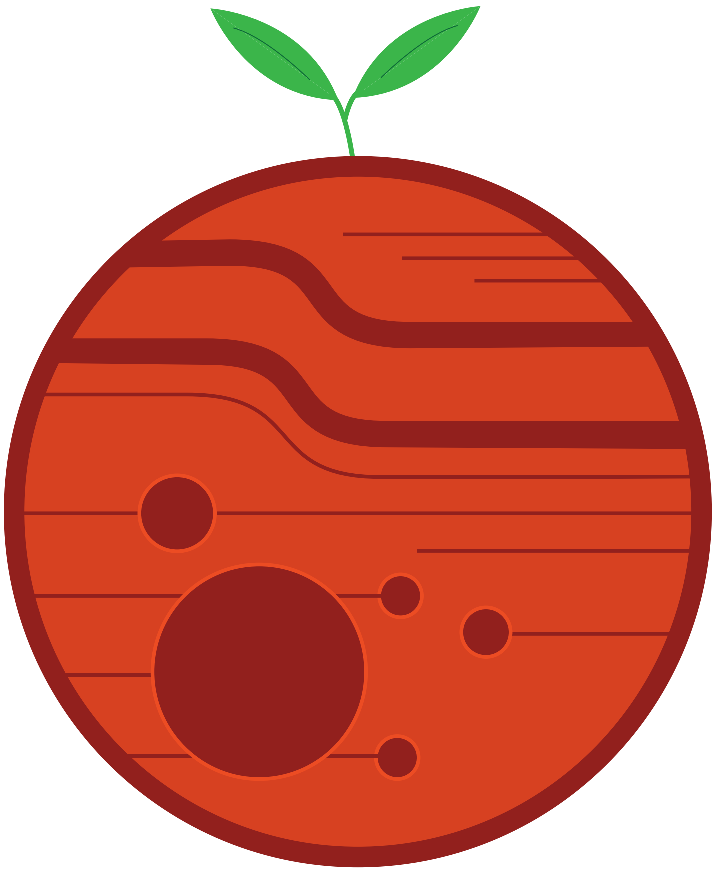
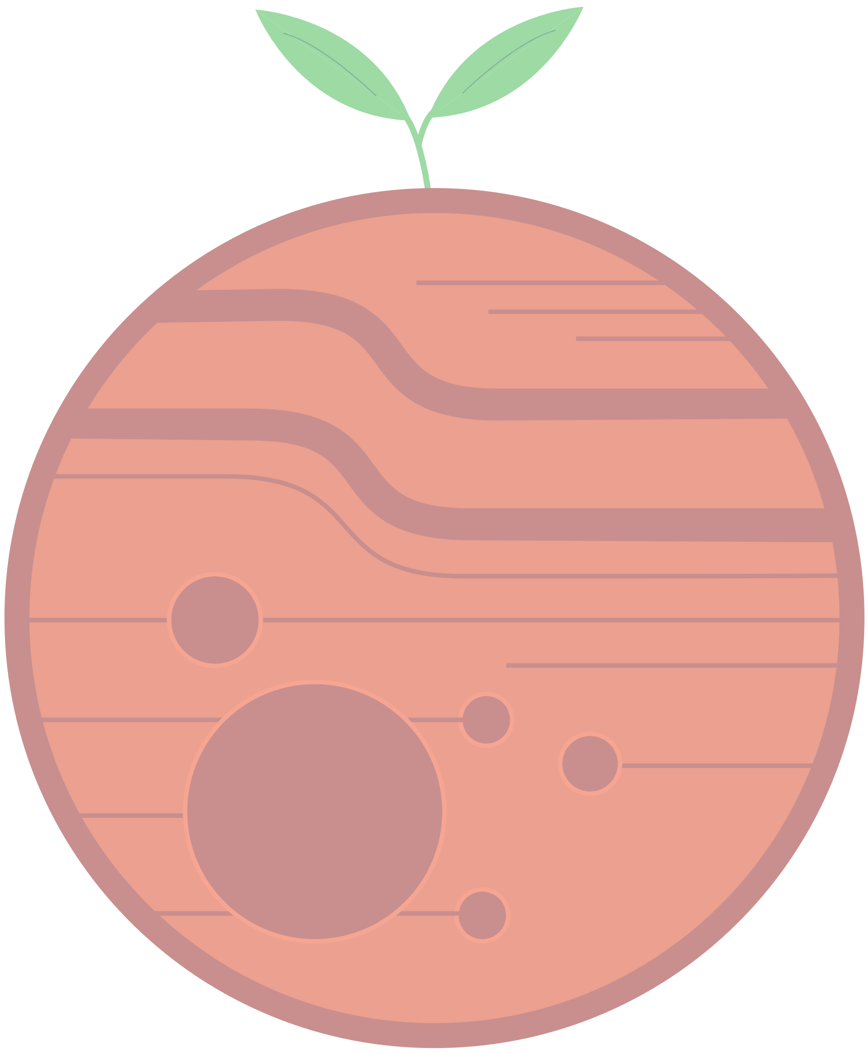
teach with space

→ PLANTS ON MARS

Build an automatic plant watering system



European Space Agency

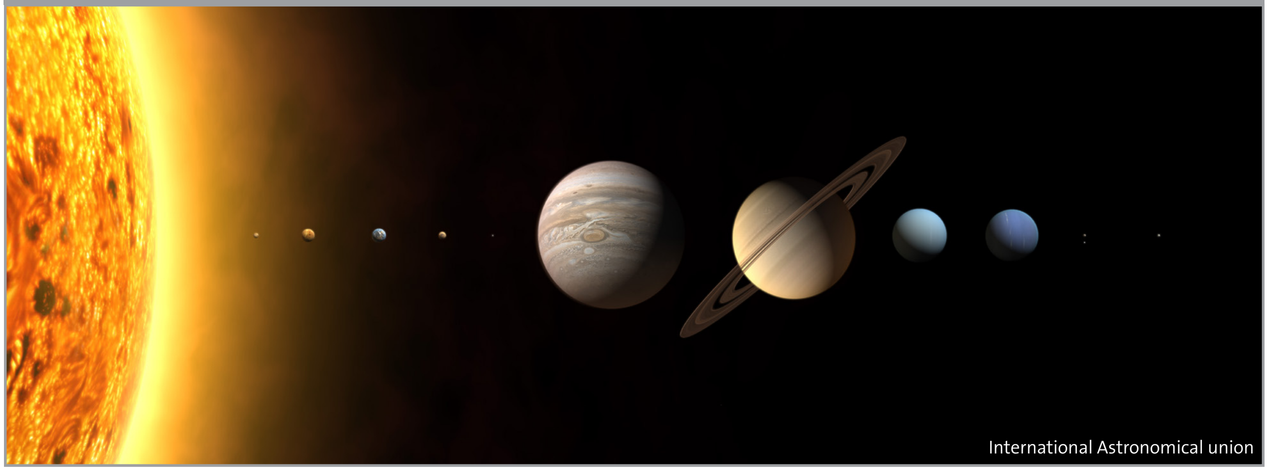


→ ACTIVITY 0: WELCOME TO MARS

Introduction

Mars is the fourth planet from the Sun and the Earth's second closest neighbour, after Venus. The average distance between Earth and Mars is a huge 55 million kilometres, compared to the relatively short 380,000 kilometres between the Earth and the Moon. This increases the complexity of any missions to Mars significantly, as it is much costlier and more difficult to send supplies.

Figure A1



↑ The solar system

A possible solution to this is for the astronauts to take plant seeds with them. This would allow the astronauts to grow the seeds once they arrive, and create a self-sufficient food source.

However, this is not an easy task. There are multiple factors that create a hazardous environment for plants on Mars.

Figure A2

↑ An artist's impression of how a greenhouse on Mars might look.

Exercise

1) To start thinking about this in more detail, list some of the things plants and other living organisms need in order to survive:

2) Decide whether the following statements are true or false. Check with your teacher once you have decided, and add any extra information in the spaces provided.

Statements about Mars - Table A1

Statement	True or False
Mars experiences seasons, just like we do on Earth.	
The orbit of Mars is a similar shape to Earth's, meaning that the temperature on the surface is fairly constant.	
Mars has a thick atmosphere, trapping heat from the Sun.	
Mars has no magnetic field, this means that there is less protection from harmful UV radiation and solar winds.	
We have found liquid water on the surface of Mars.	
The atmosphere of Mars has a similar composition to Earth's atmosphere.	
Plants on Mars would need to adapt to the massively different day and night cycles on Mars.	
Mars does not exist inside the 'Goldilocks' (habitable) zone, so it is impossible for liquid water to exist on the surface.	

In activities 1 to 7, we will act as space explorers that are on a mission to Mars in order to establish a Mars outpost. To increase the chances of success of the mission, we will build an automatic plant watering system. We are going to experiment and design a prototype here on Earth, so that we can later on adapt it to the Martian environment!

Background exercise – Introduction to Arduino

To get to grips with Arduino and the basics of C++, use the resource 'Meet Arduino!'. Here you will be guided through using several sensors to make measurements of the environment and begin to appreciate how an Arduino can be used.

→ ACTIVITY 1: PREPARING THE ITEMS AND FIRST DESIGN

Introduction

In order for a mission to Mars to be successful, the astronauts will need to be as self-sustaining as possible. This includes recycling as much of their resources as possible and growing their own food.

Plants are a valuable resource. Vegetables are a nutrient-dense source of food that can be grown from small seeds and bulbs, limiting the amount of material that is carried onboard the spacecraft. Photosynthesis, a process carried out in plants to produce glucose for growth and respiration, requires carbon dioxide, of which there is an abundance in the Martian atmosphere. However, plants require constant monitoring if they are to produce good crop, especially if their environment is not naturally abundant in the resources they require.

Maintaining an ecosystem on Mars could therefore require many hours and take up much of the astronauts' time. Our task is to begin to develop a system which would allow a computer to remotely monitor the welfare of a plant and make decisions accordingly. This would free up the astronauts to perform other tasks.

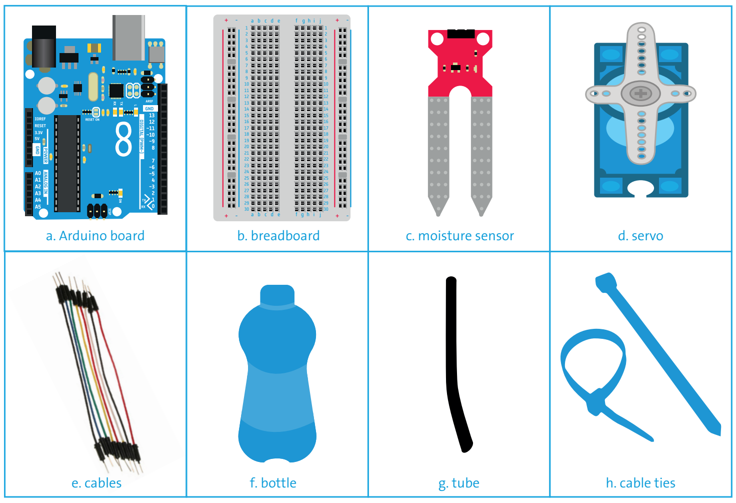
Exercise

Combined with your knowledge of how the various components work and the kit list below, your task is to devise a system that can be used to automatically water a plant depending on the moisture levels in the soil.

Equipment

- Arduino (such as Arduino Uno)
- Arduino power supply (laptop)
- Soil moisture sensor
- Servo (mini 3-5V is fine)
- Breadboard
- Tube (thin irrigation tube is perfect)
- Soil/plant
- Wires –incl. female-male and male-male
- Scissors/craft knife
- Empty bottle
- Blu-tack/adhesive
- Cable ties
- A bucket

Figure A3



↑ Plant researchers explore question of deep space food crops

Included in the list above is a servo. A servo is a small motor that, when in a fixed position, can be used to rotate a helix (circled).

First let's think about how the system might physically look – don't worry about the specific electrical connections right now!

Before we can do this we need to think about:

- How will we decide if the plant needs watering?
- How will the water be transported to the plant?
- What is possible with the equipment I have?
- How will we turn the water supply on and off?
- Is it important which tube and bottle we use?
- What problems might we face? How can we overcome these?

In the space below draw and label your initial plan for your watering system

→ ACTIVITY 2: DESIGN AND TEST YOUR WATER RESERVOIR

Introduction

We now have a basic idea of how our plant watering system will work. The next step is to refine your design through testing! In this exercise you will be guided through constructing a specific design. If your design is different, you will have to adapt the steps, or come up with your own ideas!

Exercise

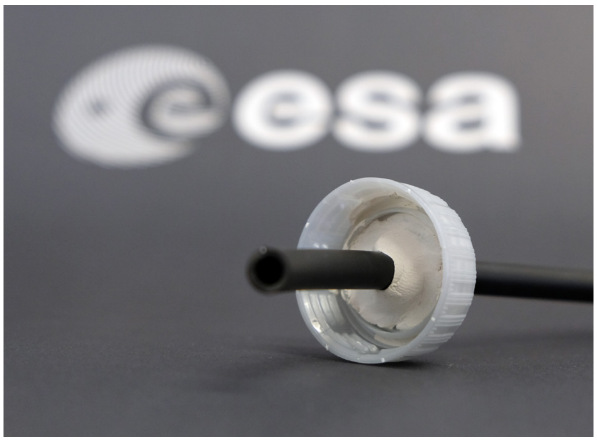
Let's start with the actual design of the system. For this step you will need:

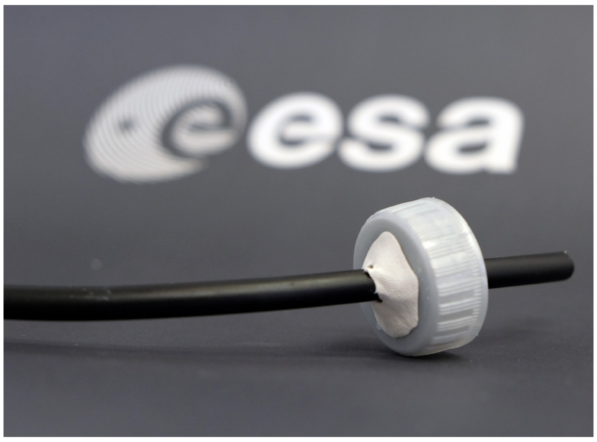
- The water bottle
- Poster tack
- Scissors
- An irrigation tube
- A bucket

A large, wide bottle makes a perfect reservoir for our watering system. Make sure you have the lid, and cut off the bottom part of the bottle to be able to keep refilling the reservoir.

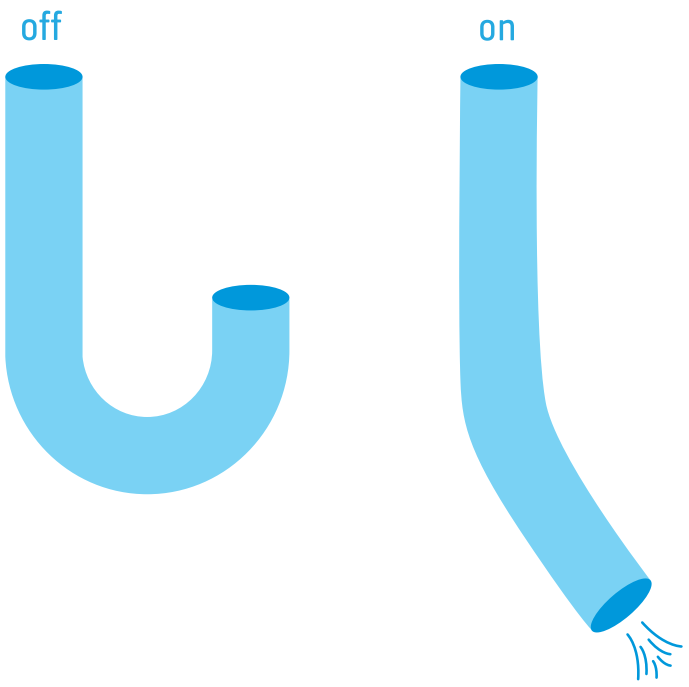
Next, we need to create a hole in the cap for our water pipe. This is best done by slowly carving a hole to the required size with a pair of scissors. Test with the water tube until you are happy with the fit. The tighter the seal the better. A teacher should perform this step! Depending on your workspace the length of tube you need might vary. But remember – there is no pump in this system so we are relying on gravity to both allow and stop the water flow. Bear this in mind!

Hopefully you were able to create a good fit for the tube, but it is likely that it is not perfectly water tight. This is easily solved with some poster-tack – a glue gun may provide a better seal, if you want to create a more permanent set-up, but it is not necessary for our project.





Now let's start thinking about how our system will work. We need to establish an 'on' and 'off' position – when the tube will be irrigating, and when it won't. An intuitive set-up will be that the end of the tube will be pointing up in the 'off' position and down in the 'on' position. For now, you only need to know that our servo motor will help us in this matter.



1) Assuming that it will be fixed, which aspects of your system do you think you will have to take into account for the setup of your water reservoir? Why?

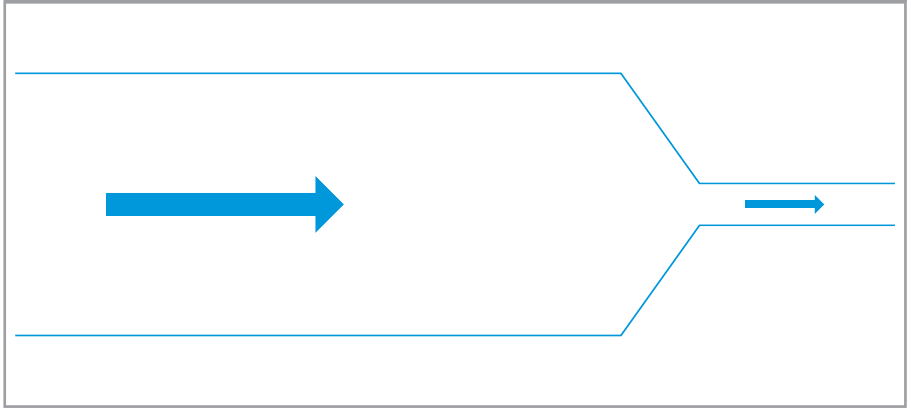
BACKGROUND

The continuity equation

Let's start to think about what aspects of our system we need to consider in our design...

$$v_1 \cdot A_1 = v_2 \cdot A_2$$

Figure A4



↑ Water flowing from a wide pipe into a thin pipe

With some mathematics, and by considering the conservation of mass and energy of the system, we can arrive at the equation above. This is known as the continuity equation. It tells us that the flow rate is constant. In fluid physics the flow rate refers to the total volume that flows per unit time, not the velocity of individual particles.

However, the continuity equation is not relevant for our watering system. But why? In our system, the water in the reservoir is said to have no velocity ($v_1 = 0$). Instead, we have to look at a piece of physics known as Torricelli's theorem.

Torricelli’s theorem

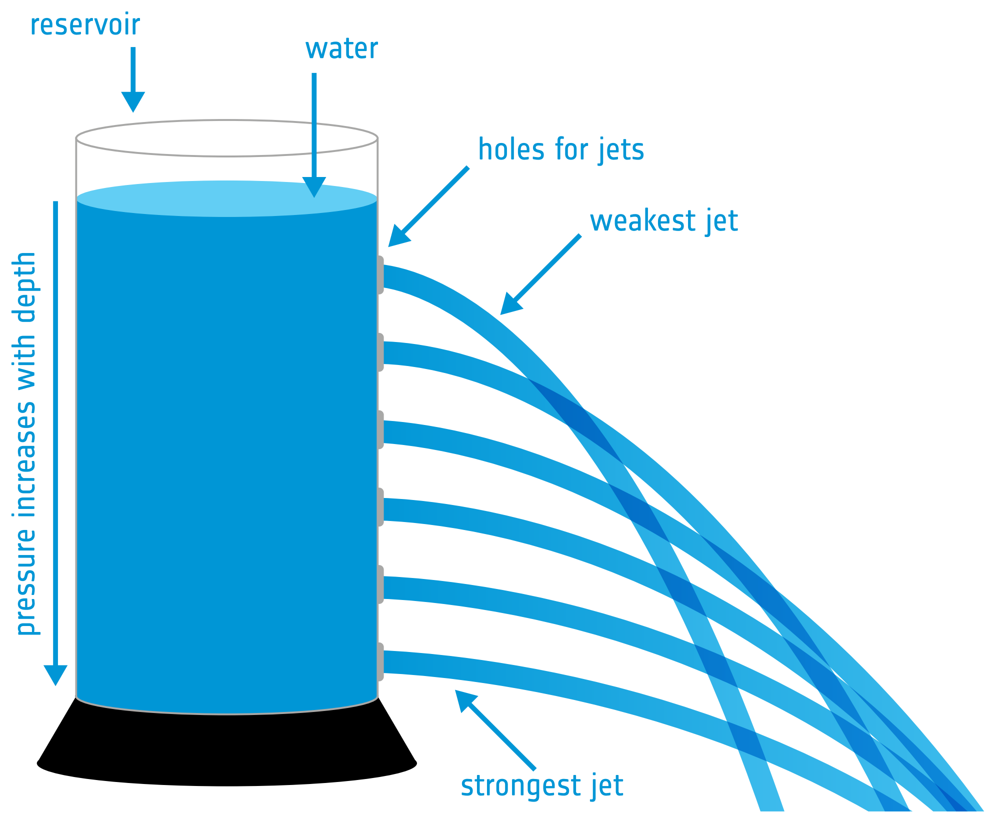
After applying the principle of conservation of energy in our system, we get to the equation known as Torricelli’s theorem:

$$v = \sqrt{2gh}$$

It tells us that the velocity of the water flow depends on the height it is falling from (h). This makes sense! It is a very important equation to consider when thinking about our system. But, just like a falling skydiver, the velocity has a limit, known as the terminal velocity.

2) What forces are responsible for the terminal velocity?

There is one final consideration that we must make when designing our system: the height of the water in the reservoir. You can understand how this works by looking at the figure below.



3) Decide if the following statements are true or false:

Statement	True or False
The water will flow more quickly through the tube than in the reservoir	
The width of the bottle is important in determining the flow rate	
The width of the pipe is important in determining the flow rate	
The height difference between the bottle and the pipe is not important	

4) Make use of your new understanding to test the reservoir and refine your design in the box below:

Draw your ideal setup after testing your water reservoir and label it.

→ ACTIVITY 3: MOUNTING THE SERVO AND CONNECTING THE WATER PIPE

Introduction

We now have a pretty good idea of what our system will look like, but at the moment it requires our input and intervention to make it work. Our aim is to make this system automated, so that astronauts can use their time more effectively. One of the ways we can do this is to make use of a servo.

Exercise

For this activity, you will need:

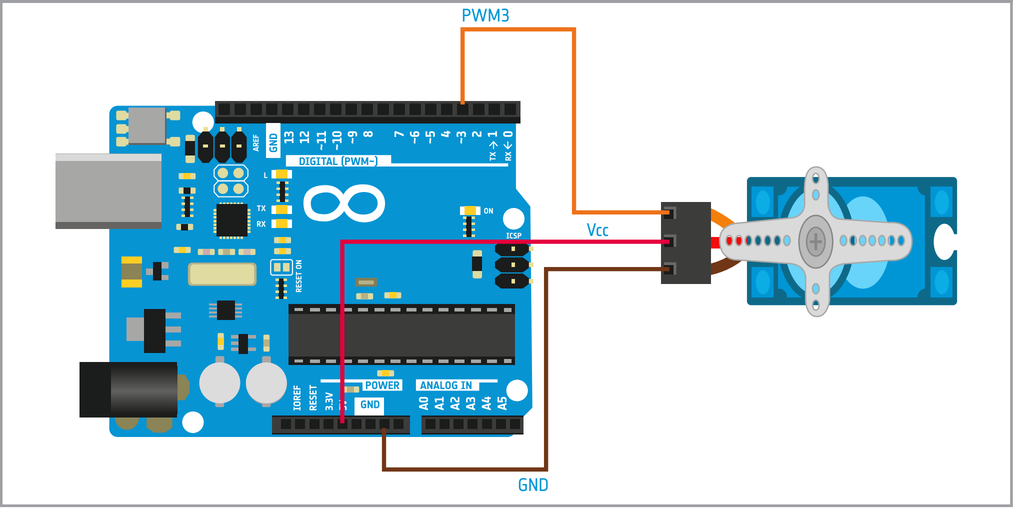
- Your water reservoir
- Poster tack
- A servo
- Cable ties
- Arduino
- Breadboard
- Wires

Step 1: Learn to program your servo motor

For this step you will need:
- A servo motor
- The Arduino
- A breadboard (optional)
- 3 male to male wires

You will need to use the wires to connect your servo to the Arduino Uno. You can either do this directly into the Arduino or plug 3 additional cables into the breadboard.

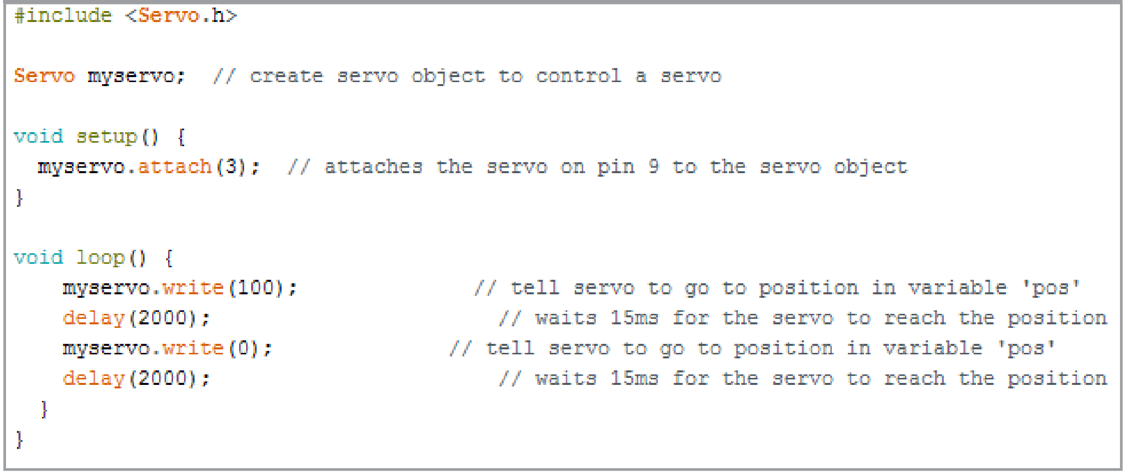
Figure A5



↑ The setup of the Arduino and servo

The Arduino IDE includes an example sketch named 'sweep'. This can be used to test the stability and the movement of the servo. The orientation can easily be altered by removing and rotating the helix. To test your servo, write a simple code like the one shown below. This code will rotate the servo by 100 degrees every two seconds.

Figure A6



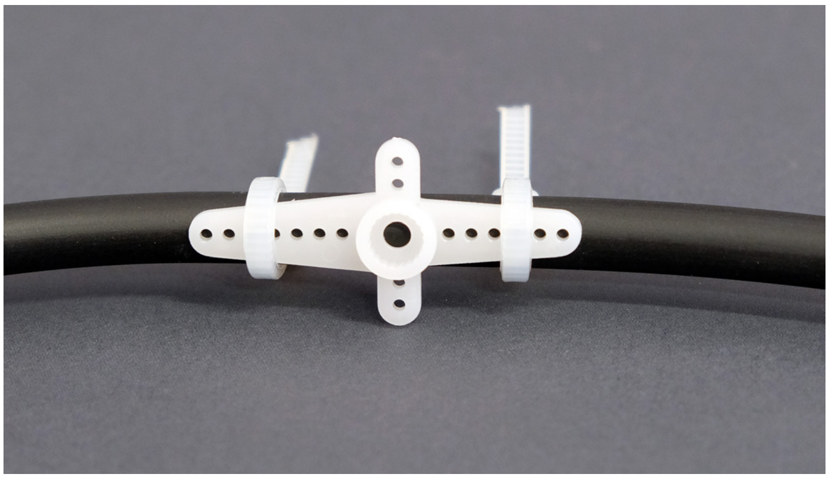
```
#include <Servo.h>

Servo myservo;  // create servo object to control a servo

void setup() {
  myservo.attach(3);  // attaches the servo on pin 9 to the servo object
}

void loop() {
    myservo.write(100);              // tell servo to go to position in variable 'pos'
    delay(2000);                       // waits 15ms for the servo to reach the position
    myservo.write(0);              // tell servo to go to position in variable 'pos'
    delay(2000);                       // waits 15ms for the servo to reach the position
  }
}
```

↑ Code to test the servo

Step 2: Motorize your system

For this step, we will need the additional items:

- Poster tack
- Your water reservoir
- A flat, vertical surface to which you can attach the servo
- Cable ties

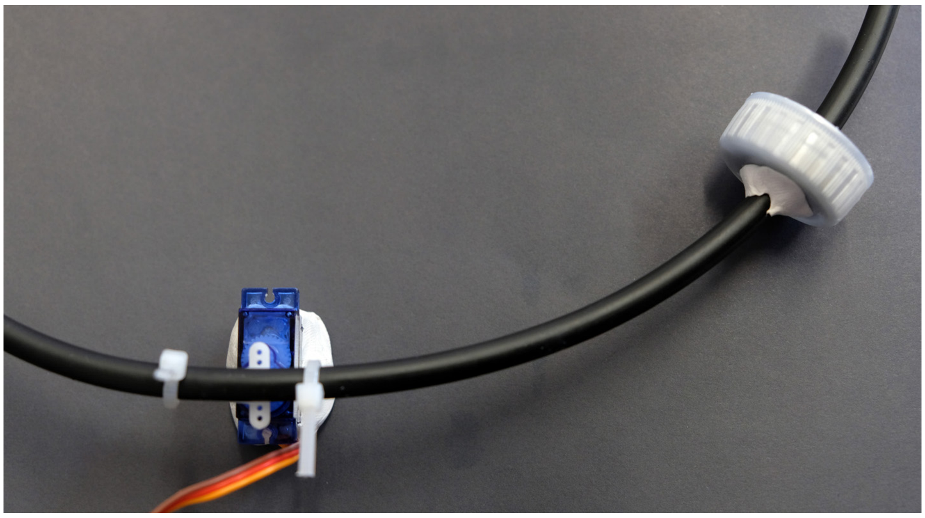


Now that we know how to program our servo, we can disconnect it from the Arduino so that we can place it in our new setup.

For this, we need to mount our servo and connect it to the water pipe. Poster tack can be used to mount the servo to a suitable wall. Again a glue gun would provide a more permanent fixture but it is likely that you will have to make adjustments to your setup during the early stages.

Included in most servo kits are a set of helixes. We will use one of these to connect the pipe to the servo, using two cable ties to secure it.

The length between the end of the pipe and the helix is important – the bend created by the servo must be sufficiently long to stop the water flow. If the bend is too small, water will continue to flow in the 'off' position – the Martian plants will not survive!



Now we are ready to fix the pipe to the servo. Simply click into place and we're almost ready!

Now we will connect our servo to the Arduino in order to establish whether or not the water can be stopped with our current set-up. Connect the servo to the Arduino as shown in the diagram below. Once the setup is complete you are ready to test.

Health & safety!

Before you begin make sure that:

- You have a bucket to collect any water
- Electronics and wires are a safe distance from the water bottle and any potential spillage

When finding the ideal location for your servo you will have to find the right balance between:
- The height of the water bottle
- The height of the servo
- The position of the helix on the pipe
- The orientation of the helix on the servo
- The degrees of rotation used in the code between the 'on' and 'off' position

Make a note of any adjustment you made to your system as a result of your testing:

→ ACTIVITY 4: TEST THE MOISTURE SENSOR

Introduction

Now that we have the first half of our system working, it is time to test the soil moisture sensor and see how we will incorporate it into our system. The two 'legs' of the moisture sensor work as a variable resistor. The more water in the soil, the better the conductivity, and vice versa. We will use this principle in order to completely automate our system.

Exercise

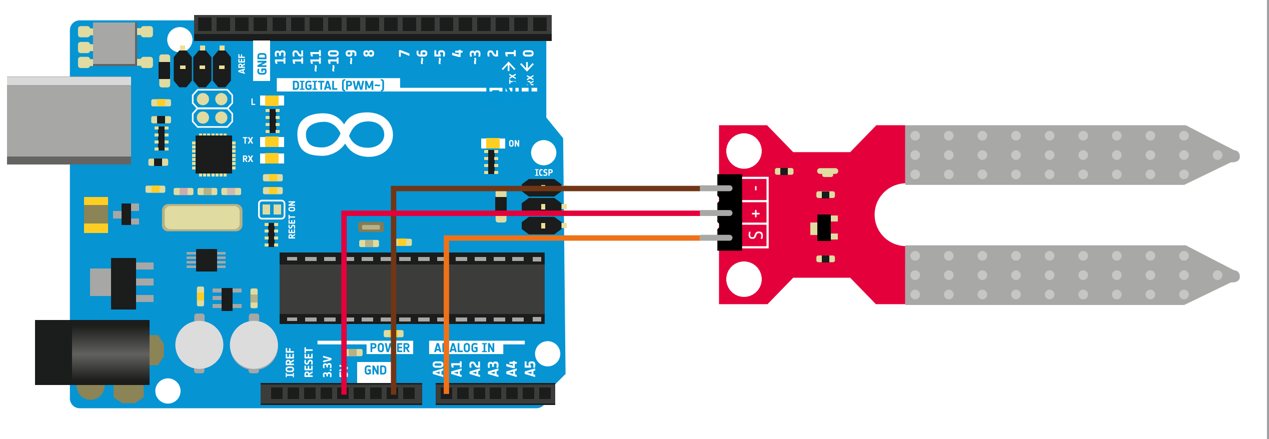
For this activity, you will need:
- Soil moisture sensor
- Male-Female wires
- Breadboard
- Arduino

Note:

Some sensors are calibrated to give a higher reading at higher conductivity, whereas others give a lower reading. To understand which sensor you have, compare a reading in the air to one in the water – do not fully submerge the sensor!

To test your moisture sensor, connect it to your Arduino like shown in the diagram.

Figure A7

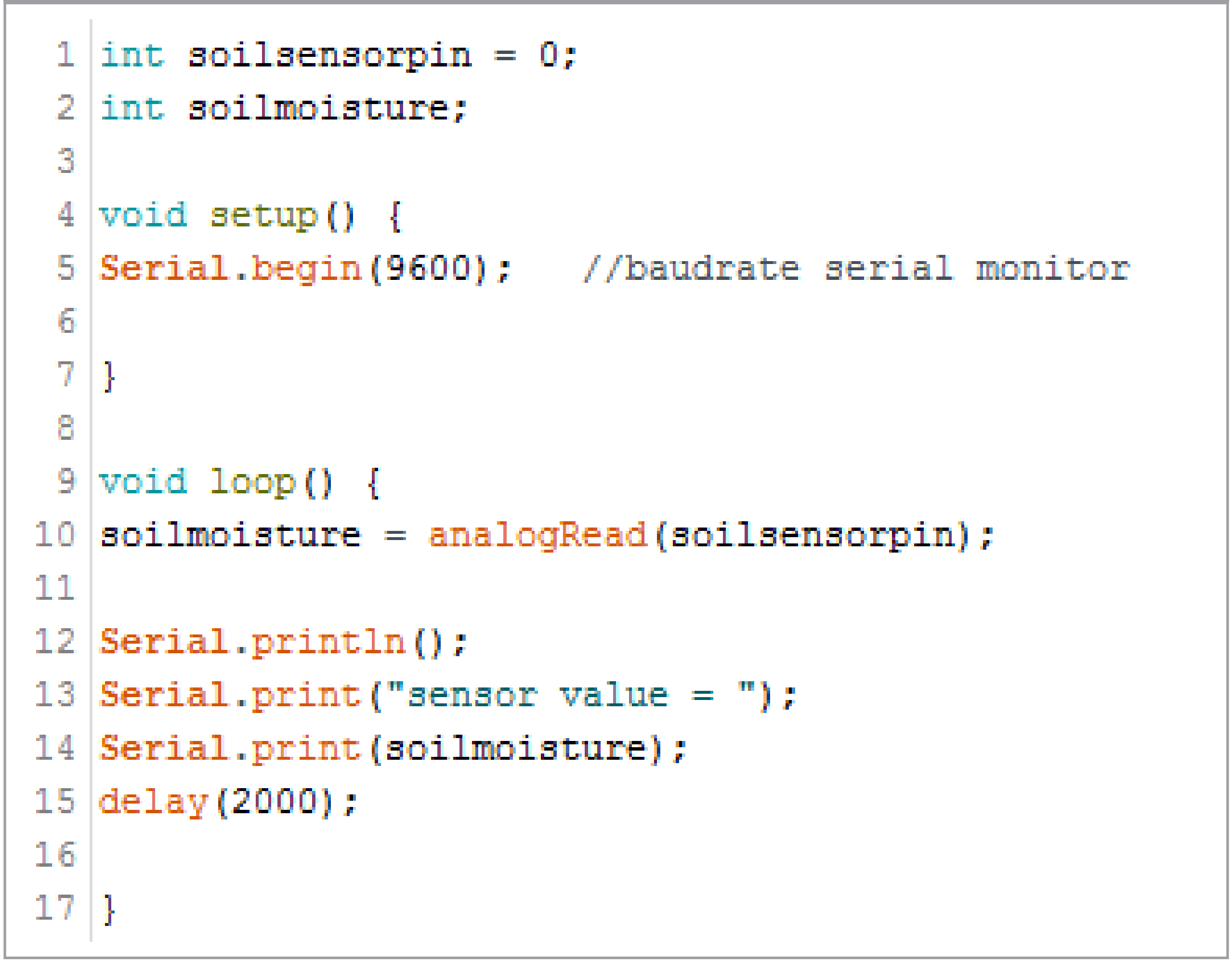


↑ The configuration for a soil moisture sensor without an external controller board

We are now ready to write a simple code to measure and display the value of the moisture sensor.

The code above takes a reading every second and prints the value in the serial monitor. Use this code to test your moisture sensor and to calibrate your watering system.

Figure A8



```
int soilsensorpin = 0;
int soilmoisture;

void setup() {
Serial.begin(9600);    //baudrate serial monitor

}

void loop() {
soilmoisture = analogRead(soilsensorpin);

Serial.println();
Serial.print("sensor value = ");
Serial.print(soilmoisture);
delay(2000);

}
```

↑ Code to calibrate the moisture sensor

Exercise

- What value does the sensor give when placed in water? ______________________
- What value does the sensor give in 'dry' air? ______________________
- What would be a suitable value to switch your system from 'on' to 'off'?

____

Once you are happy with how the sensor works, you are ready to complete the setup!

→ ACTIVITY 5: CONNECTING ALL THE COMPONENTS

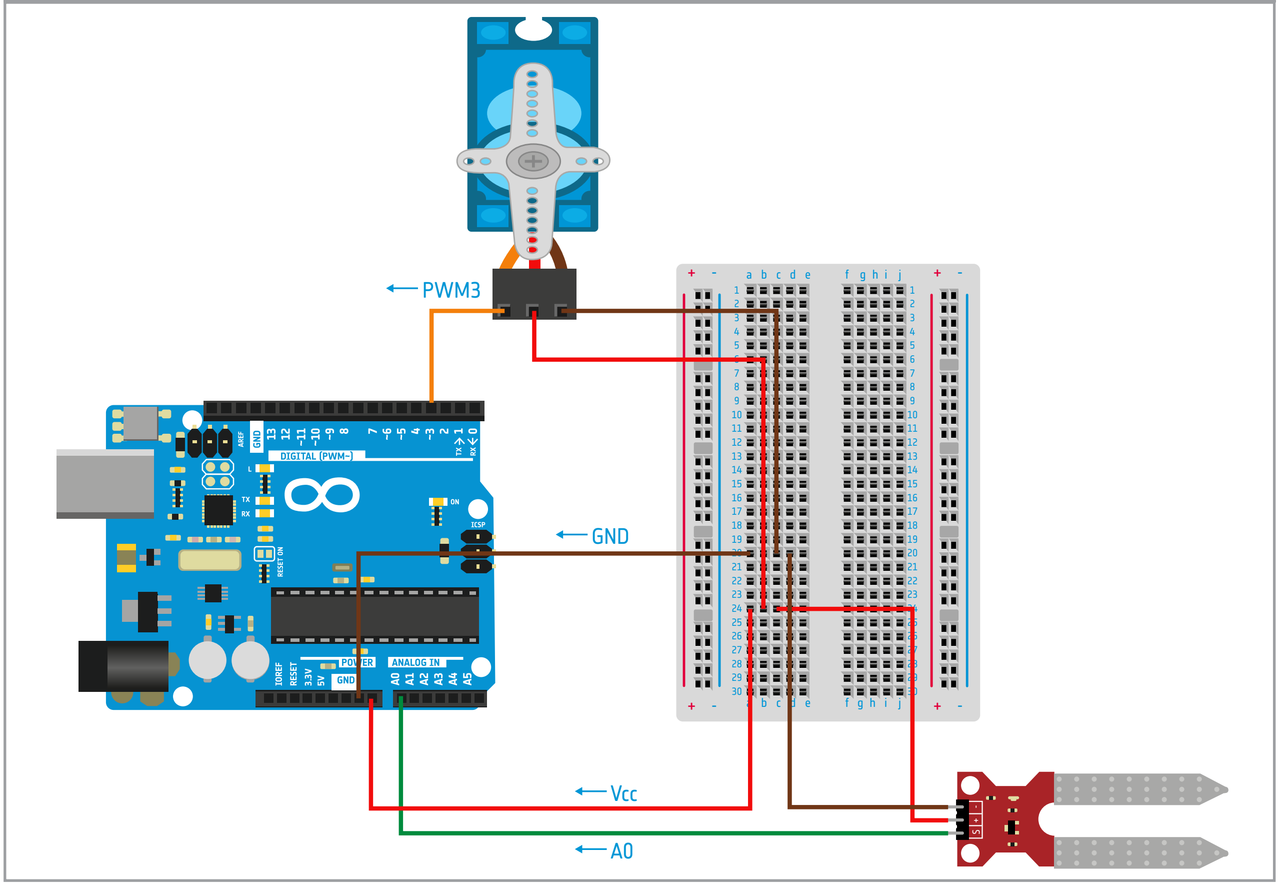
Introduction

Almost there! We now have a good understanding of all the elements of our system. Now it is time to put them all together and test the system to see if it all works.

Exercise

Once you are happy with each element of your water system you are ready to complete the setup by connecting the components to the Arduino. After following the earlier guides this should be straightforward. A diagram of the entire setup is shown below. Take care! Depending on the moisture sensor you use, the arrangement of the pins may differ. Always refer to the manufacturer's data sheet if you have any concerns.

Figure A9



↑ Electrical connections between the Arduino, servo, and soil moisture sensor

The wires on the servo are colour coded as follows: brown – ground, red – voltage in (Vcc), orange – 'pulse'. Take note of the four digital pins on the Arduino board that have ~ next to their number (3, 9, 10 and 11). This symbol denotes that the pin is a pulse-width modulation pin. If you're interested in what this means, more information can be found here. It is important to us as this is the type of pin required by the servo.

→ ACTIVITY 6: PROGRAM YOUR SYSTEM

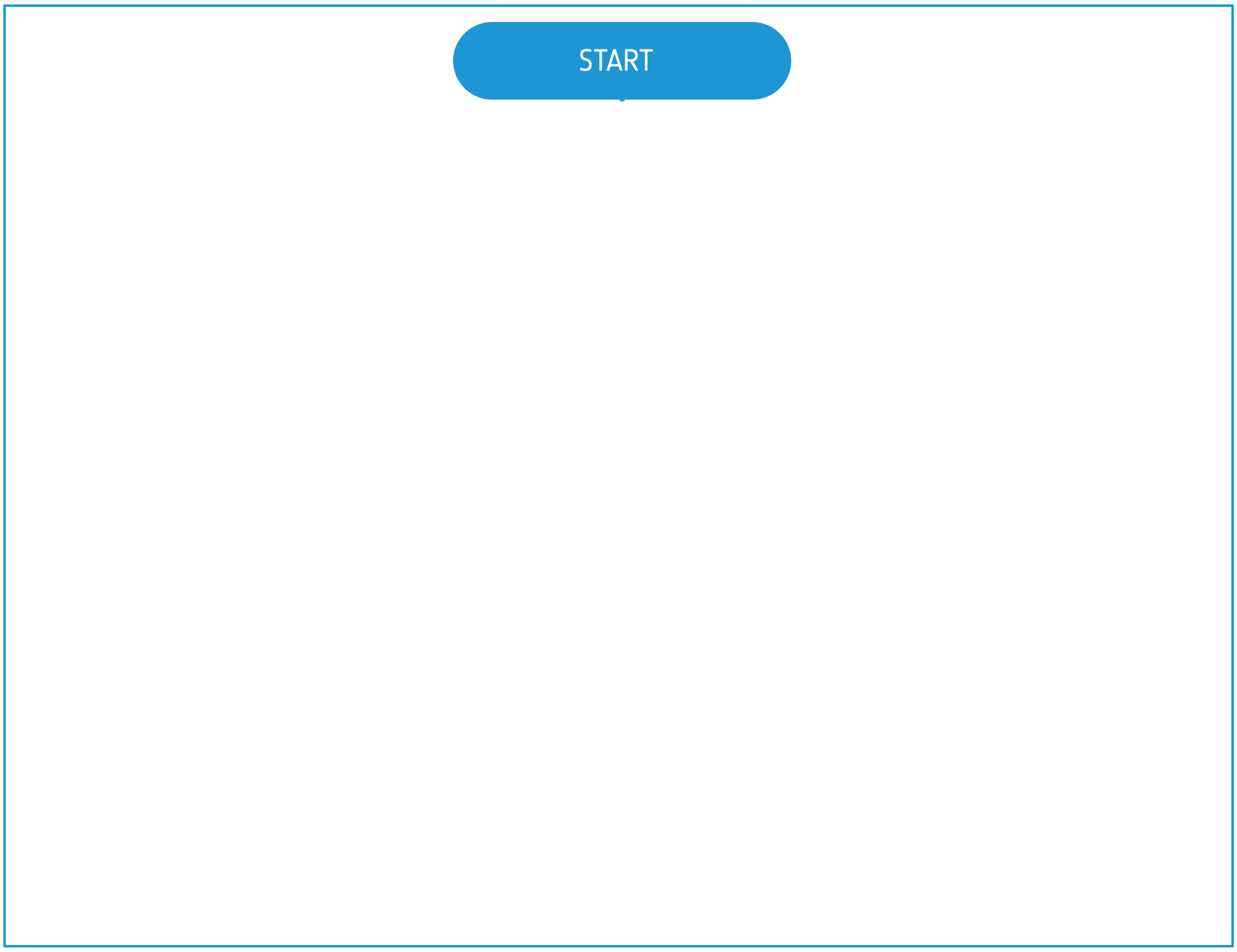
Introduction

We have tested each element of the system separately, both mechanically and using code. In the previous activity we assembled the system. Now it's time to write the single code that will be used to run the entire system! It is not as daunting as it sounds. We have already done most of the work, it's just time to put it all together.

Exercise

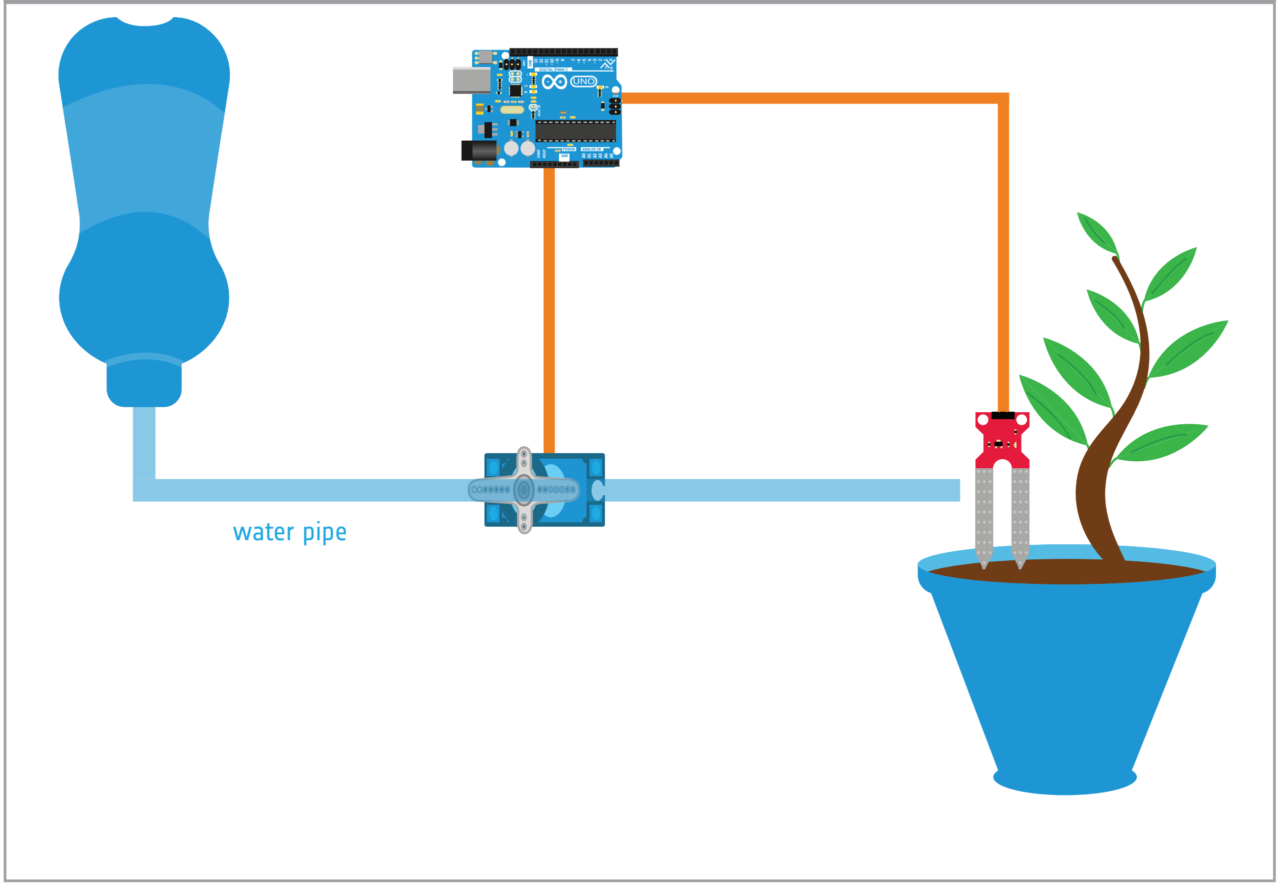
To make writing our code easier, we can think through the problem on paper first, using the same logic that you will use when you come to write the code. This is easily done in a flow chart. Typically, a rectangle in a flow chart is a command, and a diamond is a question/decision. Arrows are used to show the path through the chart, depending on the decisions made.

1) Have a go at writing your 'code' in the form of a flow chart below:



We now have everything in place to start running our automated plant watering system. All that is left is to write the code and send it to our Arduino.

Figure A10



↑ The chosen setup. The servo is used to move the water pipe into the 'off' position shown by the dotted line.

The figure above gives a simple overview of the system. In the 'on' position, shown by the solid blue line, gravity allows water to flow into the plant. In the 'off' positon, the servo rotates the water tube so that a u-bend is created, stopping water flow.

The first thing we must do in our code is establish several quantities/variables including: the pin used for the soil sensor, the pin used for the servo, a variable to store the sensor reading, and both an on and off position for the servo. All of this is done via the 'int' command. We must also define a name for our servo and ensure that the Servo library is called into the code.

Figure A11

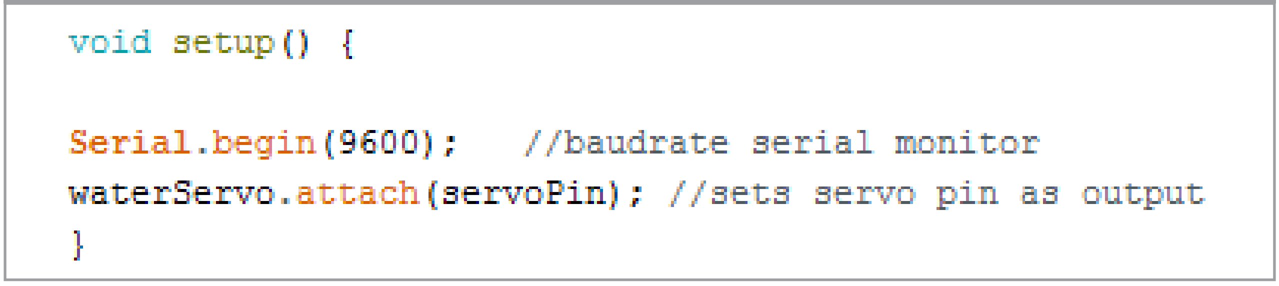
```
#include <Servo.h>
Servo waterServo;  //creates the name of your servo
int soilsensorpin = 0;  //assignes a pin for the soil sensor
int servoPin = 3; //sets the servo pin, this must be a PWM pin
int soilmoisture;    //variable to store one sensor reading
int wateringOn = 0; //position of servo to allow water to flow
int wateringOff = 120; //position of servo to hold water
```

↑ Establishing variables in our code

Take note, the servo position is given in degrees and the values you will need to use will depend on the orientation of your servo/helix, and will likely be different from the numbers you see above.

Next, as in earlier examples, we must establish a baud rate (closely linked to the bit-rate, the rate at which information/data flows). We must also tell the Arduino that the servo is an output.

Figure A12



```
void setup() {

Serial.begin(9600);    //baudrate serial monitor
waterServo.attach(servoPin); //sets servo pin as output
}
```

↑ Setting up the program

Now we are ready to begin writing our main program. We can see, looking at the flow chart, that the Arduino must measure a variable (in this case 'soilmoisture') and do one of two things depending on its value. In C++ this situation can easily be dealt with by using an 'if, else' statement.

The syntax of an 'if, else' is very simple, and can be seen below:

```
if(boolean_expression) {
// statement(s) will execute if the boolean expression is true
} else {
// statement(s) will execute if the boolean expression is false
}
```

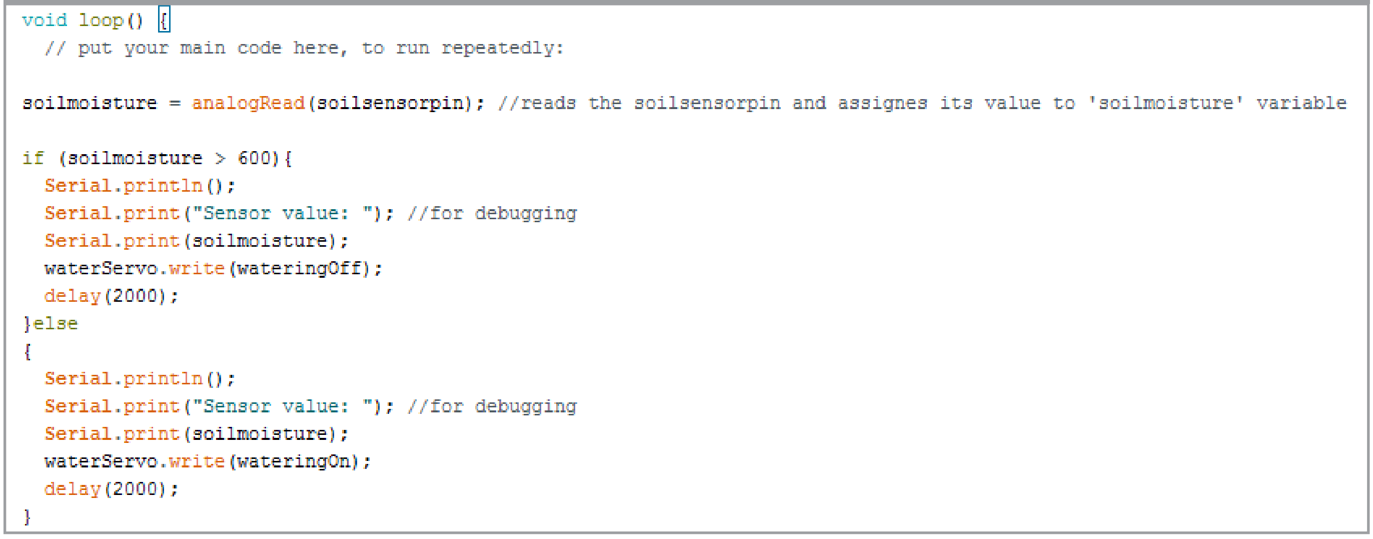
A 'boolean expression' is a mathematical statement that is either true or false and often includes the use of <, > and/or = symbols.

2) Using the variables we defined earlier, have a go at writing your own if, else statement using correct syntax in the space below. You may also want to consider printing the value of the soil moisture in the serial monitor (use the earlier guide if you have forgotten how to do this).

```
if(                        ) {




} else
{
```

A completed example is given below. Remember that there are many different ways to approach every problem in programming, so don't worry if your code does not look the same (but remember, it must complete).

Figure A13



```
void loop() {
  // put your main code here, to run repeatedly:

soilmoisture = analogRead(soilsensorpin); //reads the soilsensorpin and assignes its value to 'soilmoisture' variable

if (soilmoisture > 600){
  Serial.println();
  Serial.print("Sensor value: "); //for debugging
  Serial.print(soilmoisture);
  waterServo.write(wateringOff);
  delay(2000);
}else
{
  Serial.println();
  Serial.print("Sensor value: "); //for debugging
  Serial.print(soilmoisture);
  waterServo.write(wateringOn);
  delay(2000);
}
```

↑ A completed if, else statement

The if, else statement goes in the main loop. The program first reads the value from the soil sensor before using it in the if, else statement. In the example above a value of greater than '500' was taken as the soil being sufficiently moist. This value will vary from plant to plant, depending on the plant's needs. The command 'waterServo.write(wateringOn/Off)' is used to turn the servo motor to the required position. Notice the delay at the end of each loop. This ensures that the plant is watered.

You are now ready to run your program! It is a good idea to first test without any water in the system – water and electronics do not mix well and the program might not run exactly as you anticipate!

Whilst the code we have designed does what it is supposed to do, there are many ways it could be improved. For example:

3) Write down any improvements in the space below – could you incorporate it into your code?

Did you know?

Annual worldwide water consumption from irrigation is set to break 1500 cubic kilometres by 2025! It is therefore vital that any irrigation systems that are designed for a mission to Mars are as efficient and as self-sustaining as possible, otherwise the quantities of water required alone will make the mission unfeasible....

→ ACTIVITY 7: WHAT NOW?

Introduction

Water is of course just one of the many vital resources a plant needs to survive. How could the system be developed to be an inclusive, autonomous system capable of monitoring and keeping plants healthy in a Martian environment? Is there anything unique about the environment on Mars that we need to take into account? Are there any other concerns with a mission to Mars? Remember, everything required for the system has to make the journey with the astronauts, so simple, lightweight solutions are best!

Exercise

1) Think of the changes that you would have to make to the system if you were on Mars. Consider:
- Were there any anomaly readings? If so, how could you deal with these?
- Did the water flow only when it was required?
- What are the differences between Earth and Mars, and do any of these differences have consequences for our system?

2) Is it ethically sound to send Earth-based life to Mars? What if there is already life on Mars and this is irrevocably contaminated or killed? Divide the class in a 'for' and 'against' group, and create a list of arguments for why it is or why it isn't reasonable to send living organisms to Mars. Some starting points for each side of the argument have been given below.

For	Against
• We need to do it for humanity to survive • It could teach us about life on Earth • It could be contained • ...... •	• We could contaminate or kill already existing life • Radiation could result in unpredictable mutations in life • ...... • ......

teach with space – plants on mars | T09b
www.esa.int/education

The ESA Education Office welcomes feedback and comments
teachers@esa.int

An ESA Education production