micro:bit Reference Sheet

More info at: https://makecode.microbit.org/docs

Basic Commands

`basic.showString("text")`: scrolls text across the screen

- `text:` a string of characters (must be included inside a set of quotation marks

`basic.showNumber(number)`: scrolls a numerical value across the screen

- `number:` a numerical value

`basic.pause(duration)`: pauses the program for a set time before moving on to future commands

- `duration`: Amount of time to pause program in milliseconds

`basic.forever(function () {})`: repeats all commands inside curly braces until the program is ended

`basic.clearScreen()`: turns off all LEDs on the 5x5 grid

Music Commands

`music.playTone(note, duration)`: plays a given note for a set duration through Pin 0

- `note`: note to be played, written as `Note.C`, for example
- `duration`: Amount of time the note is played in milliseconds

LED Commands

`led.plot(xCoordinate, yCoordinate)`: turns on an LED at a given coordinate point

- `xCoordinate`: horizontal location of the LED, between 0 (left) and 4 (right)
- `yCoordinate`: vertical location of the LED, between 0 (top) and 4 (bottom)

`led.unplot(xCoordinate, yCoordinate)`: turns off an LED at a given coordinate point

- `xCoordinate`: horizontal location of the LED, between 0 (left) and 4 (right)
- `yCoordinate`: vertical location of the LED, between 0 (top) and 4 (bottom)

`led.plotBrightness(xCoordinate, yCoordinate, value)`: turns on LED at a coordinate point with a given brightness value

- `xCoordinate`: horizontal location of the LED, between 0 (left) and 4 (right)
- `yCoordinate`: vertical location of the LED, between 0 (top) and 4 (bottom)
- `value`: brightness value of the LED between 0 (off) and 255 (full brightness)

Pin Commands

Read: Returns the value of a connected component	`pins.digitalReadPin(pin)`	`pin`: pin the component is connected to, either `DigitalPin.P0`, `AnalogPin.P0`, `DigitalPin.P1`, `AnalogPin.P1`, `DigitalPin.P2`,or `AnalogPin.P2`
	`pins.analogReadPin(pin)`	
Write: Sets the value of a connected component	`pins.digitalWritePin(pin, value)`	`value`: A number `0` (off) or `1` (on)
	`pins.analogWritePin(pin, value)`	`value`: A number `0-255`

`pins.servoWritePin(pin, angle)`: sets a connected servo motor to a certain angle between `0-180`

- `pin`: pin the component is connected to, either `AnalogPin.P0`, `AnalogPin.P1`, or `AnalogPin.P2`
- `angle`: A number `0-180` which notes the angle to move the motor to

Variables

`let variable = value`: creates a variable and assigns it a value

- `variable`: name of the variable written in camelCase
- `value`: A number or string

Functions

`function name() {}`: defines a function as the commands found between the curly braces

- `name`: name of the function, written in camelCase
- To call the function, simply write the function name and a set of parentheses, ie. `myFunction()`
- Parameters can be included in the parentheses as `parameterName: type` where `type` is `number` or `string`

Control Structures in the MakeCode Editor

For Loops:
Repeat a set number of times
	Syntax:
	`for (let initialization; condition; increment) {`
	`    commands;`
	`}`
		Example:
		`for (let int i = 0; i < 5; i++) {`
		`    led.plot(i, 0)`
		`}`

While Loops:
Repeat while a condition remains true
	Syntax:
	`while (condition) {`
	`    commands;`
	`}`
		Example:
		`while (input.temperature() < 21) {`
		`    pins.digitalWritePin(DigitalPin.P0, 0)`
		`}`

If/Else Statements:
Choose actions to perform based on given conditions
	Syntax:
	`if (condition) {`
	`    commands;`
	`} else if (condition) {`
	`    commands;`
	`} else {`
	`    commands;`
	`}`
		Example:
		`    if (input.buttonIsPressed(Button.A)) {`
		`        pins.analogWritePin(AnalogPin.P0, 1000)`
		`    }`
		`    else {`
		`        pins.analogWritePin(AnalogPin.P0, 500)`
		`    }`

Sensor Conditions:

`input.buttonIsPressed(button)`: returns if a button is pressed
- `button`: name of the button, either `Button.A`, `Button.B`, or `Button.AB`

`input.lightLevel()`: returns the level of the LED screen light sensor
- Light level value is given from `0` (dark) to 255 (full brightness)

`input.temperature()`: returns the temperature in degrees Celsius

`input.acceleration(dimension)`: returns the acceleration value in milli-gravitys
- `dimension`: dimension to measure acceleration, either `Dimension.X`, `Dimension.Y`, or `Dimension.Z`
- When board is laying flat with LED screen up, `Dimension.X = 0`, `Dimension.Y = 0`, and `Dimension.Z = -1024`

Sensor Functions:

Run commands when button is pressed	`input.onButtonPressed(button, function () {})`

- `button`: name of the button, either `Button.A`, `Button.B`, or `Button.AB`

Run commands when pin is pressed or released	`input.onPinPressed(pin, function () {})`
	`input.onPinReleased(pin, function () {})`

- `pin`: pin that component is connected to, either `TouchPin.P0`, `TouchPin.P1`, or `TouchPin.P2`

Run commands when chosen gesture is made	`input.onGesture(Gesture.gesture, function () {})`

- `gesture`: `EightG`, `FreeFall`, `LogoDown`, `LogoUp`, `ScreenDown`, `ScreenUp`, `Shake`, `SixG`, `ThreeG`, `TiltLeft`, `TiltRight`

Run commands if the screen is vertically facing the ceiling (up) or the ground (down)	`input.onLogoUp(function () {})`
	`input.onLogoDown(function () {})`
Run commands if the screen is horizontally facing the ceiling (up) or the ground (down)	`input.onScreenUp(function () {})`
	`input.onScreenDown(function () {})`
Run commands when device is shaken	`input.onShake(function () {})`