

# Fun with Fusion Robots for Beginners

## Converting from Blockly to Python

Teacher/Mentor Web Edition 3.3  
(Updated by Graeme – August 2023)

Authors

[Ying Chen](#), [Yaya Lu](#) and [Graeme Faulkner](#)

<https://www.DrG2.com>

Copyright [CC BY 4.0](#)

# Introduction and Overview



This is a “first look” at Boxlight’s Fusion Robot, and while it has less detail than our usual “Absolute Beginners” courses, we aim to include sufficient detail to enable use either within a classroom, or by a home-study parent/child combination. For this audience, we emphasize detailed “how-to” videos, and minimize “talking-head” presentations. We introduce new concepts only when you need them to continue learning about your Fusion Robot.

What does this course cover?

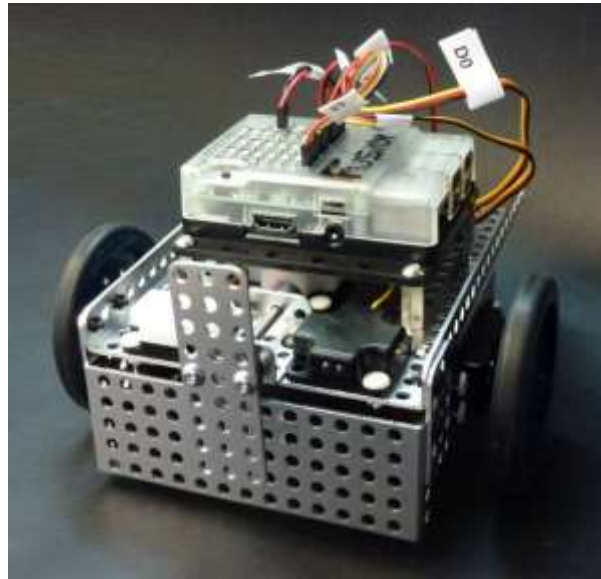
- YES: This introductory course *does* cover the conversion from the computer language Blockly to Python using the Fusion Robot Base Kit. We also check out some sensors that are available for use with Fusion, but which are not included in the Basic Fusion Base Kit; these tutorials are labeled “Extras”.
- NO: This course *does not* cover either the complete use of Python, or the use of the many impressive libraries that are available for use with Python. It is also *not* an introduction to Blockly, (to see our Blockly course [click here](#)).
- with Fusion.

Does this course require Internet access?

- YES: The videos used in this course are stored on YouTube, so you will need Internet access to use them.

Enjoy! 😊

# Building your Fusion Robot



## ***Unpacking Fusion's Mailing Box.***

One of the first things to do is to check the contents of the mailing box you received.

Video – [Web](#)

In this video we go through the components of our mailing box, to see if they are the same as those listed on Boxlight' *Fusion Base Bot Building Instructions* pdf that you can see by [clicking here](#).

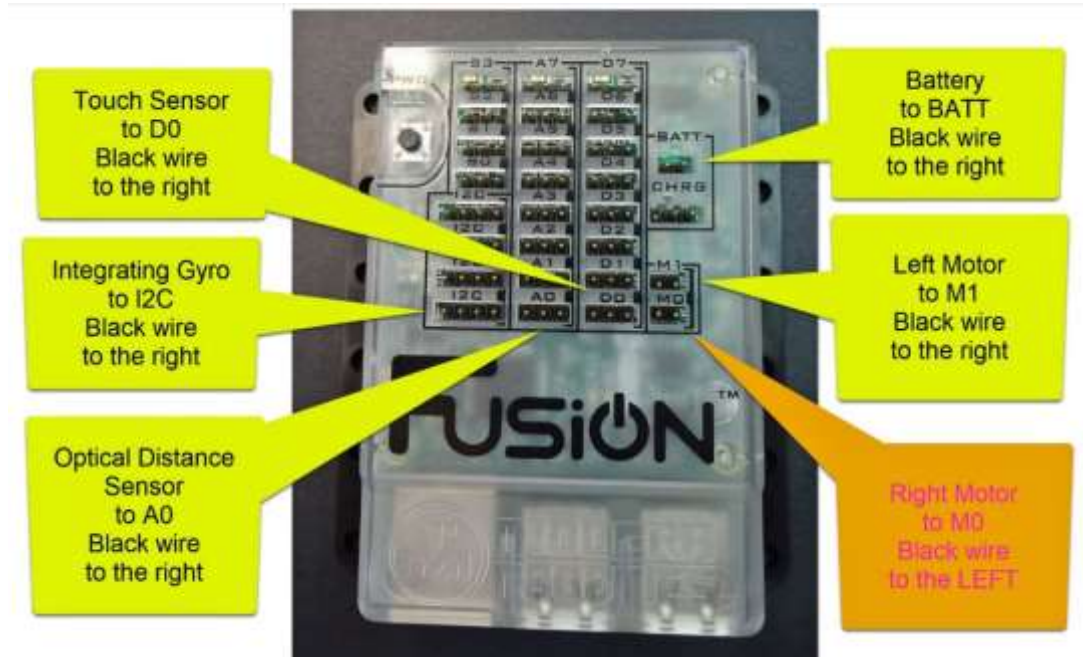
Do not throw away the mailing box. You can use it again later, to store your finished Fusion Robot.

## ***Use the Printed Build Instructions for Fusion***

Boxlight Robotics provides on-line printable instructions for building your Fusion Robot, [click here to find them](#).

If you prefer to build your Fusion Robot from a paper copy of these build instructions, for most browsers I have tried (Chrome, FireFox, Opera, Brave, Bing) there will be a little printer image on the “build instructions” screen on which you can left-mouse-click to print these instructions.

**Checking the wiring connections:** Next carefully check the connections of the cables to the Fusion controller, as indicated by the diagram on the next page.



### ***Now Charge Fusion's Battery Pack.***

It is unlikely that there will be sufficient power in Fusion's battery pack to allow you to experiment with your Robot. Turn Fusion off, because the battery will not charge when Fusion is turned on. Connect the charger supplied as part of your Fusion kit in to your building's power supply, and plug the other end in to Fusion's slot CHRG, with the black wire to the right to charge Fusion's battery. If the battery is flat, the light on the charger will glow red. It will take an hour or two to charge Fusion's battery pack. When the battery is charged, the light on the charger will change to a green glow.

***NOTE: Make sure Fusion is turned off.*** If Fusion is turned on, and you plug in the charger cable, the light built in to the charger will glow green (erroneously indicating a full battery) when in fact the battery could be almost flat. *Your Fusion's battery will only be being charged when Fusion is turned off.*

### ***Congratulations!***

You have now assembled your Fusion robot. In the next lesson you will find out how to connect your laptop to your Fusion robot using Wi-Fi. Let's go...

# How To Connect your Computer To Your Fusion Robot Using Fusion's Wi-Fi

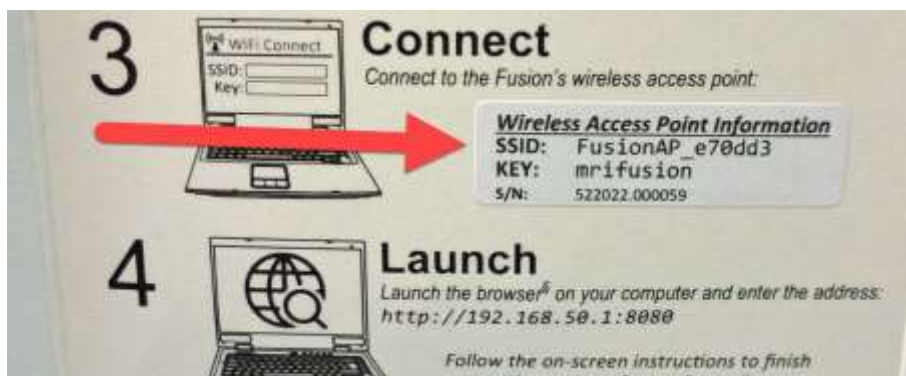
The first time you connect your computer to your Fusion Robot, you may be required to provide a security key. This security key will be supplied by Boxlight as part of your Fusion kit. At the time of writing this key is `mrifusion`. It can be seen inside the lid of the Fusion System Controller Box that comes with your Fusion kit.



The video below shows how to use Fusion's inbuilt Wi-Fi to link a Fusion Robot to a computer. In this example, the computer is a Windows 10 PC.

Video – [Web](#)

Modern Robotics also have information about [connecting your computer to your Fusion Robot](#). You may need the SSID of your Fusion Robot, which can be found here.



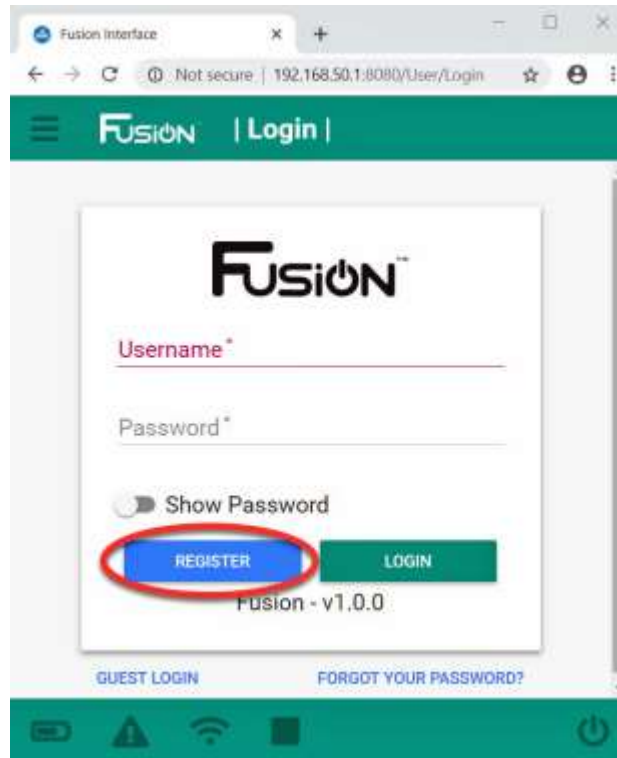
One of the unique things about a Fusion robot is that, as well as acting as a robot, it also acts as a server with an inbuilt Wi-Fi link. Laptops, desktop PCs, tablets or smart phones can connect to the Fusion robot using this Wi-Fi link. It is almost as if the Fusion robot is its own Wi-Fi hotspot, except that the Wi-Fi connection is not to the

Internet, it is just to the Fusion robot itself. In normal operation, neither the Fusion robot nor the computer need be connected to the Internet, (but can be if the student *really* desires this to be the case).

This ability to be able to operate without a connection to the Internet can be a major advantage when the Internet connection is a problem, either because of slow Internet speed or because the Internet is simply not available in the student workplace. It can also be an advantage when a school is concerned that the student may have contact with some of the less desirable content on the Internet, and therefore has banned connections between student computers in the school, and the Internet. In both these circumstances, the Fusion robot can be used and programmed independently of any Internet access. If you have access to the USB version of our tutorials, these can also be used independently of any Internet access.



# Setting up Admin and User Accounts in your Fusion Robot

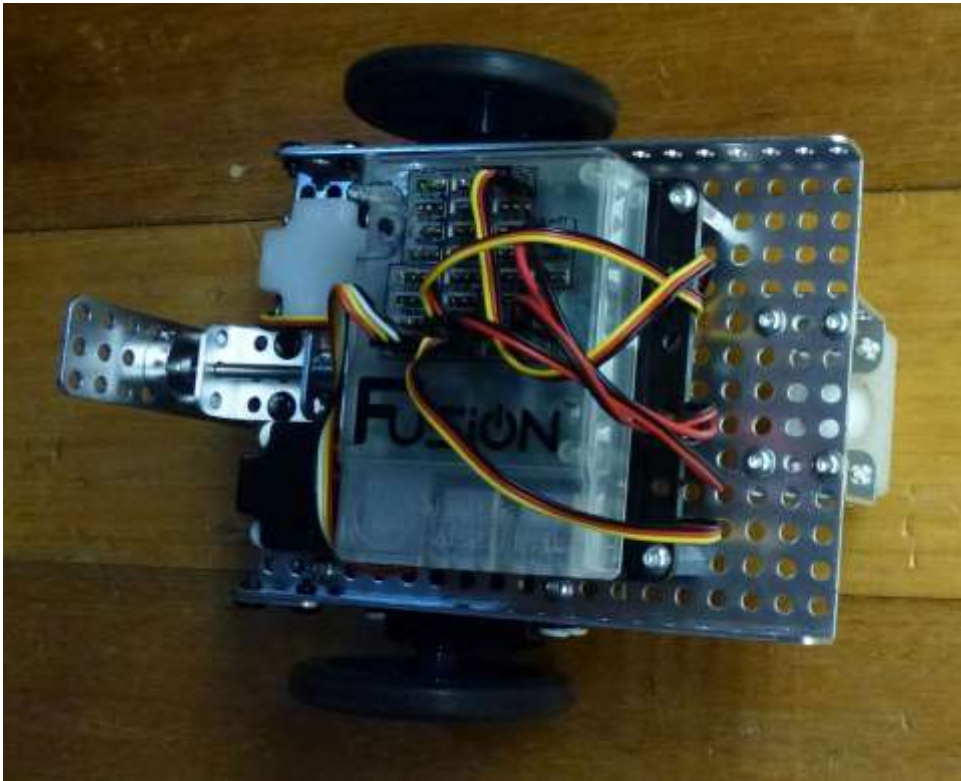


One of the advantages of the Fusion Robot is that multiple accounts can be set up inside the one Fusion Robot. Thus, students in many classes can all (one at a time) use the same Robot, each student saving their programs inside the Fusion Robot. Students are subsequently able to login in to Fusion again later, and continue their work, starting from where they left off. This makes it very easy to use a Fusion Robot across multiple classes.

However, before Fusion can be used by multiple users, accounts within the Fusion Robot must first be set up. There are two types of Fusion accounts, “Admin” and “User”. When used in a classroom, usually only the teacher or mentor will have access to the more powerful “Admin” account. Students would usually be allocated a “User” account. The video below demonstrates how to set up both types of accounts within a Fusion Robot.

Video - [Web](#)

**Note:** Some of the button colors in this tutorial may not match the button colors you see when you set up your accounts in your Fusion robot. This difference has occurred because Modern Robotics has updated their Fusion operating system since this video was produced. The method demonstrated in this video will still work well with the new version of Fusion's operating system.



### ***Converting from Blockly to Python:***

In this tutorial we will look at using the simplest possible Blockly program to move your Robot forwards, and then stop. We will then compare this Blockly program with the automatically generated Python equivalent to this Blockly program, and show how to run Python separately using Fusion's Python Editor.

It is assumed that a Wi-Fi link between your computer and your Fusion robot has already been set up, and that you have set up User and Admin accounts within your Fusion robot (if you have not already completed these items, [look at our previous tutorial](#)).

In the video below, we start up a browser (theoretically any browser, but Modern Robotics recommend Google Chrome, so that is what we will use). To teach our Fusion Robot to move, we send it commands in both the computer languages Blockly and Python.

Video – [Web](#)

Blockly is excellent if students have not used computers & robots before. It can be used in introductory courses in Schools from lower middle school levels and above. Python is an excellent computer language for more advanced work, and is appropriate for upper middle school students and above. If you are interested in more detailed information about the Python language itself, go to page 23.



## ***Some Blockly to Python Equivalents:***

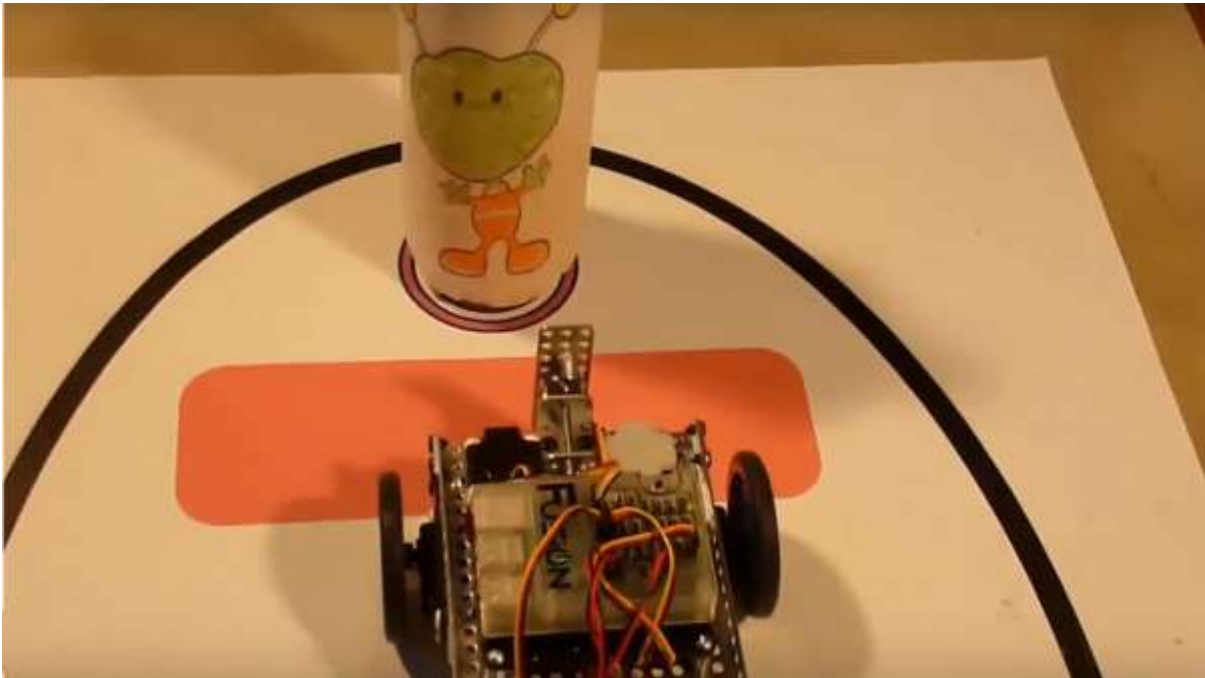
We spoke to potential users. Those who already has some skills in Blockly requested a “quick reference” list of Fusion's Python equivalents to Fusion's Basic Blockly commands, to assist them during the transition phase of their language conversion. Because a single Blockly command can group multiple Python commands into neat, useful "clusters", this list could also be useful to any new Python users who have not yet had any coding/programming experience using a text-based language.

You can download a 2-page pdf Blockly/Python equivalents of our attempt to meet this request by [clicking here](#) . This list of equivalents applies to the standard build of the Fusion Robot Base Kit. If you are using a different build, with (e.g.) motors in different positions, you may have to adjust these equivalents so that they apply to your different robot build. We comment on this “equivalents pdf” in the following video.

Video – [Web](#)

Next let us apply some of our new knowledge in the “Approaching An Alien” Challenge on the next page! 😊

# Fusion Approaches an Alien!



Let us pretend that an Alien Ambassador has come to Earth. We don't know whether the Alien is dangerous or not. We will send our Fusion Robot to approach the Alien instead of us, because if anything goes wrong it will be a Robot and not us that will be zapped. We need to teach Fusion to move to the “red carpet”, pause, and retreat (backwards - it is impolite to turn one’s back on an Alien Ambassador!) Do not let Fusion collide with the Alien Ambassador – that might start an intergalactic war!

## ***Reminder: How to Login to our Fusion Robot***

In case you need a reminder, watch the video below.

Video – [Web](#)

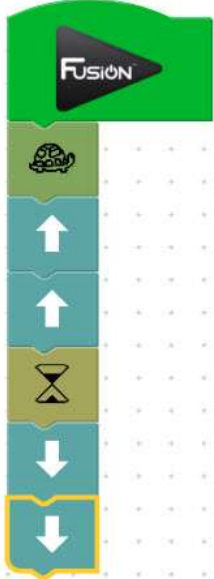
## ***Fusion Approaches An Alien***

We will initially program our robot in Blockly using Fusion's Blockly Editor. We will then take a look at the Python code that is automatically produced by our Blockly program. As an experiment, we will copy that Python code, leave the Blockly Editor, LAUNCH the Python Editor, and type in the identical Python code into this Python Editor. We will then run that Python code to confirm that it actually works... 😊 To check what we did, look at the video below.

Video – [Web](#)

The arena that the Fusion robot uses is available for download ([Web](#)). It can be printed if you have access to an A1 printer. A second similar arena, but using slightly different colors, is also available for download ([Web](#)) If you do not have a “monster alien” available (making some of these can be a fun and amusing Art class project), there are three printable A4 images ([Web](#)) available, any of which could be wrapped around a soft drink bottle to make a pretend “Alien”.

# Python or Blockly Code Best For You?



```
import Fusion
f = Fusion.driver()
import time

speed = 60

speed = 60
f.motorSpeed(f.M0+f.M1, speed)
time.sleep(1)
f.motorSpeed(f.M0+f.M1, 0)
f.motorSpeed(f.M0+f.M1, speed)
time.sleep(1)
f.motorSpeed(f.M0+f.M1, 0)
time.sleep(1)
f.motorSpeed(f.M0+f.M1, -speed)
time.sleep(1)
f.motorSpeed(f.M0+f.M1, 0)
f.motorSpeed(f.M0+f.M1, -speed)
time.sleep(1)
f.motorSpeed(f.M0+f.M1, 0)
```

```
import Fusion
f = Fusion.driver()
import time

speed = 60
f.motorSpeed(f.M0+f.M1, speed)
time.sleep(2)
f.motorSpeed(f.M0+f.M1, 0)
time.sleep(1)
f.motorSpeed(f.M0+f.M1, -speed)
time.sleep(2)
f.motorSpeed(f.M0+f.M1, 0)
```

Both Blockly and Python have advantages and disadvantages relative to each other. From the point of view of students, we could summarize these as follows:

## Blockly strengths:

- Relatively easier than Python for students to learn.
- Does not require really good typing skills
- Can be used down to Lower Middle School, (and below with capable students).
- Good for quick prototyping to see if a student project using a Fusion robot is feasible.

## Blockly Weaknesses:

- Inefficient in terms of the use of the Fusion's Raspberry Pi computer.
- Does not have easy access to pre-written libraries for use in advanced (pre-University & University) projects using Fusion.

## Python Strengths:

- Relatively efficient in terms of the use of the Fusion's Raspberry Pi computer.
- Has access to pre-written libraries for use in advanced (pre-University & University) projects using Fusion
- A comparatively good text-based coding language for Upper Middle School and pre-University students
- Arguably easier for school students to learn than competing languages (e.g. Java, C, C++)

## Python Weaknesses:

- Requires students to have reasonably good typing skills

- Much more difficult for students to debug (remove code errors) that is the case with Blockly.

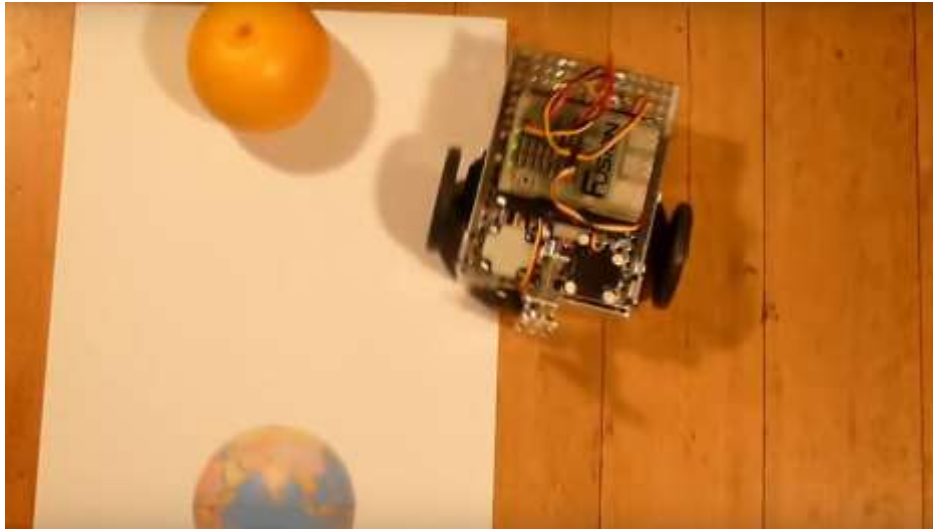
If you have suddenly decided that Blockly suits your needs better than Python, you can go to our Blockly course for Fusion Beginners by [clicking here](#).

If you are "keeping the faith" with Python, we will now take a critical look at the Python code automatically produced by the Blockly program that we used to teach our brave Robot to "Approach An Alien" in the previous tutorial.

### Video – [Web](#)

This is a tiny program, and any reduction in running time would be hardly noticeable. However, for large programs, the difference can be very significant. Many Engineers who have very tight deadlines for producing preliminary results, often start by looking at quick rough options like Blockly, and then transition to more efficient methods like Python when the project's final research/production direction has been decided, and efficient programs for use in Business are required. Which approach you need for your project is a decision that you are probably in the best position to make.

# Fusion Goes Around the Moon...



“Around the Moon” is a Challenge we have used with Middle School students. Fusion has to leave a pretend “Earth”, go into “space” around the “Moon”, and attempt to land back right on top of the pretend “Earth”.

## **Downloadable Arenas:**

If you have access to a printer, [an A3 sized “Moon/Earth” image](#), and [an A4 sized “Moon/Earth” image](#), are available for free download. You could also re-use the A1 printer arena that you may have already used in the “Alien” tutorial ([Web](#)).

## ***Fusion goes around the Moon – Blockly to Python***

We start this tutorial by developing a Blockly solution to this Challenge. We then convert the Blockly program into Python Code, and demonstrate pruning the automatically-produced Python equivalent of the Blockly program from 51 down to a more efficient 27 lines, to make a Challenge solution that would impose less computational load on Fusion’s inexpensive computer chip.

Video – [Web](#)

**Battery charge:** We have recorded these videos with recently charged batteries in our Fusion Robots. If the battery in your robot is low in charge, the distance travelled by your robot may vary from the examples shown in our videos.

## ***Fusion goes around the Moon – Methodology Change?***

As in the previous tutorial, we will send our Fusion Robot into “Space” around the (Tasmanian Apple ☺) “Moon”. Up until now, we have been using “turns” inherited from the original “Basic Blockly” commands. This meant we could only have three types of turns, 45 degrees, 90 degrees and 180 degrees. Python (and [Intermediate Blockly](#)) give us more versatility. Let us use this increased versatility of Python to investigate if a change in methodology would allow us to achieve an even more efficient “Around the Moon” program. The video below investigates this possibility.

Video – [Web](#)

## ***Extensions to the “Around the Moon” Challenge***

Why do we consider extensions to this Challenge?

When teaching robotics, right from the start with the **Tasman Turtle** in the early 1980s, through to other robots including LEGO NXTs and EV3s over the last decade, we observed our students exhibiting a tremendous range of abilities. This is why our tutorials are not of the “lock step” variety that occur so frequently, particularly in Secondary School classes. Our Challenges are carefully designed so that the slowest student in the class can achieve a successful result, and not feel left behind.

We have also tried to include “extensions” to the Challenge that will challenge faster students to continue learning. This has allowed our classes to come very close to not having any students left feeling a “failure” in our classes – every student succeeds at a level where they can be encouraged.

In this “Moon Challenge”, even the slowest student has (so far) concluded this Challenge with a robot ending up quite close, or perhaps with a tiny bit of the robot touching, some part of the “Earth”. This would be counted as a successful completion, with the student being legitimately congratulated on their good result.

With average students, we would try and encourage them to get their robot completely over the “Earth”, so that the Earth can not be seen when looking at the robot from above. This is quite a bit more difficult and requires more time, which is a suitable target for average students, while the slower students complete their less difficult version of this Challenge.

Gifted or talented students will have generally completed this Challenge faster than the average students in the class. To save them from getting bored and frustrated while the rest of the class catches up, we have “extensions” of these Challenges that will give them more difficult practice in what we are trying to teach them. Our experience over the roughly the last decade with LEGO NXT and EV3 robots, is that this “multiple-layer” approach will keep gifted students happily continuing to learn, while the slower students succeed in their less elaborate versions of the Challenge. “Multi-layer” seems to work well for our students.

### ***Fusion goes around the Moon - the Sufi Way!***

In the video below, we use what one iconoclastic and talented Grade 5 student called his “Sufi” approach to “Going Around the Moon”. He had read about the Sufi Whirling Dervish dancers who spin around during their dances ([see here](#)). He wanted his robot to spin around during its trip around the Moon, just like the Sufi dancers do. Just to make things more difficult, he had his robot turning in the opposite way (anti-clockwise looking from above) to the turns taken by virtually every other student in the class (clockwise looking from above) – and, to his credit, he succeeded.

The video below shows that his idea, which he applied to a LEGO EV3 robot, can also be achieved by our Fusion robot. Since this student was a boy, and boys very often want to use a speed of 100% for everything, this video also demonstrates what happens to our Fusion robot when it spins at a full motor speed of 100%. I’m not sure the student realized it, but this made his task much more difficult. Using higher speed “stops” and “starts” very often result in much wheelspin. If the floor is even the tiniest bit uneven, the use of high speeds resulting in wheelspin will make the robot’s final direction much less reliably predictable. This problem of high-speed wheelspin is the reason we suggest that most of the students run their robots at a speed of 40% to 50%



of full power during all testing runs, there being much less chance of wheelspin at these lower speeds. They can try higher speeds later when they are preparing for competition events!

The video below shows Fusion's version of a "Sufi" extension to the "Around the Moon" Challenge.

Video – [Web](#)

### ***Fusion Loops Around the Moon***

A second extension to the "Moon Challenge" is for the student to teach their robot to go in an orbit around the "Moon", before returning to "Earth".

At one of the Adult Education parent/child weekend robotics workshops run by Yaya and Graeme, one of the girls wanted to program her robot to have a good look at the Moon, and so she taught her Robot to do a loop around the Moon before returning back to Earth.

The video below demonstrates that a Fusion robot is capable of imitating this girl's EV3 robot solution to the Moon Challenge.

This Python program is a little different from the previous "Blockly Moon Challenge" programs. The previously used "Blockly Rotate" command spun our Fusion robot around on the spot. To go in a loop around the Moon, we will need to use separate motor commands for each motor. In our Python version, we have to add a separate "Time" command limit after the two motor commands. The motors will continue running until the "Time" command is finished. This is discussed in the video.

Video – [Web](#)

### ***Other "Moon Challenge" extensions for Fusion to Try?***

The Moon Challenge seems to have acted as a stimulus to student imagination. I particularly remember a boy student was extremely enthusiastic about his "super-doooper secret spy robot". It left Earth in the opposite direction to everyone else's robots, heading away from the Moon. His reasoning was that anyone on Earth would think his robot would be going to Mars or Venus. What his robot was actually going to do, he said, was to go around the Moon and spy on a secret enemy base on the far side of the Moon, an enemy base which we could not see from Earth. He knew that people on Earth only ever see about one half of the Moon, and that no one on Earth can see anything on the far side of the Moon when they are standing on the Earth. This is not widely known, but he knew it! He then taught his Robot to return past the Earth, turn around in Space, and then come back to the Earth from his Mars or Venus direction, so that no-one on Earth would know where he had been spying! He was very proud of his super-doooper secret spy robot, and he completed his version of this Challenge quite satisfactorily – to the applause of the other students...

How many ways can you teach your Fusion Robot to "Go around the Moon"? Perhaps inspiration can be gained from these old videos of previous student attempts (using old LEGO robots). The runs vary from wonderful to ~~disastrous~~ "innovative". 😊

Video - [Web](#)

Video - [Web](#)

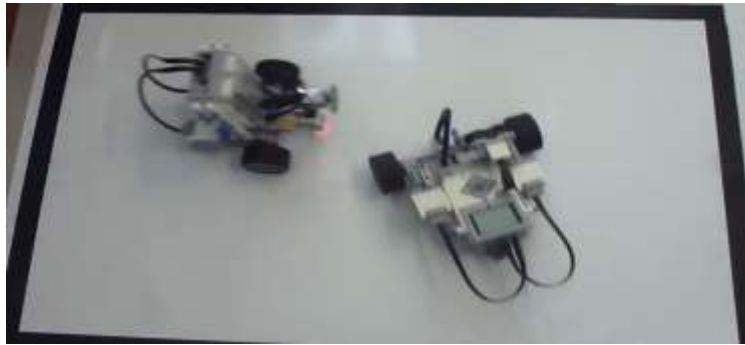
## Robot SUMO using Fusion



Japan has a sport called SUMO in which each person attempts to push the other out of an arena.



We would not have much chance of competing! But we can have fun with a robot version of SUMO, in which Fusion attempts either to push a toy, or another robot, out of the pretend SUMO rink.



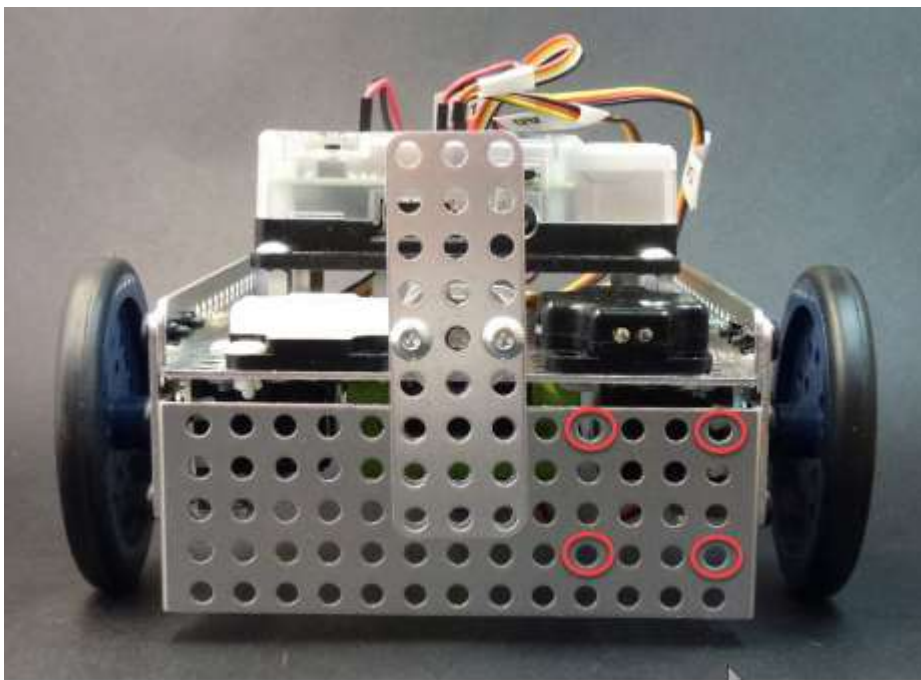
You can either [make your own arena](#), or if you have access to a shop with an A1 printer, you could download [our free SUMO pdf link](#).

### ***How to use Fusion's Optical Distance Sensor***

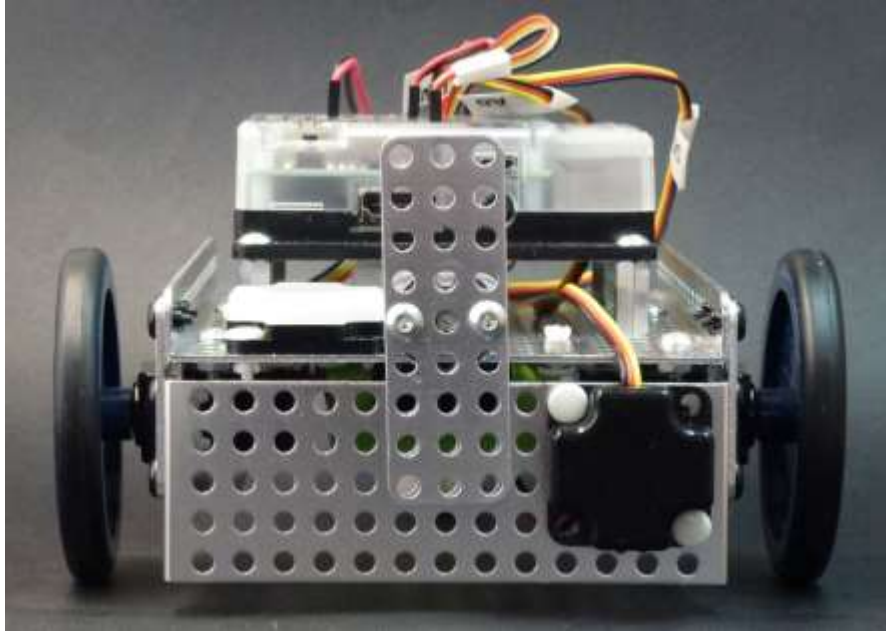
To enable our robot to play SUMO, we need to teach our Fusion Robot how to sense the edge of the SUMO arena. To detect the black line that surrounds the arena, we need to find out how to use Fusion's Optical Distance sensor.

To play SUMO, Fusion needs to be able to tell when it is somewhere inside the arena, and when it has found the thick black line surrounding the SUMO rink. We will use the Optical Distance Sensor to let Fusion know which part of the rink it is in.

First move the black Optical Distance Sensor to the front of the Fusion Robot, using the hole positions circled above.



The result of this move is shown on the next page.



This video demonstrates the use of Fusion's Optical Distance Sensor to detect whether the Sensor (and the hence the robot) is in the white center of the SUMO rink, or if it has found the black edge of that rink. The difference in sensor readings obtained from the Optical Distance Sensor when it is over the white area, and when it is over the black area of the rink, can later be used to teach Fusion how to stay inside the SUMO rink.

Video – [Web](#)

### ***Keeping Fusion inside the SUMO Rink***

Our Fusion Robot can play SUMO by:

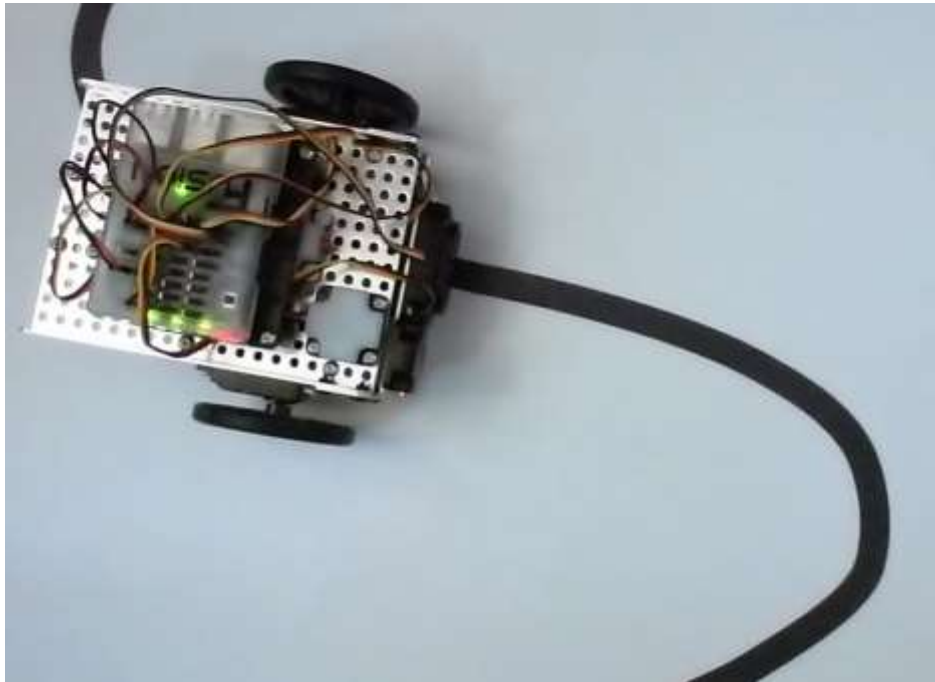
1. Charging forwards, when Fusion is in the white middle of the rink,
2. Backing away and turning, after Fusion reaches the black edge of the rink.

Previously in our Blockly beginner's course, we showed how to play SUMO using Fusion's Blockly code. In this video we demonstrate how to use Fusion's automatic generation of Python code to convert the Blockly program that was used in our Fusion Blockly SUMO tutorial, into the more advanced computer language Python. We combine this Python code with Optical Distance Sensor readings to help our Fusion Robot detect when it has left the white centre of the rink, and reached the black edge. Watch the video below to find out how...

Video – [Web](#)

The video above is our “first look” at using Python to teach our Fusion Robot to stay inside the SUMO rink. Have a play with this Python code. Change the motor speeds. Change the turning timings. See if you can make your Robot play better SUMO.

## Fusion Follows a Line Edge



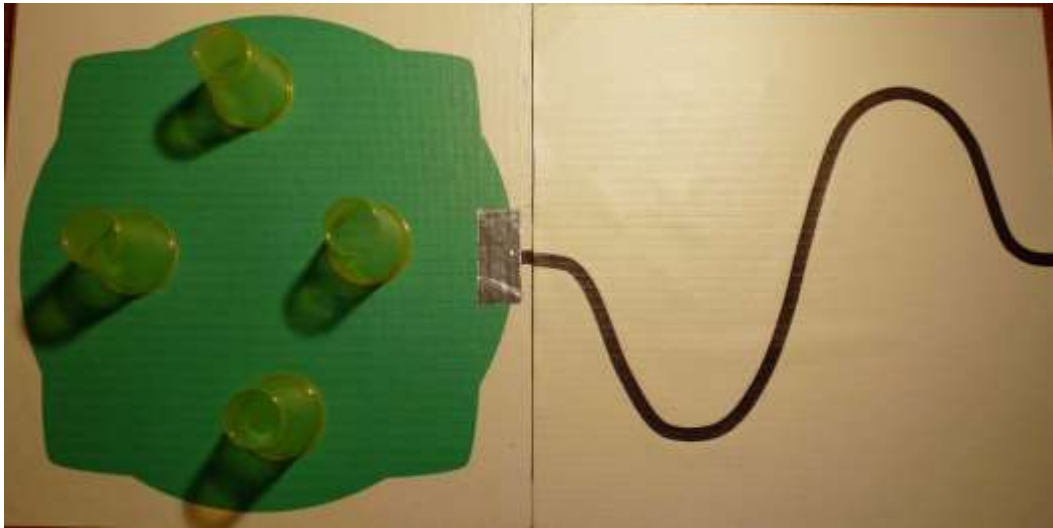
Fusion can be taught to follow the edge of a line by using Fusion's Optical Distance Sensor.

Next we look at a Blockly "Edge Follow" program, similar to the ones discussed in our Blockly tutorials here [Web](#), and [Web](#). We then convert it into a Python program, as shown in the following video.

Video – [Web](#)



# Fusion's Swimming Pool Challenge



## ***Can our Fusion robot clear the toys from the green swimming pool?***

Imagine that our class has gone on a bus trip to a beautiful swimming pool in the wilderness. We have only been playing for a little time when there is a sudden heavy shower of rain. The driver of the bus is concerned that the shallow stream the bus crossed may suddenly get deeper, so he hurries all the students into the bus and quickly leaves before the stream can rise. This leaves the teacher to retrieve all the toys with which the class was playing. Unfortunately, the teacher can not swim. So, we need to program our special amphibious Fusion robot to follow a winding trail through the wilderness, to detect the silvery white sands of a beach at the edge of the pool, and to then push all of the toys to the edge of the pool so that teacher can collect them without getting wet feet. 😊

How can we teach our Fusion robot to do that?

If we think about this Challenge, we could re-use our previous code, like this:

- Fusion uses our previous code to follow the line until it “sees” the silver sand.
- If Fusion “sees” the silver sand, then it stops edge-following, and then uses our previous SUMO code to push the toys to the edge of the pool, ready for the teacher to collect the toys. 😊

The next tutorials show how to re-use our previous code to solve this Challenge.

## ***Changing Fusion's “Edge-Follow” Code into a Function***

One of the problems with computer code is that if the code is many pages long, it is often difficult to remember on (e.g.) page 54 of our code, what we have done on (e.g.) page 17 of our code. Long experience gathered over decades has shown that even expert programmers produce more reliable code if their code is divided into small chunks that can fit on one computer screen. We call these code chunks “functions”. Let us see



if we can change our “edge-follow” code into a code chunk that will fit on one screen page.

The video below shows how to change our "edge-follow code" into an "edge-follow function" using Blockly. We will then convert the Blockly function into a Python function.

Video – [Web](#)

### ***Changing Fusion's “SUMO” Code into a Function***

In our previous Blockly SUMO tutorials ( [Web1](#), [Web2](#)) we produced Blockly code to teach our Fusion robot to stay inside the SUMO rink. In the tutorial below, we find out how to convert this code into a Blockly function. We also take a careful look at the equivalent Python SUMO function.

Video – [Web](#)

Next, we will assemble these Edge-Follow and SUMO functions into code that will teach our Fusion Robot how to clear all the left-behind toys from the swimming pool.

### ***Fusion attempts to “Clear the Swimming Pool”***

In the video below we combine the Edge-Follow and SUMO Blockly Functions into a program that will teach our Fusion Robot to follow the curving path through our pretend wilderness, cross the pretend silver sand beach, and clear left-behind toys from our pretend swimming pool by pushing them to the edge of the pool so they can be collected by our non-swimming teacher.

We then convert this Blockly code into Python Code. We examine the Python code, line by line. We then see if this code works when it is run on our Fusion Robot.

Video – [Web](#)

Fortunately, this code works beautifully with our Fusion Robot!

This Challenge also illustrates the use of "functions". Functions are tremendously useful in coding. They can encapsulate past coding wisdom in easily re-usable packages. These "packages" (called functions, procedures and other names in different coding languages) can be assembled into libraries, for future re-use. Sometimes these "libraries", can be private, containing secret expertise that a company sells. Sometimes programmers release their code for free use by other programmers - just like our videos that we release for free use by students world-wide. If you are learning Python, it is *really* worthwhile learning how to use functions. We will have more about functions in later tutorials.

### ***Extra: Keeping Fusion inside an Irregularly Edged Rink***

There are occasions when the rink is all one color, and the areas around the arena are of multiple colors. This type of rink would pose problems for the SUMO code we have used, which relies on the border around the SUMO rink being the same all the way around the rink. Fusion can handle this type of rink as well, but needs a [Color](#)

**Sensor.** In the following video, we demonstrate how to use the color sensor to teach Fusion to stay inside a rink which has an edge of an unpredictable color. The way this works is that Fusion detects the edge of the rink when it detects a color that is different from the rink color.

The color sensor is not part of the Fusion Base Kit, but is available for [separate purchase from Boxlight](#). We have added a color sensor to an experimental version of our Fusion robot. This experimental version has a variety of extra sensors, but we only use the Color Sensor in this tutorial. In the following video, we demonstrate how to use the color sensor to teach Fusion to stay inside a rink which has an edge of an unpredictable color.

Currently in this tutorial we only demonstrate the use of Blockly. We plan to extend this video to include Python when we have time to do so. In the meanwhile why not try extending our work to include Python yourself? It could be a fun project! 😊

### Video – [Web](#)

**Technical Note:** Currently only one Color Sensor may be connected to the Fusion Controller at a time. My understanding is that this is a limitation of the current Fusion implementation of Blockly, not a limitation of the *Color Sensor/Core Controller* combination itself.

If you have a good technical background, (or are brave and like base 16 numbers 😊 ), you can take a look at Modern Robotics' note about how to change things so that two color sensors can be used simultaneously on a Fusion Robot, [click here to read Modern Robotics' explanation](#).

### **Thanking you.**

Thank you for reading about our tutorials. We hope you have as much fun using them, as we had making them! 😊

## Where to go next?

It has been fun taking a "First Look" at how to use Python to teach our Boxlight's Fusion Robot to behave as we would wish it to behave.

If you have decided that you really like Python, and want to study a course that looks at a more detailed use of Python itself, without reference to a Robot, good courses are available, some free.

If you are a computer professional, Python is a useful computer language to have in your CV. It has excellent mathematical and artificial intelligence libraries available to assist in writing advanced programs. It is also currently (2023) [much in demand in industry](#).

We have not covered the use of advanced Python programs in this introductory course. If you want to study Python in more detail (probably beyond what you will need in School Python courses), an excellent University-level two-semester free on-line Python course that I can recommend from my personal experience [is available here](#). This free course is made available via [edX](#) from [what is arguably the best University in the world](#), the [Massachusetts Institute of Technology \(MIT\)](#). While this course is free, if you want a completion certificate to hang on your wall to impress your students and colleagues, there is a \$\$\$ fee. We suggest that it is worth looking at this excellent course, before looking at other courses that part you from your hard-earned dollars...

Enjoy! 😊