Table des matières

1. Succinctement pour créer un bloc, voici la démarche en prenant pour exemple les blocs Grove : 3

1. Succinctement pour créer un bloc, voici la démarche en prenant pour exemple les blocs Grove :

1 - copier le fichier arduino.xml situé dans /blocks/toolbox, ** le renommer (// grove.xml // **) en fonction des blocs à afficher dans la Toolbox (le menu de gauche) et rajouter des entrées en bas du fichier : <code> <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> //on va se servir de cet exemple de fonction <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> </code> Tous ces noms de fonctions serviront comme titre de chaque fonction dans les définitions des blocs : //points **3** & **4** ci-après// pour l'exemple 'grove_led'.**

Ils sont une référence à un fichier qui contient le texte, autorisant le multilinguisme du site : *point **6** ci-après* pour l'exemple 'grove_led'.

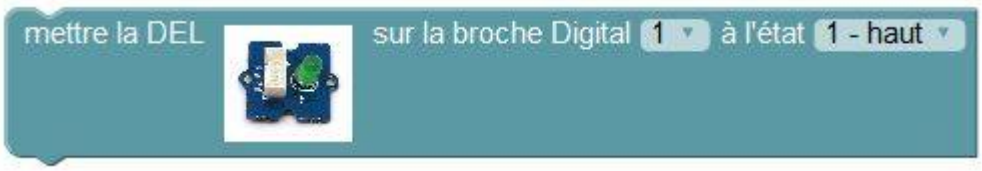
2 - dans le fichier /lang/code.js rajouter à partir de la ligne 83 la référence à ce fichier **grove.xml ** créé au point** //1**//, et qui sera ainsi dans le menu déroulant des choix :

```
Code.TOOLBOX_NAME = {
        'standard':'Standard',
        'arduino':'Arduino',
        'grove':'Grove'
};
```

ATTENTION ! Le dernier ne doit pas avoir de virgule en fin de ligne.

3 - définir les fonctions dans un fichier spécifique situé dans '/generators/arduino/grove.js '. ** Ce fichier contient les procédures pour écrire dans le code Arduino, exemple du bloc '*grove_led*' que l'on voyait dans le fichier ** grove.xml dans le// point 1 **//: <code> Blockly.Arduino.grove_led = function() { var dropdown_pin = this.getFieldValue('PIN'); var dropdown_stat = this.getFieldValue('STAT'); Blockly.Arduino.setups_['setup_green_led_'+dropdown_pin] = 'pinMode('+dropdown_pin+', OUTPUT);'; //texte nécessaire dans le setup() var code = 'digitalWrite('+dropdown_pin+','+dropdown_stat+');\n' //création de l'instruction dans la loop() return code; }; </code> **4 - définir les blocs correspondant aux fonctions dans un

fichier spécifique situé dans '/blocks/grove/<font inherit/inherit;;#FF8C00;;inherit>grove.js '. ** Ce fichier contient les détails des blocs graphiques : <code> Blockly.Blocks[‘grove_led’] = { init: function() { this.setColour(190); this.setHelpUrl(Blockly.Msg.GROVE_INOUT_LED_HELPURL); //référence relative qui sera complété par /blocks/fr.js this.appendDummyInput() .appendField(Blockly.Msg.GROVE_INOUT_LED_INPUT1) .appendField(new Blockly.FieldImage(“http://www.seeedstudio.com/wiki/images/thumb/e/e0/LED1.jpg/400px-LED1.jpg”, 64, 64)) //privilégier les images en local dans le dossier du fichier js de définition .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 - haut", "HIGH"], ["0 - bas", "LOW"), “STAT”); this.setPreviousStatement(true, null); this.setNextStatement(true, null); this.setTooltip(Blockly.Msg.GROVE_INOUT_LED_TOOLTIP); } }; </code>



5- rajouter les références à ces 2 fichiers dans le haut de ‘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.js"></script>
//définition des dessins des blocs
<script type="text/javascript" src="blocks/grove.js"></script>
<script type="text/javascript" src="generators/arduino/arduino.js"></script>
<script type="text/javascript" src="generators/arduino/base.js"></script>
<script type="text/javascript"
src="generators/arduino/controls.js"></script>
<script type="text/javascript" src="generators/arduino/grove.js"></script>
//définition des fonctions produisant le code
```

6 - on termine avec les traductions afin que les textes soient appelés en ‘relatif’ depuis les fichiers de langue.

→ Tout d’abord dans **/lang/msg/fr.js** on rajoute les entrées générales : catégories & titres des boutons.

```
Blockly.Msg.CAT_ARDUINO_SERVO = "servo-moteur";
Blockly.Msg.CAT_GROVE = "Grove";
Blockly.Msg.CAT_GROVE_IN = "capteurs";
Blockly.Msg.CAT_GROVE_OUT = "actionneurs";
Blockly.Msg.CAT_GROVE_LCD = "écran LCD";
Blockly.Msg.CAT_GROVE_COMM = "communication";
```

→ Ensuite dans **/lang/blocks/fr.js** on rajoute toutes les définitions des textes visibles dans les blocs :

```
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";
```

7 - on tweak !

Il suffit de modifier les textes qui sont ainsi TOUS contenus dans** /lang/blocks/fr.js** et permet d'aller vite pour remplacer par exemple '*Digital*' par '*Numérique*'.

On peut aussi changer la nature du bloc facilement, à vous de tester cette autre 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);
  }
};
```

et si vous décommentez (*enlevez les) // la ligne '' '' //'this.setInputsInline(true);* '' *vous obtiendrez :*

From:
https://wiki.libreduc.cc/ - **LibrEduc**

Permanent link:
https://wiki.libreduc.cc/es:arduino:blocklyduino:block

Last update: **2025/01/16 20:24**