5249Z-Ignite API

Version: 1.0

RobotConfig.h

void wait(int time)

Description: Waits the specified amount of time in milliseconds

int driver()

Description: The function called by main.cpp when driver control is run

Returns:

0; the vex::task requires a callback function that returns an int

int auton()

Description: The function called by main.cpp when autonomous is run

Returns:

0; the vex::task requires a callback function that returns an int

bool confirmDriver()

Description: Checks whether the robot is allowed to run driver control

Returns: bool whether driver is allowed

bool confirmAuton()

Description: Checks whether the robot is allowed to run auton control

Returns: bool whether auton is allowed

RobotMethods.h

void intake(double speed)

Description: Runs the intake at the specified speed

void intakeStop(brakeType stopMode = brakeType::coast)

Description: Stops the intake motors with the specified braking type (default is coast)

void chassisLeft(double speed)

Description: Runs the left side of the chassis at the specified speed

void chassisRight(double speed)

Description: Runs the right side of the chassis at the specified speed

void arm(double speed)

Description: runs the arm at the specified speed(negative is up)

void armStop(brakeType stopMode = brakeType::coast)

Description: Stops the arm motor with the specified braking type (default is coast)

void rampLift(double speed)

Description: Runs the ramp at the specified speed

void rampLiftStop(brakeType stopMode = brakeType::coast)

Description: Stops the arm motor with the specified braking type (default is coast)

bool cubesClear()

Description: uses the line tracker to determine whether a cube is at the bottom of the ramp

Returns: bool whether a cube is blocking the sensor

void deployRobot()

Description: unfolds the robot at the start of a match, brace for impact

bool liftRamp(bool moveUp, double slow = 15, double fast = 50, bool outtake = false)

Description: Macro that moves the ramp up or down; does not wait for the ramp to reach its target, must be used in a loop. The ramp will start moving fast and then slow when it reaches its target position

Parameters:

bool moveUp: true; move the ramp up, false; move the ramp down

double slow: speed that the ramp will reach at the end of its motion (default is 15)

double fast: speed that the ramp will run at the start of its motion (default is 50)

bool outtake: whether or not to outtake the cubes until they obstruct the line tracker (default is false)

Returns:

bool whether the ramp has reached its target position

void stackTower(bool waitForCubes = true)

Description: fully stacks the cubes in front of the robot; waits for it to be completed

Parameters:

bool waitForCubes: whether or not to outtake until the cubes obstruct the line tracker

NavMethods.h

int drivePID()

Description: Function called in auton in a task declaration that constantly adjust the robot’s position to the set values.

Returns:

0; vex::task requires a function that returns an int

void resetPosition()

Description: Sets the robot's position to longitude 0 and angle 0°.

void driveToPos(double distance)

Description: Drives the robot forward the specified distance.(Only sets the distance, drivePID() actually drives the robot)

void turnToAngle(double angle)

Description: Turns the robot to the specified angle.(Only sets the angle, drive() actually drives the robot)

double longitudeError()

Returns the distance the robot needs to travel to reach its target distance

double yawError()

Returns the difference between the set angle and the current angle

PID.h

class PID

PID(double kP, double kI, double kD, double samplePeriod)

Description: constructor for PID class, sets initial values

Parameters:

double kP, kI, kD: gain values for the PID

double samplePeriod: time between each run of the PID

double setPoint: The setpoint for the PID

double calculatePID(double processVar)

Description: returns the correction for one run of the PID.

Parameter:

double processVar: the current process variable for the PID

Returns: the calculated correction value

void reset()

Description: resets the integral sum and the previous error for the PID to 0.

void setGains(double kP, double kI, double kD)

Description: sets the gain values of the PID

Parameters:

double kP, kI, kD: gain values for the PID