RDRIVE SERVOMOTOR



QUICK-START GUIDE

Getting Started with RDrive

The document describes how to conduct *an unloaded test run* to get started with an RDrive motor, using **Python or C Servo API** by Rozum Robotics and a PC running under one of the operating systems:

- Linux OS
- Windows OS

Whether with Python or C, Linux or Windows, getting started with an RDrive takes three easy steps:

Step 1. Preparing

Step 2. Connecting the hardware

Step 3. Enabling servo control via API

Linux OS + Python Windows OS + Python OS Linux + C OS Windows (Cygwin) + C

Step 1. Preparing

The step is the same for both operating systems, as well as for Python and C languages.

1. Unpack your servo supply package. It should contain:

- an RDrive servo motor
- a CAN-USB dongle
- a Micro USB-USB A cable

For the test run, provide additionally:

- a 48 V power supply unit
- a motor mount to hold the servo stationary

The equipment and hardware connection instructions are FOR UNLOADED TEST RUNS ONLY!

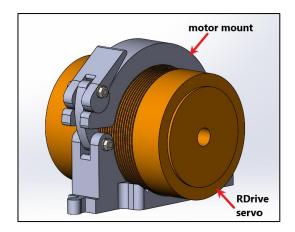


For loaded servo operation, you need to add at least one energy eater and one capacitor to the supply circuit, or use the servobox solution by Rozum Robotics. For details, refer to the <u>Servobox documentation</u>.

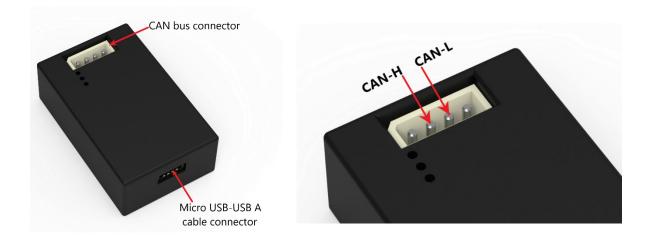
Step 2. Connecting the hardware

Just as the preceding one, the step is the same for both operating systems, as well as C and Python languages.

1. Fix the servo in place in a motor mount.



- 2. Connect the servo to the power supply unit, using the red and black wires. The red one is for "+," and the black one is for "-."
- 3. Connect the servo to the CAN bus connector on the CAN-USB dongle, using the blue and brown wires. The brown one is for CAN_{HIGH}, and the blue one is for CAN_{LOW}.



- 4. Connect the Micro USB-USB A cable to the Micro USB-USB A cable connector on the CAN-USB dongle and the USB port on your computer.
- 5. Plug the power supply connector into a power supply socket.

Now, the servo is ready for programming via API.

Step 3. Enabling servo control via API

Linux OS + Python

Prerequisites:

• Python (3.5 or a later version)

To install Python, run one of the following commands in the console:

- Ubuntu/Debian:

sudo apt install python3 python3-pip

- Fedora version 22 and higher:

sudo dnf install python3 python3-pip

- RedHat and Fedora before version 22:

sudo yum install python3 python3-pip



In case you have a Linux distribution other than specified above, proceed in the usual order of running the installation procedure in your system.

- pip 3 package (installed together with Python as described above)
- gcc MinGW compiler
- make package

Installation and setup sequence:

- 1. Install the Python API library by running one of the following commands.
 - To install the latest version, use:

pip install rdrive -i https://pip.rozum.com/simple

• To install a specific version of your choice, use:

```
pip install rdrive==v1.v2.v3 -i https://pip.rozum.com/simple
```

where v1, v2, and v3 (e.g., rdrive==1.0.31) are version numbers.

2. Find out the CAN interface name — the identifier of the CAN-USB dongle in the user environment. To do this, open the console and type in the following command: ls /dev/serial/by-id/.

In the output, look for something like: usb-Rozum_Robotics_USB-CAN_Interface_301-if00. It is the CAN interface name you need.

- 3. Find out the CAN ID of the connected servo motor. To do this, complete the steps as described below.
 - In the console, run any tutorial from the examples folder. To do this, execute the below command, specifying the CAN Interface ID from Step 2 and a random CAN Servo ID.

python3 ..\userapi\python\examples\read_servo_max_velocity.py -interface /dev/serial/by-id/usb-Rozum_Robotics_USB-CAN Interface 301-if00 --servo 1 id 32

where read_servo_max_velocity.py is the tutorial name;



If needed, replace the tutorial name from the current example with any other from the list of available <u>servo API tutorials</u>.

interface/dev/serial/by-id/usb-Rozum_Robotics_USB-

CAN_Interface_301-if00 is the parameter specifying the CAN Interface ID (the one we got at Step 2);

servo_1_id 37 is the parameter specifying a random CAN ID for the servo.

• In the command output, go to the INFO lines (see the example below). Look for IDs within the range between 32 to 37 — default servo IDs (37 in the example below).

Example:

INFO: ID: 50 Device is in operational mode

INFO: ID: 37 Device is in pre-operational mode

4. Now, you can run any tutorial from the examples folder to move your RDrive servo or read parameters from it.

To do this, run the following command in the console:

python3 ..\userapi\python\examples\read_servo_max_velocity.py -interface /dev/serial/by-id/usb-Rozum_Robotics_USB-CAN_Interface_301-if00 --servo_1_id 37

where read_servo_max_velocity.py is the tutorial name



If needed, replace the tutorial name from the current example with any other from the list of available <u>servo API tutorials</u>.

interface/dev/serial/by-id/usb-Rozum_Robotics_USB-CAN_Interface_301if00 is the parameter specifying the CAN Interface ID (the one we got at Step 2)

servo_1_id 37 is the parameter specifying the CAN ID of the connected servo (the one we got at Step 3).

If the command is executed successfully, the connected servo will behave as commanded—return parameters or move.



Some of the tutorials may require specifying other parameters in addition to the CAN Interface ID and CAN Servo ID.

Now that you know how to start the servo and use the tutorials, you can create your own code. For more information, refer to the <u>Python API documentation</u>.

Windows OS + Python

Prerequisites:

• Python (3.5 or a later version)

To install Python, navigate to Python, download, and complete the setup process.



At the installation setup screen, make sure to check Add Python v.X to PATH (where v.X. is the downloaded Python version).



If your PC runs Windows 8 or earlier versions of the operating system, download and install a driver for the CAN-USB dongle from <u>the web page</u>.

Installation and setup sequence:

- 1. Install the Servo API library by running one of the following commands.
 - To install the latest version, use:

pip install rdrive -i https://pip.rozum.com/simple

• To install a specific version of your choice, use:

pip install rdrive==v1.v2.v3 -i https://pip.rozum.com/simple

where v1, v2, and v3 (e.g., rdrive==1.0.31) are version numbers.

- 2. Open the command line interface via the **Start menu**.
- 3. Find out the CAN interface name—the identifier of the CAN-USB dongle in the user environment:
 - a. On your computer, open Start menu \rightarrow System Settings \rightarrow Devices \rightarrow Bluetooth & other devices.
 - b. On the **Other devices** list, find a serial USB device with a COM port. The COM port number (e.g., COM 3) is the CAN interface ID you need.

← Settings		- 0 ×
ය Home	Bluetooth & other devices	
Find a setting	+ Add Bluetooth or other device	Turn on Bluetooth even faster To turn on Bluetooth without opening Settings, open action center, and then select the
Bluetooth & other devices	Other devices	Bluetooth icon. Do the same to turn it off when you want.
母 Printers & scanners		Get more info about Bluetooth
① Mouse	USB Serial Device (COM3)	Related settings
Touchpad		Devices and printers
III Typing	Download over metered connections	Sound settings Display settings
🖉 Pen & Windows Ink	To help prevent extra charges, keep this off so device software	
AutoPlay	(drivers, info, and apps) for new devices won't download while you're on metered Internet connections.	Have a question? Get help
☐ USB		Make Windows better Give us feedback

- 4. Find out the CAN ID of the connected servo:
 - a. In the command console, run any tutorial from the examples folder. To do this, execute the below command, specifying the CAN Interface ID from Step 3 and a random CAN Servo ID.

python ...\userapi\python\examples\read_servo_max_velocity.py -interface COM3 --servo_1_id 32

where read_servo_max_velocity.py is the tutorial name



If needed, replace the tutorial name from the current example with any other from the list of available <u>servo API tutorials</u>.

interface COM3 is the parameter specifying the CAN Interface ID from Step 3

servo_1_id 37 is the parameter specifying a random CAN ID for the servo

b. In the command output, go to the INFO lines (see the example below). Look for IDs within the range between 32 to 37 — default servo IDs (37 in the example below).

Example:

INFO: ID: 50 Device is in operational mode

INFO: ID: 37 Device is in pre-operational mode

5. Now, you can run any of the tutorials from the examples folder to move your RDrive servo or read parameters from it.

To do this, run the following command in the command line interface:

```
python ...\userapi.python.examples.read_servo_max_velocity.py --
interface COM3 --servo_1_id 37
```

where read_servo_max_velocity.py is the tutorial name



If needed, replace the tutorial name from the current example with any other from the list of available <u>servo API tutorials.</u>

interface COM3 is the parameter specifying the CAN Interface ID (the one we got at Step 3)

servo_1_id 37 is the parameter specifying the CAN ID of the connected servo (the one we got at Step 4)

If the command is successfully executed, the connected servo will behave as commanded—return parameters or move.



Some of the tutorials may require specifying other parameters in addition to the CAN Interface ID and CAN Servo ID.

Now that you know how to start the servo and use the tutorials, you can create your own code. For more information, refer to the <u>Python API documentation</u>.

OS Linux + C

Prerequisites:

- gcc compiler
- libpthread library
- make package
- 1. Download Rozum-Servo-Drives-API from the GitHub repository.

File Edit View History Bookm	marks <u>T</u> ools <u>H</u> elp		GitHub - roz	um-robotics/Rozum-Servo-Dr	ives-API - Mozilla Fire	efox (Private Brow	sing)		e e e
	③ GitHub, Inc. (US) https://	//github.com/rozu	n-robotics/Rozum-Ser	vo-Drives-API				⊚ ☆	li\ ⊡ 🐁 =
U - L			sues o 🕺 🕅 Pull requ		Insights				
				itHub is home to over 31 million of and review code, manage proje			Dismiss		
		No description, we	bsite, or topics provid	led. ⊮ 1 branch	© 0 relea:	ises	1 7 contributors		
		Branch: master 🕶	New pull request				Find file Clone or download *		
		Matkou Merge br	anch 'dev' into 'master'			Clone with HTT	PS @		
		iii bin	'update-tool' as	debian package			with SVN using the web URL.		
		iii c	Merge branch 'fe	eature/EMB-1107' into 'dev'		https://github.o	com/rozum-robotics/Rozu		
		debian	'update-tool' as	debian package			Download ZIP		
		debug-tools	[*]: debug-tools:	linewidth setting removed			Jowinoad Zir		
		illi java	[*]: high level AP	1 template			8 months ago		
		python	Add get_state()	method to Servo class			3 days ago		
		ill ruby	[*]: high level AP	1 template			8 months ago		
		.gitignore	EMB-944				2 months ago		
		.gitlab-ci.yml	[*]: tutorial: put s	ervo into operational mode, dupl	icate tutorial ou		3 days ago		
		README.md	Update READM	E.md			8 months ago		
		E README.md							
https://github.com/rozum-robotics/Ro	ozum-Servo-Drives-API/archive/		n-Servo-D	rives-API	- de d 4- e mel 1- e 4				

- 2. Unzip the downloaded file.
- 3. In the console, run the cd c command to navigate to the c folder.

- 4. In the console, execute the make clean; make command to compile the library. In the output, apart from the rest, you get the two files:
 - a. build/libservo_api.so

b. build/libservo_api.a

```
~/work/test/Rozum-Servo-Drives-API-master/c # make clean; make
rm -fr build
[C] src/rb_tools.c ...
[C] src/usbcan_types.c ...
[C] src/logging.c ...
[C] src/co_common.c ...
[C] src/co_common.c ...
[C] src/crcl6-ccitt.c ...
[C] src/usbcan_proto.c ...
[C] src/usbcan_proto.c ...
[L] build/libservo_api.a ...
[RL] build/libservo_api.a ...
[RL] build/libservo-Drives-API-master/c #
```

5. Run the cd tutorial command to navigate to the tutorial folder.

~/work/test/Rozum-Servo-Drives-API-master/c # cd tutorial/ ~/work/test/Rozum-Servo-Drives-API-master/c/tutorial #

6. Run make. As a result, you get the build folder with executable tutorial files.

```
~/work/test/Rozum-Servo-Drives-API-master/c/tutorial # make
make -C ..
make[1]: Entering directory '/mnt/hdd/home/work/test/Rozum-Servo-Drives-API-master/c'
make[1]: Leaving directory '/mnt/hdd/home/work/test/Rozum-Servo-Drives-API-master/c'
make -f tutorial.mk -C .
make[1]: Entering directory '/mnt/hdd/home/work/test/Rozum-Servo-Drives-API-
master/c/tutorial'
make -C ..
make[2]: Entering directory '/mnt/hdd/home/work/test/Rozum-Servo-Drives-API-master/c'
make[2]: Leaving directory '/mnt/hdd/home/work/test/Rozum-Servo-Drives-API-master/c'
mkdir -p build
gcc change servo id.c -g2 -I../include -o build/change servo id
../build/libservo api.a -static -lpthread -lm
gcc read errors.c -g2 -I../include -o build/read errors ../build/libservo api.a -
static -lpthread -lm
gcc read_servo_trajectory_time.c -g2 -I../include -o build/read_servo_trajectory_time
../build/libservo api.a -static -lpthread -lm
gcc read any param.c -g2 -I../include -o build/read any param ../build/libservo api.a
-static -lpthread -lm
gcc control_servo_traj_1.c -g2 -I../include -o
build/control servo traj 1../build/libservo api.a -static -lpthread -lm
```

```
gcc time optimal movement.c -g2 -I../include -o build/time optimal movement
../build/libservo api.a -static -lpthread -lm
gcc read servo max velocity.c -g2 -I../include -o build/read servo max velocity
../build/libservo api.a -static -lpthread -lm
gcc read emcy log.c -g2 -I../include -o build/read emcy log ../build/libservo api.a -
static -lpthread -lm
gcc calibrate_cogging.c -g2 -I../include -o build/calibrate cogging
../build/libservo api.a -static -lpthread -lm
gcc hb timings.c -g2 -I../include -o build/hb timings ../build/libservo api.a -static
-lpthread -lm
gcc read any param cache.c -g2 -I../include -o build/read any param cache
../build/libservo api.a -static -lpthread -lm
gcc calibration quality.c -g2 -I../include -o build/calibration quality
../build/libservo_api.a -static -lpthread -lm
gcc control_servo_traj_2.c -g2 -I../include -o build/control_servo_traj_2
../build/libservo api.a -static -lpthread -lm
gcc control servo traj 3.c -g2 -I../include -o build/control servo traj 3
../build/libservo_api.a -static -lpthread -lm
gcc read servo motion queue.c -g2 -I../include -o build/read servo motion queue
../build/libservo api.a -static -lpthread -lm
make[1]: Leaving directory '/mnt/hdd/home/maksimkatkou/work/test/Rozum-Servo-Drives-
API-master/c/tutorial
~/work/test/Rozum-Servo-Drives-API-master/c/tutorial #
```

7. Navigate to the build folder and run an executable file of any tutorial.

```
~/work/test/Rozum-Servo-Drives-API-master/c/tutorial # cd build/
~/work/test/Rozum-Servo-Drives-API-master/c/tutorial/build #
```

Since you have set no specific parameters for the executable file, you will get a response with a 'Wrong format' warning. However, the USAGE section of the same response will illustrate the correct format for setting up the parameters for the specific tutorial.

```
~/work/test/Rozum-Servo-Drives-API-master/c/tutorial/build #
./read_any_param_cache
Wrong format!
Usage: ./read_any_param_cache interface id
~/work/test/Rozum-Servo-Drives-API-master/c/tutorial/build #
```

- 8. Find out the CAN Interface ID the identifier of the used USB-CAN dongle:
 - a. Open the **Device Manager**. In the device list, open the **Ports (COM & LPT)** dropdown list. Right-click any device on the list and select **Properties** in the context menu.

📇 Device Manager
File Action View Help
▼ apiatlitski > Audio inputs and outputs > Batteries > Bluetooth > Computer > Disk drives > Disk drives > Display adapters > IDE ATA/ATAPI controllers > Imaging devices > Keyboards Monitors > Network adapters
Image: Weight of the series Update driver Image: Weight of the series Update driver Image: Weight of the series Disable device Image: Weight of the series Update driver Image: Weight of the series

b. In the **Properties** window, go to the **Details** tab, open the dropdown list of the **Property** field, and select **Bus reported device description.**

eral Port Settings Driver Details Events USB serial device (COM 3) poperty evice description		
operty K		
operty K		
operty K		
evice description		
	~	
evice description	~	
evice instance path		
ardware lds		
ompatible lds		
apabilities		
atus		
oblem code		
onfig flags		
ass		
ass Guid f name		
iver date		
iver version		
ovider		
f section		
f section extension		
atching device Id		
ank of driver		
ervice		
dress		
OS device name		
ase container Id		
us number		
us reported device description		
us type Guid		
ontainer Id		
evice stack		
splay name		
iver description iver kev	h.	

- c. On the same tab, look at the Value field.
 - If it contains **USB-CAN Interface**, this means the selected device is the USB-CAN you need. Use the port number of the device (in this case, **COM 3**) as the CAN Interface ID.
 - If the field contains some other value, go back to **step a** and repeat the same sequence for another device on the **Device Manager** list.

USB serial device (COM3)			×
General Port Settings Driver	Details	Events	
USB serial device (CC)M3)		
Property			
Bus reported device description	n		\sim
Value			
USB-CAN Interface			
		OK	Cancel

- 9. Find out the CAN ID of your servo. To do this, run the **discovery tutorial** from the build folder. In the output of the tutorial, you will find the CAN ID of the connected servo.
- Run any of the tutorials with preset parameters the CAN Interface ID from Step 8, the CAN Servo ID from Step 9, and other parameters in the format as prompted in Step 7.

```
~/work/test/Rozum-Servo-Drives-API-master/c/tutorial/build #
./read_any_param_cache /dev/ttyACM0 123
WARN: Emergency frame received: id(123) code(0x0) reg(0x80) bits(0x45)
info(0x0):
    [Error Reset or No Error, Heartbeat consumer timeout]
INFO: ID: 123 Device is in pre-operational mode
===== Tutorial of programming and reading the device parameter cache ======
    APP_PARAM_POSITION_ROTOR value: 137.289
    APP_PARAM_VELOCITY_ROTOR value: 0.069
    APP_PARAM_VOLTAGE_INPUT value: 47.327
    APP_PARAM_CURRENT_INPUT value: 0.000
```

Now that you know how to start the servo and use the tutorials, you can create your own code. For more information, refer to the <u>API documentation</u>.

OS Windows (Cygwin) + C

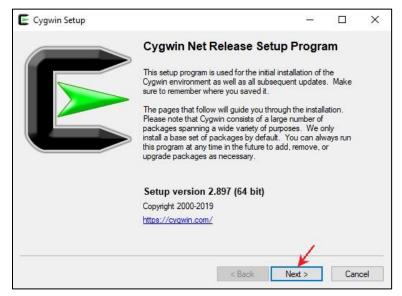
Prerequisites:

- Cygwin
- gcc compiler
- libpthread library
- make package



If your PC runs Windows 8 or earlier versions of the operating system, download and install a driver for the CAN-USB dongle from <u>the web page</u>.

- 1. Follow the link to download Cygwin.
- 2. Run the executable file. In the setup screen, click Next to start installation.



3. Choose a source for downloading and click Next.

Cygwin Setup - Choose Installation Type	_		×
Choose whether to install or download from the internet, or install from files in a local directory.			
 Install from Internet (downloaded files will be kept for future re-use) 			
O Download Without Installing			
O Install from Local Directory			
< Back Nex	t >	Car	ncel

4. Select a directory for installing the files. Leave the default one or browse to set up another installation path. Click **Next**.

E Cygwin Setup - Choose Installation Directory - C X	
Select Root Install Directory Select the directory where you want to install Cygwin. Also choose a few installation parameters.	
Root Directory	
Install For	
All Users (RECOMMENDED)	
Cygwin will be available to all users of the system.	
⊖ Just Me	
Cygwin will still be available to all users, but Desktop Icons, Cygwin Menu Entries, and important Installer information are only available to the current user. Only select this if you lack Administrator privileges or if you have specific needs.	
< Back Next > Cancel	

5. Select a local directory and click **Next**.

Cygwin Setup - Select Local Package Directory	_		Х
Select Local Package Directory Select a directory where you want Setup to store the installation files it downloads. The directory will be created if it does not already exist.			
Local Package Directory			
C:\Users\Downloads		Browse	
< Back	Next >	Can	cel

6. Select a connection type and click **Next.**

E Cygwin Setup - Select Connection Type			×
Select Your Internet Connection Setup needs to know how you want it to connect to the internet. Choose the appropriate settings below.		l	Å
 Use System Proxy Settings Direct Connection Use HTTP/FTP Proxy: 			
Proxy Host Port 80			
< Back Ne	xt >	Can	

7. Choose a source site for downloading. Click Next.

	wnload Site ite from this list, or add your own sites to the list			
	Available Download Sites:			
	http://cygwin.mirror.constant.com http://cygwin.mirrors.hoobly.com	^		
	http://mirrors.koehn.com http://mirrors.metapeer.com http://www.pirbot.com			
	http://mirror.team-cymru.com ftp://mirrors.xmission.com			
	http://mirrors.xmission.com http://mirror.clark.son.edu			
	http://mirror.cs.vt.edu http://mirror.koddos.net			
	http://mirror-hk.koddos.net ftp://mirrors.netix.net	~		
User URL:			Add	
User URL:			Add	

8. In the **View** field, select **Full.** Then, select **MinGW**, **make**, and **gcc** packages to install as shown in the figures below:

Select Packages Select packages to install			
∕iew Full ✔ ∨ Search ^{make}	◯ Keep	Sync	Test
Package	Current New		
libpagemaker0.0-devel	Skip		
libpagemaker0.0-doc	Skip		
libpagemaker0.0_0	Skip		
make	4.2.1-2		
make-debuginfo	Skip		
makedepend	Skip		
makedepend-debuginfo	Skip		
makepasswd	Skip		
makeself	Skip		
<			>
Hide obsolete packages			

S	elect packa	ges to ins	tall						
View	Full	~	Search	gcc	Кеер	Best	⊖ Sync	Te	st
Pac	kage				Current	New			^
gcci	makedep					Skip			-
libgo	:c1					7.4.0-1			
libgo	cpp1					Skip			
ming	w64-i686-g	cc-core				7.4.0-1			
ming	w64-i686-g	cc-debug	info			Skip			
ming	w64-i686-g	cc-fortran				Skip			
ming	w64-i686-g	cc-g++				Skip			
ming	w64-i686-g	cc-objc				Skip			
mino <	1w64-x86 6	4-qcc-co	e			Skip		>	~
	de obsolete								

View	Full	✓ Search	pthread	Clear	Best	◯ Sync	Test	
Pac	kage			Current	New			
libsu	ndials_nvecpth	nreads2			Skip			
ming	jw64-i686-pthre	ads			20100619	-5		ľ
ming	jw64-i686-winpi	threads			6.0.0-1			
ming	w64-i686-winpi	threads-debugin	fo		Skip			
ming	w64-x86_64-pt	hreads			Skip			
ming	w64-x86_64-w	inpthreads			Skip			
ming	jw64-x86_64-w	inpthreads-debu	ıginfo		Skip			
<								>

9. Wait for the setup to complete. In the final setup screen, select the checkboxes to create desktop and **Start Menu** icons. Click **Finish** to quit setup.

Cygwin Setup - Installation Status and Create Icons	<u></u> 8		>
Create Icons Tell setup if you want it to create a few icons for convenient access t Cygwin environment.	o the	1	
4			
✓ Create icon on Desktop ✓ Add icon to Start Menu			
Installation Status Installation Complete			
	1		
	Finish	-	ncel

@ 562 con	mits P 1 branch	♥ 0 releases	41 7 contributors	
Branch: master •	ew pull request		Find file Clone or download *	
Matkou Merge bran	ch 'dev' into 'master'	Clone wit	th HTTPS 💿	
iin bin	'update-tool' as debian package	Use Git or o	checkout with SVN using the web URL.	
iii c	Merge branch 'feature/EMB-1107' into 'dev'	https://	github.com/rozum.robotics/Rozu	
ille debian	'update-tool' as debian package		Download 70	
debug-tools	[*]: debug-tools: linewidth setting removed		Download ZIP	
ille java	[*]: high level API template		8 months ago	
in python	Add get_state() method to Servo class		3 days ago	

10. Download Rozum-Servo-Drives-API from the GitHub repository.

- 11. Unzip the downloaded file to the folder C/Cygwin/home/user name.
- 12. Double-click the **Cgwyn** icon to start the console.
- 13. In the console, run the cd c command to navigate to the c folder.



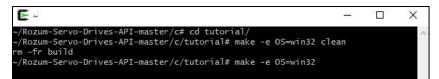
14. In the console, execute the make clean; make commands to compile the library. Make sure to add -e OS = win32 (see figure below).

E ~	-	×
~#		^
~#		
~#		
~#		
~# ~# cd Rozum-Servo-Drives-API-master/		
~/Rozum-Servo-Drives-API-master# cd c		
~/Rozum-Servo-Drives-API-master/c# make -e OS=win32 clean		
rm -fr build-win-32bit		
~/Rozum-Servo-Drives-API-master/c# <u>make -e OS=win32</u> [C] src/usbcan_proto.c		
[C] src/api.c		
[C] src/crc16-ccitt.c		
[C] src/co_common.c		
<pre>[C] src/usbcan_util.c</pre>		
[C] src/logging.c		
<pre>[C] src/usbcan_types.c</pre>		
[C] src/rb_tools.c		
<pre>[L] build-win-32bit/libservo_api-32bit.dll [AR] build-win-32bit/libservo_api-32bit.a</pre>		
[RL] build-win-32bit/libservo_api-32bit.a		
<pre>~/Rozum-Servo-Drives-API-master/c#</pre>		~

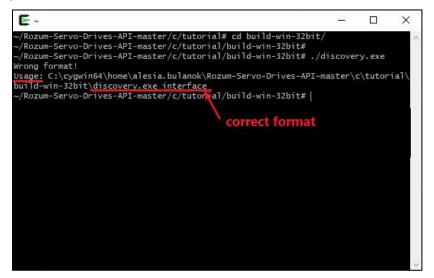
Apart from the rest, the command output will contain the following two files:

- c. build/libservo_api.dll
- d. build/libservo_api.a
- 15. Run the cd tutorial command to navigate to the tutorial folder.

16. Run make. As a result, you get the build folder with executable tutorial files.

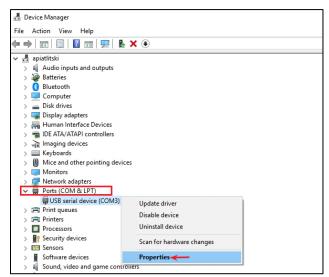


17. Navigate to the build folder and run an executable file of any tutorial without setting command parameters.



Since you have set no specific parameters, you will get a response with a 'Wrong format' warning. However, the USAGE of the same response will illustrate the correct format for setting up the parameters in the specific tutorial.

- 18. Find out the CAN Interface ID the identifier of the used USB-CAN dongle.
 - d. Open **Device Manager**. In the device list, open the **Ports (COM & LPT)** dropdown list. Right-click any device on the list and select **Properties** in the context menu.



e. In the property window, go to the **Details** tab, open the dropdown list of the **Property** field, and select **Bus reported device description.**

USB serial device (COM 3)	×
General Port Settings Driver Details Events	
USB serial device (COM 3)	
Device description	\sim
Device instance path Hardware Ids Compatible Ids Capabilities Status Problem code Config flags Class Class Class Guid Inf name Driver version Provider Inf section Inf section Inf section Inf section Matching device Id Rank of driver Service Address BIOS device name Base container Id Bus number	
Bus reported device description	
Bus type Guid Container Id Device stack Display name Driver description	
Driver key	4

- f. On the same tab, look at the **Value** field.
 - If it contains USB-CAN Interface, this means the selected device is the USB-CAN dongle you are using. Use the port number of the device (in this case, COM 3) as the CAN Interface ID.
 - If the field contains some other value, go back to **step a** and repeat the same sequence for another device on the Device Manager list.

USB serial device (COM3)	\times
General Port Settings Driver Details Events	
USB serial device (COM3)	
Property	
Bus reported device description	1
Value	
USB-CAN Interface	
OK Cano	el

- 19. Find out the CAN ID of the connected servo. To do this, run the **discovery** tutorial from the build folder. In the output of the tutorial, you will find the CAN ID you need.
- 20. Select a tutorial from the tutorial folder. Before running it, make sure to replace the tutorial variables with the CAN Interface ID from **Step 18** and the CAN ID from **Step 19**.

Now that you know how to start the servo and use the tutorials, you can create your own code. For more information, refer to the <u>API documentation</u>.