Clean Code

Code Smell Summary

Smells and Heuristics

- Code Comments
    - Inappropriate Information
    - Obsolete
    - Redundant
    - Poorly Written
    - Commented-Out Code
- Environment
    - Build Requires More Than One Step
    - Tests Require More Than One Step
- Functions
    - Too Many Arguments
    - Output Arguments
    - Flag Arguments (booleans)
    - Dead Functions (unused code)
- General
    - Multiple Languages in One Source File
    - Obvious Behavior Is Unimplemented
    - Incorrect Behavior at the Boundaries
    - Overridden Safeties (i.e. overriding serialVersionUID in Java)
    - Duplication
    - Code at Wrong Level of Abstraction
    - Base Classes Depending on Their Derivatives
    - Too Much Information
    - Dead Code
    - Vertical Separation
    - Inconsistency
    - Clutter

Smells and Heuristics

- Artificial Coupling
- Feature Envy (classes should be interested in what they have rather than other classes)
- Selector Arguments
- Obscured Intent
- Misplaced Responsibility
- Inappropriate Static
- Use Explanatory Variables
- Function Names Should Say What They Do
- Understand the Algorithm
- Make Logical Dependencies Physical
- Prefer Polymorphism to If/ Else or Switch/ Case
- Follow Standard Conventions
- Replace Magic Numbers with Named Constants
- Be Precise
- Structure over Convention
- Encapsulate Conditionals
- Avoid Negative Conditionals
- Functions Should Do One Thing
- Hidden Temporal Couplings
- Don't Be Arbitrary
- Encapsulate Boundary Conditions
- Functions Should Descend Only One Level of Abstraction
- Keep Configurable Data at High Levels
- Avoid Transitive Navigation

- Java
  - Avoid Long Import Lists by Using Wildcards
  - Don't Inherit Constant

Smells and Heuristics

    - Constants versus Enums (don’t use enums)
- Names
    - Choose Descriptive Names
    - Choose Names at the Appropriate Level of Abstraction
    - Use Standard Nomenclature Where Possible
    - Unambiguous Names
    - Use Long Names for Long Scopes
    - Avoid Encodings (prefixes such as m_)
    - Names Should Describe Side-Effects Tests
    - Insufficient Tests
    - Use a Coverage Tool!
    - Don’t Skip Trivial Tests
    - An Ignored Test Is a Question about an Ambiguity
    - Test Boundary Conditions
    - Exhaustively Test Near Bugs
    - Patterns of Failure Are Revealing
    - Test Coverage Patterns Can Be Revealing
    - Tests Should Be Fast