JavaScript is disabled on your browser.

[Skip navigation links](#1fob9te)

* [Overview](http://docs.google.com/overview-summary.html)
* [Package](http://docs.google.com/com/kauailabs/navx/ftc/package-summary.html)
* Class
* [Tree](http://docs.google.com/package-tree.html)
* [Deprecated](http://docs.google.com/deprecated-list.html)
* [Index](http://docs.google.com/index-all.html)
* [Help](http://docs.google.com/help-doc.html)
* [Prev Class](http://docs.google.com/com/kauailabs/navx/ftc/navXPerformanceMonitor.html)
* [Next Class](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.navXTimestampedDataSource.html)
* [Frames](http://docs.google.com/index.html?com/kauailabs/navx/ftc/navXPIDController.html)
* [No Frames](http://docs.google.com/navXPIDController.html)
* [All Classes](http://docs.google.com/allclasses-noframe.html)
* Summary:
* [Nested](#3znysh7) |
* Field |
* [Constr](#2et92p0) |
* [Method](#tyjcwt)
* Detail:
* Field |
* [Constr](#1t3h5sf) |
* [Method](#17dp8vu)

com.kauailabs.navx.ftc

## Class navXPIDController

* java.lang.Object
  + com.kauailabs.navx.ftc.navXPIDController
* All Implemented Interfaces: [IDataArrivalSubscriber](http://docs.google.com/com/kauailabs/navx/ftc/IDataArrivalSubscriber.html)  
    
  public class navXPIDController  
  extends java.lang.Object  
  implements [IDataArrivalSubscriber](http://docs.google.com/com/kauailabs/navx/ftc/IDataArrivalSubscriber.html)  
  The navXPIDController implements a timestamped PID controller (designed to deal with the jitter which is typically present in a networked control system scenario).  
  The navXPIDController can use any of the various data sources on a navX-Model device as an input (process variable); when instantiating a navXPIDController simply provide an AHRS class instance and specify which navX-Model device variable you wish to use as the input. Then, configure the navXPIDController's setPoint, outputRange, whether it should operate in continuous mode or not, and the P, I, D and F coefficients which will be used to calculate the output value.  
  An example of using the navXPIDController to rotate a FTC robot to a target angle is provided [online](http://navx-micro.kauailabs.com/examples/rotate-to-angle/).  
  The general PID algorithm used herein is [discussed in detail on Wikipedia.](https://en.wikipedia.org/wiki/PID_controller)  
  In addition to the P,I,D terms, a FeedForward term is optionally available which may be useful in cases where velocity is being controlled (e.g., to achieve a continuous rotational velocity using a yaw rate gyro). The FeedForward concept is discussed further [here.](http://www.expertune.com/articles/UG2007/PIDControlsPLCEnviron.pdf)  
  This algorithm implements two features with respect to the integral gain calculated based on the integral (i) coefficient:  
  - Anti-Windup: Ensures the integral gain doesn't exceed the min/max output range, as discussed [here.](http://www.expertune.com/articles/UG2007/PIDControlsPLCEnviron.pdf) - Time-Correction: Adjust the integral gain in cases when timestamps indicate that data samples were lost.  
  This algorithm implements this feature with respect to the derivative gain, as discussed [here.](http://www.diva-portal.org/smash/get/diva2:570067/FULLTEXT01.pdf)

### Nested Class SummaryNested Classes

|  |  |
| --- | --- |
| * + Modifier and Type | * + Class and Description |
| * + static class | * + [navXPIDController.navXTimestampedDataSource](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.navXTimestampedDataSource.html) The navXTimestampedDataSources specifies the navX-Model device sensor data source type used by the navXPIDController as it's input data source. |
| * + static class | * + [navXPIDController.navXUntimestampedDataSource](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.navXUntimestampedDataSource.html) The navXUntimestampedDataSources specifies the navX-Model device sensor data source type used by the navXPIDController as it's input data source. |
| * + static class | * + [navXPIDController.PIDResult](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.PIDResult.html) The PIDResult class encapsulates the data used by the navXPIDController to communicate current state to a client of the navXPIDController. |
| * + static class | * + [navXPIDController.TimestampType](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.TimestampType.html) |
| * + static class | * + [navXPIDController.ToleranceType](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.ToleranceType.html) The ToleranceType enumeration defines the type of tolerance to be used by the navXPIDController to determine whether the controller is "on\_target". |

### Constructor SummaryConstructors

|  |
| --- |
| * + Constructor and Description |
| * + [navXPIDController](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#navXPIDController-com.kauailabs.navx.ftc.AHRS-com.kauailabs.navx.ftc.navXPIDController.navXTimestampedDataSource-)([AHRS](http://docs.google.com/com/kauailabs/navx/ftc/AHRS.html) navx\_device, [navXPIDController.navXTimestampedDataSource](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.navXTimestampedDataSource.html) src) This navXPIDController constructor is used when the PID Controller is to be driven by a navX-Model device input data source which is accompanied by a high-accuracy "sensor" timestamp. |
| * + [navXPIDController](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#navXPIDController-com.kauailabs.navx.ftc.AHRS-com.kauailabs.navx.ftc.navXPIDController.navXUntimestampedDataSource-)([AHRS](http://docs.google.com/com/kauailabs/navx/ftc/AHRS.html) navx\_device, [navXPIDController.navXUntimestampedDataSource](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.navXUntimestampedDataSource.html) src) This navXPIDController constructor is used when the PID Controller is to be driven by a navX-Model device input data source which is not accompanied by a high-accuracy "sensor" timestamp. |

### Method SummaryAll Methods Instance Methods Concrete Methods

|  |  |
| --- | --- |
| * + Modifier and Type | * + Method and Description |
| * + void | * + [close](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#close--)() |
| * + void | * + [enable](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#enable-boolean-)(boolean enabled) Enables/disables the PID controller. |
| * + double | * + [get](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#get--)() Returns the current output value calculated by the navXPIDController based upon the last navX-Model device data sample received. |
| * + double | * + [getError](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#getError--)() Returns the current amount of error calculated by the navXPIDController based upon the last navX-Model device data sample. |
| * + double | * + [getSetpoint](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#getSetpoint--)() Returns the currently configured setpoint. |
| * + boolean | * + [isEnabled](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#isEnabled--)() Returns true if the navXPIDController is currently enabled, otherwise return false. |
| * + boolean | * + [isNewUpdateAvailable](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#isNewUpdateAvailable-com.kauailabs.navx.ftc.navXPIDController.PIDResult-)([navXPIDController.PIDResult](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.PIDResult.html) result) isNewUpdateAvailable() should be called by clients of the navXPIDController which need to "poll" to determine whether new navX-Model device data has been received. |
| * + boolean | * + [isOnTarget](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#isOnTarget--)() Indicates whether the navXPIDController has determined that it is "on\_target" based upon whether the current error is within the range defined by the tolerance. |
| * + void | * + [reset](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#reset--)() Resets the PIDController's current error value as well as the integrated error. |
| * + void | * + [setContinuous](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#setContinuous-boolean-)(boolean continuous) setContinuous() is used to enable/disable the continuous mode of operation. |
| * + void | * + [setInputRange](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#setInputRange-double-double-)(double min\_input, double max\_input) Defines the range of possible input values received from the currently-selected navX-Model device data source. |
| * + void | * + [setOutputRange](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#setOutputRange-double-double-)(double min\_output, double max\_output) Defines the range of output values calculated by the navXPIDController. |
| * + void | * + [setPID](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#setPID-double-double-double-)(double p, double i, double d) setPID() is used to set the Proportional, Integral and Differential coefficients used by the navXPIDController. |
| * + void | * + [setPID](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#setPID-double-double-double-double-)(double p, double i, double d, double ff) setPID() is used to set the Proportional, Integral, Differential and FeedForward coefficients used by the navXPIDController. |
| * + void | * + [setSetpoint](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#setSetpoint-double-)(double setpoint) Defines the "target" value the navXPIDController attempts to reach. |
| * + void | * + [setTolerance](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#setTolerance-com.kauailabs.navx.ftc.navXPIDController.ToleranceType-double-)([navXPIDController.ToleranceType](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.ToleranceType.html) tolerance\_type, double tolerance\_amount) Used to specify the tolerance used to determine whether the navXPIDController is "on\_target". |
| * + double | * + [stepController](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#stepController-double-int-)(double process\_variable, int num\_missed\_samples) stepController() is the PIDController worker function, which is used internally by the navXPIDController whenever a new navX-Model device sensor data value is received. |
| * + void | * + [timestampedDataReceived](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#timestampedDataReceived-long-long-java.lang.Object-)(long curr\_system\_timestamp, long curr\_sensor\_timestamp, java.lang.Object kind) |
| * + void | * + [untimestampedDataReceived](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#untimestampedDataReceived-long-java.lang.Object-)(long curr\_system\_timestamp, java.lang.Object kind) |
| * + boolean | * + [waitForNewUpdate](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#waitForNewUpdate-com.kauailabs.navx.ftc.navXPIDController.PIDResult-int-)([navXPIDController.PIDResult](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.PIDResult.html) result, int timeout\_ms) waitForNewUpdate() should be called by clients of the navXPIDController which want to "wait" until new navX-Model device data has been received. |
| * + void | * + [yawReset](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.html#yawReset--)() |

### Methods inherited from class java.lang.Objectclone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

### Constructor Detail

#### navXPIDController public navXPIDController([AHRS](http://docs.google.com/com/kauailabs/navx/ftc/AHRS.html) navx\_device, [navXPIDController.navXTimestampedDataSource](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.navXTimestampedDataSource.html) src) This navXPIDController constructor is used when the PID Controller is to be driven by a navX-Model device input data source which is accompanied by a high-accuracy "sensor" timestamp. The data source specified automatically determines the navXPIDController's input data range.

#### navXPIDController public navXPIDController([AHRS](http://docs.google.com/com/kauailabs/navx/ftc/AHRS.html) navx\_device, [navXPIDController.navXUntimestampedDataSource](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.navXUntimestampedDataSource.html) src) This navXPIDController constructor is used when the PID Controller is to be driven by a navX-Model device input data source which is not accompanied by a high-accuracy "sensor" timestamp. The data source specified automatically determines the navXPIDController's input data range.

### Method Detail

#### untimestampedDataReceived public void untimestampedDataReceived(long curr\_system\_timestamp, java.lang.Object kind)Specified by: [untimestampedDataReceived](http://docs.google.com/com/kauailabs/navx/ftc/IDataArrivalSubscriber.html#untimestampedDataReceived-long-java.lang.Object-) in interface [IDataArrivalSubscriber](http://docs.google.com/com/kauailabs/navx/ftc/IDataArrivalSubscriber.html)

#### timestampedDataReceived public void timestampedDataReceived(long curr\_system\_timestamp, long curr\_sensor\_timestamp, java.lang.Object kind)Specified by: [timestampedDataReceived](http://docs.google.com/com/kauailabs/navx/ftc/IDataArrivalSubscriber.html#timestampedDataReceived-long-long-java.lang.Object-) in interface [IDataArrivalSubscriber](http://docs.google.com/com/kauailabs/navx/ftc/IDataArrivalSubscriber.html)

#### yawReset public void yawReset()Specified by: [yawReset](http://docs.google.com/com/kauailabs/navx/ftc/IDataArrivalSubscriber.html#yawReset--) in interface [IDataArrivalSubscriber](http://docs.google.com/com/kauailabs/navx/ftc/IDataArrivalSubscriber.html)

#### close public void close()

#### isNewUpdateAvailable public boolean isNewUpdateAvailable([navXPIDController.PIDResult](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.PIDResult.html) result) isNewUpdateAvailable() should be called by clients of the navXPIDController which need to "poll" to determine whether new navX-Model device data has been received. Whether or not new data has been received, this method returns immediately and does not block. Returns: Returns true if new data has been received since the last time this function was called, otherwise returns false. If true, the result will updated to reflect the newly-calculated controller values.

#### waitForNewUpdate public boolean waitForNewUpdate([navXPIDController.PIDResult](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.PIDResult.html) result, int timeout\_ms) throws java.lang.InterruptedException waitForNewUpdate() should be called by clients of the navXPIDController which want to "wait" until new navX-Model device data has been received. This method will return immediately only if new data has been received since the last time it was called; otherwise, it will block and not return until new data has been received, or a specified timeout period has passed. Returns: Returns true when new data has been received. If false is returned, this indicates a timeout has occurred while waiting for new data. Throws: java.lang.InterruptedException

#### getError public double getError() Returns the current amount of error calculated by the navXPIDController based upon the last navX-Model device data sample. By definition, this error is equal to the set point minus the last device data sample.

#### setTolerance public void setTolerance([navXPIDController.ToleranceType](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.ToleranceType.html) tolerance\_type, double tolerance\_amount) Used to specify the tolerance used to determine whether the navXPIDController is "on\_target". The tolerance threshold is defined by the tolerance\_type and the tolerance amount. For example, if the YAW data source is used and thus the values used are in units of degrees, ToleranceType.ABSOLUTE and tolerance\_amount of 2.0 would result in the navXPIDController deciding it was "on\_target" whenever the error was within a range of +/- 2 degrees.

#### isOnTarget public boolean isOnTarget() Indicates whether the navXPIDController has determined that it is "on\_target" based upon whether the current error is within the range defined by the tolerance.

#### stepController public double stepController(double process\_variable, int num\_missed\_samples) stepController() is the PIDController worker function, which is used internally by the navXPIDController whenever a new navX-Model device sensor data value is received.

#### setPID public void setPID(double p, double i, double d) setPID() is used to set the Proportional, Integral and Differential coefficients used by the navXPIDController.

#### setPID public void setPID(double p, double i, double d, double ff) setPID() is used to set the Proportional, Integral, Differential and FeedForward coefficients used by the navXPIDController.

#### setContinuous public void setContinuous(boolean continuous) setContinuous() is used to enable/disable the continuous mode of operation. When continuous mode is disabled, the min/max input range values are used as two separate points at the ends of the range of possible input values. This mode of operation is typically used for reaching a position within a linear range. When continuous mode is enabled, the min/max input range are considered to represent the sampe point. This mode of operation is typically used for reaching a position within a circular range, and allows the navXPIDController to determine the shortest possible route to the setpoint. For example, when using YAW as the input data source, and using the PID controller to rotatie to a given angle, if the setpoint is 150 degrees and the current input value is -150 degrees, the controller will calculate an output such that it travels across the boundary between -180 and 180 degrees (for a total traveled distance of 60 degrees), rather than traveling 300 degrees as it would if continuous mode were disabled.

#### get public double get() Returns the current output value calculated by the navXPIDController based upon the last navX-Model device data sample received.

#### setOutputRange public void setOutputRange(double min\_output, double max\_output) Defines the range of output values calculated by the navXPIDController. For example, when the navXPIDController is used to calculate an output value to be sent to a motor controller whose valid range is -1 to 1, the output range should be sent to -1, 1. Note that the units of the output range are not necessarily the same as the units of the input range.

#### setInputRange public void setInputRange(double min\_input, double max\_input) Defines the range of possible input values received from the currently-selected navX-Model device data source. For example, if YAW is the data source, the input range would be -180.0, 180.0. Note that the navXPIDController constructor automatically sets the input range based upon the data source specified to the constructor.

#### setSetpoint public void setSetpoint(double setpoint) Defines the "target" value the navXPIDController attempts to reach. This value is in the same units as the input, and should be within the input range. For example, if YAW is the data source, the setput should be between -180.0 and 180.0.

#### getSetpoint public double getSetpoint() Returns the currently configured setpoint.

#### enable public void enable(boolean enabled) Enables/disables the PID controller. By default, the navXPIDController is disabled, thus this method must be invoked before attempting to use the navXPIDController's output values.

#### isEnabled public boolean isEnabled() Returns true if the navXPIDController is currently enabled, otherwise return false.

#### reset public void reset() Resets the PIDController's current error value as well as the integrated error. Also disables the controller. The enable() method must be used to re-enable the controller.

[Skip navigation links](#ihv636)

* [Overview](http://docs.google.com/overview-summary.html)
* [Package](http://docs.google.com/com/kauailabs/navx/ftc/package-summary.html)
* Class
* [Tree](http://docs.google.com/package-tree.html)
* [Deprecated](http://docs.google.com/deprecated-list.html)
* [Index](http://docs.google.com/index-all.html)
* [Help](http://docs.google.com/help-doc.html)
* [Prev Class](http://docs.google.com/com/kauailabs/navx/ftc/navXPerformanceMonitor.html)
* [Next Class](http://docs.google.com/com/kauailabs/navx/ftc/navXPIDController.navXTimestampedDataSource.html)
* [Frames](http://docs.google.com/index.html?com/kauailabs/navx/ftc/navXPIDController.html)
* [No Frames](http://docs.google.com/navXPIDController.html)
* [All Classes](http://docs.google.com/allclasses-noframe.html)
* Summary:
* [Nested](#3znysh7) |
* Field |
* [Constr](#2et92p0) |
* [Method](#tyjcwt)
* Detail:
* Field |
* [Constr](#1t3h5sf) |
* [Method](#17dp8vu)