PRIME LESSONS
By the Makers of EV3Lessons

MOVING OBJECTS & STALL DETECTION

BY SANJAY AND ARVIND SESHAN

This lesson uses SPIKE 3 software

LESSON OBJECTIVES

- Learn how to move non-drive motors
- Learn about motor stalls

SINGLE MOTOR FUNCTIONS (ACTIONS)

- To use single motor functions, import the motor module:

```
import motor
```

- Each motor function inputs the motor port as a parameter.
- To run for a certain duration, use the following methods (see Knowledge Base for more info). These functions are **asynchronous** so use **await** if you want to wait for them to complete.

```
run_for_degrees(port, degrees, velocity)

run_for_time(seconds,  duration, velocity)
```

- To start running the motors, until stopped at a later spot

```
run(port, velocity)

stop()
```

- `To run the motor to a specific position`

```
run_to_relative position(port, position, velocity)

run_to_absolute_position(port, position, velocity, direction=motor.SHORTEST_PATH)
```

  - Direction options are CLOCKWISE, COUNTERCLOCKWISE, LONGEST_PATH
  - These functions are **asynchronous** so use **await** if you want to wait for them to complete.

SINGLE MOTOR FUNCTIONS (MEASUREMENTS/SETTINGS)

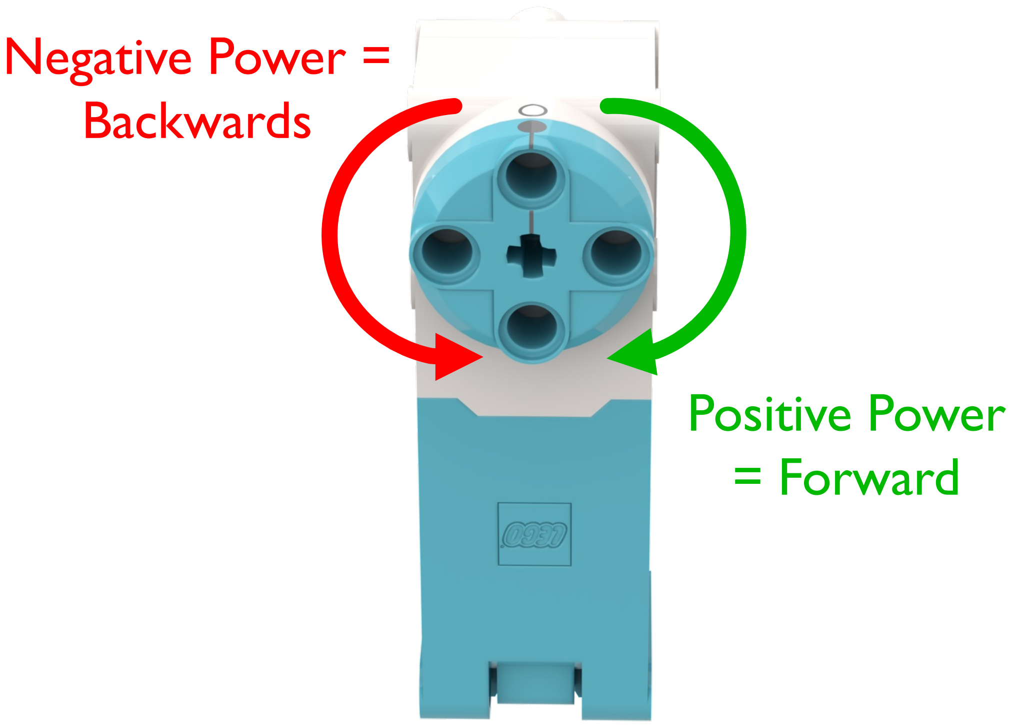
- The rotation sensor in the motor can be used to tell the number of degrees the motor has turned
- To do this, use `reset_relative_position(port, position)` and `relative_position(port)`
- Just like motor pairs, you can use parameters to change motor behavior
  - Set the stop action (BRAKE, COAST etc.)
  - Set acceleration, velocity etc.
- You can also read different measurement methods associated with the motor

```
velocity(port)

relative_position(port)

absolute_position(port)
```

NEGATIVE VALUES



- You can enter negative values for velocity or degrees
- This will make the robot move backwards
- If you negate two values (e.g., velocity and degrees or degrees and backwards direction), the robot will move forward.

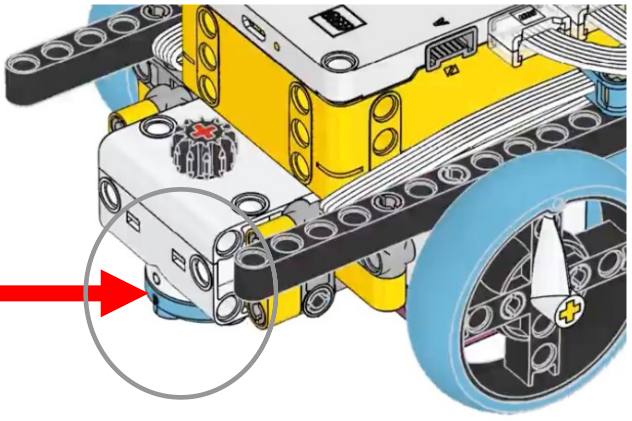
STALL DETECTION

- Often, you program the motor to move a particular amount. However, the motor gets stuck before it reaches that amount.
- Stall Detection allows your program to automatically move on to the next line in the code when a particular motor command is stuck (unable to complete its move)
- SPIKE Prime has a built-in Stall Detection
- By default, Stall Detection is **on** when using single motor functions
- As of version 3.4, SP3 does not allow stall detection to be changed or queried

CHALLENGE 1: LEARN ABOUT STALL WITH DB1

- Write a program to turn the attachment motor E by1000 degrees.
- Add a beeping sound after the motor command.
- Hold the motor with your hand to prevent it from completing 1000 degrees. Hold for a couple of seconds.
- Will the sound play?

Cause a stall by holding the motor blue part and preventing it from turning



CHALLENGE 1 SOLUTION

- Stall detection allowed the code to move on to the next line even when the motor got stuck. The beep played even though the motor did not finish completing its command.

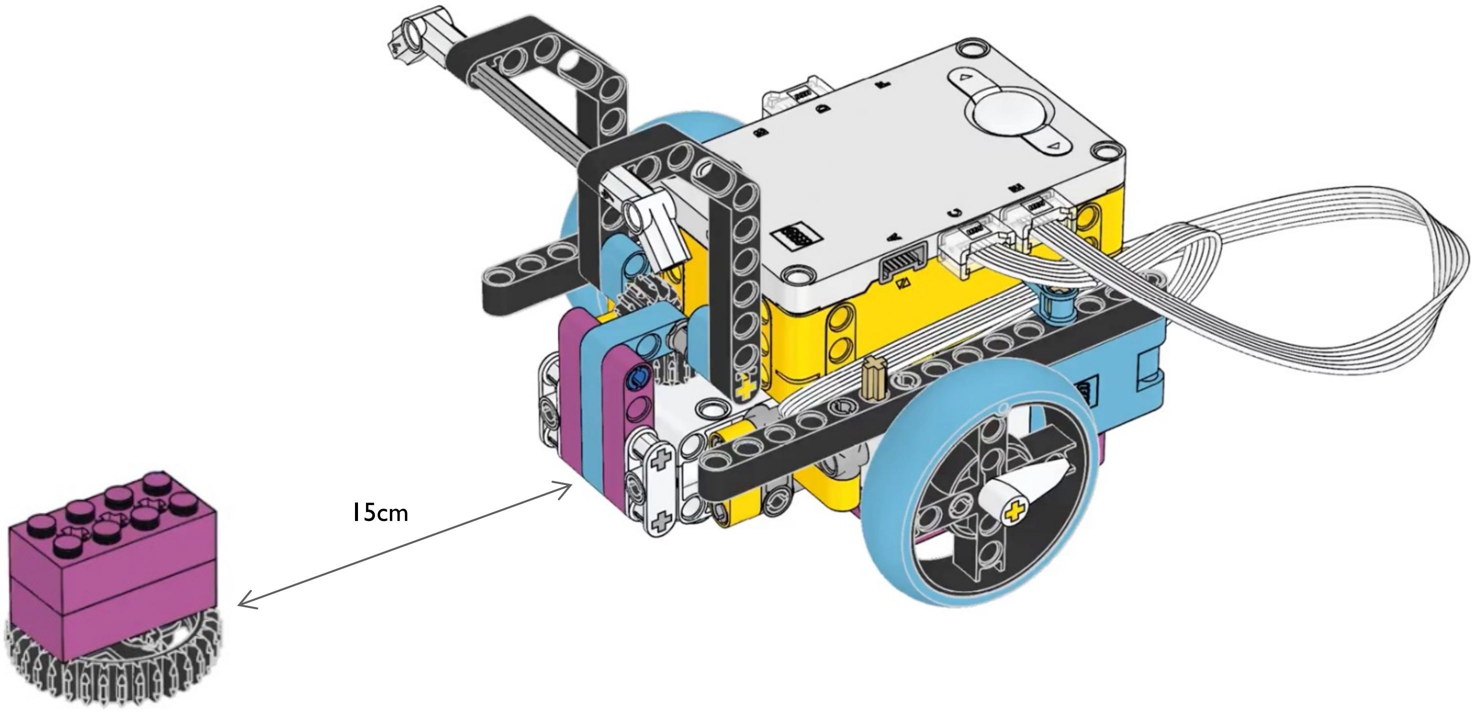
```
from hub import port, sound
import motor, runloop, sys

async def main():
    # Start the motor
    await motor.run_for_degrees(port.E, 1000, 100)
    await sound.beep()
    sys.exit("Stopping")

runloop.run(main())
```

CHALLENGE 2: GRAB OBJECT (DB2 CHALLENGE)

- Build the attachment part of DB2
- Drive forward, grab a marker and drag it back to the start and release it.
- Marker is about 15cm from the front magenta-and-cyan panel.



CHALLENGE 2 SOLUTION

```python
from hub import port
import motor, motor_pair, runloop, sys

# Constants for Drive Base 1, adjust for your robot
motor_pair.pair(motor_pair.PAIR_1, port.C, port.D)
WHEEL_CIRCUMFERENCE = 17.5 # cm

# See lesson on More Accurate Turns for explanation of math
def degrees_for_distance(distance_cm):
    return int((distance_cm/WHEEL_CIRCUMFERENCE) * 360)

async def main():
    # Move forward 14.5cm
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, degrees_for_distance(14.5),0)
    # lower the arm
    await motor.run_for_degrees(port.E, -90, 100)
    # move backward
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, degrees_for_distance(-14.5),0)
    # raise the arm
    await motor.run_for_degrees(port.E, 90, 100)
    sys.exit("Done")

runloop.run(main())
```

EXTENSIONS

- Think about situations in FIRST LEGO League when stall detection would be helpful
  - When might the robot get stuck?

CREDITS

- This lesson was created by Sanjay and Arvind Seshan for Prime Lessons
- Additional contributions by FLL Share & Learn community members
- More lessons are available at www.primelessons.org