Getting Start with Ardublock

Ardublock is a graphical plug-in for Arduino coding tool. It allows you to build your code in a similar way to Scratch and translate it to Arduino C programming language. When you press the Upload button in Ardublock, the Arduino software will then take over the checking and uploading of your code into the Engduino.

Ardublock Coding

Ardublok is just an alternative interface to Arduino C programming. The Blocks you created get translated to Arduino-C program, checked and upload into Engduino.

The Arduino Checklist

1. Open Arduino-Engduino IDE
2. Plug in Your Engduino and turn the cube ON
   1. Under Tools->Board, choose “Engduino v3”
   2. Under Tools->Port, choose the right serial

Now you can click on Tools->ArduBlock to start the Ardublock plug-in.

You may also need to install the Engduino cube driver the first time you ever run Ardublock. Go on our website www.engduino.org for more information.
Workspace and Toolbar

The Ardublock Toolbar

The toolbar provides you with all the tools you would need when you do your coding with Ardublock.

Categories

Start a new Ardublock sketch
Save your Ardublock Sketch
Open your Ardublock Sketch
Upload your Ardublock Sketch to Engduino
Open a "Serial" connection with your Engduino, look at printout from Engduino and send messages to the device

Coding blocks area

Your position
Your First Blinking Code

1. Click on “Engduino” category to reveal the blocks available.

2. Drag “Set All LEDs” in “Engduino” category and put it in “loop”. You should hear a “click” sound.

3. Drag “delay MILLIS” in “Control” category and put it under “Set All LEDs”. You should hear a “click” sound.

4. Finish the code as in this picture

5. Now press “Upload” to translate your blocks and upload the code to the Engduino. You should see the light blink-
### Some Notes

- The blocks have to be inside a block from the **Code Blocks** category e.g. “loop”, “setup” to be translated to Arduino C. Loose blocks will not be translated.
- There must only be one **Loop** block in your design.
- A socket connector will only accept plug connector of the same type.
- When constructing strings (e.g. for Serial print), you can use the **glue** blocks to combine elements of different types together.

<table>
<thead>
<tr>
<th>Type</th>
<th>Socket</th>
<th>Plug</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Program - Loop</strong></td>
<td><img src="image" alt="Socket" /></td>
<td><img src="image" alt="Plug" /></td>
<td>The blocks inside loop will run forever.</td>
</tr>
<tr>
<td><strong>Instruction/command</strong></td>
<td><img src="image" alt="Socket" /></td>
<td><img src="image" alt="Plug" /></td>
<td>The instruction block goes inside the program or loop block. It has a little dent at the top and a little nob at the bottom.</td>
</tr>
<tr>
<td><strong>Number</strong></td>
<td><img src="image" alt="Socket" /></td>
<td><img src="image" alt="Plug" /></td>
<td>The socket block has an inverted triangle shape at the right side of the block to take in a plug block with a triangular shape to the left of the block.</td>
</tr>
<tr>
<td><strong>String</strong></td>
<td><img src="image" alt="Socket" /></td>
<td><img src="image" alt="Plug" /></td>
<td>The socket block has an inverted square shape at the right side of the block to take in a plug block with a square shape to the left of the block.</td>
</tr>
<tr>
<td><strong>Boolean/Logic</strong></td>
<td><img src="image" alt="Socket" /></td>
<td><img src="image" alt="Plug" /></td>
<td>The socket has an inverted curve shape at the right side of the block to take in a plug block with a curve shape to the left of the block.</td>
</tr>
</tbody>
</table>