Introduction to programming the littleBits Arduino

Year level band: 7-8

Description:
This project introduces students to the littleBits kit and to how it can be programmed using the Arduino IDE software.

Type: General purpose programming language

Resources:
- littleBits Arduino Coding Kit
- littleBits Arduino Coding Kit Invention Guide (included with the kit)
- Mac or PC with latest Arduino IDE installed:
- Spare 9V batteries
- Phillips-head screwdrivers (possibly not required in introduction lesson)

Prior Student Learning:
A basic understanding of circuits is useful.
An understanding of general programming concepts - input and output, algorithms, loops and debugging.
It's useful to determine if any of the class are familiar with littleBits and Arduino, ask questions such as:
- What is an Arduino?
- Who has used littleBits before
- Who has heard of a microcontroller?

<table>
<thead>
<tr>
<th>Digital Technologies Summary</th>
<th>This activity introduces students to programming in the Arduino language through exploration of the littleBits Arduino bit.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter digital technologies summary specific to the level your lesson is tailored for.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Band</th>
<th>Content Descriptors</th>
</tr>
</thead>
<tbody>
<tr>
<td>7-8</td>
<td>Design the user experience of a digital system, generating, evaluating and communicating alternative designs (ACTDIP028)</td>
</tr>
<tr>
<td></td>
<td>Design algorithms represented diagrammatically and in English, and trace algorithms to predict output for a given</td>
</tr>
</tbody>
</table>
input and to identify errors (ACTDIP029)
Implement and modify programs with user interfaces involving branching, iteration and functions in a general-purpose programming language (ACTDIP030)

The particular elements of Critical and Creative Thinking addressed by this content description
Inquiring – identifying, exploring and organising information and ideas
- Identify and clarify information and ideas
- Organise and process information
Generating ideas, possibilities and actions
- Consider alternatives
- Seek solutions and put ideas into action
- Imagine possibilities and connect ideas
Analysing, synthesising and evaluating reasoning and procedures
- Apply logic and reasoning
Reflecting on thinking and processes
- Transfer knowledge into new contexts

<table>
<thead>
<tr>
<th>Element</th>
<th>Summary of tasks</th>
</tr>
</thead>
</table>
| Learning hook    | **A.** For students have not used littleBits before (if they are generally familiar with littleBits, proceed to B)  
We are going to use an Arduino Bit to add interactivity to our littleBits prototypes. First we need to explore what Bits we have and how they work and interact.  
You may want to go through the littleBits basics with the class (p2-3 in the Invention Guide): Anatomy, Color-coded categories, Magnets, Order of Bits.  
Ask students to put the Arduino Bit to one side and explore the other Bits so they know their features and functions. Pages 8-13 introduce all the available Bits and how they can interact.  

**B.** The littleBits Arduino Bit is a great way to add complexity to littleBits circuits - you can add logic and timing options to create projects not possible without computer control.  
We are going to learn to program the littleBits Arduino Bit and begin exploring how to control the outputs (bargraph and servo motor) with different inputs (button and dimmer ).  

_You may want to introduce Arduino more generally as a prototyping platform. In simple terms, an Arduino is a little computer that you can program - it_
interacts with the world through sensors, lights, buzzers, and motors.

Some example videos of Arduino creations in action can be found here: [http://www.makeuseof.com/tag/arduino-technology-explained/](http://www.makeuseof.com/tag/arduino-technology-explained/)


Introduce the concept of functions and discuss how arduino has pre-defined functions for all the LilyPad components. Some of these functions include:

- loop - for repetition
- delay - delays action
- pinMode - for configuring the specified pin to behave either as an input or an output
- digitalWrite - for writing a low or high value to a pin

A full documentation and description of these functions is available at:


| Achievement Standards | Students plan and manage digital projects to create interactive information. They define and decompose problems in terms of functional requirements and constraints. Students design user experiences and algorithms incorporating branching and iterations, and test, modify and implement digital solutions. They evaluate information systems and their solutions in terms of meeting needs, innovation and sustainability. They analyse and evaluate data from a range of sources to model and create solutions. |
| Learning Map (Sequence) | Students explore the components of the littleBits Arduino Kit
Students run existing code and modify it to achieve planned outcomes with their designed circuits.
Students create new code and learn to debug to achieve outcomes
Students identify the ways in which the algorithm can be decomposed into functions
Students identify the key arduino functions and learn how to change parameters and use these these functions to achieve planned outcomes
Students work in teams to solve problems and improve their coding skills
Students reflect on their work and make suggestions for improvements or extensions |
| Learning input | 1. Ask students to look at their Arduino Bits and highlight the input and output connections. Remember the colour coding - input pink - output green. |
2. Explain that inputs can be digital or analogue. Digital has two states (on/off, high/low, or true/false) but analogue can take values in between. In our kits the button is digital and the dimmer is analogue - and they need to use the appropriate pins.

3. Now we are going to set up the Arduino software so we can run our first simple program. Demonstrate how to connect the Arduino Bit to the computer via micro USB. Open the Arduino software and show how to select the board and port. You will need to select the 'littleBits w6 Arduino Module (Leonardo)' board. You may need to select the port as well - if you are using a PC, the correct serial port is usually the highest numbered “COM” port.
4. Open the example program 'Blink'

Highlight the code (Arduino uses a programming language called C) used in 'Blink'; show where the LED is defined; explain that HIGH and LOW refer to the voltage and is how outputs are turned on and off.

Explain that the Arduino Bit does not have a built-in LED so we will need to change the pin mode. In this example we will use the bargraph Bit as the LED output on Pin 5.
Change the code so that `LED_BUILTIN` is replaced by 5.

```c
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

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

Verify your code (click the tick mark) and upload it to your Arduino Bit (click the arrow mark).

5. Identify the key blocks of the code, in particular the setup() and loop() functions, and discuss their purpose. What about the delay() function? What other functions of interest can students find?

6. Highlight the loop and ask students how we could change the code to make it blink faster or slower. What about blinking in an uneven pattern?

7. Make a change to your code, e.g. `delay(500)` and show how to save your new program with a different name using "File > Save As...".

8. It is also a good idea to change any comments in the code - for your reference and it explains what you are doing if sharing with others.

| Learning construction | Students work in pairs to program their littleBits Arduino Bit. The activity is about experimenting, trying new solutions, and debugging. Encourage students to help each other - and look for help on the internet. Ask a friend. Ask Google. Then ask the teacher. |
A - considering outputs

Some example questions for students to explore...

1. Load the 'Blink' example program and run it on your Arduino.
2. Students write the pseudocode of this algorithm showed in the program, focusing on how to decompose their algorithm into the key functions such as initializing the pins/board, writing a value to a pin and delaying.
3. Can you identify the key functions in the code?
4. Can you make the LED blink faster or slower? What is the fastest it can go? Will the code work with decimal numbers?
5. Can you make the servo motor turn on and off? What effect do the swing and turn modes have on the servo? Flick the switch on the servo Bit and note what you observe.
6. Add a fork so that Pin 5 has two different outputs.
7. Can you use two output Pins and alter the code so that the bargraph and servo are on at different times?
8. Load the 'Fading' example sketch (File > Examples > 03.Analog > Fading). Can you describe what the code does? What changes could you make and what would the effects be on the bargraph and/or servo motor.

B - considering inputs

We have only considered outputs so far - students may want to start considering inputs by themselves. The built in example 'Button' (Examples> 02.Digital> Button) would be a good start pointing point.

You will need to use Pin 0 for the input (Button) as this is the only digital input. Change the code in the Button example sketch so that the constants correspond to your inputs and output pins:

```cpp
// constants won't change. They're used here to
// set pin numbers:
const int buttonPin = 0;  // the number of the pushbutton pin
const int ledPin = 1;     // the number of the LED pin
```

Some example questions:

1. Can you code your Arduino so that the bargraph is always on, but turns off when you press the button?
2. Can you code your Arduino so that the bargraph is always on, but the servo only moves when you press the button?
3. Does it matter where you place the output? For example, would the following work?
### Learning demo

While students are working in pairs, ask questions to give them the opportunity to demonstrate their thinking and understanding:

- What have you changed in this code and why?
- What do you think are the most important functions?
- Have you had to do any debugging?
- Can you think of an alternative way to achieve the same outcome?

Check that students are adding comments in their code - and ask them to share with another to group to sense check.

### Learning reflection

**A. Reflect on students’ experiences modifying and writing code.**

- What challenges did you have when programming the littleBits Arduino? How did you resolve those?
- Were you able to easily change existing code to produce a different outcome?
- Did you learn any new terminology?
- What were the rewarding parts of coding in pairs?
- What are the advantages/disadvantages of using functions in code?
- Did they discover an interesting/useful function they want to share?

**B. Remind students that littleBits is a prototyping platform but that the code and circuits we have been exploring exist in real-world products.**

- What real-world situations would you expect similar code to be used?
- Perhaps give the example of traffic lights - how do you think these worked prior to digital technology?
- Can you think of any exciting products that could be created with this sort of technology?
- What other components could be used as inputs or outputs? Consider what digital and analogue inputs and outputs you see each day at school, at home, in transport, sports grounds, etc..
**Assessment:**
Observation of students building littleBits circuits
Successful uploading of example sketches
Successful uploading of modified code - behaviour of littleBits circuit reflects what is commented in the code

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Quantity of knowledge</th>
<th>Quality of understanding</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pre-structural</td>
<td>Uni-structural</td>
</tr>
<tr>
<td>Algoritms</td>
<td>Example code can be run.</td>
<td>Example code can be changed, e.g. delay or pin number.</td>
</tr>
<tr>
<td>Programming</td>
<td></td>
<td>A consistent and executable program is written, showing knowledge of inputs and outputs.</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>No specific / technical terms used.</td>
<td>The terms program or code may be used as a general description. The terms analogue and digital are known and used correctly.</td>
</tr>
</tbody>
</table>

**Teacher/Student Instructions:**
When you turn on the power bit it can take a few seconds for the Arduino bit to start up - wait for the flashing light on the Arduino bit and then you are ready to go.

If using the mounting board:
- Snap your circuit together before pressing into the mounting board
- Press down Bitsnaps (the coloured edges) rather than the white circuit board

**CSER Professional Learning:**

This lesson plan corresponds to professional learning in the following CSER Digital Technologies MOOCs:

**7 & 8 Digital Technologies: Next Steps**

Unit 2 - Next Steps 7 & 8


**Further Resources:**

1. Information about the littleBits Arduino and some example projects: [https://littlebits.cc/bits/w6-arduino](https://littlebits.cc/bits/w6-arduino)

2. littleBits Arduino sketches - eight example sketches to get started with: [http://littlebits.cc/arduino-sketches](http://littlebits.cc/arduino-sketches)

3. This instructable will enable you to code a simple digital and analogue circuit as well as provide a starter sketch that you can use to get started quickly and also to debug your circuit: [http://www.instructables.com/id/Arduino-At-Heart-littlebits-Quick-Start/?ALLSTEPS](http://www.instructables.com/id/Arduino-At-Heart-littlebits-Quick-Start/?ALLSTEPS)

This work is licensed under a [Creative Commons Attribution-NonCommercial 4.0 International License](https://creativecommons.org/licenses/by-nc/4.0). Computer Science Education Research (CSER) Group, The University of Adelaide.