Function reference

The following are some functions that you can use in your programming. You can use this document as a reference during your work.

drawHorizontalLine(posx, posy, size, red, green, blue)

Draw a horizontal line

Draws a horizontal line on the screen. You have to specify the position, size and color of the line. *posx* and *posy* are the starting x- and y-positions of the line. *size* is how many pixels the line should consist of. *red*, *green* and *blue* sets the colour of the line.

For example here's some code that draws a horizontal line that is three pixels wide, lies at the position (2, 3), and is purple:

```
drawHorizontalLine( 2, 3, 3, 255, 0, 255 )
```

Another way to do it would be

```
posx = 2
posy = 3
size = 3
red = 255
green = 0
blue = 255
drawHorizontalLine(posx, posy, size, red, green, blue)
```

drawVerticalLine(posx, posy, size, red, green, blue)

Draw a vertical line

Draws a vertical line on the screen. You have to specify the position, size and color of the line. *posx* and *posy* are the starting x- and y-positions of the line. *size* is how many pixels the line should consist of. *red*, *green* and *blue* sets the colour of the line.

For example here's some code that draws a horizontal line that is three pixels long, lies at the position (2, 3), and is purple:

```
drawVerticalLine( 2, 3, 3, 255, 0, 255 )
```

Another way to do it would be

```
posx = 2
posy = 3
size = 3
red = 255
green = 0
blue = 255
drawVerticalLine(posx, posy, size, red, green, blue)
```

wait(seconds)

Halt the program for some time

Will halt the program for the amount of seconds that you specify. For example to wait for 5 seconds:

```
wait(5)
```

draw_two_digit_number(number, red, green, blue)

Draws any one-digit or two-digit number

Draws any number from 0 to 99 on the screen. You can also specify the color of the number. For example, to draw 45 in the colour red:

```
draw_two_digit_number(45, 255, 0, 0)
```

sense.clear()

Removes all drawings on the screen

If you want to clear all drawings on the screen (reset all pixels to black), you use this function.

```
sense.clear()
```

sense.set_pixel(positionx, positiony, red, green, blue)

Sets a pixel to a color

This function can be used to draw things pixel-by-pixel on the screen.
positionx and *positiony* is the x and y position of the pixel to draw.
If you don't know what that means, the x-position is how many squares the ball is away from the left side of the screen, and the y-position is how many squares the ball is away from the top side of the screen.
The following code will draw a yellow pixel at position (5,1).

```
pos = [5, 1]
color = [0, 255, 255]
sense.set_pixel(5, 1, color[0], color[1], color[2])
```

sense.show_message(message, time)

Displays a text message on the screen

You can use this function to display a message on the screen. You can also decide how fast the message will scroll past the screen. The following displays the message "Hello World!" on the screen over 2 seconds.

```
sense.show_message("Hello world!", 2)
```

Checking the joystick

You can let the user control the program using the joystick. The way to do this can seem a bit complicated, but read this section to get the hang of it. If you don't understand something, ask a supervisor!

It's easiest to learn through an example. The following displays "You pressed up!" if the user pressed up on the joystick.

```
for event in sense.stick.get_events():
  if event.action == "pressed":
    if event.direction == "up":
      sense.show_message("You pressed up!")
```

That's a lot of code, and you probably don't know what it all means. Don't worry about it! By looking at the code you can modify it for your own purposes. To change it so that it checks whether a button is released, rather than pressed, change the second line in the code snippet to

```
if event.action == "released":
```

Quite simple right? Similarly, if you want to check the down-button rather than the up-button, change the third line to

```
if event.direction == "down":
```

And you can change it to "left" or "right" to check the left or right buttons.

As a final example, here's some code that checks all the 5 different buttons (up, down, left, right, middle). Feel free to use this code directly in your program

```
for event in sense.stick.get_events():
  if event.action == "pressed":

    if event.direction == "up":
      sense.show_message("You pressed up!")

    if event.direction == "down":
      sense.show_message("You pressed down!")

    if event.direction == "left":
      sense.show_message("You pressed left!")

    if event.direction == "right":
      sense.show_message("You pressed right!")
```

Note that these code snippets should be in the *main loop* of your program.

random.randint(minNumber, maxNumber)

Gives a random number between two specified number

To get random number between 1 and 10:

```
random.randint(1, 10)



---
```

Author: Lukas Kikuchi

Date: August 09, 2017