Arduino Blink Challenge

Provided by TryEngineering - www.tryengineering.org

Lesson Focus

Lesson explores computer programming and the impact of computers on society. Students build and test a program to turn a light on and off using an Arduino board. They connect the hardware, program the code, test their system, adapt it for variations in blinking times, evaluate their results, and share observations with their class.

Lesson Synopsis

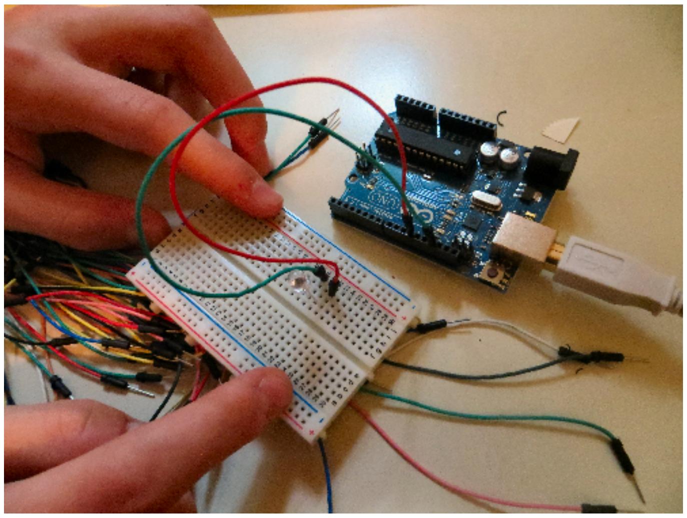
The "Arduino Blink Challenge" lesson explores how computer and software engineers work to solve the challenges of a society, such as providing systems for turning lights on and off automatically. Students work in teams to set up and program an Arduino board to turn a light on and off at a 5 second on and 2 second off interval. Teams build their system, program and test it, reflect on the challenge, and present their experiences to their class.

Age Levels

14-18.

Objectives

- Learn about engineering design and redesign.
- Learn about circuits, computers, and software coding.
- Learn how engineering can help solve society's challenges.
- Learn about teamwork and problem solving.



Anticipated Learner Outcomes

As a result of this activity, students should develop an understanding of:

- engineering design
- computer engineering
- software engineering
- teamwork

Lesson Activities

Students explore how engineers have solved societal problems such as applying computing to turn lights on and off. Students work in teams to set up and program an Arduino board to turn a light on and off at a 5 second on and 2 second off interval. Teams build their system, program and test it, reflect on the challenge, and present their experiences to their class.

Resources/Materials

- Teacher Resource Documents (attached)
- Student Resource Sheet (attached)
- Student Worksheet (attached)

Alignment to Curriculum Frameworks

See curriculum alignment sheet at end of lesson.

Internet Connections

- TryEngineering (www.tryengineering.org)
- TryComputing (www.trycomputing.org)
- Arduino (http://arduino.cc/en/)
- National Science Education Standards (www.nsta.org/publications/nses.aspx)
- ITEA Standards for Technological Literacy (www.iteaconnect.org/TAA)

Recommended Reading

- Beginning Programming All-In-One Desk Reference For Dummies (ISBN: 978-0470108543)
- Hello World! Computer Programming for Kids and Other Beginners (ISBN: 978-1933988498)
- Arduino Cookbook (ISBN: 978-1449313876)
- Programming Arduino Getting Started with Sketches (ISBN: 978-0071784221)

Optional Writing Activity

- Write an essay or a paragraph that identifies three major products or systems that have been overwhelmingly impacted by computer programming.

Optional Extension Activity

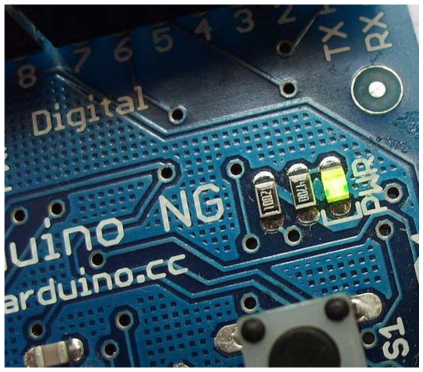
- Require students to make the Arduino blink in the pattern of S-O-S…or have them select three other activities (run a fan, make a stoplight, etc.).

Arduino Blink Challenge

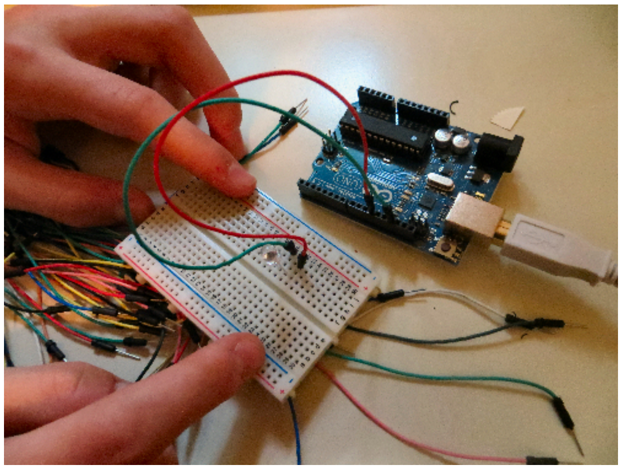
For Teachers:
Teacher Resources

◆ Lesson Goal
The "Arduino Blink Challenge" lesson explores how computer and software engineers work to solve the challenges of a society, such as providing systems for turning lights on and off automatically. Students work in teams to set up and program an Arduino board to turn a light on and off at a 5 second on and 2 second off interval. Teams build their system, program and test it, reflect on the challenge, and present their experiences to their class.

◆ Lesson Objectives
- Learn about engineering design and redesign.
- Learn about circuits, computers, and software coding.
- Learn how engineering can help solve society's challenges.
- Learn about teamwork and problem solving.



◆ Materials
- Student Resource Sheets
- Student Worksheets
- Classroom Materials: computer with internet access (for set up); Arduino Board kit (You can purchase individual items or a starter kit that contains an Arduino Uno, a 3' USB Cable, a solderless breadboard, 65 jumper wires, and a breadboard holder for about $38 US via Amazon; items can also be purchased directly from other electronics suppliers and links are available through Arduino at http://arduino.cc/en/)
- Student Team Materials: Arduino board, connectors, optional breadboard, led lights, fan, insulators.



◆ Procedure
1. Show students the student reference sheets. These may be read in class or provided as reading material for the prior night's homework.
2. To introduce the lesson, consider asking the students how stoplights are timed to change for traffic.
3. Teams of 2 or more students will consider their challenge, and conduct research into how the Arduino operates.
4. Students set up their Arduino, program it and test it until it meets the challenge.
5. Teams reflect on the challenge, and present their experiences to the class.

◆ Time Needed
Two to three 45 minute sessions.

Student/Teacher Resource: Advanced Arduino Applications?

◆ Beyond Flashing Lights

If you have completed the Arduino Blink Challenge, there are many other applications for using an Arduino board that you can try on your own, as a team, in the classroom, or on your own time. You can also find advanced tutorials at http://arduino.cc/en/Tutorial/HomePage to show you how to read a switch, read a potentiometer, use a pushbutton to control an LED, play tones on multiple speakers, make an LED bar graph, send a text string, or even output the values from a barometric pressure sensor as a web page.

The following are some sample ideas so you can see how versatile your new computing skills are!

Control Sensors

Try controlling an external sensor with your Arduino. You can sample air for pollution, identify how bright an area is, set up a flood alarm with a water sensor, or attach a motion detector. Or, consider using the Arduino to control sensors that measure electromagnetic Fields, sample air for humidity levels, take the temperature, identify if a gas is present in the air, or gather data from an anemometer that measures wind speed. You can even attach a bar code scanner (which simulates a keyboard) or a keyboard to the Arduino.

Visual and Audio Applications

Your arduino can be set up to manage a camera system, and control photography settings. This can be done with most equipment, including Nikon, Canon, Sony, Minolta, Olympus and Pentax cameras. You can even attach a bar code scanner (which simulates a keyboard) or a keyboard to the Arduino.

Motors and Robotics

Arduino is a great tool for controlling motors and robotics. Try connecting DC motors or stepper motors. You can control a highly accurate stepper motor using a potentiometer with an Arduino.

◆ Community of Developers

There is a growing commuity of Arduino application developers who share code, ideas, and example. Additional documentation has been created by the Arduino community on the publicly-editable playground wiki at www.arduino.cc/playground.

Student Resource: What is Arduino?

◆ Open Source Computing

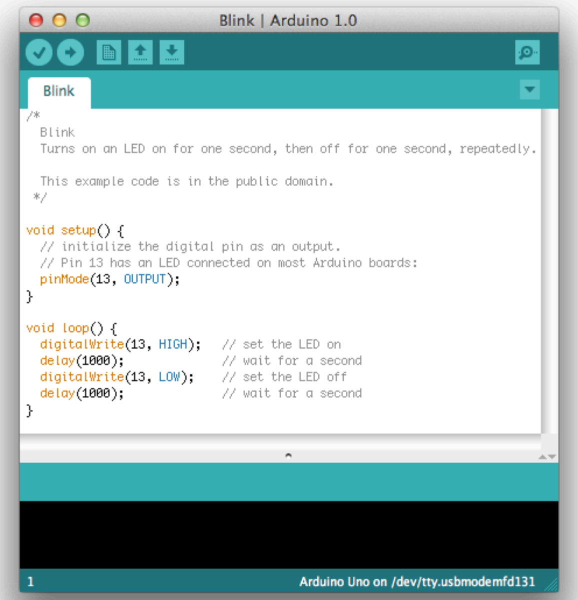
Arduino is an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board. Arduino can be used to develop interactive objects, taking inputs from a variety of switches or sensors, and controlling a variety of lights, motors, and other physical outputs. Arduino projects can be simple -- such as turning on and off a light -- or very complex. The boards can be assembled by hand or purchased preassembled; the open-source software can be downloaded for free. The Arduino software runs on Windows, Macintosh OSX, and Linux operating systems.

◆ Computer Programming

Computer programming (often shortened to programming or coding) is the process of designing, writing, testing, debugging, and maintaining source code of computer programs. The code can be written in many different programming languages. Programming is basically a set of instructions that a computer or other device uses to perform a task -- it might be turning on a light, opening a door, or writing a document.

◆ Arduino Development Environment

The Arduino development environment or software contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions, and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them. Software written using Arduino is called a "sketch." These sketches are written in the text editor. Sketches are saved with the file extension .ino. There are features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino environment including complete error messages and other information. The bottom right hand corner of the window displays the current board and serial port.



```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
 */

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);   // set the LED on
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // set the LED off
  delay(1000);              // wait for a second
}
```

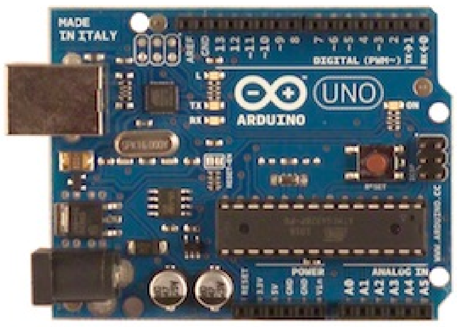
```
1                                Arduino Uno on /dev/tty.usbmodemfd131
```

Note: Some content and images on this page are derived from Arduino.cc via their Arduino getting started guide (http://arduino.cc/en/Guide/HomePage)

Student Resource: Arduino Tutorial -- Step by Step - P 1 of 2

This document explains how to connect your Arduino board to the computer and upload your first sketch. This is provided and available online by Arduino at http://arduino.cc/en/Guide/HomePage. These pages apply to the windows environment, but Arduino works on Mac OS X and Linux and the website provides step by step instructions for all operating systems.

1 | Get an Arduino board and USB cable



In this tutorial, we assume you're using an Arduino Uno, Arduino Duemilanove, Nano, or Diecimila. If you have another board, visit http://arduino.cc/en/Guide/HomePage and read the corresponding page for that product in the getting started guide. You also need a standard USB cable (A plug to B plug): the kind you would connect to a USB printer, for example. (For the Arduino Nano, you'll need an A to Mini-B cable instead.) (Note: TryEngineering.org recommends purchasing kits via Amazon or other retailers which are already bundled with breadboards, lights, cable, or other parts you might use down the road as you explore more advanced Arduino challenges.)

2 | Download the Arduino environment

Get the latest version from the download page.
When the download finishes, unzip the downloaded file. Make sure to preserve the folder structure. Double-click the folder to open it. There should be a few files and sub-folders inside.

3 | Connect the board

The Arduino Uno, Mega, Duemilanove and Arduino Nano automatically draw power from either the USB connection to the computer or an external power supply. If you're using an Arduino Diecimila, you'll need to make sure that the board is configured to draw power from the USB connection. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks. Check that it's on the two pins closest to the USB port. Connect the Arduino board to your computer using the USB cable. The green power LED (labeled PWR) should go on.

4 | Install the drivers

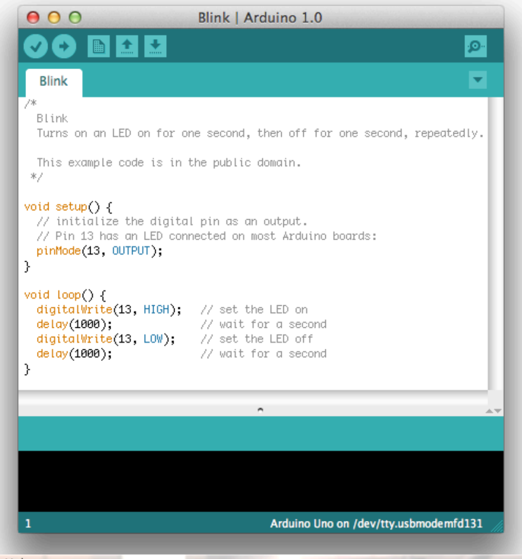
Installing drivers for the Arduino Uno with Windows7, Vista, or XP:
Plug in your board and wait for Windows to begin its driver installation process. After a few moments, the process will fail, despite its best efforts. Click on the Start Menu, and open up the Control Panel. While in the Control Panel, navigate to System and Security. Next, click on System. Once the System window is up, open the Device Manager.
Look under Ports (COM & LPT). You should see an open port named "Arduino UNO (COMxx)"

Arduino Blink Challenge

Student Resource: Arduino Tutorial -- Step by Step - P 2 of 2

4 | Install the drivers (continued)

Right click on the "Arduino UNO (COmxx)" port and choose the "Update Driver Software" option. Next, choose the "Browse my computer for Driver software" option. Finally, navigate to and select the Uno's driver file, named "ArduinoUNO.inf", located in the "Drivers" folder of the Arduino Software download (not the "FTDI USB Drivers" sub-directory). If your software does not allow you to select a specific file, just select the "Drivers" folder and Windows will finish up the driver installation from there. When you connect the board, Windows should initiate the driver installation process (if you haven't used the computer with an Arduino board before).



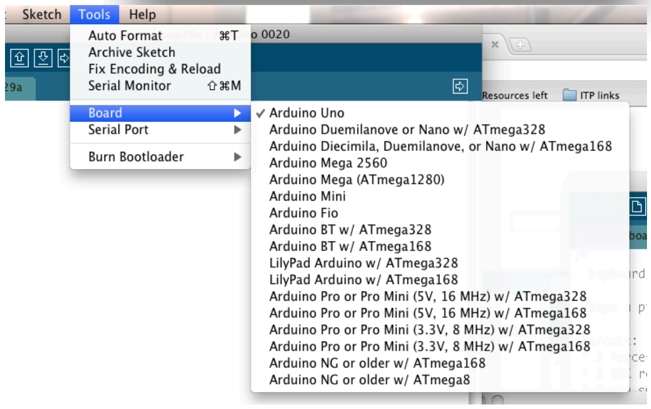
```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
 */

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);   // set the LED on
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // set the LED off
  delay(1000);              // wait for a second
}
```

5 | Launch the Arduino application

Double-click the Arduino application.

6 | Open the blink example

Open the LED blink example sketch: File > Examples > 1.Basics > Blink.


Sketch
Tools
Help
Auto Format ⌘T
Archive Sketch
Fix Encoding & Reload
Serial Monitor ⇧⌘M
Board
Serial Port
Burn Bootloader
✓ Arduino Uno
Arduino Duemilanove or Nano w/ ATmega328
Arduino Diecimila, Duemilanove, or Nano w/ ATmega168
Arduino Mega 2560
Arduino Mega (ATmega1280)
Arduino Mini
Arduino Fio
Arduino BT w/ ATmega328
Arduino BT w/ ATmega168
LilyPad Arduino w/ ATmega328
LilyPad Arduino w/ ATmega168
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168
Arduino NG or older w/ ATmega168
Arduino NG or older w/ ATmega8

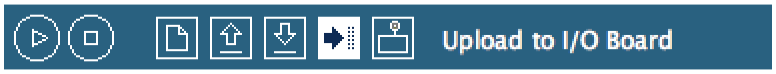
7 | Select your board

You'll need to select the entry in the Tools > Board menu that corresponds to the type of Arduino you are using.

8 | Select your serial port

Select the serial device of the Arduino board from the Tools | Serial Port menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu; the entry that disappears should be the Arduino board. Reconnect the board and select that serial port.

9 | Upload the program



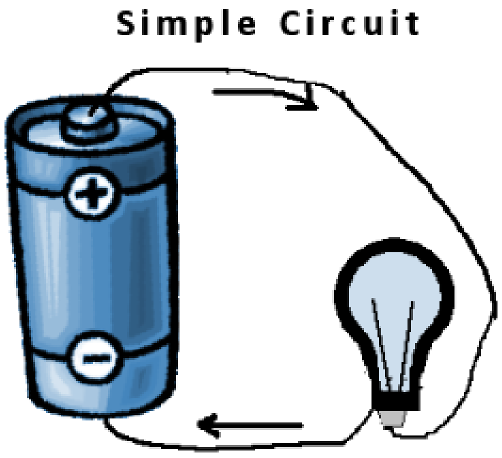
Now, simply click the "Upload" button in the environment which may look like an arrow pointing to the right. Wait a few seconds - you should see the RX and TX LEDs on the board flashing. If the upload is successful, the message "Done uploading." will appear in the status bar. A few seconds after the upload finishes, you should see the pin 13 (L) LED on the board start to blink (in orange). If it does, congratulations! You've gotten Arduino up-and-running.

Note: Some content and images on this page are derived from Arduino.cc via their Arduino getting started guide (http://arduino.cc/en/Guide/HomePage)

Student Resource:
Circuit Basics

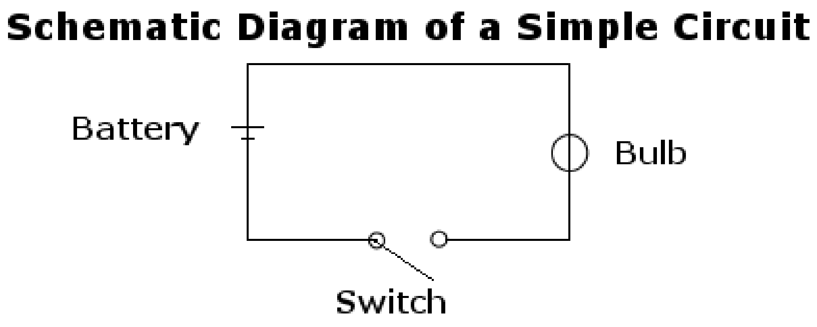
◆ Simple Circuit

A simple circuit consists of three minimum elements that are required to complete a functioning electric circuit: a source of electricity (battery), a path or conductor on which electricity flows (wire) and an electrical resistor (lamp) which is any device that requires electricity to operate. The illustration below shows a simple circuit containing, one battery, two wires, and a bulb. The flow of electricity is caused by excess electrons on the negative end of the battery being attracted to flow toward the positive end, or terminal, of the battery. When the simple circuit is complete, electrons flow from the negative terminal through the wire conductor, then through the bulb (lighting it up), and finally back to the positive terminal - in a continual flow.



◆ Schematic Diagram of a Simple Circuit

The following is a schematic diagram of the simple circuit showing the electronic symbols for the battery, switch, and bulb.

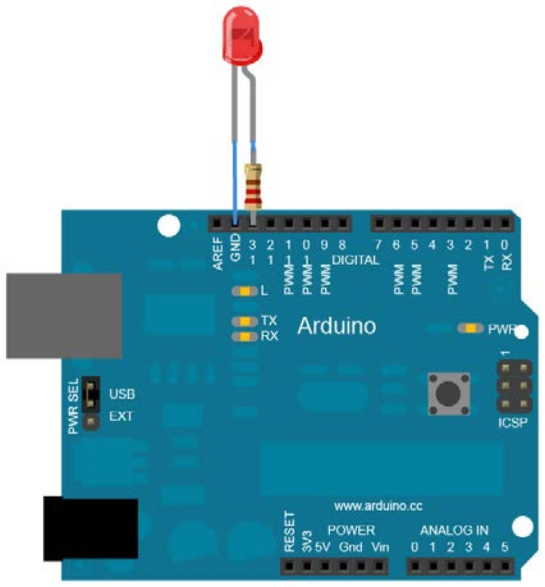


Arduino Blink Challenge

Student Worksheet:

◆ Engineering Teamwork and Planning

You are part of a team of engineers given the challenge of using Arduino to make a light blink on for 5 seconds and off for 2. The instructions and code below will help you see how to program the Arduino to blink for one second on and one second off.

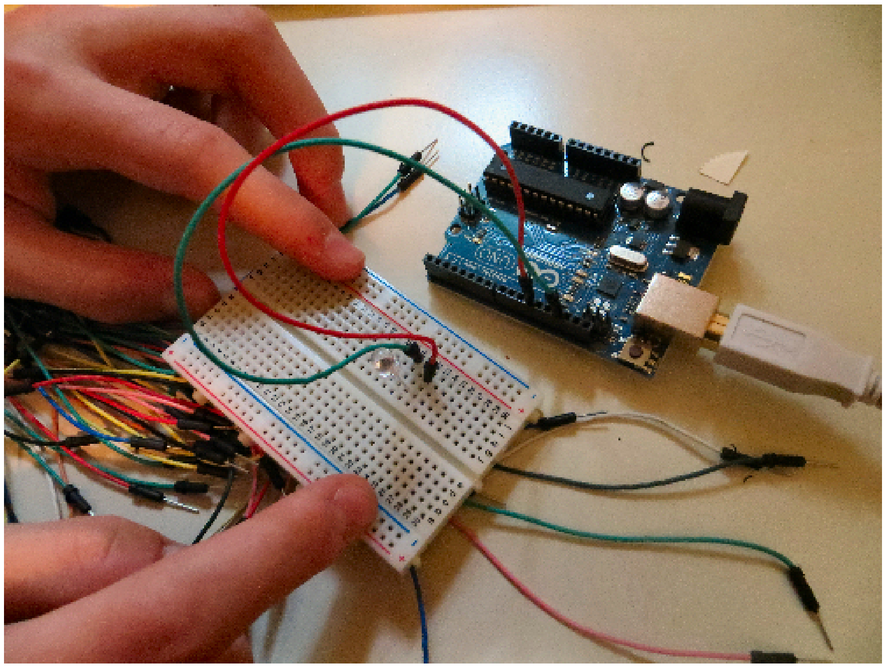


◆ Research Phase

Read the materials provided to you by your teacher. If you have access to the internet ahead of the activity, explore the Arduino website and become familiar with the logic of the programming code.

◆ Building the Circuit

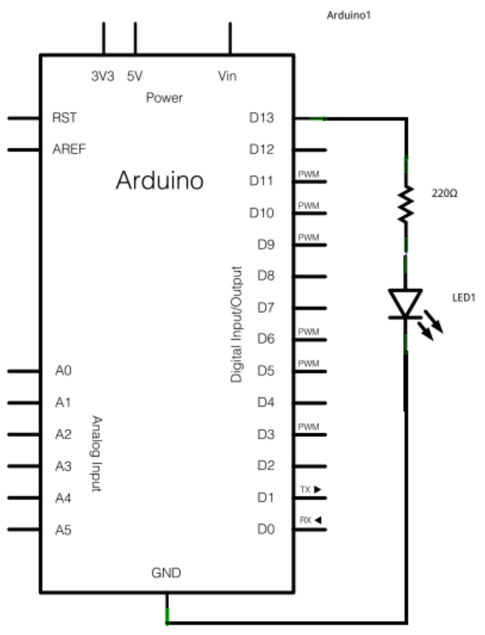
To build the circuit, attach a 220-ohm resistor to pin 13. Then attach the long leg of an LED (the positive leg, called the anode) to the resistor. Attach the short leg (the negative leg, called the cathode) to ground. Then plug your Arduino board into your computer, start the Arduino program, and enter the code below. Note: Most Arduino boards already have an LED attached to pin 13 on the board itself. If you run this example with no hardware attached, you should see that LED blink.



You may also set up your blinking light using a breadboard as in the image above. In this case, you'll use the connectors to link the Arduino to the breadboard and make a complete circuit by having another connector going back to Arduino. You would need to include a separate light on the breadboard too. You can do this lesson either way...with or without the breadboard -- it just depends on what materials you have provided to you and if your team wishes to gain some experience on the breadboard.

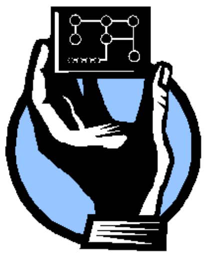
Note: Some content and images on this page are derived from Arduino.cc via their Arduino getting started guide (http://arduino.cc/en/Guide/HomePage)

◆ **Schematic**
The illustration or schematic on the right shows how the circuit for the light works in the Arduino.



◆ **Code**
In the program below, the first thing you do is to initialize pin 13 as an output pin with the line
`pinMode(13, OUTPUT);`
In the main loop, you turn the LED on with the line:
`digitalWrite(13, HIGH);`
This supplies 5 volts to pin 13. That creates a voltage difference across the pins of the LED, and lights it up. Then you turn it off with the line:
`digitalWrite(13, LOW);`
That takes pin 13 back to 0 volts, and turns the LED off. In between the on and the off, you want enough time for a person to see the change, so the `delay()` commands tell the Arduino to do nothing for 1000 milliseconds, or one second. When you use the `delay()` command, nothing else happens for that amount of time.

```
/*
Blink
Turns on an LED on for one second, then off for one second,
repeatedly.

This example code is in the public domain.
*/

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
// initialize the digital pin as an output.
pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage
level)
delay(1000); // wait for a second
digitalWrite(led, LOW); // turn the LED off by making the voltage LOW
delay(1000); // wait for a second
}
```



Student Worksheet:

◆ **Reflection**

Complete the reflection questions below:

1. What challenges did you have, if any, to programming the Arduino to blink on and off at 1 second intervals? How did you resolve any challenges you encountered?

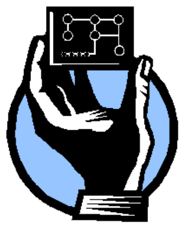
2. Were you able to adjust the code to change the intervals to 5 seconds on and 2 seconds off?

3. Do you think that this activity was more rewarding to do as a team, or would you have preferred to work alone on it? Why?

4. What do you think about the Arduino? Was it a good way for you and your team to explore basic computer programming?

5. How complicated or different do you think code would be to provide instructions to a cell phone to play a particular ringtone?

6. Do you think that stoplights use computer programming to provide a pattern for lights changing at intersections? How do you think stoplights were coordinated prior to computer technology?



For Teachers:
Alignment to Curriculum Frameworks

Note: All lesson plans in this series are aligned to the National Science Education Standards which were produced by the National Research Council and endorsed by the National Science Teachers Association, and if applicable, also to the International Technology Education Association's Standards for Technological Literacy, the Computer Science Teachers Association K-12 Computer Science Standards, or the National Council of Teachers of Mathematics' Principles and Standards for School Mathematics.

◆National Science Education Standards Grades 5-8 (ages 10-14)

CONTENT STANDARD A: Science as Inquiry

As a result of activities, all students should develop

- ✦ Abilities necessary to do scientific inquiry

CONTENT STANDARD B: Physical Science

As a result of their activities, all students should develop an understanding of

- ✦ Properties and changes of properties in matter
- ✦ Transfer of Energy

CONTENT STANDARD E: Science and Technology

As a result of activities in grades 5-8, all students should develop

- ✦ Abilities of technological design
- ✦ Understandings about science and technology

CONTENT STANDARD F: Science in Personal and Social Perspectives

As a result of activities, all students should develop understanding of

- ✦ Risks and benefits
- ✦ Science and technology in society

CONTENT STANDARD G: History and Nature of Science

As a result of activities, all students should develop understanding of

- ✦ Science as a human endeavor
- ✦ History of science

◆National Science Education Standards Grades 9-12 (ages 14-18)

CONTENT STANDARD A: Science as Inquiry

As a result of activities, all students should develop

- ✦ Abilities necessary to do scientific inquiry

CONTENT STANDARD B: Physical Science

As a result of their activities, all students should develop understanding of

- ✦ Interactions of energy and matter

CONTENT STANDARD E: Science and Technology

As a result of activities, all students should develop

- ✦ Abilities of technological design
- ✦ Understandings about science and technology

CONTENT STANDARD F: Science in Personal and Social Perspectives

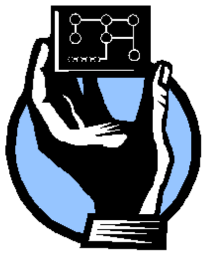
As a result of activities, all students should develop understanding of

- ✦ Science and technology in local, national, and global challenges

CONTENT STANDARD G: History and Nature of Science

As a result of activities, all students should develop understanding of

- ✦ Science as a human endeavor
- ✦ Historical perspectives



For Teachers: Alignment to Curriculum Frameworks (cont.)

◆ CSTA K-12 Computer Science Standards (Grades 6-9) (ages 11-15)

Computational Thinking:

- ✦ 13. Understand the notion of hierarchy and abstraction in computing including high level languages, translation, instruction set, and logic circuits.

Collaboration:

- ✦ 1. Apply productivity/multimedia tools and peripherals to group collaboration and support learning throughout the curriculum.
- ✦ 3. Collaborate with peers, experts, and others using collaborative practices such as pair programming, working in project teams, and participating in group active learning activities.
- ✦ 4. Exhibit dispositions necessary for collaboration: providing useful feedback, integrating feedback, understanding and accepting multiple perspectives, socialization.

Computing Practice & Programming:

- ✦ 1. Select appropriate tools and technology resources to accomplish a variety of tasks and solve problems.
- ✦ 2. Use a variety of multimedia tools and peripherals to support personal productivity and learning throughout the curriculum.
- ✦ 5. Implement problem solutions using a programming language, including: looping behavior, conditional statements, logic, expressions, variables, and functions.
- ✦ 8. Demonstrate dispositions amenable to open ended problem solving and programming (e.g., comfort with complexity, persistence, brainstorming, adaptability, patience, propensity to tinker, creativity, accepting challenge).
- ✦ 9. Collect and analyze data that is output from multiple runs of a computer program.

Computers & Communications Devices:

- ✦ 1. Recognize that computers are devices that execute programs.
- ✦ 3. Demonstrate an understanding of the relationship between hardware and software.

◆ CSTA K-12 Computer Science Standards (Grades 9-10) (ages 14-16)

Computational Thinking:

- ✦ 2. Describe a software development process used to solve software problems (e.g., design, coding, testing, verification).

Collaboration:

- ✦ 1. Work in a team to design and develop a software artifact.
- ✦ 4. Identify how collaboration influences the design and development of software products.

Computing Practice & Programming:

- ✦ 3. Use various debugging and testing methods to ensure program correctness (e.g., test cases, unit testing, white box, black box, integration testing)
- ✦ 4. Apply analysis, design, and implementation techniques to solve problems (e.g., use one or more software lifecycle models).
- ✦ 8. Explain the program execution process.

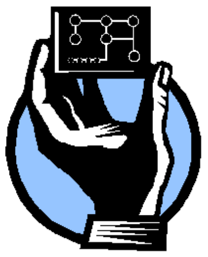
Computers & Communications Devices:

- ✦ 4. Compare various forms of input and output.

◆ CSTA K-12 Computer Science Standards (Grades 10-12) (ages 16-18)

Collaboration:

- ✦ 3. Evaluate programs written by others for readability and usability.



For Teachers:
Alignment to Curriculum Frameworks (cont.)

◆Standards for Technological Literacy - All Ages

The Nature of Technology

- ✦ Standard 3: Students will develop an understanding of the relationships among technologies and the connections between technology and other fields of study.

Technology and Society

- ✦ Standard 4: Students will develop an understanding of the cultural, social, economic, and political effects of technology.
- ✦ Standard 6: Students will develop an understanding of the role of society in the development and use of technology.
- ✦ Standard 7: Students will develop an understanding of the influence of technology on history.

Design

- ✦ Standard 8: Students will develop an understanding of the attributes of design.
- ✦ Standard 9: Students will develop an understanding of engineering design.
- ✦ Standard 10: Students will develop an understanding of the role of troubleshooting, research and development, invention and innovation, and experimentation in problem solving.

Abilities for a Technological World

- ✦ Standard 11: Students will develop abilities to apply the design process.
- ✦ Standard 12: Students will develop abilities to use and maintain technological products and systems.
- ✦ Standard 13: Students will develop abilities to assess the impact of products and systems.

The Designed World

- ✦ Standard 17: Students will develop an understanding of and be able to select and use information and communication technologies.