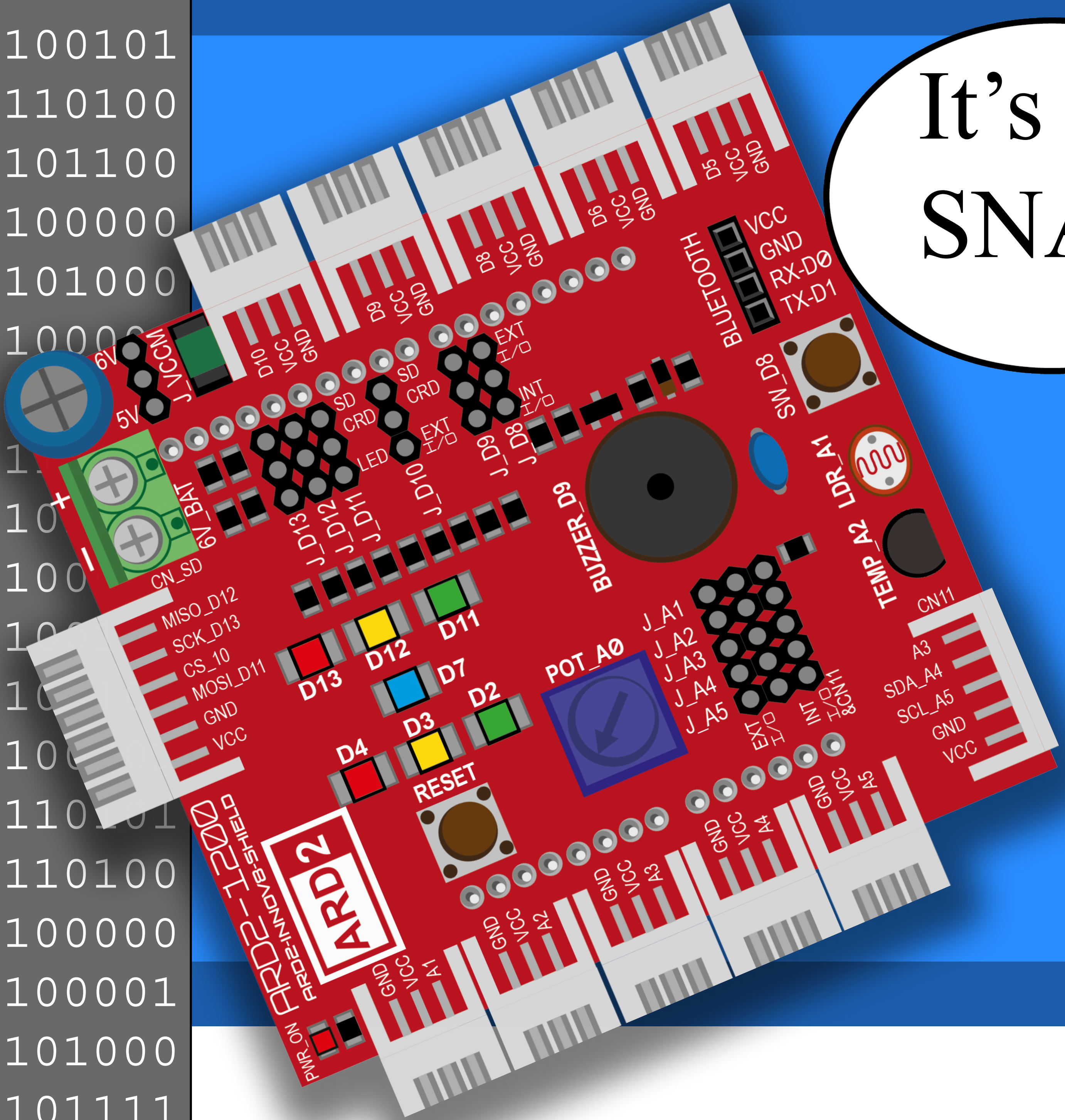
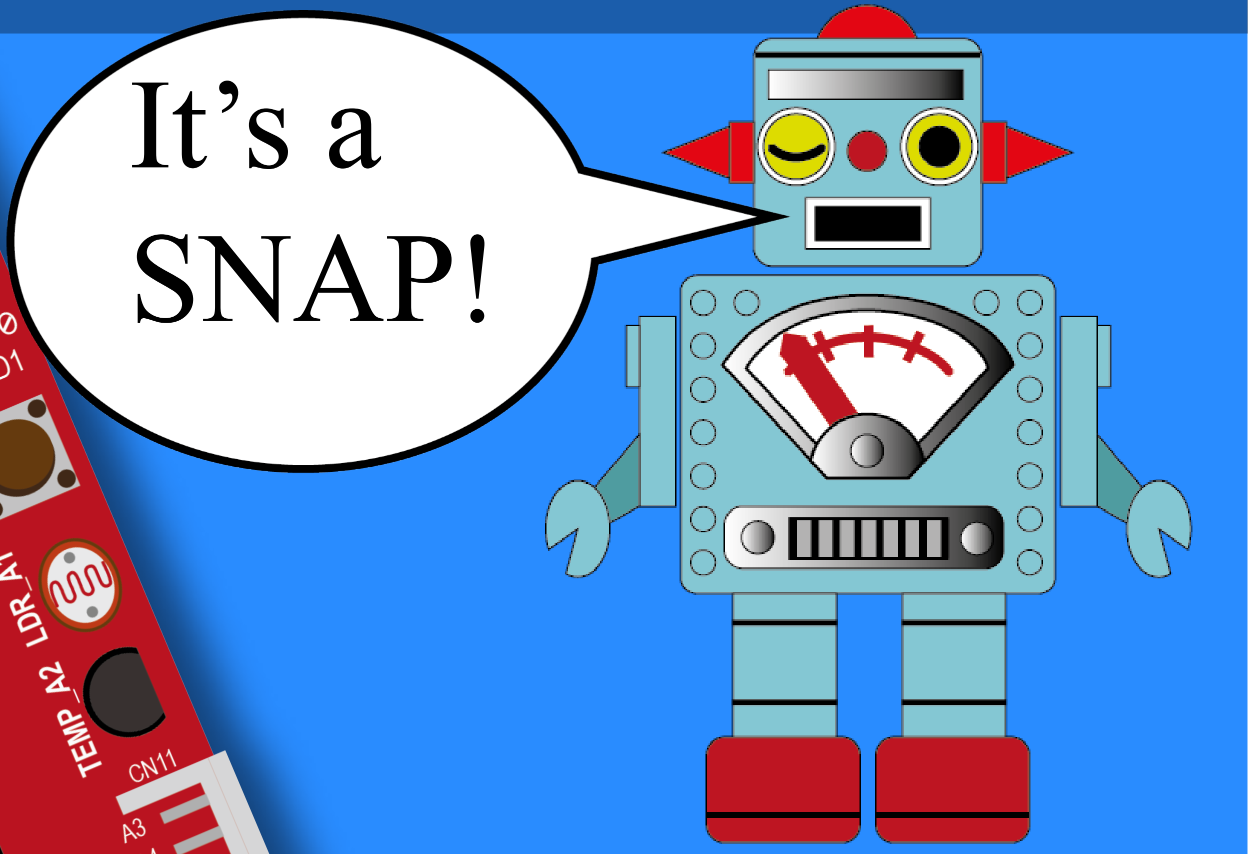
01001100 01100101
01110100 00100000
01110100 01101000
01100101 00100000
01100110 01110101
01110100 01110101
01110010 01100101
00100000 01110100
01100101 01101100
01101100 00100000
01110100 01101000
01100101 0010000
01110100 01
01110101 011
01101000 0010
00100000 01100
01101110 0110
00100000 0110
01110110 01100
01101100 01110101
01100001 01110100
01100101 00100000
01100101 01100001
01100011 01101000
00100000 01101111
01101110 01100101

Coding Innov8ion





with the ARD2-INNOV8

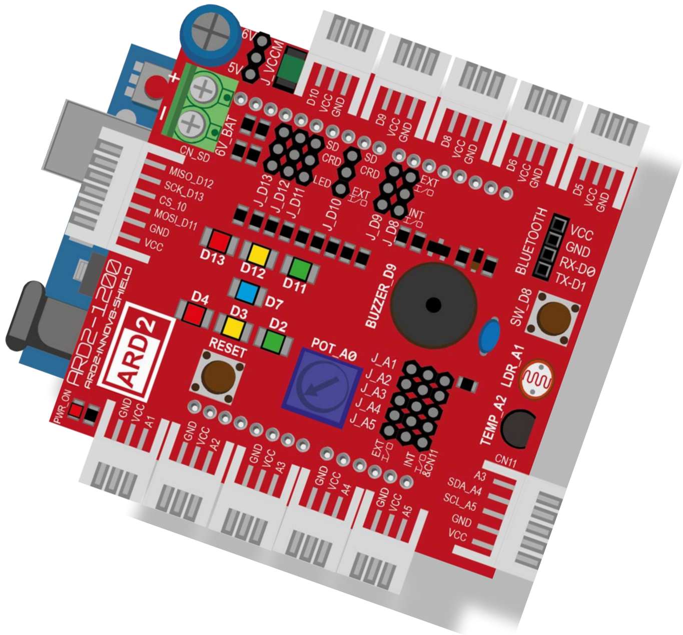
and Snap4Arduino

1st edition

Visual - block coding

Seven Vinton and Mark Trezise

Book Sample



Coding Innov8ion, it’s a Snap!

with the ARD2- INNOV8 and Snap4Arduino

Visual-block programming guide

Coding INNOV8ion it’s a Snap! – with the ARD2-INNOV8 and Snap4Arduino - visual-block programming guide by Seven Vinton and Mark Trezise. A self-published eBook.

Illustrations by Seven Vinton.
Some of the photographs in this book were obtained under the creative commons CC0 1.0 Universal (CC0 1.0).

ISBN: 978-0-6480792-0-0

Acknowledgements

Thank you to Joan Guillén i Pelegay and Bernat Romagosa (Snap4Arduino Developer) for their help and assistance with Snap4Arduino code blocks.

Thank you to Ethan Zerafa (aged 11) for testing the book and the code.

Thank you to Rain Richardson for proofreading.

FireBugS

Firebugs Educational Resources

Disclaimer

The information provided in this book is for educational purposes only. Whilst great care has been taken in the preparation of the contents of this book, the authors of this book and the ARD2-INNOV8 team take no liability or responsibility for any errors or omissions, or any loss or damage resulting from the use of any information provided in this book or any of the linked websites or online video tutorials. Any use of the information provided is at your own risk.
There are no warranties, expressed or implied, about the accuracy or reliability of the information provided, or as to the availability and suitability of any of the technologies discussed in this book, as this will vary from case to case.

Content Index

Contents

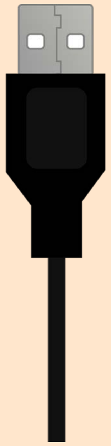
Section A - Getting Started

Section B - ARD2-INNOV8 Programming Lessons

Section C - Reference

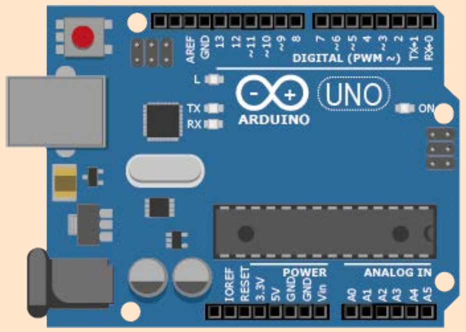
Here's what you need to start programming with the ARD2-INNOV8

A computer (Windows, Mac, Linux)

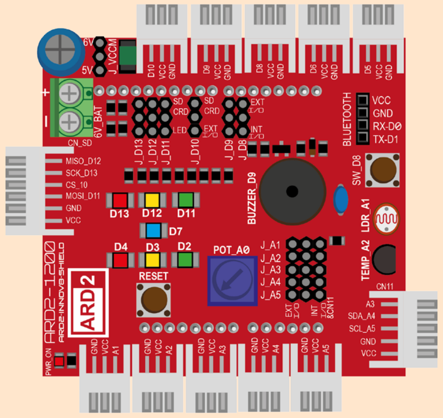


USB Cable

+

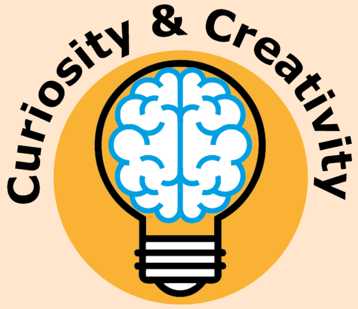


Arduino Uno



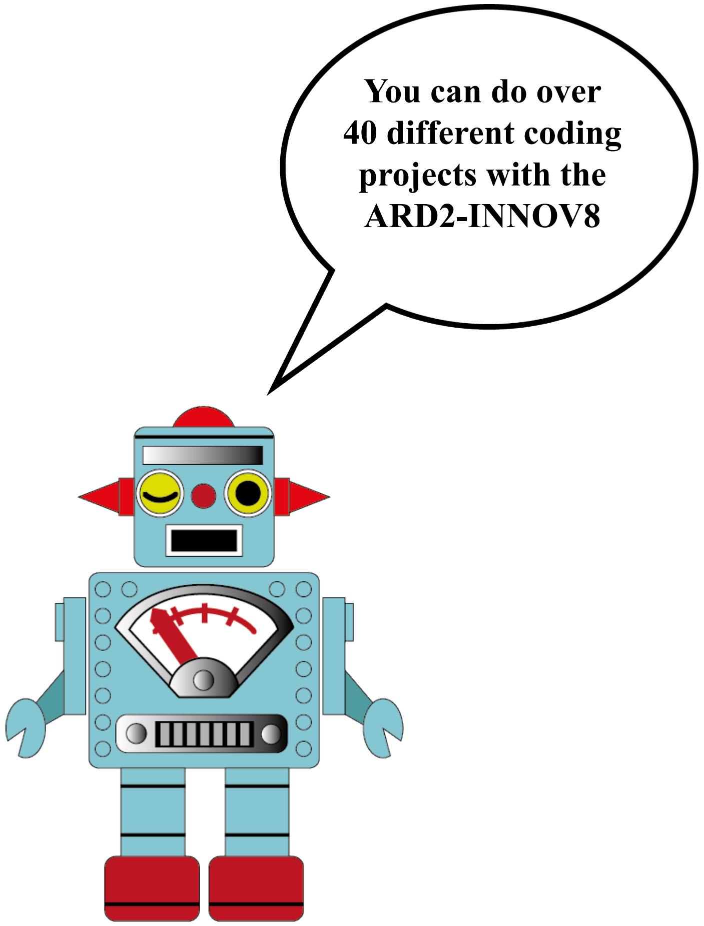
ARD2-INNOV8 Shield

+



What is the INNOV8 Shield?

The ARD2-INNOV8 shield is a piece of technology which has various *outputs* and *inputs*, and sits on top of an *Arduino Uno* compatible *micro-controller* board (other compatible boards can be found here: http://playground.arduino.cc/Main/SimilarBoards)
The ARD2-INNOV8 shield has been fitted with 7 LEDs (light emitting diodes), a piezo buzzer, a bush button switch, a potentiometer, a light sensor, a temperature sensor, a 6 volt input, and convenient snap in connectors to use with a wide range of external modules such as servo motors, joystick module, and a real-time-clock module.
The ARD2-INNOV8 shield was designed to take the headache out of learning how to program a microcontroller like the Arduino Uno. It allows the user to learn a wide range of Arduino lessons without the hassle of wiring up all of the required components. The ARD2-INNOV8 shield helps make learning Arduino programming easy, by allowing the user to concentrate on the programming without being distracted by bugs in the circuit.


You can do over 40 different coding projects with the ARD2-INNOV8

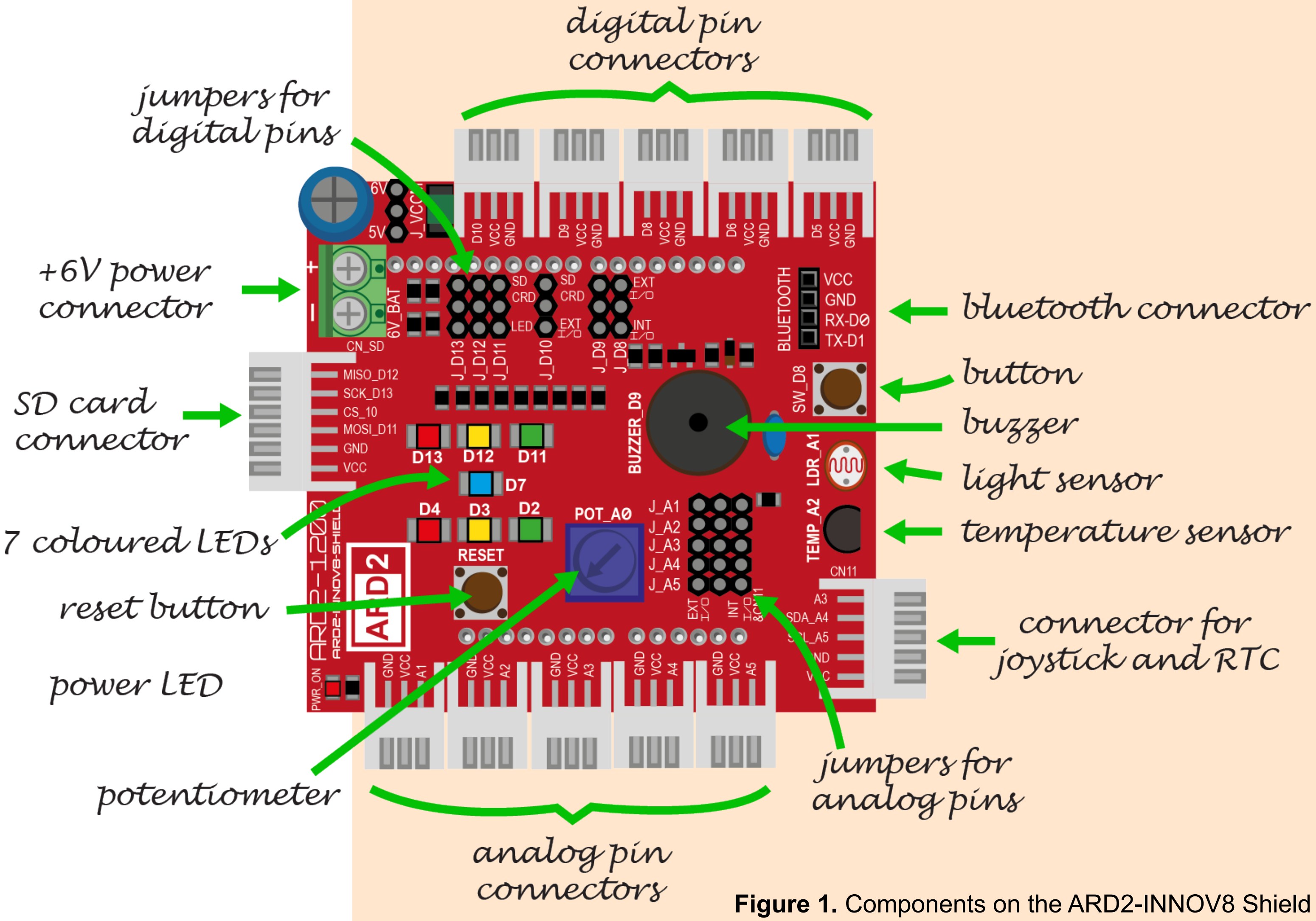


Figure 1. Components on the ARD2-INNOV8 Shield

The ARD2-INNOV8 story

The ARD2-INNOV8 story is a story of innovation and collaboration; it is a story of perseverance, life-long learning, and community.

The two authors of this book Seven Vinton and Mark Trezise are both technology enthusiasts. Their passion for learning inspired them to form a partnership to help make this learning available to others via online videos, learning resources, and learning seminars & expos.

During one of these student seminars Seven and Mark started to discuss ways to make Arduino programming easier for students. They had used many devices which aimed to make Arduino programming easier, however, all of these devices fell short in one way or another.

Seven and Mark met with the CEO of *Wiltronics*, Richard Wilson in December of 2016 to discuss how they could work together to promote learning of technologies like the Arduino devices, and at this meeting they aired their frustrations about the failings of the available technologies. Richard's answer was simple, "Let's make our own!"

From this meeting the prototype design for the ARD2-INNOV8 shield was developed by Seven and Mark along with its name, and upon receiving these designs Richard enlisted the help of his colleague Ben Sieira at *CognetronicS* to design the wiring layout and electronics engineering for the shield.

As previously mentioned, this story is a story of collaboration, however, this story of collaboration is a little different from other stories you may have heard, because Seven is a secondary school teacher, and Mark is a student (year 10 at the time). This collaboration between teacher, student, and industry experts is rare, but it is a form of collaboration which needs to happen more if our country is going to be competitive in the world's technology market.

This partnership is truly innovative, which is why the name ARD2-INNOV8 was chosen for the shield.

The ARD2-INNOV8 shield is a piece of technology designed for students and teachers by a student and teacher in collaboration with two great Australian technology companies.

This story should inspire you to realise that with effort, creativity, and thinking, you could become the creator of the next innovative piece of technology.

What is Arduino and why do I need it?

Your ARD2-INNOV8 shield can let you do many cool things, however, it cannot think on its own, because it does not have its own *micro-controller*. To run programs through your ARD2-INNOV8, you need to sit it on top of a micro-controller like the Arduino Uno.

Arduino is a hardware platform which is built around a micro-controller. Many different boards have been developed as part of the open source movement by the Arduino community. Arduino boards have been used in thousands of projects over the years by innovators both young and old, producing a wide range of designs from: robots to: musical instruments, from: children's toys to: scientific measuring devices.

Arduino boards are used in schools for learning how to program, and they are used by professional electrical engineers to make innovative products.

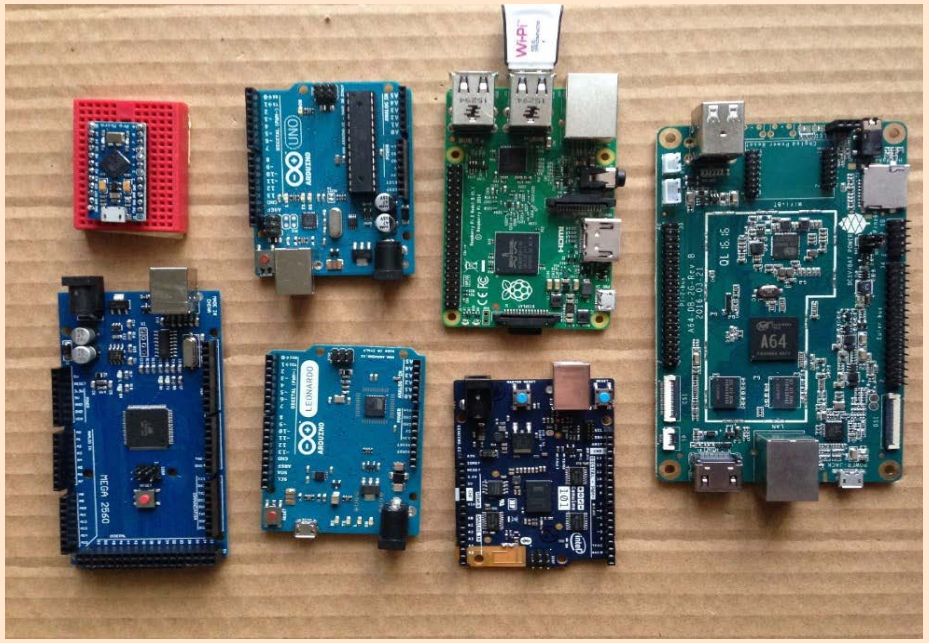


Figure 2. A selection of different micro-controller development boards

The most widely known and used board is the Arduino Uno. The Arduino Uno board is very versatile and reliable, and the design has been replicated by a number of technology companies and novices worldwide, ensuring that this technology is easily available across the world. The Arduino Uno will be the board used in all of the lessons and projects in this book.

Content Index

The first Arduino board was created in 2005 at the Interaction Design Institute in Ivrea, Italy, as a low cost and easy way of helping students to *prototype* and program with micro-controllers. The ARD2-INNOV8 shield follows this same guiding principle – low cost and easy to use.
Connections to the micro-controller are called pins; the Arduino has 13+ *digital* pins available for use, and 5 *analog* pins. These pins are linked to your ARD2-INNOV8 shield by connecting it onto your Arduino board. These connections to the Arduino mean that you can control the LEDs, buzzer, and other components on the ARD2-INNOV8 by creating programs which tell the Arduino's micro-controller what to do.

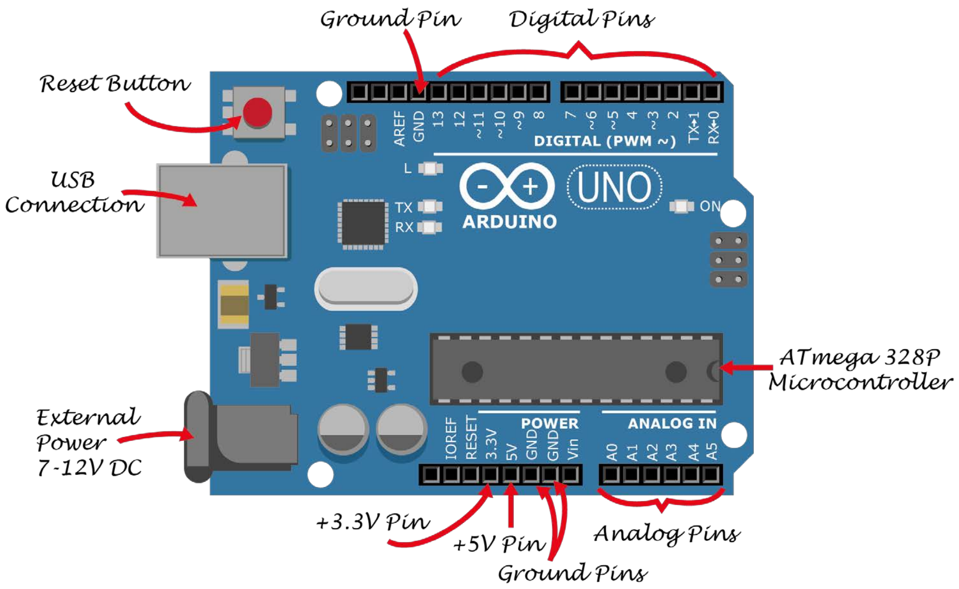


Figure 3. The Arduino Uno board, *analog* pins are numbered A1 to A5, and *digital* pins 1 to 13

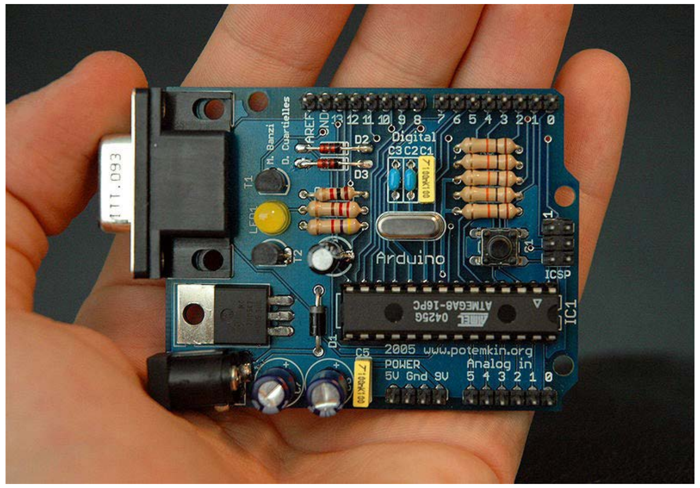


Figure 4. An early Arduino board which uses the Atmel ATmega8 chip as its microcontroller.
Photograph curtesy of Nicholas Zambetti - http://www.arduino.cc/, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=9182627

Why use the ARD2-INNOV8?

Firstly, it is a pretty inspiring story to share with others, about how, with persistance and effort, any student can work collaboratively with experts to do amazing things.
Secondly, the ARD2-INNOV8 is like a set of training wheels that you use on a bike when you are first learning to ride. These training wheels make the job of balancing easier so that you can concentrate on other things, like steering and stopping.

Because there are no messy wires to deal with and no chance of placing components the wrong way round, users of the ARD2-INNOV8 can more thoroughly concentrate on the programming language and algorithm design. Once you have gained confidence with Arduino programming, you can remove the ARD2-INNOV8 shield and move on to more complex systems.

The ARD2-INNOV8 shield's interface gives learners an easy way to learn the fundamental lessons in micro-controller programming, therefore providing a good solid start to learning about embedded systems and programming.
Seven and Mark designed the ARD2-INNOV8 shield in collaboration with Richard Wilson (Wiltronics) and Ben Sieira (CognetronicS) with the aim of providing students, teachers, and enthusiasts an easier pathway to learning how to program an Arduino micro-controller board.

Figure 5. The ARD2-INNOV8 is like training wheels on a bike

Getting started with your ARD2-INNOV8

A match made in heaven - Putting the ARD2-INNOV8 and Arduino together

There are a few things that you need to do before you can start programming and interfacing with your ARD2-INNOV8 Shield.
The most important of these things is that you need to have an Arduino compatible board such as the Arduino Uno that you can attach to the ARD2-INNOV8.
The reason that you need this Arduino compatible board is the ARD2-INNOV8 shield only provides an interface to components like LEDs and a button, but it does not have its own micro-controller, and without a micro-controller you cannot do any programming.
So the first step in programming with the ARD2-INNOV8 is to ensure that you have your ARD2-INNOV8 firmly attached to an Arduino Uno compatible board.
When attaching your ARD2-INNOV8 to an Arduino Uno compatible board, you should take extra care to ensure that all of the pins under the ARD2-INNOV8 are perfectly aligned with the socket headers on the Arduino (see **Figure 6**).

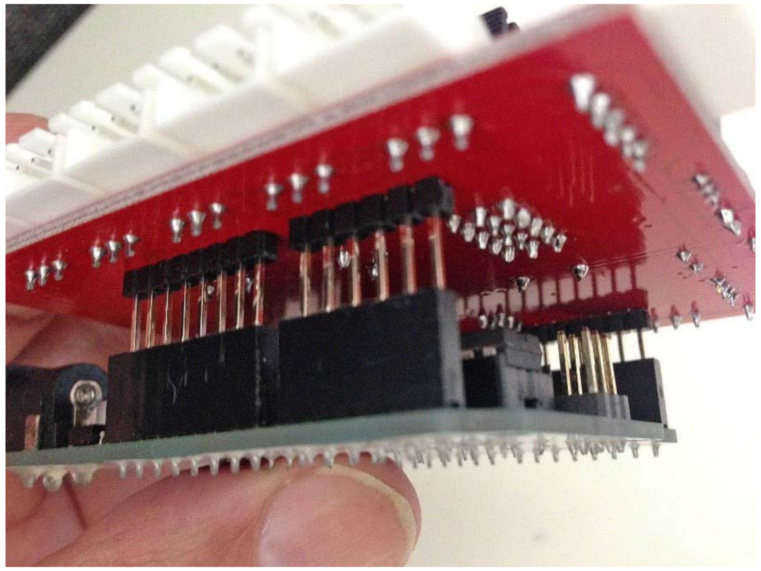


Figure 6. Ensure the pins under your ARD2-INNOV8 line up with the header sockets on your Arduino board

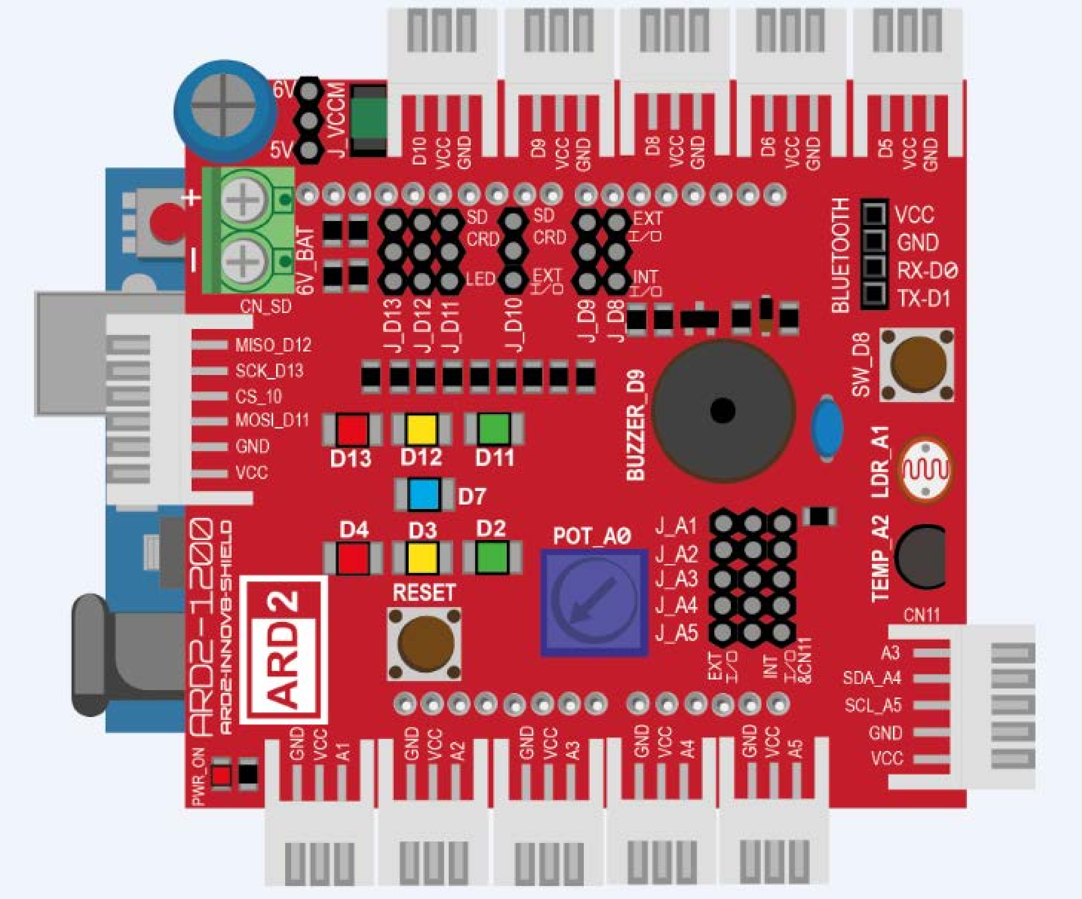


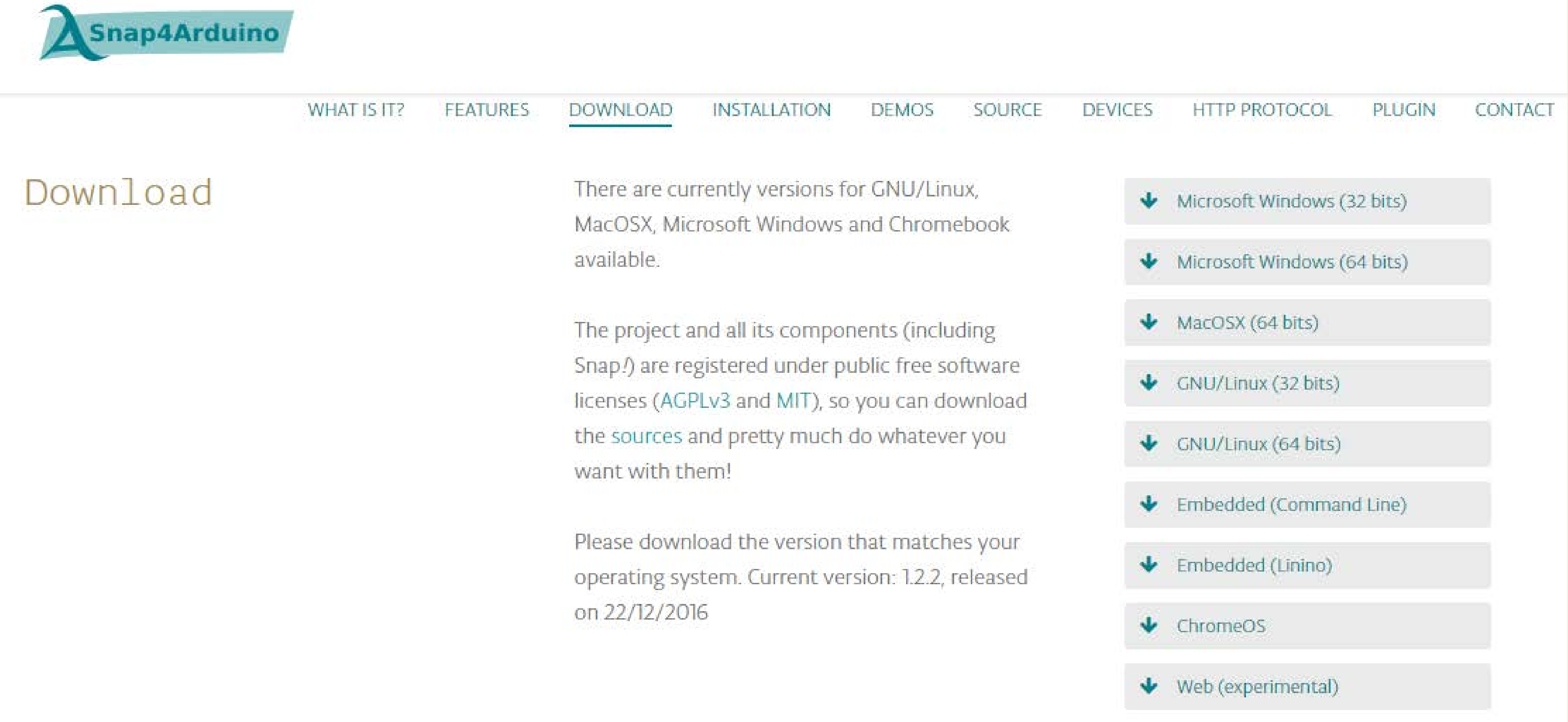
Figure 7. Your ARD2-INNOV8 and Arduino board should look like this when fitted together

Using Snap4Arduino

Programs run on Arduino micro-controller boards are referred to as *sketches*. There are many programming platforms available to use for Arduino programming, but most use *text-based code*.
Snap4Arduino is a modification of Snap!, a *visual block* programming environment which lets you drag-and-drop *code blocks* to create programs. Snap! (presented by University of California at Berkeley) is an alternative extension to Scratch (presented by the MIT Media Lab).
Snap4Arduino was created to allow drag-and-drop programming of Arduino boards and shields. It is a great place to start for younger learners and first-time Arduino programmers, but you may wish to use *text-based* code with the Arduino *IDE* (integrated development environment) when you advance to more complex programs.

Programs created in the Arduino IDE are uploaded (*compiled*) to the board, whereas programs created in Snap4Arduino are sent as instructions to the board when the program is running via a *firmware* program which has been loaded onto the Arduino board.

When you design code for your ARD2-INNOV8 shield and use its hardware components, you will be doing what is known as *prototyping*. Prototyping is when you make a limited working model of something to test a concept or idea. Prototyping with the ARD2-INNOV8 is faster and much easier to setup.



Download

There are currently versions for GNU/Linux, MacOSX, Microsoft Windows and Chromebook available.

The project and all its components (including *Snap!*) are registered under public free software licenses (AGPLv3 and MIT), so you can download the sources and pretty much do whatever you want with them!

Please download the version that matches your operating system. Current version: 1.2.2, released on 22/12/2016

Figure 8. The Snap4Arduino website, where you can download the programming software

How to set up Snap4Arduino on your computer

The Snap4Arduino website will provide you with all of the information you will need to successfully install Snap4Arduino onto your computer. You will also need to download and install the Arduino IDE from the Arduino website. This is discussed in a later section of this book.
Video guides demonstrating how to install this software are available on the Firebugs Youtube channel.

The Snap4Arduino programming environment

The Snap4Arduino programming environment or IDE (integrated development environment) is very easy to use, and is very similar to Scratch and other drag-and-drop programming environments.
You can select code blocks from the programming block libraries: *Motion*, *Looks*, *Sound*, *Pen*, *Control*, *Sensing*, *Operators*, *Variables, Arduino*, and *Other*.

In this guidebook we will be using code blocks from most of these libraries, but not all of them.
The centre area of the IDE is where you build your program by dragging on blocks from the code block libraries (**Figure 9**).

On the left of the IDE is where you access the code libraries, and on the right of the IDE is the ***Stage*** where you see the action and see your *variables* and how they are changing. You will get to see how this works when you start programming.
The Snap4Arduino IDE has a special library which contains code blocks that are for programming the Arduino. This library also contains buttons to connect and disconnect your Arduino board (**Figure 9**).

The ***Other*** code block library does not contain any blocks, but instead contains a single button which allows you to create your own code block *functions*. You will get to see how this works in **Lesson 9**.

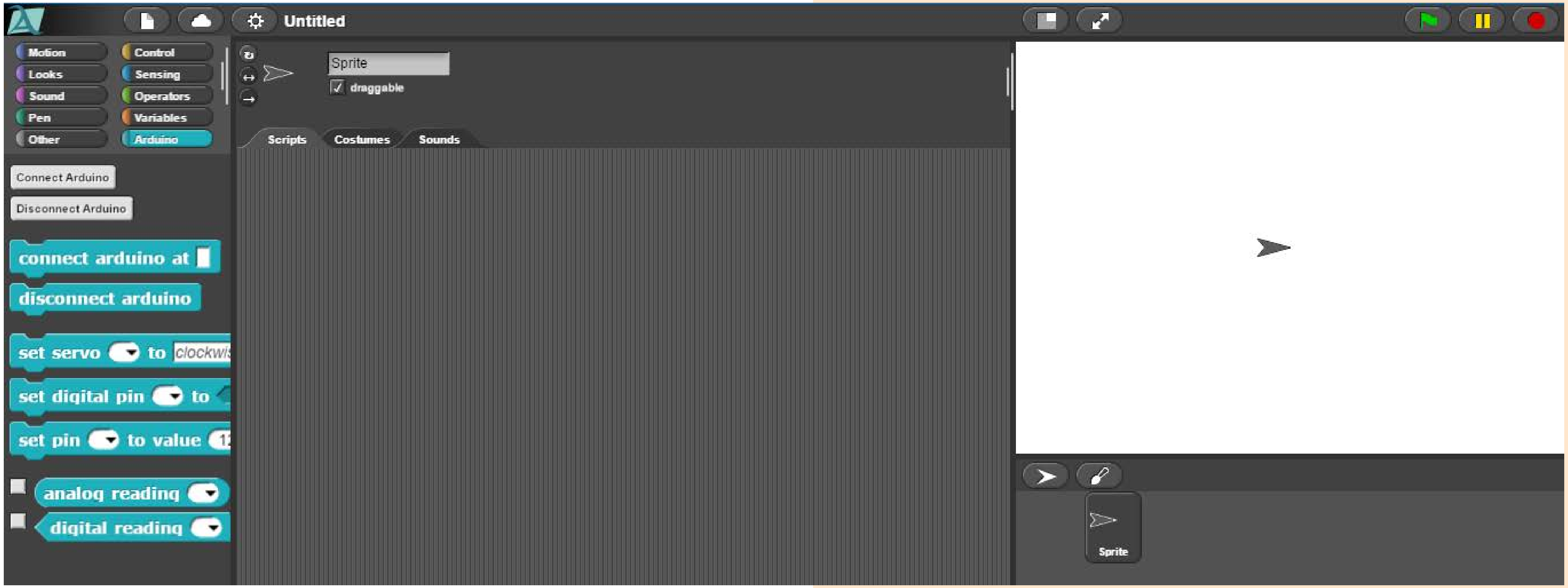


Figure 9. The Snap4Arduino programming environment

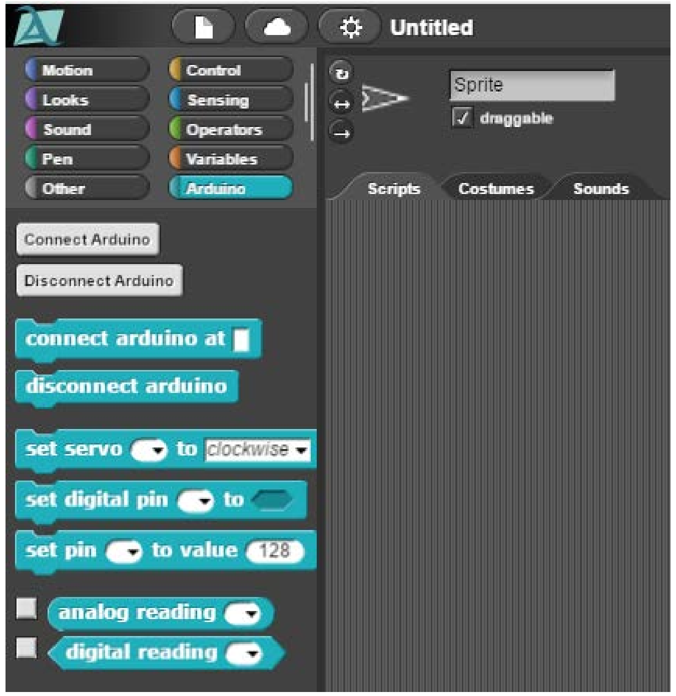


Figure 10. Programming blocks within the Arduino library

Creating and duplicating blocks

Creating new code blocks is easy; simply drag the block you want from its library into the programming window (**Figure 11**).

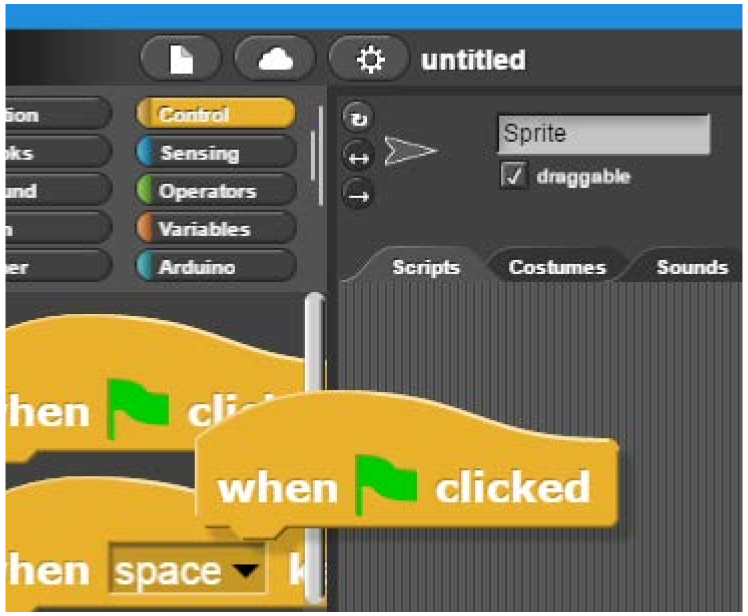


Figure 11. New code blocks can be created by dragging blocks into the programming window

You can easily duplicate blocks by right-clicking on the block you want to copy, and then selecting ***duplicate.*** You can also delete blocks this way, or simply drag the block back to its library and it will disappear.

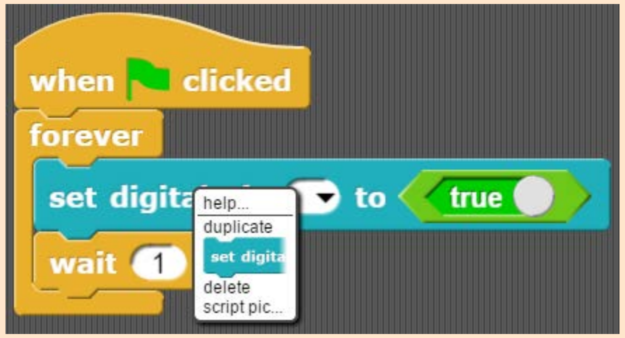


Figure 12. Options available when right-clicking on a block

Zooming in and out

You can increase or decrease the size of your blocks so they are easier to see by selecting ***Zoom blocks*** from the *Settings* menu (cog icon – **Figure 13**).
In the *Zoom blocks* window you can choose what size you want your blocks to be.

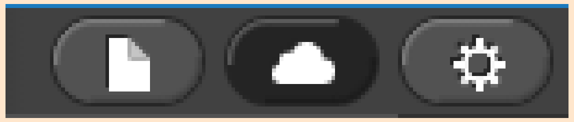


Figure 13. The File, Cloud, and Settings menus

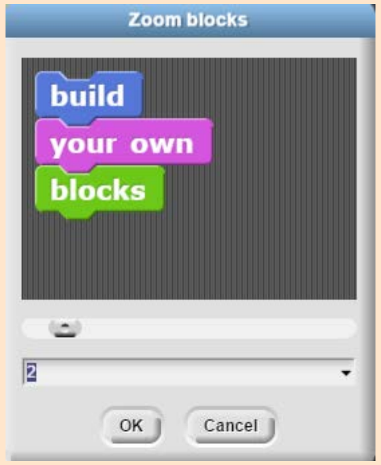


Figure 14. The Zoom blocks menu, where you can choose the size of your blocks

Saving, importing, and exporting

Saving, importing, and exporting is easy, and is done via the *File* menu. Common key shortcuts like *Ctrl-S* (Command +s) for save can also be used, and you can choose to save your projects into the projects folder or to the Snap4Arduino cloud if you have an account (account activation is free).

About the lessons in this book

Note to teacher

Each lesson in this book is designed to build capacity with a particular set of programming knowledge, skills, and problem solving. There is a *key focus*, and *key words* in each lesson, and most lessons build onto each other. Learning scaffolding is reduced as the lessons proceed, but the teacher may need to introduce further scaffolding when teaching junior levels.
There is a difficulty or challenge rating for each lesson represented by a number of heads:

This head of course belongs to Nicola Tesla, the famous inventor from the early 1900s whose creativity and ingenuity knew no bounds.
Learners who are having difficulty grasping the concepts in the more challenging lessons should skip them and come back to them later. Lessons can be approached by first copying and implementing the code, making observations, and then returning to the text once an understanding of the working system has been gained.
There is also a ***Quick Blocks*** section just after Lesson 1, which provides all of the basic sketches needed to run components on the ARD2-INNOV8 shield.

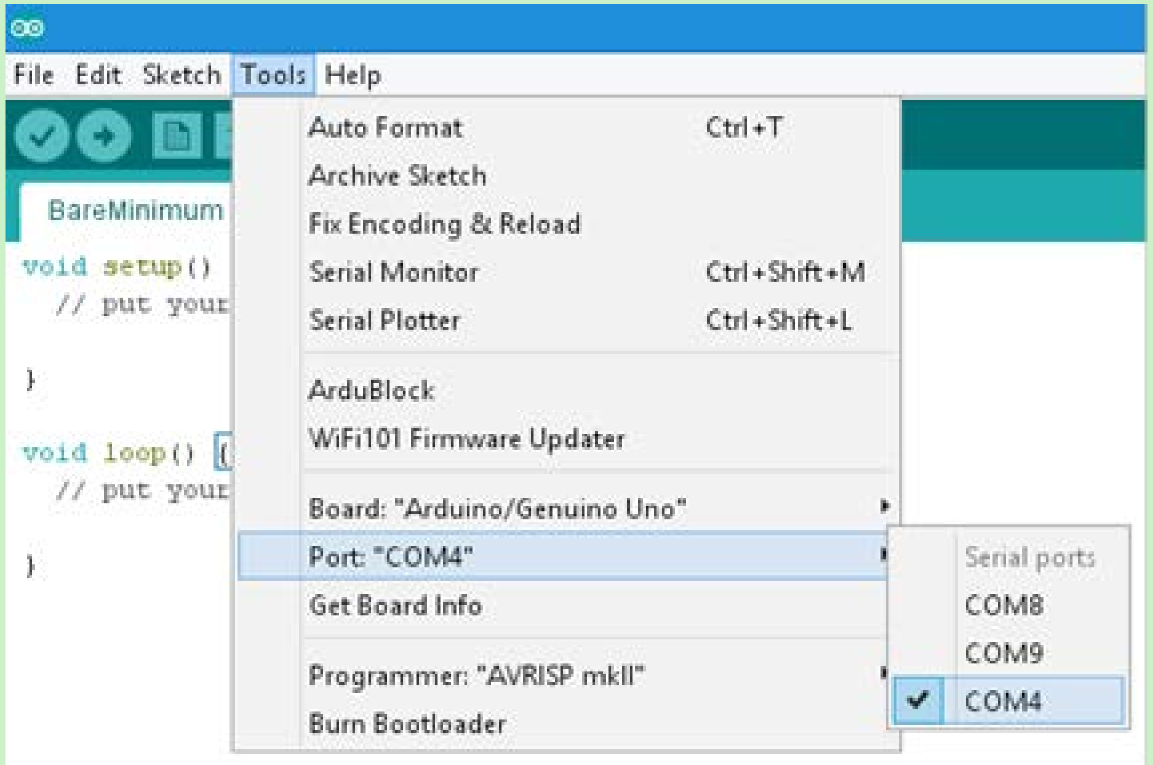
Loading the lesson code and resources onto your computer

Download the ***ARD2-INNOV8-SNAP*** resources folder from the Wiltronics or Firebugs websites.
Unzip this folder to an easy to find location somewhere on your computer.
This folder contains all of the lesson code which you can import into Snap4Arduino if you get stuck, and it also contains other learning resources.

Loading the firmware

To program your board with Snap4Arduino you will first need to install *firmware* using the Arduino IDE.Download and install the Arduino IDE onto your computer from the Arduino website.
Once you have this software installed onto your computer:

- Run the Arduino program on your computer.
- Connect your Arduino board with ARD2-INNOV8 shield attached to your computer using the USB cable.
- Check that your board is connect by looking under: ***Tools > Port:***
- You should see a ***tick*** next to one of the *serial ports* listed (this looks slightly different on a MAC and Linux computer, but the process is much the same) [see **Figure 15**].



```
void setup()
  // put your
}

void loop() {
  // put your

}
```

Figure 15. How to check for port connection under the Tools menu (PC screenshot)

Open the ***StandardFirmata*** sketch from: File > Examples > **Firmata**

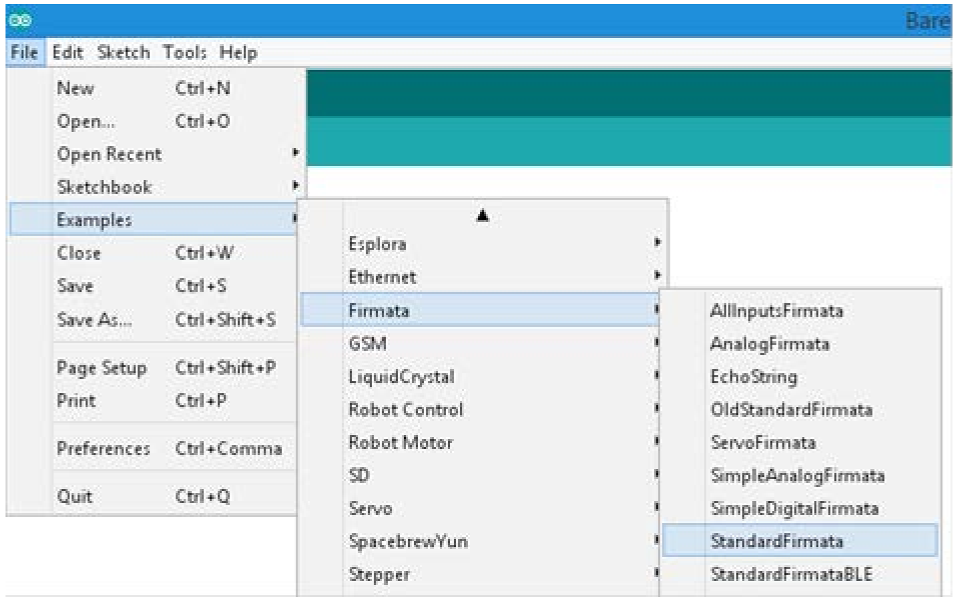


Figure 16. Where to find the *StandardFirmata* sketch (firmware)

Load this *StandardFirmata* sketch onto your board by clicking on the ***Upload*** button

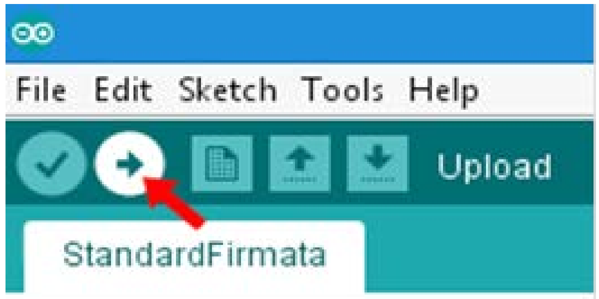


Figure 17. Clicking the Upload button loads the sketch to your board

If you see the message ***Done uploading*** at the bottom of the programming window, then you are all set to program with Snap4Arduino.

Connecting your board to Snap4Arduino

Next you will need to connect your board to the Snap4Arduino program. Within the ***Arduino*** blocks library you will see two buttons at the top. These are ***Connect Arduino*** and ***Disconnect Arduino***.

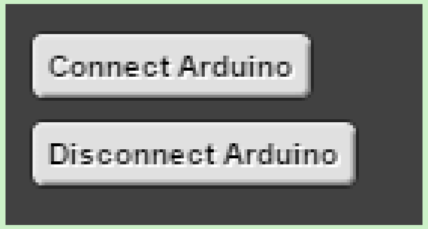


Figure 18. The connect and disconnect buttons for your board

When you click on the *Connect Arduino* button you should see the port to which your board is connected (the port number should be the same as when you connected to the Arduino IDE) [this looks slightly different on MAC and Linux]. Select the correct port number and you should then be greeted with the message in **Figure 20**.

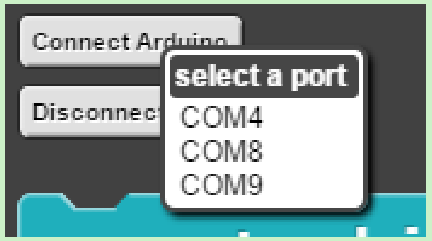
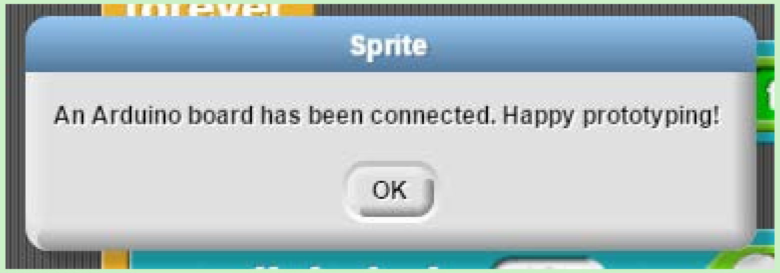


Figure 19. COM ports are visible when the Connect Arduino button is pressed

If you have trouble connecting your board, watch the video called ***Blink – First Sketch*** under the Snap4Arduino section on the Firebugs Youtube channel.
You will need to do this every time you open or create a new sketch.



An Arduino board has been connected. Happy prototyping!

Figure 20. The message seen when a board has been successfully connected

Alternatively, you can include this block at the top of your sketch which will then automatically connect the Arduino board at the chosen port once the sketch is run.

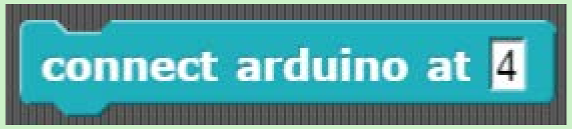
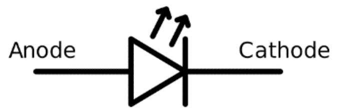
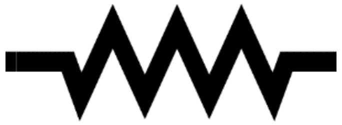
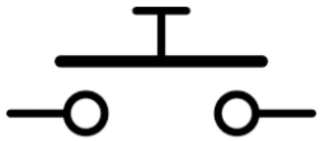
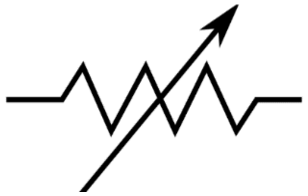
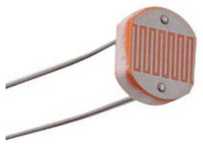
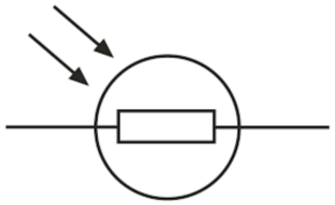
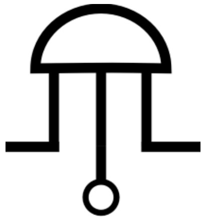


Figure 21. When this block is initiated it will connect the Arduino at COM port 4

Some basic electronics

You will need to know a little bit of electronics to fully understand the lessons in this book. Here are explanations of some of the electronic components found on the ARD2-INNOV8 shield.

Image	Diagram	Name	What it does
		LED (light emitting diode)	Gives off light when an electric current is applied. Electricity can only flow one way through a LED. There are many colours available. This image shows a standard LED; the LEDs on the ARD2-INNOV8 are small '*surface mount'* LEDs.
		Resistor	Slows the electric current down. It acts like a bumpy road slowing cars down. The coloured bands on a resistor show its value – measured in Ohms. You use a resistor in series with a LED to protect the LED. This image shows a standard resistor; the resistors on the ARD2-INNOV8 are small *'surface mount'* resistors.
		Push Button	Connects a circuit when the button is pressed. There are many different types of push buttons; this one is known as a momentary SPST switch. There are two of these on the ARD2-INNOV8: one can be used in program sketches, and the other is used to reset the code.
		Potentiometer	Much like a resistor, but its resistance value can be changed by turning its knob. It has three legs. Used as a control knob to vary a value. Can be used to turn volume up and down, or to dim a LED from dull to bright.
		LDR (light dependent resistor)	Changes its resistance value when light falls onto it. Can be used as part of a circuit to switch lights on when it gets dark.
		Temperature Sensor	Used to sense temperature. Can read temperature from -40C to +125C. Reads as a voltage which then needs to be converted in the code to temperature.

Content Index

Lesson 1 - Blink – Your first sketch

Key words: firmware, input, output, process, system, LED, sketch, control structure, loop, operator, Boolean, digital, pseudo-code, debugging
Key focus: Snap4Arduino IDE, basic coding, basic electronics, systems knowledge, using loops

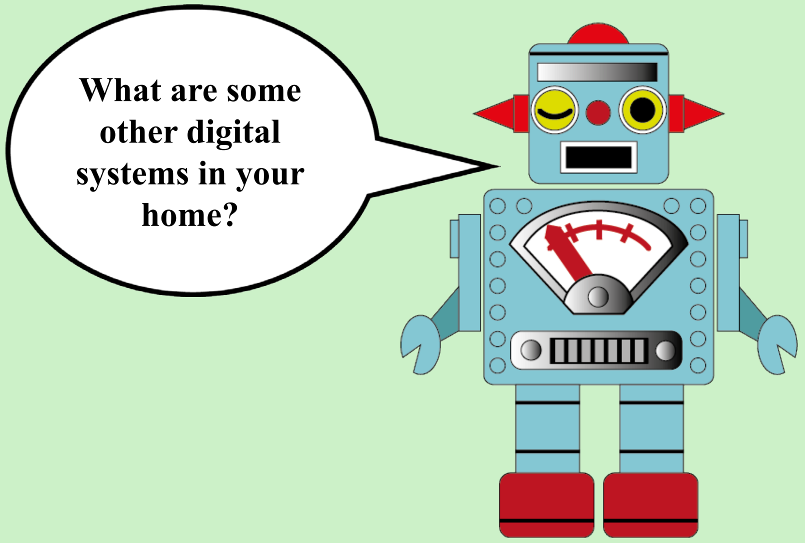
Difficulty Level

Inputs, Processes, and Outputs

In all of the lessons in this book we are going to be working with digital devices (the ARD2-INNOV8 and Arduino Uno) which have *inputs*, *processes*, and *outputs*.
All systems have at least one input, one process, and one output, however, most systems have more than one of each of these.
In the following lessons you will be working on the process side of *digital systems*, and programming them to use a number of inputs (such as a button press) and outputs (such as the lighting of a LED).
To get us started, think about some of the digital systems in your home and try to fill in the missing information in the table opposite.

Fill in the blanks in the table below:

System	Inputs	Process	Desired Outputs
Washing Machine	Electricity, button presses, sensor data	Fill (water), wash (agitate), spin, various options	Motion (rotary), sound (beeps), light(LEDs)
Smart TV	Electricity, data(signal), button presses, IR remote signal, Internet data, BlueTooth data		
Game System		User interface, provide games, connect to internet, connect to hand-controller, play disks	
Dish Washer			Motion (rotary), water pressure, heat, sound (beeps), light (LEDs)
Air Conditioner	Electricity, button presses, sensor data		



Setting up the code blocks

The very first sketch that Arduino programmers run on their boards is the *Blink* sketch.
This sketch blinks an *LED* (light emitting diode) on and off at specific timings.
The Blink sketch allows you to test your board, and gives you an opportunity to learn the code basics.
The blocks that we will use in this sketch are:

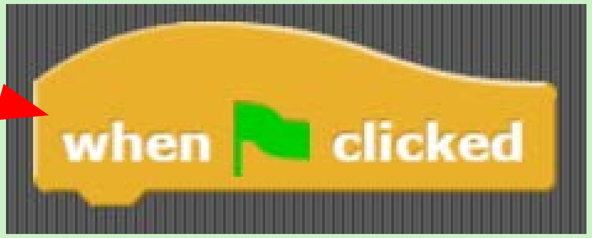


Figure 22. The When clicked block

The ***when clicked block***, found within the *Control* blocks library, runs/starts the program when you click on it.
This block is placed at the top of your code blocks, with all other blocks connected underneath.
There are a few different blocks which you can use instead of this block such as the ***when _ key pressed*** block.

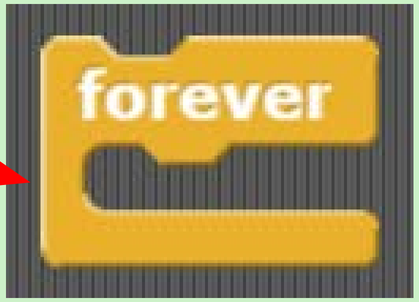


Figure 23. The forever block

The ***forever*** block is a *control structure*, found within the *Control* blocks library.
This block runs the code within it forever (in a *loop*), until the program is stopped by clicking back onto the ***when clicked block*** or if the board is disconnected.

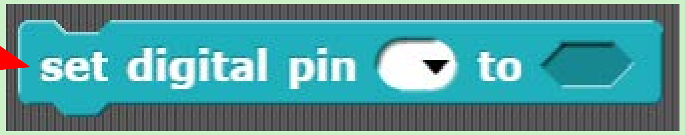


Figure 24. The Set digital pin block

The ***set digital pin_ to_*** block, found within the *Arduino* blocks library.
This block does two things: it *defines* the pin number which will be used, and it sets this pin to either *TRUE* or *FALSE* (HIGH or LOW – ON or OFF) with the use of the ***true/false*** *operators* (**Figure 25**). Pin numbers can be typed into the space provided.

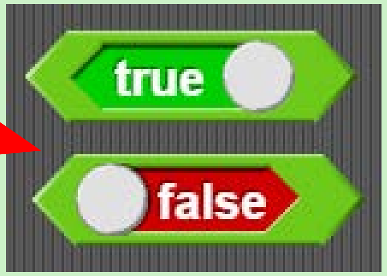


Figure 25. The true/false operators

The ***true/false*** operators, found within the *Operators* blocks library, are used to set the state of something. Their value is *Boolean* - either TRUE or FALSE. In the case of the *Blink* sketch, we will use these operators to set the LED attached to pin 13 to ON or OFF. These operator blocks can be dropped inside other blocks by dragging them over the appropriate part of the block.

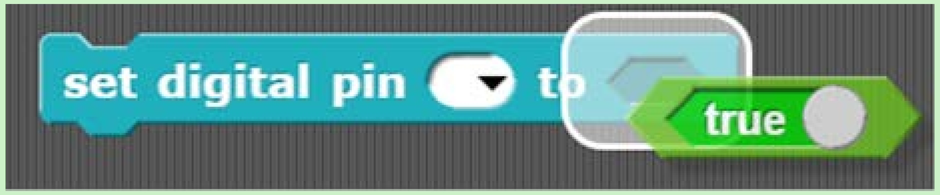


Figure 26. Blocks can be dropped inside of other blocks

The ***wait*** block, found within the *Control* blocks library, causes the program to pause for the time specified. The block pictured in **Figure 27** shows a block which will cause the code to pause for **1 second**.

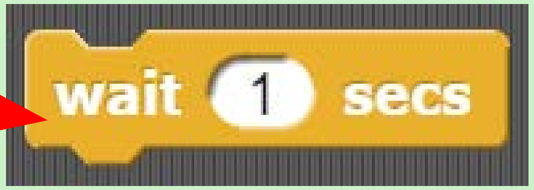


Figure 27. The wait block

Jumper position

The ARD2-INNOV8 board has ***pin jumpers*** which need to be placed in certain positions to use components on the shield. For lessons 1 to 9, make sure that your ARD2-INNOV8 shield has its pin jumpers placed in the position shown in **Figure 28**.

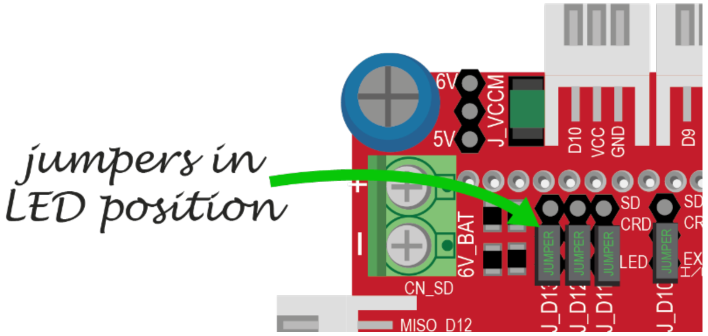


Figure 28. Ensure the jumpers are in this position for the Blink sketch

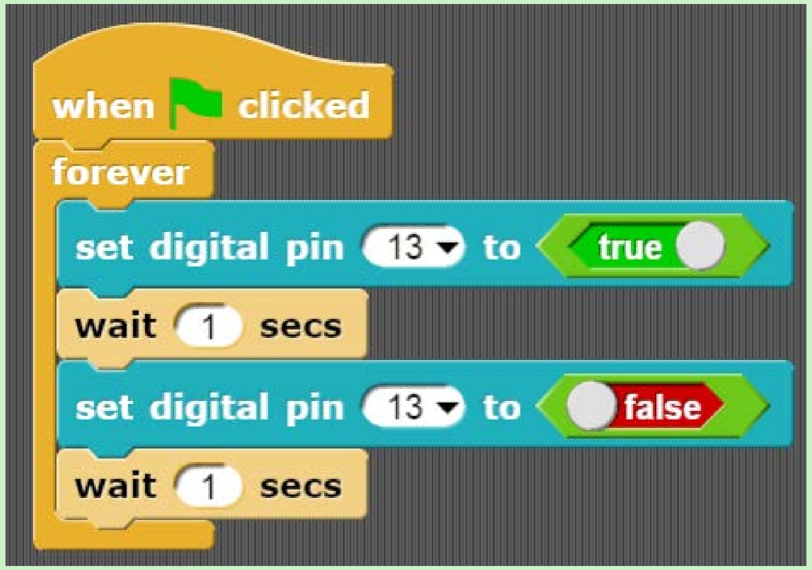
The Blink Sketch

The sketch shown in **Figure 29** would look like this if written in *structured English* (*pseudo-code*).

```
Blink Sketch
WHILE
     SET digital pin 13 to TRUE
     WAIT 1 second
     SET digital pin 13 to FALSE
     WAIT 1 second
ENDWHILE
EXIT
```

Study this code and the code in **Figure 29** and try to work out what will happen, then create the blocks as shown.
After you have constructed your sketch using the right blocks, make sure that your board is connected and then click on the ***when clicked block***.

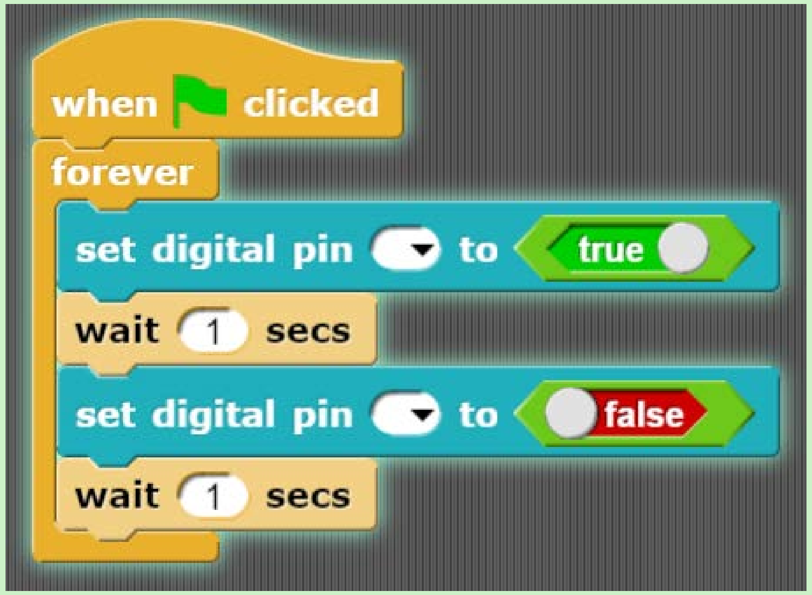
You should see a halo form around the blocks (**Figure 30**), and your ARD2-INNOV8 shield should be doing something.



```
when 🏁 clicked
forever
  set digital pin 13 ▾ to true
  wait 1 secs
  set digital pin 13 ▾ to false
  wait 1 secs
```

Figure 29. The Blink sketch

What is happening on your ARD2-INNOV8, did you see any LEDs blinking on and off?
If you did not see one of the LEDs on your ARD2-INNOV8 blinking on and off, then something has gone wrong.
When something goes wrong with digital systems we say that there is a *bug* in the system. The process of fixing the problem is known as *debugging*.
Fix your system and sketch by *debugging* the code. Check that your board is connected and that the pin jumpers are in the correct position (**Figure 28**).



```
when 🏁 clicked
forever
  set digital pin ▾ to true
  wait 1 secs
  set digital pin ▾ to false
  wait 1 secs
```

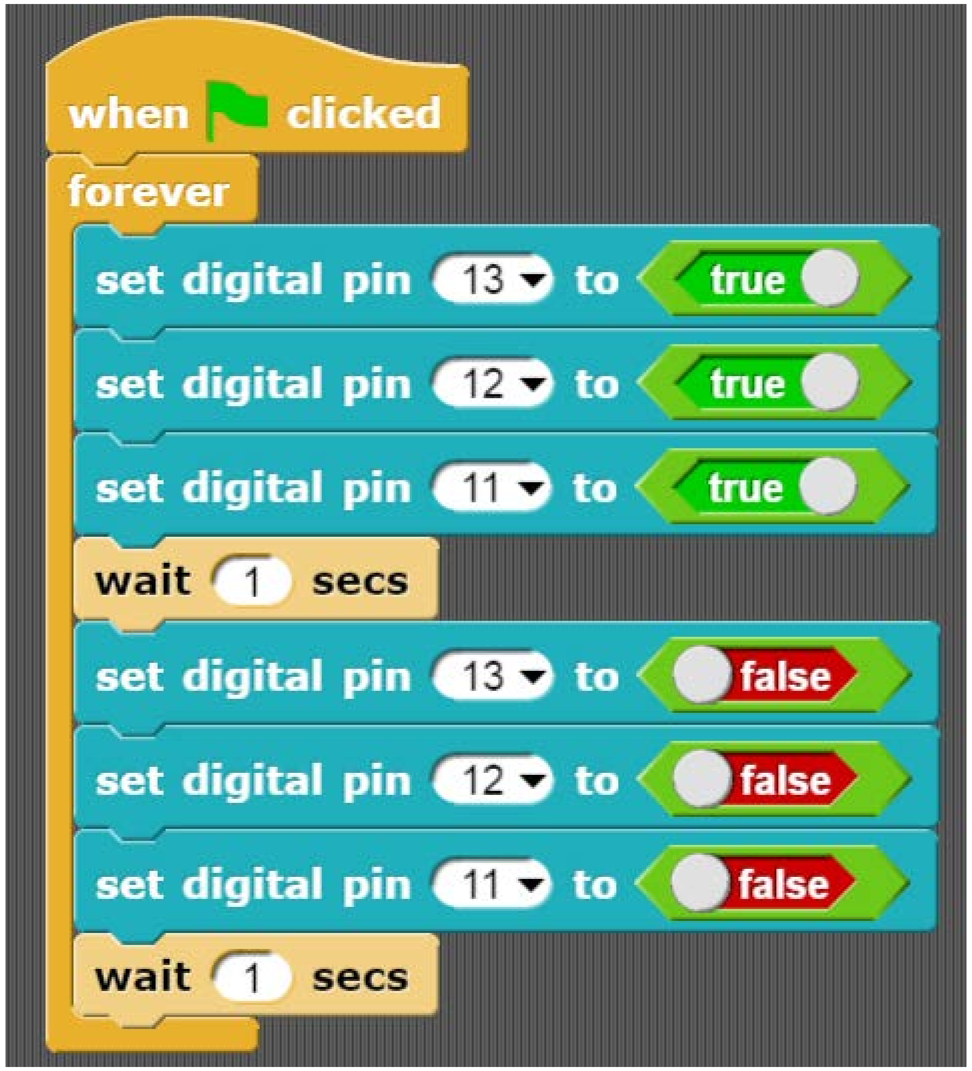
Figure 30. Halo around the blocks when active (clicked)

Now try this

Add in more digital pin blocks and select different pin numbers to see what happens.
The LED pins on your ARD2-INNOV8 are **2, 3, 4, 7, 11, 12, and 13**.

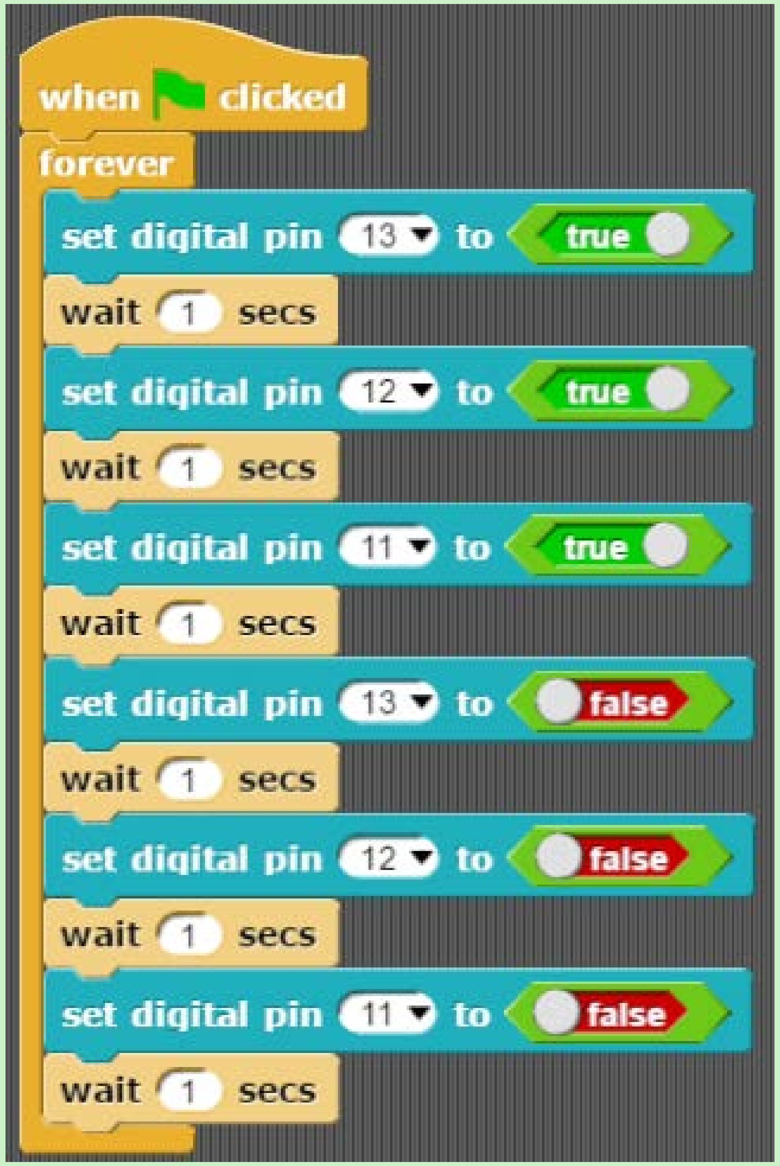
Try playing with the code blocks to see if you can:

- Make all 7 LEDs blink on and off
- Make the 7 LEDs blink on and off one after the other
- Changing the blinking speed (rate) to create a light pattern



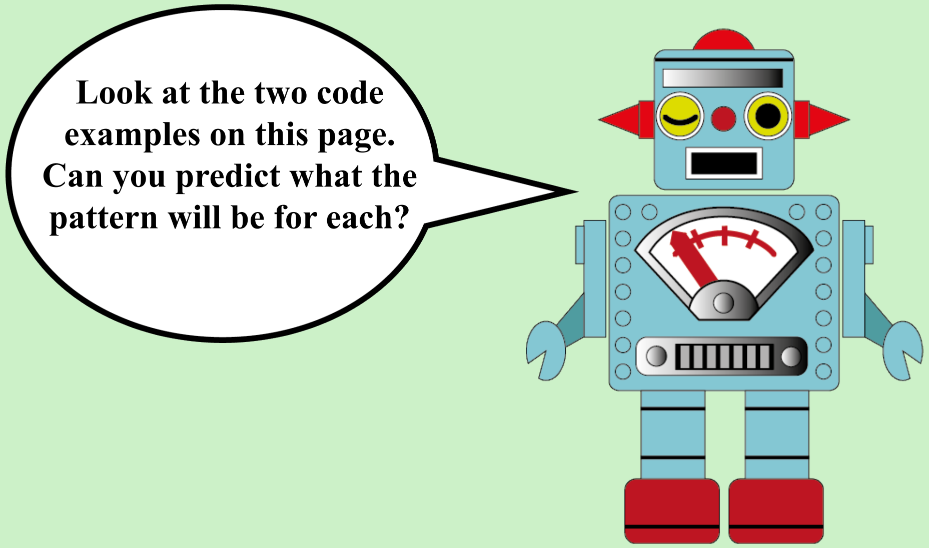
```
when [flag] clicked
forever
    set digital pin 13 to true
    set digital pin 12 to true
    set digital pin 11 to true
    wait 1 secs
    set digital pin 13 to false
    set digital pin 12 to false
    set digital pin 11 to false
    wait 1 secs
```

Figure 31b. The Blink sketch with added LED pins



```
when [flag] clicked
forever
    set digital pin 13 to true
    wait 1 secs
    set digital pin 12 to true
    wait 1 secs
    set digital pin 11 to true
    wait 1 secs
    set digital pin 13 to false
    wait 1 secs
    set digital pin 12 to false
    wait 1 secs
    set digital pin 11 to false
    wait 1 secs
```

Figure 31b. A different version of the sketch in Figure 31a

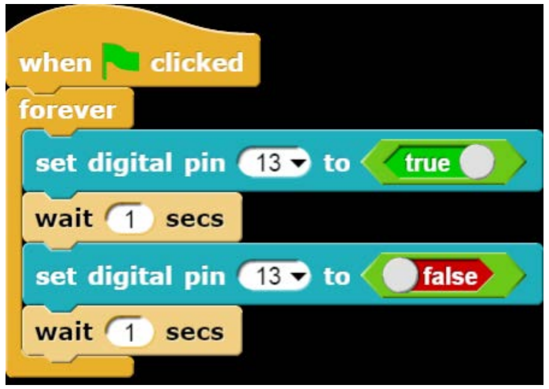

Look at the two code examples on this page. Can you predict what the pattern will be for each?

QuickBlocks

Difficulty Level

The following seven images show how you can build quick program sketches to use the inputs and outputs on the ARD2-INNOV8 shield. These examples provide only the very basics, however, the following lessons will explain how to build proper programs.
If you found **Lesson 1** difficult, then you may wish to play with the 7 examples in this Quickblocks section, but if you are keen to continue to learn real programming then you can skip ahead to **Lesson 2.**

Blink a light

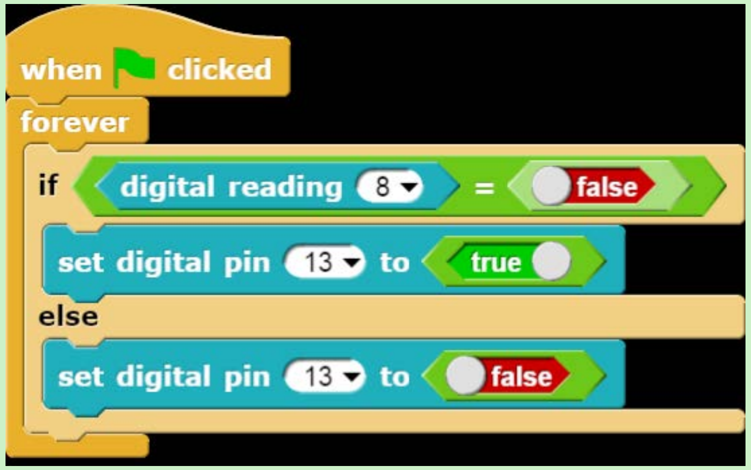


```
when [flag] clicked
forever
    set digital pin 13 to true
    wait 1 secs
    set digital pin 13 to false
    wait 1 secs
```

Figure 32. Blink an LED (this code will blink the LED on pin 13)

Blink a LED with a button

Press button D8 and observe what happens.

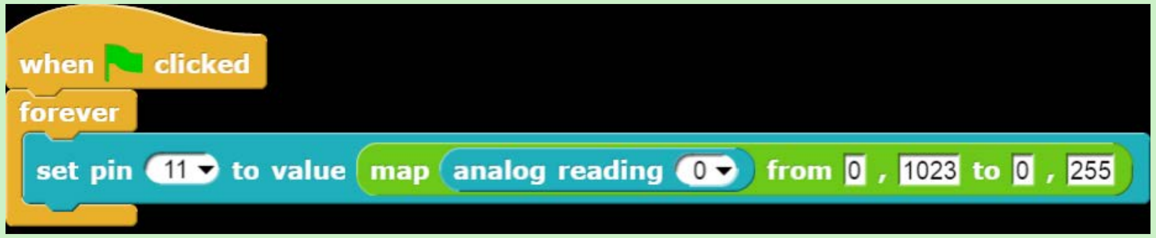


```
when [flag] clicked
forever
    if digital reading 8 = false
        set digital pin 13 to true
    else
        set digital pin 13 to false
```

Figure 33. Turn an LED on with a button press

Fade a LED by turning a knob

Twist the knob of the potentiometer and observe what happens.

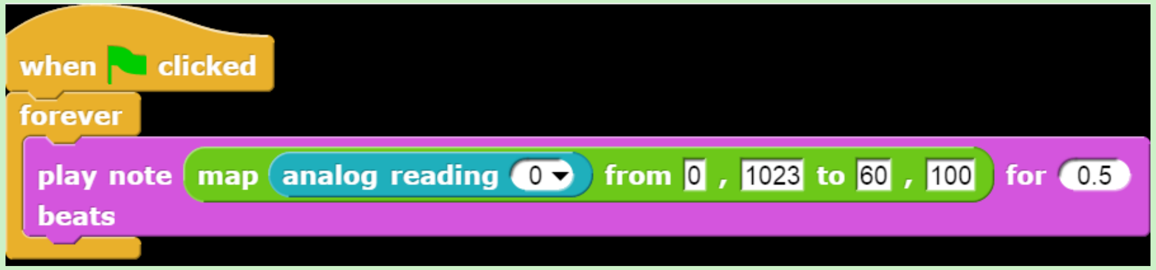


```
when [flag] clicked
forever
    set pin 11 to value map analog reading 0 from 0 , 1023 to 0 , 255
```

Figure 34. Fade an LED with the potentiometer (you need to import the map block for this program – refer to Lesson 6)

Change sounds by turning a knob

Twist the knob of the potentiometer and observe what happens.

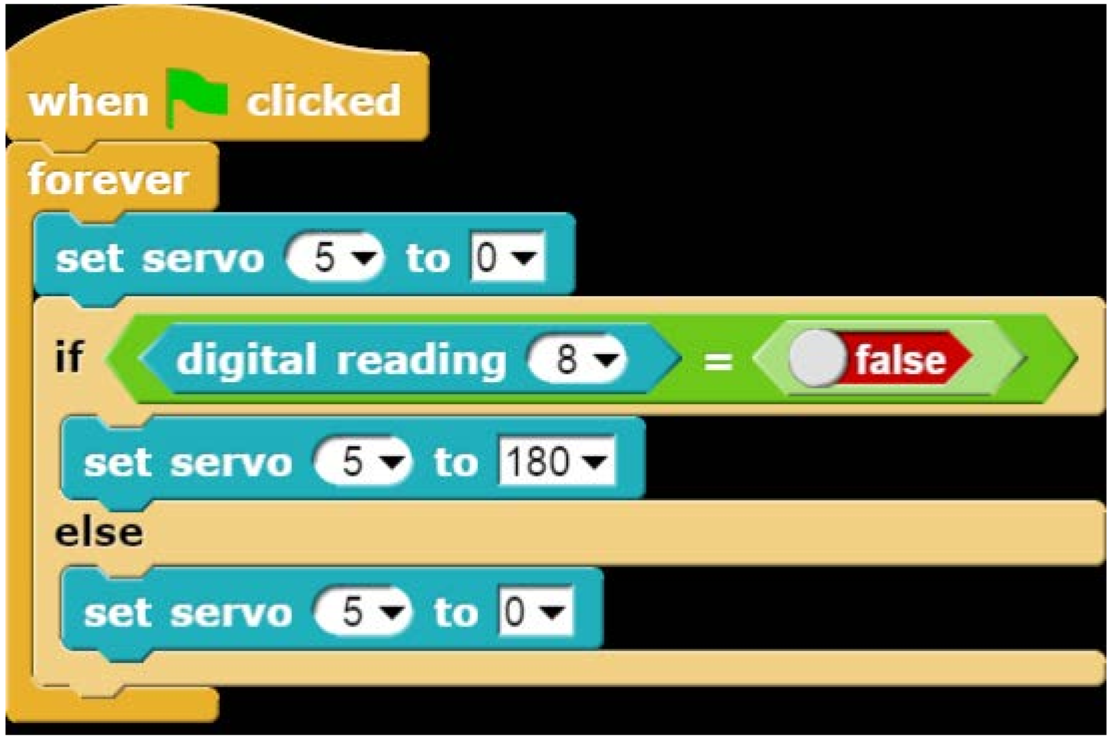


```
when [flag] clicked
forever
    play note map analog reading 0 from 0 , 1023 to 60 , 100 for 0.5
    beats
```

Figure 35. Change the sound pitch with the potentiometer (you need to import the map block for this program– refer to Lesson 6)

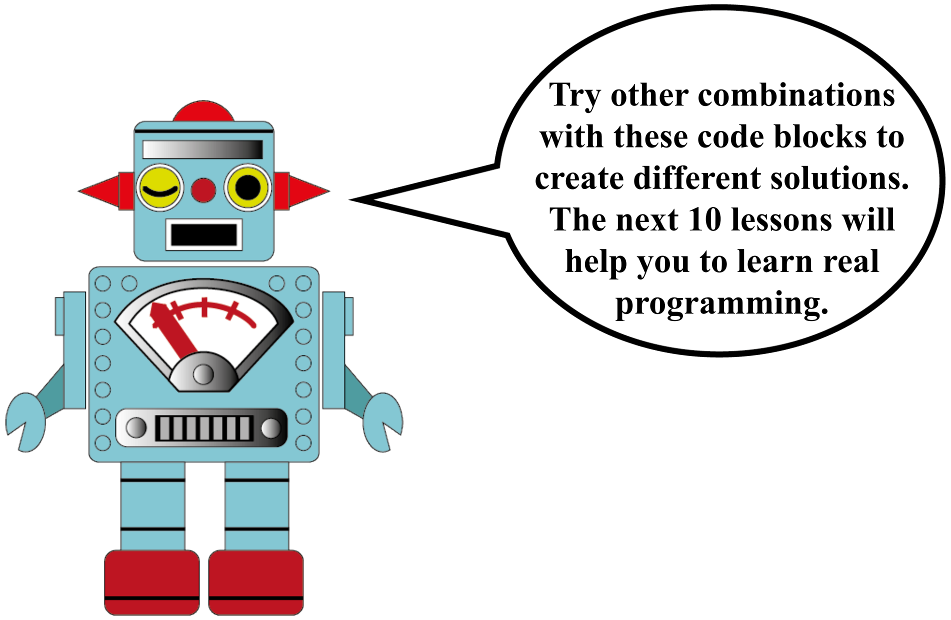
Move a motor with a button

Press button D8 and observe what happens.



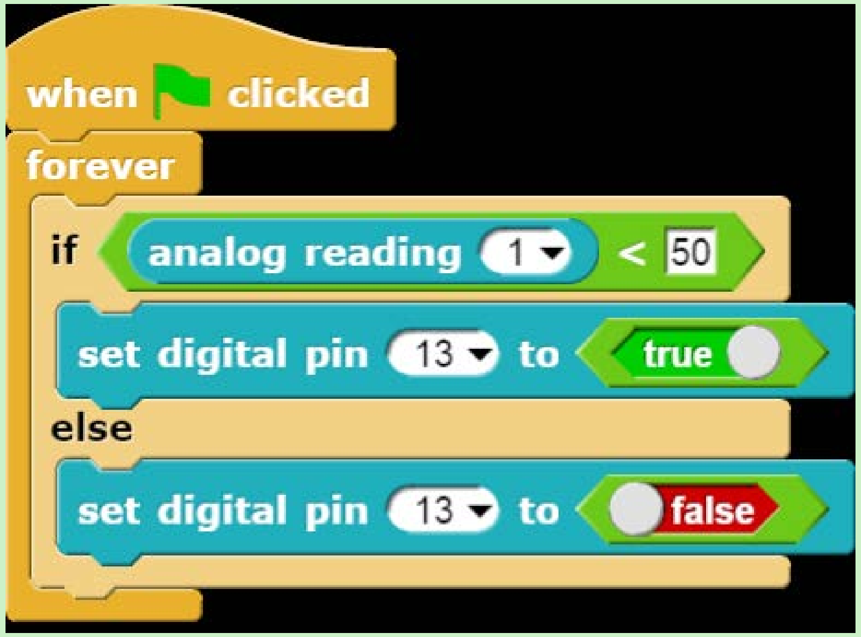
```
when ⚑ clicked
forever
    set servo 5 to 0
    if digital reading 8 = false
        set servo 5 to 180
    else
        set servo 5 to 0
```

Figure 36. Move a servo motor with a button press (Lesson 10 shows how to attach the servo motor)


Try other combinations with these code blocks to create different solutions. The next 10 lessons will help you to learn real programming.

Turn a LED on when it gets dark

Place your finger over the LDR and observe what happens.

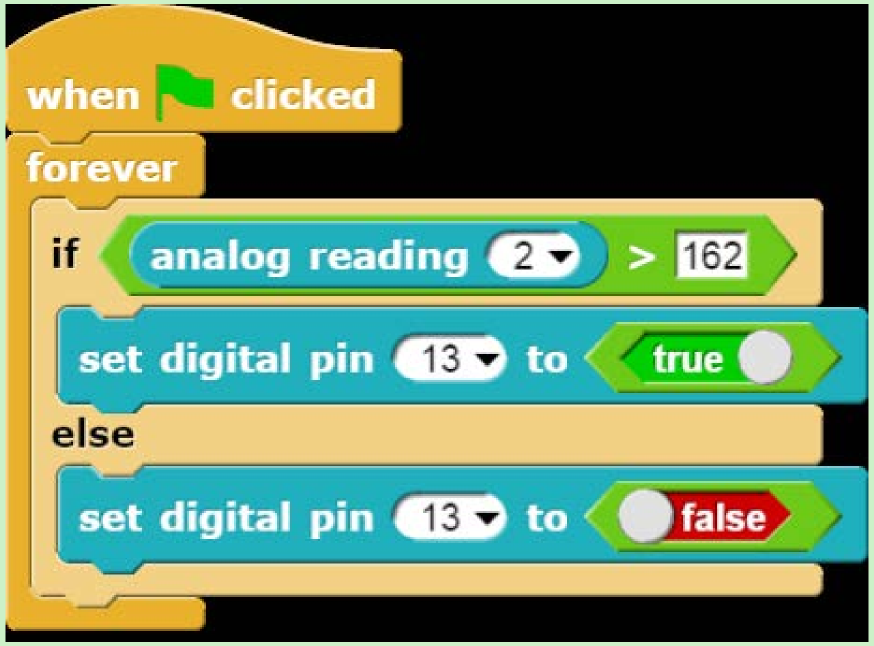


```
when ⚑ clicked
forever
    if analog reading 1 < 50
        set digital pin 13 to true
    else
        set digital pin 13 to false
```

Figure 37. Turn an LED on when the light level falls on the LDR sensor (you may need to adjust the number '50' up or down)

Turn a LED on when it gets hot

Press your finger against the temperature sensor and observe what happens.



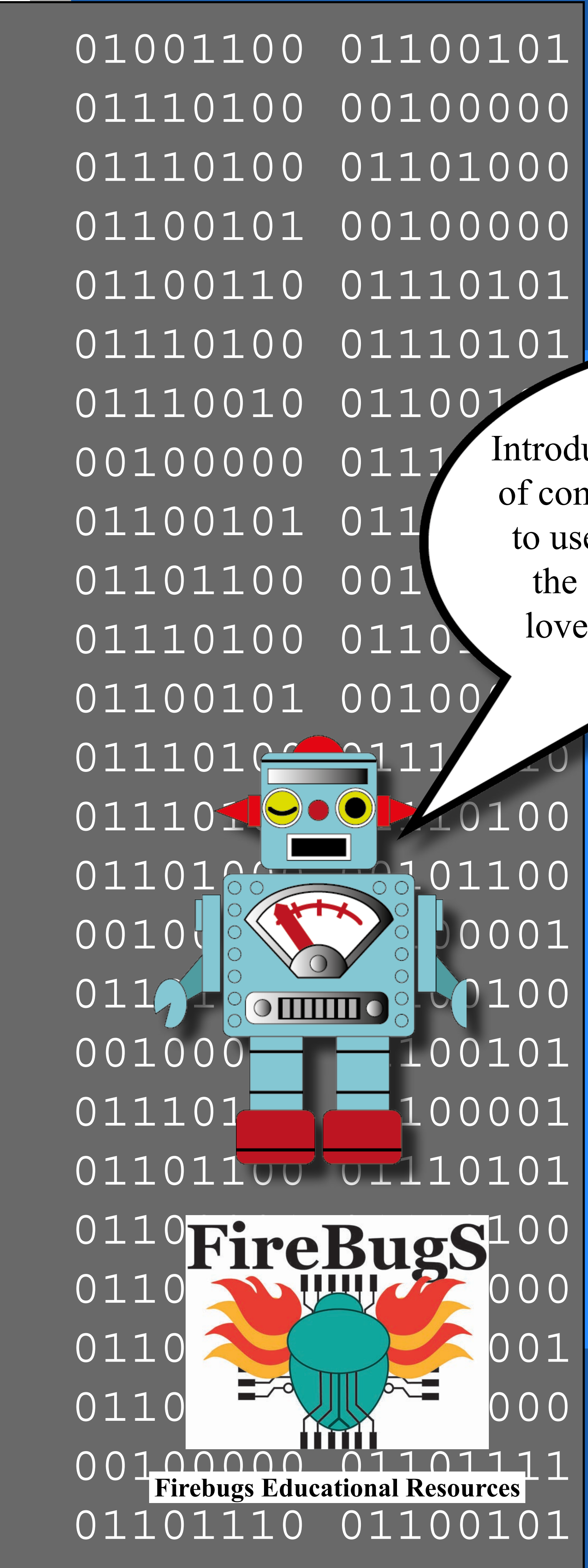
```
when ⚑ clicked
forever
    if analog reading 2 > 162
        set digital pin 13 to true
    else
        set digital pin 13 to false
```

Figure 38. Turn an LED on when the temperature rises above a certain point (you may need to adjust the number '162' up or down a little)

Coding INNOV8ion, it's a Snap! with the ARD2-INNOV8

and Snap4Arduino

1st edition



Introduces the language and concepts of computer programming with easy to use visual – block coding, using the favourite hardware platform loved by engineers and backyard inventors worldwide.

Seven Vinton

Seven has shared his knowledge of ICT and digital systems technology with students and teachers via State Conferences over his 18 year teaching career. He has inspired many students to take up careers in technology areas and supports world-wide sharing of knowledge through his Youtube channel ICT Tools for Teachers. Currently the leader of Curriculum at Oberon High School, Seven has provided support and assistance to help schools and teachers throughout the world implement digital learning into their curriculum.

Authors

Mark Trezise & Seven Vinton

Mark Trezise

At the time of writing this book Mark was studying year 11 at Oberon High School. Mark got into programming via a Code Club at his school, and from there his passion grew and along with his knowledge. Mark has taught digital programming to students around Victoria and interstate. He attended the Questacon Invention Convention in 2016, and was asked back as an ambassador for the 2017 convention. He has been programming for 3 years and has implemented several inventions to make work easier on his farm. Mark has established a large following of learners which he supports via his websites and Youtube channel Trez Tech.

ISBN 978-0-6480792-0-0

To purchase and download the complete version of:

“Coding Innov8ion, it’s a Snap”

go to

www.wiltronics.com.au/ard2