#include <rclcpp/rclcpp.hpp>

#include <geometry\_msgs/msg/twist.hpp>

#include <chrono>

#include <thread>

class DrawSquare : public rclcpp::Node {

public:

DrawSquare() : Node("draw\_square") {

// 创建一个发布者，发布到 /turtle1/cmd\_vel 话题

publisher\_ = this->create\_publisher<geometry\_msgs::msg::Twist>("/turtle1/cmd\_vel", 10);

// 创建一个定时器，每秒调用一次回调函数

timer\_ = this->create\_wall\_timer(

std::chrono::seconds(1),

std::bind(&DrawSquare::timer\_callback, this)

);

// 初始化状态

state\_ = FORWARD;

}

private:

enum State { FORWARD, TURN };

void timer\_callback() {

geometry\_msgs::msg::Twist msg;

switch (state\_) {

case FORWARD:

msg.linear.x = 2.0; // 前进

msg.angular.z = 0.0;

forward\_count\_++;

if (forward\_count\_ == 2) {

forward\_count\_ = 0;

state\_ = TURN;

}

break;

case TURN:

msg.linear.x = 0.0;

msg.angular.z = M\_PI / 2; // 90度转弯

turn\_count\_++;

if (turn\_count\_ == 4) {

turn\_count\_ = 0;

state\_ = FORWARD;

}

break;

}

RCLCPP\_INFO(this->get\_logger(), "Publishing: '%f', '%f'", msg.linear.x, msg.angular.z);

publisher\_->publish(msg);

}

rclcpp::Publisher<geometry\_msgs::msg::Twist>::SharedPtr publisher\_;

rclcpp::TimerBase::SharedPtr timer\_;

State state\_;

int forward\_count\_ = 0;

int turn\_count\_ = 0;

};

int main(int argc, char \* argv[]) {

rclcpp::init(argc, argv);

auto node = std::make\_shared<DrawSquare>();

rclcpp::spin(node);

rclcpp::shutdown();

return 0;

}