A start block has buttons to press force a reset or force attempting to run the code and has LED's to indicate its status. It connects to the bottom most left most of the other blocks.

Each programing block connects to its neighbors to form lines of code read from left to right, top to bottom. On each programing block a turnable wheel allows the user to select the meaning of the block as shown in the “this block” window. Hints for what blocks ought to be placed next to the block appear in the “neighbors” windows. An LED shows if the block is the reason for the program not being able to run.

Output blocks have some way of acting on the world (ie a light that blinks). The programing to control an output is connected to the output block, and a hint for what that programing ought to be is shown.

In any scheme designed to be easy to use there is the danger of being easy only for the designers. What we think of as intuitive is not certain to be easy to understand for students or teachers.

The more complex our manuals or instructions are the clearer it can be how to use the blocks, but there is a limit to how much a user is willing to learn before using the blocks. If there is not enough user targeted documentation learning to use the blocks could be frustrating.

The solution to both of those dangers appears to be testing, which invites a tradeoff of time and resources spent on testing versus spent on designs and implementations.