**Purpose:** Provide a standard way to determine the direction the robot is facing. This could use a gyroscope, compass, another sensor, or a combination of sensors. Regardless of how it works internally, the main method to get the heading should return a value in a predetermined range and type (like a float/double in the range of 0-359 degrees).

|  |  |
| --- | --- |
| **Priority:** Medium-Low | **Reason:** We can do navigation without a heading (using encoders, timing, etc.), but tracking direction will improve accuracy, flexibility, and ability to do course correction. |

**Programmers (first is the primary programmer):**

1. Adam

**Public Constants (public static final):**

* <required type if any> <NAME>\*
  + Purpose: <1-4 sentences> Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque in turpis posuere, pretium velit in, tempor turpis.
* int MAX\_MOTOR\_POWER
  + Purpose: The maximum value for motor power. Should be used in algorithms like motor scaling and course correction.

**Constructors (called when an object instance is created):**

* (<#> arguments)
  + Must Initialize: <Minimum initialization it must preform>
  + Arguments:
    - <required type if applicable> <arg1>
      * <purpose>

**Interface Instance Methods (used on an instance of this class):**

* public <other tags> <return type> <method name>(<#> arguments)
  + Purpose: <1-4 sentences> Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque in turpis posuere, pretium velit in, tempor turpis.
  + Priority: <Same options as the class priority> (<reason>)
  + Arguments:
    - <type> <arg1>
      * What to put there (see also: <something else in this file>)
  + Returns: (<type>) <what data is returned>
    - <Can go onto a separate bullet(s) if it would be cleaner>
* public synchronized void setPower(1 argument)
  + Purpose: Adjust the motor power, and start or stop it when necessary.
  + Priority: Very High (cannot start the motor without it)
  + Arguments:
    - int power
      * The power to set the motor to (see also: MAX\_MOTOR\_POWER)
  + Returns: N/A
* public synchronized void stopMotor(0 arguments)
  + Purpose: Stop the motor, syntactically equivalent to ‘setPower(0)’.
  + Priority: Medium (only a shortcut method, but all it needs to do is call setPower(0))
  + Arguments: N/A
  + Returns: N/A

**Interface Static Methods (used without an instance of a class):**

* public static synchronized long getCurrentTime(1 argument)
  + Purpose: Get how much time has elapsed, syntactically equivalent to ‘System.currentTimeMillis() - someothertime’
  + Priority: Low (prevents other programmers from having to do this manually every time)
  + Arguments:
    - long start
      * when this ‘timer’ started (specify 0 to get an initial start time)
  + Returns: (long) the current system time in milliseconds

\*Side note for the asterisk goes here.