Warszawa, 27 maja 2020

ZMWO – Sprawozdanie – AspectJ

Autor: **Adam Jędrzejowski**
Wybrane zadanie: **Repozytorium**

Do testowania danego aspektu, polecam wyłączyć inne, np. przez za komentowanie ich kodu.
W celu łatwiejszej nawigacji i projektowaniu aspektów podzieliłem kod na dwa pakiety:

- `pl.edu.pw.zmwo.aspect` – kody aspektów
- `pl.edu.pw.zmwo.repo` – oryginalny kod aplikacji

Zadanie 1

Punkt przecięcia to `everyMethod()`, który przecina każde wywołanie funkcji w pakiecie z oryginalnym kodem repozytorium. Każdemu przecięciu nadawany jest unikalny identyfikator w celu łatwiejszej identyfikacji. Dodatkowo czas wykonywania funkcji jest mierzony w nanosekundach, ponieważ najszybsze trwają w przedziałach 0,0005 milisekund. Wypisywanie informacji odbywa się za pomocą funkcji `log()`, która zapisuje tekst do pliku.

Kod aspektu:

```
package pl.edu.pw.zmwo.aspect;

import java.io.FileWriter;
import java.io.IOException;

public aspect Zadanie1 {

    static String EOL = System.getProperty("line.separator");
    private FileWriter log_file = null;
    private static long id = 0;

    pointcut everyMethod(): execution(public * *(..)) &&
            within(pl.edu.pw.zmwo.repo.*);

    void log(long id, String format, Object... args) {

        String line = String.format("[%s] ", id) + String.format(format, args);

        try {
            if (log_file == null) {
                log_file = new FileWriter("zadanie1_log.log");
            }

            log_file.write(line + EOL);
            log_file.flush();
        } catch (IOException exception) {
            System.err.println(line);
        }
    }

    Object around(): everyMethod() {
        final long my_id = ++id;

        log(my_id, "what = %s", thisJoinPoint.toLongString());
        log(my_id, "where = %s", thisJoinPointStaticPart.getSourceLocation());

        for (Object arg : thisJoinPoint.getArgs()) {
            log(my_id, "arg = %s", arg.getClass());
        }

        long start_time = System.nanoTime();
        Object return_value = proceed();
        long end_time = System.nanoTime();

        log(my_id, "execution time = %s [ns]", end_time - start_time);
        log(my_id, "return type = %s",
                return_value != null ? return_value.getClass() : "null");

        return return_value;
    }
}
```

Wynik w postaci zawartości pliku zadanie1_log.log:

```
[1] what = execution(public static void pl.edu.pw.zmwo.repo.Main.main(java.lang.String[]))
[1] where = Main.java:5
[1] arg = class [Ljava.lang.String;
[2] what = execution(public void pl.edu.pw.zmwo.repo.Project.addTask(pl.edu.pw.zmwo.repo.Task))
[2] where = Project.java:35
[2] arg = class pl.edu.pw.zmwo.repo.Task
[3] what = execution(public java.lang.Integer pl.edu.pw.zmwo.repo.Task.getId())
[3] where = Task.java:48
[3] execution time = 1623 [ns]
[3] return type = class java.lang.Integer
[2] execution time = 518937 [ns]
[2] return type = null
[4] what = execution(public void pl.edu.pw.zmwo.repo.Project.addTask(pl.edu.pw.zmwo.repo.Task))
[4] where = Project.java:35
[4] arg = class pl.edu.pw.zmwo.repo.Task
[5] what = execution(public java.lang.Integer pl.edu.pw.zmwo.repo.Task.getId())
[5] where = Task.java:48
[5] execution time = 432 [ns]
[5] return type = class java.lang.Integer
[4] execution time = 495800 [ns]
[4] return type = null
[6] what = execution(public void pl.edu.pw.zmwo.repo.Project.addTask(pl.edu.pw.zmwo.repo.Task))
[6] where = Project.java:35
[6] arg = class pl.edu.pw.zmwo.repo.Task
[7] what = execution(public java.lang.Integer pl.edu.pw.zmwo.repo.Task.getId())
[7] where = Task.java:48
[7] execution time = 397 [ns]
[7] return type = class java.lang.Integer
[6] execution time = 435882 [ns]
[6] return type = null
[8] what = execution(public void pl.edu.pw.zmwo.repo.Repository.addProject(pl.edu.pw.zmwo.repo.Project))
[8] where = Repository.java:27
[8] arg = class pl.edu.pw.zmwo.repo.Project
[9] what = execution(public java.lang.String pl.edu.pw.zmwo.repo.Project.getName())
[9] where = Project.java:22
[9] execution time = 2566 [ns]
[9] return type = class java.lang.String
[8] execution time = 410422 [ns]
[8] return type = null
[10] what = execution(public void pl.edu.pw.zmwo.repo.ProjectExporter.export(pl.edu.pw.zmwo.repo.Project,
java.io.PrintStream))
[10] where = ProjectExporter.java:7
[10] arg = class pl.edu.pw.zmwo.repo.Project
[10] arg = class java.io.PrintStream
[11] what = execution(public java.lang.String pl.edu.pw.zmwo.repo.Project.getName())
[11] where = Project.java:22
[11] execution time = 363 [ns]
[11] return type = class java.lang.String
[12] what = execution(public java.util.List pl.edu.pw.zmwo.repo.Project.getTasks())
[12] where = Project.java:26
[12] execution time = 80604 [ns]
[12] return type = class java.util.LinkedList
[13] what = execution(public java.lang.Integer pl.edu.pw.zmwo.repo.Task.getId())
[13] where = Task.java:48
[13] execution time = 386 [ns]
[13] return type = class java.lang.Integer
[14] what = execution(public java.lang.String pl.edu.pw.zmwo.repo.Task.getDescription())
[14] where = Task.java:40
[14] execution time = 2347 [ns]
[14] return type = class java.lang.String
[15] what = execution(public pl.edu.pw.zmwo.repo.Employee pl.edu.pw.zmwo.repo.Task.getReporter())
[15] where = Task.java:32
[15] execution time = 2668 [ns]
[15] return type = class pl.edu.pw.zmwo.repo.Employee
[16] what = execution(public java.lang.String pl.edu.pw.zmwo.repo.Employee.getName())
[16] where = Employee.java:16
[16] execution time = 1585 [ns]
[16] return type = class java.lang.String
[17] what = execution(public pl.edu.pw.zmwo.repo.Employee pl.edu.pw.zmwo.repo.Task.getReporter())
[17] where = Task.java:32
[17] execution time = 316 [ns]
[17] return type = class pl.edu.pw.zmwo.repo.Employee
[18] what = execution(public java.lang.String pl.edu.pw.zmwo.repo.Employee.getLastName())
[18] where = Employee.java:20
[18] execution time = 2222 [ns]
[18] return type = class java.lang.String
[19] what = execution(public pl.edu.pw.zmwo.repo.Employee pl.edu.pw.zmwo.repo.Task.getAssignee())
[19] where = Task.java:24
[19] execution time = 2661 [ns]
[19] return type = class pl.edu.pw.zmwo.repo.Employee
[20] what = execution(public java.lang.String pl.edu.pw.zmwo.repo.Employee.getName())
```

```
[20] where = Employee.java:16
[20] execution time = 414 [ns]
[20] return type = class java.lang.String
[21] what = execution(public pl.edu.pw.zmwo.repo.Employee pl.edu.pw.zmwo.repo.Task.getAssignee())
[21] where = Task.java:24
[21] execution time = 417 [ns]
[21] return type = class pl.edu.pw.zmwo.repo.Employee
[22] what = execution(public java.lang.String pl.edu.pw.zmwo.repo.Employee.getLastName())
[22] where = Employee.java:20
[22] execution time = 277 [ns]
[22] return type = class java.lang.String
[23] what = execution(public pl.edu.pw.zmwo.repo.Task.Status pl.edu.pw.zmwo.repo.Task.getStatus())
[23] where = Task.java:52
[23] execution time = 1669 [ns]
[23] return type = class pl.edu.pw.zmwo.repo.Task$Status
[24] what = execution(public java.lang.Integer pl.edu.pw.zmwo.repo.Task.getId())
[24] where = Task.java:48
[24] execution time = 207 [ns]
[24] return type = class java.lang.Integer
[25] what = execution(public java.lang.String pl.edu.pw.zmwo.repo.Task.getDescription())
[25] where = Task.java:40
[25] execution time = 273 [ns]
[25] return type = class java.lang.String
[26] what = execution(public pl.edu.pw.zmwo.repo.Employee pl.edu.pw.zmwo.repo.Task.getReporter())
[26] where = Task.java:32
[26] execution time = 209 [ns]
[26] return type = class pl.edu.pw.zmwo.repo.Employee
[27] what = execution(public java.lang.String pl.edu.pw.zmwo.repo.Employee.getName())
[27] where = Employee.java:16
[27] execution time = 271 [ns]
[27] return type = class java.lang.String
[28] what = execution(public pl.edu.pw.zmwo.repo.Employee pl.edu.pw.zmwo.repo.Task.getReporter())
[28] where = Task.java:32
[28] execution time = 240 [ns]
[28] return type = class pl.edu.pw.zmwo.repo.Employee
[29] what = execution(public java.lang.String pl.edu.pw.zmwo.repo.Employee.getLastName())
[29] where = Employee.java:20
[29] execution time = 246 [ns]
[29] return type = class java.lang.String
[30] what = execution(public pl.edu.pw.zmwo.repo.Employee pl.edu.pw.zmwo.repo.Task.getAssignee())
[30] where = Task.java:24
[30] execution time = 257 [ns]
[30] return type = class pl.edu.pw.zmwo.repo.Employee
[31] what = execution(public java.lang.String pl.edu.pw.zmwo.repo.Employee.getName())
[31] where = Employee.java:16
[31] execution time = 241 [ns]
[31] return type = class java.lang.String
[32] what = execution(public pl.edu.pw.zmwo.repo.Employee pl.edu.pw.zmwo.repo.Task.getAssignee())
[32] where = Task.java:24
[32] execution time = 249 [ns]
[32] return type = class pl.edu.pw.zmwo.repo.Employee
[33] what = execution(public java.lang.String pl.edu.pw.zmwo.repo.Employee.getLastName())
[33] where = Employee.java:20
[33] execution time = 299 [ns]
[33] return type = class java.lang.String
[34] what = execution(public pl.edu.pw.zmwo.repo.Task.Status pl.edu.pw.zmwo.repo.Task.getStatus())
[34] where = Task.java:52
[34] execution time = 684 [ns]
[34] return type = class pl.edu.pw.zmwo.repo.Task$Status
[35] what = execution(public java.util.List pl.edu.pw.zmwo.repo.Project.getDocuments())
[35] where = Project.java:47
[35] execution time = 28268 [ns]
[35] return type = class java.util.LinkedList
[10] execution time = 8420408 [ns]
[10] return type = null
[1] execution time = 16572957 [ns]
[1] return type = null
```

Zadanie 2

Do serializacji obiektu repozytorium, użyłem biblioteki Gson do obsługi formatu json. Repozytorium jest zapisywane po każdym wywołaniu `Repository.addProject`, `Repository.deleteProject`, `Repository.updateProject`. Wczytywanie jest dokonywane jako zastąpienie konstruktora.

Kod aspektu:

```
package pl.edu.pw.zmwo.aspect;

import com.google.gson.Gson;
import pl.edu.pw.zmwo.repo.Repository;

import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

public aspect Zadanie2 {

    static String error_msg = "Zadanie2: Błąd wczytywania repozytorium";
    Gson gson = new Gson();

    void saveRepo(Object repo) {
        saveRepo((Repository) repo);
    }

    Repository loadRepo() {
        try {
            Repository repo = gson.fromJson(
                    new FileReader("repository.json"),
                    Repository.class
            );

            if (repo == null) {
                System.err.println(error_msg);
                return new Repository();
            }

            return repo;
        } catch (IOException e) {
            System.err.println(error_msg);
            return new Repository();
        }
    }

    void saveRepo(Repository repo) {
        try {
            FileWriter writer = new FileWriter("repository.json");
            gson.toJson(repo, writer);
            writer.flush();
            writer.close();
        } catch (IOException e) {
            System.err.println("Błąd zapisywania");
        }
    }

    Repository around(): call(pl.edu.pw.zmwo.repo.Repository+.new())
            && within(pl.edu.pw.zmwo.repo.*) {
        return loadRepo();
    }

    after(): execution(* pl.edu.pw.zmwo.repo.Repository.addProject(..)) ||
            execution(* pl.edu.pw.zmwo.repo.Repository.deleteProject(..)) ||
            execution(* pl.edu.pw.zmwo.repo.Repository.updateProject(..)) {
        saveRepo(thisJoinPoint.getThis());
    }
}
```

Treść pliku `repostory.json`:


```
{
  "projects": {
    "project1": {
      "name": "project1",
      "tasks": {
        "1": {
          "id": 1,
          "description": "task1",
          "assignee": {
            "name": "user1",
            "lastName": "user1"
          },
          "reporter": {
            "name": "user1",
            "lastName": "user1"
          },
          "status": "NEW"
        },
        "2": {
          "id": 2,
          "description": "task2",
          "assignee": {
            "name": "user2",
            "lastName": "user2"
          },
          "reporter": {
            "name": "user1",
            "lastName": "user1"
          },
          "status": "NEW"
        }
      },
      "documents": {}
    }
  }
}
```

Zadanie 3

Aspekt dodaje dwa nowe pola do klasy Project, które służą jako pamięć podręczna. Przy każdym wywołaniu metody dodającej, usuwającej lub aktualizującej pola są zerowane.

Kod aspektu:

```
package pl.edu.pw.zmwo.aspect;

import pl.edu.pw.zmwo.repo.Document;
import pl.edu.pw.zmwo.repo.Project;
import pl.edu.pw.zmwo.repo.Task;

import java.util.List;

public aspect Zadanie3 {

    private List<Document> Project._cache_getDocuments = null;
    private List<Task> Project._cache_getTasks = null;

    List<Document> around(): call(public List<Document> Project.getDocuments()) {
        Project project = (Project) thisJoinPoint.getTarget();

        if (project._cache_getDocuments == null) {
            project._cache_getDocuments = proceed();
        }

        return project._cache_getDocuments;
    }

    before(): call(public void Project.addDocument(Document)) ||
            call(public void Project.updateDocument(Document)) ||
            call(public void Project.deleteDocument(Document)) {
        Project project = (Project) thisJoinPoint.getTarget();
        project._cache_getDocuments = null;
    }

    List<Task> around(): call(public List<Task> Project.getTasks()) {
        Project project = (Project) thisJoinPoint.getTarget();

        if (project._cache_getTasks == null) {
            project._cache_getTasks = proceed();
        }

        return project._cache_getTasks;
    }

    before(): call(public void Project.addTask(*)) ||
            call(public void Project.updateTask(*)) ||
            call(public void Project.deleteTask(*)) {
        Project project = (Project) thisJoinPoint.getTarget();
        project._cache_getTasks = null;
    }
}
```

W celu przedstawienia działania z mieniłem funkcje main, do następującej postaci:

```
public class Main {

    public static void main(String[] args) {
        Employee employee1 = new Employee("user1", "user1");
        Employee employee2 = new Employee("user2", "user2");

        Repository repository = new Repository();

        Project project1 = new Project("project1");
        Project project2 = new Project("Project2");

        Task task1 = new Task(1, "task1", employee1, employee1);
        Task task2 = new Task(2, "task2", employee2, employee1);
        Task task3 = new Task(3, "task3", employee2, employee2);

        project1.addTask(task1);
        project1.addTask(task2);
        project2.addTask(task3);

        repository.addProject(project1);

        ProjectExporter projectExporter = new ProjectExporter();
        projectExporter.export(project1, System.out);
        project1.addTask(task2);
        projectExporter.export(project1, System.out);
    }
}
```

Wynik działania:

```
Projekt: project1
executing: Project.getTasks
Liczba zadań: 2
Zadanie: 1
Opis: task1
Zgłaszający: user1 user1
Przypisany: user1 user1
Status: NEW
Zadanie: 2
Opis: task2
Zgłaszający: user1 user1
Przypisany: user2 user2
Status: NEW
executing: Project.getDocuments
Liczba dokumentów: 0
Projekt: project1
executing: Project.getTasks
Liczba zadań: 2
Zadanie: 1
Opis: task1
Zgłaszający: user1 user1
Przypisany: user1 user1
Status: NEW
Zadanie: 2
Opis: task2
Zgłaszający: user1 user1
Przypisany: user2 user2
Status: NEW
Liczba dokumentów: 0
```

Zadanie 4

Na początku jest lista z użytkownikami i ich hasła. Przy każdej edycji projektu, weryfikowany jest użytkownik. Po zweryfikowaniu, repozytorium jest dodawane do listy zautoryzowanych repozytoriów.

Kod aspektu:

```
package pl.edu.pw.zmwo.aspect;

import pl.edu.pw.zmwo.repo.Repository;

import java.lang.ref.WeakReference;
import java.util.Arrays;
import java.util.LinkedList;
import java.util.List;
import java.util.Scanner;

public aspect Zadanie4 {

    List<WeakReference<Repository>> auths = new LinkedList<>();

    class UserAuth {
        public final String username;
        public final String password;

        public UserAuth(String username, String password) {
            this.username = username;
            this.password = password;
        }
    }

    List<UserAuth> passwds = Arrays.asList(new UserAuth[]{
            new UserAuth("adam", "qwerty"),
            new UserAuth("ewelina", "asdfg"),
            new UserAuth("stefan", "zxcvb"),
    });

    void verifyUser(Object repo) {
        verifyUser((Repository) repo);
    }

    void verifyUser(Repository repo) {

        for (WeakReference<Repository> weak_repo : auths) {
            if (weak_repo.get() == repo) {
                return;
            }
        }

        Scanner scanner = new Scanner(System.in);
        System.out.print("Użytkownik: ");
        String username = scanner.next();
        System.out.print("Hasło: ");
        String password = scanner.next();

        for (UserAuth passwd : passwds) {
            if (passwd.password.equals(password) &&
                    passwd.username.equals(username)) {

                auths.add(new WeakReference<>(repo));
                return;
            }
        }

        System.err.println("Błędny użytkownik i hasło");
        throw new RuntimeException();
    }

    before(): (execution(* pl.edu.pw.zmwo.repo.Repository.updateProject(..)) ||
            execution(* pl.edu.pw.zmwo.repo.Repository.deleteProject(..))||
```

```
            execution(* pl.edu.pw.zmwo.repo.Repository.addProject(..)) ) {
        verifyUser(thisJoinPoint.getThis());
    }
}
```

Wynik ze złym użytkownikiem/hasłem:

```
Użytkownik: adam
Hasło: qwe
Exception in thread "main" java.lang.RuntimeException
...
```

Wynik ze dobrym użytkownikiem/hasłem:

```
Użytkownik: adam
Hasło: qwerty
Projekt: project1
...
```

Zadanie 5

Zaimplementowałem SoftException i program się kompiluje.

Kod aspektu:

```
package pl.edu.pw.zmwo.aspect;

public aspect Zadanie5 {
    declare soft :Exception: execution(* pl.edu.pw.zmwo.repo.Main.throwException());
}
```

Kod funkcje main:

```
package pl.edu.pw.zmwo.repo;

public class Main {

    public static void main(String[] args) {
        ...
        Main.throwException();
    }

    static void throwException() {
        throw new Exception();
    }
}
```

Wynik:

```
Projekt: project1
...
Liczba dokumentów: 0
Exception in thread "main" org.aspectj.lang.SoftException
        at pl.edu.pw.zmwo.repo.Main.throwException(Main.java:33)
        at pl.edu.pw.zmwo.repo.Main.main_aroundBody2(Main.java:29)
        at pl.edu.pw.zmwo.repo.Main.main_aroundBody3$advice(Main.java:42)
        at pl.edu.pw.zmwo.repo.Main.main(Main.java:1)
Caused by: java.lang.Exception
        ... 4 more

Process finished with exit code 1
```