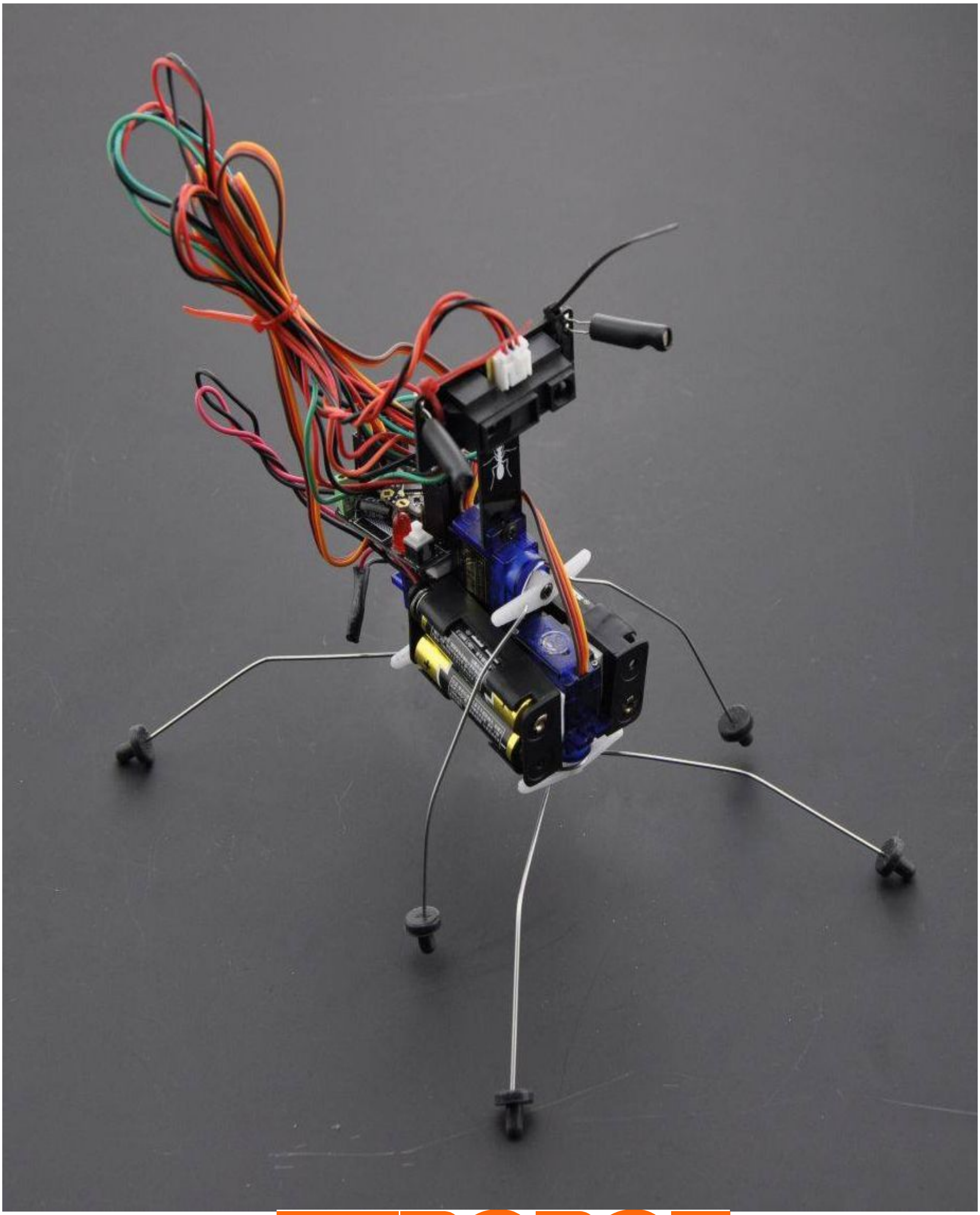
Insectbot Hexa – Fun, technology and education



DFROBOT
DRIVE THE FUTURE

Welcome to the Insectbot Hexa assembling manual. Here you will learn how to put all the parts together and how to program the robot.

NOTE:
If you got the Insectbot kit with already soldered Beetle/Beetle Shield and the light sensors/sensor holder then you can skip chapter 1 and the soldering parts of chapter 5 and 6

The Insectbot Hexa is not a toy but a way to learn how things work together. By assembling the Insectbot Hexa you will learn about some mechanical and electronic parts as well the procedures how to connect all the parts.

The new version V2 uses Bluno beetle instead of Beetle board, so the pictures in the passage may be not same as what is in your hand.

IMPORTANT:
Please pay attention to the right polarity when connecting cables and the batteries to the robot. Wrong connections could lead top permanent damage of parts and may cause a fire.

Work careful when handling the stainless steel wire. Make sure the wire will not bounce to prevent injuries of you or any bystanders. Wear protective leather gloves to prevent injuries by puncturing your fingers when bending the wires legs, attaching them to the servo arm and putting the rubber feet on them.

TIP:
To make the Insectbot Hexa walking properly you need to make sure that the legs are bent in the right angles and shapes. There is no example which angle and shape will work best since every robot is build by hand and may have a different center of weight. Try and re-try to bend the legs in different shapes and angles if the walking pattern does not satisfy you.

By changing the code for the light sensors you can make the robot either chasing light or avoiding light.

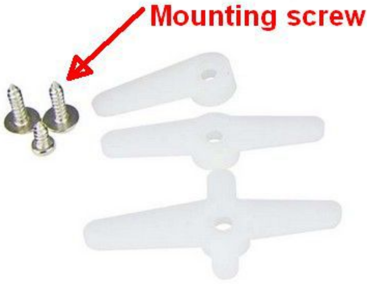
DISCLAIMER:
This manual is provided as it is. You may face some difficulties by assembling the robot but that's part of the plan. That Insectbot Hexa Kit is meant for education, so people learn how to solve minor problems. However, if you cannot solve your problem then please do not hesitate to contact us for help.

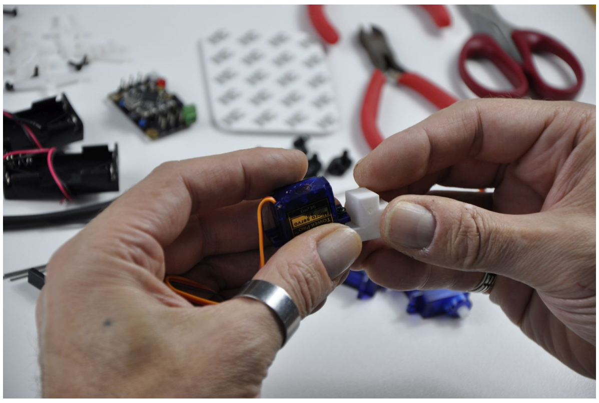
May 2015
Author: Lutz Michaelis, Robotic Engineer at DFRobot, Modified by Leff
All content including pictures are copyright by DFRobot.com

Insect bot Hexa assembling instructions

1. Assembling the body

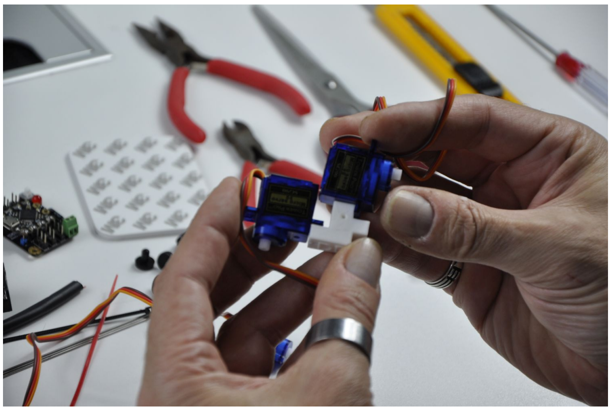
Mount all three servos to the center piece in the order as shown in the images using the provided screws. Make sure the cables are showing right direction and your servos are aligned and firmly attached so they do not get loose when the robot is walking.

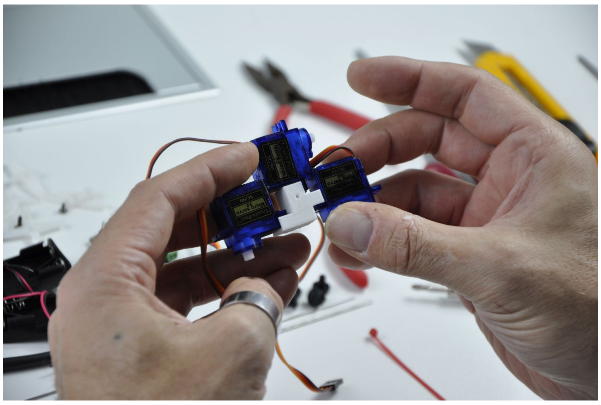




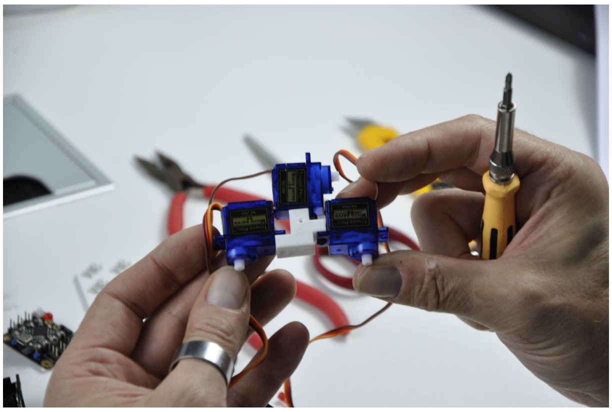
Servo 1: Rear servo

Servo 2: Middle Servo



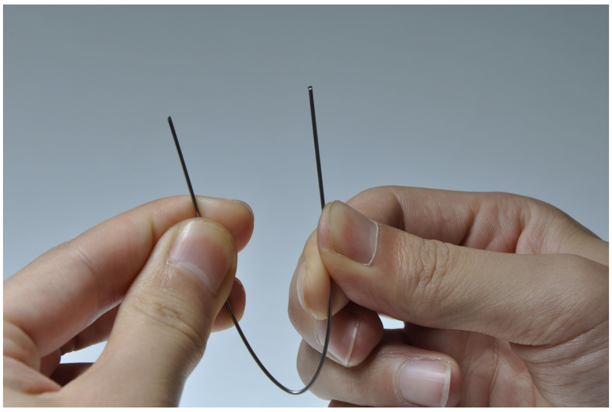


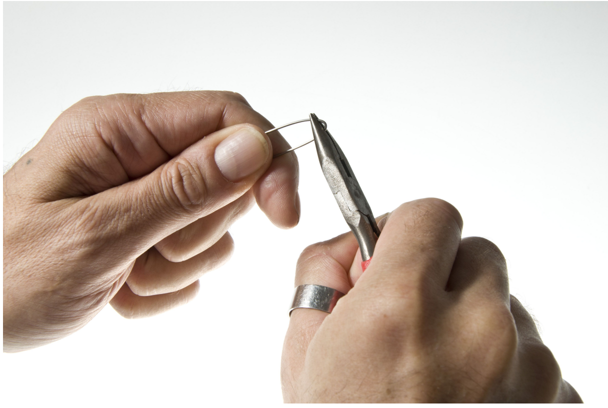
Servo 3: Front servo

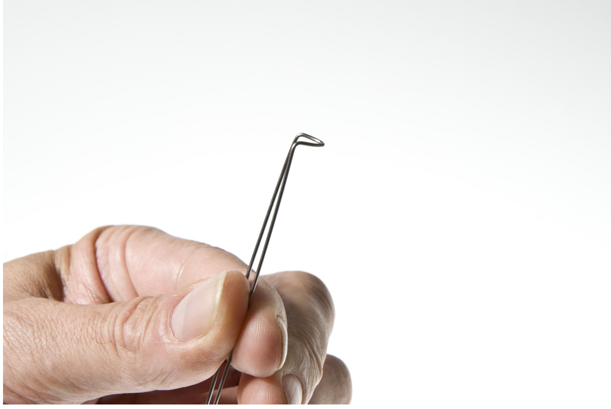


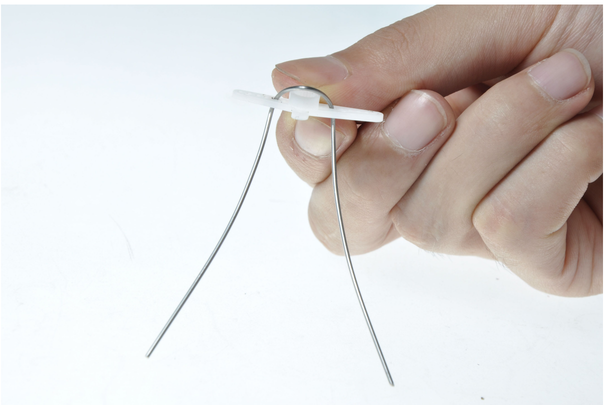
2. Bending the legs

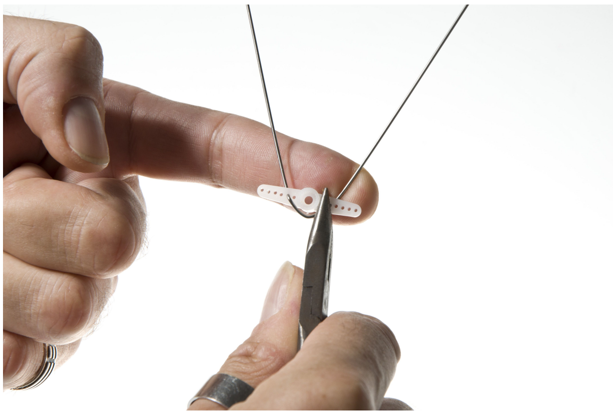
Bending the legs requires some strength or small pliers in order to bend them. The front and the rear legs are bend in a V shape and the middle leg in an U shape.
The middle part of the legs needs to be bend again to make sure the legs get not jammed when the servo is moving. See the picture for the shape.

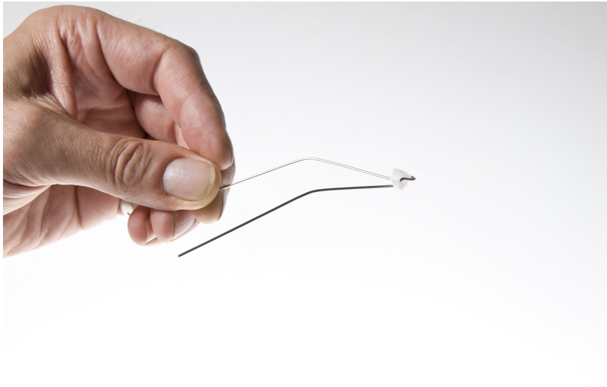


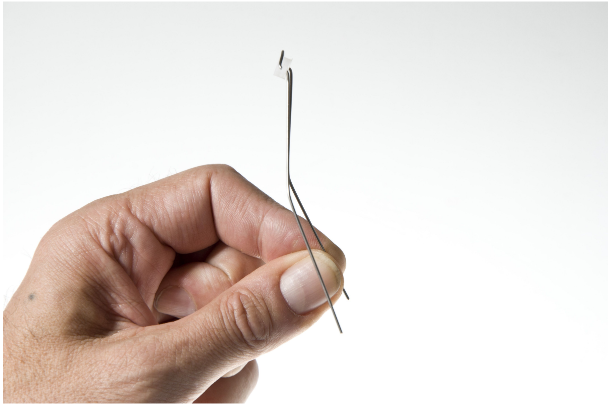










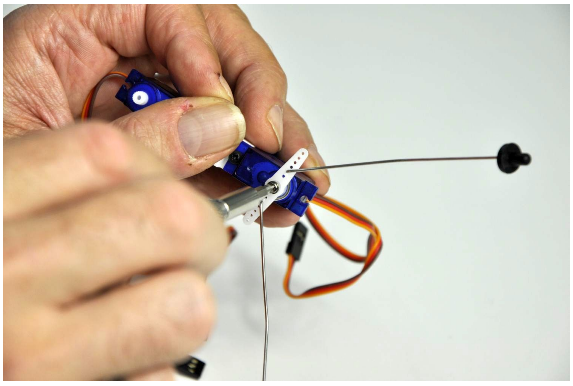


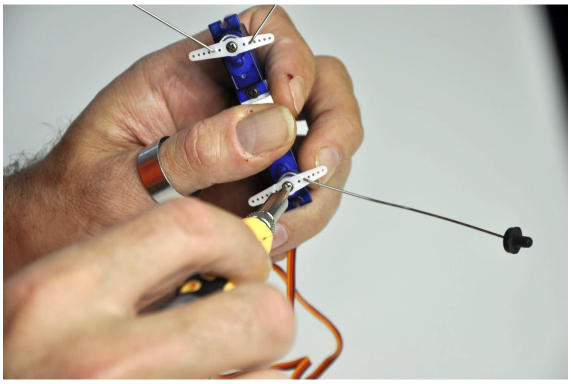
3. Attaching the legs to the servo

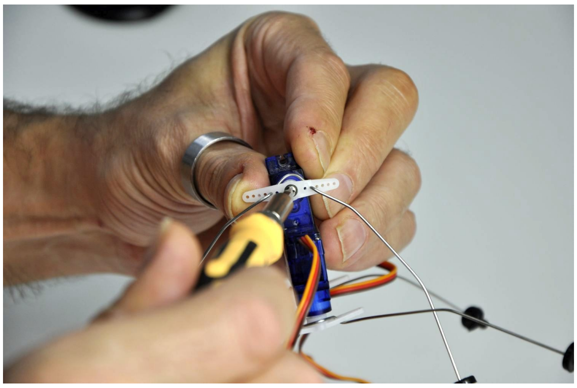
Before you attach the legs to the servos please make sure your servos are centered. To do this, please connect all three servos to the digital pins D3, D4 and D5 (see more details below) of the Beetle shield and upload the following program to the Beetle. The file name is **insect_bot_servo_center.ino** and available on the product/Wiki website at DFRobot.com

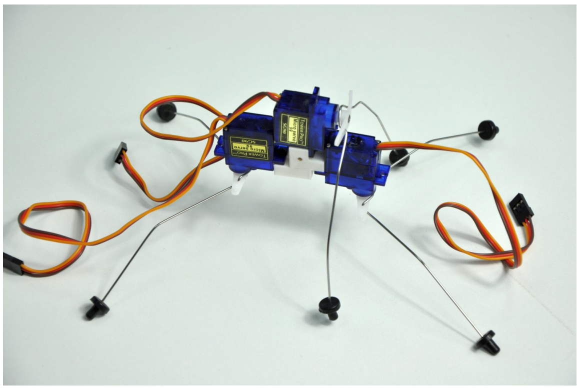
```
#include <Servo.h>
// creating the servo objects for front, rear and mid servo
Servo frontLeg;
Servo rearLeg;
Servo midLeg;
// setting the servo angle to 90° for startup
byte frontAngle = 90;
byte rearAngle = 90;
byte midAngle = 90;
// Setup function
void setup(){
  frontLeg.attach(3);
  rearLeg.attach(4);
  midLeg.attach(5);
  // move servos to center position -> 90°
  frontLeg.write(frontAngle);
  rearLeg.write(rearAngle);
  midLeg.write(midAngle);
  delay(2000);
}
// The loop remains empty
void loop(){
}
```

Put the legs with the servo horn on the servo shaft and secure them with the provided screw.

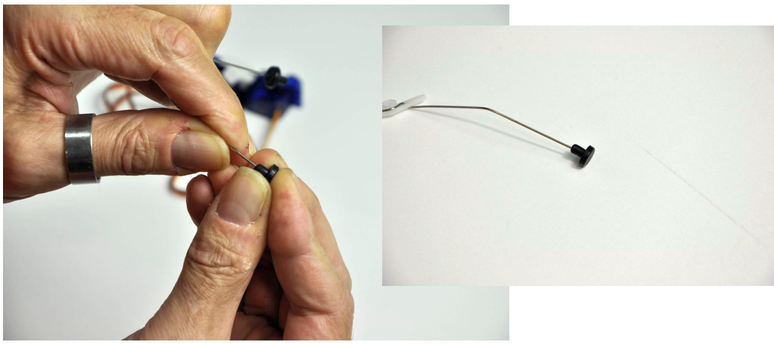








Once the legs are attached to the servo put the supplied rubber caps on the end of the wire legs. These rubber caps will prevent the legs slipping and may also protect your furniture from scratches.

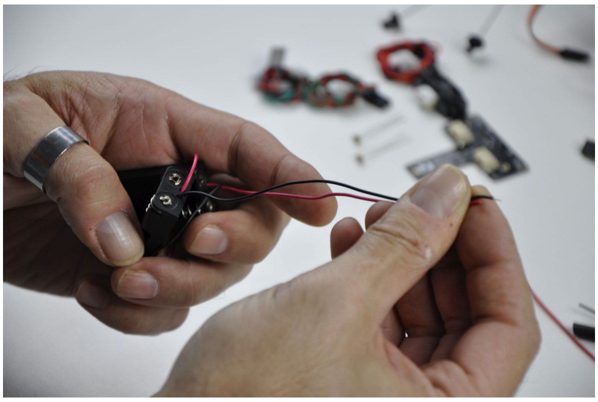


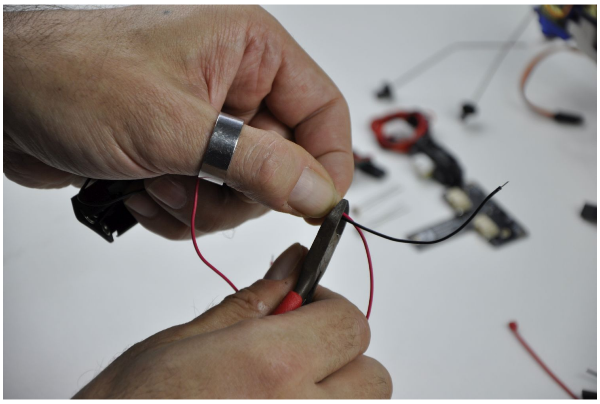
You may put the rubber feet in either way on the legs. In this detail picture we put them in the opposite direction than in the other pictures. Choose whatever you find more functional or better looking.

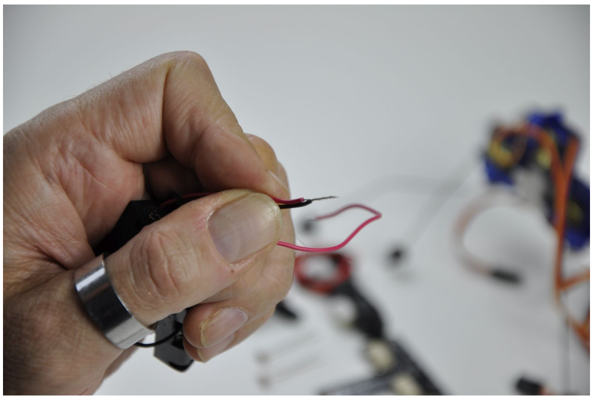
4. Attaching the battery holders

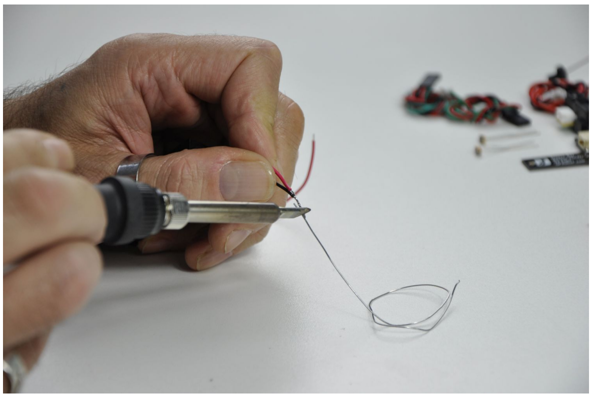
Before you attach the battery holders to the main robot body you will need to solder their power wires in series connection together. Connect the red wire of one battery holder with the black wire of the other battery holder. Refer to the following picture to make sure you connect the right wires together.

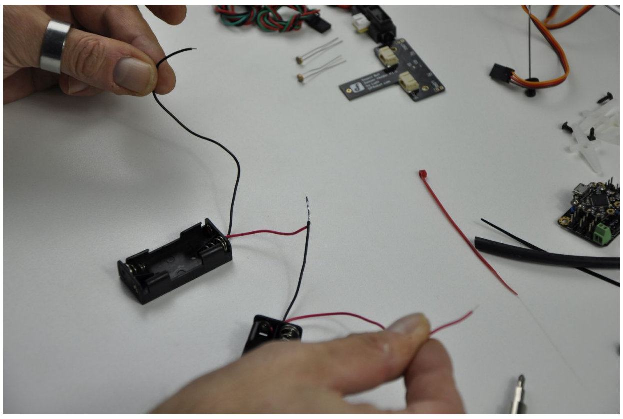
After soldering the battery holder wires together cover the wire with a piece of shrink tube to prevent short circuits. To do this, cut 2cm of the provided shrink tube, put it on the wire end and heat it carefully with a lighter or your soldering iron. Heat the tube only for 1 second or less at the time to prevent it from burning.











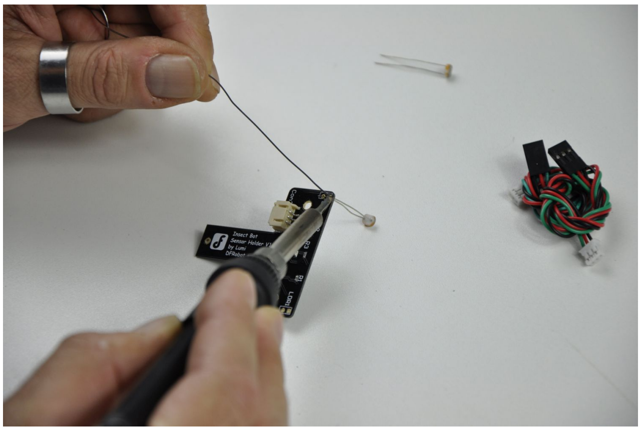
The battery holders are attached on the left and the right side of the robot body. Take the 3M foam tape and cut one for each battery holder approximately 1cm x 1cm and stick it on the backside of the battery holder opposite to the cable. Then remove the other protection layer and stick it on the robot body, precisely on the front servo. Make sure the screw hole on the other side will align with the hole in the center mounting piece. Use one of the screws (above in section 2 marked as mounting screw) from the servos for each battery holder to secure it on the middle mounting piece.

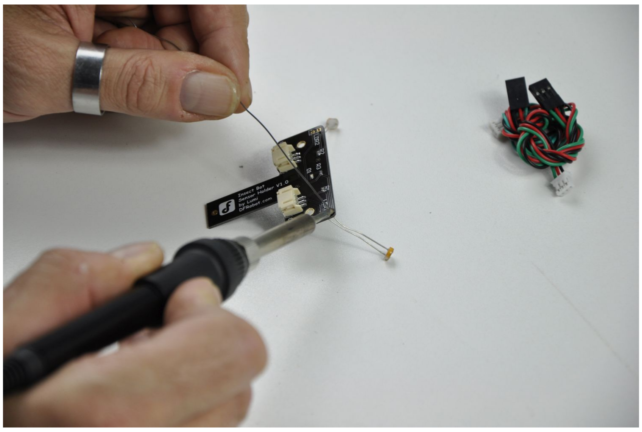
5. Attaching the sensor holder and the IR sensor

Optional: Solder the two LDR's (also called photo resistors) to the designated place on the sensor holder. The two holes on the left and the right side of the sensor holder are labeled with LDR 1 and LDR 2. After soldering the LDR's to the sensor holder cut 2cm length of shrink tube to cover the sensors to avoid getting light from the side. Make sure that the two leads of the LDR are not shorted.

Attach the sensor holder by fixing it with one of the supplied servo screws to the upper mounting hole of the middle servo. Make sure you do not overturn the screw.

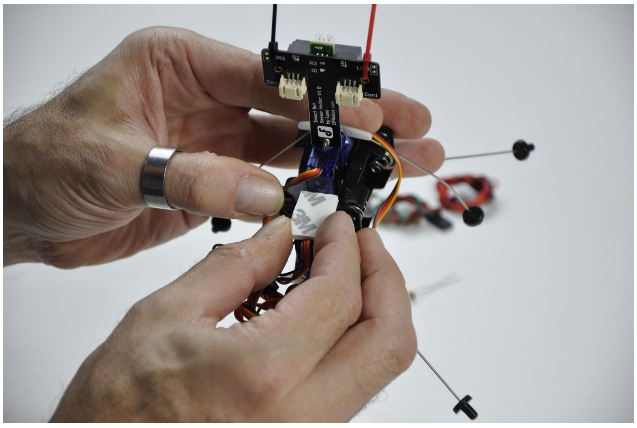
Use two of the cable ties to attach the IR sensor to the sensor holder. The connector of the IR sensor may face up or down depending on your way to route the cable later.





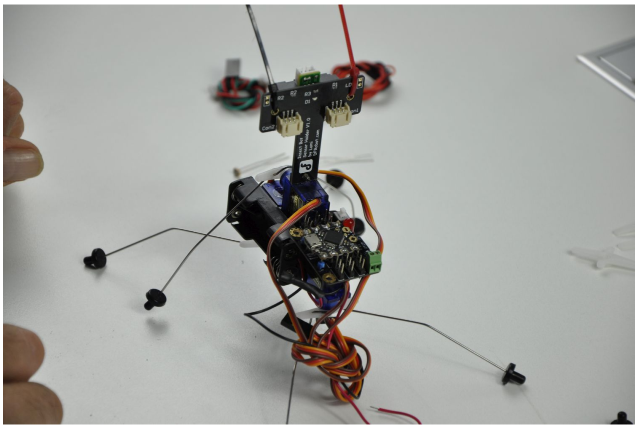
6. Attaching the Beetle to the robot body

Use the supplied 3M double side foam tape to stick the Beetle shield with the Beetle controller on the back of the robot body.



7.Adjust the legs

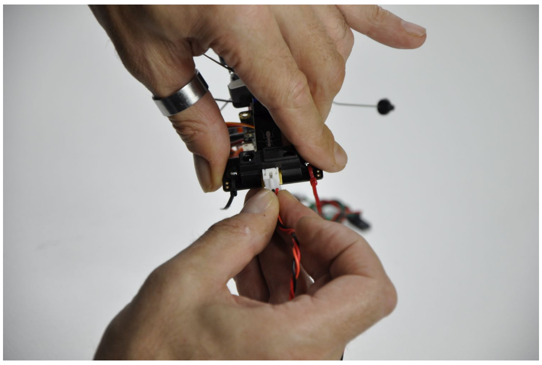
Adjust the 3 pairs of legs to the level ground, it means a lot to walk then,

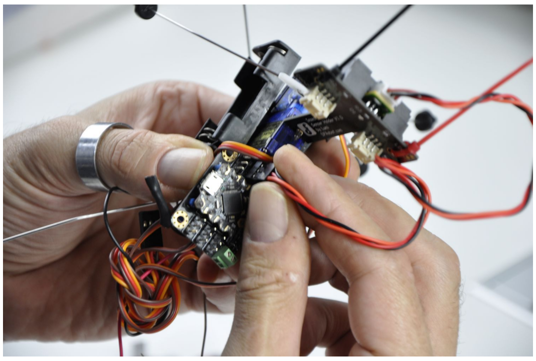


8. Wire-up

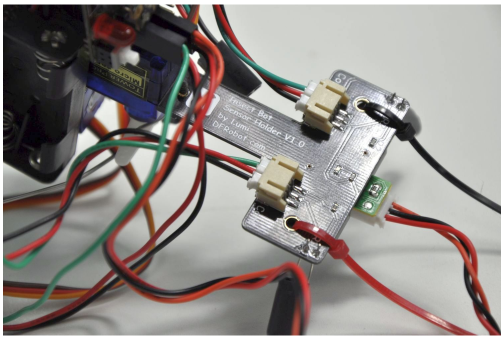
The final task is to wire up the robot.

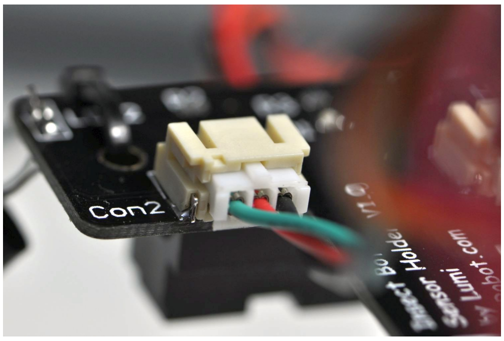
8.1.IR Sensor: Connect the sensor cable to the socket on the sensor and then to the pin A1 on the Beetle shield. The socket on the sensor is coded, so only one direction is possible. However, the other side of the cable has a black plug. The wire colors are Black, Red, Blue. The Black wire connects to ground. The ground pins are the pins on the outer edge of the shield. The blue wire connects to the signal pin. All signal pins on the shield are the ones which faces directly to the Beetle.

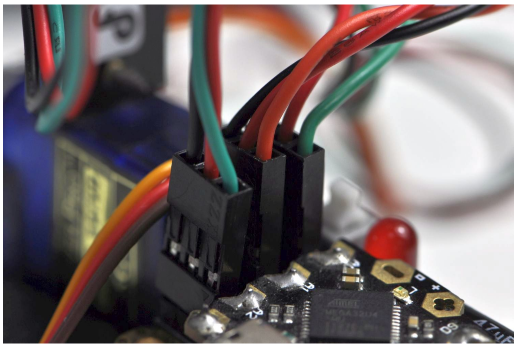




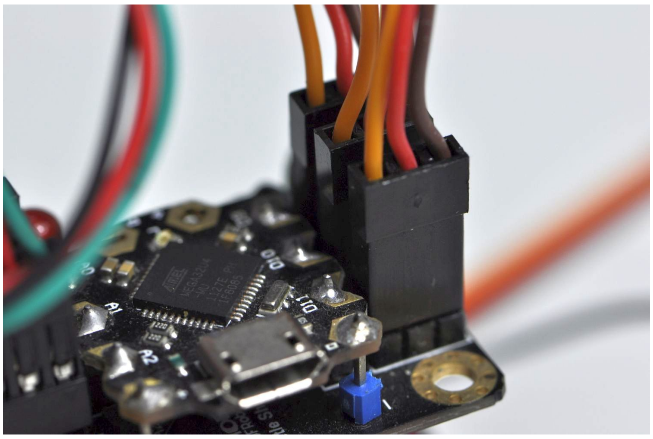
8.2.Optional LDR's: Connect the sensor cable with the sockets on the sensor holder and the other end with the Beetle shield. The right LDR connects with A0 and the left sensor with A2. The sensor cable for the LDR may have a different color for the signal pin but ground and VCC stays the same. Black for ground and blue or green for the signal pin.



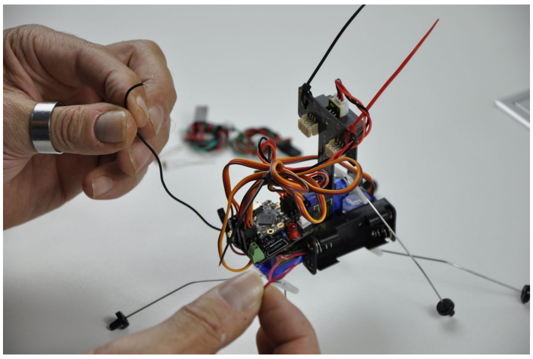


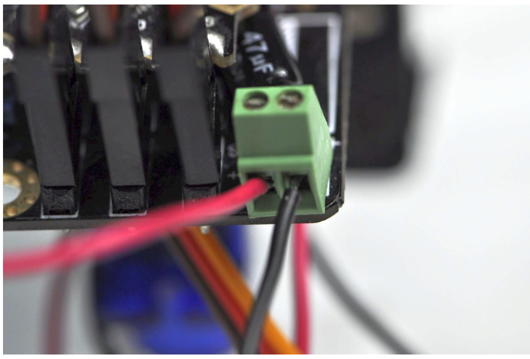


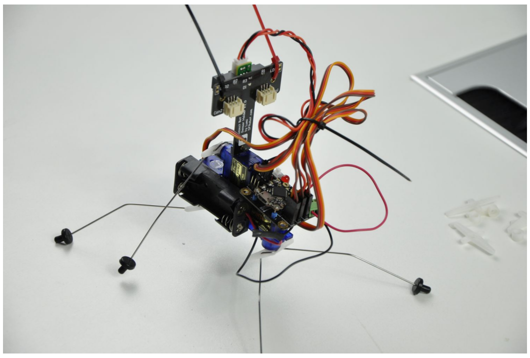
8.3. Servos: The servo cables also comes with three different wire colors. Brown, red and yellow (or orange). Brown connects with ground and the yellow wire with the signal pin. The order of the servos is the following: Front servo connects to D3, the rear servo to D4 and the middle servo to D5. Also here is the signal pin on the inside, facing to the Beetle and the ground pin on the outside of the Beetle shield.



8.4. Battery: The last task is to connect the battery wires to the Beetle shield battery terminal. Make sure the red wire is connected to „+“ and the black wire to „-“.

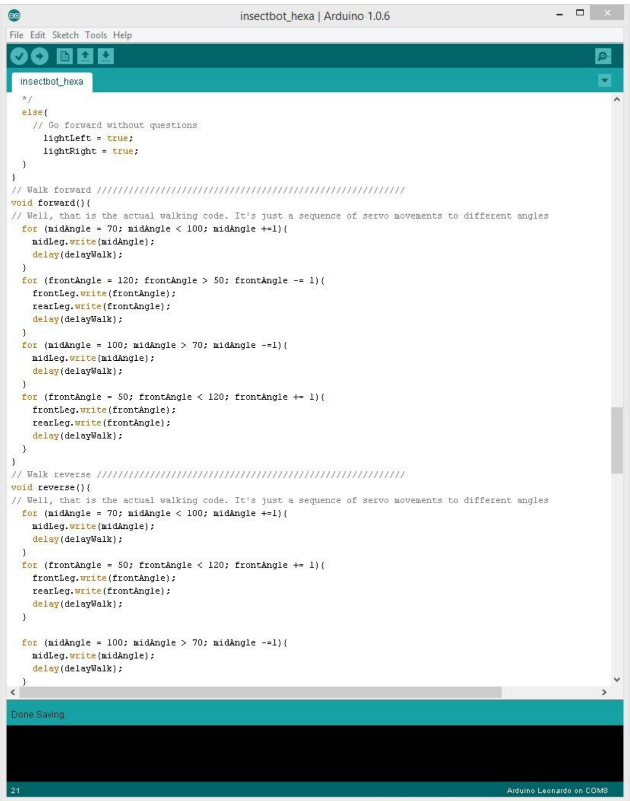






9. Program upload

To program the robot you need to have a computer with the Arduino IDE installed.
The connected Beetle will show up as a Uno. Please choose this and select the proper COM port.
Open the previous downloaded file **insectbot_hexa_en.ino** and upload it to the Beetle.
Once it's done without errors the InsectBot Hexa is ready to take his first steps.



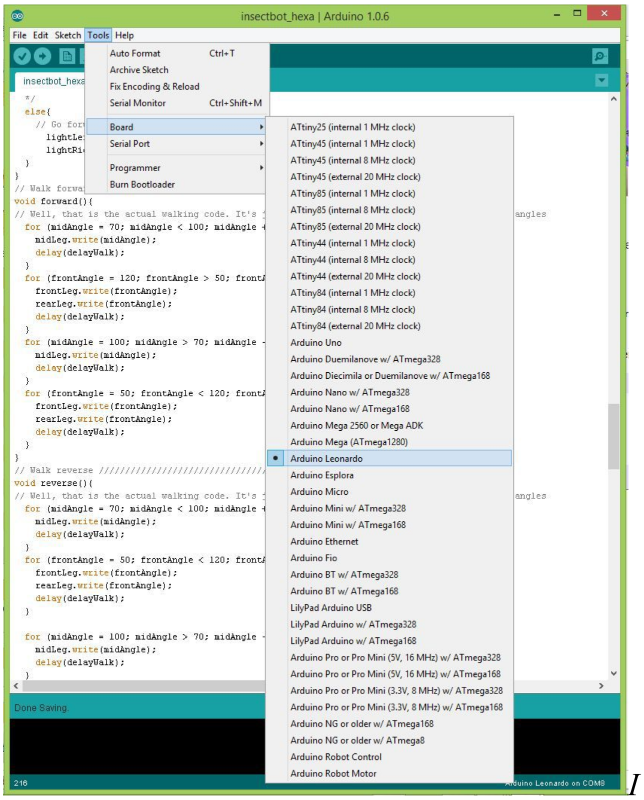
```
  */
  else{
    // Go forward without questions
      lightLeft = true;
      lightRight = true;
  }
}
// Walk forward ///////////////////////////////////////////////////////////
void forward(){
// Well, that is the actual walking code. It's just a sequence of servo movements to different angles
  for (midAngle = 70; midAngle < 100; midAngle +=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
  for (frontAngle = 120; frontAngle > 50; frontAngle -= 1){
    frontLeg.write(frontAngle);
    rearLeg.write(frontAngle);
    delay(delayWalk);
  }
  for (midAngle = 100; midAngle > 70; midAngle -=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
  for (frontAngle = 50; frontAngle < 120; frontAngle += 1){
    frontLeg.write(frontAngle);
    rearLeg.write(frontAngle);
    delay(delayWalk);
  }
}
// Walk reverse ///////////////////////////////////////////////////////////
void reverse(){
// Well, that is the actual walking code. It's just a sequence of servo movements to different angles
  for (midAngle = 70; midAngle < 100; midAngle +=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
  for (frontAngle = 50; frontAngle < 120; frontAngle += 1){
    frontLeg.write(frontAngle);
    rearLeg.write(frontAngle);
    delay(delayWalk);
  }

  for (midAngle = 100; midAngle > 70; midAngle -=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
```

Done Saving.

21 Arduino Leonardo on COM8

Illustration 1: Open the program file

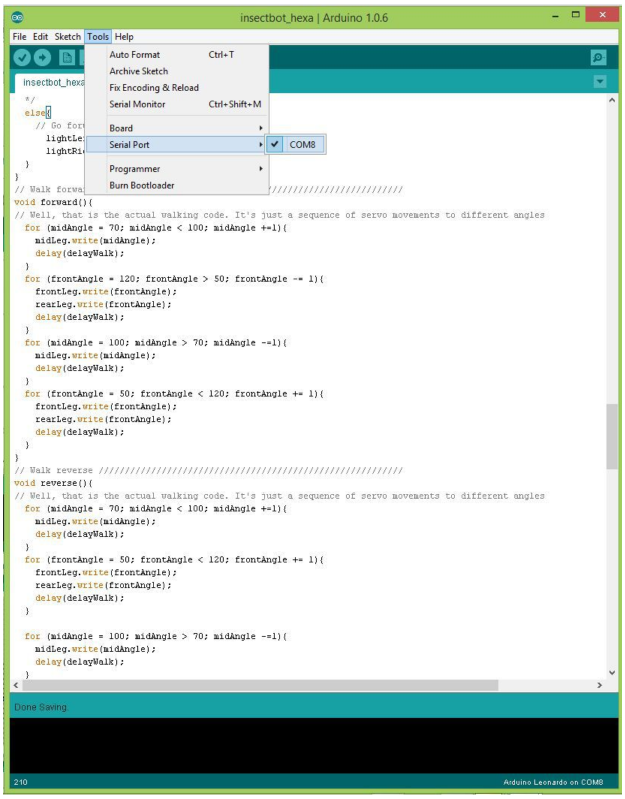

insectbot_hexa | Arduino 1.0.6
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Board
Serial Port
Programmer
Burn Bootloader
ATtiny25 (internal 1 MHz clock)
ATtiny45 (internal 1 MHz clock)
ATtiny45 (internal 8 MHz clock)
ATtiny45 (external 20 MHz clock)
ATtiny85 (internal 1 MHz clock)
ATtiny85 (internal 8 MHz clock)
ATtiny85 (external 20 MHz clock)
ATtiny44 (internal 1 MHz clock)
ATtiny44 (internal 8 MHz clock)
ATtiny44 (external 20 MHz clock)
ATtiny84 (internal 1 MHz clock)
ATtiny84 (internal 8 MHz clock)
ATtiny84 (external 20 MHz clock)
Arduino Uno
Arduino Duemilanove w/ ATmega328
Arduino Diecimila or Duemilanove w/ ATmega168
Arduino Nano w/ ATmega328
Arduino Nano w/ ATmega168
Arduino Mega 2560 or Mega ADK
Arduino Mega (ATmega1280)
● Arduino Leonardo
Arduino Esplora
Arduino Micro
Arduino Mini w/ ATmega328
Arduino Mini w/ ATmega168
Arduino Ethernet
Arduino Fio
Arduino BT w/ ATmega328
Arduino BT w/ ATmega168
LilyPad Arduino USB
LilyPad Arduino w/ ATmega328
LilyPad Arduino w/ ATmega168
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168
Arduino NG or older w/ ATmega168
Arduino NG or older w/ ATmega8
Arduino Robot Control
Arduino Robot Motor

Done Saving.

216 Arduino Leonardo on COM8

I

llustration 2: Choose Uno as board



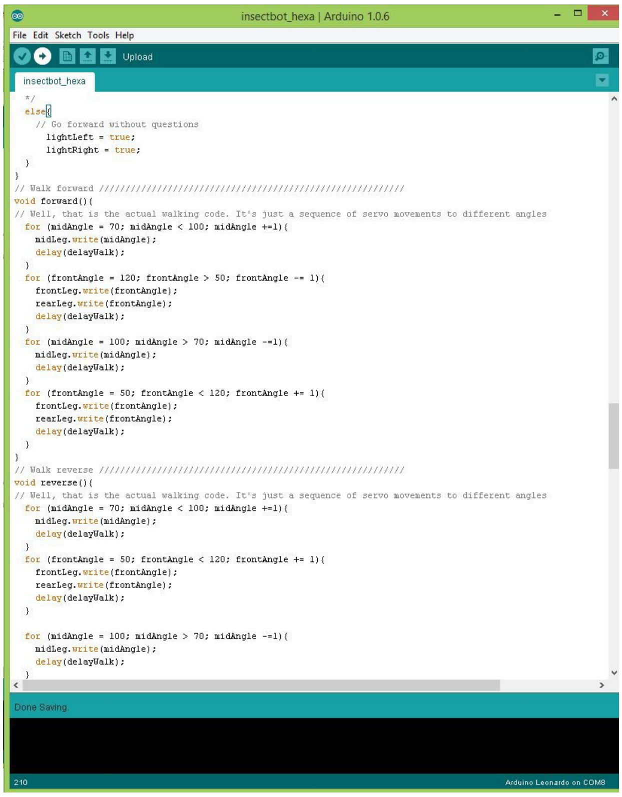
```
void forward(){
// Well, that is the actual walking code. It's just a sequence of servo movements to different angles
  for (midAngle = 70; midAngle < 100; midAngle +=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
  for (frontAngle = 120; frontAngle > 50; frontAngle -= 1){
    frontLeg.write(frontAngle);
    rearLeg.write(frontAngle);
    delay(delayWalk);
  }
  for (midAngle = 100; midAngle > 70; midAngle -=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
  for (frontAngle = 50; frontAngle < 120; frontAngle += 1){
    frontLeg.write(frontAngle);
    rearLeg.write(frontAngle);
    delay(delayWalk);
  }
}
// Walk reverse ///////////////////////////////////////////////////////////
void reverse(){
// Well, that is the actual walking code. It's just a sequence of servo movements to different angles
  for (midAngle = 70; midAngle < 100; midAngle +=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
  for (frontAngle = 50; frontAngle < 120; frontAngle += 1){
    frontLeg.write(frontAngle);
    rearLeg.write(frontAngle);
    delay(delayWalk);
  }

  for (midAngle = 100; midAngle > 70; midAngle -=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
```

Done Saving.

210 Arduino Leonardo on COM8

Illustration 3: Choose the port (this my be different on your computer)



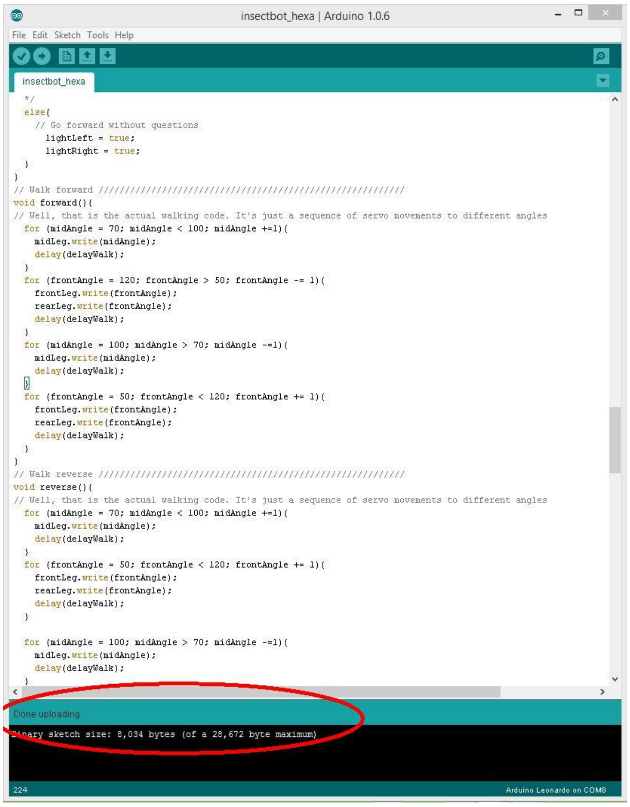
```
  */
  else{
    // Go forward without questions
      lightLeft = true;
      lightRight = true;
  }
}
// Walk forward ///////////////////////////////////////////////////////////
void forward(){
// Well, that is the actual walking code. It's just a sequence of servo movements to different angles
  for (midAngle = 70; midAngle < 100; midAngle +=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
  for (frontAngle = 120; frontAngle > 50; frontAngle -= 1){
    frontLeg.write(frontAngle);
    rearLeg.write(frontAngle);
    delay(delayWalk);
  }
  for (midAngle = 100; midAngle > 70; midAngle -=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
  for (frontAngle = 50; frontAngle < 120; frontAngle += 1){
    frontLeg.write(frontAngle);
    rearLeg.write(frontAngle);
    delay(delayWalk);
  }
}
// Walk reverse ///////////////////////////////////////////////////////////
void reverse(){
// Well, that is the actual walking code. It's just a sequence of servo movements to different angles
  for (midAngle = 70; midAngle < 100; midAngle +=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
  for (frontAngle = 50; frontAngle < 120; frontAngle += 1){
    frontLeg.write(frontAngle);
    rearLeg.write(frontAngle);
    delay(delayWalk);
  }

  for (midAngle = 100; midAngle > 70; midAngle -=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
```

Done Saving.

210 Arduino Leonardo on COM8

Illustration 4: Upload the program by clicking on the white arrow button in the top



```
  */
  else{
    // Go forward without questions
      lightLeft = true;
      lightRight = true;
  }
}
// Walk forward ////////////////////////////////////////////////////////////
void forward(){
// Well, that is the actual walking code. It's just a sequence of servo movements to different angles
  for (midAngle = 70; midAngle < 100; midAngle +=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
  for (frontAngle = 120; frontAngle > 50; frontAngle -= 1){
    frontLeg.write(frontAngle);
    rearLeg.write(frontAngle);
    delay(delayWalk);
  }
  for (midAngle = 100; midAngle > 70; midAngle -=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
  for (frontAngle = 50; frontAngle < 120; frontAngle += 1){
    frontLeg.write(frontAngle);
    rearLeg.write(frontAngle);
    delay(delayWalk);
  }
}
// Walk reverse ////////////////////////////////////////////////////////////
void reverse(){
// Well, that is the actual walking code. It's just a sequence of servo movements to different angles
  for (midAngle = 70; midAngle < 100; midAngle +=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
  for (frontAngle = 50; frontAngle < 120; frontAngle += 1){
    frontLeg.write(frontAngle);
    rearLeg.write(frontAngle);
    delay(delayWalk);
  }

  for (midAngle = 100; midAngle > 70; midAngle -=1){
    midLeg.write(midAngle);
    delay(delayWalk);
  }
```

Done uploading.

```
Binary sketch size: 8,034 bytes (of a 28,672 byte maximum)
```

224 Arduino Leonardo on COM8

Illustration 5: Once "Done uploading" is shown, you are finished

Congratulation, you just finished your own little robot. You may keep it that way or you may even try to re-program the Insectbot Hexa for your own experience and fun.

DFROBOT

DRIVE THE FUTURE

© by DFRobot.com