com._604robotics.utils

Class SpringableVictor

java.lang.Object
  edu.wpi.first.wpilibj.SensorBase
    edu.wpi.first.wpilibj.PWM
      edu.wpi.first.wpilibj.SafePWM
        edu.wpi.first.wpilibj.Victor
          com._604robotics.utils.SpringableVictor

All Implemented Interfaces:

MotorSafety, IDevice, IDeviceController, PIDOutput, SpeedController

---

```
public class SpringableVictor
extends Victor
```

Extender of a Victor providing an easier control flow. When an output is set for the Victor, it is considered "sprung". When the "reload" method is called, if the victor is sprung, it unsprings the Victor. If the Victor is not sprung, then the output is set to zero. In this way, the Victor will only be moving when you tell it to. Use this in a loop or something, and call "reload" at the end. No more worries about code paths that don't update the Victors!

Author:

Michael Smith

Nested Class Summary

Nested classes/interfaces inherited from class edu.wpi.first.wpilibj.PWM

PWM.PeriodMultiplier

Field Summary

Fields inherited from class edu.wpi.first.wpilibj.PWM

kDefaultMinPwmHigh, kDefaultPwmPeriod, kPwmDisabled

Fields inherited from class edu.wpi.first.wpilibj.SensorBase

kAnalogChannels, kAnalogModules, kDigitalChannels, kPwmChannels, kRelayChannels, kSolenoidChannels, kSolenoidModules, kSystemClockTicksPerMicrosecond

Fields inherited from interface edu.wpi.first.wpilibj.MotorSafety

DEFAULT_SAFETY_EXPIRATION

Constructor Summary

Constructors

Constructor and Description
SpringableVictor(int port) Initializes a new SpringableVictor on the given PWM port.
SpringableVictor(int slot, int port) Initializes a new SpringableVictor on the given module slot and PWM port.

Method Summary

Methods

Modifier and Type	Method and Description
boolean	getSprung()

boolean	**getSprung**() Has the victor been sprung?
void	**pidWrite**(double output) Function to hook into the PIDController.
void	**reload**() If the Victor has been sprung, unspring it; if not, set the output to 0.
void	**set**(double speed) Sets the power of the Victor.
void	**setController**(**PIDController** controller) Sets the PIDController for this Victor, if there is one.
void	**spring**() Springs the victor.

Methods inherited from class edu.wpi.first.wpilibj.Victor

get, set

Methods inherited from class edu.wpi.first.wpilibj.SafePWM

disable, Feed, getDescription, getExpiration, isAlive, isSafetyEnabled, setExpiration, setSafetyEnabled, stopMotor

Methods inherited from class edu.wpi.first.wpilibj.PWM

enableDeadbandElimination, free, getChannel, getModuleNumber, getPosition, getRaw, getSpeed, setBounds, setPeriodMultiplier, setPosition, setRaw

Methods inherited from class edu.wpi.first.wpilibj.SensorBase

checkAnalogChannel, checkAnalogModule, checkDigitalChannel, checkDigitalModule, checkPWMChannel, checkPWMModule, checkRelayChannel, checkRelayModule, checkSolenoidChannel, checkSolenoidModule, getDefaultAnalogModule, getDefaultDigitalModule, getDefaultSolenoidModule, setDefaultAnalogModule, setDefaultDigitalModule, setDefaultSolenoidModule

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface edu.wpi.first.wpilibj.SpeedController

disable

Constructor Detail

SpringableVictor

```
public SpringableVictor(int port)
```

Initializes a new SpringableVictor on the given PWM port.

Parameters:

`port` - The PWM port the Victor is connected to.

SpringableVictor

```
public SpringableVictor(int slot,
                        int port)
```

Initializes a new SpringableVictor on the given module slot and PWM port.

Parameters:

`slot` - The module slot the Victor is connected to.

`port` - The PWM port the Victor is connected to.

Method Detail

getSprung

getSprung

```
public boolean getSprung()
```

Has the victor been sprung?

Returns:

Whether or not the victor has been sprung.

spring

```
public void spring()
```

Springs the victor.

set

```
public void set(double speed)
```

Sets the power of the Victor.

Specified by:

`set` in interface `SpeedController`

Overrides:

`set` in class `Victor`

Parameters:

`speed` - The speed to set.

pidWrite

```
public void pidWrite(double output)
```

Function to hook into the PIDController. Sets the power of the Victors.

Specified by:

`pidWrite` in interface `PIDOutput`

Overrides:

`pidWrite` in class `Victor`

Parameters:

`output` - The speed to set.

reload

```
public void reload()
```

If the Victor has been sprung, unspring it; if not, set the output to 0.

setController

```
public void setController(PIDController controller)
```

Sets the PIDController for this Victor, if there is one. If the PIDController is enabled, reload will assume it's updating it, and won't reset the output to 0.

Parameters:

`controller` - The PIDController for this Victor.