# Writing the Default Program with Functions (Blocks)

Different sections of a program often do different things. A program can be divided up into separate parts with each part performing a particular function. From the main program, these functions can be "called" or run. This makes the code much more readable.

In mBlock, we make a function (or block) by clicking on "Make a Block" in the "Data&Blocks" script:
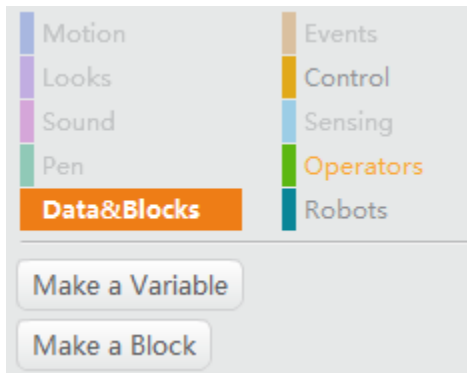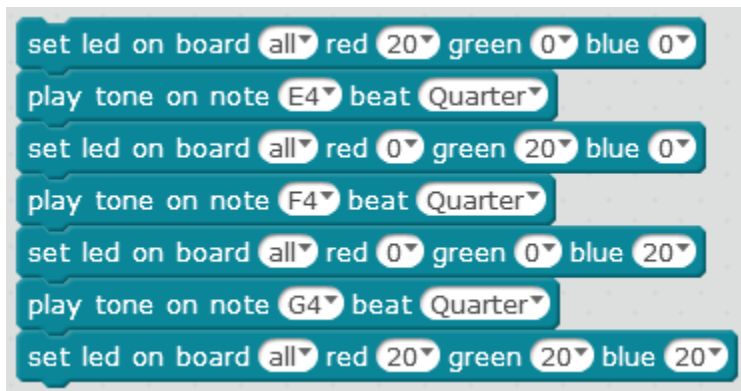

Then choose a name that represents what the block is doing:
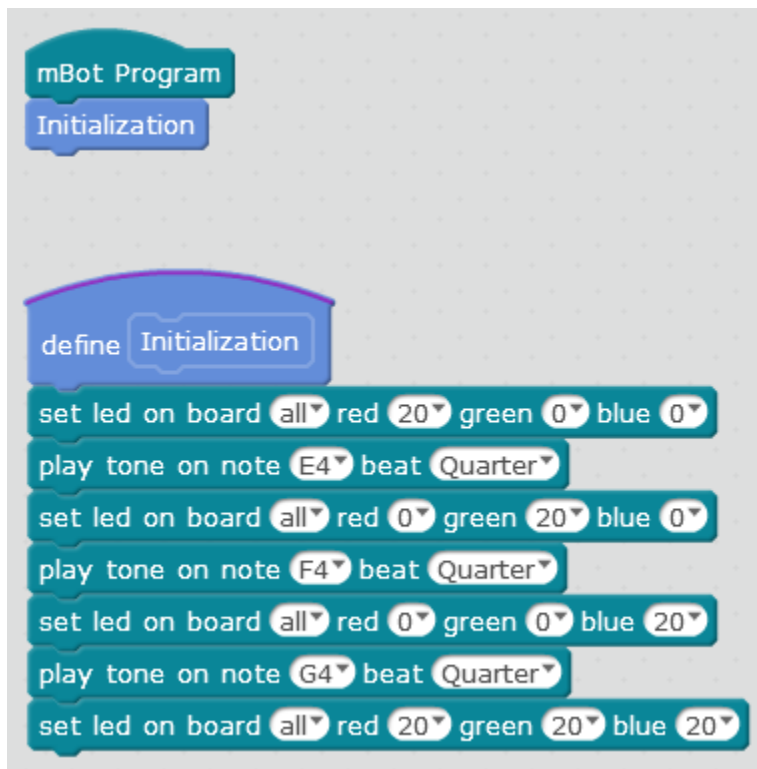

And then the block is defined and ready to use.

So in this tutorial I want to recreate the default program. It will be a close approximation to the default program though, not a complete replica. First, the robot initializes with the beeps and lights:

I am going to put this into its own initialization block and call it from the mBot program:

This now runs exactly the same as if the set LED commands and the play tone commands were called directly from the mBot program. The code is much easier to read like this though. Next, I want to write the code for the remote control part. Again, I am going to write this code in its own block:

```
define  Remote Control

set led on board  all▾  red  20▾  green  20▾  blue  20▾
repeat until   on board button  pressed ▾
    if   ir remote  ↑ ▾  pressed   then
        run forward ▾  at speed  100▾
    else
        if   ir remote  ↓ ▾  pressed   then
            run backward ▾  at speed  100▾
        else
            if   ir remote  ← ▾  pressed   then
                turn left ▾  at speed  100▾
            else
                if   ir remote  → ▾  pressed   then
                    turn right ▾  at speed  100▾
                else
                    run forward ▾  at speed  0▾
```

I then write similar blocks for a line following behavior and an object avoidance behavior. Now, I have to write these behaviors into my main program. As you can see, the final program looks very simple:

```
mBot Program

Initialization

forever                          ▾
    Remote Control        Each mode will be run
                          until the on board
    Object Avoidance      button is pressed
    Line Following
```

Challenge:

1. Can you write programs for the Object Avoidance and Line Following functions and add them to the program?
2. Sometimes when the button is pressed, a mode or 2 is skipped. Can you fix this bug?
3. How close to the real Default Program can you make your program?