With funding from Canada

Scratch Catch with micro:bit Joystick

Participants Perspective

Today I learned to create a joystick for my Scratch games **by** programming a micro:bit as a joystick in Scratch 3.0.

Learning Goal

- Use Scratch 3.0's micro:bit module to connect a micro:bit to Scratch 3.0
- Work with the micro:bit hardware

Logistics

- Suggested length: 1-2 hours
- Age range: Grade 5 or Older
- Additional requirements: Basic programming skills in Scratch (2.0 or 3.0), including the use of variables and control loops is beneficial.

Part Number	Section Title	Group Size	Materials
Opening Hook	micro:bit	Whole group	• Digital device with access to Scratch 3.0 and the internet • micro:bit • Method of showing screen to class
Part 1	Catch me if you can	Partners	• Digital device with access to Scratch 3.0 and the internet • micro:bit • Scrap paper and pencil (Optional)
Part 2	High Score	Partners	• Digital device with access to Scratch 3.0 and the internet • micro:bit • Sticky Notes & Pens
Part 3	Closing discussion	Whole group	• Method of showing a video (Youtube) to class

Safety Considerations

- **Laptops/Computers-** Electrocution hazard from frayed or damaged cords and tripping over unsecured cords. These issues can be mitigated by securing all electrical cords, particularly extension cords, reaching over the floor (using tape) and immediately unplug them after use. Cords are plugged in/unplugged by adults, or by participants under adult supervision.

- **Internet:** Participants may come across materials that is not appropriate (including but not limited to violent or sexual images, racist/sexist commentary, and so on). Participants may use websites unrelated to the activity (social media, email, etc). When participants are using computers, activities should be monitored. When possible, each computer should be pre-set with the desired website already loaded. Review expectations with participants before you start the activity and develop a plan that they should follow if they come across inappropriate content.

Framework Connections

Toolsets	Skillsets	Mindsets
Knowledge Developing a working interface between a micro:bit and Scratch 3.0 Understand uses for loops and other flow control devices in Scratch	**Digital Skills** Coding and bug fixing in Scratch, including flow control and micro:bit hardware integration Understanding limits of hardware. Building interaction between the digital and physical world	**Digital Intelligence** Awareness of interface between the digital and physical worlds
Resources Scratch 3.0 (beta) or newer micro:bit	**STEM Skills** Problem solving using available tools Algorithm development and implementation. Computational thinking to organize the code in a logical sequence	**Computational Thinking** Understanding how hardware might interact with software. Logical sequencing and cyclical processes. Chunking of larger problem.
Experiences Develop a working joystick for controlling a Scratch game. Explore the functionality of a micro:bit Computational thinking - Using flow control to move objects in a logical manner	**Essential Employability and Life Skills** Problem Solving & critical thinking to evaluate how to make things work Creative design and solving a problem by integrating multiple tools Peer feedback and communication	**Digital Action** Awareness of using simple processes and tools to combine and make a bigger solution. Computer science/ electrical engineering

Nuts and Bolts

Setup:

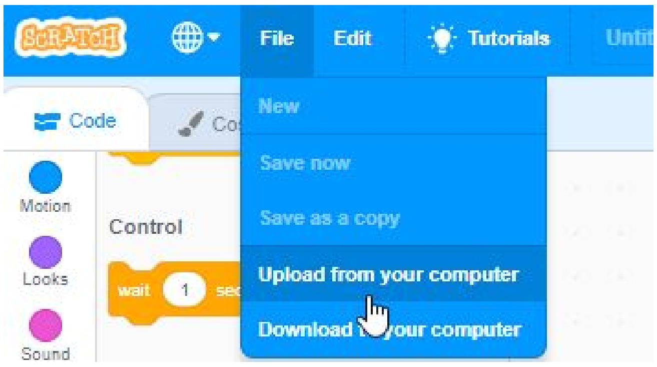
For micro:bits to connect with Scratch 3.0, there are two things that need to be in place:

- The device running Scratch must have the **Scratch Link** software installed and running. See ZIP file attached for Windows version or visit: https://scratch.mit.edu/microbit to download and install the latest version for your devices
- The micro:bit needs to have a special .hex file placed on it. This is done by connecting the micro:bit to your computer using the USB cable and dropping the attached

scratch-microbit-1.0.hex file onto the micro:bit (drag and drop the file). The newest version of this file can be found here: https://scratch.mit.edu/microbit along with instructions on how to install it. Once this file is loaded, the micro:bit can be connected wirelessly (via Bluetooth) to the device; no USB connection is needed.

These steps can be done by the participants if desired.

Prepare one display setup for the Hook by loading a Scratch micro:bit demonstration program (see Starter Projects: https://scratch.mit.edu/microbit), such as the microbit-guitar.sb3 (attached). This is done by loading the .sb3 file in Scratch 3.0 and making sure you have a connected micro:bit. This demo will require sound.



Hook: micro:bit
Show the participants the demo micro:bit. Explain that a micro:bit is a small printed circuit board that can be used to:

- Display dots on its LED display
- Detect tilt of the micro:bit using an accelerometer/magnetometer
- Detect button presses on its 2 buttons
- Be connected to other electronics to control them
- It can also be connected to Scratch 3.0 to interact with your scratch programs using the above features.

Load microbit-guitar.sb3 (attached) in Scratch 3.0 and show the participants how the micro:bit can control the sounds that the guitar can make. This will start to formulate ideas of how this might be used to control other items in Scratch.

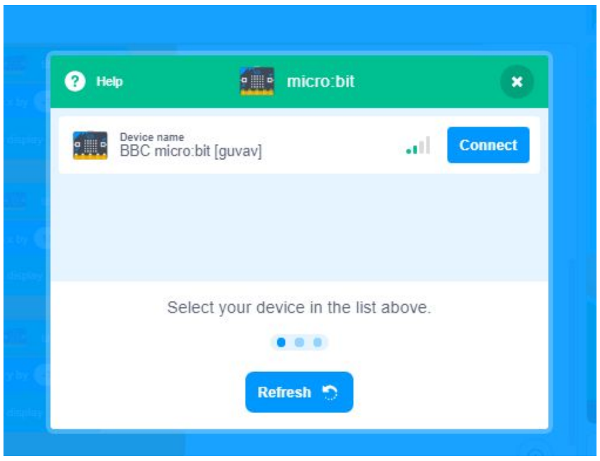
Allow a few volunteers to come up and try it out for the whole group. Explain that the task for this activity is to learn to program the micro:bit to work in a simple game.

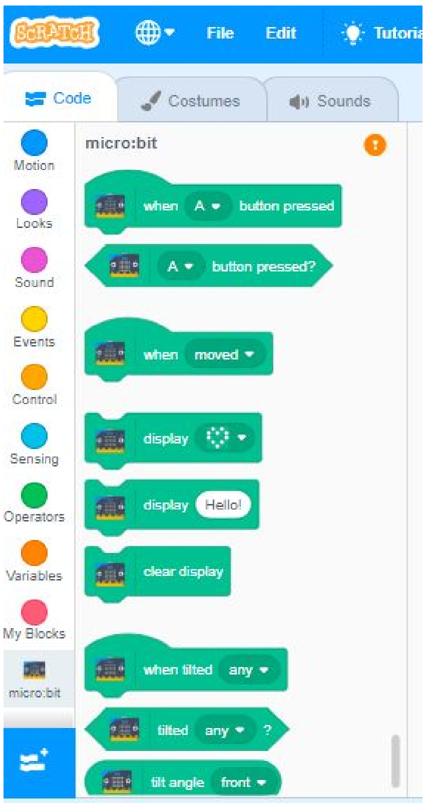
Part 1: Catch me if you can

/code:makers_ actua Google™

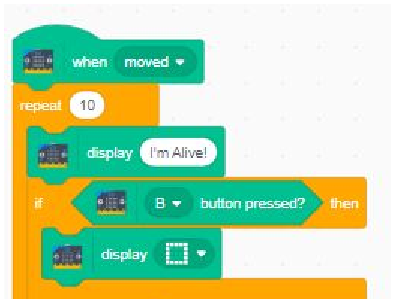
With funding from Canada

- Participants should pair up with a partner and load Scratch 3.0 on their connected device. This is currently (Nov 2018) available under BETA release at: https://beta.scratch.mit.edu/. Full release is expected early in 2019
    - Note: if some students are not familiar with Scratch, they should pair up with someone who has used the software before.
- We will start with loading the micro:bit extension and connecting. Open the extension in the bottom left, by clicking the add extensions icon
- This will bring up the bluetooth connection box. Make sure:
    - micro:bit is powered on either using a USB cable or a battery pack
    - Scratch Link is running
    - Bluetooth is enabled on the device that you are running Scratch on
- You should now be able to connect to your device. Note that there is a code scrolling across the screen of the micro:bit. Select "Connect" for this device from the bluetooth connection box. Any issues see https://makecode.microbit.org/reference/bluetooth/bluetooth-pairing





- Select "Go to editor" and this will add the micro:bit sub-menu to your scratch menu (see image on right).
- Participants should take some time to review the available programming blocks. Note the different options for button pressing, display, and tilt. These are all designed to fit into the Scratch ecosystem
- Next we are going to try these out with a very simple program that completes 2 tasks:
    - When the micro:bit moves, it will say "I'm Alive"
    - Then once the B button is pressed it will display a square
- Have the participants try it out themselves. After a few minutes share what it should look like with those who are struggling. One example of this code is this (though many options are possible):



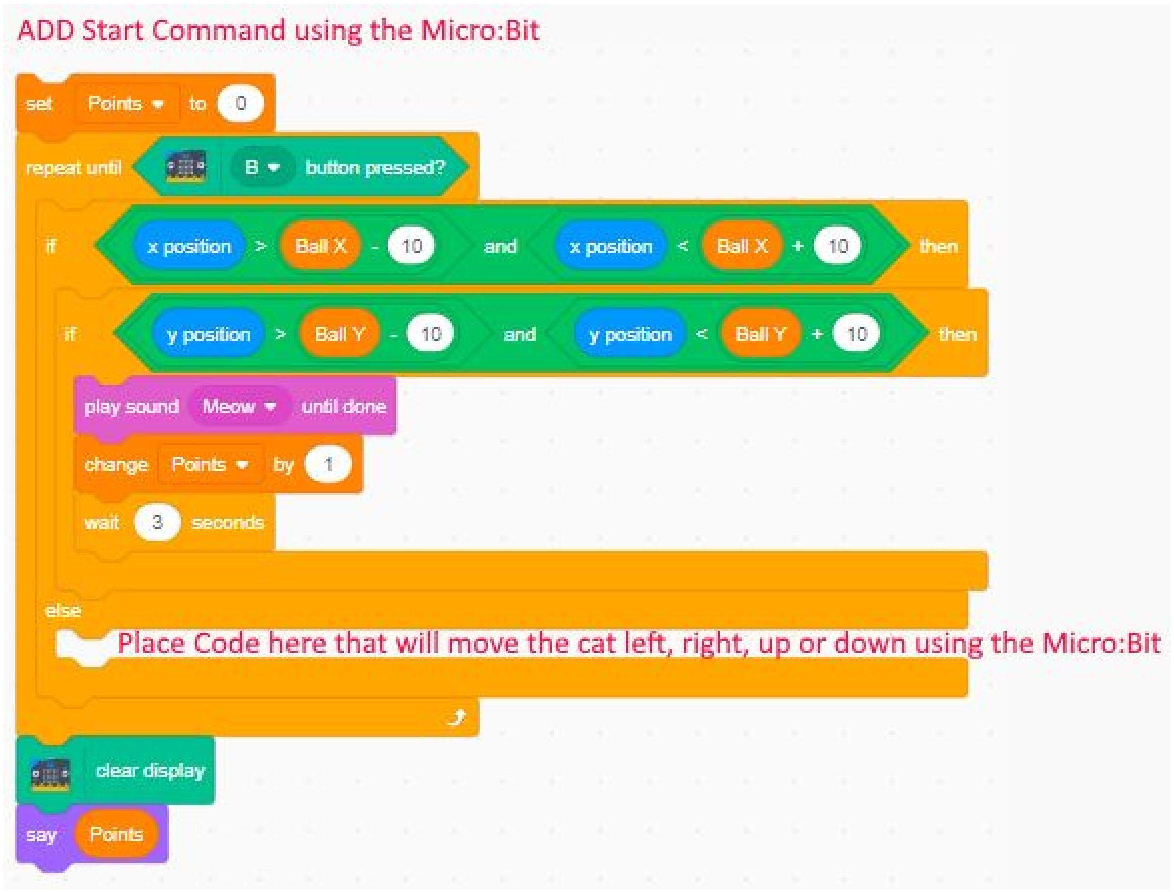
```
when moved
repeat 10
    display I'm Alive!
    if B button pressed? then
        display ⬚
```

- Verify that all of the partnerships have been able to get this working. If some are done early, ask them to try adding additional displays for tilting in different directions. Have the participants consider how this might be useful in controlling the movement of a character in Scratch.

Now participants are going to embark on a challenge to make a game controller for a Scratch game, using the micro:bit.

- **For Advanced Scratch Programmers** - Allow participants to design a game where a sprite moves are controlled by the micro:bit. Programmers are free to design the game as they wish, but should be encouraged to use the tilt, button and display features of the micro:bit in their design. For programmers who are looking for a concept, try making one sprite move randomly around the back drop while the sprite connected to the micro:bit tries to catch it.

- **For Novice Scratch Programmers** - Distribute the Catch_Start.sb3 file. This is the core of a game that needs some additional code added to make the micro:bit more the sprite. Share the image below with the participants and have them come up with:
    - A method for starting the game using the micro:bit
    - The code that will move the cat using the micro:bit
    - Code that uses the display of the micro:bit



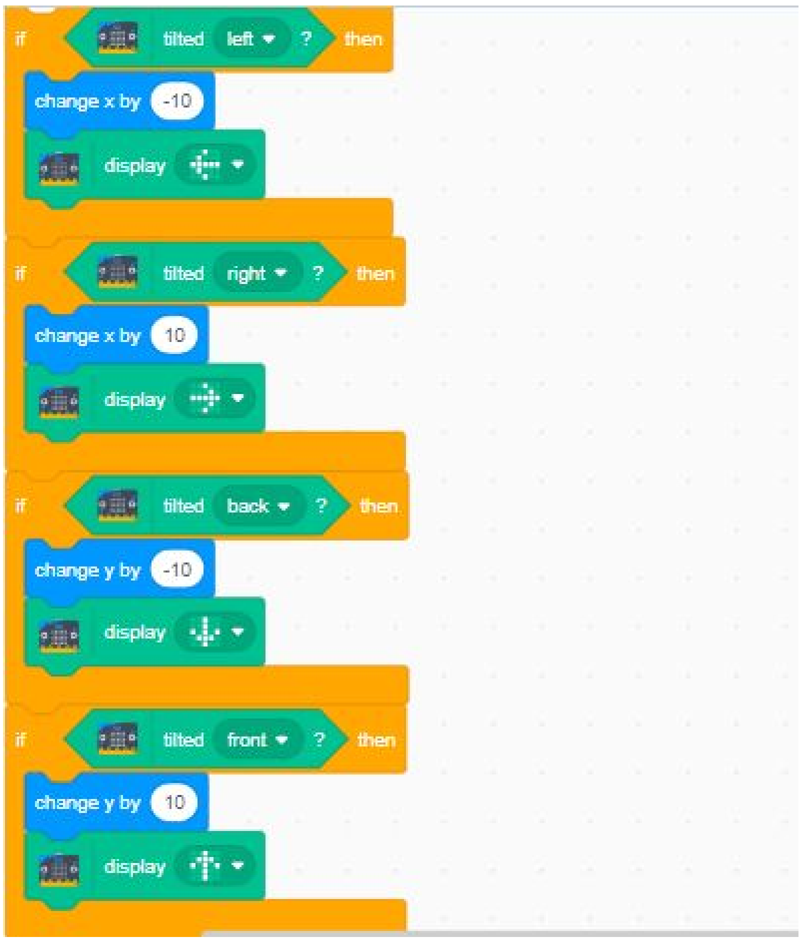
ADD Start Command using the Micro:Bit

```
set Points to 0
repeat until B button pressed?
    if x position > Ball X - 10 and x position < Ball X + 10 then
        if y position > Ball Y - 10 and y position < Ball Y + 10 then
            play sound Meow until done
            change Points by 1
            wait 3 seconds
    else
        Place Code here that will move the cat left, right, up or down using the Micro:Bit
clear display
say Points
```

***Note-**Full code with explanation is provided in the Appendix. Full working code (Catch_Full.sb3) is also included with this package

- **Controls -** This program works by the ball moving around the screen. The ball is started by pressing the spacebar. The game is stopped by pressing the "B" button on the micro:bit
- Tips:
    - If there are glitches in the program responding, try to save the file (File>Download to Computer) and **Reload Scratch again**, reloading the file (File>Upload from Computer).
    - Plan out on paper first. What might the program look like? Planning first will lead to faster coding, better communication and problem solving, and will result in a better program overall.
    - Consider how the program needs to be structured, such that each tilt direction is represented.
- The following is sample code that will provide the basic functionality of the micro:bit motion (can be placed within the "else" statement, as indicated above). As with any program there are numerous ways to do this and how participants approach the problem will help them to develop their problem solving and computational thinking skills. Note again that the appendix contains this code with comments about what each line is doing for reference.

With funding from



```
if tilted left ? then
    change x by -10
    display
if tilted right ? then
    change x by 10
    display
if tilted back ? then
    change y by -10
    display
if tilted front ? then
    change y by 10
    display
```

- Participants should be encouraged to test out their program regularly and make changes as needed. Consider how you might want to test out the software? This iterative process will help to build a better overall program as it will help to fix software bugs and give the participants a chance to experience what the program might be like for the actual user.
- Once the participants have working games, it is time to share their creations. HAve all the participants Save a copy of their current game to their computer (File>Download to your computer)

Part 2: High Score

Participants should now go around the room and try out other players games.

- Play the game and see how it works. Try to get a high score!
- Take a look at the code. Did they do it differently than you?
- Using a sticky note and pen, leave some feedback for the programmer. What did you like, what would you change? Be descriptive! The act of both providing and receiving feedback helps to build a better understanding of the needs for the product and creates a better product in the end. It is also helpful step in the learning process and building strong communication skills.

Pairs will now tweak their program based on both the feedback they received and what they learned from using and seeing the other pairs game. Allowing the participants time to iterate on their original idea will improve problem solving skills, critical thinking and develop a better product in the end.

- Once the participants have completed their changes, they should test it out again.
- Remember to save a copy of the final product (if desired)

Part 3: Closing Discussion
Discuss what the participants learned from this experience:

- Are they able to program a micro:bit in Scratch?
- What other ways could you use the micro:bit to control scratch?
- What other projects, besides a game could you use this for?
    - Security alarm?
    - Measurement device?

Give participants some time to explore some other Scratch micro:bit Projects

- https://microbit.org/scratch/

Modifications and Extensions

Ways to make this activity more approachable:

- Groups of 4 work together to complete the design.
- Develop the code as a whole group with students as they following along on their own computers
- Complete a sample of code for 1 tilt direction, before participants complete the rest

Ways to make this activity more challenging:

- Complete the code from scratch (advanced)
- Have the micro:bit display show the ball location. Making the game 100% played on the microbit.
- Try programming in Javascript : https://makecode.microbit.org/#

Assessment and Evaluation

- Iterative design cycle and peer feedback on the code design
- Group questions and discussion

Credits, Kudos, Shoutouts

- Scratch 3.0 Programming language: https://beta.scratch.mit.edu/
- micro:bit: https://microbit.org/scratch/ & https://makecode.microbit.org/#

Terms of Use

Prior to using this activity or parts thereof, you agree and understand that:

- It is your responsibility to review all aspects of this activity and ensure safety measures are in place for the protection of all involved parties.
- Any safety precautions contained in the “Safety Considerations” section of this write-up are not intended as a complete list or to replace your own safety review process.

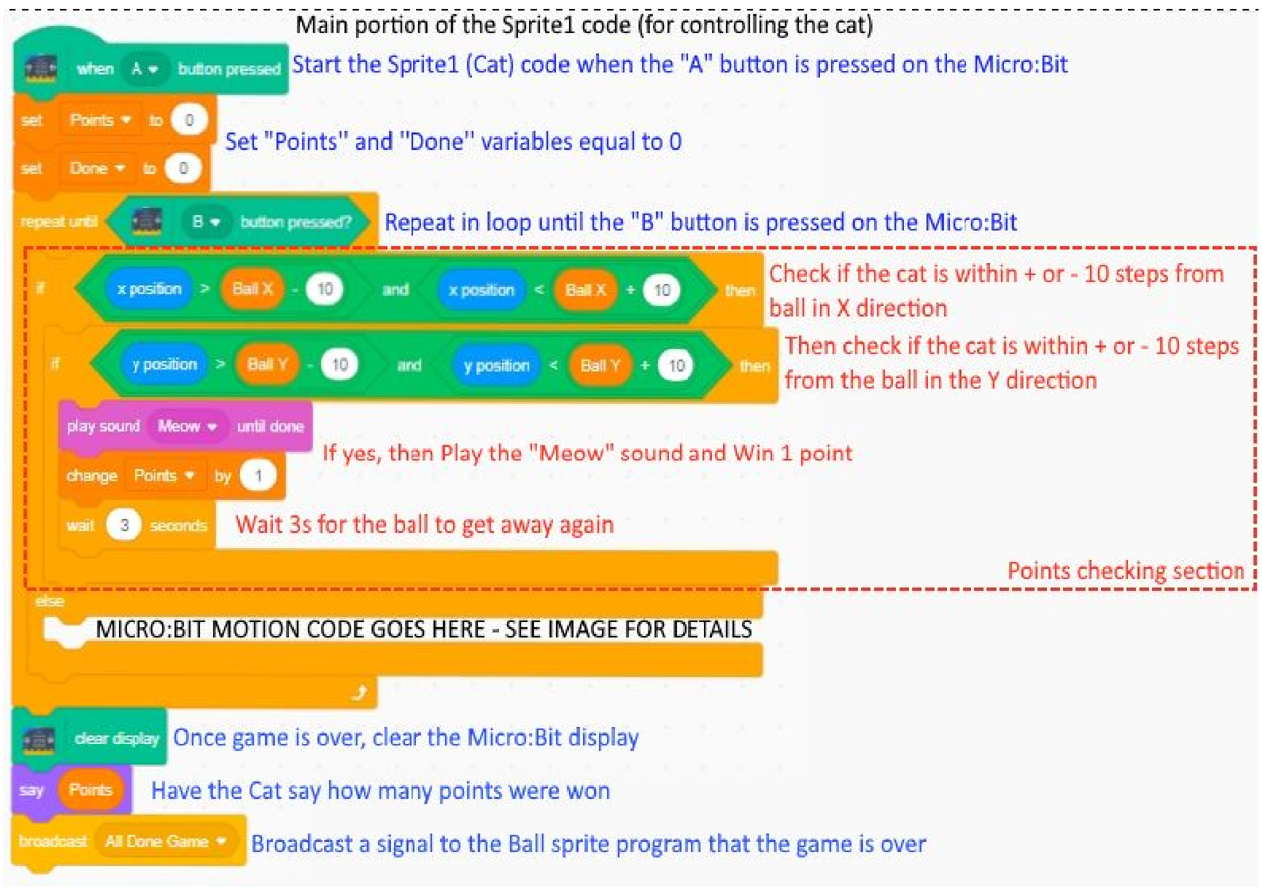
About Actua

Actua is Canada's leading science, technology, engineering and mathematics (STEM) youth outreach network representing a growing network of university and college based members. Each year 250,000 young Canadians in over 500 communities nationwide are inspired through hands-on educational workshops, camps and community outreach initiatives. Actua focuses on the engagement of underrepresented youth through specialized programs for Indigenous youth, girls and young women, at-risk youth and youth living in Northern and remote communities. For more information, please visit us online at www.actua.ca and on social media: Twitter, Facebook, Instagram and YouTube!

With funding from

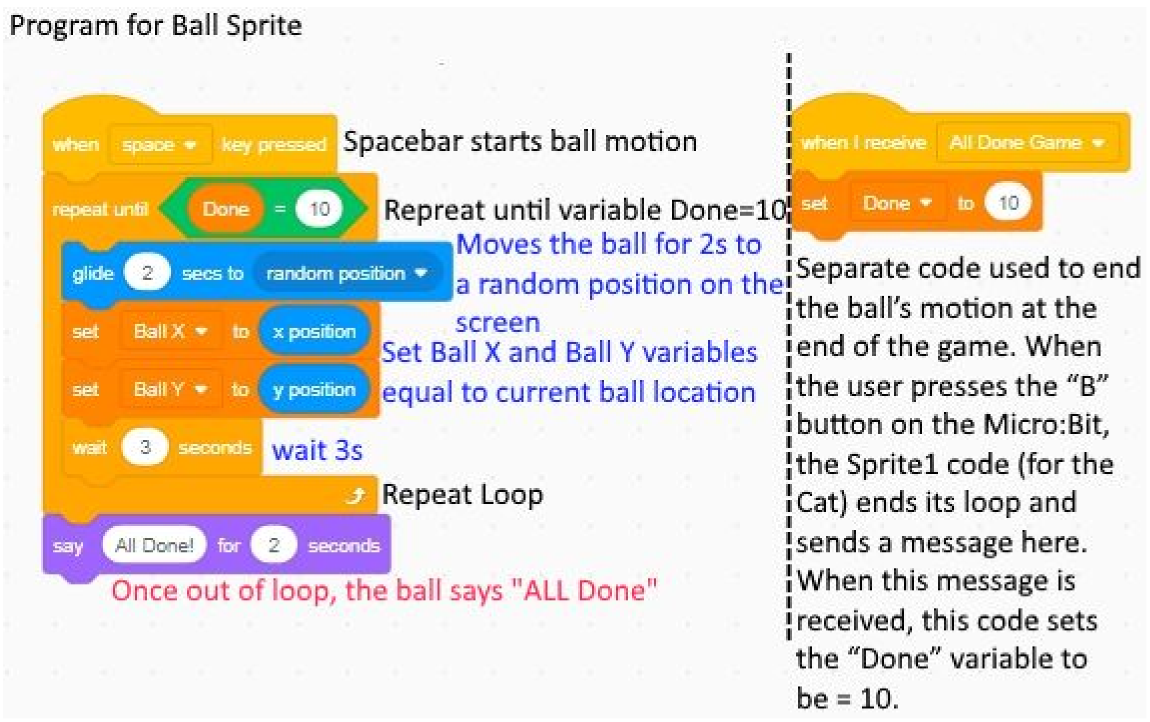
Sample Catch_Full code with line-by-line comments:

See Catch_Full.sb3 file for code



Main portion of the Sprite1 code (for controlling the cat)

```
when A button pressed                          Start the Sprite1 (Cat) code when the "A" button is pressed on the Micro:Bit
set Points to 0                                Set "Points" and "Done" variables equal to 0
set Done to 0
repeat until <B button pressed?>               Repeat in loop until the "B" button is pressed on the Micro:Bit
    if <<(x position) > ((Ball X) - 10)> and <(x position) < ((Ball X) + 10)>> then
                                               Check if the cat is within + or - 10 steps from ball in X direction
        if <<(y position) > ((Ball Y) - 10)> and <(y position) < ((Ball Y) + 10)>> then
                                               Then check if the cat is within + or - 10 steps from the ball in the Y direction
            play sound Meow until done         If yes, then Play the "Meow" sound and Win 1 point
            change Points by 1
            wait 3 seconds                     Wait 3s for the ball to get away again
                                               Points checking section
    else
        MICRO:BIT MOTION CODE GOES HERE - SEE IMAGE FOR DETAILS
clear display                                  Once game is over, clear the Micro:Bit display
say Points                                     Have the Cat say how many points were won
broadcast All Done Game                        Broadcast a signal to the Ball sprite program that the game is over
```



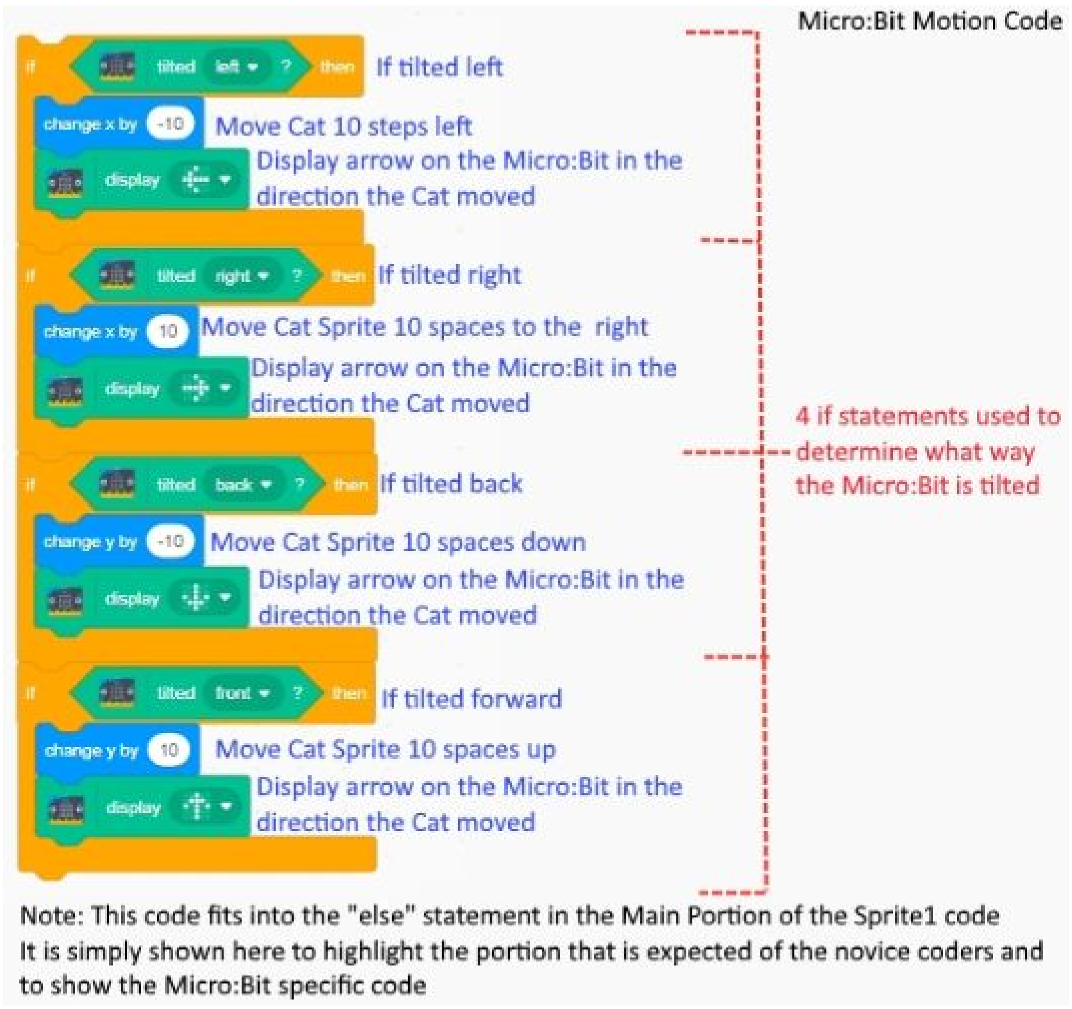
Program for Ball Sprite

```
when space key pressed                         Spacebar starts ball motion
repeat until <(Done) = 10>                     Repreat until variable Done=10
    glide 2 secs to random position            Moves the ball for 2s to a random position on the screen
    set Ball X to x position                   Set Ball X and Ball Y variables equal to current ball location
    set Ball Y to y position
    wait 3 seconds                             wait 3s
                                               Repeat Loop
say All Done! for 2 seconds
                                               Once out of loop, the ball says "ALL Done"
```

```
when I receive All Done Game
set Done to 10
```

Separate code used to end the ball's motion at the end of the game. When the user presses the "B" button on the Micro:Bit, the Sprite1 code (for the Cat) ends its loop and sends a message here. When this message is received, this code sets the "Done" variable to be = 10.

With funding from



Micro:Bit Motion Code

```
if < tilted left ▾ ? > then        If tilted left
  change x by -10                  Move Cat 10 steps left
  display ▾                        Display arrow on the Micro:Bit in the
                                   direction the Cat moved
if < tilted right ▾ ? > then       If tilted right
  change x by 10                   Move Cat Sprite 10 spaces to the  right
  display ▾                        Display arrow on the Micro:Bit in the
                                   direction the Cat moved
if < tilted back ▾ ? > then        If tilted back
  change y by -10                  Move Cat Sprite 10 spaces down
  display ▾                        Display arrow on the Micro:Bit in the
                                   direction the Cat moved
if < tilted front ▾ ? > then       If tilted forward
  change y by 10                   Move Cat Sprite 10 spaces up
  display ▾                        Display arrow on the Micro:Bit in the
                                   direction the Cat moved
```

4 if statements used to determine what way the Micro:Bit is tilted

Note: This code fits into the "else" statement in the Main Portion of the Sprite1 code
It is simply shown here to highlight the portion that is expected of the novice coders and to show the Micro:Bit specific code