



Robot Web Tools

LIBRARIES TOOLS ROS NODES

Open-source libraries and tools for building web-based robot apps with ROS.

Libraries

```
19 rclnodejs.init().then(() => {  
20   const node = rclnodejs.createNode('publisher_exam  
21   const publisher = node.createPublisher('std_msgs/  
22  
23   let counter = 0;  
24   setInterval(() => {  
25     console.log(`Publishing message: Hello ROS ${co  
26     publisher.publish(`Hello ROS ${counter++}`);  
27   }, 1000);  
28  
29   rclnodejs.spin(node);
```

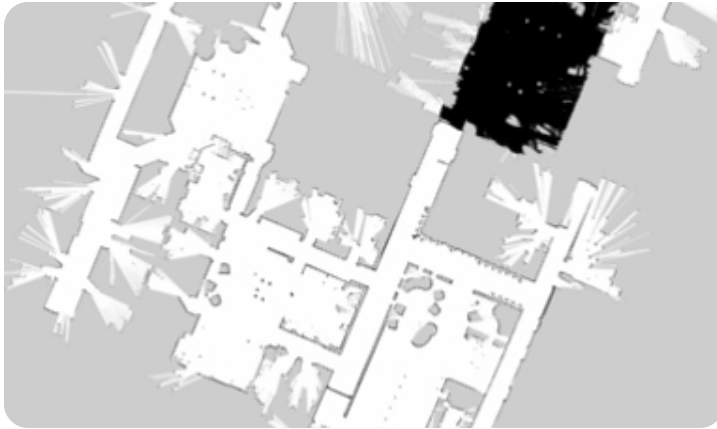
rclnodejs

Node.js client library for the ROS 2

```
98 var feed  
99   ros : this.ros,  
100   name : this.serverName  
101   messageType : this.actionName  
102   });  
103  
104   var statusListener = new ROSLIB.Topic({  
105     ros : this.ros,  
106     name : this.serverName + '/status',  
107     messageType : 'actionlib_msgs/GoalStatusA  
108   });  
109  
110   var resultListener = new ROSLIB.Topic({  
111     ros : this.ros,  
112     name : this.serverName + '/result',  
113     messageType : this.actionName + 'Resu
```

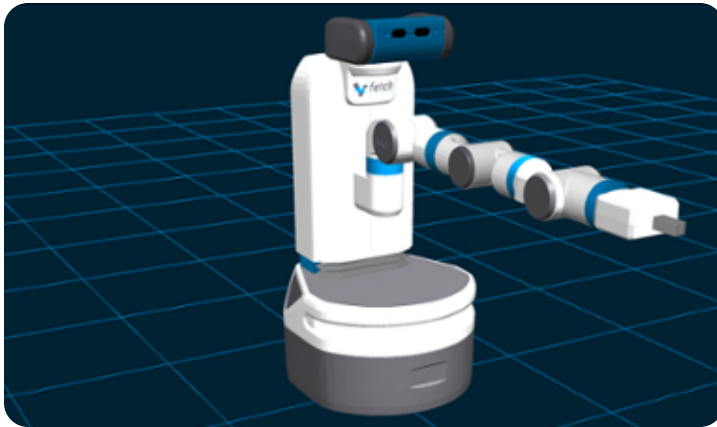
roslibjs

ROS Javascript Library



ros2djs

2d visualization library for use with the ros javascript libraries



ros3djs

3d visualization library for use with the ros javascript libraries

```
const { open } = require('rosbag');

// open a new bag at a given file location:
const bag = await open('../path/to/ros.bag');

// read all messages from both the '/foo' and '/bar' topics:
await bag.readMessages({ topics: ['/foo', '/bar'] }, (result) => {
  // topic is the topic the data record was in
  // in this case it will be either '/foo' or '/bar'
  console.log(result.topic);

  // message is the parsed payload
  // this payload will likely differ based on the topic
  console.log(result.message);
});
```

rosbag.js

Read bag files in javascript

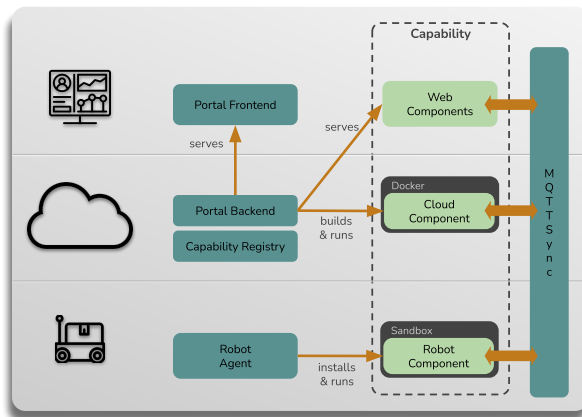
```

1  const rosnodejs = require('rosnodejs');
2  rosnodejs.initNode('/my_node')
3    .then((nodeHandle) => {
4
5      const sub = nodeHandle.subscribe('/chatter',
6        'std_msgs/String', (msg) => {
7        console.log('Got msg on chatter: %j', msg);
8      });
9
10     const pub = nodeHandle.advertise('/chatter',
11       'std_msgs/String');
12     pub.publish({ data: "hi" });
13   });

```

rosnodejs

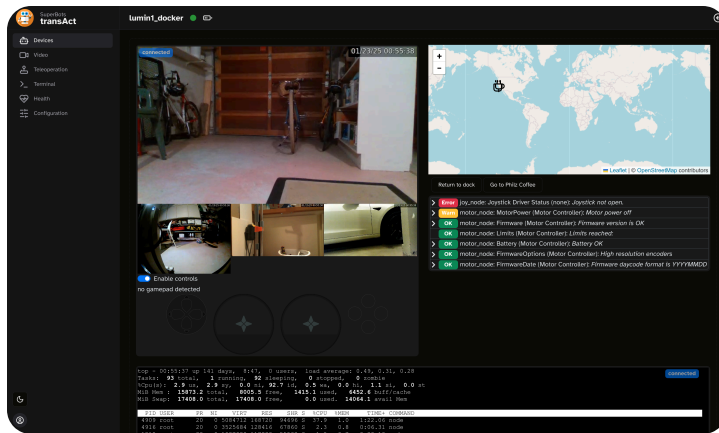
Native Node.js ROS1 client implementation



Transitive

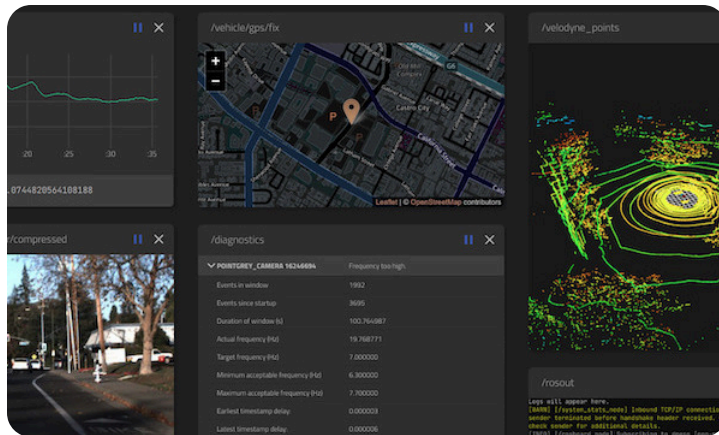
A framework for building robot web portals

Tools



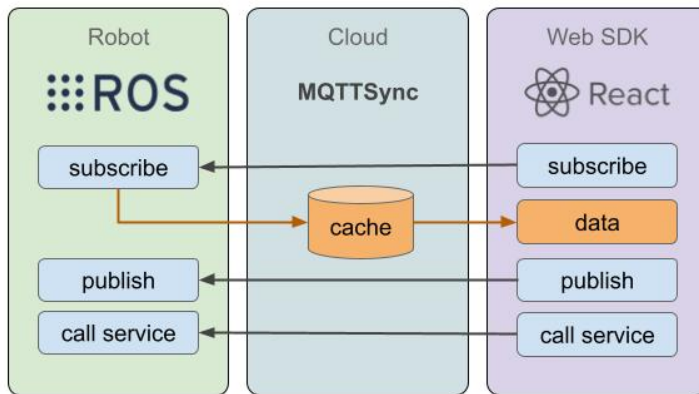
transAct

An open-source robot fleet management dashboard



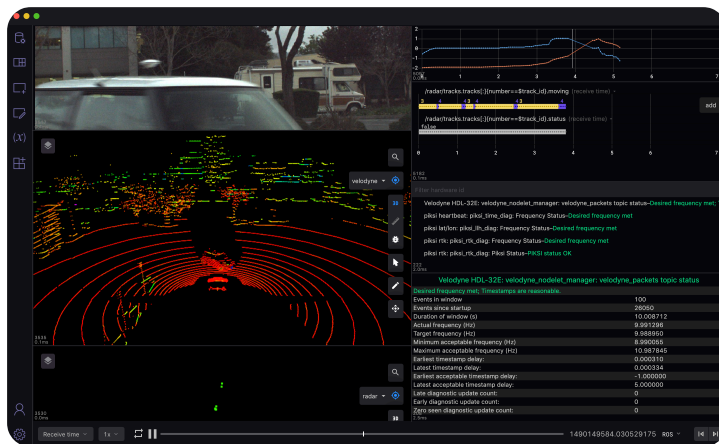
ROSboard

turn your robot into a web server to visualize ROS topics



ROS Tool

A React and Node.js API for communicating with ROS robots from web and cloud



Foxglove Studio

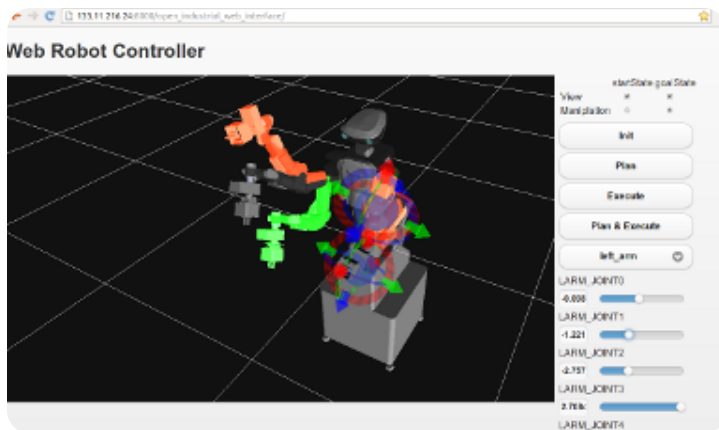
Visualization and debugging tool for live or recorded data

Nodes



rosbridge

websocket interface to ROS systems



visualization rwt

suite of nodes for web based robot visualization

Image Topics using WebRTC

WebRTC peer that can be configured to stream a ROS image topic. The node hosts a webserver that serves the stream. The node can be used to create and configure a WebRTC peer.

For more information, see [the ROS wiki](#).

Developed as part of the [Robot Web Tools](#) effort.

Webrtc ROS

Streaming of ROS Image Topics using WebRTC

