primary occupation is not programming (artists, architects, teachers, composers, data scientists, etc.). Furthermore, there are plug-ins for NetBeans IDE and IntelliJ IDE to provide useful and real-time assist for Processing 4 program development.

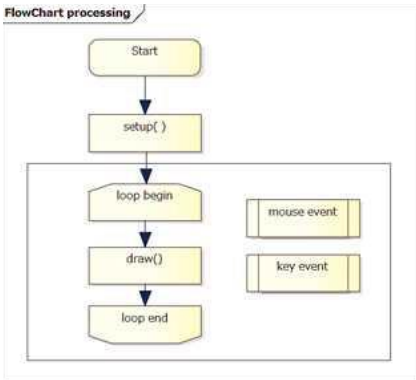

Figure 2. Fundamental program flow in Processing 4

Table 1. Primary function categories for Processing

Category	Primary Contents
3D	3D Graphics
Animation	Animations
Compilation	Packaging of various functions
Data	Data Communications
GUI	Graphical User Interface
Geometry	Generation of 2-D and 3-D geometric figures
Hardware	Interfacing with sensors and other hardware
I/O	Data input/output
Language	Natural Language Processing
Math	Mathematics
Simulation	Physics simulation and flocking simulation
Sound	Sound reproduction, sound generation and analysis
Typography	Typography
Utilities	Utilities
Video & Vison	Video and Computer Vision
Other	Miscellaneous

3. Comparison of Java and Processing 4 Features by Example

3.1 Visual illusion program development

In the visual illusion, we take the Münsterberg illusion as one example of a basic, interactive program. The Münsterberg illusion was introduced by the physician and psychologist Munsterberg in 1897. When squares of white and black, etc., are shifted to each other above and below a straight line, the straight line at the boundary of the squares appears to tilt. Interactivity was incorporated by using the mouse event function to shift the horizontal position of the entire row of squares by dragging the mouse over the row of squares. Figure 3 shows the screen in Processing 4. To compare the program volume, I compared the number of program lines after reformatting on the NetBeans IDE and the Processing 4 IDE, with all comment lines removed. The Java program had 113 lines, while the Processing 4 program had 79 lines, indicating that the exact same functionality can be described more concisely in Processing 4.

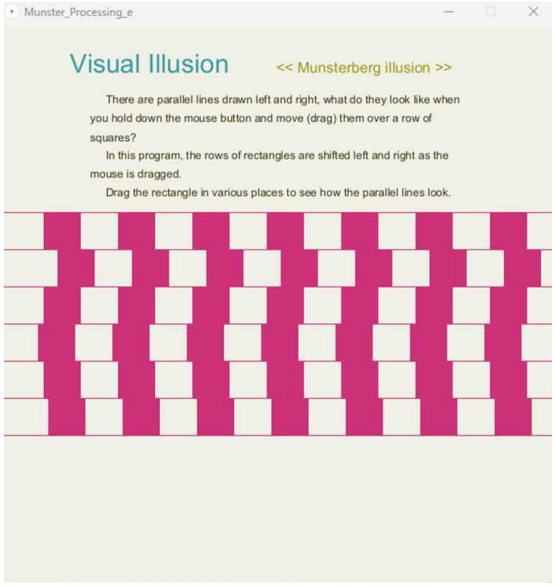

Figure 3. Munsterberg visual illusion by Processing 4

3.2. Hooked on Java examples [3]

3.2.1. "Neon sign" example. This example consists of a simple program structure that switches images at random times between the image of a glowing Neon tube and the image of a tuned-off Neon tube to represent a blinking neon sign (see Figure 4).

Figure 4. "Neon Sign" example

The length of the source code was 52 lines in the Java version, but the length of rewritten code in Processing 4 was 42 lines. The time required for porting from Java to Processing was about 30 minutes.

3.2.2. "Electric circuit" example. This example is to learn the relationship between current, voltage, and resistance. When the appropriate battery and resistor are selected and the switch is turned on, the light bulb lights up; if the current is too high, the bulb breaks and makes a noise; if the current is too low, the bulb does not light up (see Figure 5). The length of the source code was 613 lines in the Java version, but the rewritten code in Processing was 466 lines.