Table des matières

To create new block, here is a simple way using Grove LED block as example:

1 - modify index.html, at the end of file add categories and **name of blocks** you want to add in the Toolbox (*sidebar menu*) :

```
    <category name="CAT_ARDUINO_SERVO">
            <block type="servo_move"></block>
            <block type="servo_read_degrees"></block>
        </category>
    </category>
    <sep></sep>
    <category name="CAT_GROVE">
        <category name="CAT_GROVE_IN">
            <block type="grove_button"></block>
            <block type="grove_rotary_angle"></block>
            <block type="grove_tilt_switch"></block>
            <block type="grove_temporature_sensor"></block>
            <block type="grove_sound_sensor"></block>
            <block type="grove_pir_motion_sensor"></block>
            <block type="grove_line_finder"></block>
            <block type="grove_ultrasonic_ranger"></block>
            <block type="grove_thumb_joystick"></block>
        </category>
        <category name="CAT_GROVE_OUT">
            <block type="grove_led"></block>                                    //we
will use this one as example
            <block type="grove_piezo_buzzer"></block>
            <block type="grove_relay"></block>
            <block type="grove_motor_shield"></block>
            <block type="grove_rgb_led"></block>
        </category>
        <category name="CAT_GROVE_LCD">
            <block type="grove_serial_lcd_print"></block>
            <block type="grove_serial_lcd_power"></block>
            <block type="grove_serial_lcd_effect"></block>
        </category>
        <category name="CAT_GROVE_COMM">
            <block type="grove_bluetooth_slave"></block>
        </category>
    </category>
</xml>
```

The different block name will be used for the block declaration **and** the function: *parts #**2** & **#3*** ‘grove_led’ example.

The name are reference inside the file (en.js ; fr.js ; es.js; etc) with the text that will be dispalyed, so it’s the multilingual way: *part #**5** for* ‘grove_led’ example.

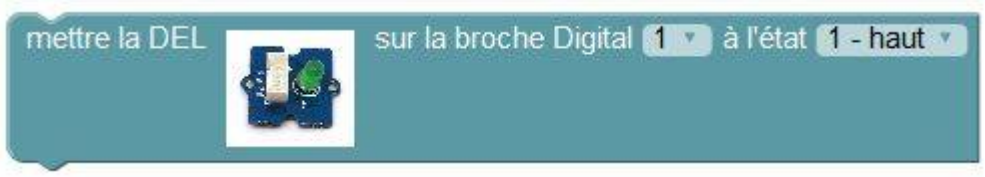
2 - define functions in a particular file '/generators/arduino/grove.js '. This file contains the way the block will create Arduino code, example of '*grove_led*' block quoted in ***part #1*** :

```
Blockly.Arduino.grove_led = function() {
  var dropdown_pin = Blockly.Arduino.valueToCode(this, 'NUM',
Blockly.Arduino.ORDER_ATOMIC);
  var dropdown_stat = this.getFieldValue('STAT');
  Blockly.Arduino.setups_['setup_green_led_'+dropdown_pin] =
'pinMode('+dropdown_pin+', OUTPUT);'; //will be declared in setup()
  var code = 'digitalWrite('+dropdown_pin+','+dropdown_stat+');\n'
//will be used in loop()
  return code;
};
```

3 - define blocks with the same name as function, in a specific file in '/blocks/grove/grove.js '. This file contains instructions for Blockly engine to display it:

```
Blockly.Blocks['grove_led'] = {
  init: function() {
    this.setColour(190);
    this.setHelpUrl(Blockly.Msg.GROVE_INOUT_LED_HELPURL); //relative
information to real text in /lang/blocks/en.js
    this.appendDummyInput()
        .appendField(Blockly.Msg.GROVE_INOUT_LED_INPUT1)
        .appendField(new
Blockly.FieldImage("http://www.seeedstudio.com/wiki/images/thumb/e/e0/LED1.j
pg/400px-LED1.jpg", 64, 64))
            //picture in local storage seems to be better idea, in the same
directory as in js block file definition

        .appendField(Blockly.Msg.GROVE_INOUT_LED_INPUT2)
        .appendField(new Blockly.FieldDropdown(profile.default.digital),
"PIN")
        .appendField(Blockly.Msg.GROVE_INOUT_LED_INPUT3)
        .appendField(new Blockly.FieldDropdown([["1 - HIGH", "HIGH"], ["0 -
low", "LOW"]]), "STAT");
               //you can also put FieldDropdown in Lang file
    this.setPreviousStatement(true, null);
    this.setNextStatement(true, null);
    this.setTooltip(Blockly.Msg.GROVE_INOUT_LED_TOOLTIP);
  }
};
```



4- add the name of this 2 file up in 'index.html' :

```
<html>
<head>
<link rel="icon" type="image/png" href="images/favicon.bmp" />
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Blockly Arduino</title>
<script type="text/javascript" src="blocks/blockly_compressed.js"></script>
<script type="text/javascript" src="blocks/blocks_compressed.js"></script>

<script type="text/javascript"
src="blocks/arduino_base/arduino_base.js"></script>          //block display
<script type="text/javascript" src="blocks/grove/grove.js"></script>

<script type="text/javascript"
src="generators/arduino/blocklyduino.js"></script>
<script type="text/javascript"
src="generators/arduino/arduino_base.js"></script>
<script type="text/javascript" src="generators/arduino/control.js"></script>
<script type="text/javascript" src="generators/arduino/grove.js"></script>
//block function for Arduino code
```

5 - at last translating in different language file.

→ If needed , **in /lang/msg/en.js** you can add global specification:

```
var MSG = {
  title: "éditeur graphique pour aider à la programmation des interfaces
Arduino",
  labelArduinoCard: "carte Arduino :",
  span_config: " configurer les blocs",
  span_delete: " effacer TOUS les blocs",
  span_saveXML:" sauver en fichier XML",
```

→ Then in **/lang/blocks/en.js** we add all the text displayed:

```
Blockly.Msg.ARDUINO_SERIAL_PRINT_CONTENT = "envoyer sur le port série la
donnée :";
Blockly.Msg.ARDUINO_SERIAL_PRINT_TOOLTIP = "envoie des données sur le port
série pour surveillance par le moniteur en ASCII";

Blockly.Msg.GROVE_INOUT_LED_HELPURL =
"http://www.seeedstudio.com/wiki/index.php?title=GROVE_-_Starter_Bundle_V1.0
b#LED";
Blockly.Msg.GROVE_INOUT_LED_INPUT1 = "mettre la DEL";
Blockly.Msg.GROVE_INOUT_LED_INPUT2 = "sur la broche Digital";
Blockly.Msg.GROVE_INOUT_LED_INPUT3 = "à l'état";
Blockly.Msg.GROVE_INOUT_LED_TOOLTIP = "active la sortie Digital sur laquelle
la DEL est branchée";
```

6 - tweak !

You could also change the way the block is created or displayed, example with this different version:

```
Blockly.Blocks['grove_led'] = {
  init: function() {
    this.setColour(190);
    this.setHelpUrl(Blockly.Msg.GROVE_INOUT_LED_HELPURL);
    this.appendDummyInput()
        .appendField(Blockly.Msg.GROVE_INOUT_LED_INPUT1)
        .appendField(new
Blockly.FieldImage("http://www.seeedstudio.com/wiki/images/thumb/e/e0/LED1.j
pg/400px-LED1.jpg", 64, 64))
    this.appendValueInput("PIN", 'Number')
        .setCheck('Number')
        .setAlign(Blockly.ALIGN_RIGHT)
        .appendField(Blockly.Msg.GROVE_INOUT_LED_INPUT2);
    //this.setInputsInline(true);
    this.appendDummyInput("")
        .setAlign(Blockly.ALIGN_RIGHT)
        .appendField(Blockly.Msg.GROVE_INOUT_LED_INPUT3)
        .appendField(new Blockly.FieldDropdown([["1 - haut", "HIGH"], ["0 -
bas", "LOW"]]), "STAT");
    this.setPreviousStatement(true, null);
    this.setNextStatement(true, null);
    this.setTooltip(Blockly.Msg.GROVE_INOUT_LED_TOOLTIP);
  }
};
```

if you uncomment line *`this.setInputsInline(true);`* *here what you will see:*