
robocadSim

Release 1.3.4

crackanddie

Feb 11, 2021

GETTING STARTED

1	Getting started	3
2	Robot documentation	5
3	Field documentation	33
4	Funcad documentation	41
5	Libraries documentation	49
6	robocadSim documentation	65
7	Be our sponsor	67
8	Need Help	69
9	License	71

Welcome to the robocadSim v1.3.4 - v1.3.6.1 documentation page. Here you will find lots of information about how to program robocadSim' virtual robots.

GETTING STARTED

Where to begin?

First you need to download robocadSim and run it. [Latest version](#).

I hope you have already chosen the language with which you will program your robot and the IDE for it.

You can read how to start using libraries for the selected language [here](#).

Write a program, turn on the robot and run the code!

ROBOT DOCUMENTATION

Here You can choose the robot you are programming. Read the documentation about all functions in the selected language and see some helpful examples. Good luck!

2.1 RE21

Some info here

2.2 RE21mini

Everything about RE21mini

2.2.1 Functions for RE21mini

ReadButtons

ReadButtons function is used to get buttons values from a robot.

Python

C++

C#

Location and name: RobocadSim.RE21mini.read_buttons()

Inputs:

Output:

bool map that includes:

- EMS button value
- Start button value
- Limit button value

Example:

```
1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 ems_button, start_button, limit_button = robot.read_buttons()
```

Additional info:

Location and name: "RE21mini.h".RE21mini.ReadButtons()

Inputs:

Output:

*bool** that includes:

- EMS button value
- Start button value
- Limit button value

Example:

```
1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
5 {
6     RE21mini robot;
7     bool* all_buttons = robot.ReadButtons();
8     bool ems_button = all_buttons[0];
9     bool start_button = all_buttons[1];
10    bool limit_button = all_buttons[2];
11 }
```

Additional info:

Location and name: RobocadSim.RE21mini.ReadButtons()

Inputs:

Output:

bool[] that includes:

- EMS button value
- Start button value
- Limit button value

Example:

```

1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             RE21mini robot = new RE21mini();
11             bool[] allButtons = robot.ReadButtons();
12             bool emsButton = allButtons[0];
13             bool startButton = allButtons[1];
14             bool limitButton = allButtons[2];
15         }
16     }
17 }

```

Additional info:

ReadButtonsVoid

ReadButtonsVoid function is used to write buttons values from a robot into variables.

Python

C++

C#

Location and name: RobocadSim.RE21mini.read_buttons_void()

Inputs:

Output:

Example:

```

1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 robot.read_buttons_void()
6 ems_button = robot.button_ems
7 start_button = robot.button_start
8 limit_button = robot.button_limit

```

Additional info:

Location and name: "RE21mini.h".RE21mini.ReadButtonsVoid()

Inputs:

Output:

Example:

```
1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
5 {
6     RE21mini robot;
7     robot.ReadButtonsVoid();
8     bool ems_button = robot.buttonEMS;
9     bool start_button = robot.buttonStart;
10    bool limit_button = robot.buttonLimit;
11 }
```

Additional info:

Location and name: RobocadSim.RE21mini.ReadButtonsVoid()

Inputs:

Output:

Example:

```
1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             RE21mini robot = new RE21mini();
11             robot.ReadButtonsVoid();
12             bool emsButton = robot.buttonEMS;
13             bool startButton = robot.buttonStart;
14             bool limitButton = robot.buttonLimit;
15         }
16     }
17 }
```

Additional info:

ReadCamera

ReadCamera function is used to get camera image from a robot.

Python

C++

C#

Location and name: RobocadSim.RE21mini.read_camera()

Inputs:

Output:

numpy.ndarray of image from robot

Example:

```
1 from robocadSimPy import RobocadSim
2 import cv2
3
4
5 robot = RobocadSim.RE21mini()
6 image = robot.read_camera()
```

Additional info:

Location and name: "RE21mini.h".RE21mini.ReadCamera()

Inputs:

Output:

Mat of image from robot

Example:

```
1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
5 {
6     RE21mini robot;
7     Mat image = robot.ReadCamera();
8 }
```

Additional info:

Location and name: RobocadSim.RE21mini.ReadCamera()

Inputs:

Output:

Mat of image from robot

Example:

```
1 using System;
2 using RobocadSim;
3 using Emgu.CV;
4
5 namespace TestLib
6 {
7     class Program
8     {
9         static void Main(string[] args)
10         {
11             RE21mini robot = new RE21mini();
12             Mat image = robot.ReadCamera();
13         }
14     }
15 }
```

Additional info:

ReadCameraBytes

ReadCameraBytes function is used to get camera bytes from a robot.

Python

C++

C#

Location and name: RobocadSim.RE21mini.read_camera_bytes()

Inputs:

Output:

bytes of image from robot

Example:

```
1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 image_bytes = robot.read_camera_bytes()
```

Additional info:

Location and name: “RE21mini.h”.RE21mini.ReadCameraBytes()

Inputs:

Output:

*char** of image from robot

Example:

```

1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
5 {
6     RE21mini robot;
7     char* imageBytes = robot.ReadCameraBytes();
8 }

```

Additional info:

Location and name: RobocadSim.RE21mini.ReadCameraBytes()

Inputs:

Output:

byte[] of image from robot

Example:

```

1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             RE21mini robot = new RE21mini();
11             byte[] imageBytes = robot.ReadCameraBytes();
12         }
13     }
14 }

```

Additional info:

ReadCameraBytesVoid

ReadCameraBytesVoid function is used to write camera bytes from a robot into variables.

Python

C++

C#

Location and name: RobocadSim.RE21mini.read_camera_bytes_void()

Inputs:

Output:

Example:

```
1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 robot.read_camera_bytes_void()
6 image_bytes = robot.bytes_from_camera
```

Additional info:

Location and name: "RE21mini.h".RE21mini.ReadCameraBytesVoid()

Inputs:

Output:

Example:

```
1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
5 {
6     RE21mini robot;
7     robot.ReadCameraBytesVoid();
8     char* imageBytes = robot.bytesFromCamera;
9 }
```

Additional info:

Location and name: RobocadSim.RE21mini.ReadCameraBytesVoid()

Inputs:

Output:

Example:

```
1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
```

(continues on next page)

(continued from previous page)

```

10         RE21mini robot = new RE21mini();
11         robot.ReadCameraBytesVoid();
12         byte[] imageBytes = robot.bytesFromCamera;
13     }
14 }
15 }

```

Additional info:

ReadCameraVoid

ReadCameraVoid function is used to write camera image from a robot into variables.

Python

C++

C#

Location and name: RobocadSim.RE21mini.read_camera_void()

Inputs:

Output:

Example:

```

1 from robocadSimPy import RobocadSim
2 import cv2
3
4
5 robot = RobocadSim.RE21mini()
6 robot.read_camera_void()
7 image = robot.image_from_camera

```

Additional info:

Location and name: "RE21mini.h".RE21mini.ReadCameraVoid()

Inputs:

Output:

Example:

```

1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
5 {

```

(continues on next page)

(continued from previous page)

```
6   RE21mini robot;  
7   robot.ReadCameraVoid();  
8   Mat image = robot.imageFromCamera;  
9 }
```

Additional info:

Location and name: RobocadSim.RE21mini.ReadCameraVoid()**Inputs:**

Output:

Example:

```
1  using System;  
2  using RobocadSim;  
3  using Emgu.CV;  
4  
5  namespace TestLib  
6  {  
7      class Program  
8      {  
9          static void Main(string[] args)  
10         {  
11             RE21mini robot = new RE21mini();  
12             robot.ReadCameraVoid();  
13             Mat image = robot.imageFromCamera;  
14         }  
15     }  
16 }
```

Additional info:

ReadEncs

ReadEncs function is used to get encoder values from a robot.

Python

C++

C#

Location and name: RobocadSim.RE21mini.read_encs()**Inputs:**

Output:

float map that includes:

- Right motor' encoder value
- Left motor' encoder value
- Back motor' encoder value
- Lift motor' encoder value

Example:

```

1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 right_enc, left_enc, back_enc, lift_enc = robot.read_encs()

```

Additional info:

- You should use Transfunction with encoders for a more convenient representation of values

Location and name: "RE21mini.h".RE21mini.ReadEncs()

Inputs:

Output:

*float** that includes:

- Right motor' encoder value
- Left motor' encoder value
- Back motor' encoder value
- Lift motor' encoder value

Example:

```

1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
5 {
6     RE21mini robot;
7     float* all_encs = robot.ReadEncs();
8     float right_enc = all_encs[0];
9     float left_enc = all_encs[1];
10    float back_enc = all_encs[2];
11    float lift_enc = all_encs[3];
12 }

```

Additional info:

- You should use Transfunction with encoders for a more convenient representation of values

Location and name: RobocadSim.RE21mini.ReadEncs()

Inputs:

Output:

float[] that includes:

- Right motor' encoder value
- Left motor' encoder value
- Back motor' encoder value
- Lift motor' encoder value

Example:

```
1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             RE21mini robot = new RE21mini();
11             float[] allEncs = robot.ReadEncs();
12             float rightEnc = allEncs[0];
13             float leftEnc = allEncs[1];
14             float backEnc = allEncs[2];
15             float liftEnc = allEncs[3];
16         }
17     }
18 }
```

Additional info:

- You should use Transfunction with encoders for a more convenient representation of values

ReadEncsVoid

ReadEncsVoid function is used to write encoder values from a robot into variables.

Python

C++

C#

Location and name: RobocadSim.RE21mini.read_encs_void()

Inputs:

Output:

Example:

```
1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 robot.read_encs_void()
6 right_enc = robot.right_motor_enc
7 left_enc = robot.left_motor_enc
```

(continues on next page)

(continued from previous page)

```

8 back_enc = robot.back_motor_enc
9 lift_enc = robot.lift_motor_enc

```

Additional info:

- You should use Transfunction with encoders for a more convenient representation of values

Location and name: "RE21mini.h".RE21mini.ReadEncsVoid()

Inputs:

Output:

Example:

```

1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
5 {
6     RE21mini robot;
7     robot.ReadEncsVoid();
8     float right_enc = robot.rightMotorEnc;
9     float left_enc = robot.leftMotorEnc;
10    float back_enc = robot.backMotorEnc;
11    float lift_enc = robot.liftMotorEnc;
12 }

```

Additional info:

- You should use Transfunction with encoders for a more convenient representation of values

Location and name: RobocadSim.RE21mini.ReadEncsVoid()

Inputs:

Output:

Example:

```

1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             RE21mini robot = new RE21mini();
11             robot.ReadEncsVoid();
12             float rightEnc = robot.encRight;
13             float leftEnc = robot.encLeft;
14             float backEnc = robot.encBack;

```

(continues on next page)

(continued from previous page)

```
15         float liftEnc = robot.encLift;
16     }
17 }
18 }
```

Additional info:

- You should use Transfunction with encoders for a more convenient representation of values

ReadSensors

ReadSensors function is used to get sensors values from a robot.

Python

C++

C#

Location and name: RobocadSim.RE21mini.read_sensors()

Inputs:

Output:

float map that includes:

- Right US value
- Left US value
- Right IR value
- Left IR value
- Gyroscope value

Example:

```
1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 right_us, left_us, right_ir, left_ir, gyro = robot.read_sensors()
```

Additional info:

Location and name: "RE21mini.h".RE21mini.ReadSensors()

Inputs:

Output:

*float** that includes:

- Right US value
- Left US value

- Right IR value
- Left IR value
- Gyroscope value

Example:

```

1  #include "RE21mini.h"
2  #include <iostream>
3
4  int main()
5  {
6      RE21mini robot;
7      float* all_sens = robot.ReadSensors();
8      float right_us = all_sens[0];
9      float left_us = all_sens[1];
10     float right_ir = all_sens[2];
11     float left_ir = all_sens[3];
12     float gyro = all_sens[4];
13 }

```

Additional info:

Location and name: RobocadSim.RE21mini.ReadSensors()**Inputs:**

Output:*float[]* that includes:

- Right US value
- Left US value
- Right IR value
- Left IR value
- Gyroscope value

Example:

```

1  using System;
2  using RobocadSim;
3
4  namespace TestLib
5  {
6      class Program
7      {
8          static void Main(string[] args)
9          {
10             RE21mini robot = new RE21mini();
11             float[] allSens = robot.ReadSensors();
12             float rightUS = allSens[0];
13             float leftUS = allSens[1];
14             float rightIR = allSens[2];
15             float leftIR = allSens[3];

```

(continues on next page)

(continued from previous page)

```
16         float gyro = allSens[4];
17     }
18 }
19 }
```

Additional info:

ReadSensorsVoid

ReadSensorsVoid function is used to write sensors values from a robot into variables.

Python

C++

C#

Location and name: RobocadSim.RE21mini.read_sensors_void()

Inputs:

Output:

Example:

```
1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 robot.read_sensors_void()
6 right_us = robot.right_us
7 left_us = robot.left_us
8 right_ir = robot.right_ir
9 left_ir = robot.left_ir
10 gyro = robot.navX
```

Additional info:

Location and name: "RE21mini.h".RE21mini.ReadSensorsVoid()

Inputs:

Output:

Example:

```
1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
```

(continues on next page)

(continued from previous page)

```

5 {
6     RE21mini robot;
7     robot.ReadSensorsVoid();
8     float right_us = robot.rightUS;
9     float left_us = robot.leftUS;
10    float right_ir = robot.rightIR;
11    float left_ir = robot.leftIR;
12    float gyro = robot.navX;
13 }

```

Additional info:

Location and name: RobocadSim.RE21mini.ReadSensorsVoid()**Inputs:**

Output:

Example:

```

1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             RE21mini robot = new RE21mini();
11             robot.ReadSensorsVoid();
12             float rightUS = robot.rightUS;
13             float leftUS = robot.leftUS;
14             float rightIR = robot.rightIR;
15             float leftIR = robot.leftIR;
16             float gyro = robot.navX;
17         }
18     }
19 }

```

Additional info:

WriteMotors

WriteMotors function is used to set speed values to a robot.

Python

C++

C#

Location and name: RobocadSim.RE21mini.write_motors()

Inputs:

- *float* speed to right motor
- *float* speed to left motor
- *float* speed to back motor

Output:

Example:

```
1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 robot.write_motors(10, -10, 0)
```

Additional info:

- Range of speed is from -50 to 50

Location and name: "RE21mini.h".RE21mini.WriteMotors()

Inputs:

- *float* speed to right motor
- *float* speed to left motor
- *float* speed to back motor

Output:

Example:

```
1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
5 {
6     RE21mini robot;
7     robot.WriteMotors(10, -10, 0);
8 }
```

Additional info:

- Range of speed is from -50 to 50

Location and name: RobocadSim.RE21mini.WriteMotors()

Inputs:

- *float* speed to right motor
- *float* speed to left motor
- *float* speed to back motor

Output:

Example:

```

1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             RE21mini robot = new RE21mini();
11             robot.WriteMotors(10, -10, 0);
12         }
13     }
14 }
```

Additional info:

- Range of speed is from -50 to 50

WriteMotorsVoid

WriteMotorsVoid function is used to set speed values to a robot from variables.

Python

C++

C#

Location and name: RobocadSim.RE21mini.write_motors_void()

Inputs:

Output:

Example:

```

1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 robot.right_motor_speed = 10
6 robot.left_motor_speed = -10
```

(continues on next page)

(continued from previous page)

```
7 robot.back_motor_speed = 0
8 robot.write_motors_void()
```

Additional info:

- Range of speed is from -50 to 50

Location and name: “RE21mini.h”.RE21mini.WriteMotorsVoid()**Inputs:**

Output:

Example:

```
1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
5 {
6     RE21mini robot;
7     robot.rightMotorSpeed = 10;
8     robot.leftMotorSpeed = -10;
9     robot.backMotorSpeed = 0;
10    robot.WriteMotorsVoid();
11 }
```

Additional info:

- Range of speed is from -50 to 50

Location and name: RobocadSim.RE21mini.WriteMotorsVoid()**Inputs:**

Output:

Example:

```
1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             RE21mini robot = new RE21mini();
11             robot.speedRight = 10;
12             robot.speedLeft = -10;
13             robot.speedBack = 0;
14             robot.WriteMotorsVoid();
15         }
16     }
17 }
```

(continues on next page)

(continued from previous page)

```

16     }
17 }

```

Additional info:

- Range of speed is from -50 to 50

WriteOMS

WriteOMS function is used to set speed and angle values to a robot.

Python

C++

C#

Location and name: RobocadSim.RE21mini.write_oms()

Inputs:

- *float* speed to lift motor
- *float* angle to big servo motor
- *float* direction of small servo motor

Output:

Example:

```

1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 robot.write_oms(10, 1600, 1400)

```

Additional info:

- Range of speed is from -50 to 50
- Range of angles for big servo motor is from 1490 to 1750
- Range of values for small servo motor is from 1400 to 1600

Location and name: "RE21mini.h".RE21mini.WriteOMS()

Inputs:

- *float* speed to lift motor
- *float* angle to big servo motor
- *float* direction of small servo motor

Output:

Example:

```
1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
5 {
6     RE21mini robot;
7     robot.WriteOMS(10, 1600, 1400);
8 }
```

Additional info:

- Range of speed is from -50 to 50
- Range of angles for big servo motor is from 1490 to 1750
- Range of values for small servo motor is from 1400 to 1600

Location and name: RobocadSim.RE21mini.WriteOMS()

Inputs:

- *float* speed to lift motor
- *float* angle to big servo motor
- *float* direction of small servo motor

Output:

Example:

```
1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             RE21mini robot = new RE21mini();
11             robot.WriteOMS(10, 1600, 1400);
12         }
13     }
14 }
```

Additional info:

- Range of speed is from -50 to 50
- Range of angles for big servo motor is from 1490 to 1750
- Range of values for small servo motor is from 1400 to 1600

WriteOMSVoid

WriteOMSVoid function is used to set speed and angle values to a robot from variables.

Python

C++

C#

Location and name: RobocadSim.RE21mini.write_oms_void()

Inputs:

Output:

Example:

```
1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 robot.lift_motor_speed = 10
6 robot.angle_for_big = 1600
7 robot.dir_for_small = 1400
8 robot.write_oms_void()
```

Additional info:

- Range of speed is from -50 to 50
- Range of angles for big servo motor is from 1490 to 1750
- Range of values for small servo motor is from 1400 to 1600

Location and name: "RE21mini.h".RE21mini.WriteOMSVoid()

Inputs:

Output:

Example:

```
1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
5 {
6     RE21mini robot;
7     robot.liftMotorSpeed = 10;
8     robot.bigServoAngle = 1600;
9     robot.smallServoDir = 1400;
10    robot.WriteOMSVoid();
11 }
```

Additional info:

- Range of speed is from -50 to 50
- Range of angles for big servo motor is from 1490 to 1750
- Range of values for small servo motor is from 1400 to 1600

Location and name: RobocadSim.RE21mini.WriteOMSVoid()

Inputs:

Output:

Example:

```
1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             RE21mini robot = new RE21mini();
11             robot.speedLift = 10;
12             robot.angleBig = 1600;
13             robot.dirSmall = 1400;
14             robot.WriteOMSVoid();
15         }
16     }
17 }
```

Additional info:

- Range of speed is from -50 to 50
- Range of angles for big servo motor is from 1490 to 1750
- Range of values for small servo motor is from 1400 to 1600

WriteReset

WriteReset function is used to set reset values to a robot.

Python

C++

C#

Location and name: RobocadSim.RE21mini.write_reset()

Inputs:

- *bool* reset right motor encoder
- *bool* reset left motor encoder
- *bool* reset back motor encoder
- *bool* reset lift motor encoder

- *bool* reset gyroscope

Output:

Example:

```

1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 robot.write_reset(True, True, True, False, False)

```

Additional info:

- You should write Your own gyro reset (cause my doesn't work well :))

Location and name: "RE21mini.h".RE21mini.WriteReset()

Inputs:

- *bool* reset right motor encoder
- *bool* reset left motor encoder
- *bool* reset back motor encoder
- *bool* reset lift motor encoder
- *bool* reset gyroscope

Output:

Example:

```

1 #include "RE21mini.h"
2 #include <iostream>
3
4 int main()
5 {
6     RE21mini robot;
7     robot.WriteReset(true, true, true, false, false);
8 }

```

Additional info:

- You should write Your own gyro reset (cause my doesn't work well :))

Location and name: RobocadSim.RE21mini.WriteReset()

Inputs:

- *bool* reset right motor encoder
- *bool* reset left motor encoder
- *bool* reset back motor encoder
- *bool* reset lift motor encoder
- *bool* reset gyroscope

Output:

Example:

```
1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             RE21mini robot = new RE21mini();
11             robot.WriteReset(true, true, true, false, false);
12         }
13     }
14 }
```

Additional info:

- You should write Your own gyro reset (cause my doesn't work well :))

WriteResetVoid

WriteResetVoid function is used to set reset values to a robot from variables.

Python

C++

C#

Location and name: RobocadSim.RE21mini.write_reset_void()

Inputs:

Output:

Example:

```
1 from robocadSimPy import RobocadSim
2
3
4 robot = RobocadSim.RE21mini()
5 robot.reset_right_enc = True
6 robot.reset_left_enc = True
7 robot.reset_back_enc = True
8 robot.reset_lift_enc = False
9 robot.reset_gyro = False
10 robot.write_reset_void()
```

Additional info:

- You should write Your own gyro reset (cause my doesn't work well :))

Location and name: "RE21mini.h".RE21mini.WriteResetVoid()

Inputs:

Output:

Example:

```

1  #include "RE21mini.h"
2  #include <iostream>
3
4  int main()
5  {
6      RE21mini robot;
7      robot.resetRightEnc = true;
8      robot.resetLeftEnc = true;
9      robot.resetBackEnc = true;
10     robot.resetLiftEnc = false;
11     robot.resetGyro = false;
12     robot.WriteResetVoid();
13 }
```

Additional info:

- You should write Your own gyro reset (cause my doesn't work well :))

Location and name: RobocadSim.RE21mini.WriteResetVoid()

Inputs:

Output:

Example:

```

1  using System;
2  using RobocadSim;
3
4  namespace TestLib
5  {
6      class Program
7      {
8          static void Main(string[] args)
9          {
10             RE21mini robot = new RE21mini();
11             robot.resetEncRight = true;
12             robot.resetEncLeft = true;
13             robot.resetEncBack = true;
14             robot.resetEncLift = false;
15             robot.resetGyro = false;
16             robot.WriteResetVoid();
17         }
18     }
19 }
```

Additional info:

- You should write Your own gyro reset (cause my doesn't work well :))

FIELD DOCUMENTATION

Here You can choose a field and read some info about it.

3.1 Field E21

Here You can read some info about E21 field.

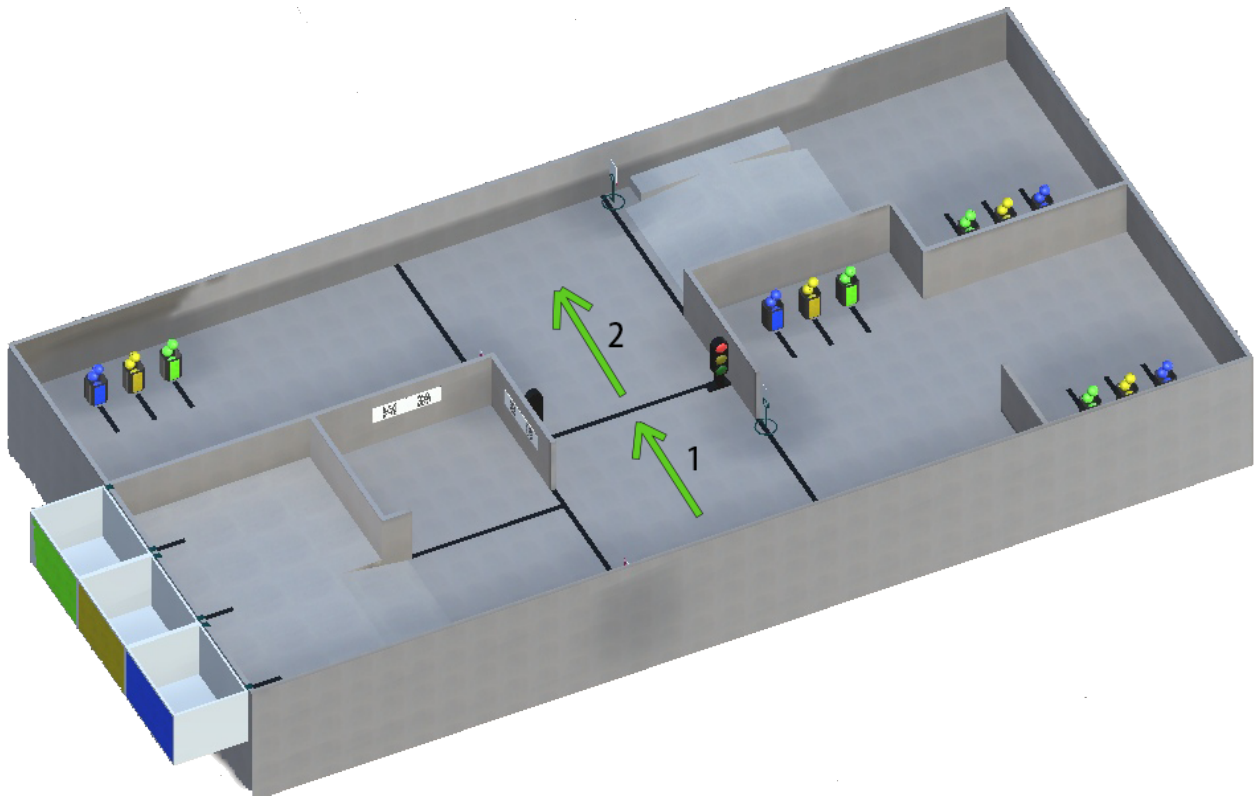
3.1.1 Traffic lights on E21 field

There are 2 traffic lights on the field and they can burn with such colors:

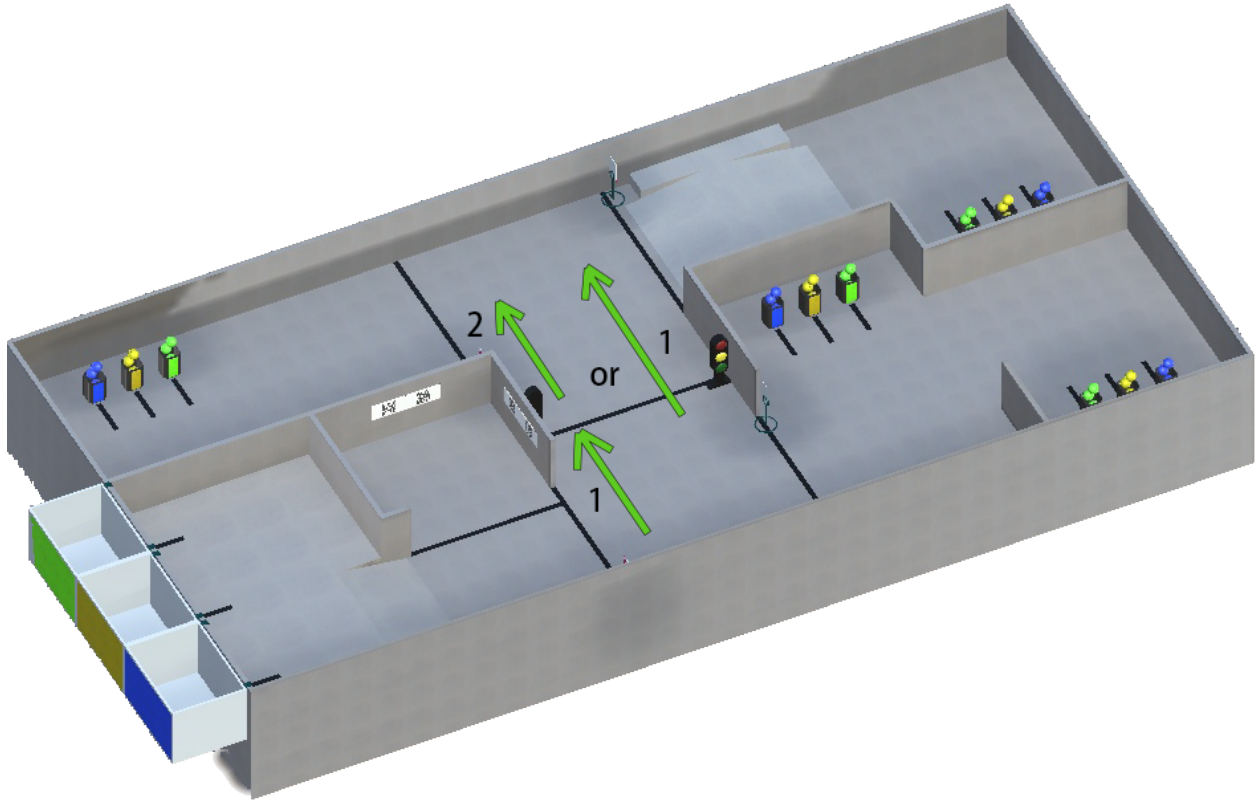
- **Red** color
- **Yellow** color
- **Green** color

Red:

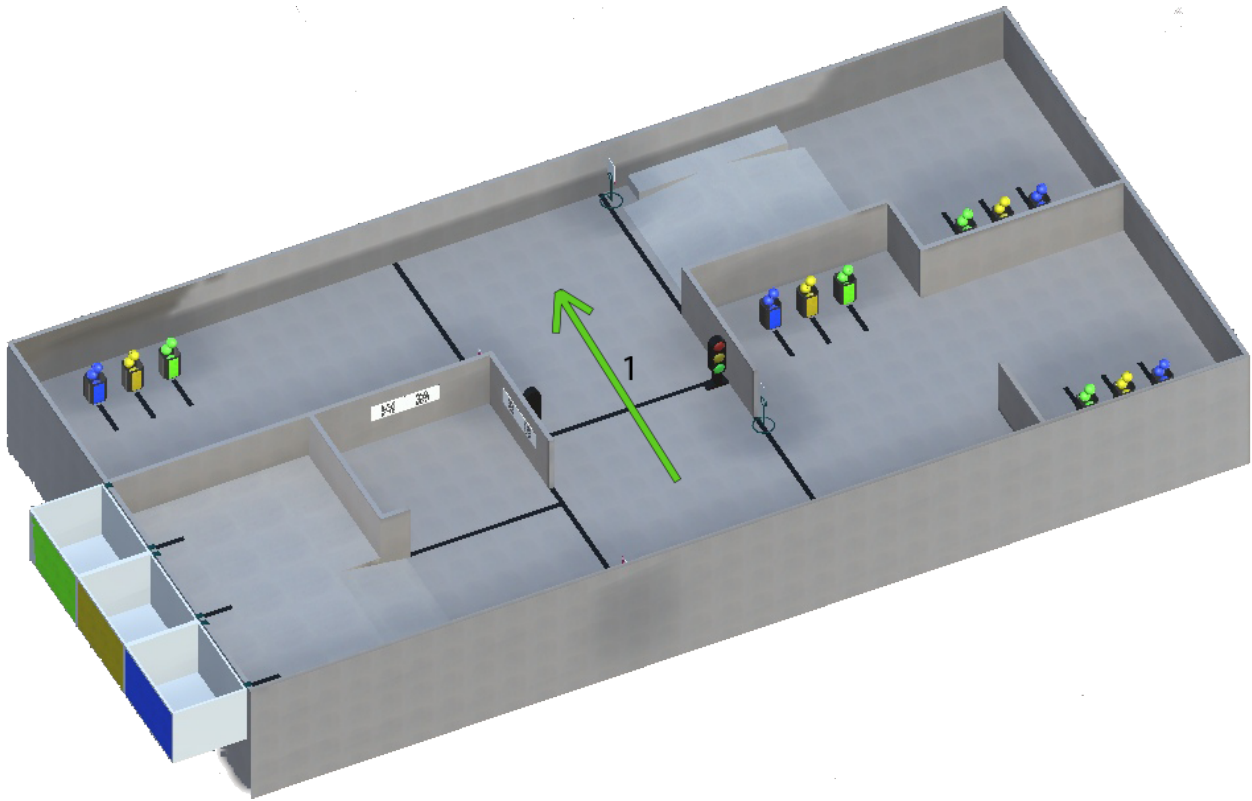
If a **red** light is on at a traffic light, robot should stop in front of it or in front of a black line. And continue movement when a **green** light is on at a traffic light:

**Yellow:**

If a **yellow** light is on at a traffic light, robot should stop in front of it or in front of a black line. And continue movement when a **green** light is on at a traffic light. **But** if robot already crossed the line when the yellow light came on, it can complete movement:

**Green:**

If a **green** light is on at a traffic light, robot can continue its movement:



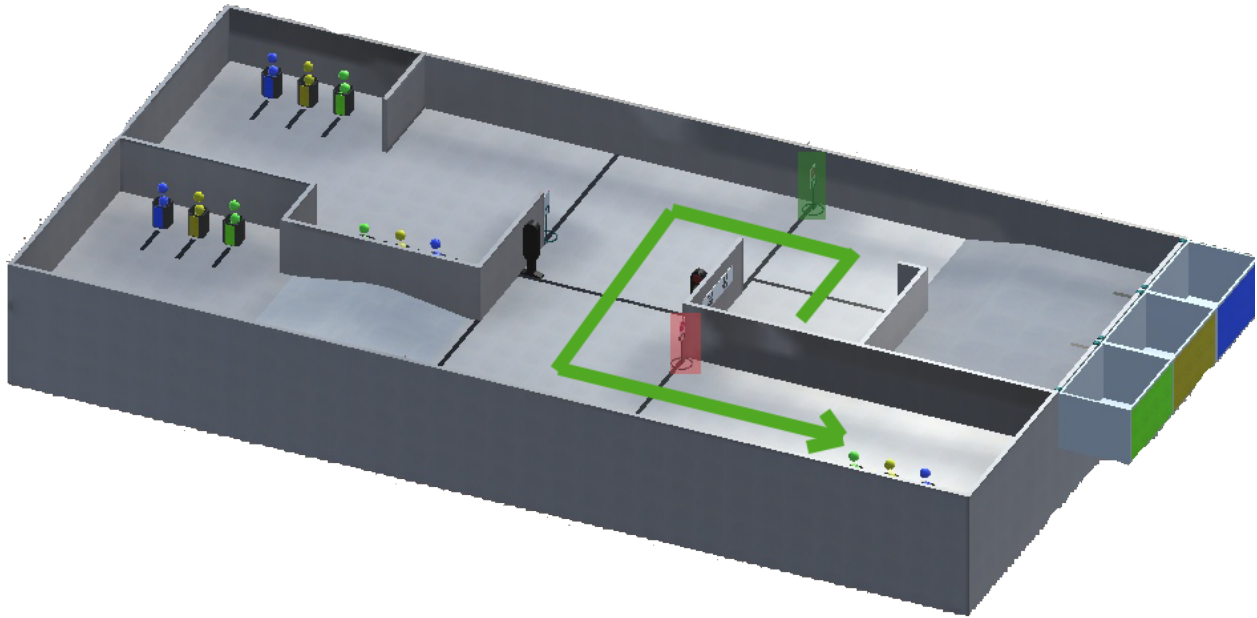
3.1.2 Signs on E21 field

There are 4 signs on the field and they may be:

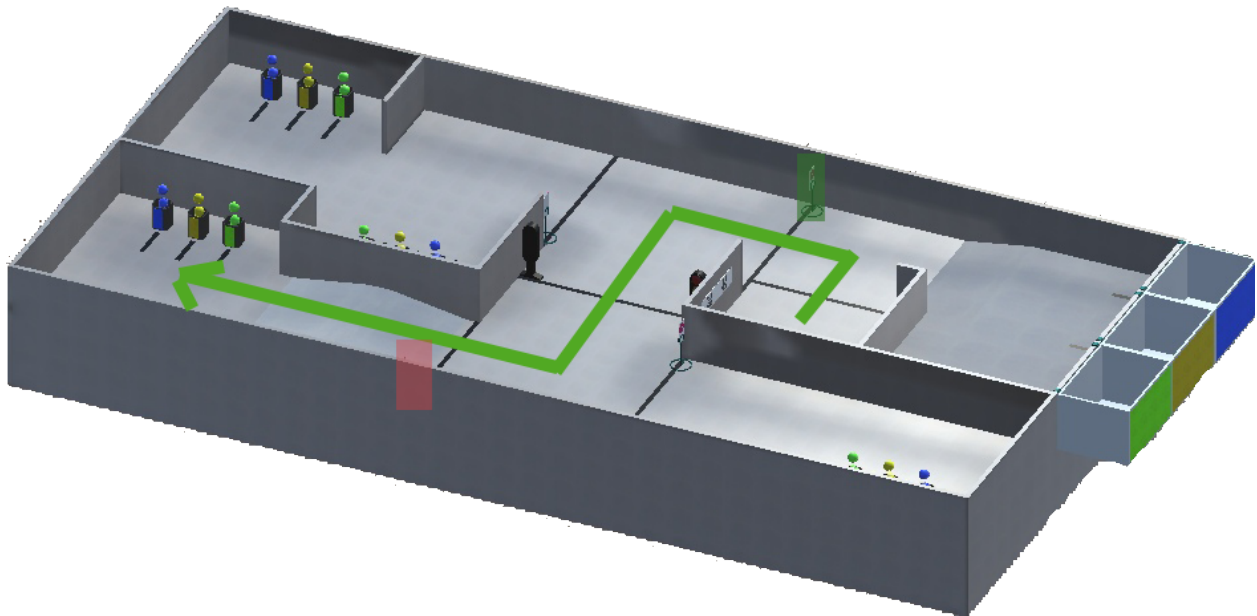
- **Turn Left** sign
- **Turn Right** sign
- **Move Forward** sign
- **Stop** sign

Turn Left:

For example, if You have **Turn Left** sign at the start (green rect) You should move like that:



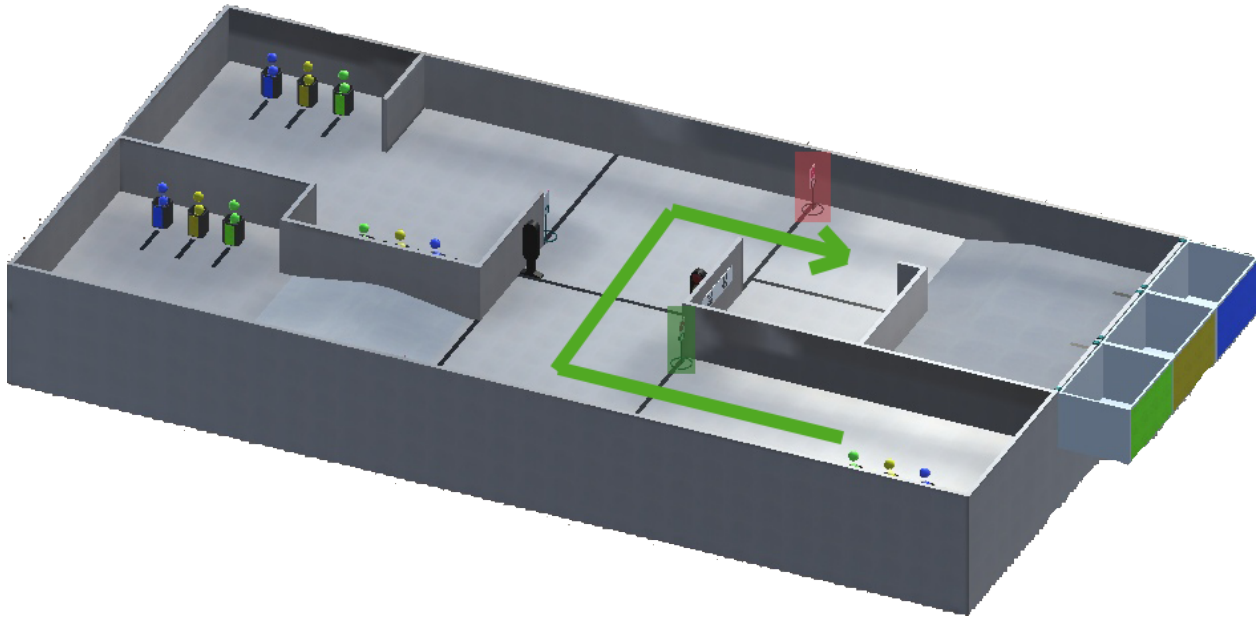
or like that:



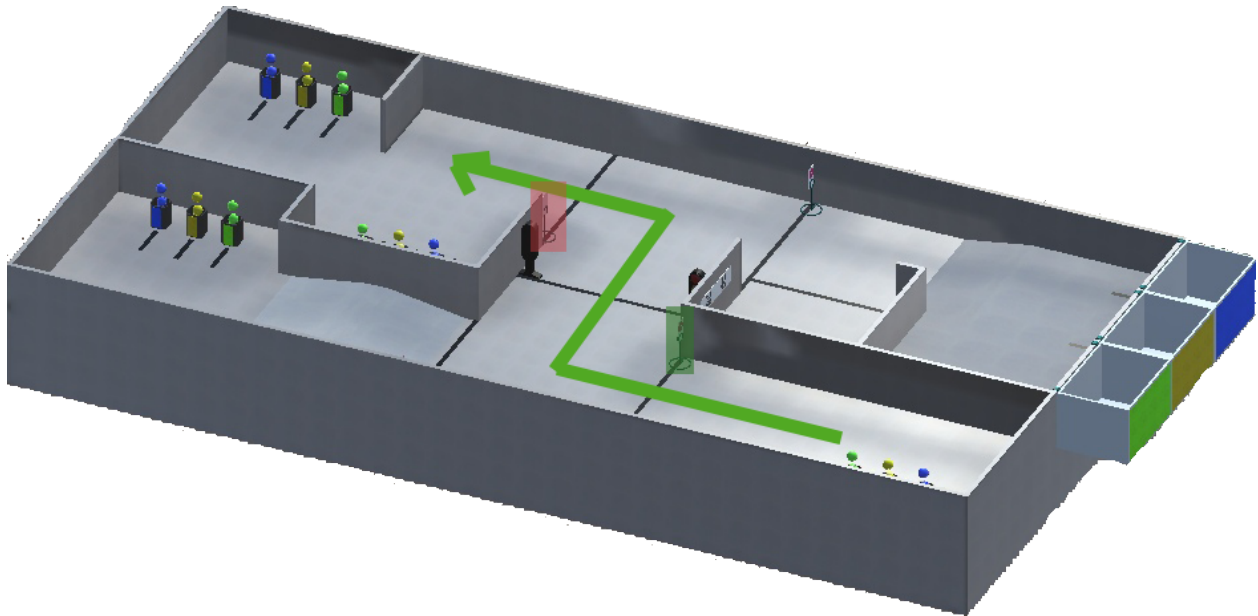
and check next sign (red rect) if You need it.

Turn Right:

For example, if there is **Turn Right** sign at the Distant village (green rect) You should move like that:



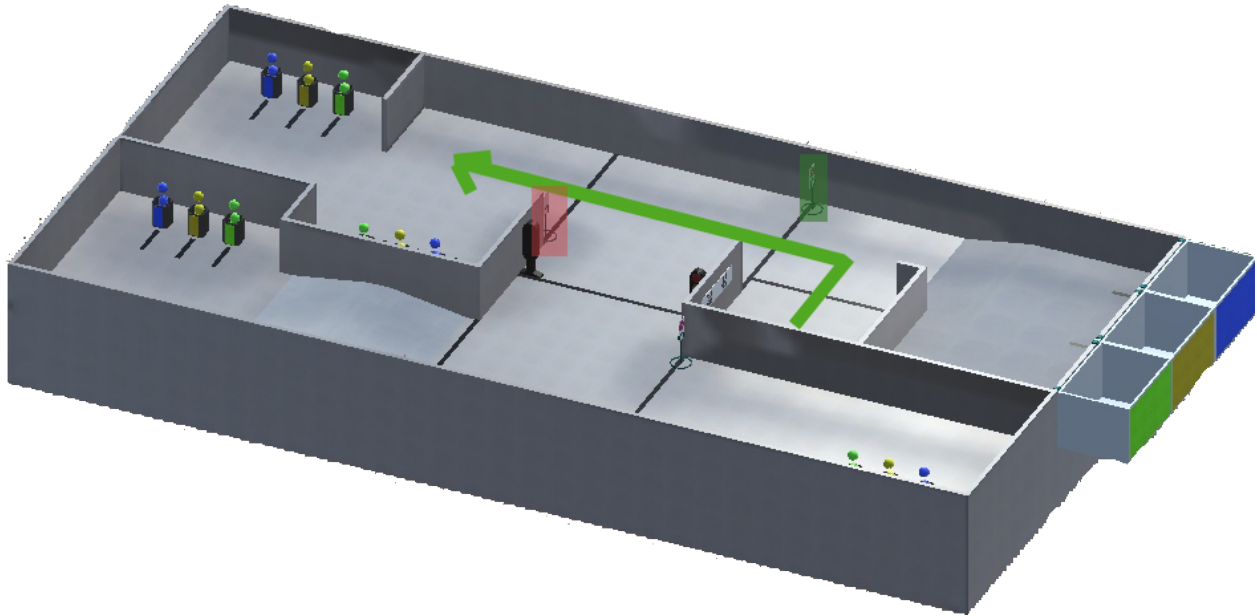
or like that:



and check next sign (red rect) if You need it.

Move Forward:

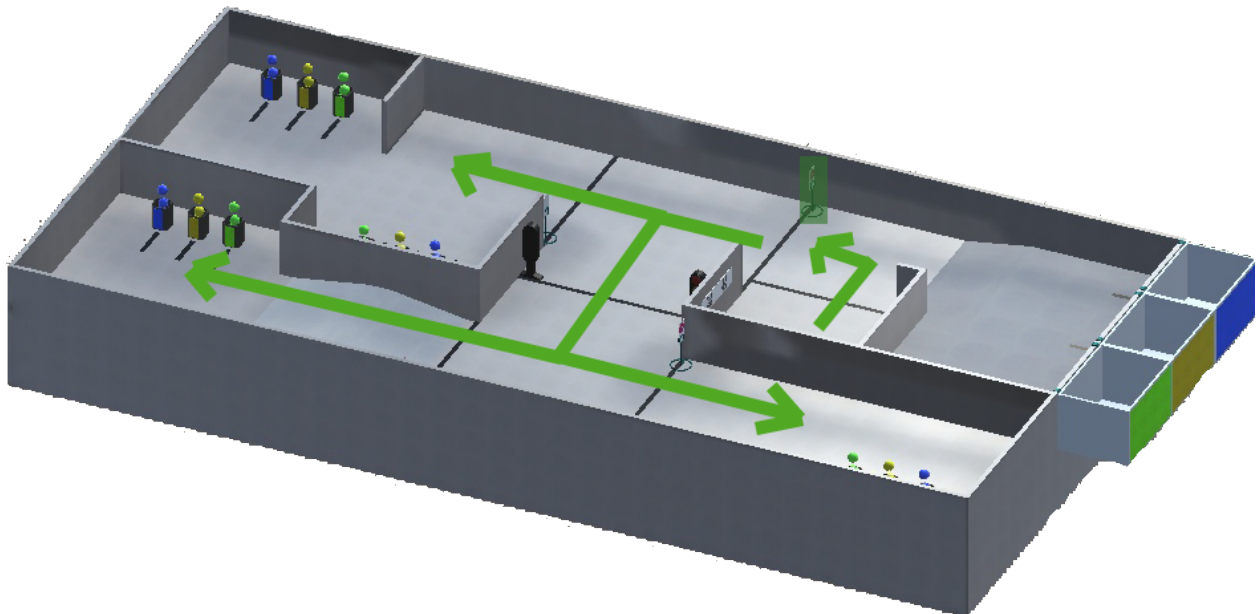
For example, if there is **Move Forward** sign at the start (green rect) You should move like that:



and check next sign (red rect) if You need it.

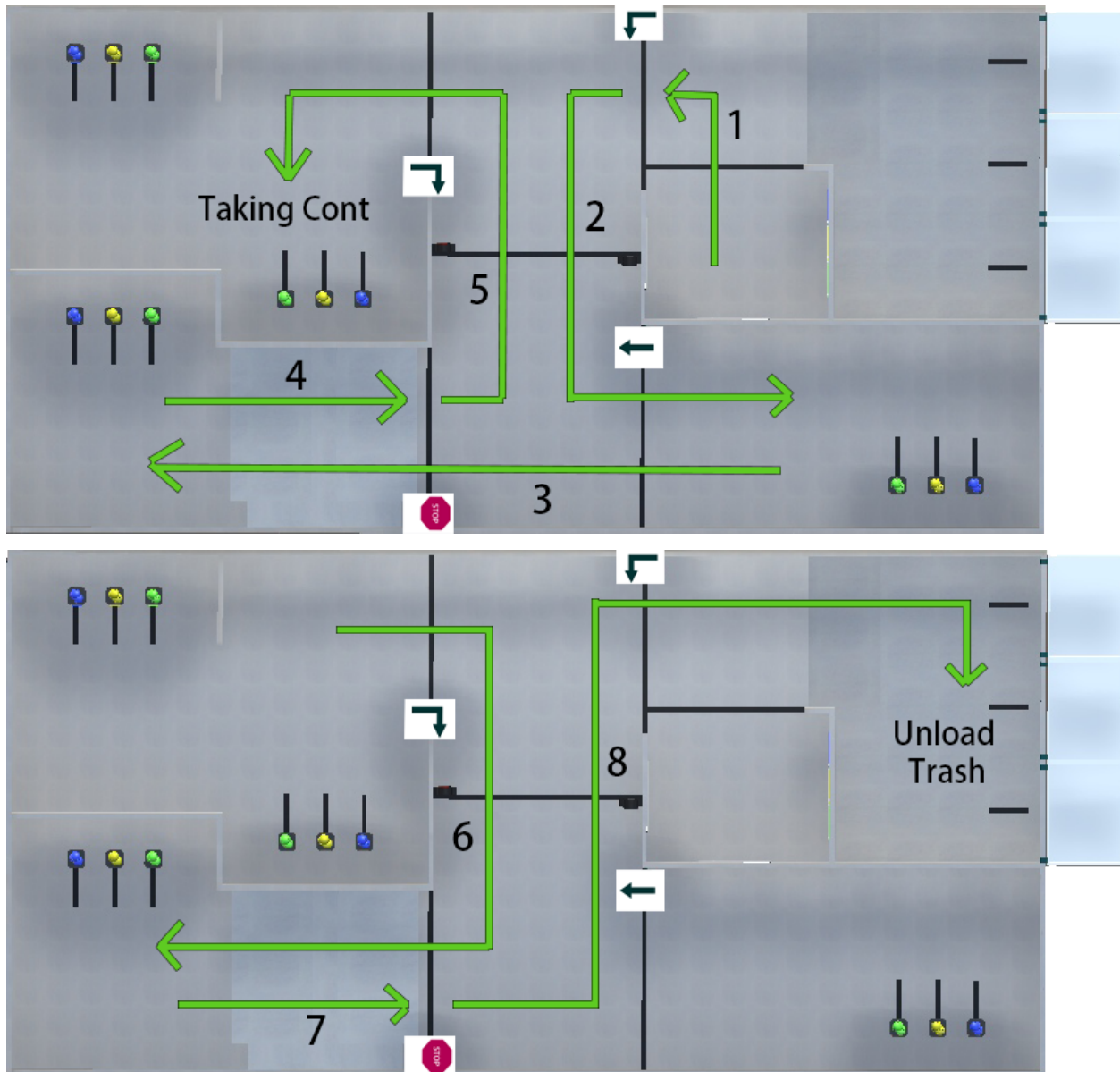
Stop:

For example, if there is **Stop** sign at the start (green rect) You should stop in front of the sign (or line) and wait for 2 seconds. After that You can continue Your movement wherever You want. Example:



All signs:

Here is an example of movement when robot has to go to the Pine village and take containers from there:



FUNCAD DOCUMENTATION

Here You can find some info about Funcad.

4.1 FromAxisToMotors

FromAxisToMotors function is used to remake input axis values into motors values for **tricycle** robot.

Python

C++

C#

Location and name: Funcad.Funcad.from_axis_to_motors()

Inputs:

- *float* speed to X axis
- *float* speed to Y axis
- *float* speed to Z axis

Output:

numpy.ndarray that includes:

- Speed to right motor
- Speed to left motor
- Speed to back motor

Example:

```
1 from robocadSimPy import Funcad
2
3
4 funcad = Funcad.Funcad()
5 out = funcad.from_axis_to_motors(5, -5, 3) # [ 2.273672 -9.273672  4.      ]
```

Additional info:

Location and name: "Funcad.h".Funcad.from_axis_to_motors()

Inputs:

- *float* speed to X axis

- *float* speed to Y axis
- *float* speed to Z axis

Output:

*float** that includes:

- Speed to right motor
- Speed to left motor
- Speed to back motor

Example:

```
1 #include "Funcad.h"
2 #include <iostream>
3
4 int main()
5 {
6     Funcad funcad;
7     float* out = funcad.from_axis_to_motors(5, -5, 3); // 2.2735 -9.2735 4
8 }
```

Additional info:

Location and name: RobocadSim.Funcad.FromAxisToMotors()

Inputs:

- *float* speed to X axis
- *float* speed to Y axis
- *float* speed to Z axis

Output:

System.Numerics.Vector3 that includes:

- Speed to right motor
- Speed to left motor
- Speed to back motor

Example:

```
1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             Funcad funcad = new Funcad();
11             Vector3 vec = funcad.FromAxisToMotors(5, -5, 3); // 2,273672 -9,273672 4
12         }
13     }
14 }
```

Additional info:

4.2 FromMotorsToAxis

FromMotorsToAxis function is used to remake input motors values into axis values for **tricycle** robot.

Python

C++

C#

Location and name: Funcad.Funcad.from_motors_to_axis()

Inputs:

- *float* speed to right motor
- *float* speed to left motor
- *float* speed to back motor

Output:

numpy.ndarray that includes:

- Speed to X axis
- Speed to Y axis
- Speed to Z axis

Example:

```

1 from robocadSimPy import Funcad
2
3
4 funcad = Funcad.Funcad()
5 out = funcad.from_motors_to_axis(5, -5, 0) # [8.66 0. 0. ]

```

Additional info:

Location and name: "Funcad.h".Funcad.from_motors_to_axis()

Inputs:

- *float* speed to right motor
- *float* speed to left motor
- *float* speed to back motor

Output:

*float** that includes:

- Speed to X axis
- Speed to Y axis
- Speed to Z axis

Example:

```
1 #include "Funcad.h"
2 #include <iostream>
3
4 int main()
5 {
6     Funcad funcad;
7     float* out = funcad.from_motors_to_axis(5, -5, 0); // 8.66 0 0
8 }
```

Additional info:

Location and name: RobocadSim.Funcad.FromMotorsToAxis()**Inputs:**

- *float* speed to right motor
- *float* speed to left motor
- *float* speed to back motor

Output:*System.Numerics.Vector3* that includes:

- Speed to X axis
- Speed to Y axis
- Speed to Z axis

Example:

```
1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             Funcad funcad = new Funcad();
11             Vector3 vec = funcad.FromMotorsToAxis(5, -5, 0); // 8.66 0 0
12         }
13     }
14 }
```

Additional info:

4.3 InRangeBool

InRangeBool function is used to check that input value is in range.

Python

C++

C#

Location and name: Funcad.Funcad.in_range_bool()

Inputs:

- *float* input value
- *float* lower threshold
- *float* upper threshold

Output:

bool is in range

Example:

```

1 from robocadSimPy import Funcad
2
3
4 funcad = Funcad.Funcad()
5 out = funcad.in_range_bool(5, 0, 12)  # True

```

Additional info:

Location and name: "Funcad.h".Funcad.in_range_bool()

Inputs:

- *float* input value
- *float* lower threshold
- *float* upper threshold

Output:

bool is in range

Example:

```

1 #include "Funcad.h"
2 #include <iostream>
3
4 int main()
5 {
6     Funcad funcad;
7     bool out = funcad.in_range_bool(5, 0, 12); // true
8 }

```

Additional info:

Location and name: RobocadSim.Funcad.InRangeBool()

Inputs:

- *float* input value
- *float* lower threshold
- *float* upper threshold

Output:

bool is in range

Example:

```
1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {
8         static void Main(string[] args)
9         {
10             Funcad funcad = new Funcad();
11             bool output = funcad.InRangeBool(5, 0, 12); // true
12         }
13     }
14 }
```

Additional info:

4.4 TransfunctionCoda

TransfunctionCoda function is used to rearrange input value. Created by subsystems developer Coda.

Python

C++

C#

Location and name: Funcad.Funcad.transfunc_coda()

Inputs:

- *float* value to remake
- *list* input array
- *list* output array

Output:

float remaded input

Example:

```
1 from robocadSimPy import Funcad
2
3
```

(continues on next page)

(continued from previous page)

```

4 funcad = Funcad.Funcad()
5 out = funcad.transfunc_coda(5, [2, 10], [20, 100]) # out will be 50

```

Additional info:

Location and name: “Funcad.h”.Funcad.transfunc_coda()**Inputs:**

- *float* value to remake
- *float** input array
- *float** output array
- *int* size of input or output array

Output:*float* remaded input**Example:**

```

1 #include "Funcad.h"
2 #include <iostream>
3
4 int main()
5 {
6     Funcad funcad;
7     float in_arr[] = { 2, 10 };
8     float out_arr[] = { 20, 100 };
9     float out = funcad.transfunc_coda(5, in_arr, out_arr, 2); // out will be 50
10 }

```

Additional info:

Location and name: RobocadSim.Funcad.TransfunctionCoda()**Inputs:**

- *float* value to remake
- *List<float>* input array
- *List<float>* output array

Output:*float* remaded input**Example:**

```

1 using System;
2 using RobocadSim;
3
4 namespace TestLib
5 {
6     class Program
7     {

```

(continues on next page)

(continued from previous page)

```
8      static void Main(string[] args)
9      {
10         Funcad funcad = new Funcad();
11         float[] inArr = { 2, 10 };
12         List<float> inList = new List<float>(inArr);
13         float[] outArr = { 20, 100 };
14         List<float> outList = new List<float>(outArr);
15         float outVal = funcad.TransfunctionCoda(5, inList, outList); // out will
↪be 50
16     }
17 }
18 }
```

Additional info:

LIBRARIES DOCUMENTATION

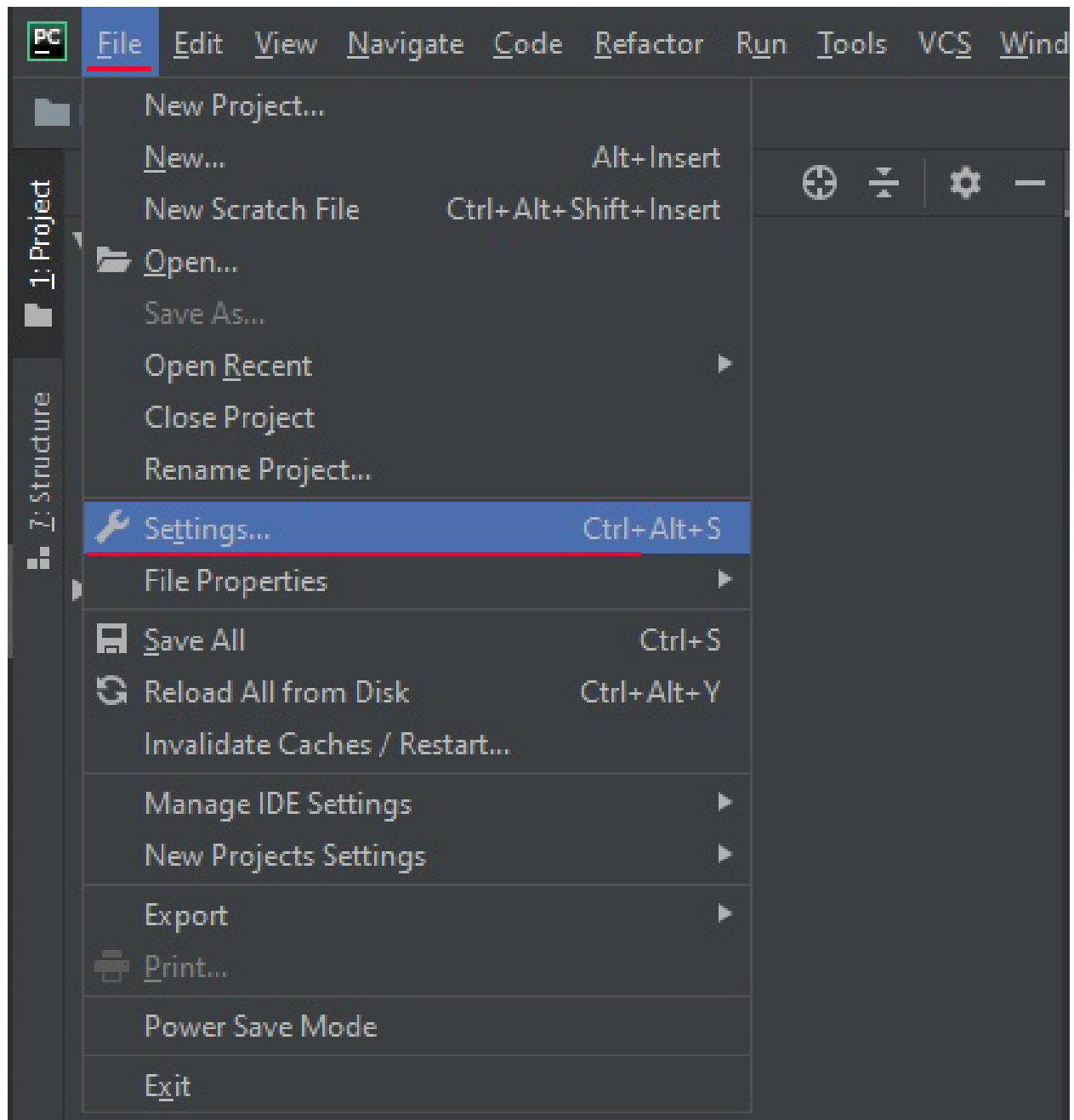
Here You can choose the favourite programming language and read about how to install the library into Your project.

5.1 Python library

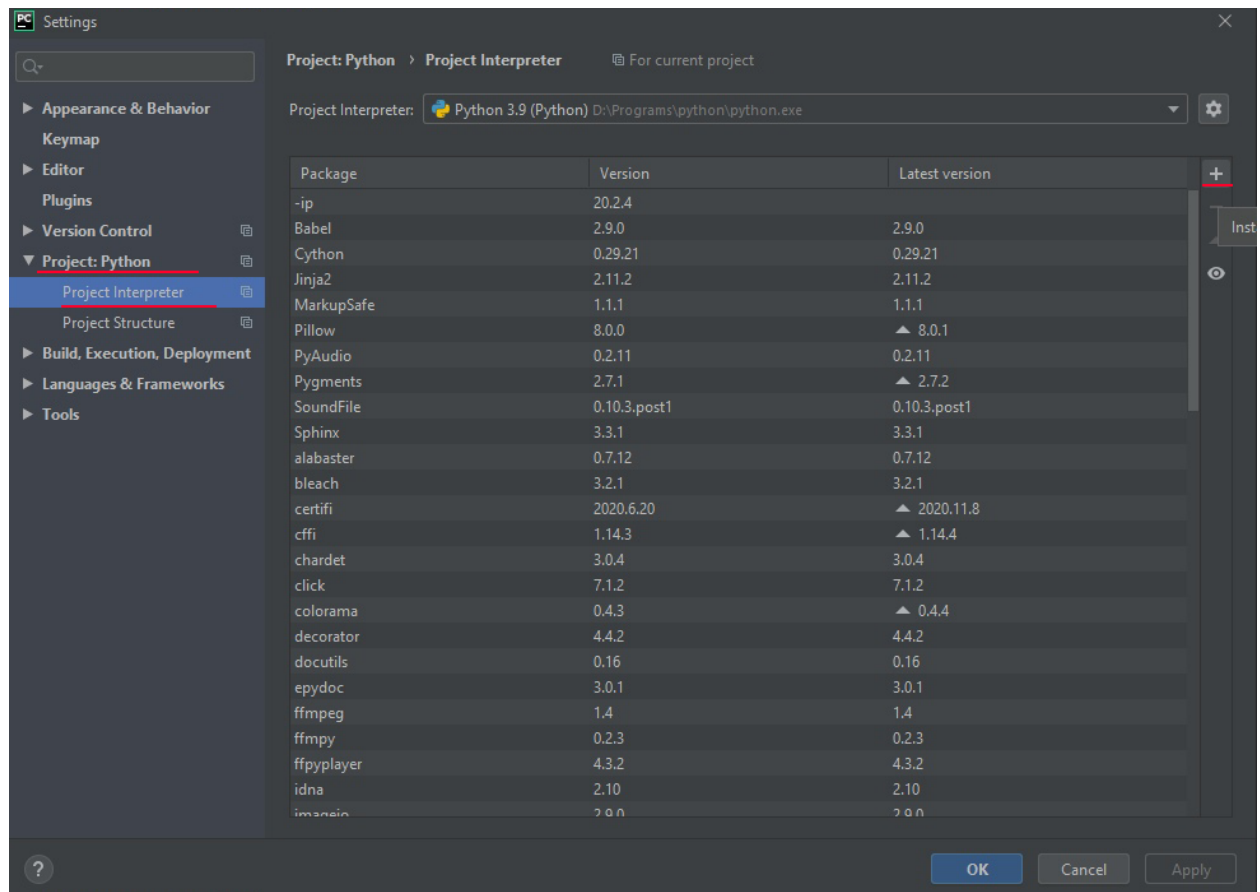
Here is some info about how to download robocadSim Python library. I am going to use PyCharm 2020.

5.1.1 First way:

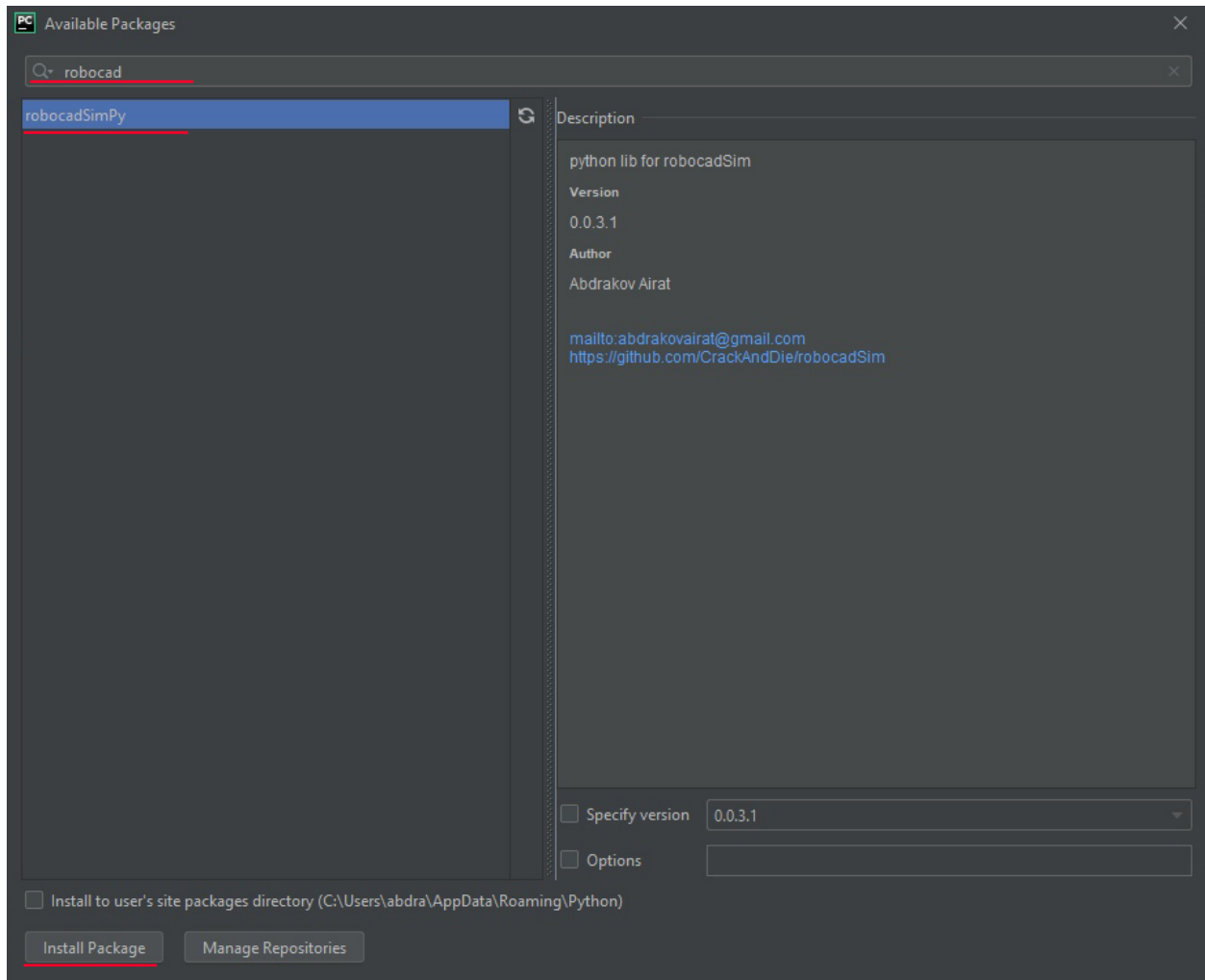
1. Open Your project in PyCharm -> click on **File** -> **Settings**



2. Click on **Project: Python** -> **Project Interpreter** -> **Install** (Plus button)



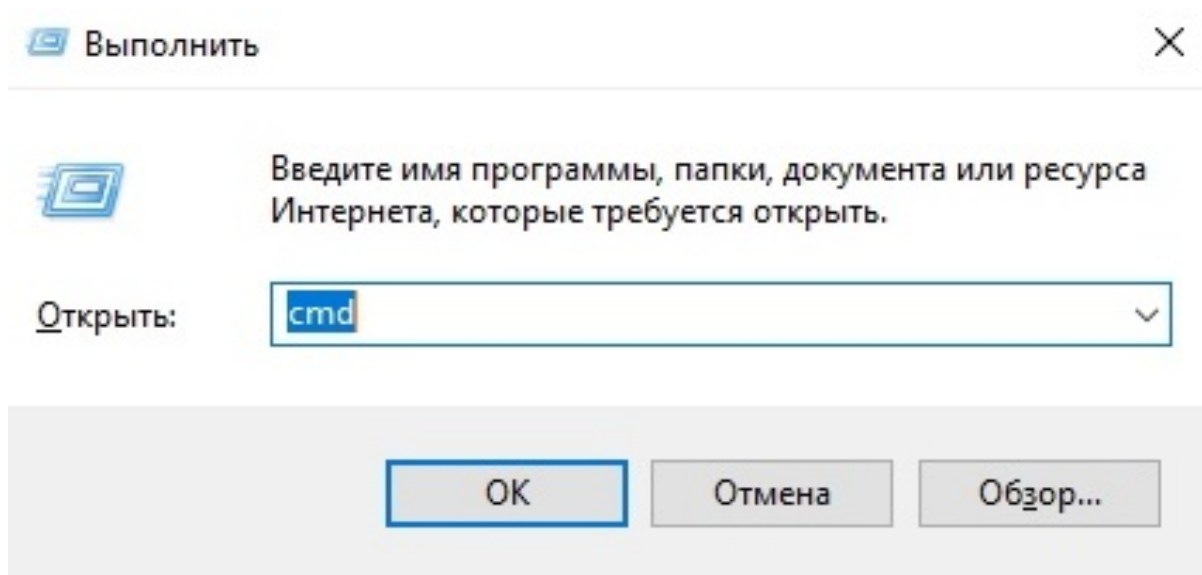
3. Write **robocadSimPy** in Search Line -> select **robocadSimPy** -> click **Install Package**



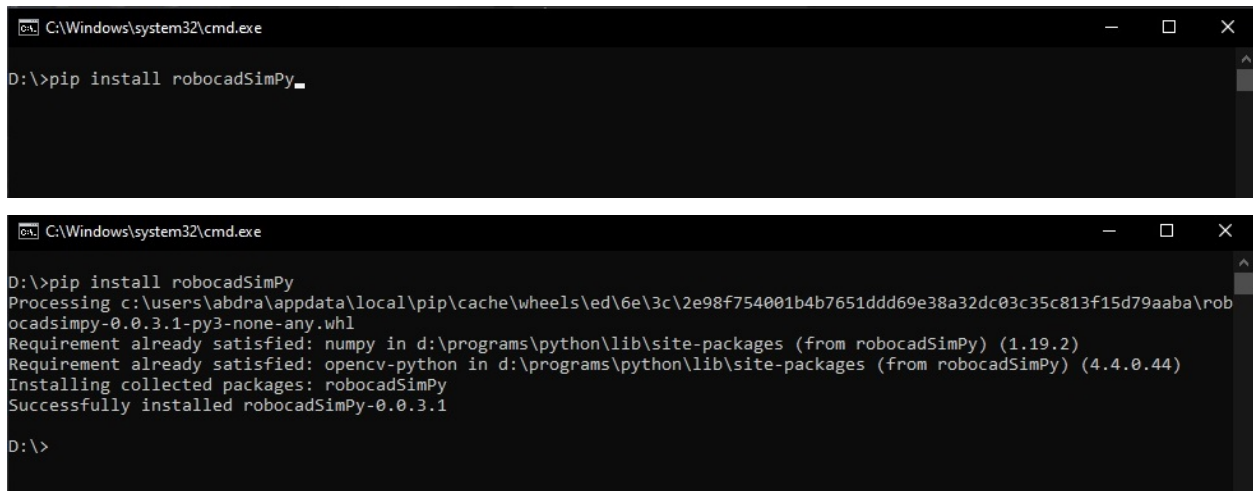
4. Now You can use robocadSim Python library in Your project!

5.1.2 Second way:

1. **Win + R** -> write **cmd** here -> press **Enter**



2. Write here **pip install robocadSimPy** or **pip3 install robocadSimPy** -> press **Enter**



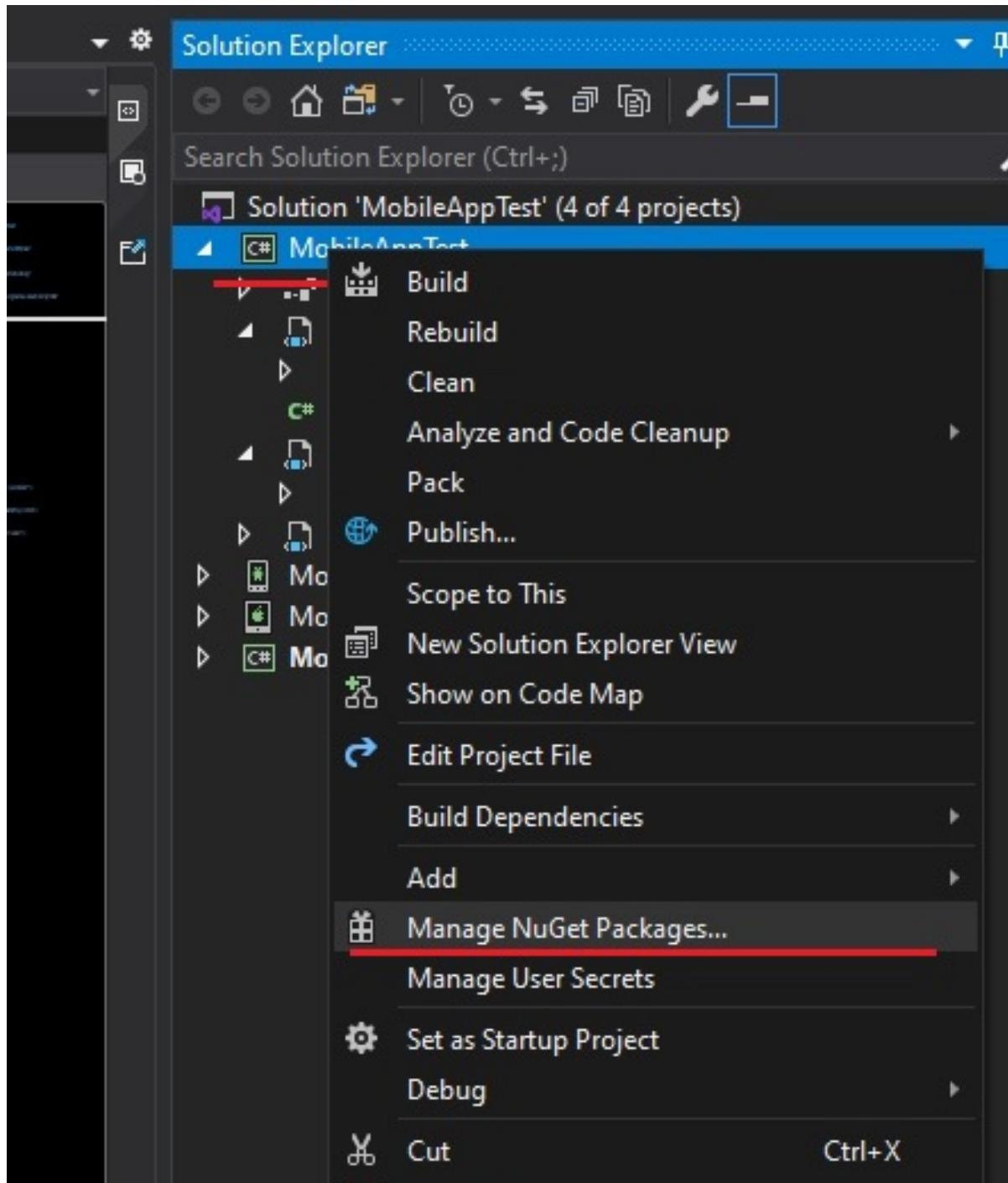
3. Now You can use robocadSim Python library in Your project!

5.2 C# library

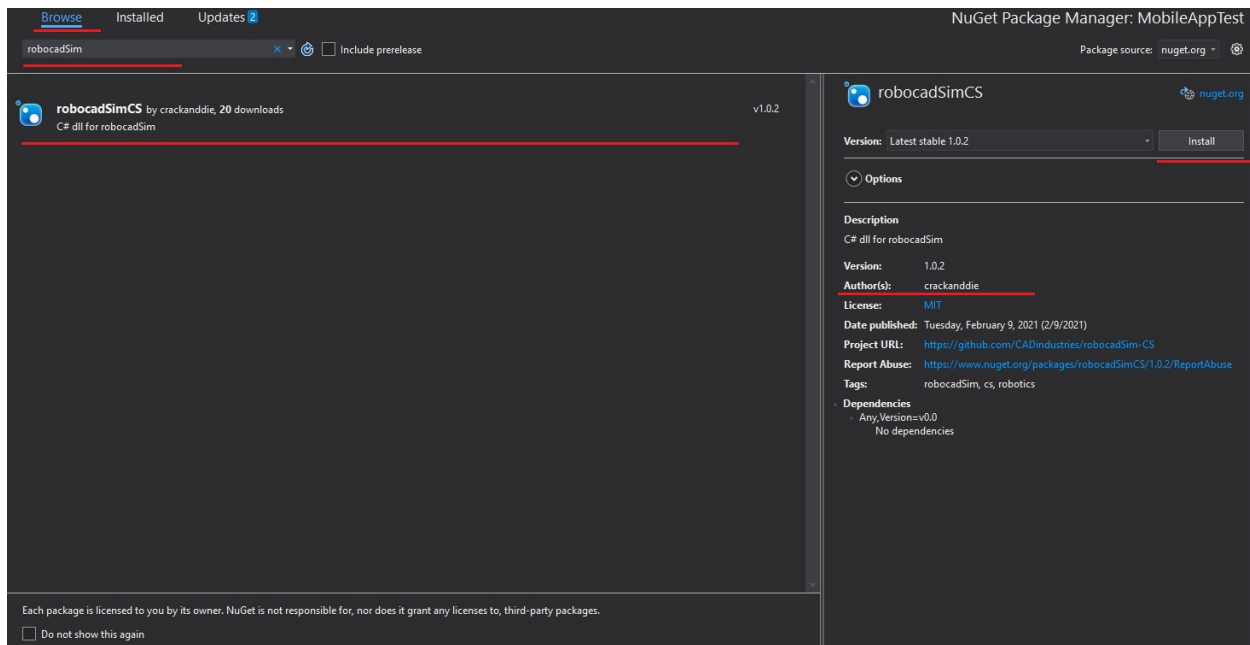
Here is some info about how to use robocadSim C# library in your project. I am going to use Visual Studio 2019. You need emgu-cv installed in your project if You use robocadSim version < v1.3.7. ([How to install example](#)).

5.2.1 First way:

1. Right click on Your project name in **Solution explorer** -> click on **Manage NuGet Packages**

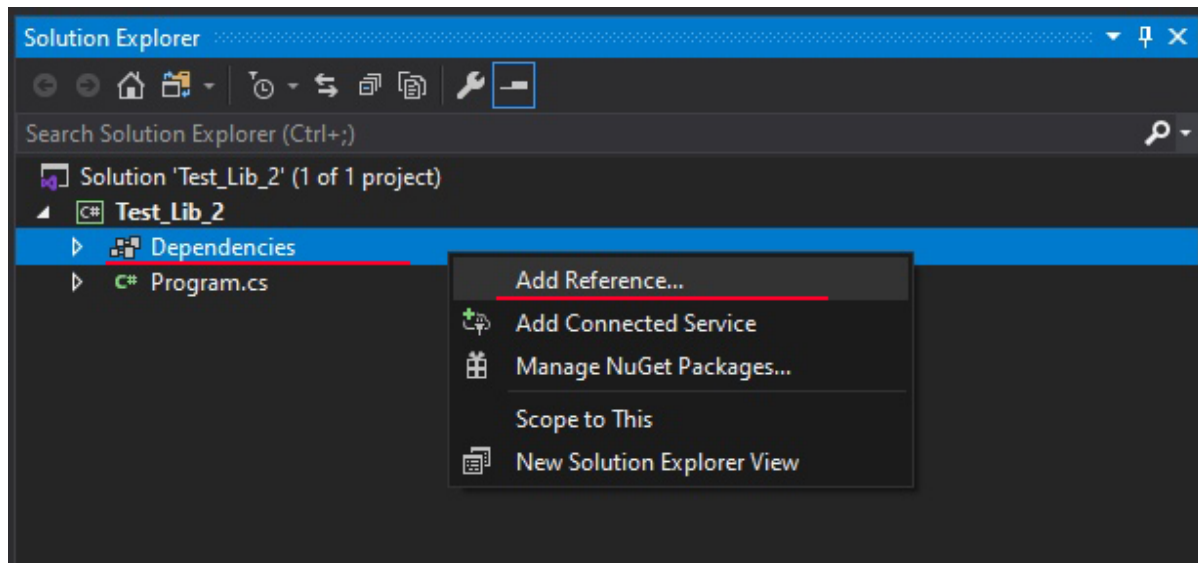


2. Click on **Browse** -> write there **robocadSim** and click Enter -> choose **robocadSimCS** created by **crackanddie** or **Abdrakov corp.** and click on install button

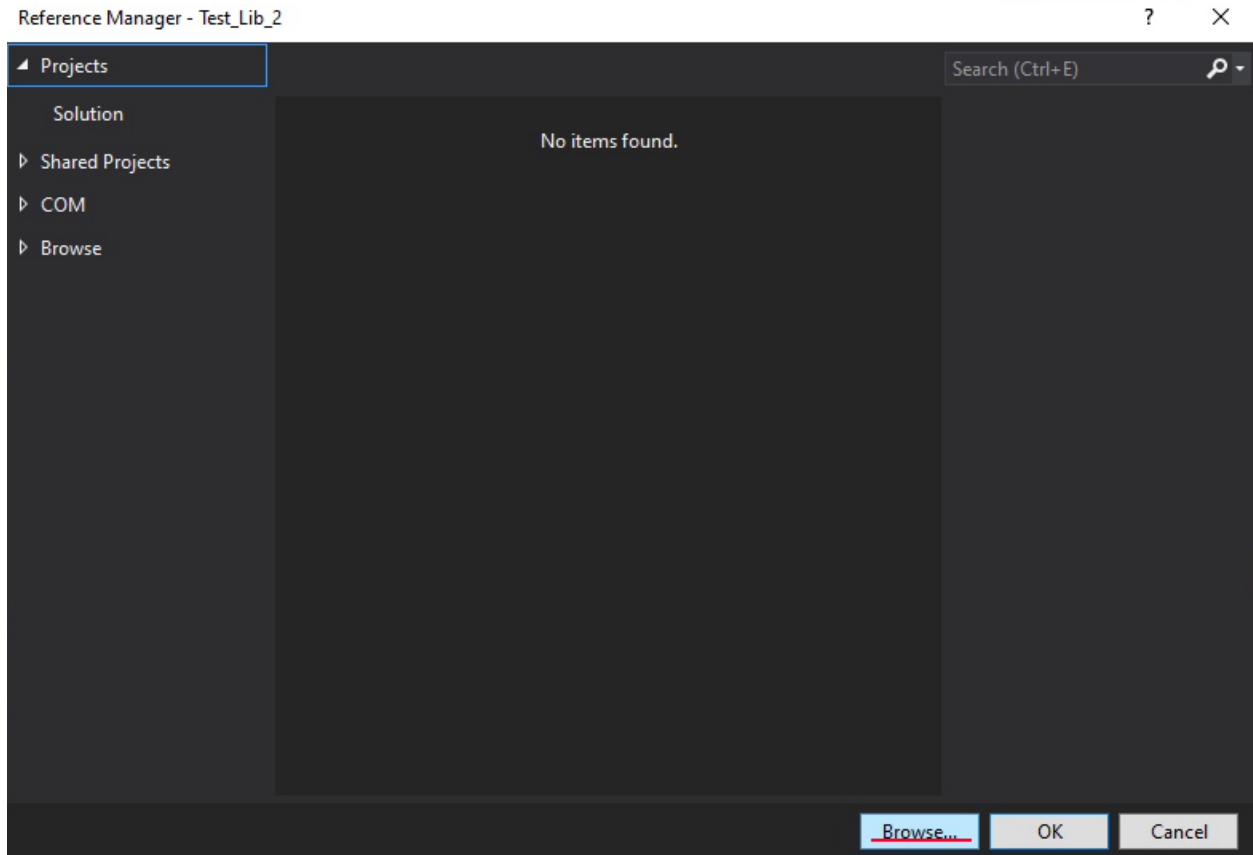


5.2.2 Second way:

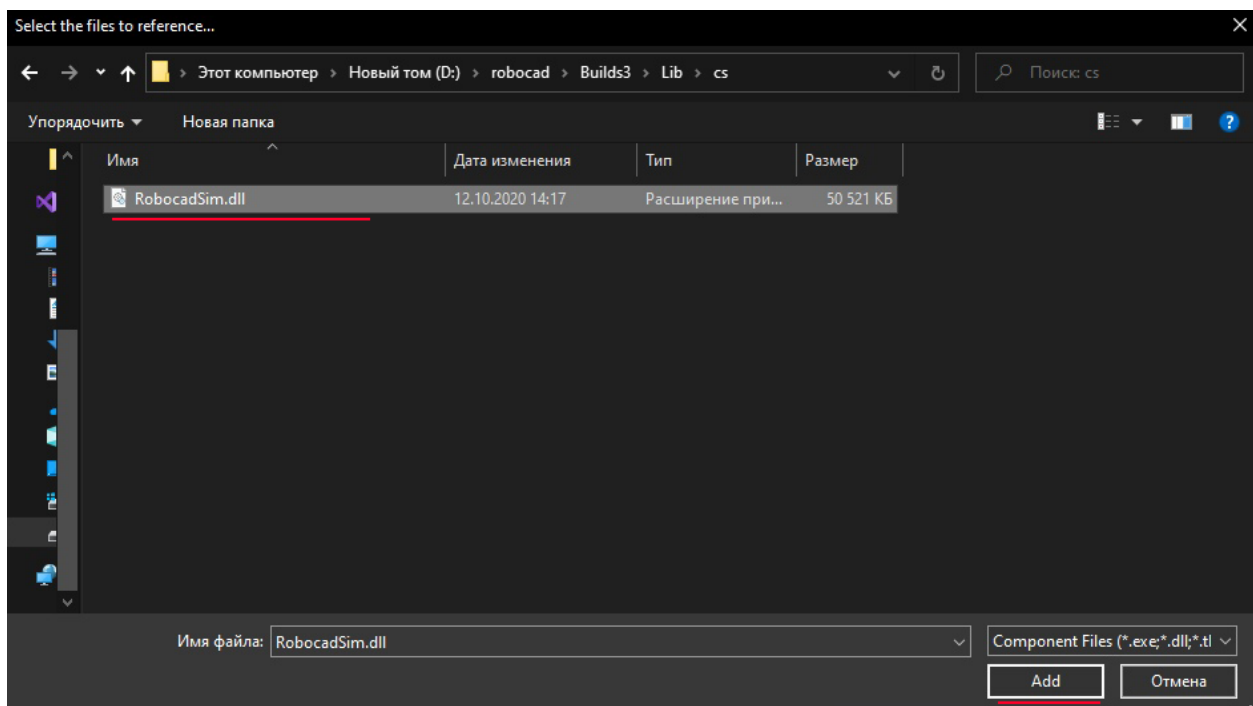
1. Click on Your project name in **Solution explorer** -> right click on **Dependencies** -> **Add reference...**



2. Click on **Browse...**



3. Select **RobocadSim.dll** in **./Lib/cs/** and click **Add**

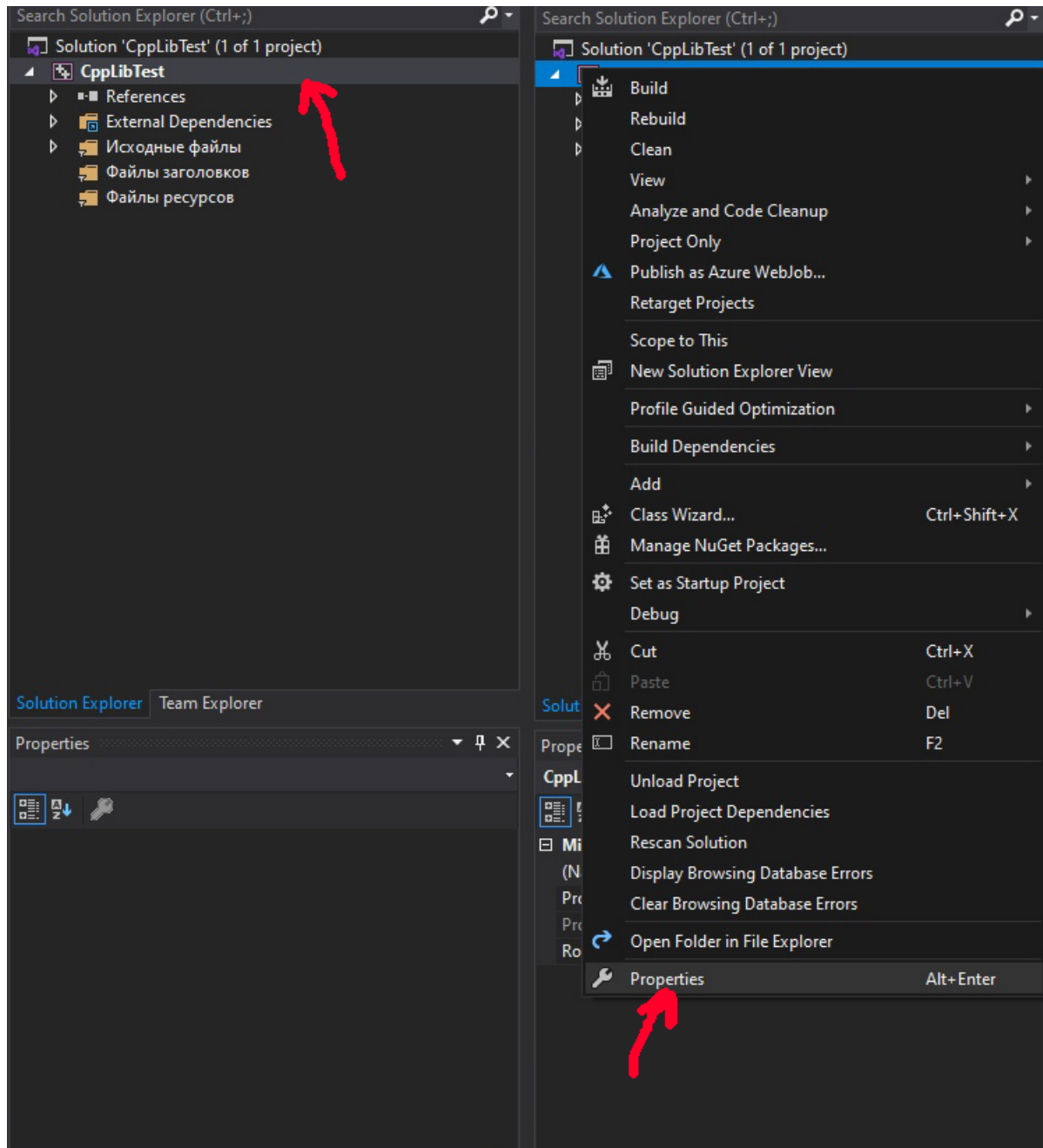


4. Now You can use robocadSim C# library in Your project!

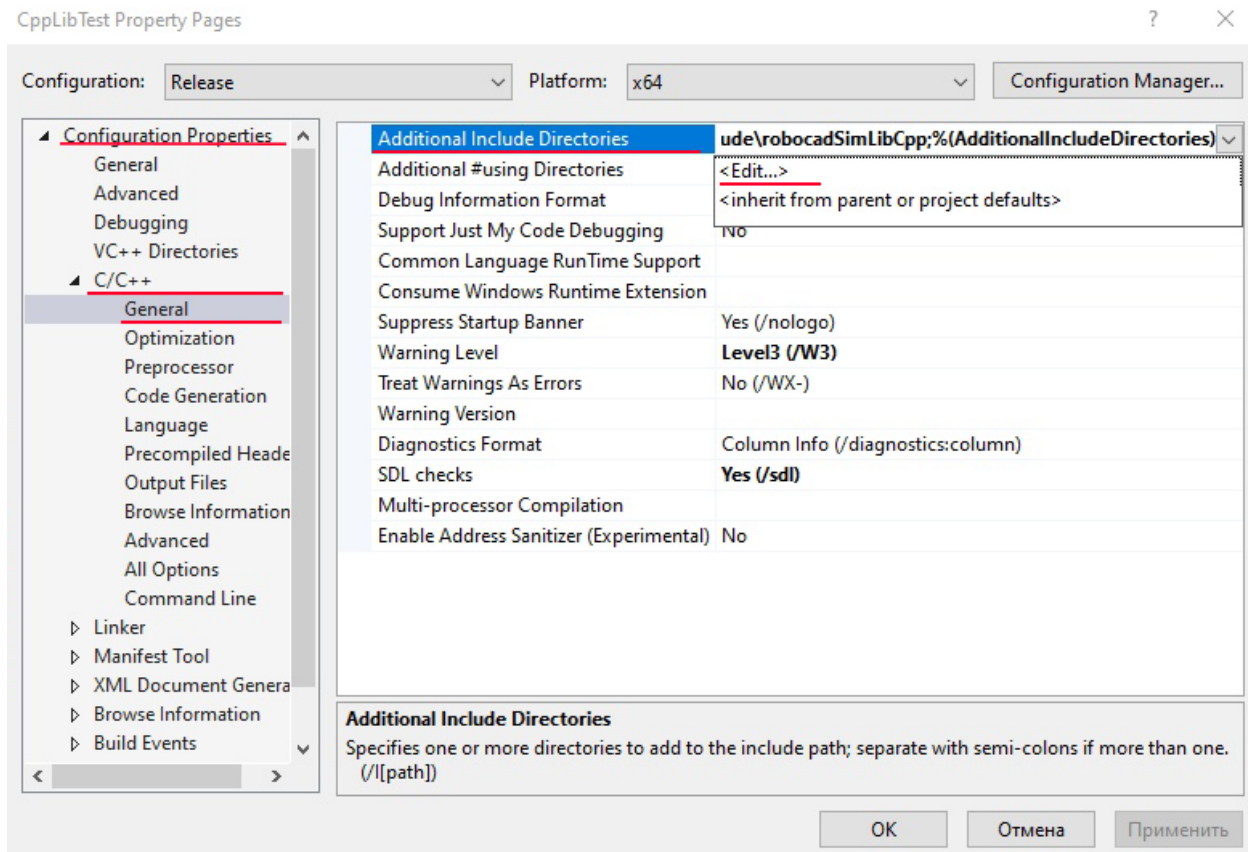
5.3 C++ library

Here is some info about how to use robocadSim C++ library in your project. I am going to use Visual Studio 2019.

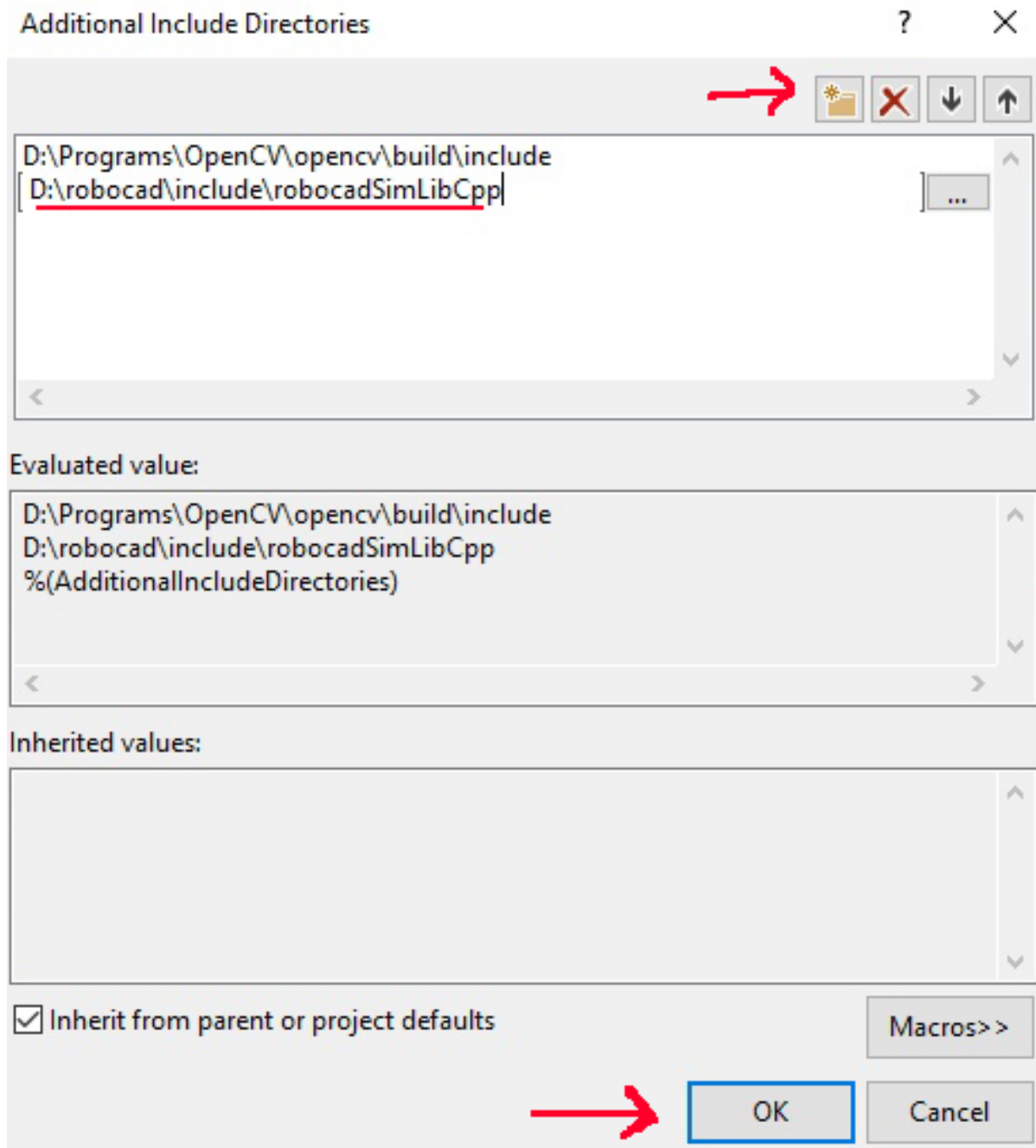
1. You need open-cv installed in your project. ([How to install example](#)).
2. Right click on Your project name in **Solution explorer** -> **Properties**



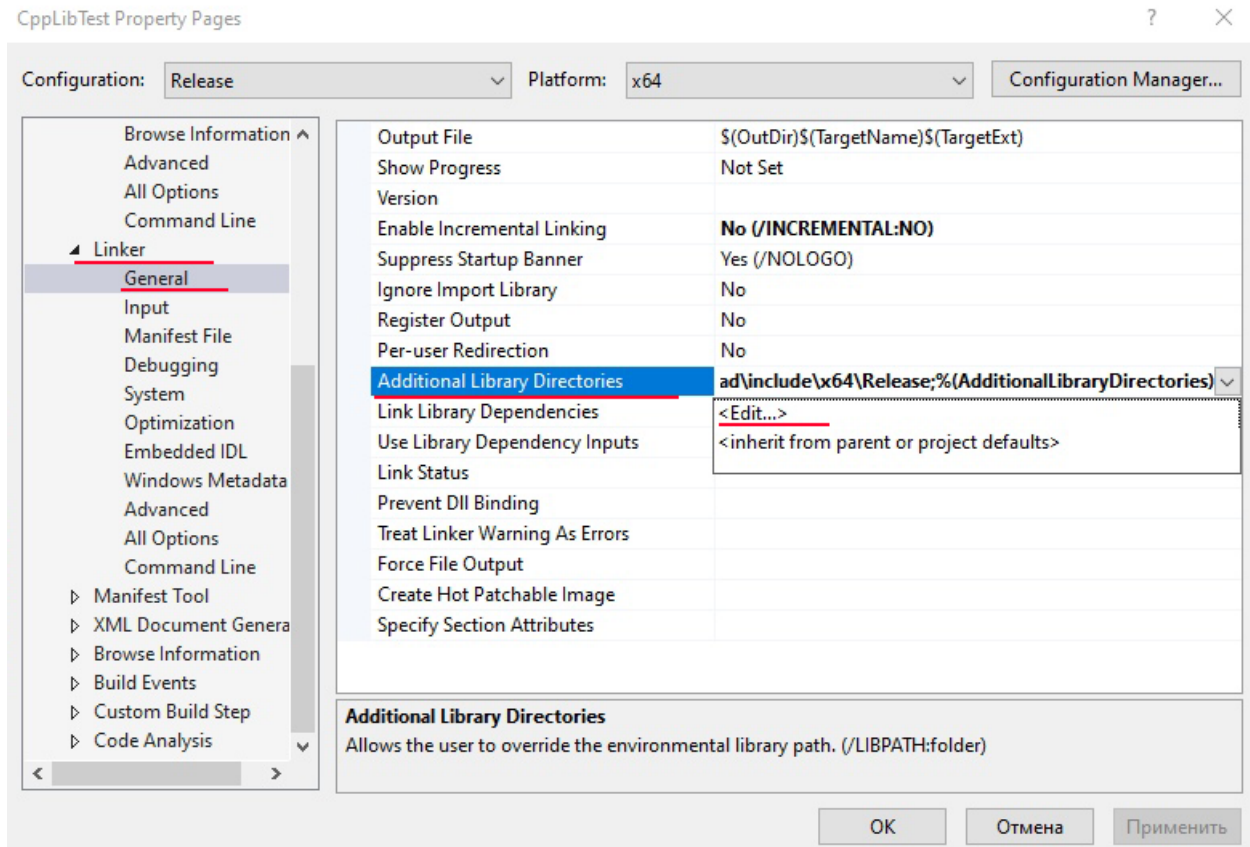
3. Click on **Configuration Properties** -> **C/C++** -> **General** -> **Additional Include Directories** -> **Edit**



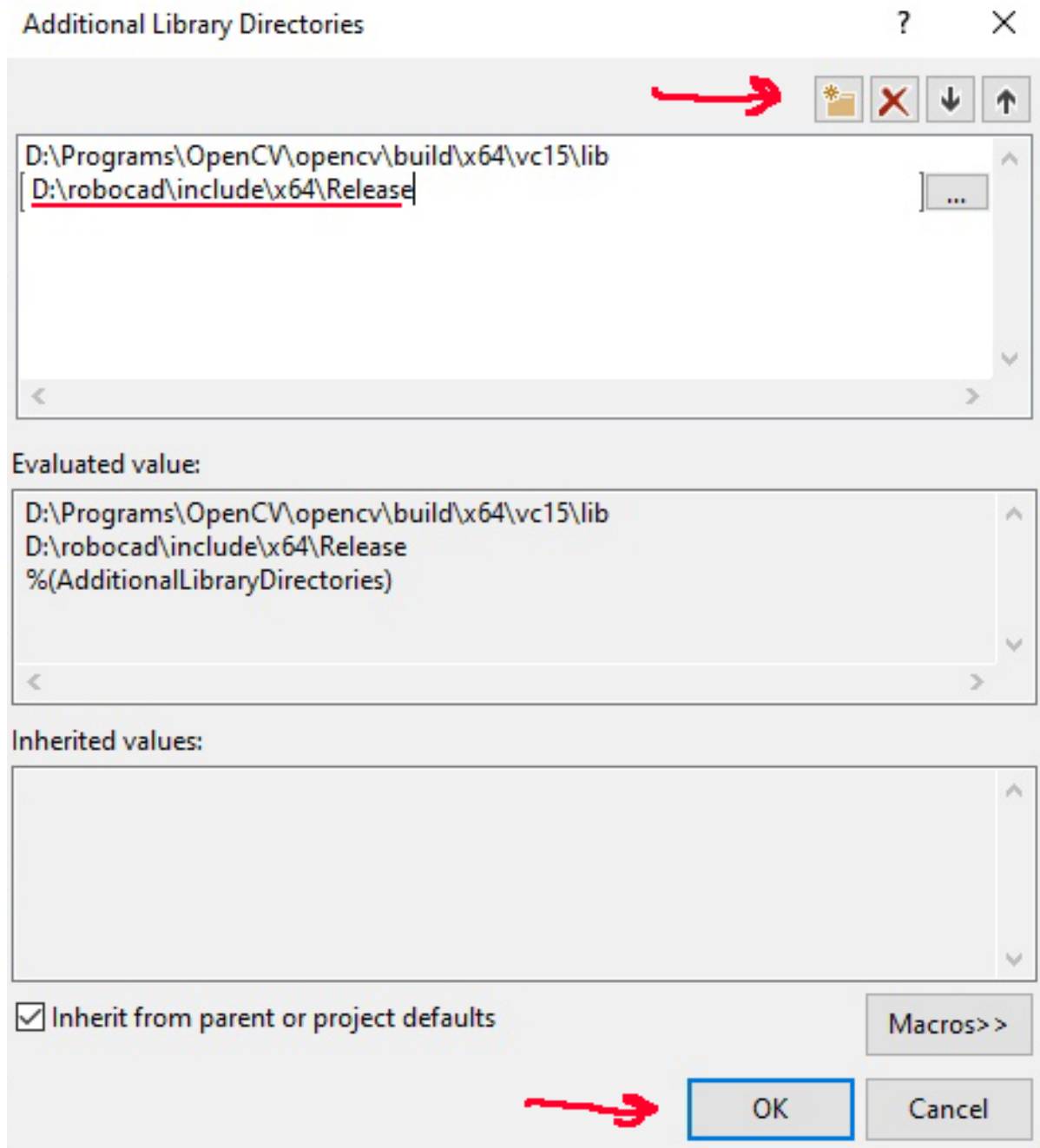
4. Create a new line and paste here path to C++ header files (./robocadSim/Lib/cpp/robocadSimLibCpp) -> click **OK**



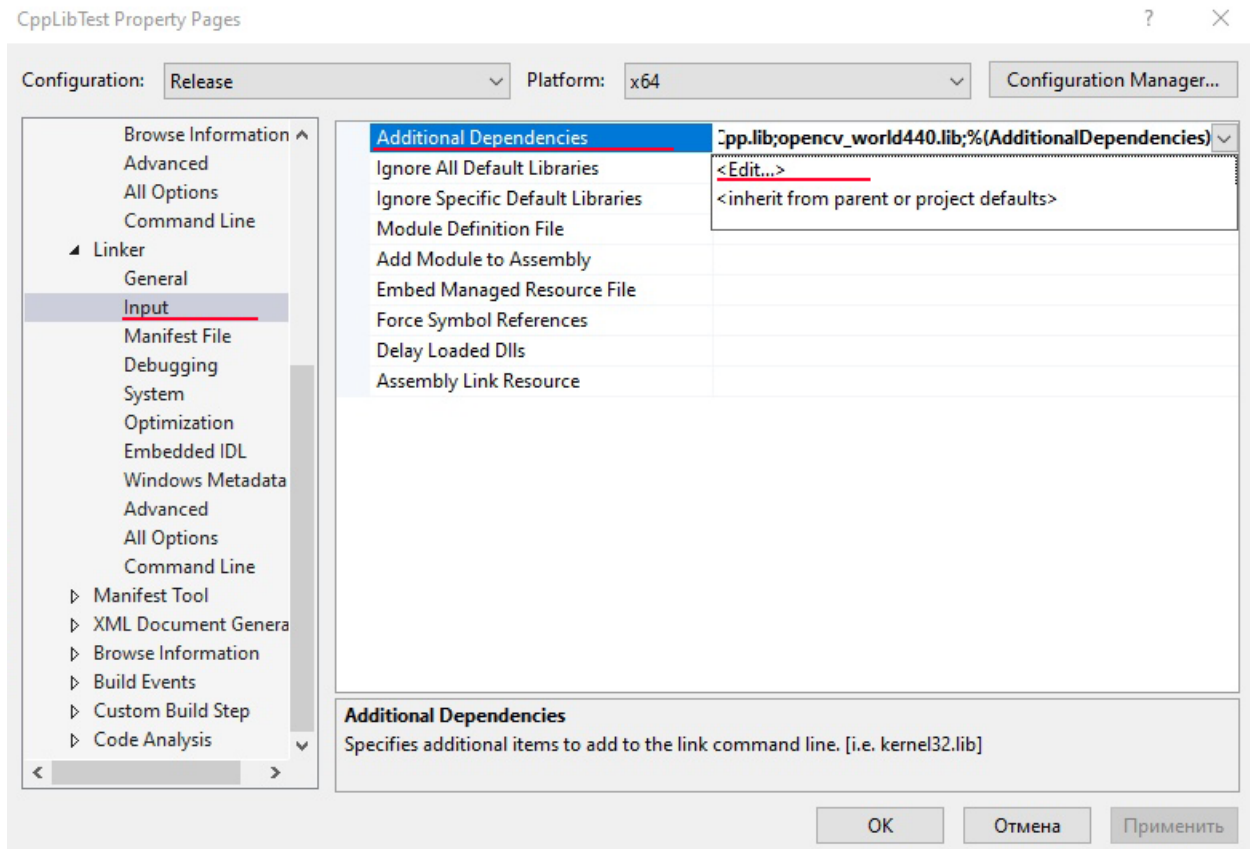
5. Go to **Linker -> General -> Additional Library Directories -> Edit**



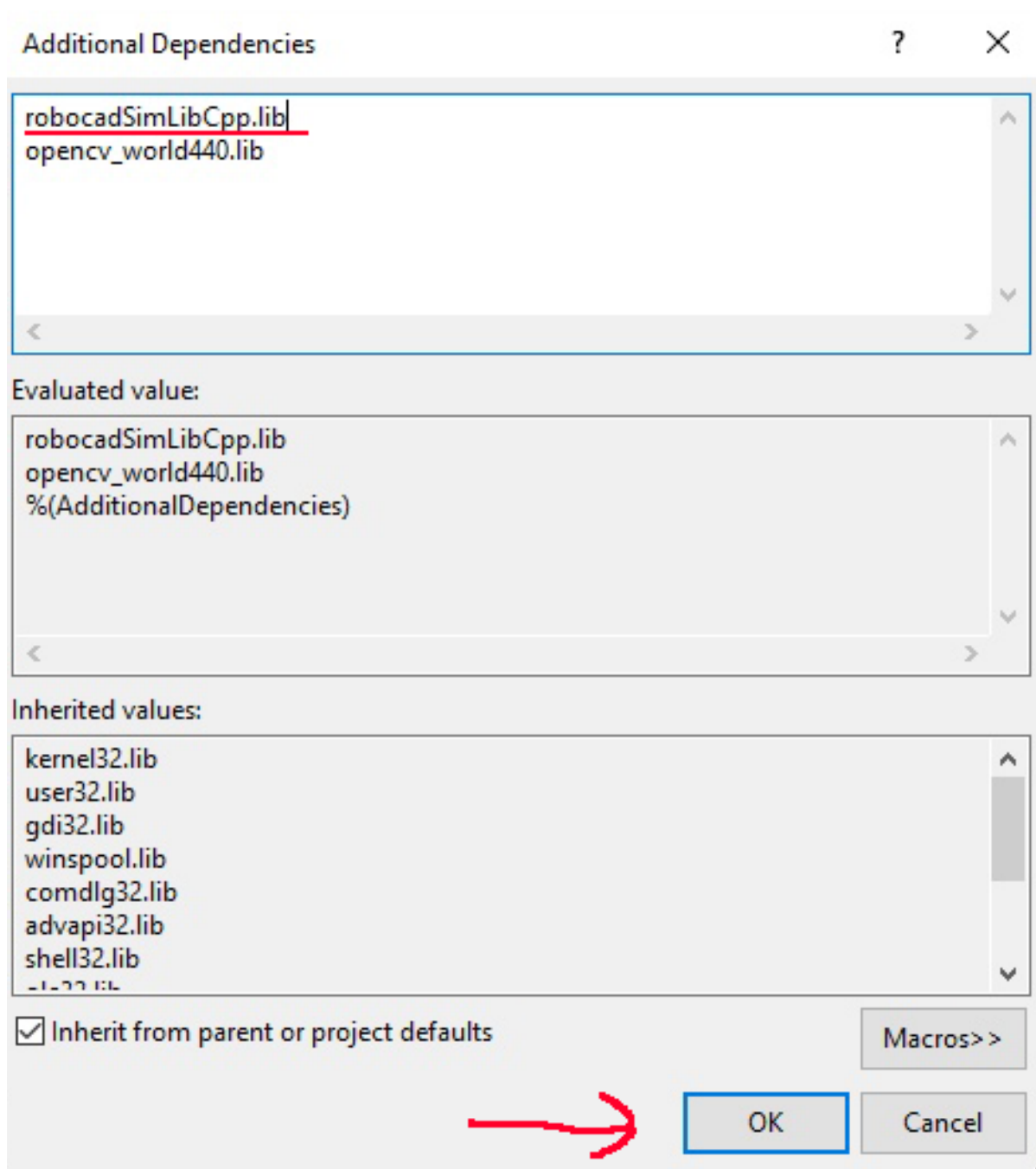
6. Create new line and paste here path to **.lib** file (./robocadSim/Lib/cpp/x64/Release) -> click **OK**



7. Go to **Linker** -> **Input** -> **Additional dependencies** -> **Edit**



8. Paste here robocadSimLibCpp.lib line -> click **OK**



9. Now You can use robocadSim C++ library in Your project!

5.3.1 If You can't use some header files:

1. Copy **.dll** file in robocadSim release folder (./robocadSim/Lib/cpp/x64/Release)
2. Paste it to the path: **path_to_your_project/your_project_name/your_project_name/**

ROBOCADSIM DOCUMENTATION

Here You can find some info about robocadSim program.

6.1 Hot Keys and Keys

6.1.1 Keys:

- **Esc:** open “pause” menu

6.1.2 Hot Keys:

- **LeftCtrl + E:** open power panel
- **LeftCtrl + D:** open control panel

BE OUR SPONSOR

If you want to help the project financially, then you can send money to any of these accounts:

- **PayPal:** paypal.me/crackanddie
- **Qiwi:** qiwi.com/p/79656098785
- **Visa:** 4000 7934 7377 7474

Thanks!

NEED HELP

If You are have any troubles please contact me:

- Inst: [robocadSim](#)
- Email: robocadsim@gmail.com
- Facebook: [RobocadSim](#)

LICENSE**MIT License**

Copyright (c) 2020 Airat

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.