

朱俊泽

**电机驱动：**

7-27

**1.void vex::motor::spin(directionType dir, double voltage, voltageUnits units):**

程序组

参数dir：旋转方向，forward正向，reverse反向，stop停止，设置之后控制电机如何旋转（相对于自己定义的时候）

参数voltage：旋转的电压，是电机旋转的速度力量控制的单位

参数units：指定了上一个参数的单位，可以选择volts（伏特）或者pct（百分比）。通常，新生赛的时候都使用了pct用来控制电压与最大电压的比例，更加方便控制。

**2.bool vex::motor::spinTo(double rotation, rotationUnits units, double velocity, velocityUnits units\_v, bool waitForCompletion=true)**

参数rotation：设置电机要旋转到的目标位置

参数units：设置rotation的单位。Degrees度数，radians弧度，rotation旋转度

参数velocity：设置电机旋转到目标位置的速度，力量。这个参数过大容易导致不稳定，这个参数过小会很慢。

参数units-v：设置velocity的单位。Rpm每秒转速，radians/second弧度每秒

参数waitforcompletion：缺省为true，此时会停止这个线程，这样设置的话，在旋转电机直到目标角度之前本线程其他函数都不会执行。

**3.void vex::motor::setVelocity(double velocity, percentUnits units)**

参数velocity：设置电机旋转的速度力量

参数units：设置velocity的单位

Spin函数自带pid比较精确，自动的时候选择spin函数。

设置电压setvelocity比较快，手动的时候选择setvelocity函数

**l::inertial(int32 t index, turnType dir=turnType::right)**

惯性传感器的构造函数

参数index：指定imu的索引，区分一些每个控制器的对应传感器

参数turntype dir：指定了imu的正向方向，缺省为right

**2.public bool vex::inertial::installed()**

返还一个bool值表示本imu是否和控制器正常链接

**3.public void vex::inertial::startCalibration(int32\_t value=0)**

开始校准，在这之前需要校准几秒钟

参数value：默认为0，理解为imu的初始值

1. **public void vex::inertial::calibrate(int32\_t value=0)**

进行imu校准

参数value：imu校准的时长，校准个4-5秒

1. **5.public bool vex::inertial::isCalibrating(void)**

判断是不是正在进行校准，是就返回true

**6.public void vex::inertial::resetHeading()**

重新设置imu的heading数值为0

**7.public void vex::inertial::resetRotation()**

重新设置imu的rotation数值为0

1. **public void vex::inertial::setHeading(double value, [rotationUnits](https://api.vexcode.cloud/v5/namespace/namespacevex?member=namespacevex_1a22877525a92a7b34c603d2e97d2bec16) units)**

设置heading数值

参数value:heading的数值

参数units：heading的单位

1. **public void vex::inertial::setRotation(double value, [rotationUnits](https://api.vexcode.cloud/v5/namespace/namespacevex?member=namespacevex_1a22877525a92a7b34c603d2e97d2bec16) units)**

重新设置rotation数值

参数value：rotation的数值

参数units：rotation的单位

**获取imu目前值函数：**

1. **public double vex::inertial::angle([rotationUnits](https://api.vexcode.cloud/v5/namespace/namespacevex?member=namespacevex_1a22877525a92a7b34c603d2e97d2bec16) units=rotationUnits::deg)**
2. **public double vex::inertial::roll([rotationUnits](https://api.vexcode.cloud/v5/namespace/namespacevex?member=namespacevex_1a22877525a92a7b34c603d2e97d2bec16) units=rotationUnits::deg)**
3. **public double vex::inertial::pitch([rotationUnits](https://api.vexcode.cloud/v5/namespace/namespacevex?member=namespacevex_1a22877525a92a7b34c603d2e97d2bec16) units=rotationUnits::deg)**
4. **public double vex::inertial::yaw([rotationUnits](https://api.vexcode.cloud/v5/namespace/namespacevex?member=namespacevex_1a22877525a92a7b34c603d2e97d2bec16) units=rotationUnits::deg)**
5. **（重要）public double vex::inertial::heading([rotationUnits](https://api.vexcode.cloud/v5/namespace/namespacevex?member=namespacevex_1a22877525a92a7b34c603d2e97d2bec16) units=rotationUnits::deg)**

返回imu当前的heading值

1. **（重要）public double vex::inertial::rotation([rotationUnits](https://api.vexcode.cloud/v5/namespace/namespacevex?member=namespacevex_1a22877525a92a7b34c603d2e97d2bec16) units=rotationUnits::deg)**

返回imu当前的rotation值

**Pwm：**

**1.public vex::pwm\_out::pwm\_out([triport::port](https://api.vexcode.cloud/v5/class/classvex_1_1triport_1_1port) &port)**

初始化pwm

参数&port：明确接口

**2.public void vex::pwm\_out::state(int32\_t value, [percentUnits](https://api.vexcode.cloud/v5/namespace/namespacevex?member=namespacevex_1a9c2894217ae8fda59835332195565cc7) units)**

设置pwm的状态

参数value：设置pwm驱动的大小。

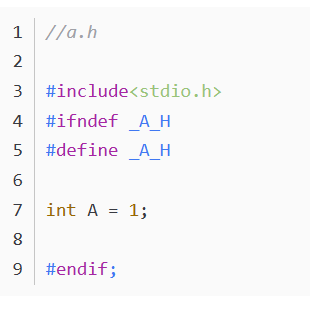
参数units：设置value的单位，per（百分比）。

**工程项目思考题：在大型项目中，多源文件cpp、头文件h、hpp等非常常见，但是在项目编译过程中，很容易会出现嵌套定义等问题，请查找相关资料并给出可能的解决办法**

首先需要避免重复定义，这个问题靠个人代码习惯解决吧，函数命名在头文件，实现在cpp文件里面。

其次要避免重复编译，以下是错误的发生：

如果头文件中定义了类外的函数或者全局变量，那么当多个源文件同时包含这一头文件时，会发生重复定义的错误（如下例 int A =1）。



此时b.h.想针对A操作而包含了对a.h的编译，在这之后c.h也想针对A操作也包含了对a.h的编译。此时int A作为全局变量被编译了两次，导致重复定义。

1.此时可以进行条件编译  
#ifndef \_A\_H

#define \_A\_H

对a来说如果a没有被编译过，那么编译一次a

1. 在vs中可以使用#program once。在头文件开头加上#pragma once
2. 参考文案https://blog.csdn.net/goldcarpenter/article/details/85345767