Coding using Microbits - Python — Reflections

Module 1: Design & Making with Microbit - Python

This module introduces the microbit as a piece of hardware that has a specific size and weight, and generally must be supported and incorporated as an essential component of a tangible artifact. Focus on making a pet or robot and incorporating the physical microbit as the face of the project.

Module 1: Project MicroRobot

Questions, Interview, and Responses

_____

_____

_____

_____

_____

_____

Sketches MicroRobot

 C Lyman July 2019

01.2a Name Display Activity

LIst steps to create a microbit program and install it.

01.2b Icons Display Activity

Here are a list of predefined (Image.XXX) icons:

Image.HEART	Image.CLOCK5,	Image.COW
Image.HEART_SMALL	Image.CLOCK4,	Image.MUSIC_CROTCHET
Image.HAPPY	Image.CLOCK3,	Image.MUSIC_QUAVER
Image.SMILE	Image.CLOCK2,	Image.MUSIC_QUAVERS
Image.SAD	Image.CLOCK1	Image.PITCHFORK
Image.CONFUSED	Image.ARROW_N,	Image.XMAS
Image.ANGRY	Image.ARROW_NE,	Image.PACMAN
Image.ASLEEP	Image.ARROW_E,	Image.TARGET
Image.SURPRISED	Image.ARROW_SE,	Image.TSHIRT
Image.SILLY	Image.ARROW_S,	Image.ROLLERSKATE
Image.FABULOUS	Image.ARROW_SW,	Image.DUCK
Image.MEH	Image.ARROW_W,	Image.HOUSE
Image.YES	Image.ARROW_NW	Image.TORTOISE
Image.NO	Image.TRIANGLE	Image.BUTTERFLY
Image.CLOCK12,	Image.TRIANGLE_LEFT	Image.STICKFIGURE
Image.CLOCK11,	Image.CHESSBOARD	Image.GHOST
Image.CLOCK10,	Image.DIAMOND	Image.SWORD
Image.CLOCK9,	Image.DIAMOND_SMALL	Image.GIRAFFE
Image.CLOCK8,	Image.SQUARE	Image.SKULL
Image.CLOCK7,	Image.SQUARE_SMALL	Image.UMBRELLA
Image.CLOCK6,	Image.RABBIT	Image.SNAKE

LIst 2 ways to use the `display._____` command to view the LED display screen.

_____

_____

0.1.2c Icon Animation Activity

Put at least 4 icons together to tell a story. Write each line of the story below.

_____

_____

_____

_____

_____

_____

01.2d Creative Design Activity

Program individual LEDs using the (Sketch LED face designs for your project)

01.3 Project

Microbit Project Faces (Sketch LED face designs for your project)

List the steps to create your robot face animation project.

Reflection

Summarize the feedback from your partner. __

How would you revise your design, if you were to go back and create another?

What was it like designing a project?

Was it a project you enjoyed? Why or why not?

What would you do to redesign the project?

What was it like to interview your partner? What was it like to be listened to?

What was something that surprised you about the process of designing a micro:project?

Describe a difficult point in the process of designing a micro:project and how you resolved it?

Rubric

For creative projects such as these, we normally don't use a qualitative rubric to grade the creativity or the match with their partner's needs. We just check to make sure that the micro:project meets the required specifications:

- Program properly downloaded to microbit
- microbit supported so the face is showing
- microbit can be turned on and off without taking critter apart
- Turned in notes on interview process
- Written reflection

Notes

Coding using Microbits - Python — Reflections

Module 2: Software & Hardware (Algorithms)

This module introduces a conceptual framework for thinking of a computing device as something that uses code to process one or more inputs and send them to an output(s). Questions to be answered include: What is a computer? What is a microbit and what can it do? Students will be making projects that utilize the sensors and screen output of the micro:bit.

What computers are in your house?

_____

_____

_____

What are the parts of a computer? List examples of each.

Blackbox - What are the parts in a Blackbox?

⇒		⇒

What are some examples of Blackboxes in society?

_____

_____

_____

_____

02.2a Sensors Temperature Activity

Pseudocode for Sensors Temperature program

Button A Press Event

What is the code for a button A pressed event?

What comments should be included at the top of a program?

How could your microbit be calibrated to reflect the current room temperature?

What are some other **sensor** inputs on the microbit?

02.2b Sensors Temperature & Compass Activity

Add a button B pressed event to your Sensor code to find the compass direction.
Pseudocode to add a compass to the Sensors Temperature program

02.2c Accelerometer Tilt Activity

Code different tilt events and display an arrow or a word showing the tilt direction.
Pseudocode to create a tilt events program. (See Python Microbit Notes)

02.3 Project: Blackbox

In this project you will plan, design, and create your own Blackbox using different events, microbit sensors, output to LEDs, and MakeCode's block programming. You will also use a maker elements as part of you design and construction.

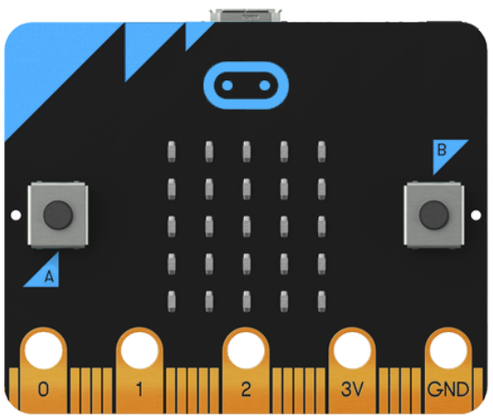
Brainstorm Ideas

Project: _____

Description: _____

_____

Project Sketch:



Blackbox Algorithm & Pseudocode

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Materials Needed: _____

_____

Coding Plan: _____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Notes & Reflections

Beta Testing: _____

_____

Revision Ideas: _____

_____

Photos:

Assessment Rubric - Competency scores

Competencies	4	3	2	1
Inputs	At least 4 different inputs are successfully implemented.	At least 3 different inputs are successfully implemented.	At least 2 different inputs are successfully implemented.	Fewer than 2 different inputs are successfully implemented.
Outputs	At least 4 different outputs are successfully implemented.	At least 3 different outputs are successfully implemented.	At least 2 different outputs are successfully implemented.	Fewer than 2 different outputs are successfully implemented.
Micro:bit Program	micro:bit program: 1) uses event handlers in a way that is integral to the program 2) compiles and runs as intended 3) includes meaningful comments	micro:bit program lacks 1 of the required elements.	micro:bit program lacks 2 of the required elements.	micro:bit program lacks all of the required elements.
Collaboration Reflection	Reflection piece includes: 1) brainstorming ideas 2) construction 3) programming 4) beta testing	Reflection piece lacks 1 of the required elements.	Reflection piece lacks 2 of the required elements.	Reflection piece lacks 3 of the required elements.

Notes

Python Microbits Code Notes

Events Code

Here is a link to all the possible button and gesture events associated with the microbit. Any section(s) can be copied and used. This code is meant to make it is easier to get started using events on the microbit.

```
# 2.0 Event Structures
# by C Lyman
# July 2019
# Module 2 of Coding & Innovation using Microbits - Python
# Structures for different events using Microbits

from microbit import *

# forever loop for Events
while True:
     # Event - button A pressed?
     if button_a.is_pressed():
          # action when A is pressed
          display.show("A")

     # Event - button B pressed?
     if  button_b.is_pressed():
          # action when B is pressed
          display.show("B")

     # Event - buttons AB pressed?
     if  button_a.was_pressed() and button_b.was_pressed():
          # action when A&B are pressed
          display.scroll("AB")

     # Event - pin0 touched?
     if  pin0.is_touched():
          # action when pin0 & ground are touched
          display.show("0")

     # Event - pin1 touched?
     if  pin1.is_touched():
          # action when pin1 & ground are touched
          display.show("1")

     # Event - pin2 touched?
     if  pin2.is_touched():
          # action when pin2 & ground are touched
          display.show("2")

     # Event gesture face up
     faceUp = accelerometer.was_gesture("face up")
          if faceUp:
          display.scroll("UP")
```

```
# Event gesture face down
     faceDown = accelerometer.was_gesture("face down")
     if faceDown:
          display.scroll("DWN")

     # Event gesture shake
     shake = accelerometer.was_gesture("shake")
     if shake:
          display.scroll("SHK")

     # Event gesture up
     up = accelerometer.was_gesture("up")
     if up:
          display.scroll("^")

     # Event gesture down
     down = accelerometer.was_gesture("down")
     if down:
          display.show("v")

     # Event gesture right
     right = accelerometer.was_gesture("right")
     if right:
          display.show(">")

     # Event gesture left
     left = accelerometer.was_gesture("left")
     if left:
          display.show("<")

     # Event - freefall?
     freefall = accelerometer.was_gesture("freefall")
     if freefall:
          # action when microbit is in freefall
          display.scroll("FF")

     # Event - 3g?
     threeG = accelerometer.was_gesture("3g")
     if threeG:
          #6g & 8g are also options
          # action when microbit is accelerated at 3G
          display.scroll("3G")
```

Sensors Code

There a many sensors that can be used as input from the microbit. Below are snippets of code that can be used to access the sensors using MicroPython. For the most part they return a numeric value that can be displayed on the LED screen or used in a calculation.

```
# 2.0 Sensors
# by C Lyman
# April 2019
# Module 2 of Coding & Innovation using Microbits - Python
# Code for different sensors using Microbits

from microbit import *

while True:
# Sensor code lines and examples with value stored in a variable

# Temperature in Celsius
# Code: temperature()
# Example:
    temp = temperature()

# Light level from the display 0-255
# Code: display.read_light_level()
# Example:
    light = display.read_light_level()

# Acceleration x - tilting left - right +
# 0 when flat facing up
# Code: accelerometer.get_x()
# Example:
    accelX = accelerometer.get_x()

# Acceleration y - tilting forward + back -
# 0 when flat facing up
# Code: accelerometer.get_y()
# Example:
    accelY = accelerometer.get_y()

# Acceleration z - moving up + down -
# -1024 when flat face up (Gravity acting downwards)
# 1024 when face down
# vigorous movement will get values +-2048
# Code: accelerometer.get_z()
# Example:
    accelZ = accelerometer.get_z()

# Acceleration all axes
# Code: accelerometer.get_values()
# Example:
    accelXYZ = accelerometer.get_values()
```

```
# Compass Calibrate
# The compass must be calibrated before it can be used.
# The microbit asks you to "tilt until the screen is filled".
# When that is completed a smiley face shows on the screen.
# Then the compass will work.
# Code: compass.calibrate()
# Example:
    compass.calibrate()

# Compass Heading
# Gives a compass degrees for the direction top of the microbit
# (away from the pins) is pointed. 0 or 360 North, 90 East,
# 180 South, and 270 West.
# Code: compass.heading()
# Example:
    compassHeading = compass.heading()

# Compass x gives a magnetic field strength reading in nano tesla
# Code: compass.get_x()
# Example:
    magnetismX = compass.get_x()

# Compass Strength gives an indication of the magnitude
# of the magnetic field strength around the device in nano tesla
# Code: compass.get_field_strength()
# Example:
    magnetismStrength = compass.get_field_strength()
```

Coding using Microbits - Python — Reflections

Module 3: Everything Counts (Variables)

Computer programs process information. Some of the information that is input, stored, and used in a computer program as values that vary or change during the running of a program. Programmers create variables to hold the value of information that may change. In a game program, a variable may be created to hold the player's current score, since that value would change (hopefully!) during the course of the game. Students will be making projects like a people counter, pedometer, score keeper, and/or dice roll.

Module 3: Everything Counts (Variables)

List constants and variables in your life.

_____

_____

_____

Kinds of variables (list examples):

number: _____

string: _____

boolean: _____

list: _____

Play **Newspaper Toss** with a team:

Game 1	Score
Team 1	
Team 2	

Game 2	Score
Team 1	
Team 2	

 July 2019

Rules for naming variables and identifiers:

- Use descriptive names
- Start with lowercase letters
- Only use letters (a-z) and numbers (0-9). **No** spaces or symbols
- Use camelCase when putting 2 words together
- An underscore "_" can also be used to connect words
- Constants are done in all CAPS
- Math operators: +, -, *, /, %, **, //

What is the assignment operator and how does it work?

03.2a People Counter Activity

Pseudocode to create a People Counter program.

03.2b Score Keeper Activity

Pseudocode to create a Score Keeper program.

Modifications to People Counter or Score Keeper programs

03.3 Project: Everything Counts

In this project you will plan, design, and create something that counts. It should keep track of input by storing values in variables, process the input, and output in some visual and useful way. It should also perform mathematical operation on the variables to give useful output. It should also use a maker elements as part of the design and construction.

Brainstorm Ideas _____

_____

_____

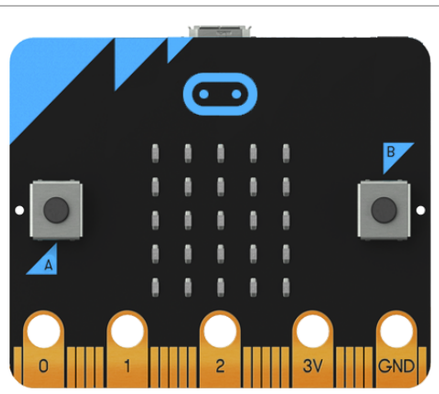
_____

Project: _____

Description: _____

_____

Project Sketch:



Everything Counts Algorithm & Pseudocode

_____

_____

_____

Materials Needed: _____

Coding Plan: _____

_____

_____

_____

_____

_____

_____

_____

_____

Notes & Reflections

What problem were you trying to solve? _____

_____

How well did your prototype work? _____

_____

What did you change? _____

_____

Describe a difficult point and how you resolved it: _____

_____

_____

Photos

Assessment Rubric - Competency scores

Competency	4	3	2	1
Variables	At least 3 different variables are implemented in a meaningful way.	At least 2 variables are implemented in a meaningful way.	At least 1 variable is implemented in a meaningful way.	No variables are implemented.
Variable Names	All variable names are unique and clearly describe what information values the variables hold using CamelCase	The majority of variable names are unique and clearly describe what information values the variables hold.	A minority of variable names are unique and clearly describe what information values the variables hold.	None of the variable names clearly describe what information values the variables hold.
Mathematical Operations	Uses a mathematical operation on at least two variables in a way that is integral to the program.	Uses a mathematical operation on at least one variable in a way that is integral to the program.	Uses a mathematical operation incorrectly or not in a way that is integral to the program.	No mathematical operations are used.
Micro:bit Program	micro:bit program: 1) Uses variables in a way that is integral to the program 2) Uses mathematical operations to add, subtract, multiply, and/or divide variables 3) Compiles and runs as intended 4)Meaningful comments in code.	micro:bit program lacks 1 of the required elements.	micro:bit program lacks 2 of the required element.s	micro:bit program lacks 3 or more of the required elements.
Collaboration Reflection	Reflection piece addresses all prompts.	Reflection piece lacks 1 of the required elements.	Reflection piece lacks 2 of the required elements.	Reflection piece lacks 3 of the required elements.

Coding using Microbits - Python — Reflections

Module 4: Making Decisions (Conditionals)

Computer programs are instructions telling the computer how to process input and deliver output. An important part of programming is telling the computer WHEN to perform a certain task. For this, we use something called ‘conditionals’. Conditionals get their name because a certain Condition or Rule has to be met. Conditionals are usually implemented using an ‘if (condition) then action statement. Students will be creating and making projects like coin toss, Magic 8 Ball, and/or dice toss with dots instead of numbers.

Module 4: Making Decisions (Conditionals)
List 3 decisions you have made today.

_____

_____

_____

Conditionals:

if (*condition*) then

    Action if true

else

    Action if condition is false

Red Light - Green Light conditionals:

if (_____) then

    _____

if (_____) then

    _____

if (_____) then

    _____

else

    _____

Conditions and Boolean expressions

All of the conditions in an if...then statement have to be an expression that can be evaluated as either True or False. These are called Boolean expressions when they are either True or False. These expressions usually use comparison operators to decide if it is True or False.

Comparison Operators		
Operator	Name	Example
==	equals	`x == y`
!=	not equal	`x != y`
>	greater than	`x > y`
<	less than	`x < y`
>=	greater than or equal to	`x >= y`
<=	less than or equal to	`x <= y`

Logical Operators		
Operator	Description	Example
and	Returns True if both statements are true	x < 5 and x < 10
or	Returns True if one of the statements are true	x < 5 or x < 4
not	Reverse the result, returns False if the result is true	not(x < 5 and x < 10)

04.2a Coin Toss Activity

Algorithm & Pseudocode:

04.2b Dice Roll Activity

Algorithm & Pseudocode:

04.3 Project: Board Game (done with a partner)

In this project you will plan, design, and create a board game. It should have clear rules on how to play. It should use conditionals on the microbit in a way that is central to the game. It should also use a maker elements as part of the design and construction. (Do a search for “DIY board games.)

Brainstorm Ideas _____

_____

_____

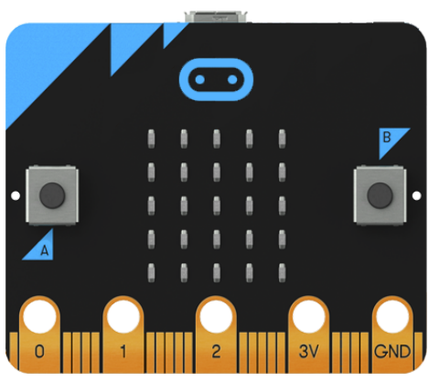
_____

Project: _____

Description: _____

_____

Microbit Project Sketch:



Game Board Sketches

Board Game Rules and Conditionals in Playing:

Board Game Algorithm & Pseudocode:

Materials Needed: _____

Coding Plan: _____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Photos:

Notes & Reflections

How did you decide with your partner on your board game? _____

_____

_____

_____

What was something that was surprising to you in the creation of your board game?

_____

_____

_____

How well did your prototype work? _____

_____

Describe a difficult point in designing your game and how you resolved it:

_____

_____

_____

What feedback did you get from your beta testers?

_____

_____

What did you change to improve your game? _____

_____

Describe how you and your partner shared the work on the project.

_____

_____

_____

Assessment Rubric

Competency scores

Competency	4	3	2	1
Rules	All game rules are clear and complete.	A game rule is missing or not complete or not clear.	More than one game rule is missing or not complete or not clear.	Most of the game rules are missing or it is not clear what the rules are.
Game Board	Game board is: 1) Complete 2) Neat 3) Fits with the theme of the game 4) micro:bit is a central part of the game	Game board meets only 3 of the conditions listed for a score of 4.	Game board meets only 2 of the conditions listed for a score of 4.	Game board meets only 1 of the conditions listed for a score of 4.
Micro:bit Program	micro:bit program: 1) Uses the micro:bit in a way that is integral to the game 2) Uses conditionals correctly 3) Compiles and runs as intended 4) JavaScript includes comments in code	micro:bit program lacks 1 of the required elements.	micro:bit program lacks 2 of the required elements.	micro:bit program lacks 3 of the required elements.
Photo Documentation	Complete photo documentation that includes photos of game board and code and captions.	A photo is missing or of poor quality or a caption is missing.	Multiple photos and/or captions missing or of poor quality.	Most photos and/or captions missing or of poor quality.
Collaboration Reflection	Reflection piece includes: 1) Brainstorming ideas 2) Construction 3) Programming 4) Beta testing	Reflection piece lacks 1 of the required elements.	Reflection piece lacks 2 of the required elements.	Reflection piece lacks 3 of the required elements.

Comments or Photos:

Notes