name already exists, it will be overwritten. Example 2: Microcontroller sends the string “**file 8\n**”. App will interpret the next 8 Bytes as file data. After receiving the 8 Bytes, the app will generate the file using the data provided and the default file name, and will save it to the storage folder. Default file name is “file”. App will try to save the file under the name “file0”. If a file with this name already exists, app will use the name “file1”, “file2”, etc. until will find a name that is still not used. If default file name is “data.txt”, the generated file names will be “data0.txt”, “data1.txt”, “data2.txt”, etc. Keep in mind that the file name length is the number of bytes in UTF-8 encoding, so that “**abcd**” has 4 Bytes, “**файл**” has 8 Bytes, “**ação**” has 6 Bytes, “**ţuică**” has 7 Bytes, “**フォン**” has 9 Bytes.
Label	Text string to appear under the item.
Label color	Color for the label text.
T.Size	Font size for the label text.
Storage folder	Folder that will be used by the app to store the received files.
Default file name	Default file name to be used if a file name is not provided by the microcontroller. Default is “file”.

Speaker settings

setting	description
ID	ID string for the speaker. Example: ID = “spk”, command ending = “\n”, MCU sending “spk\n” will make it beep with the default frequency and duration. Sending “spk 4000 100\n” will make it beep with 4000 Hz for 100 ms.
Label	Text string to appear under the item.
Label color	Color for the label text.
T.Size	Font size for the label text.
Volume	Sound volume for the speaker. Default is 50%.
Sample rate	Sample rate for the speaker. Default is 44100Hz.
Default freq.	Default beep frequency. Default is 1000Hz.
Default duration	Default beep duration. Default is 500ms.
Waveform	Waveform (sine / square) for the beep. Default is square.

Printf() settings

setting	description
Config.	Configure the packet format. Examples: printf(“%02X\n”,s) → will get the integer value from the item with id “s” and will send it as a 2-digit upper case hex value, followed by LF. printf(“%c%c%c”,0xFF,s.B1,s.B0) → will send a 3-Byte packet: the first Byte will be 0xFF (255), the next 2 Bytes will be the Little-Endian integer value of the item with id “s”. The arguments after the “” in printf are the ids of the items, or constant values. Constant values can be **integers** (10, 255, 1024, ...), **hex integers** (0xFF, 0x3B), and **float numbers** (0.213, 10.7, ...). Adding **.Bx** after the id or constant will **select** the corresponding **Byte**. Examples: 0xAABBCC.B1 → 0xBB, 0x3F55AA00.B3 → 0x3F Currently the items that can be used with printf are: button, slider, accelerometer, heartbeat sender. The parser of the argument list ignores the characters **' ' (space)** and **'\n' (LF)**, so the **ids** of the items must **not contain** these **characters**, in order to work. Since **button** does not have a “set id” option, it has a **pseudo-id**, that is the common part of the press and release actions until the first space. Example: press action = “a 1”, release action = “a 0” → pseudo-id = “a”, value = “1/0” Since **heartbeat sender** has only id, its **value** will be always the constant **0**. **Adding more arguments than specifiers:** **This can be used, in order to trigger the printf without processing the value of that items.** Example: Add 2 sliders with ids “a” and “b”, add heartbeat sender with id “h”, add printf with config. printf(“%02X %02X\n”,a,b,h). The heartbeat sender will also trigger the printf, but the formatted packet will contain only the values from the sliders.
Trigger IDs	Set the ID(s) (separated by ',') that should trigger the printf. (They also must be included in the argument list of the printf in config.)
Text	Text that appears on the printf item. Default is “printf()”.
T.Size	Font size for the text that appears on the printf item.

Menu options

option	description
connect / disconnect	Connects to a remote device / disconnects from a remote device. To connect to a Bluetooth remote device, it has to be paired. To pair with a Bluetooth device, open Android settings → Bluetooth → search for devices. Pin code for Bluetooth to serial adapter is usually “1234” or “0000”. Select port 1 (if port selection is set to manual from RFCOMM settings). To connect via Internet or WiFi, data connection or WiFi has to be activated from Android settings. Then in RoboRemo app choose menu → connect → Internet and type / select the address and port. Address can be numerical IP or domain name.
edit UI / exit editor	Enters / exits the interface edit mode.
interface	Opens interface settings.
help	Shows help screen containing a download link for this User Manual.
about	Shows info about the app.
undo	Undo the last modification in the current interface.
settings	Open app. settings.

App. settings

setting	description
Autoconnect	Enables / disables the auto connect option. If enabled, the app will try to connect to the last connected device at next app start, if the user did not select disconnect before closing app. (App disconnects automatically when closed).
Lock autorotate	Locks / unlocks the screen autorotate function for the app.
Keep screen ON	Used to disable the automatic screen lock.
Char delay	Slow microcontrollers need time to process each character received via serial port. User can set the char delay so that RoboRemo will wait after each character sent. It is **strongly recommended** to use **char delay 0** when connecting via **UDP** since **UDP packets are not guaranteed to arrive in** the same **order** as they were send, and **setting char delay > 0 makes separate packets for each character.** **This is global char delay**. Starting with version 2.0.0, **each interface has its own char delay**. If the interface char delay is set to empty string (default), it will use this global char delay.
Disable hardware keys when connected: disable back key disable menu key	Options to disable the back and menu hardware keys when the app. is connected to a remote device, so that the used doesn't accidentally press them.
Back key function in editor	exit editor = pressing back key will exit editor. undo = pressing back key will undo.
RFCOMM settings:	Bluetooth Serial protocol (RFCOMM) settings
port (channel) selection	manual = will ask each time fixed(1) = will use port 1 auto (default) = will select automatically
UUID selection	Used only if port selection = auto. Ignored otherwise. default = use the well known UUID for Bluetooth to Serial modules, which is 00001101-0000-1000-8000-00805F9B34FB first = use first UUID of the remote device, from the local cache
RFCOMM encrypted	Enable / disable Bluetooth RFCOMM encryption. (default = disabled)
BLE Settings	Settings for connecting via Bluetooth Low Energy (BLE)
Service selection	manual = Manually select BLE Service automatic (default) = Automatically select a known UART Service. If UART Service not found, will try to select first unknown (custom) Service. If

	custom Service not found, will select first found Service.
Characteristic selection	manual = Manually select BLE Characteristics automatic (default) = Automatically select BLE Characteristics.

Note: Some Bluetooth devices may fail to connect with certain settings. In this case it is recommended to try different settings.
Note2: BLE connection is still experimental, it may cause app crash / ANR.

Interface options and settings

option	description
Select	Select other interface. RoboRemo has 8 interfaces, numbered 0 to 7.
Name	Name of the current interface.
Connect action	Command string to be sent once to the remote devices after connecting.
Command ending	Command ending for the current interface. The default is “\n”. Command ending can also be set to empty string, but it is not recommended.
Import	Import an interface from file.
Export	Export the current interface to a file.
Share	Share the current interface file. (Requires Android 5.1 and up).
Unlock edit all	Unlock all items that were locked with lock edit.
Background color	Background color for the interface. Format is ARGB_8888 (hex code). Examples: FFFFFFFF = white, FFD891EF = Bright Lilac, FF92A1CF = Ceil.
Clear	Clear the current interface.
Touch processing	Touch processing mode selection for this interface. Default = nearest item. exact = the exact touched item receives the touch. Also the “touch stopper” item disappears from the “add item” list as it is not needed in this mode. nearest = the closest item receives the touch.

New features and fixed bugs

App. version	New features	Fixed bugs
1.2	- accelerometer support - Internet / WiFi connectivity - option for sliders to return to center when released.	N/A
1.3	- option to change command ending - string inputs from user are interpreted for escape characters ('\t', '\b', '\n', '\r','\f'). For '\' character, enter '\\'.	- fixed interface import / export for interfaces with non-ASCII characters. - fixed text wrapping and auto scrolling when resizing text logs.
1.4	- local action “interface n” same as “iface n” where n is the interface number (0 to 7) - auto connect option - local actions for connecting and disconnecting by pressing buttons. - removed ”exit” from menu (app can be closed by pressing back key).	- fixed the bug with “\r\n” string in interfaces. - fixed a bug where app crashed if n was not an integer in “iface n” or “interface n” local action.
1.5	- option to use the slider or level indicator value inside their labels - made slider with "send when moved" send only when the value changes - option to change on and off commands for leds - removed the dialog showing error when sending a non-integer value to a level indicator - undo option in interface editor.	- fixed a bug in processing received commands. Now if there are more items with same id, all of them will respond to it.
1.6	- added option to set repeat period for sliders - added plots and text fields - improved touch processing algorithm so that it will send touch events only to interactive items (now user can have buttons / sliders hidden under leds, plots, etc) - added menu option to change RFCOMM settings	- fixed a bug in sliders where slider with auto return enabled did not send data when it was the same value as before return.
1.7	- added option to set text size for text fields and text logs - added option to set id for text field so that user can change the text remotely - added image item in RoboRemo Full version	- when command ending is empty string, do not append “\n” at the end in text logs - fixed a bug where autoconnect dialog could not be canceled after

	- reduced TCP connect timeout from 90000ms to 5000ms - improved menus (now they are scrollable) - added touchpad item, kbd connector item - added option to keep screen on	screen orientation change - interface → clear now also resets the command ending and connect action
1.8	- added item option to lock edit and interface option to unlock edit all - added option to set return value for slider - added option to set led on/off timeout - added option to import image from SD card and store it in RoboRemo interface - added items: heartbeat sender, touch stopper - changed default RFCOMM settings to automatic port selection - added option to start a TCP server - can connect to domain name - added option to use plot's last value inside the label	N/A
1.9	- added image option to set as background - added USB connectivity - added “don't repaint” option for plots - hw menu button also opens menu	- fixed bug with text size on text log and text field items - fixed bug with pasting an item after lock edit - fixed small menu bug - fixed touchpad bug where it was sending data even if id was empty string
1.9.1	- added UDP connectivity - added “log to file” option (in RoboRemo Full version) for text log and plot. - added vibrator item in RoboRemo Full version - added option for accelerometer to set gain, output type (float / int), min. and max. values and enable / disable the value limiting (replaced “set x/y/z id” with “x/y/z config.“). - added local action “send” - now local action can contain multiple actions, separated by command ending - added option in slider, level indicator and plot items label to set the decimal count. - slider and level indicator items now accept setting min. value greater than max. value and behave as expected.	- fixed bug in USB connection where data sent too fast from microcontroller was not received correctly by the app. - fixed USB autoconnect and connect by local action. - fixed text log bug where it was stalling on some devices after receiving about 10000 characters. - clearing an interface now clears the interface name too. - fixed float numbers in exported interface files. - fixed a bug where app was crashing when trying to connect via USB on devices that don't have USB API. - on slider item, now selecting autoreturn or changing the return

	- improved some user interface (in RFCOMM settings, image import, plot set trigger). - now the options that are available only in RoboRemo full version, are shown (disabled) in RoboRemoFree too. - added “100% FREE” mark on the icon of RoboRemoFree	value updates the position of the handle. - fixed default aspect ratio of heartbeat sender item.
1.9.2	- added hints in UiEditor - added links in menu → about - added option to set the line width for plots - added option to set text size for plot, slider and level indicator labels, and buttons. - added file sender and file receiver items in RoboRemo Full version	- Increased maximum size of UDP rx packet from 8192 to 65507 Bytes
1.9.3	- file sender and receiver glowing when active - removed the delay before opening streams for TCP - added options to set block size and block delay for file sender	- fixed some bugs in file sender and file receiver, now they should work OK with files up to 2GB
1.9.4	- easier setup for local action connect - added local actions sendnce, sendhex - added speaker item - added option to set button images - added option to repeat only the press action at button repeat - added option for slider, level indicator and plot to work with float numbers	- fixed bug with '.' in command ending - fixed bug where slider was not displaying correctly in Android 6 - fixed bug with negative values in slider
2.0.0	- moved all app. settings to menu → settings - added option to disable back and / or menu hardware keys when connected - added printf item that gets data from other items and user can configure the format of the packet that will be sent - Bluetooth made optional, now users should be able to install the app. on devices that don't have Bluetooth hardware. - added option to set the char delay for each interface separately (there is also a global char delay in menu → settings, that will be used by default, if the interface char delay is set to empty string).	- fixed touchpad bug, now it should work with negative values too. - fixed touchpad bug where it was sending multiple times the same value. - undo option (if undo is available) now appears in menu only in edit mode. - fixed bug where items did not send data in Android 7 (not tested) - fixed bug where some items (like heartbeat sender, accelerometer) were sending data before the connect action.

	- added file sender options “set GNB cmd.” (get next block) and “set cancel cmd.” (cancel sending). - added option (in menu → settings) to set the editor back key function to undo instead of exiting editor. - added option to change the text and background color for text field. - added option to change the background color of the interfaces. - added some hints and some syntax checking in the edit options of the items. - changed the default (disabled) value from “0” to empty string for some options, to be more intuitive. Old version interfaces are updated automatically, but new interfaces may not work correctly in old app. versions, so make sure you have the latest app. version.	
2.0.1	- added USB support for Microchip CDC devices (VID 0x04D8) - changed help link from Dialog to TextDisplay	- changes made with “set connect action” and “interface rename” now will persist in case of crash. - fixed bug where app was crashing when setting LED off timeout to empty string
2.0.2	- added Bluetooth Low Energy (BLE) support - improved USB write speed - improved UDP write speed	N/A
2.1.0	- changed USB library - added support for CH340/HL-340 USB to Serial chip found in many Arduino clones. - added more high-speed baud rates for the chips that support them.	- added “connection lost” detection for FTDI chips (when removing from USB while connected).
2.1.1	N/A	- fixed some bugs in the USB library
2.1.2	N/A	- fixed bug where clearing an interface was not clearing its char delay. - fixed BLE bug where interface's connect action was causing a connection drop.
2.1.3	N/A	- fixed some bugs.
2.1.4	Updated targetSdkVersion to 26.	- fixed some bugs

2.1.5	Updated targetSdkVersion to 29.	fixed UI bugs: - added paint fill for UI text - select all when editing field
2.1.6	N/A	- fixed some bugs
2.2.0	- code cleanup & some optimizations - options added in menu → about: - share this app - links to related apps you may like - interface →share (Android 5.1 and up) - experimental promotion in RoboRemoFree : menu →free full version (Android 5.0 and up) (instructions to get the full version for free)	- fixed file access for Android Q
2.3.0	- improved some items appearance - redesigned items settings screen (now all settings appear in one screen – easier to see and faster to configure) - image item can now import more than one image, currently displayed image can be selected by command.	- fixed some bugs
2.3.1	- connection establishment process redesign - UI improvements	- fixed issue with app hanging in “Please wait...” - fixed bug related to disabling button custom image - fixed BLE issues - fixed some other bugs
2.3.2	- UI improvements - added refresh button to BLE manual Service Selection - TCP Server now displays server IP	- fixed more BLE issues - fixed UI bugs
2.3.3	- code cleanup & some optimizations	- fixed some bugs
2.3.4	- now items can be copied from one interface and pasted to another one - added interface setting to select touch processing mode (exact / nearest item)	- fixed “item paste” bug - fixed some bugs in the USB Serial connection
2.3.5	- removed the “free full version” experimental promotion - temporary removed links to other apps in	N/A

	“About” screen, as that view requires redesign with more info about the advertised apps.	
2.4.2	- set targetSdkVersion 31 - redesigned app launcher icons - RoboRemoFree renamed to RoboRemoDemo - rewritten parts of the app code for better maintainability. - improved menu design	N/A
2.4.3	- set targetSdkVersion 33 - updated “about” window	N/A
2.4.4	- improved parsing for the input field where the user enters address:port	- fixed bug where the app could not turn the Bluetooth ON and was stalling with the “please wait...” dialog.
2.4.5	N/A	- fixed bug introduced in v2.4.2 where status bar was displayed on top of the interface on some devices and was not hiding.
2.5.0	- separate input fields for address and port for TCP and UDP connection - color selector for faster color input across the app - extended color options for level indicators and LEDs - custom line color for plots - custom color and text size for items labels - custom color for button text - new UI item: joystick - added division (/) support in interactive labels (of level indicator / slider / plot) - added multi-line support in button text and item labels - “Cancel” and “OK” on app buttons replaced with “X” and checkmark.	- fixed bug in USB drivers. More boards are supported now, including ST-LINK/V2-1 found on STM32 Nucleo-64 boards.