
python

1

Robot kodunuzu oluşturun!

Robot kodunuz robot py adlı bir dosya içinde başlamalıdır. Kodunuz, diğer python modüllerini ve paketlerini içe aktarmak gibi normal bir python programının yapabileceği her şeyi yapabilir. İşte robot kod çalışma almak için bilmeniz gereken temel şeyler!

2

Gerekli modülleri içe aktarma:

Robotunuzun donanımı ile gerçekten etkileşime giren tüm kod, Wpılıb adlı bir kütüphanede bulunur. Bu kütüphane başlangıçta C++ ve Java'da uygulandı. Robot kodunuz bu kütüphane modülünü içe aktarmalı ve robot donanımı ile arabirim oluşturmak için kullanılabilecek çeşitli nesneler oluşturmalıdır.

Wpilib aktarmak işte bu kadar basit:

```
import wpilib
```

Robot nesne

Her geçerli robot programı, wpılıb'den devralan bir robot nesnesi tanımlamalıdır wpilib.Iterativerobot, wpilib.TimedRobot veya wpilib.SampleRobot. Bu sınıflar, çeşitli zamanlarda çağrılan geçersiz kılmanız gereken bir dizi işlevi tanımlar.

- `wpilib.IterativeRobot` functions
- `wpilib.TimedRobot` functions
- `wpilib.SampleRobot` functions

```
class MyRobot(wpilib.TimedRobot):

    def robotInit(self):
        self.motor = wpilib.Jaguar(1)
```

Not:

Deneyimsiz programcıların bu kılavuzun tartışacağı TimedRobot sınıfını kullanması önerilir.

3

Motorlar ve sensörler ekleme

Robotun ana robot nesnenizde çağrıldığı sırada veya sonrasında sadece Motorlarınızın ve diğer Wpılıb donanım aygıtlarının (Jiroskoplar, Joystickler, sensörler, vb.) örneklerini oluşturmalısınız. Eğer yapmazsanız, başarısız olacak bir çok şey vardır.

4

Bireysel cihazlar oluşturma:

Diyelim ki PWM ile bir Jaguar motor kontrolörü ile etkileşime giren bir nesne oluşturmak istediniz. İlk olarak, tabloyu (wpilib) okuyacak ve bir Jaguar nesnesi olduğunu göreceksiniz. Daha fazla baktığımızda, yapıcının hangi PWM bağlantı noktasına bağlanacağını gösteren tek bir argüman aldığını görebilirsiniz. RobotİnitYönteminizde aşağıdaki python kodunu kullanarak port 4 kullanan Jaguar nesnesini oluşturabilirsiniz:

```
self.motor = wpilib.Jaguar(4)
```

Belgelere biraz daha baktığımızda, motorun PWM değerini ayarlamak için Jaguarı aramanız gerektiğini fark edersiniz.set () işlevi. Dokümanlar, değerin -1.0 ve 1.0 arasında olması gerektiğini söylüyor, bu yüzden motorun tam hızını ayarlamak için bunu yapabilirsiniz:

```
self.motor.set(1)
```

Diğer motorlar ve sensörler benzer sözleşmelere sahiptir

5

Robot drivetrain kontrolü:

Standart tip drivetrains (2 veya 4 tekerlek, mecanum, kiwi) için, bunu yapmak için kendi kodunuzu yazmak yerine motorları kontrol etmek için çeşitli dahil sınıfı kullanmak isteyeceksiniz. Çoğu standart drivetrains için, üç sınıftan birini kullanmak isteyeceksiniz:

wpilib.Diferansiyel sürücü için DifferentialDrive/skid-steer driveplatformları gibi 2 veya 4 tekerlek platformları, Parçaları Kiti sürücü tabanı, "tank drive" ” veya West Coast drive.

wpilib.Killough Üçgen tahrik platformları için KilloughDrive.

wpilib.Mecanum sürücü platformları için MecanumDrive.

Örneğin, oluşturduğunuzda bir DifferentialDrive nesnesi, motor denetleyicisi örneklerinde geçirebilirsiniz:

```
l_motor = wpilib.Talon(0)
r_motor = wpilib.Talon(1)
self.robot_drive =
wpilib.drive.DifferentialDrive(l_motor, r_motor)
```

Veya yan başına birden fazla denetleyici kullanmak için motor kontrol gruplarında geçebilirsiniz:

```
self.frontLeft = wpilib.Spark(1)
self.rearLeft = wpilib.Spark(2)
self.left = wpilib.SpeedControllerGroup(self.frontLeft, self.rearLeft)
self.frontRight = wpilib.Spark(3)
self.rearRight = wpilib.Spark(4)
self.right = wpilib.SpeedControllerGroup(self.frontRight,
self.rearRight)
self.drive = wpilib.drive.DifferentialDrive(self.left, self.right)
```

Bu nesnelerden birine sahip olduğunuzda, robotu joystick üzerinden kontrol etmek için kullanabileceğiniz çeşitli yöntemler vardır veya kontrol girişlerini manuel olarak belirtebilirsiniz.

Robot Çalışma Modları (TimedRobot)

Bir yarışma sırasında, robot oyunun durumuna bağlı olarak çeşitli modlara geçer. Her mod sırasında robot sınıfınızdaki işlevler çağrılır. İşlevin adı, robotun hangi modda olduğuna bağlı olarak değişir:

disabledXXX-robot devre dışı bırakıldığında çağrılır
autonomousXXX-robot özerk modda olduğunda denir
teleopXXX-robot teleoperated modunda olduğunda denir
testXXX-robot test modunda olduğunda denir

Her modun onunla ilişkili iki işlevi vardır. xxx ın robot ilk önce moda geçtiğinde çağrılır ve xxxperiodic- saniyede 50 kez denir (yaklaşık olarak – aslında sürücü istasyonundan alınan paketler olarak adlandırılır).

Örneğin, robotu tek bir joystick kullanarak yönlendiren basit bir robot, şöyle görünen bir teleopperiodik fonksiyona sahip olabilir:

```
def teleopPeriodic(self):
    self.robot_drive.arcadeDrive(self.stick)
```

7

Ana blok

Java gibi diller, bir 'statik ana' işlevi tanımlamanızı gerektirir. Python'da, her. py dosyası diğer python programlarından kullanılabilir olduğundan, __main__için kontrol eden bir kod bloğu tanımlamanız gerekir. Ana blok içinde, aşağıdaki çağırma kullanarak robotun kodunu başlatmak için WPİLib söyle:

```
if __name__ == '__main__':
    wpilib.run(MyRobot)
```

Bu basit çağırma, robot kodunuzu robot üzerinde başlatmak için yeterlidir ve ayrıca pyfrc ve robotpy-frcsim gibi robot kodunuzu test etmek için mevcut olabilecek çeşitli RobotPy özellikli uzantılara erişim sağlar.

8

Hepsini bir araya getirmek:

Yukarıdaki tüm parçaları birleştirirseniz, github deposundaki örneklerden birinden alınan aşağıdaki gibi bir şeyle sonuçlanır:

Mevcut birkaç farklı python tabanlı robot örneği vardır ve bunları github örnekler havuzumuzda bulabilirsiniz.

Ayrıca bakınız:

RobotPy, robot kodunuzu oluşturmayı kolaylaştıran çeşitli çerçevelerle birlikte gelir. Robot kod çerçeveleri sayfasına bakın.

```
#!/usr/bin/env python3
"""
    This is a good foundation to build your robot code on
"""
import wpilib
import wpilib.drive

class MyRobot(wpilib.TimedRobot):
    def robotInit(self):
        """
        This function is called upon program startup and
        should be used for any initialization code.
        """
        self.left_motor = wpilib.Spark(0)
        self.right_motor = wpilib.Spark(1)
        self.drive = wpilib.drive.DifferentialDrive(self.left_motor, self.right_motor)
        self.stick = wpilib.Joystick(1)
        self.timer = wpilib.Timer()
    def autonomousInit(self):
        """This function is run once each time the robot enters autonomous mode."""
        self.timer.reset()
        self.timer.start()
    def autonomousPeriodic(self):
        """This function is called periodically during autonomous."""
        # Drive for two seconds
        if self.timer.get() < 2.0:
            self.drive.arcadeDrive(-0.5, 0)  # Drive forwards at half speed
        else:
            self.drive.arcadeDrive(0, 0)  # Stop robot
    def teleopPeriodic(self):
        """This function is called periodically during operator control."""
        self.drive.arcadeDrive(self.stick.getY(), self.stick.getX())

if __name__ == "__main__":
    wpilib.run(MyRobot)
```