如何为 Mixly 写一个公司库？

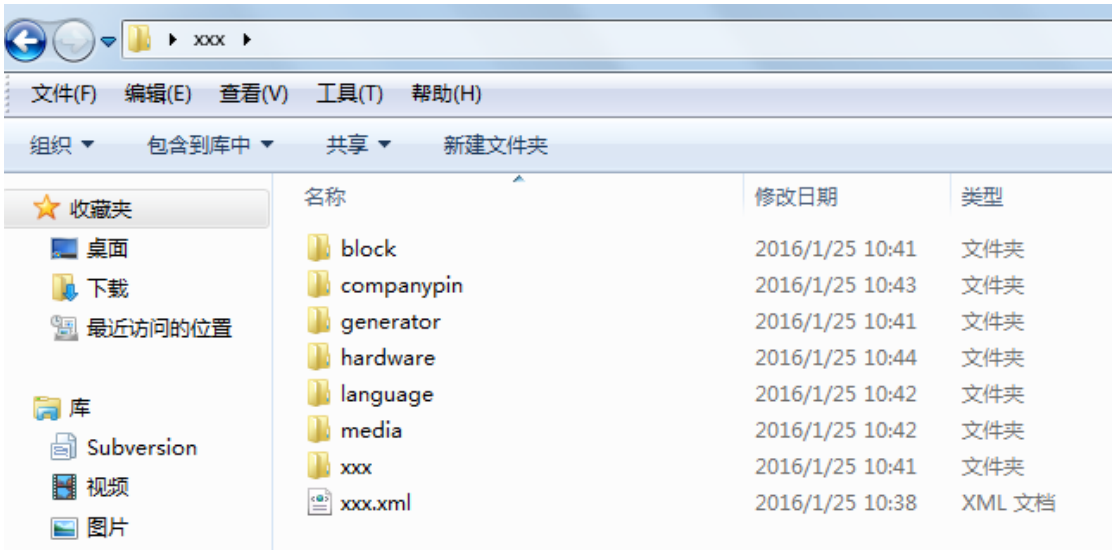
本文档适用于 Mixly0.964 及更高版本

目录

如何为 Mixly 写一个公司库？ 1
一、 一个完整的公司库的目录 1
二、 block 和 generator 2
三、 hardware 目录 3
四、 xxx 目录 3
五、 companypin 目录 3
六、 language 目录 3
七、 media 目录 5
八、 xxx.xml 文件 5
九、 把写好的库导入 Mixly 6

一、 一个完整的公司库的目录

下面是一个名为 xxx 的公司库的完整目录：



该目录下不是所有的文件都是必须的，视公司库的具体情况而定。

下面就对每一个目录进行说明。

二、block 和 generator

block 和 generator 文件夹下分别有一个 JS 文件，一般是同名的，比如 xxx.js。

block/xxx.js 文件定义了你定制的图形化模块的样子。

generator/xxx.js 文件定义了每一个图形化模块对应的 Arduino 代码。

比如下图就是“延时”的图形化模块的样子及其对应的 Arduino 代码：


```
void setup()
{

}


void loop()
{
  delay(1000);

}
```

block 和 generator 目录通常是必不可少的，因此保证 **block/xxx.js** 和 **generator/xxx.js** 的正确性是成功定制公司库的关键。

比如“延时”模块的两部分 JS 代码分别如下：

```
Blockly.Blocks.base_delay = {
  init: function() {
    var UNIT =
        [[Blockly.LKL_DELAY_MS, 'delay'],
         [Blockly.LKL_DELAY_US, 'delayMicroseconds']];
    this.setColour(Blockly.Blocks.loops.HUE);
    this.appendValueInput("DELAY_TIME", Number)
        .appendTitle(Blockly.LKL_DELAY)
        .appendTitle(new Blockly.FieldDropdown(UNIT), 'UNIT')
        .setCheck(Number);
    this.setInputsInline(true);
    this.setPreviousStatement(true, null);
    this.setNextStatement(true, null);
    this.setTooltip(Blockly.LKL_TOOLTIP_CONTROL_DELAY);
  }
};
```

```
Blockly.Arduino.base_delay = function() {
  var delay_time = Blockly.Arduino.valueToCode(this, 'DELAY_TIME', Blockly.Arduino.ORDER_ATOMIC) || '1000'
  var unit = this.getTitleValue('UNIT');
  var code = unit+'(' + delay_time + ');\n';
  return code;
};
```

对于这两个 JS 文件如何书写，这里不再列举更多的例子，因为在软件的 blockly/blocks 和 blockly/generators/arduino 目录下有大量的例子，这里面的例子基本包含了全部类型的图形化模块的定制方法。当然，如果您想更深入的了解这些代码，您可以访问 Google 的 blockly

源码进行更加深入的研究。

三、 hardware 目录

hardware 目录不是必需的目录，通常在公司使用了自己的 Arduino 板子（即官方的 Arduino IDE 没有我们所需要的板子）时，才需要 hardware 目录，这个目录包含了一些跟硬件相关的信息，比如板子信息，管脚信息等等。因此，这个目录需要比较专业的技术人员进行定制。最后是一个以公司名，比如 xxx 的文件夹形式存在，直接拷贝到 hardware 目录下即可。

四、 xxx 目录

xxx 目录是自己的 Arduino 库，xxx 一般以公司名命名，里面是一些.cpp 和.h 文件，这个目录也不是必需的，一般在生成的代码有**#include <xxx.h>**时需要。

五、 companypin 目录

companypin 目录定义了公司所需板子的管脚对应关系，该目录是非必需的，通常在公司使用了全新的 Arduino 板子时才需要。

下面以 lubot 的新板子 Lubot MK 为例：

我们想为这块板子定义新的管脚对应关系，我们先在 companypin 目录下建立子文件夹 lubot（通常以公司名命名），再建立 pin.js 文件，打开 pin.js 文件，管脚定义如下图所示（包括数字管脚、模拟管脚、PWM 管脚、中断管脚以及 Serial 选择和默认波特率）：

pin.js

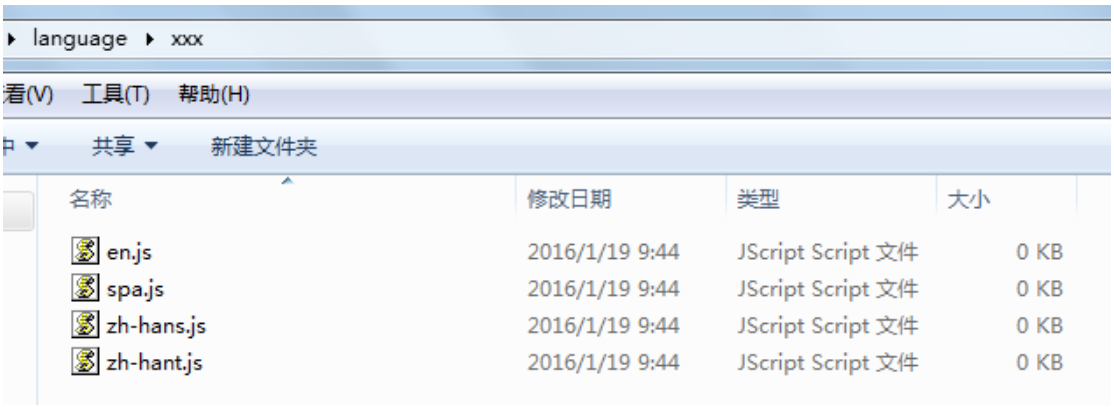
```
profile["LuBot MK"]= {
    description: "lubot",
    digital: [["V0", "V0"],["V1", "V1"], ["V2", "V2"], ["V3", "V3"],["A0", "A0"], ["A1", "A1"], ["A2", "A2"], ["A3", "A3"]],
    analog: [["A0", "A0"], ["A1", "A1"], ["A2", "A2"], ["A3", "A3"]],
    pwm:[["V0", "V0"],["V1", "V1"], ["V2", "V2"], ["V3", "V3"]],
    interrupt : [["V0", "V0"]],
    serial_select:[["Serial", "Serial"]],
    serial: 9600
}
```

注：pin.js 中可定义多块板子，即多个 profile["板子名称"]。

六、 language 目录

如果想让公司库支持多国语言，就需要 language 目录，这个也是非必需的。

先在 language 目录下建立子文件夹 xxx（通常以公司名命名），再在 xxx 文件夹下建立多国语言文件。如下图所示：分别是英语、西班牙语、简体中文、繁体中文。在这些多国语言的 js 文件中定义了一些 JS 常量。



下面讲解如何定义这些 js 文件，还是以“延时”模块为例，下面分别是英语、西班牙语、简体中文、繁体中文对应的样子：

再来看如何在语言文件中定义对应的文字。

en.js

```
Blockly.LKL_MAP_TO="] to [";
Blockly.LKL_MILLIS='millis';
Blockly.LKL_DELAY='Delay';
Blockly.LKL_ATTACHINTERRUPT_PIN='attachInterrupt pin#';
Blockly.LKL_DETACHINTERRUPT_PIN='detachInterrupt pin#';
```

spa.js

```
Blockly.LKL_MAP_TO="] hasta [";
Blockly.LKL_MILLIS='milisg';
Blockly.LKL_DELAY='Retardo';
Blockly.LKL_ATTACHINTERRUPT_PIN='conectarInterrupción pin#';
Blockly.LKL_DETACHINTERRUPT_PIN='desconctarInterrupción pin#';
```

zh-hans.js

```
Blockly.LKL_MAP_TO="] 到 [";
Blockly.LKL_MILLIS='毫秒';
Blockly.LKL_DELAY='延时';
Blockly.LKL_ATTACHINTERRUPT_PIN='中断  管脚#';
Blockly.LKL_DETACHINTERRUPT_PIN='取消中断  管脚#';
```

zh-hant.js

```
Blockly.LKL_MAP_TO="]到[";
Blockly.LKL_MILLIS='毫秒';
Blockly.LKL_DELAY='延時';
Blockly.LKL_ATTACHINTERRUPT_PIN='中斷管脚#';
Blockly.LKL_DETACHINTERRUPT_PIN='取消中斷管脚#';
```

最后在 block/xxx.js 文件中引用即可：

```
Blockly.Blocks.base_delay = {
  init: function() {
    var UNIT =
        [[Blockly.LKL_DELAY_MS, 'delay'],
         [Blockly.LKL_DELAY_US, 'delayMicroseconds']];
    this.setColour(Blockly.Blocks.loops.HUE);
    this.appendValueInput("DELAY_TIME", Number)
        .appendTitle(Blockly.LKL_DELAY)
        .appendTitle(new Blockly.FieldDropdown(UNIT), 'UNIT')
        .setCheck(Number);
    this.setInputsInline(true);
    this.setPreviousStatement(true, null);
    this.setNextStatement(true, null);
    this.setTooltip(Blockly.LKL_TOOLTIP_CONTROL_DELAY);
  }
};
```

七、media 目录

media 目录下是一些媒体文件，主要是图片，这个目录是非必需的。只有当我们的模块需要使用图片是才需要该文件夹。比如下面这个模块里面就嵌入了一张图片：

所有的媒体文件放在 media/xxx 下即可，xxx 是子文件夹，通常以公司名命名。

八、xxx.xml 文件

xxx.xml 文件是非常重要的一个文件，它是必不可少，通常以公司名命名。在这个文件中定义了所有需要呈现的模块，以及整个库中相关文件的路径（也就前面提过的这些文件目录）。

下面是一个空例子，可以基于这个例子进行修改。

xxx.xml

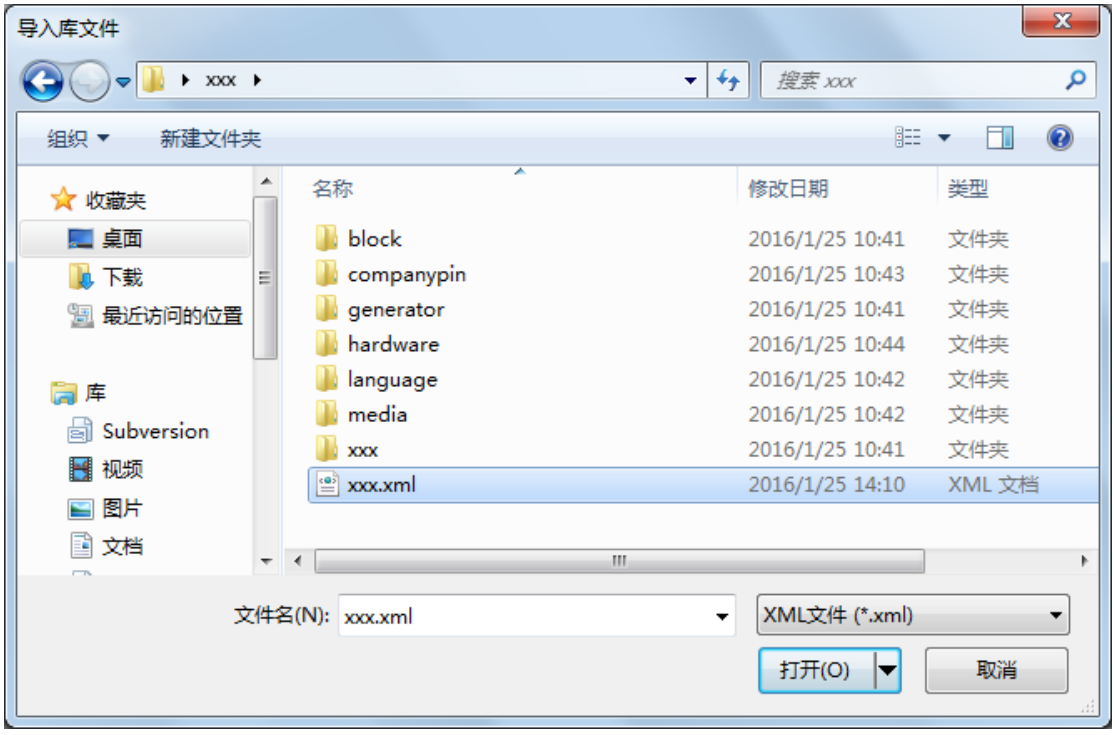
```
<!-- type="company"
     block="block/xxx.js"
     generator="generator/xxx.js"
     lib="xxx"
     hardware="hardware/xxx"
     media="media/xxx"
     language="language/xxx"
     pin="companypin/xxx" -->
  <script type="text/javascript" src="../../blocks/company/xxx.js"></script>
  <script type="text/javascript" src="../../generators/arduino/company/xxx.js"></script>
  <!--引用所有写好的模块，这里引入了"延时"模块-->
<category name="xxx" colour="120">
    <block type="base_delay"></block>
</category>
```

九、 把写好的库导入 Mixly

当我们把整个库写好之后，就可以把写好的库导入到 Mixly 中去了。打开 Mixly 软件（版本要求 0.964 及以上）。

点击“导入库”

找到 xxx.xml 文件



点击打开就完成整个公司库的导入（当然，前提是公司库必须正确）。