**Constructors**

**Method Signature:** public Finch()  
**Description:** Constructor that creates an object corresponding to a Finch. This method assumes that the Finch is the only device connected in the BlueBird Connector.  
**Example:** Finch bird = new Finch();

**Method Signature:** public Finch(device)  
**Description:** Constructor that creates an object corresponding to a Finch. This method requires a string equal to “A”, “B”, or “C” that specifies the letter of the device in the BlueBird Connector.  
**Example:** Finch bird = new Finch("A");

**Output Methods**

**Method Signature:** public void setMove(String direction, double distance, double speed)  
**Description:** Moves the Finch forward or backward for a specified distance at a specified speed. The method requires a direction (“F” for forward or “B” for backward), a distance in centimeters, and a speed from 0-100.  
**Example:** bird.setMove("F",10,50);

**Method Signature:** public void setTurn(String direction, double angle, double speed)  
**Description:** Turns the Finch right or left for a specified angle at a specified speed. The method requires a direction (“R” for right or “L” for left), an angle in degrees, and a speed from 0-100.  
**Example:** bird.setTurn("R",90,50);

**Method Signature:** setMotors(double leftSpeed, double rightSpeed)  
**Description:** Sets the Finch wheels to spin at the given speeds. The method requires two speeds between -100 and 100 for the left and right wheels. Setting the speed to 0 turns the motor off.  
**Example:** bird.setMotors(-50,50);

**Method Signature:** public void stop()  
**Description:** Stops the Finch wheels.  
**Example:** bird.stop();

**Method Signature:** public void setBeak(int red, int green, int blue)  
**Description:** Sets a tri-color LED in the Finch beak to a given color by setting the intensities of the red, green, and blue elements inside it. The method requires three intensity values from 0-100. Setting all three intensity values to 0 turns the beak off.  
**Example:** bird.setBeak(0,100,0);

**Method Signature:** public void setTail(int ledNum, int red, int green, int blue)  
**Description:** Sets a tri-color LED in the Finch tail to a given color by setting the intensities of the red, green, and blue elements inside it. The method requires the port number of the LED (1-4) and three intensity values from 0-100. Setting all three intensity values to 0 turns the LED off.  
**Example:** bird.setTail(1,0,100,0);

**Method Signature:** public void setTail(String ledNum, int red, int green, int blue)  
**Description:** Sets all the tri-color LEDs in the Finch tail to a given color by setting the intensities of the red, green, and blue elements inside it. The method requires a String equal to “all” and three intensity values from 0-100. Setting all three intensity values to 0 turns the LED off.  
**Example:** bird.setTail("all",0,100,0);

**Method Signature:** public void playNote(int note, double beats)  
**Description:** Plays a note using the buzzer on the Finch. The method requires an integer representing the note (32-135) and a number giving the number of beats (0-16). The number of beats can be a decimal number.  
**Example:** bird.playNote(60,0.5);

**Method Signature:** public void setDisplay(int[] ledValues)  
**Description:** Sets the LED array of the micro:bit to display a pattern defined by an array. Each value in the array must be 0 (off) or 1 (on). The first five values in the array correspond to the five LEDs in the first row, the next five values to the second row, etc.  
**Example:** int patternArray[] = {0,0,0,0,0,1,1,1,1,1,0,0,1,0,0,1,1,1,1,1,0,0,0,0,0};  
bird.setDisplay(patternArray);

**Method Signature:** public void setPoint(int row, int column, int value)  
**Description:** Turn on or off a single LED on the micro:bit display. The position of the LED is given by the row and column parameters, which should both be between 1 and 5. The value of the LED must be 0 (off) or 1 (on).  
**Example:** bird.setPoint(3,3,1);

**Method Signature:** public void print(String message)  
**Description:** Print a string on the micro:bit LED array. The string must have 15 or fewer characters and should contain only digits and English letters (upper or lower case).  
**Example:** bird.print("hello");

**Method Signature:** public void stopAll()  
**Description:** Stops all outputs. This includes the LED display for the micro:bit and all lights and motors for the Finch.  
**Example:** bird.stopAll();

**Input Methods**

**Method Signature:** public int getDistance()  
**Description:** Returns the distance in centimeters to the closest object.  
**Example:** System.out.println("Distance to obstacle: " + bird.getDistance());

**Method Signature:** public int getLight(String direction)  
**Description:** Takes a string corresponding to a direction (“R” for right or “L” for left) and returns the intensity measured by the corresponding Finch light sensor. The light measurement is an integer between 0 and 100 (arbitrary units).  
**Example:** System.out.println("Left light sensor: "+ bird.getLight("L"));

**Method Signature:** public int getLine(String direction)  
**Description:** Takes a string corresponding to a direction (“R” for right or “L” for left) and returns the intensity measured by the corresponding Finch line tracking sensor. The line measurement is an integer between 0 and 100 (arbitrary units).  
**Example:** System.out.println("Right line sensor: "+ bird.getLine("R"));

**Method Signature:** public void resetEncoders()  
**Description:** Sets the value of both Finch encoders to 0.  
**Example:** bird.resetEncoders();

**Method Signature:** public double getEncoder(String direction)  
**Description:** Takes a string corresponding to a direction (“R” for right or “L” for left) and returns the value measured by the corresponding Finch encoder. The encoder measurement is the number of rotations that the wheel has turned since resetEncoders() was last called.  
**Example:** System.out.println("Right wheel rotations: "+ bird.getEncoder("R"));

**Method Signature:** public boolean getButton(String button)  
**Description:** Takes a string corresponding to a micro:bit button (“A” or “B”) and returns a boolean value that is true if the button is being pressed, and false otherwise.  
**Example:** System.out.println("Button A is pressed: " + bird.getButton("A"));

**Method Signature:** public boolean isShaking()  
**Description:** Returns true if the Finch is shaking and false otherwise.  
**Example:** System.out.println("Shake Status: " + bird.isShaking());

**Method Signature:** public String getOrientation()  
**Description:** Returns a string that represents the orientation of the Finch. The possible values are “Beak up”, “Beak down”, “Tilt left”, “Tilt right”, “Level”, “Upside down”, and “In between”.  
**Example:** System.out.println("Orientation: " + bird.getOrientation());

**Method Signature:** public double[] getAcceleration()  
**Description:** Returns an array that contains the Finch acceleration in Returns a list that contains the Finch acceleration in m/s2 in the *x*, *y*, and *z* directions.  
**Example:** System.out.println("Acceleration in x-direction: " + bird.getAcceleration()[0]);

**Method Signature:** public int getCompass()  
**Description:** Returns the direction of the Finch in degrees from magnetic north (0°-359°). The compass should be calibrated in the BlueBird Connector before using this method.  
**Example:** System.out.println("Compass Heading: " + bird.getCompass());

**Method Signature:** public int[] getMagnetometer()  
**Description:** Returns an array that contains the value of the magnetic field in µT in the *x*, *y*, and *z* directions. The compass should be calibrated in the BlueBird Connector before using this method.  
**Example:** System.out.println("Magnetic Field in x-direction: " + bird.getMagnetometer()[0]);

**Other**

**Method Signature:** public void pause(double numSeconds)  
**Description:** Pauses the program for a number of seconds.  
**Example:** bird.pause(1.5);

**Method Signature:** public void disconnect()  
**Description:** Closes the http connection to the Finch. This method should be called at the end of each program.  
**Example:** bird.disconnect();