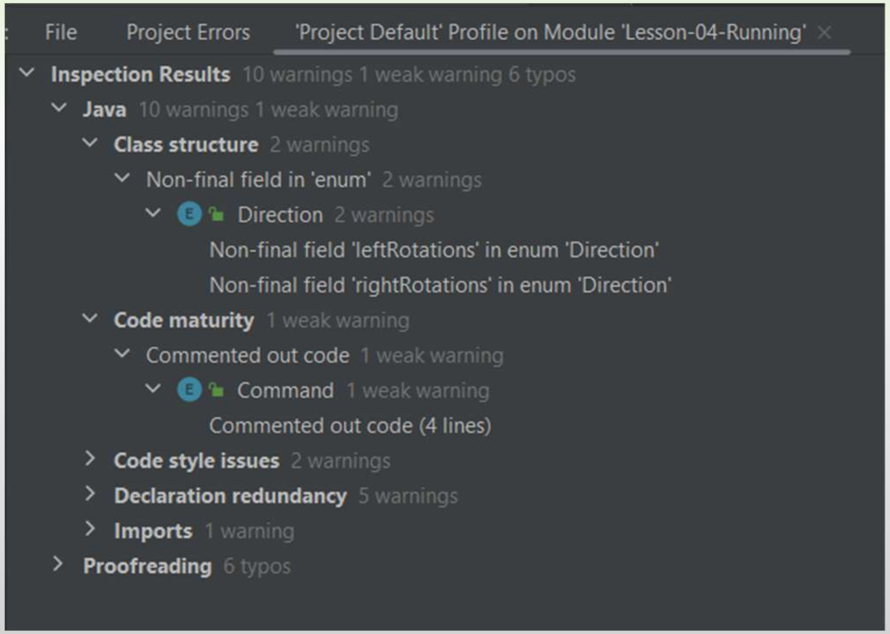
DETECTING CODE SMELLS

Can I use some tools?

INTELLIJ TOOLS

Inspection (Code/Inspect code)

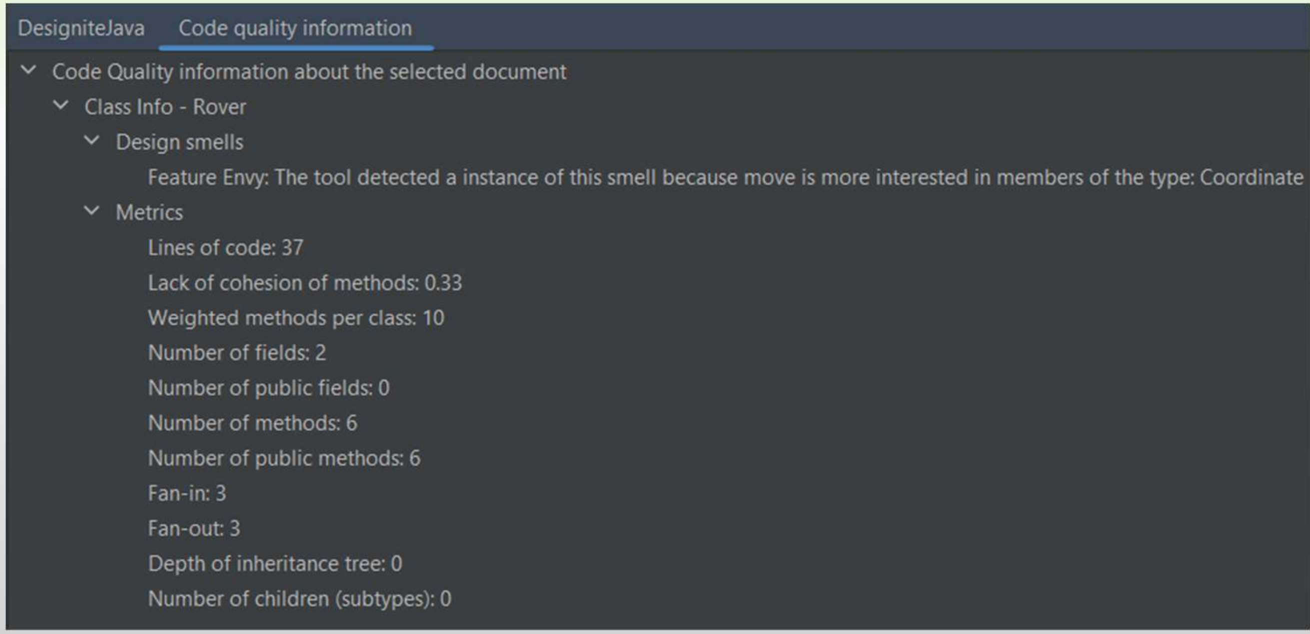
- Built-in



```
Inspection Results 10 warnings 1 weak warning 6 typos
  Java 10 warnings 1 weak warning
    Class structure 2 warnings
      Non-final field in 'enum' 2 warnings
        Direction 2 warnings
          Non-final field 'leftRotations' in enum 'Direction'
          Non-final field 'rightRotations' in enum 'Direction'
    Code maturity 1 weak warning
      Commented out code 1 weak warning
        Command 1 weak warning
          Commented out code (4 lines)
    Code style issues 2 warnings
    Declaration redundancy 5 warnings
    Imports 1 warning
  Proofreading 6 typos
```

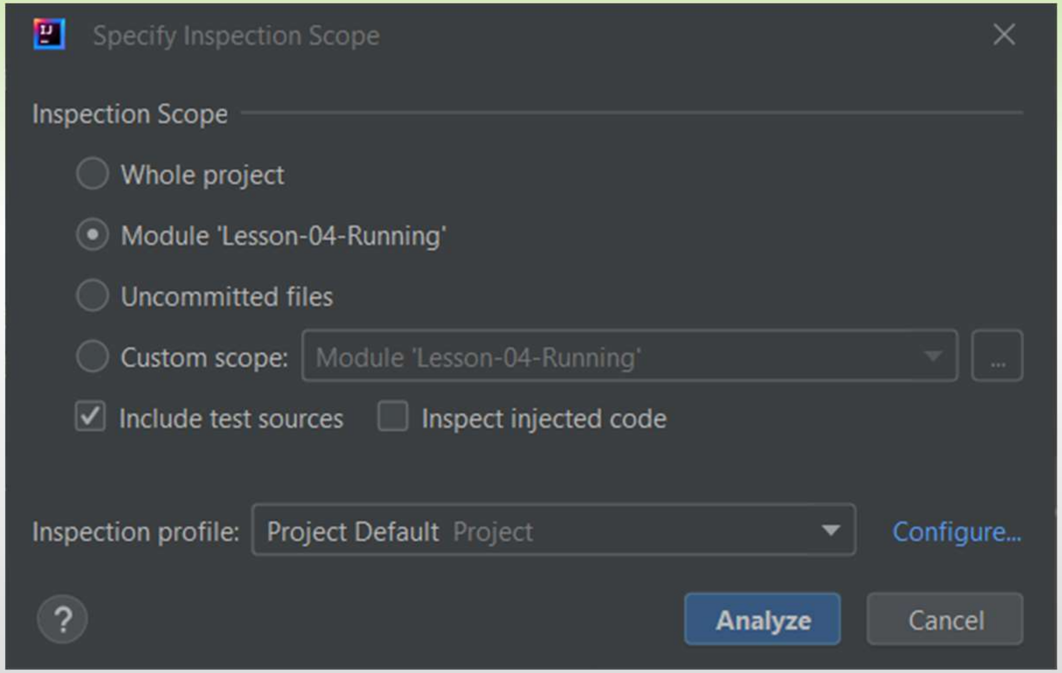
DesigniteJava

- Plugin



```
Code Quality information about the selected document
  Class Info - Rover
    Design smells
      Feature Envy: The tool detected a instance of this smell because move is more interested in members of the type: Coordinate
    Metrics
      Lines of code: 37
      Lack of cohesion of methods: 0.33
      Weighted methods per class: 10
      Number of fields: 2
      Number of public fields: 0
      Number of methods: 6
      Number of public methods: 6
      Fan-in: 3
      Fan-out: 3
      Depth of inheritance tree: 0
      Number of children (subtypes): 0
```

INSPECTION



Specify Inspection Scope

Inspection Scope

○ Whole project

● Module 'Lesson-04-Running'

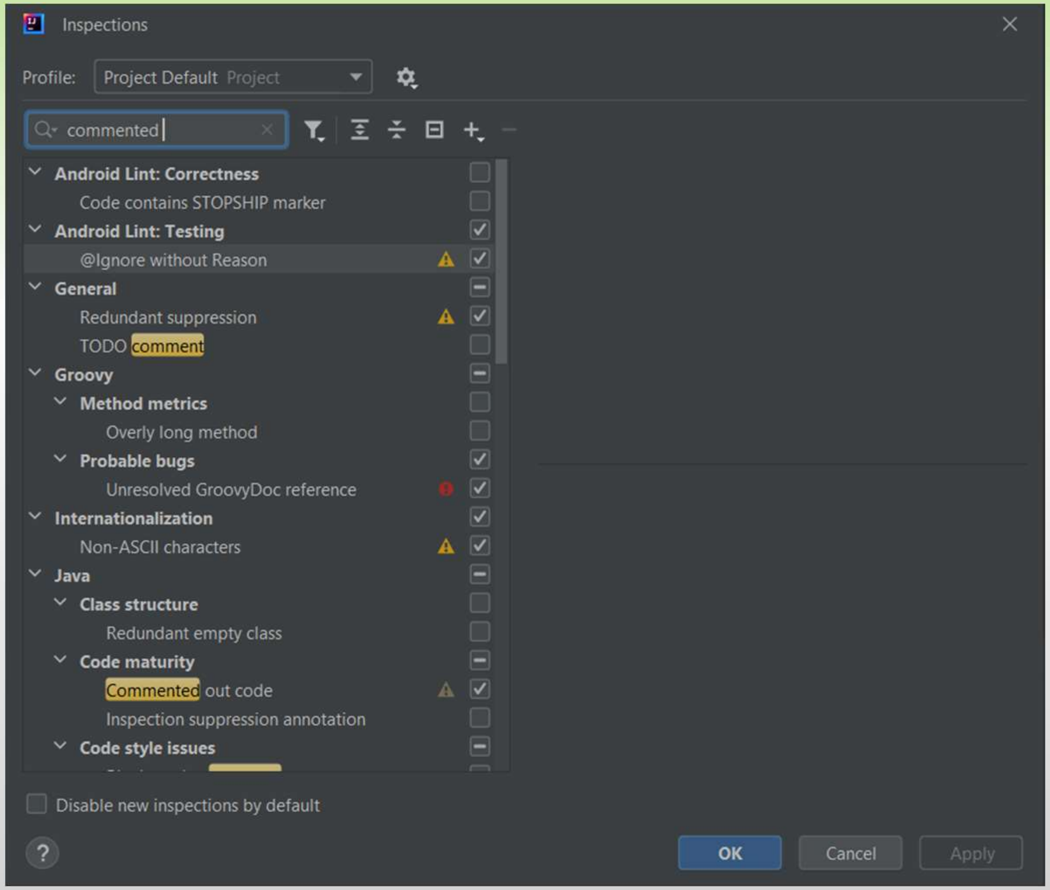
○ Uncommitted files

○ Custom scope: Module 'Lesson-04-Running'

☑ Include test sources ☐ Inspect injected code

Inspection profile: Project Default Project

INSPECTION



Inspections

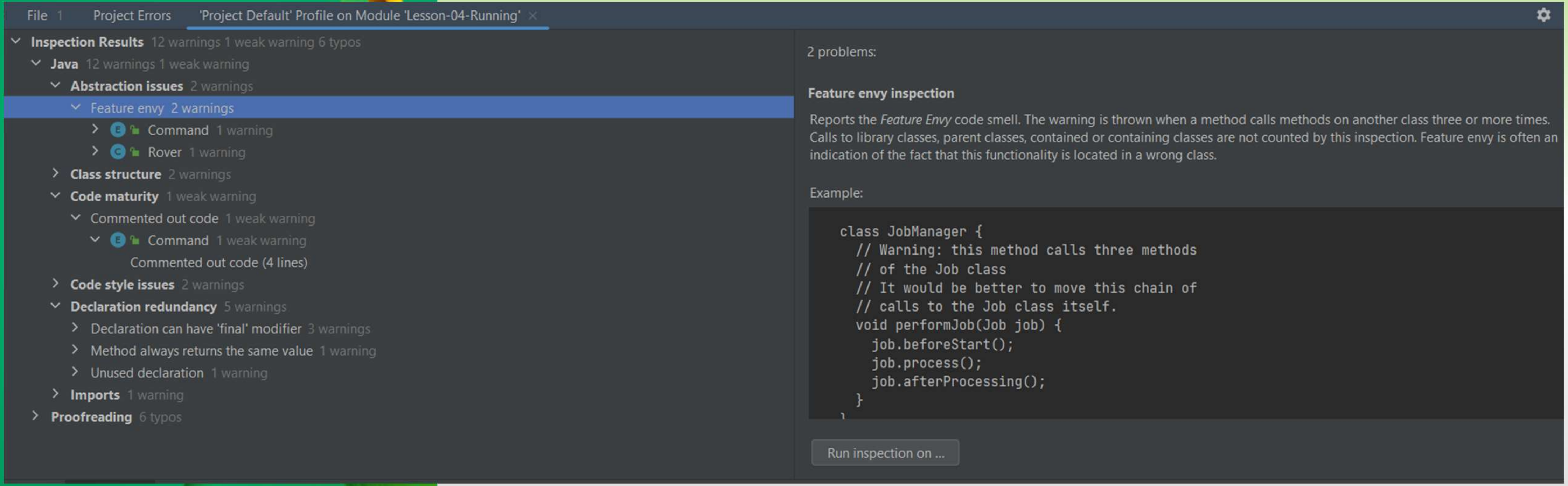
Profile: Project Default Project

commented

- Android Lint: Correctness □
  - Code contains STOPSHIP marker □
- Android Lint: Testing ☑
  - @Ignore without Reason ☑
- General ⊟
  - Redundant suppression ☑
  - TODO comment □
- Groovy ⊟
  - Method metrics □
    - Overly long method □
  - Probable bugs ☑
    - Unresolved GroovyDoc reference ☑
- Internationalization ☑
  - Non-ASCII characters ☑
- Java ⊟
  - Class structure □
    - Redundant empty class □
  - Code maturity ⊟
    - Commented out code ☑
    - Inspection suppression annotation □
  - Code style issues ⊟

□ Disable new inspections by default

INSPECTION



File 1 | Project Errors | 'Project Default' Profile on Module 'Lesson-04-Running' ×

- **Inspection Results** 12 warnings 1 weak warning 6 typos
  - **Java** 12 warnings 1 weak warning
    - **Abstraction issues** 2 warnings
      - Feature envy 2 warnings
        - E Command 1 warning
        - C Rover 1 warning
    - **Class structure** 2 warnings
    - **Code maturity** 1 weak warning
      - Commented out code 1 weak warning
        - E Command 1 weak warning
          - Commented out code (4 lines)
    - **Code style issues** 2 warnings
    - **Declaration redundancy** 5 warnings
      - Declaration can have 'final' modifier 3 warnings
      - Method always returns the same value 1 warning
      - Unused declaration 1 warning
    - **Imports** 1 warning
  - **Proofreading** 6 typos

2 problems:

Feature envy inspection

Reports the *Feature Envy* code smell. The warning is thrown when a method calls methods on another class three or more times. Calls to library classes, parent classes, contained or containing classes are not counted by this inspection. Feature envy is often an indication of the fact that this functionality is located in a wrong class.

Example:

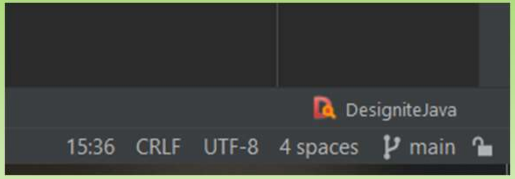
```
class JobManager {
  // Warning: this method calls three methods
  // of the Job class
  // It would be better to move this chain of
  // calls to the Job class itself.
  void performJob(Job job) {
    job.beforeStart();
    job.process();
    job.afterProcessing();
  }
}
```

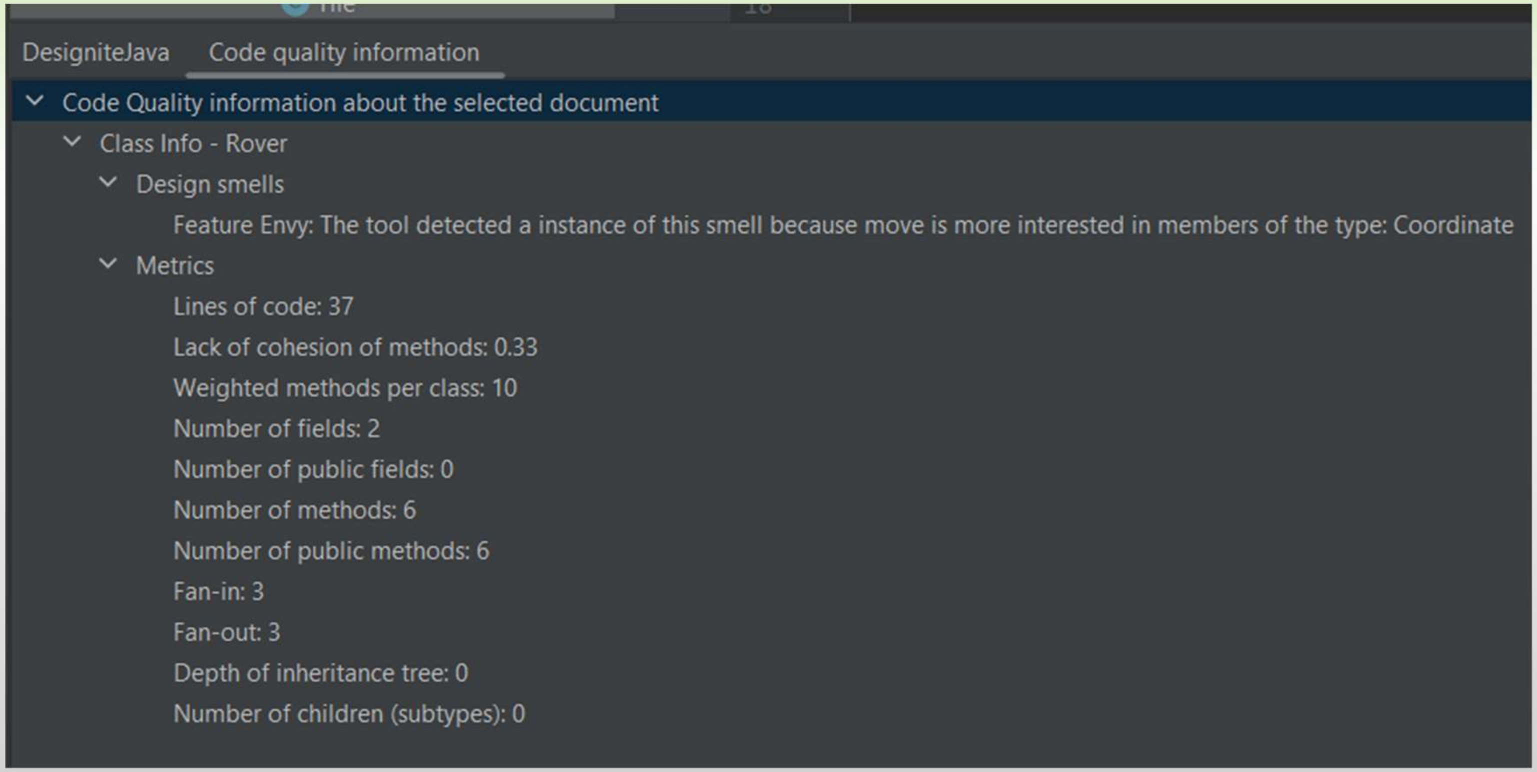
Run inspection on ...

INSPECTION

- switch statement
- duplicated code
- long method
- feature envy
- commented out code

DESIGNITEJAVA





```
Code Quality information about the selected document
  Class Info - Rover
    Design smells
      Feature Envy: The tool detected a instance of this smell because move is more interested in members of the type: Coordinate
    Metrics
      Lines of code: 37
      Lack of cohesion of methods: 0.33
      Weighted methods per class: 10
      Number of fields: 2
      Number of public fields: 0
      Number of methods: 6
      Number of public methods: 6
      Fan-in: 3
      Fan-out: 3
      Depth of inheritance tree: 0
      Number of children (subtypes): 0
```

DESIGNITEJAVA

- Architecture smells
  - cyclic dependency
  - god component

- Design smells
  - feature envy
  - hierarchy smells

INSPECTION VS DESIGNITEJAVA

Inspection

- configurable
- rerun after new code
- no metrics
- explanation

DesigniteJava

- not configurable
- „on the run” analysis
- Metrics
- explanation

Inspection > DesigniteJava

REFERENCES

- https://www.designite-tools.com/features/#implSmells

QUESTIONS?

THANK YOU FOR YOUR ATTENTION!

Ildikó Takács

Ildiko.takacs@css.ch