# Programming with rcl and rclc

## Creating a Node

To simplify the creation of a node with rcl, rclc provides two functions: rclc\_support\_init(..)

rclc\_node\_init\_default(..)

in [rclc/init.h](https://github.com/micro-ROS/rclc/blob/master/rclc/include/rclc/init.h) and [rclc/node.h](https://github.com/micro-ROS/rclc/blob/master/rclc/include/rclc/node.h), respectively.

rcl\_allocator\_t allocator = rcl\_get\_default\_allocator();

rclc\_support\_t support;

rcl\_ret\_t rc;

rc = rclc\_support\_init(&support, argc, argv, &allocator);

if (rc != RCL\_RET\_OK)

{

... // Some error reporting.

return -1;

}

rcl\_node\_t my\_node = rcl\_get\_zero\_initialized\_node();

rc = rclc\_node\_init\_default(&my\_node, "my\_node\_name", "my\_namespace", &support);

if (rc != RCL\_RET\_OK)

{

... // Some error reporting.

return -1;

}

## Publishers and Subscriptions

Creating a publisher by rclc\_publisher\_init\_default(..) from [rclc/publisher.h](https://github.com/micro-ROS/rclc/blob/master/rclc/include/rclc/publisher.h):

rcl\_publisher\_t my\_pub;

std\_msgs\_\_msg\_\_String my\_msg;

const char \* my\_topic = "topic\_0";

const rosidl\_message\_type\_support\_t \* my\_type\_support = ROSIDL\_GET\_MSG\_TYPE\_SUPPORT(std\_msgs, msg, String);

rc = rclc\_publisher\_init\_default(&my\_pub, &my\_node, &my\_type\_support, &my\_topic\_name);

if (RCL\_RET\_OK != rc)

{

printf("Error in rclc\_publisher\_init\_default.\n");

return -1;

}