Übungen mit Servos

Lies dir bitte zuerst das Dokument

→ **Servo-Motor**

aufmerksam durch.

Neu für dich könnte hier auch die Verwendung des ***Sensor-Shield*** sein.
Es ist eine Platine zum Aufstecken auf das Arduino und es bietet dir in unserem Fall zu jedem Pin des Arduino zusätzlich eine eigene Spannungsversorgung:

- V = Voltage = 5V = Plus-Pol
- G = Ground (GND) = 0V = Minus-Pol
- S = Sensor = der gewöhnliche Pin

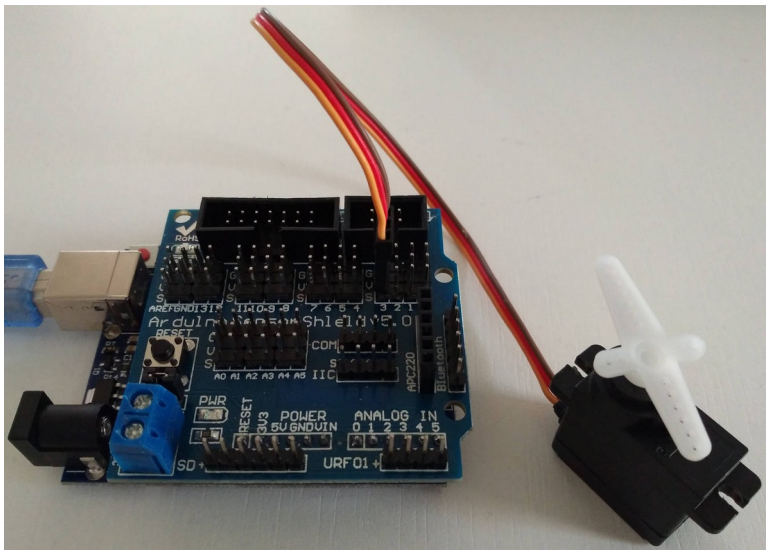
Außerdem hast du dank des Sensor-Shield statt der gewohnten „weiblichen“ Pins „männliche“ Pins. Damit kannst du den 3-poligen Servo-Anschluss ganz einfach aufstecken und brauchst keine Adapter-Stecker oder zusätzliche Kabel ...

Grundlegende Funktion

Hardware

Du benötigst ***1 Servo-Motor***
1 Arduino (Uno)
1 Sensor-Shield

Aufbau



Verbinde den Servo mit dem Sensor-Shield deines Arduino. Stecke ihn auf den digitalen Pin 3 und achte darauf, dass das braune Kabel auf G (GND), das rote Kabel auf V (Voltage, 5V) und das orange Kabel auf S (Sensor) steckt.

Grundlegende Funktion

1. Erzeuge in BlueJ durch Rechts-Klick interaktiv ein Objekt der Klasse Servo.
   Der Servo kann dabei ein erstes Mal zucken … das ist normal.

2. Verwende nun interaktiv die Methode `setValue(...)` und beginne mit dem Übergabe-Wert 20.
   Der Servo sollte sich nun nach links drehen. Rufe die Methode mehrmals erneut auf und taste dich in 5er-Schritten an den Wert 0 heran. Sollte der Servo im Stillstand plötzlich brummen, so erhöhe den Wert so lange wieder um 1, bis das Brummen aufhört.
   Du hast nun für den von dir verwendeten Servo den Wert für „ganz links“ ermittelt.
   Rufe nun die Methode `setLeft(...)` auf und übergib ihr genau diesen Wert.

3. Verfahre ebenso für die Drehung nach rechts.
   Beginne mit dem Wert 140 und taste dich in 5er-Schritten immer weiter nach oben, bis der Servo im Stillstand wieder brummt. Gehe dann in 1er-Schritten wieder zurück, bis das Brummen aufhört. Das ist der Wert für „ganz rechts“.
   Rufe die Methode `setRight(...)` auf und übergib ihr diesen Wert.
   Dein Servo ist nun so eingerichtet, dass er nicht kaputt gehen kann …

4. Experimentiere noch ein Wenig auf diese Art mit dem Servo herum. Testete insbesondere nun die Methoden `left()` , `right()` , `center()` und `getValue()` …

5. Bisher dreht sich der Servo relativ langsam. Du kannst einstellen, wie schnell sich der Servo an seine neue Position dreht. Benutze hierfür die Methode `setDelay(...)`. Der voreingestellte Wert ist 20. kleinere Werte (>=0) lassen den Servo schneller drehen, größere Werte als 20 lassen ihn langsamer drehen.
   Teste deine Einstellungen z.B. mit den Methoden `left()` , `right()` , `center()` …

Ein kleines Rate-Spiel

6. Stelle den Servo mittig auf ein Blatt Papier. Schreibe auf die linke Seite „zu klein“, auf die rechte Seite „zu groß“ und mittig über den Servo „richtig“.

7. Schreibe eine ***Klasse Ratespiel***, die ein Referenz-Attribut vom Typ Servo hat.
   Außerdem brauchst du ein ganzzahliges Attribut.
   Erzeuge im Konstruktor das Servo-Objekt und weise der Zahl eine Zufallszahl von 1 bis Hundert zu: `this.zahl = (int)(Math.random()*101)`

8. Schreibe eine ***Methode*** `raten(...)` die vom Benutzer eine ganze Zahl als Übergabe erwartet.
   Diese Methode soll die übergeben Zahl mit der (unbekannten) Zufallszahl vergleichen.
   Ist die geratene Zahl zu klein, soll der Servo nach links laufen, ist sie zu groß, nach rechts.
   Hat man richtig geraten, so stellt sich der Servo mittig.
   Wenn du willst, kannst du auch noch einen Summer integrieren, der drei Mal kurz summt, wenn die Zahl richtig geraten wurde.

9. Schreibe eine Methode `neueZufallsZahl()`, die der Zufallszahl einen neuen Wert gibt.