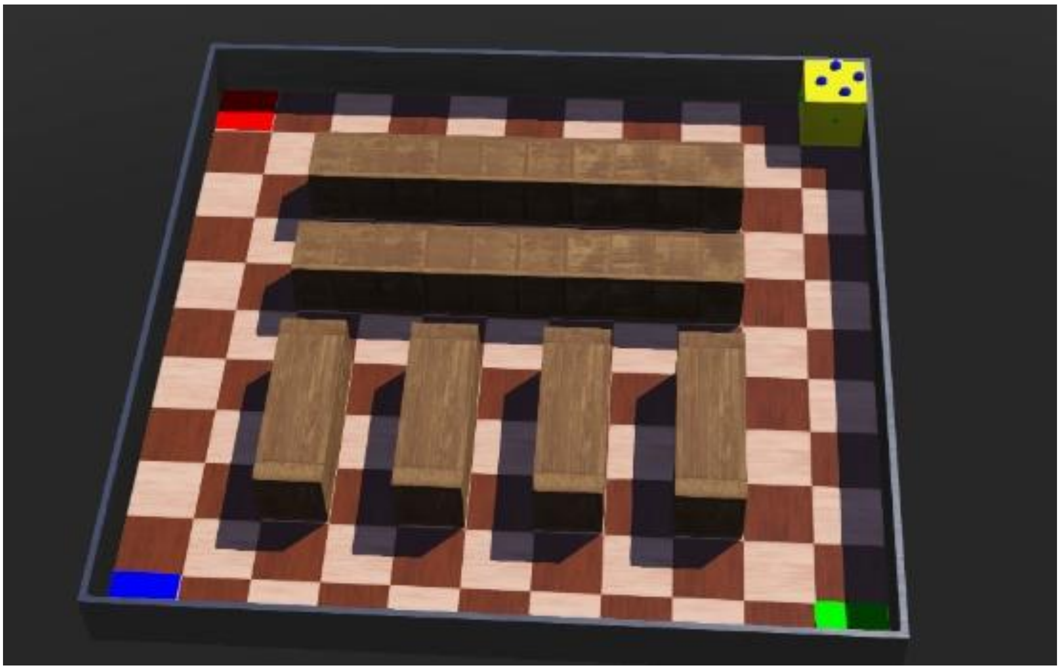
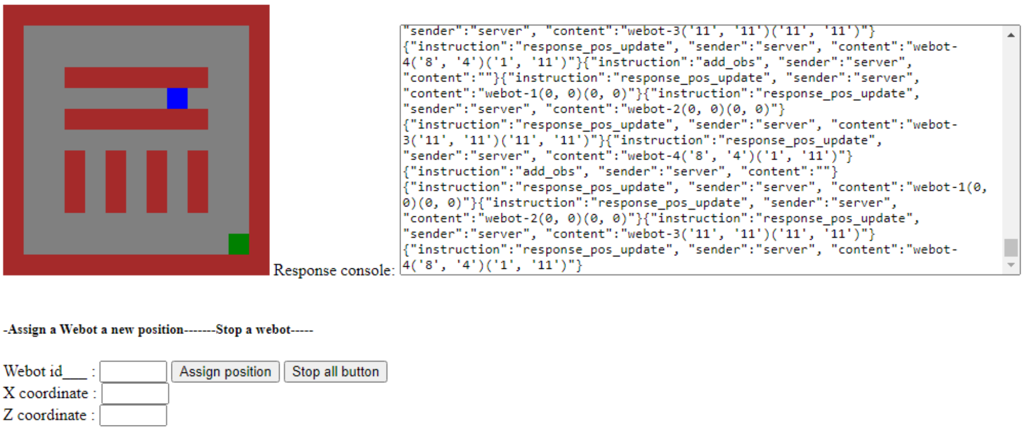
Connectes Systems protocol Webots.





Response console:

```
"sender":"server", "content":"webot-3('11', '11')('11', '11')"}
{"instruction":"response_pos_update", "sender":"server", "content":"webot-
4('8', '4')('1', '11')"}{"instruction":"add_obs", "sender":"server",
"content":""}{"instruction":"response_pos_update", "sender":"server",
"content":"webot-1(0, 0)(0, 0)"}{"instruction":"response_pos_update",
"sender":"server", "content":"webot-2(0, 0)(0, 0)"}
{"instruction":"response_pos_update", "sender":"server", "content":"webot-
3('11', '11')('11', '11')"}{"instruction":"response_pos_update",
"sender":"server", "content":"webot-4('8', '4')('1', '11')"}
{"instruction":"add_obs", "sender":"server", "content":""}
{"instruction":"response_pos_update", "sender":"server", "content":"webot-1(0,
0)(0, 0)"}{"instruction":"response_pos_update", "sender":"server",
"content":"webot-2(0, 0)(0, 0)"}{"instruction":"response_pos_update",
"sender":"server", "content":"webot-3('11', '11')('11', '11')"}
{"instruction":"response_pos_update", "sender":"server", "content":"webot-
4('8', '4')('1', '11')"}
```

-Assign a Webot a new position-------Stop a webot-----

Webot id___ :

X coordinate :

Z coordinate :

Contents -- Protocol messages and their functionalities :

Protocol format :

All messages are sent via websockets in Python and Javascript via ws://145.24.222.208:8765

They all need to keep the format:
‘{“instruction”:”{instruction}”,”sender”:”{sender}”,”content”:”{content}”}’

This is necessary in order to parse the string received via the websocket to a JSON object that can be used in the sourcecodes.

This document will describe the functionalities of the messages and give a syntax example of how they might look like.

Webot to server

1 Instruction : send_coordinate.

2 Instruction : request_instr

3 Instruction : send_obstacle

Server to Webot

Response to instruction 1

4 Instruction : response

5 Instruction : response_instr

6 Instruction : response_instr

Response to instruction 2

7 Instruction : response_instr

8 Instruction : response_instr

Response to instruction 3

9 Instruction : response

Webpage to server

10 Instruction : request_obs

11 Instruction : request_pos_update

12 Instruction : changePos

13 Instruction : stopwebots

Server to webpage

Response to instruction 10

14 Instruction : add_obs

15 Instruction : add_obs

Response to instruction 11

16 Instruction : response_pos_update

Response to instruction 12 & 13

17 – No reply

18 – No reply

Protocol messages and their functionalities:

Messages sent fromt webot to the server :

1. Send the coordinates of a webot and its id to the server. The server keeps track of the coordinates of all the webots. The website might request the locations of the webots later for drawing out the map.

```
'{"instruction":"send_coordinate", "sender":'+ str(WEBOTID) + ', "co
ntent":"(' + str(x) + ',' + str(y) + ')"}'
```

2. Request open tasks from website given by the server.
   Server needs to know who to send the task back to, therefor the webot has to give its id. It also gives its coordination to check wether the task is already achieved or not.

```
'{"instruction":"request_instr", "sender":'+ str(WEBOTID) + ', "cont
ent":"' + str(x) + ',' + str(z) + '"}'
```

3. This message is sent to the server once a sensor on a webot notifies an obstacle. Its very similar to the previous message, the only difference would be the coordinates. Instead, its giving the server the coordinates of the obstacle to the server. The server then marks the coordinates of the obstacle in a 2D list and puts the coordinates inside a different list. The coordinates then later can be popped by the website to draw on the map, but we've turned this functionality off because we've already preplaced the obstacles on the drawn map.

```
'{"instruction":"send_obstacle", "sender":'+ str(WEBOTID) + ', "cont
ent":"(' + str(x) + ',' + str(y) + ')"}'
```

Messages sent by the server to webot responding to instruction 1 :

4. Response sent to the webot after the webot coordinates equals the destination coordinates.

```
'{"instruction":"response", "sender":"server",
"content":"Goal reached"}'
```

5. The webots get instructions one at a time, so whenever there is an open task, it'll also check if the goal is reached yet. In case it isn't it won't reply with a "goal reached" response, but instead with a 'response_instr' reply. This reply contains the next coordinate the webot has to travel to.

```
'{"instruction":"response_instr", "sender":"server", "content":"'+ 
str(xpos) + "," + str(ypos) + '"}'
```

6. Whenever the webot does not have a task or reached a goal, it'll send the webot the content : "No instructions" notifying the webot it does not need to make any movements.

```
'{"instruction":"response_instr", "sender":"server",
"content":"No instructions"}'
```

Messages sent by the server to the webot responding to instruction 2 :

7. Whenever the webot requests instructions, the server will respond by giving a coordinate in the right direction.

```
'{"instruction":"response_instr", "sender":"server", "content":"'+ 
str(xpos) + "," + str(ypos) + '"}'
```

8. When there are no instructions given by the website, the server will send back empty content to the webot.

```
'{"instruction":"response_instr", "sender":"server", "content":""}'
```

Messages sent by the server to the webot responding to instruction 3 :

9. After receiving the obstacle coordinate from a webot, the server will reply by notifying the webot it has successfully received the position of obstacle.

```
'{"instruction":"response", "sender":"server",
"content":"Obstacle observed"}'
```

Messages sent by the webpage to the server :

10. This messages requests the server on wether there are any obstacles that need to be drawn. Does not require any content.

```
'{"instruction":"request_obs", "sender":"dashboard", "content":""}'
```

11. This message requests the server to return the coordinates of given webot so it can draw them on the map.

```
'{"instruction":"request_pos_update", "sender":"dashboard", "content":'+ i +'}'
```

12. Message changing the position of a webot. Content requires you to give the ID of the webot you want to change, the new x position and the new z position. It needs to have commas between the numbers so the server can easily split the data into an array.

```
'{"instruction":"changePos", "sender":"dashboard", "content":"'+document.getElementById("wid").value +','+document.getElementById("xpos").value+','+document.getElementById("zpos").value+ '"}'
```

13. A button on the website will stop all the webots by sending a “stopwebots” instruction. The instruction will cause the server to remove all the currently open tasks.

```
'{"instruction":"stopwebots", "sender":"dashboard", "content":""}'
```

Messages sent by the server to webpage responding to instruction 10 :

14. Whenver there are obstacles that need to be drawn, the server will return the coordinates of the obstacle by popping it from the list.

```
'{"instruction":"add_obs", "sender":"server", "content":"'
+ self.obstacleStack.pop() +'"}'
```

15. Whenever it can’t pop anything in the list, the server will throw an exception. The exception gets caught and it’ll return empty content.

```
'{"instruction":"add_obs", "sender":"server", "content":""}'
```

Messages sent by the server to webpage responding to instruction 11 :

16. Response to a webot position update. Once received, the server will redraw the correct position of the webot on the map. Content requires the id of the webot, its current location and the old location. (Apart from needing to redraw its new location, it’ll also redraw the old position back to grey)

```
'{"instruction":"response_pos_update", "sender":"server",
"content":"webot-'+ str(packet['content'])
+ str(self.webotCoordinates[packet['content']])
+ str(self.oldWebotCoordinates[packet['content']]) + '"}'
```

Message sent by the server to webpage responding to instruction 12 :

17. – This instruction does not return any messages or confirmation.

Message sent by the server to webpage responding to instruction 13 :

18. – This instruction does not return any messages or confirmation.