

## 專案報告

專案名稱：ArduBlock 新增選單功能

功能描述：

1. 新增感測器的 Block 如附件 1
2. 提供新增 Block 的 SOP (標準製作程序)

所需材料：

1. PlayDuino/Arduino 控制板
2. 各項感測器如附件 1

所需工具：

1. Eclipse

設計規劃：

1. 如附件 1

檔案列表

1. Java\_pro.7z
2. repository.7z
3. ardublock-all.jar

颯機器人 PlayRobot Inc. 提供

## 一、開發環境

Ardublocka 是使用 Maven 管理的開發專案，所以期開發工具如下：

- 1、[Java SDK](#)
- 2、[Maven](#)
- 3、[eclipse](#)
- 4、[eclipse Maven plugin](#)

### 安裝步驟

- 1、下載安裝 Java SDK

依照上方連結下載，安裝如一般安裝流程，在此不詳述。

- 2、下載安裝 Maven

依照上方連結下載 Maven 如圖.1，下載完成後將其解壓縮到任意目錄如圖.2，環境變數設定，增加 M2\_HOME，內容即為解壓縮路徑。在 Path 裡增加 %M2\_HOME%\bin。完後切換 cmd 畫面輸入 mvn -version，若成功即顯示 maven 版本。

在 maven 解壓目錄下 conf/setting.xml 設置資源庫位置為（找 localRepository 標籤） C:\Users\Administrator\.m2\repository (windows7) C:\Documents and Settings\Administrator\.m2\repository (windows XP)，Administrator 視自己使用者名稱設定。

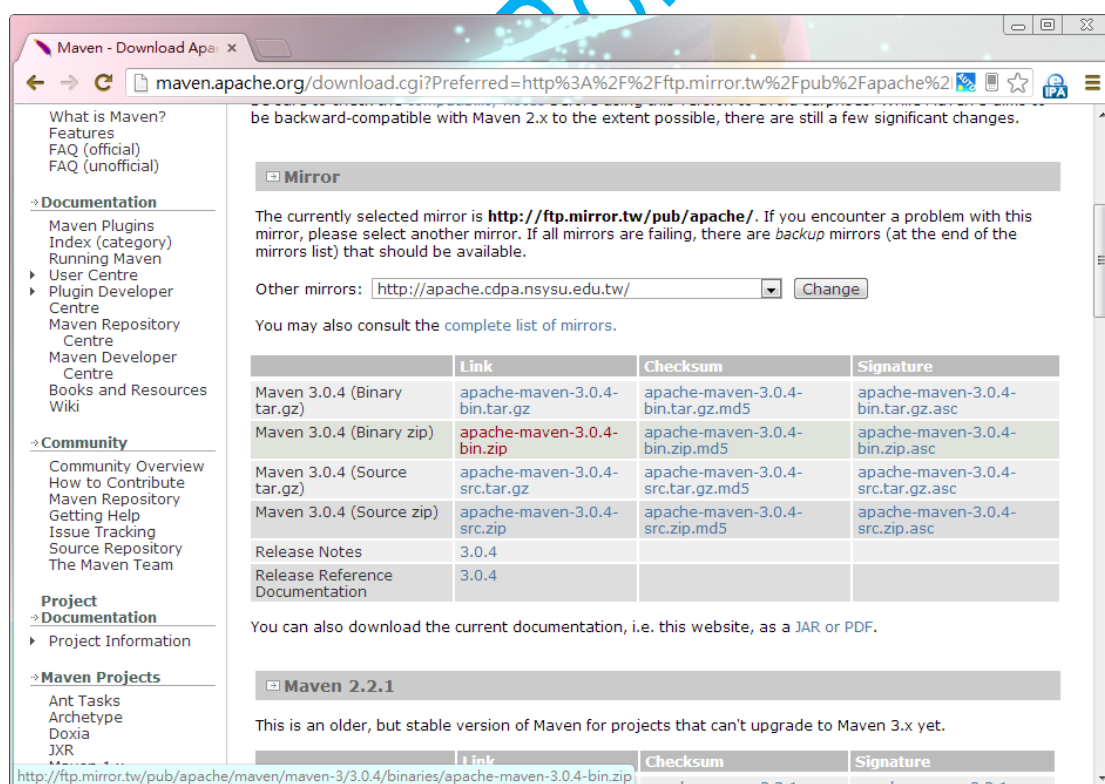


圖.1

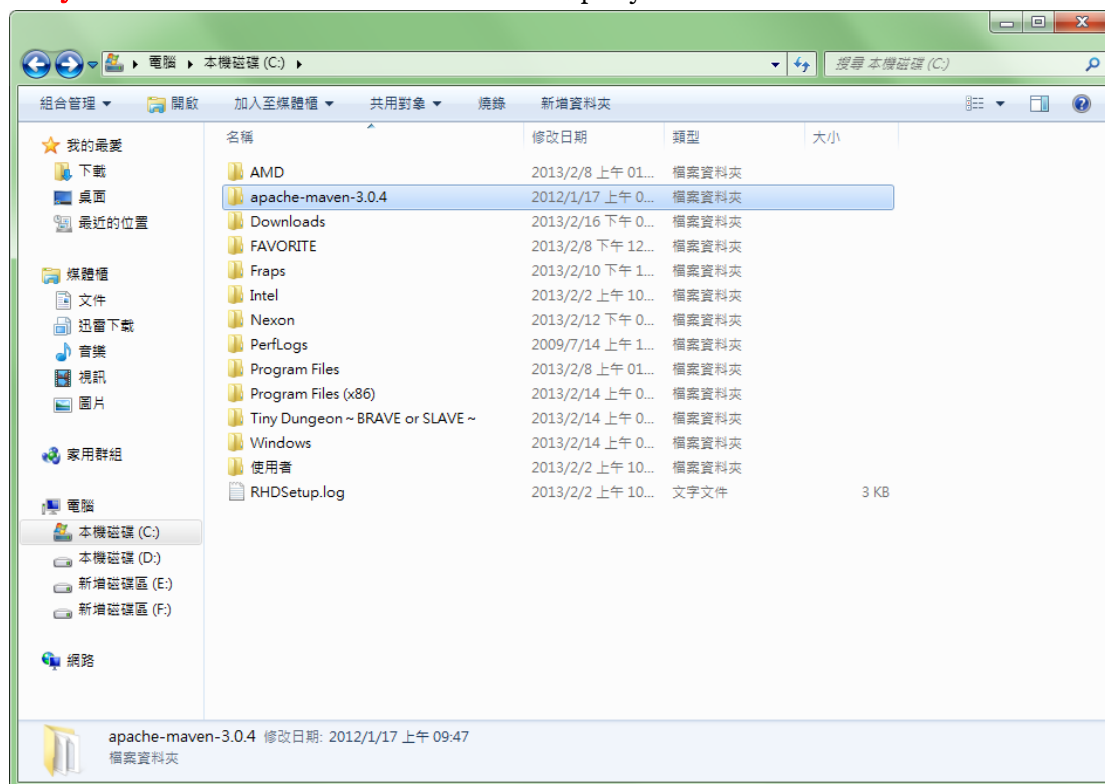


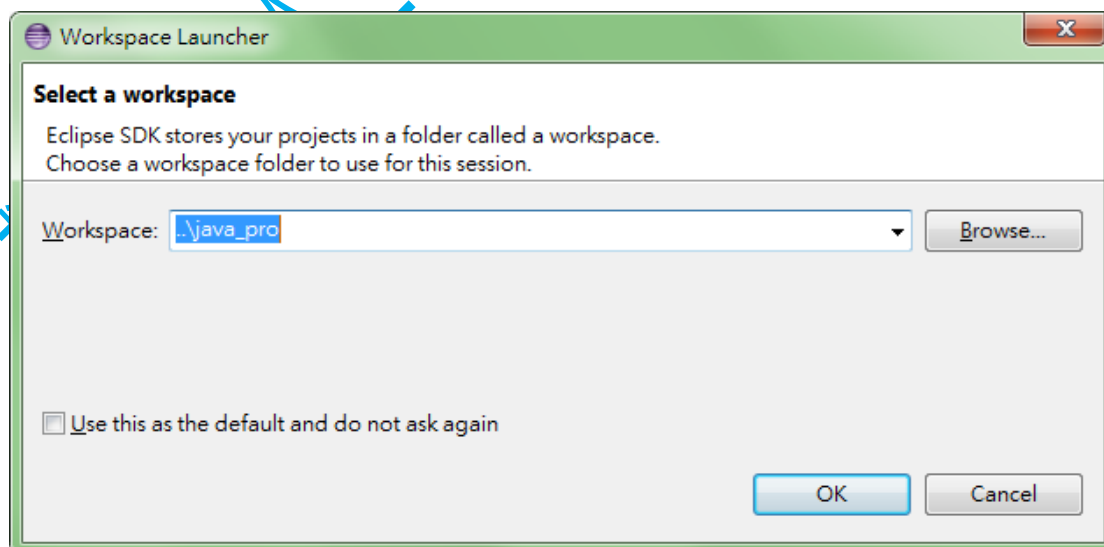
圖.2

### 3、下載安裝 eclipse

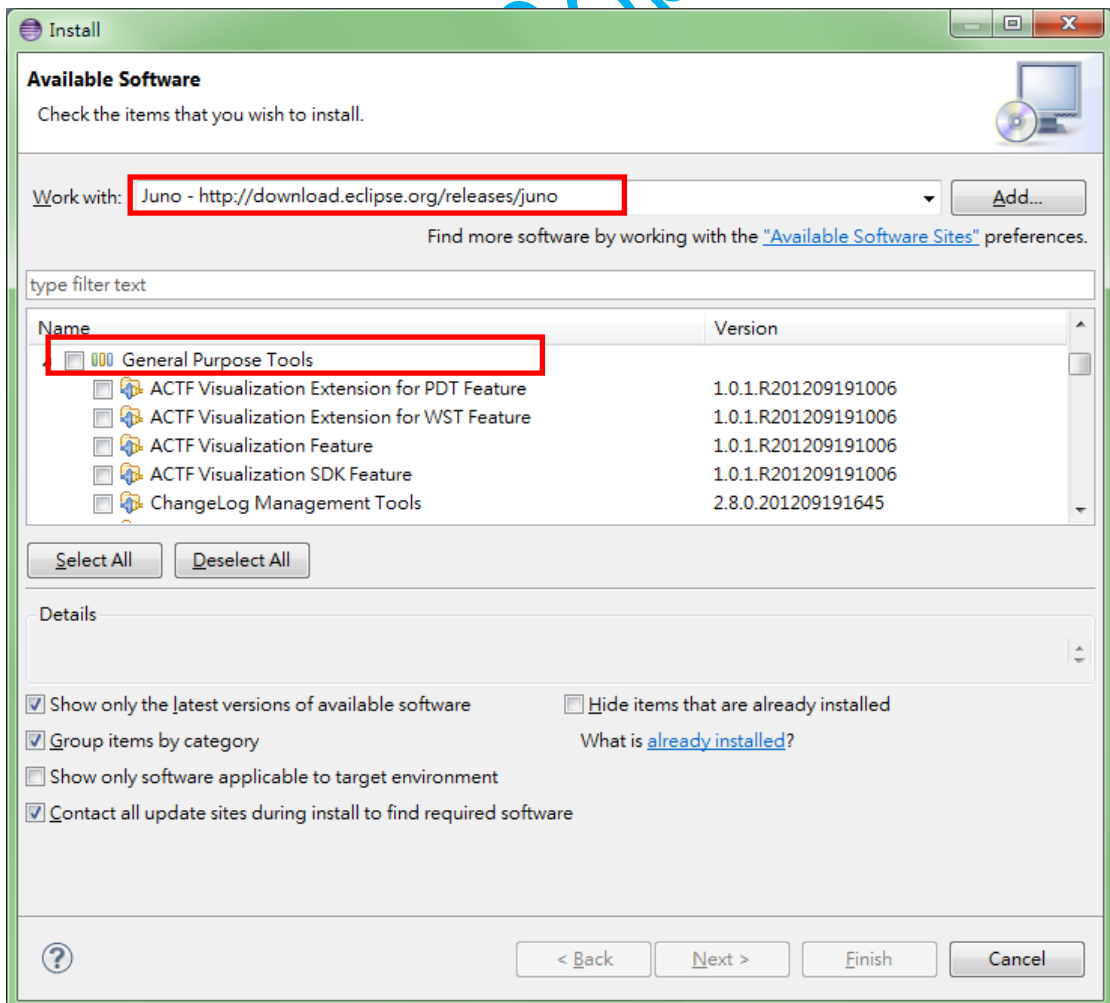
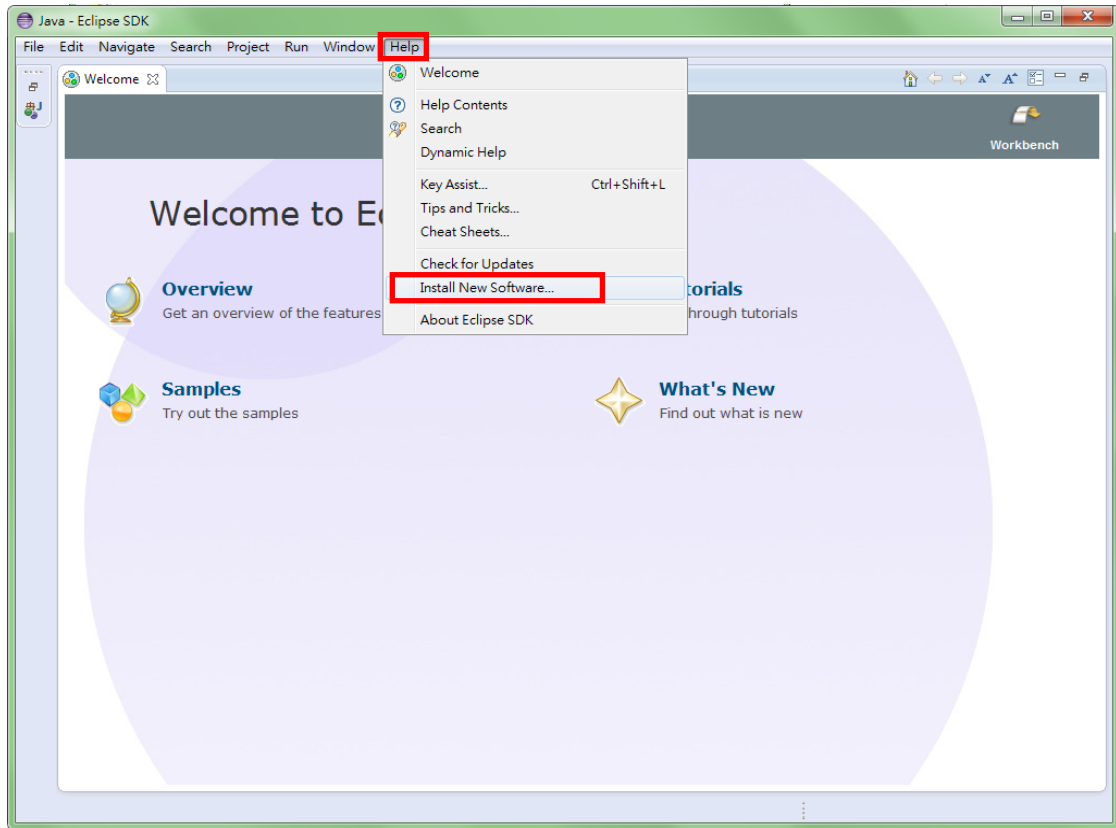
依照上方連結下載 eclipse，下在完成後將其解壓縮到任意目錄。

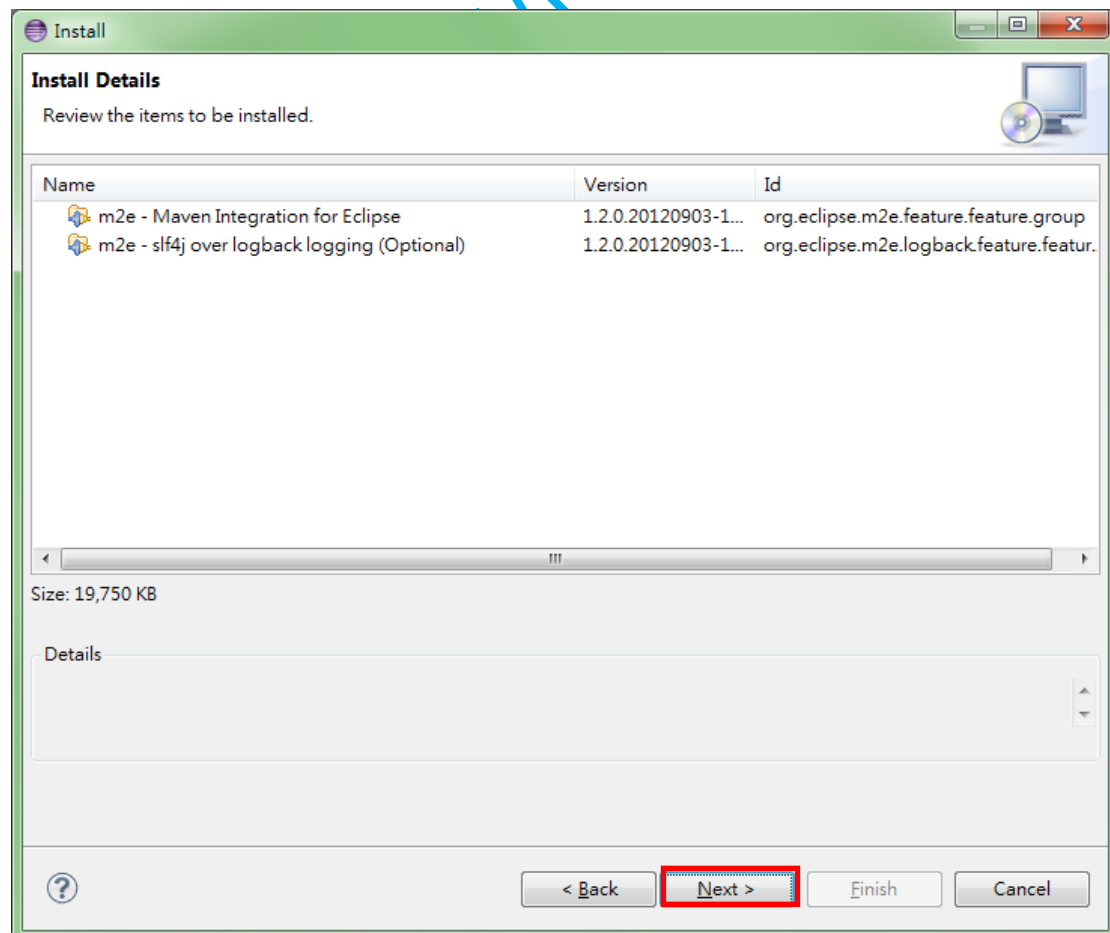
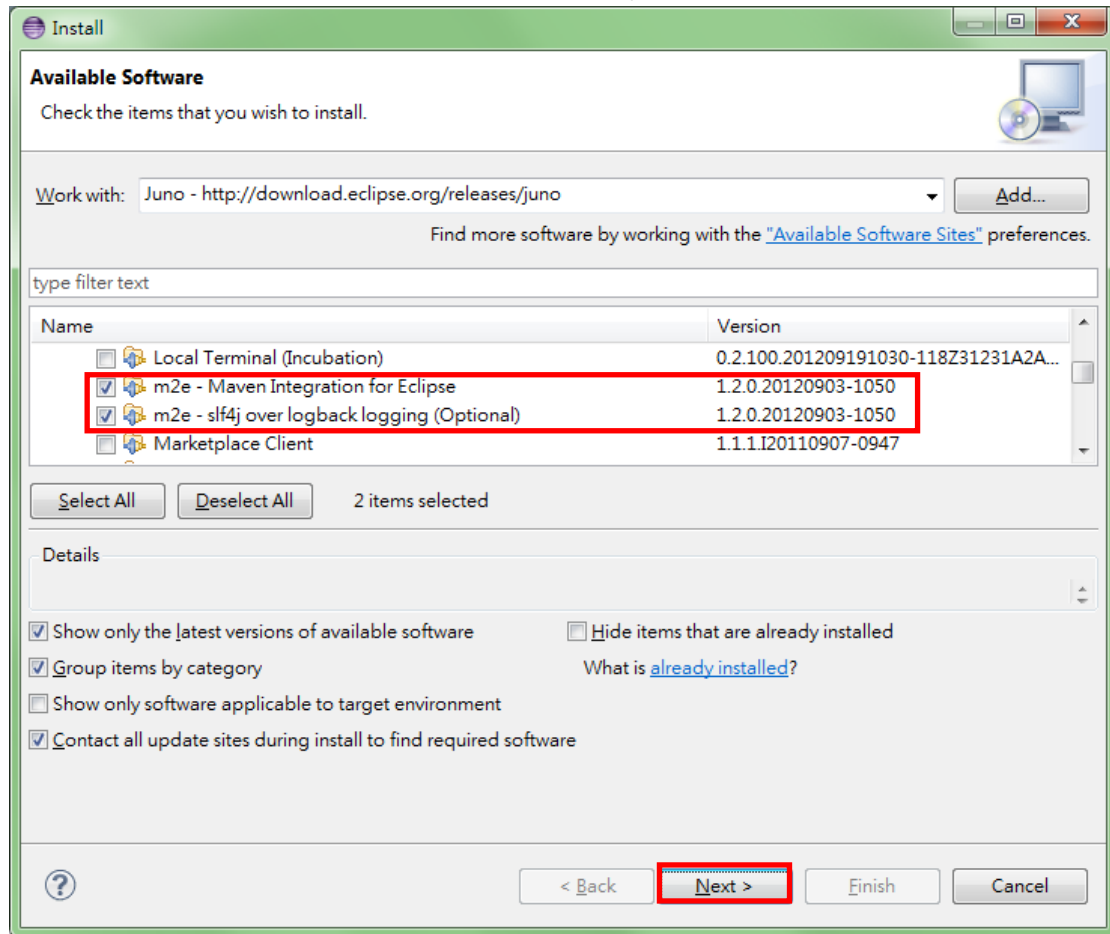
### 4、安裝 eclipse Maven plugin

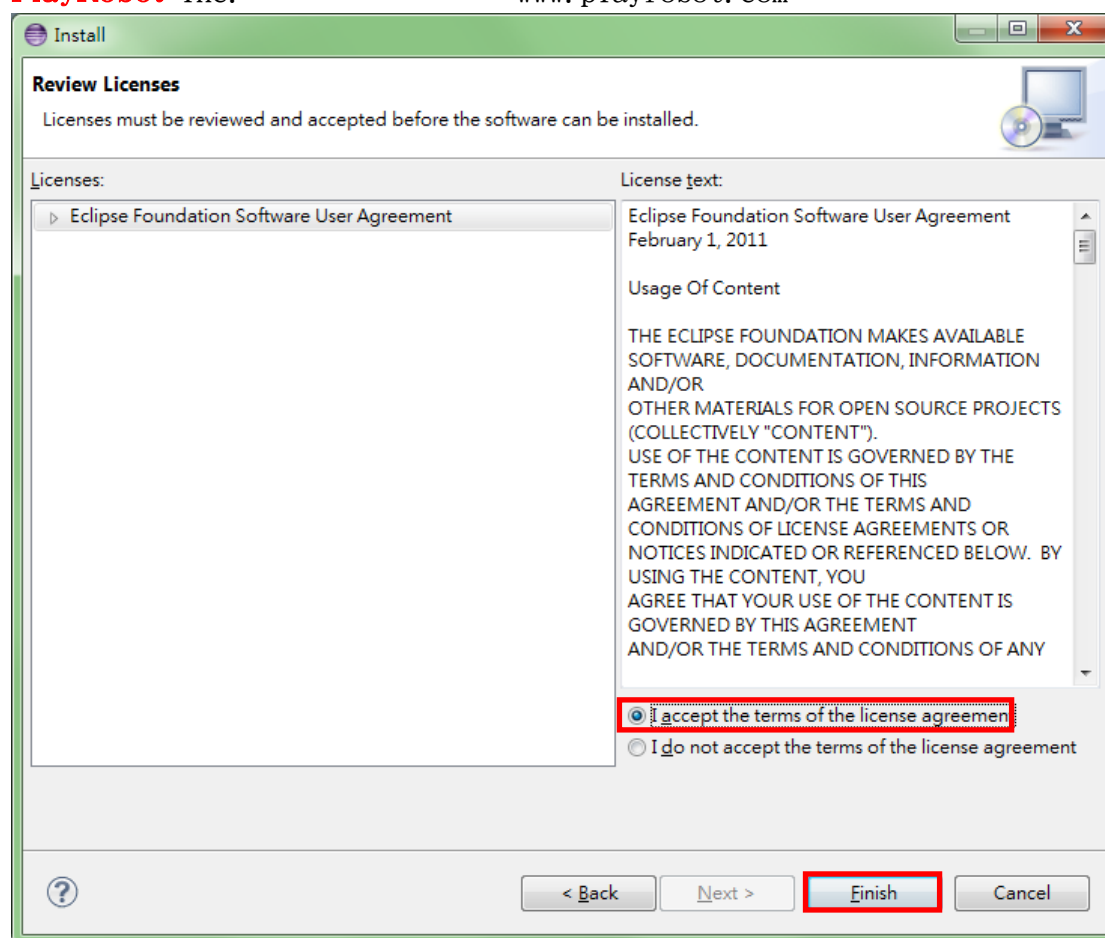
在 eclipse 的壓縮目錄執行 eclipse.exe 開啟 eclipse 開發工具，依據喜好設定專案目錄。點選 eclipse 工具列 => Help => Install New



Software => 選擇 eclipse 內建好的版本 => General Purpose Tools => 選擇 m2e 開頭的項目全部安裝 => restart







## 5、設定 eclipse

點選 eclipse 工具列 => preferences => General => Workspace => Text file encoding 設為 UTF-8。

點選 eclipse 工具列 => preferences => General => Content Types => Text => Java Properties File : Default encoding 設為 UTF-8。

點選 eclipse 工具列 => preferences => maven => 一般只勾 debug output，其他視情況需求勾選。

點選 eclipse 工具列 => preferences => maven => installations => add => 選擇 maven 解壓縮位置 => 下方 Global settings from installation directory 會變成解壓縮位置之 settings.xml。

點選 eclipse 工具列 => preferences => maven => User Settings => User Settings => 選擇解壓縮位置之 settings.xml，下方之 Local Respository 不予更動。

## 6、重製專案

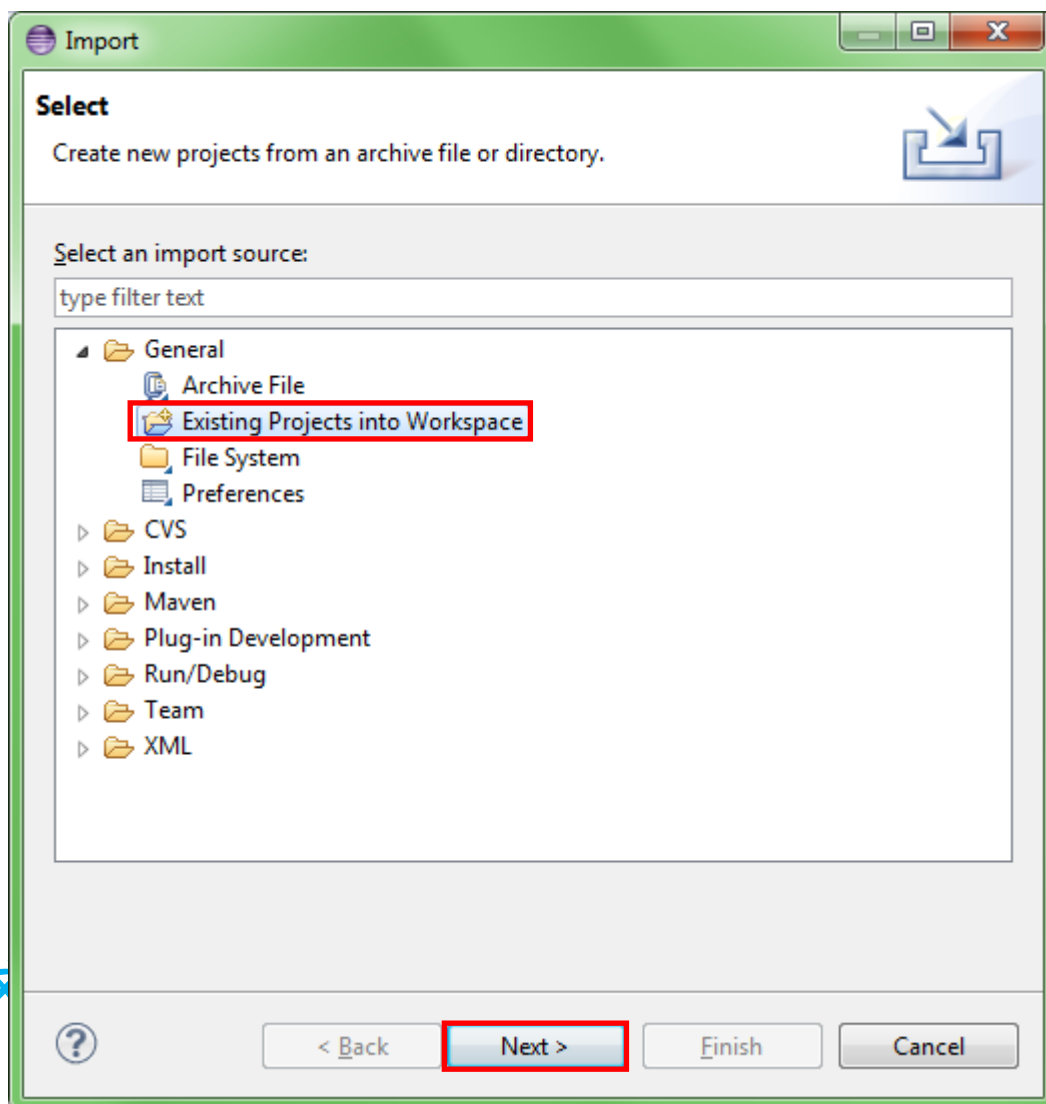
將 Java\_pro2.7z 解壓縮到任意目錄會得到 openblocks、ardublock 專案

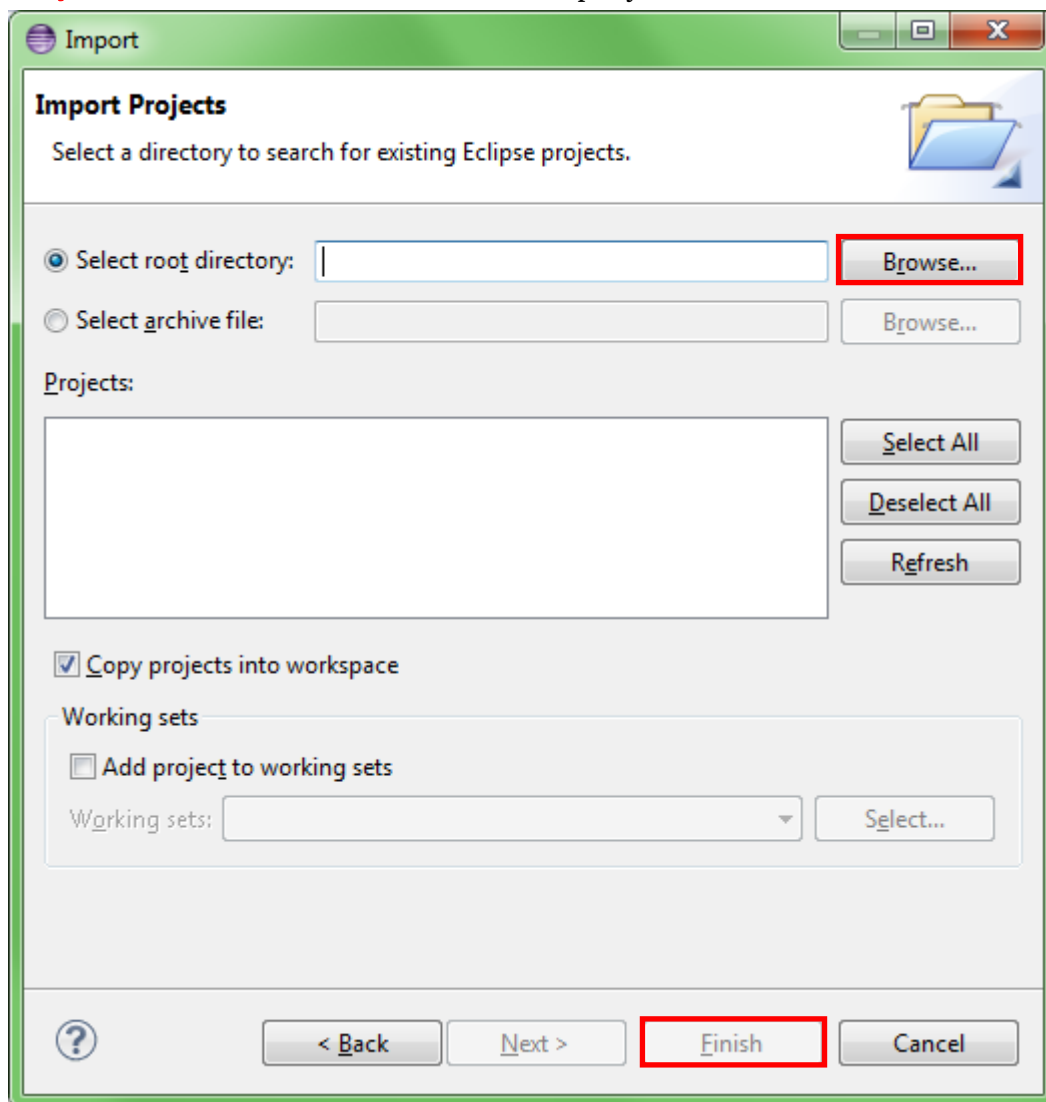
點選 eclipse 工具列 => import => General => Existing Project Into Workspace => Next => 選擇 openblocks 專案 => Copy Project Into Workspace => Finish

點選 eclipse 工具列 =>import=>General=>Existing Project Into Workspace=>Next=> 選擇 ardublock 專案 =>Copy Project Into Workspace =>Finish

錯誤處理

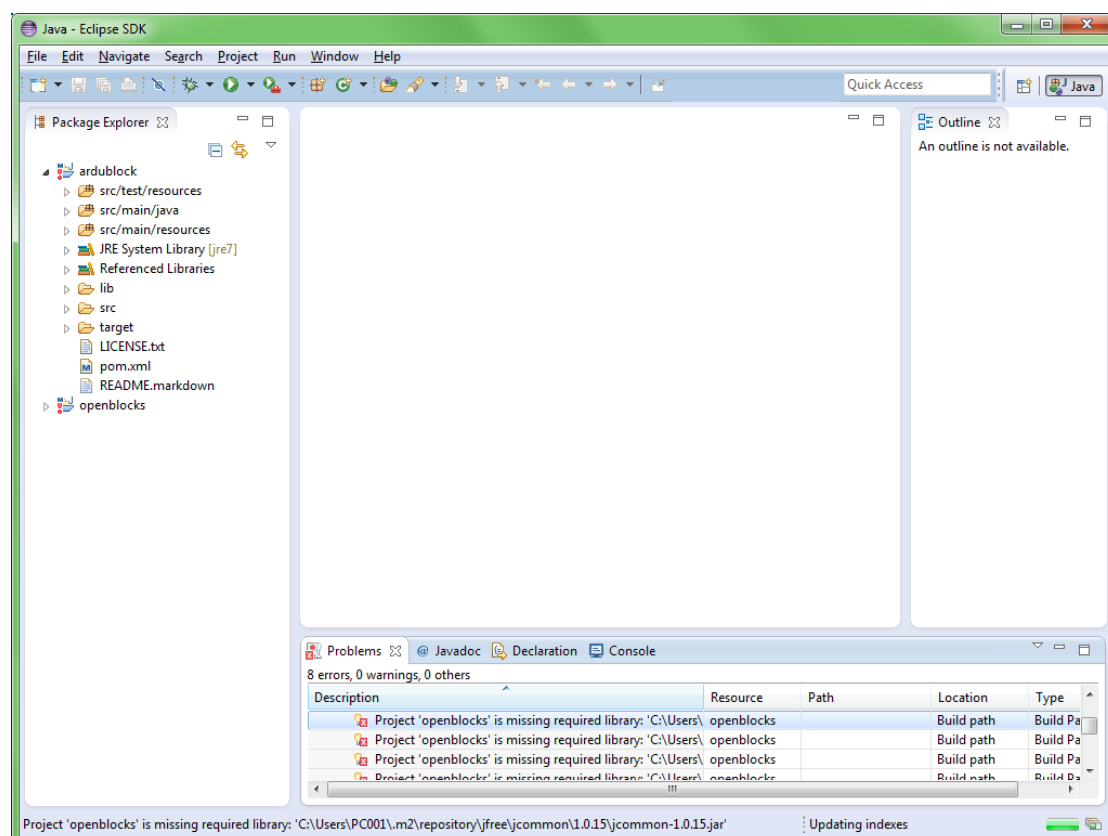
如果出現 is missing required library 錯誤，請將 repository.7z 解壓縮到 C:\Users\Administrator\.m2\repository (windows7) 或 C:\Documents and Settings\Administrator\.m2\repository (windows XP) 中取代之，Administrator 視自己使用者名稱設定。





颯機器人





## 二、加入元件 block 的標準作業流程

依據作者的說明文件想要在 ardublock 中加入自己的 block 至少必須更動三個檔案分別為：

- 1、ardublock.properties 位於 src/main/resources/com/ardublock/block 用於設定 block 的文字標籤。(如需設定繁體中文於 ardublock\_zh\_TW.properties 中設定)
- 2、ardublock.xml 位於 src/main/resources/com/ardublock/block 用於設定 block 的顯示介面群組。
- 3、TranslatorBlockFactory.java 位於 src/main/java/com/ardublock/translator/block 用於設定 block 的轉換類別。

所以綜合以上所述加入 block 的標準作業流程如下：

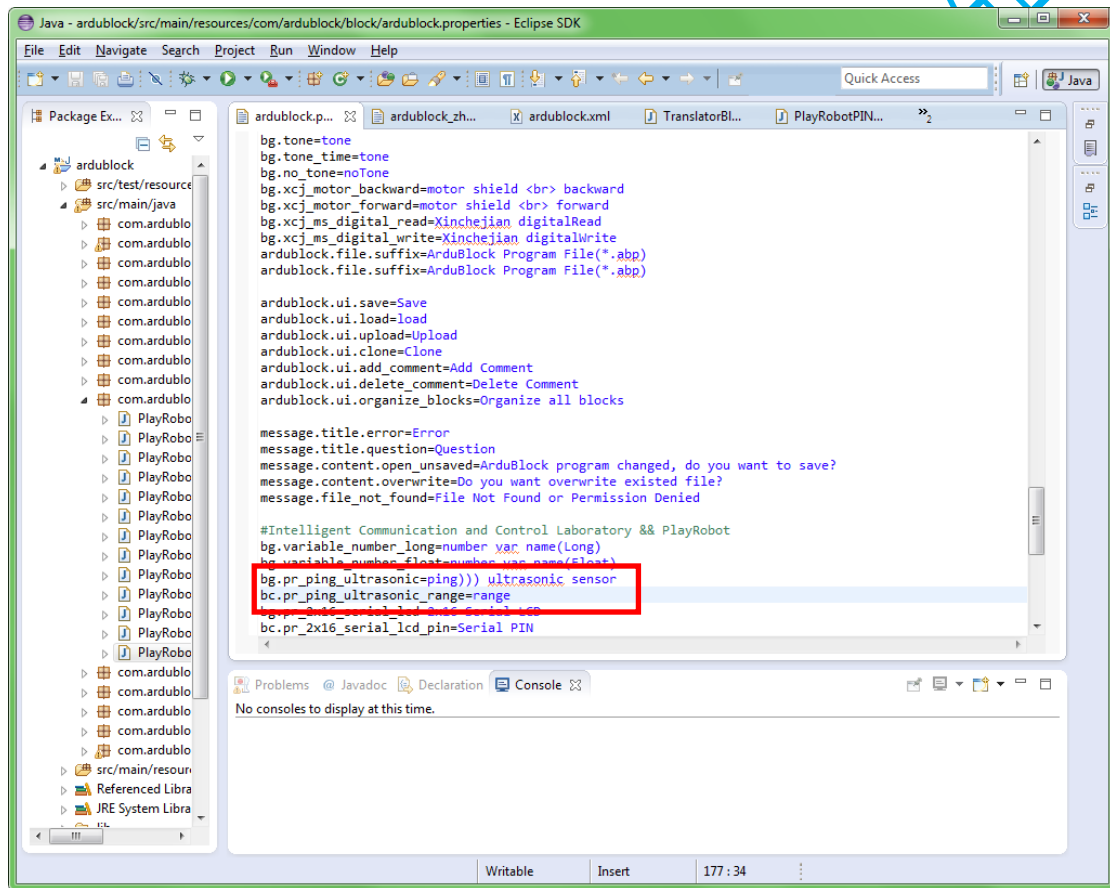
- 1、決定 block 的型態，建議可先用 arduino 開發工具撰寫對應元件 block 的程式再決定 block 對應的輸出輸入
- 2、依照所決定的 block 型態加入元件描述到 ardublock.xml
- 3、撰寫欲加入元件 block 的轉換類別
- 4、將欲加入元件 block 轉換類別設定加入 TranslatorBlockFactory.java 位於 com.ardublock.translator.block

### 三、範例

#### 1. 超音波感測器

i. 這個範例使用 PING)))的超音波感測器使用者給予 pulse 至超音波感測器，超音波感測器會回傳一個型態 long 的回傳值，所以設計上使用 command block 將超音波感測器的回傳值直接放入一個變數中。所以這個 Block 會有兩個需要連接的 Block，一個是指定接角一個是指定存放的變數。如上圖，接角必須是一個數值，指定存放的變數為 int。

ii. 設定 block 的文字標籤於 ardublock.properties



iii. 加入元件描述到 ardublock.xml

在 ardublock.xml 中搜尋 </BlockGenuses> 標籤在這個標籤之前加入以下標籤，這個 Block 主要的目的是讀取超音波感測器所感測的距離值並將其指定給一個變數，所以這個 Block 對定義為一個 command kind 的 Block 較為合適。這個 Block 需要倆的 Blockconnector 參數一個是接到超音波的接角，另一個是所要存放的變數名稱

```
<BlockGenus name="pr_ping_ultrasonic" kind="command" color="255 0 0" initlabel="bg.pr_ping_ultrasonic">
  <description>
    <text>超音波測距器</text></description>
  <BlockConnectors>
    <BlockConnector label="bc.pin_number" connector-kind="socket" connector-type="number">
      <DefaultArg genus-name="number" label="1"></DefaultArg></BlockConnector>
    <BlockConnector label="bc.pr_ping_ultrasonic_range" connector-kind="socket" connector-type="number">
      <DefaultArg genus-name="variable_number" label="range"></DefaultArg></BlockConnector>
  </BlockConnectors>
  <Images>
    <Image block-location="west">
      <FileLocation>com/ardublock/block/playrobot/pr000001.png</FileLocation></Image>
  </Images>
</BlockGenus>
```

上一段完成了 Block 的描述但在還未將新增的 Block 加入 BlockBrowsers 所以 ardublock 是不會顯示出這個 Block 要顯示需在 ardublock.xml 中搜尋</BlockDrawerSet>在這個標籤之前加入以下標識語言

```
<BlockDrawer button-color="255 0 0" type="factory" name="bd.playrobot">
  <BlockGenusMember>pr_axis-x</BlockGenusMember>
  <BlockGenusMember>pr_ping_ultrasonic</BlockGenusMember>
  <BlockGenusMember>pr_2x16_serial_lcd</BlockGenusMember>
  <BlockGenusMember>pr_pir_sensor</BlockGenusMember>
  <BlockGenusMember>pr_pressure_sensor</BlockGenusMember>
  <BlockGenusMember>pr_bending_sensor</BlockGenusMember>
  <BlockGenusMember>pr_sound_impact_sensor</BlockGenusMember>
  <BlockGenusMember>pr_dual-axis_accelerometer</BlockGenusMember>
  <BlockGenusMember>pr_3-axis_compass_module</BlockGenusMember>
  <BlockGenusMember>pr_3-axis_gyroscope_module</BlockGenusMember>
</BlockDrawer>
```

標籤說明：

ardublock 產生 Block 的方式為讀取 ardublock.xml 中的標識語言，標識語言必須有頭有尾這點必須注意。

<BlockGenuses></BlockGenuses>描述 Block 的集合

<BlockGenus></BlockGenus>描述個別 Block 的屬性<BlockGenus>必須有幾個參數必須設定：name 描述這個 Block 的名子，kind 描述這個 Block 的類別，color 描述這個 Block 的顏色，initlabel 描述這個 Block 的顯示標籤。

<BlockConnectors></BlockConnectors>描述 Block 連結狀態的集合。

<BlockConnector></BlockConnector>描述 Block 連結狀態，一個 Block 不一定需要連接其他 Block，但一旦需要連接其它 Block 時必須在<BlockConnectors></BlockConnectors>內描述 BlockConnector 屬性，BlockConnector 有幾個重要的參數必須設定：label 為顯示描述這個

連接處的標籤，connector-kind 描述這個連接的型態，connector-type 描述這個連接的資料型態。

<Images></Images>描述 Block 欲顯示的圖示。

<Image></Image>描述 Block 欲顯示的圖示，block-location 為圖片位置。

<FileLocation></FileLocation>描述 Block 欲顯示的圖示的位置，與顯示的圖片必須放在 \src\main\resources\com\ardublock\block\playrobot 中。

<BlockDrawerSets></BlockDrawerSets>描述欲顯示 Block 的集合

<BlockDrawerSet></BlockDrawerSet>描述欲顯示 Block 的集合

<BlockDrawer></BlockDrawer>描述欲顯示 Block 的集合群組

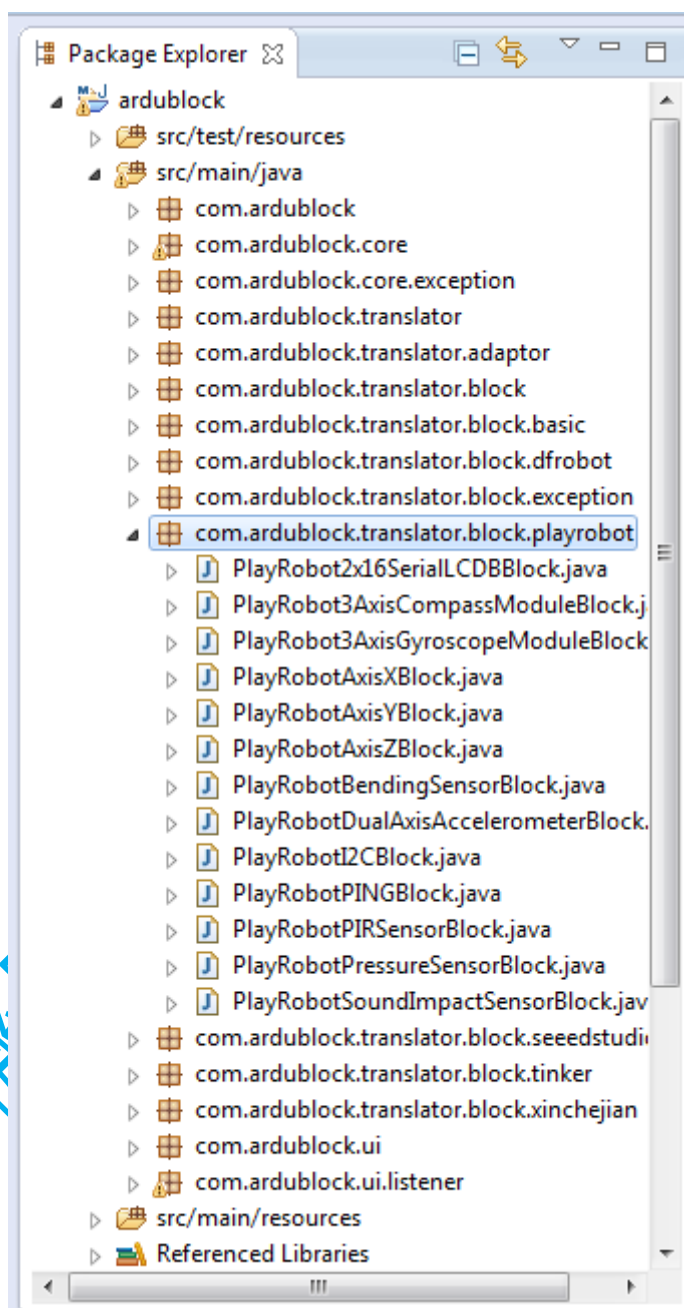
<BlockGenusMember></BlockGenusMember>描述集合群組內的成員

颯機器人 PlayRobot Inc. 普特

iv. 加入元件 block 的轉換類別

在 src/main/java 加入一個 Package 命名為 com.ardublock.translator.block.playrobot，這個 Package 將用來存放所有的轉換類別

在 Package com.ardublock.translator.block.playrobot 加入一個 class 命名為 PlayRobotPINGBlock 並開啟



每一個轉換類別都繼承於 TranslatorBlock 類別，這個類別有一個建構子和一個必要 void 必須宣告，依照下列程式完成。

超音波的 Block 被設計成只要呼叫 ardublockUIPlayRobotSensorPING(int pingPin) 函數並給予接角的數字，ardublockUIPlayRobotSensorPING(int pingPin) 會回傳 int type 距離值，

所以超音波 Block 所要做的工作有取得接角，宣告一個 int 變數，呼叫 ardublockUIPlayRobotSensorPING(int pingPin)。

程式一開始宣告兩個字串變數 SensorPIN 存放接角值 ret 存放回傳值。宣告第一個接到超音波 Block 的其他 Block 判別它是否是 NumberBlock，如果不是給予使用者警告，如果是使用 toCode()取的回傳字串存放到 SensorPIN。宣告第二個接到超音波 Block 的其他 Block 判別它是否是 VariableNumberBlock，如果不是給予使用者警告，如果是使用 toCode()宣告這個變數並將函數名稱加入 ret，使用 translator.addDefinitionCommand(String S) 命令產生 ardublockUIPlayRobotSensorPING(int pingPin)函數，設定回傳字串 ret 為呼叫 ardublockUIPlayRobotSensorPING(int pingPin)函數並將回傳值存入以宣告變數，最後將這個字串回傳到上層 Block。

```
public class PlayRobotPINGBlock extends TranslatorBlock{
    public PlayRobotPINGBlock(Long blockId, Translator translator, String codePrefix, String codeSuffix, String label) {
        super(blockId, translator, codePrefix, codeSuffix, label);
    }
    private final static String PINGSensorFunction = "long ardublockUIPlayRobotSensorPING(int pingPin){\n long cm = 0,duration = 0;\n pinMode(pingPin,
@Override
    public String toCode() throws SocketNullException {
        // TODO Auto-generated method stub
        String SensorPIN = "";
        String ret = "";
        TranslatorBlock mTranslatorBlock = this.getRequiredTranslatorBlockAtSocket(0);
        if(!(mTranslatorBlock instanceof NumberBlock)){
            throw new BlockException(blockId, "Number var must be Number var");
        }else{
            SensorPIN = mTranslatorBlock.toCode();
        }
        mTranslatorBlock = this.getRequiredTranslatorBlockAtSocket(1);
        if(!(mTranslatorBlock instanceof VariableNumberBlock)){
            throw new BlockException(blockId, "VariableNumber var must be VariableNumber var");
        }else{
            ret = mTranslatorBlock.toCode();
        }
        translator.addDefinitionCommand(PINGSensorFunction);
        ret = ret + " = ardublockUIPlayRobotSensorPING( " + SensorPIN + " );\n";
        return codePrefix + ret + codeSuffix;
    }
}
```

程式說明：

PlayRobotPING(Long blockId, Translator translator, String codePrefix, String codeSuffix, String label)為這個 class 的建構子

toCode()是這個 class 的函數，此為必要函數所有轉換工作都在此函數中完成。此函數結束時必須回傳一個值可以是 Null 或一個字串此例為回傳一個字串。

要取得 Block 的程式必須宣告這個 Block，每個 Block 都有自己的轉換類別所以宣告時須使用(TranslatorBlock 變數名稱)，在使用 this.getRequiredTranslatorBlockAtSocket(int c)取得已連接的 Block this.getRequiredTranslatorBlockAtSocket(0)命令指的是取的連接這個 Block 的其他 Block，括號中必須為數值，數值代表每一個連接的 Block 的編號由上到下從零開始。

mTranslatorBlock.toCode()為將已連接的 Block 轉換為程式碼，此函數會回傳一個字串所以必須宣告一個 String 來存放。

translator.addDefinitionCommand(String S)命令可以將字串 S 以函數的方式貼到 Arduino 開發程式中。

ret 為這個 Block 所要回傳的程式字串。

完成後將建構子宣告加入 TranslatorBlockFactory.java ， BlockName 為在 ardublock.xml 中加入<BlockGenus>時所給予的名子

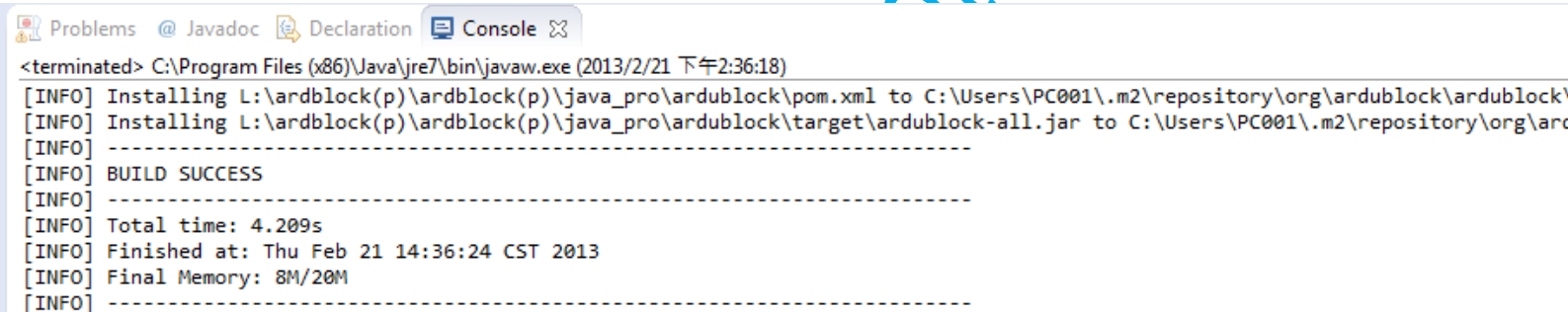
```
//Intelligent Communication and Control Laboratory && PlayRobot  
if(blockName.equals("pr_ping_ultrasonic")){  
    return new PlayRobotPINGBlock(blockId, translator, codePrefix, codeSuffix, label);  
}
```

#### v. 編譯專案

在專案 ardublock 上按滑鼠右鍵=>Run as=>Maven install 既可編譯

在專案目錄中會有一個 target 的資料夾，資料夾中會生成一個 ardublock-all.jar 的檔案，將它複製到 arduino 的 sketchbook location

既可。



```
Problems @ Javadoc Declaration Console  
<terminated> C:\Program Files (x86)\Java\jre7\bin\javaw.exe (2013/2/21 下午2:36:18)  
[INFO] Installing L:\ardblock(p)\ardblock(p)\java_pro\ardublock\pom.xml to C:\Users\PC001\.m2\repository\org\ardublock\ardublock\pom.xml  
[INFO] Installing L:\ardblock(p)\ardblock(p)\java_pro\ardublock\target\ardublock-all.jar to C:\Users\PC001\.m2\repository\org\ardublock\ardublock-all.jar  
[INFO] -----  
[INFO] BUILD SUCCESS  
[INFO] -----  
[INFO] Total time: 4.209s  
[INFO] Finished at: Thu Feb 21 14:36:24 CST 2013  
[INFO] Final Memory: 8M/20M  
[INFO] -----
```

颯機器人