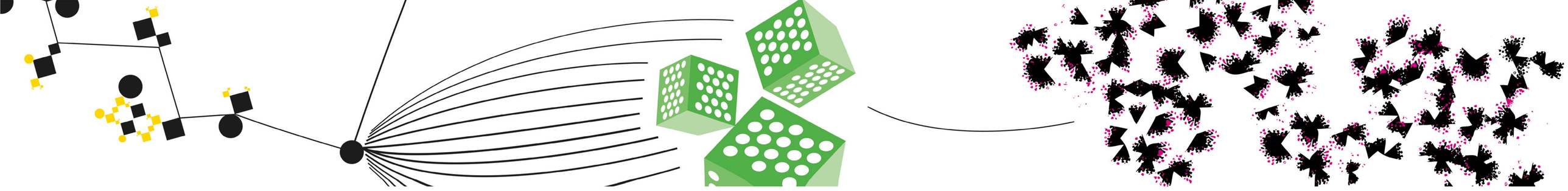


The Smell of Processing*

Remco de Man and Ansgar Fehnker**
University of Twente

Processing

- Processing is a language and an IDE.
- A language for interactive animations.
- A language for teaching programming.
- The language is a subset of Java.
  - With built-in methods for graphics, user I/O, audio, ...
  - Omits access modifiers, or static or final modifiers.
  - Usually simple inheritance, if at all.
- A vibrant community of all stripes.
  - With “liberal” programming standards.

Design Smells

- Not bugs, but symptoms that possibly indicate a deeper problem.
- Make it difficult to extend, modify or maintain software.
- Indicate a violation of fundamental design principles.

Novices

- Novices can write code that solves a given problem. Sort of.
- But as soon as they have to build bigger applications, they face unnecessary complexities of their own making.
- They find it difficult to
  - understand,
  - maintain, or
  - extend their code.

Consequences of poor design

- Poor encapsulation
- No separation of concerns
- High coupling
- Low cohesion

Questions

Especially novices

- What design smells apply to Processing code?
- To what extend do they occur in Processing code?
- Can you find them automatically?

Processing Specific Smells

New Smells!

- Pixel hardcode ignorance
  - ➢ No abstraction for positioning of graphical elements.
- Jack-in-the-box event handling
  - ➢ Use of global event variables outside of event handlers.
- Drawing state change
  - ➢ Using drawing method for updating states.

Decentralized Drawing

- ➢ Drawing in methods not called by the main draw method.

Customized OO Smells

Made to work with Processing

- Stateless class
- Long method
- Long Parameter list
- God class

Implementation

- Transform to Java code, run PMD, and filter warnings.
- Extended and modified PMD to handle Processing
  - Added new PMD checks.
  - Implemented call stack algorithms.
  - Adapted existing checks.
  - Uses metrics for God class smell.
  - Adapted to deal with inner classes vs. outer class.

Sources

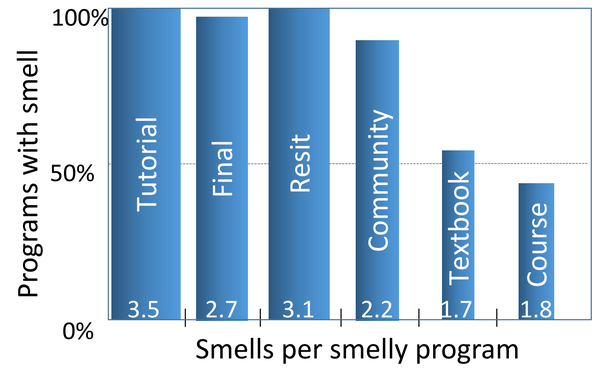
- 61 student programs, last week of the tutorials.
- 79 student programs, final project.
- 178 community programs (`openprocessing.org`).
- 17 student programs, final project, resit.
- 149 textbook examples (`learningprocessing.com`).
- 32 examples from course lectures and manuals.

manual analysis / analysis with PMD

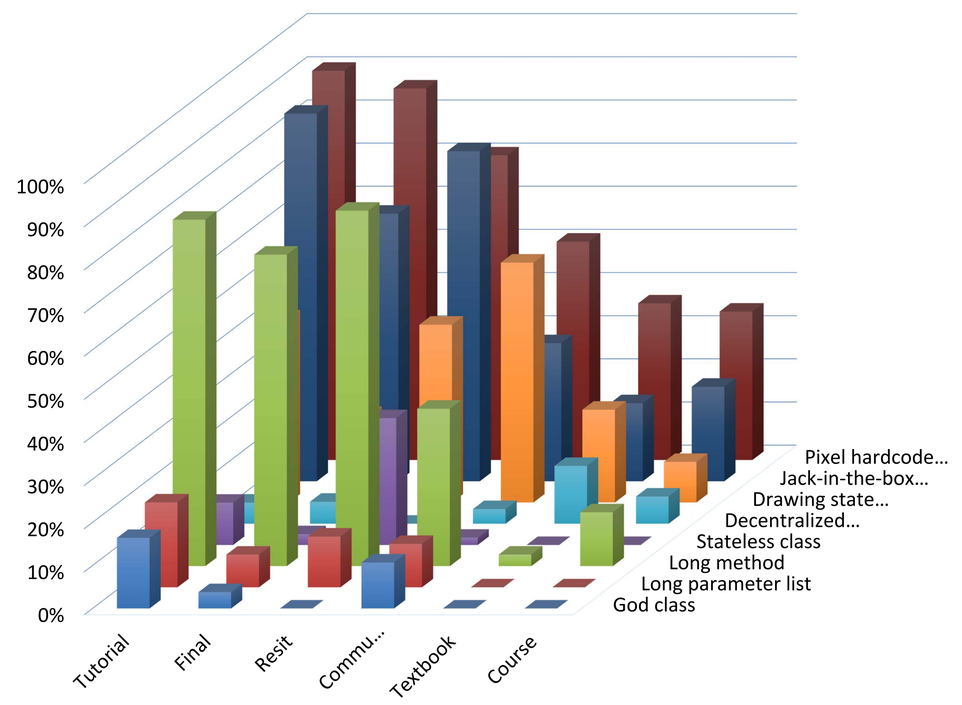
Results

- Almost all student code contains smells.
- Almost all community code contains smells. Fewer per program.
- A surprising number of textbook examples contain smells.

Fraction of smelly programs by source



Smells By Source



Future Work

- Define more smells.
- Define practical refactorings.
- Integrate in course.
- Improve teaching material.

* *Published at 10th International Conference on Computer Supported Education, March 2018.*
** *Email: ansgar.fehnker@utwente.nl*

Formal Methods & Tools

UNIVERSITY OF TWENTE.