KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS
KOMPIUTERIŲ KATEDRA

Programų sistemų testavimas (T120B162)

Laboratorinio darbo Nr. 3 ataskaita

Darbą atliko IFF-7/14
Grupės studentai:

Rimvydas Neverauskas

Valentinas Kasteckis

Henrikas Juzuitis

Eligijus Kiudys

Darbą priėmė:

lekt. BARISAS Dominykas

Kaunas, 2020

Turinys

1. Darbo tikslas

Laboratorinio darbo tikslas yra įvertinti mūsų testuojamos programos kokybę naudojant statinio analizavimo metodus, bei įrankius

2. Įvadas

Laboratoriniam darbui pasirinktas objektinio programų projektavimo projektas laivų mūšis, kuris yra rašomas Java kalba. Statinė kodo analizė bus atliekama peržiūrint keletą klasių ir nustatant pažeidimus, bei naudosime statinės kodo analizės Intellij įskiepį PMD.

3. Kodo peržiūros kontrolinis sąrašas

Kodo peržiūrai naudosime šį kontrolinį kodo peržiūros sąrašą:

- Kodo formatavimas – ar tinkami eilučių atitraukimai nuo krašto, ar kodas rašomas nuosekliai, ar yra taikomos kodo rašymo taisyklės pvz., CamelCase naudojamas.
- Kodo architektūra – kodas turi būti struktūrizuotas, ar kodas išskaidytas į failus ar komponentus, ar kodas atitinka taikytus šablonus.
- Kodo gerosios praktikos – ar nėra „hard coding“, naudojamos konstantos ar konfiguracinės reikšmės. Komentarai, turi būti parašyta ne kas daroma, o kodėl tai daroma, aprašomi laikini sprendimai (jei tokie yra), yra pažymimi reikiami padaryti komentarai (TODO). Vengimas daugybės if/else blokų.
- Ne funkciniai reikalavimai – pakartotinis naudojimas, ar yra naudojamas „nepasikartok“ principas (tas pats kodas neturėtų būti kartojamas daugiau kaip du kartus). Išplėčiamumas – ar nesunku praplėsti egzistuojantį kodą. Naudingumas – ar vartotojui yra suprantama naudotojo sąsaja.

Lentelė 1 Kodo peržiūros lentelės šablonas

Failo pavadinimas	
Kodo fragmentas	
Kodo eilutė faile	
Klaidos kategorija	
Klaidos apibūdinimas	
Galimas sprendimas	

4. Programos kodo peržiūra

Failo pavadinimas	ClientShipPart.java
Kodo fragmentas	

```
151     public void RotatePart() {
152         Icon oldIcon = this.partImage;
153         BufferedImage bi = new BufferedImage(
154                 oldIcon.getIconWidth(),
155                 oldIcon.getIconHeight(),
156                 BufferedImage.TYPE_INT_ARGB);
157         Graphics g = bi.createGraphics();
158         oldIcon.paintIcon( c: null, g,  x: 0,  y: 0);
159         g.dispose();
160         this.partImage = new ImageIcon(rotateImage(bi, this.angle));
161     }
```

Kodo eilutė faile	151
Klaidos kategorija	Kodo formatavimas
Klaidos apibūdinimas	Nenaudojamas „CamelCase“ formatavimas
Galimas sprendimas	Pakeisti metodo pavadinimą į „rotatePart“

Failo pavadinimas	GameController.java

Kodo fragmentas

```
879                 String coodrinates = String.format("%d,%d", seaButton.getGridX(), seaButton.getGridY());
880
881                 if(GameData.shootType.equals(GameConstants.underWaterShoot))
882                 {
883                     GameData.UnderWaterUsedInRound = true;
884                 }
885
886                 Command shootShipOnCommand = new ShootShipOnCommand(GameData.GameId, GameData.PlayerId, coodrinates, GameData.shootType, connect
887                 invoker.register( commandName: "shootRegular", shootShipOnCommand);
888                 String data = invoker.execute( commandName: "shootRegular");
889                 JSONObject jsonData = new JSONObject(data);
```

```
940                 if(GameData.shootType.equals(GameConstants.underWaterShoot))
941                 {
942                     GameData.UnderWaterUsedInRound = true;
943                 }
944
945                 Command shootShipOnCommand = new ShootShipOnCommand(GameData.GameId, GameData.PlayerId, coodrinates, GameData.shootType, connec
946                 invoker.register( commandName: "shootRegular", shootShipOnCommand);
947                 String data = invoker.execute( commandName: "shootRegular");
948                 JSONObject jsonData = new JSONObject(data);
```

Kodo eilutė faile	879-889 ir 940-948
Klaidos kategorija	Ne funkciniai reikalavimai
Klaidos apibūdinimas	Kodo duplikacija
Galimas sprendimas	Galime sukurti metodą šiam kodui ir panaudoti tą metodą vietoje kodo duplikacijų

Failo pavadinimas	ClientMap.java

Kodo fragmentas

```
28
29      public int getRows() { return rows; }
32
33      public int getCols() { return cols; }
36
```

Kodo eilutė faile	29, 33
Klaidos kategorija	Architektūrinė
Klaidos apibūdinimas	Nenaudojami metodai
Galimas sprendimas	Panaikinti nenaudojamus metodus

Failo pavadinimas	GameController.java

Kodo fragmentas

```
874         if(GameData.GameLive) {
875             if(GameData.PlayerTurn) {
876                 if (callerGridIndex != 0 && refShipButton != null && refShipButton.getType() == 7) {
877                     if (seaButton.getBackground() != GameConstants.buttonShotSurface && seaButton.getBackground() != GameConstants.buttonShotColor && seaButton.getBackground() != GameConstants.buttonShotSuc
878                             ((GameData.shootType.equals(GameConstants.superShoot) && GameData.RoundShootSuper - GameData.RoundId <= 0 ) || (GameData.shootType.equals(GameConstants.underWaterShoot) && GameDa
```

Kodo eilutė faile	877,878
Klaidos kategorija	Kodo formatavimas
Klaidos apibūdinimas	877 eilutėje if formuluotė netelpa į ekraną
Galimas sprendimas	Suskaidyti į kelias eilutes

Failo pavadinimas	ClientShip.java

Kodo fragmentas

```
196                 else if (shipPart.getX() == shot.getX() && shipPart.getY() == shot.getY() && shipPart.getPart().getShipType() == ShipType.SUBMARINE && shot.
197
198                     shipHits.add(shot);
199
200                 }
201                 else if(shipPart.getX() == shot.getX() && shipPart.getY() == shot.getY() && shipPart.getPart().getShipType() == ShipType.SUBMARINE && ( shot
202                 {
203                     if(shipPart.getHitOnce() && shield) {
```

Kodo eilutė faile	Kodo formatavimas
Klaidos kategorija	197,201
Klaidos apibūdinimas	if formuluotė netelpa į ekraną
Galimas sprendimas	Suskaidyti į kelias eilutes

Failo pavadinimas	ClientShipPartBaseShip.java ir ClientShipPartDamaged.java

Kodo fragmentas

```
24                  Icon oldIcon = new ImageIcon(shipPartImage);
25                  BufferedImage bi = new BufferedImage(
26                          oldIcon.getIconWidth(),
27                          oldIcon.getIconHeight(),
28                          BufferedImage.TYPE_INT_ARGB);
29                  Graphics g = bi.createGraphics();
30                  oldIcon.paintIcon( c: null, g,  x: 0, y: 0);
31                  g.dispose();
32                  shipPartImage = rotateImage(bi, this.getPart().getAngle());
```

Kodo eilutė faile	24-32
Klaidos kategorija	Ne funkciniai reikalavimai
Klaidos apibūdinimas	Pasikartojantis kodo fragmentas
Galimas sprendimas	Iškelti logiką į atskirą klasę

Failo pavadinimas	ClientShipPartArmour.java

Kodo fragmentas

```
15      public ClientShipPartArmour(ClientShipPartInterface decoratedShipPart) {
16          super(decoratedShipPart);
17          try {
18              shipPartImage = ImageIO.read(new FileInputStream(GameConstants.shipShieldImage));
19              if (shipPartImage != null) {
20                  shipPartImage = shipPartImage.getScaledInstance(ComponentsSizeConstants.gridButtonWidth, ComponentsSizeConstants.gridButtonHeight, Image.SCALE_DEFAULT);
21              }
22          } catch (Exception e) {
23              System.out.println(e.toString());
24          }
25      }
```

Kodo eilutė faile	22-24
Klaidos kategorija	Kodo gerosios praktikos
Klaidos apibūdinimas	Nesusidorojimas su iškilusia klaida
Galimas sprendimas	Perduoti klaidą atsakingam padaliniui arba susidoroti su ja iškart

Failo pavadinimas	Shoot.java

```
52  //              error = false;
53  //              resultSet = statement.executeQuery("SELECT COUNT(id) as cnt FROM shoots WHERE fk_player_id = "+ playerId +" AND x = "+ sp
54  //              resultSet.next();
55  //              Integer shootCordinateCheck = resultSet.getInt("cnt");
56  //              if(!error && shootCordinateCheck > 0) {
57  //                  error = true;
58  //              }
59
60  //              System.out.println(coordinates.getString(i));
61                  if(!error) {
62                      resultSet = statement.executeQuery( sql: "SELECT ship_parts.id as part_id, ships.id as ship_id, ship_parts.armour as armo
```

Kodo eilutė faile	52
Klaidos kategorija	Vizualinė
Klaidos apibūdinimas	Paliktas užkomentuotas kodas
Galimas sprendimas	Ištrinti komentarus

Failo pavadinimas	Log.java, Shoot.java, Message.java ir kt.

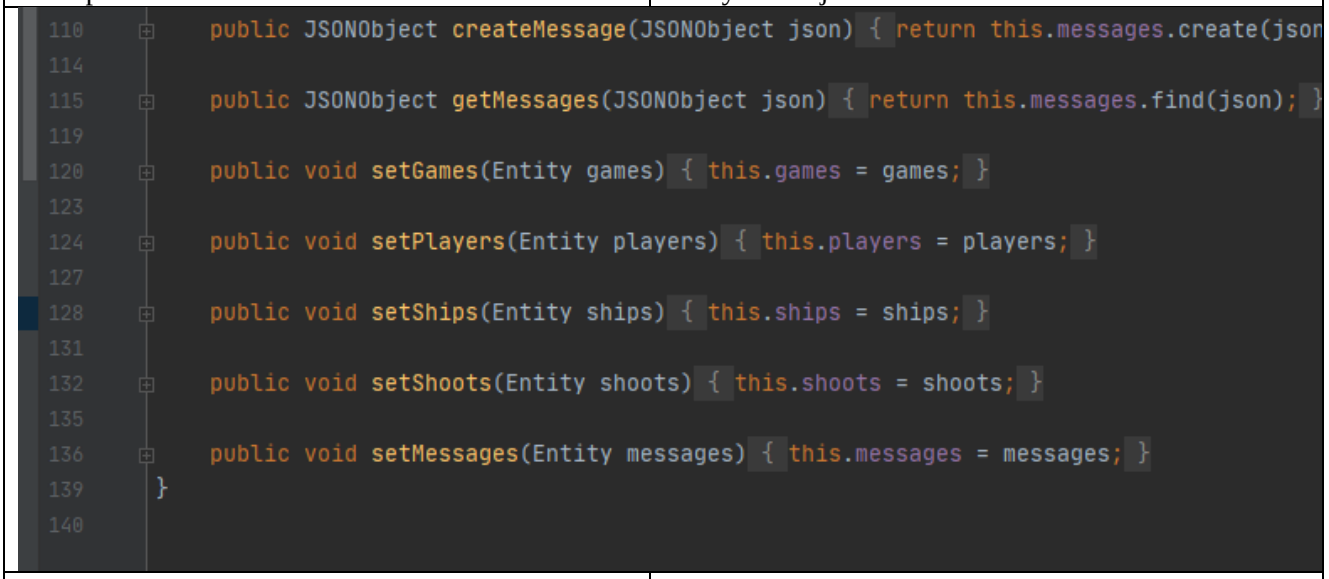
```java
package server.Entities;

import ...

public class Log implements Entity {
    @Override
    public JSONObject find(JSONObject json) { return null; }

    @Override
    public void update(JSONObject json) {

    }

    @Override
    public void remove(JSONObject json) {

    }
```

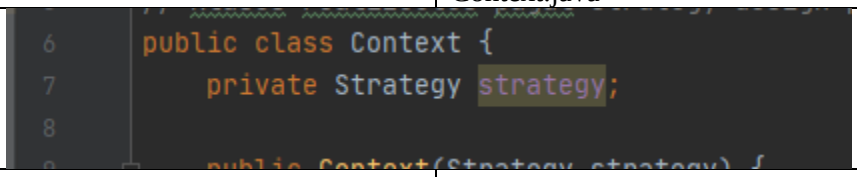
Kodo eilutė faile	-
Klaidos kategorija	Architekturinė
Klaidos apibūdinimas	Kiekviena klasė implementuoja Entity interface, tačiau tas interface turi kelis metodus ir tos klasės nevisus metodus realizuoja.
Galimas sprendimas	Panaikinti interface nes galimai yra perteklinis

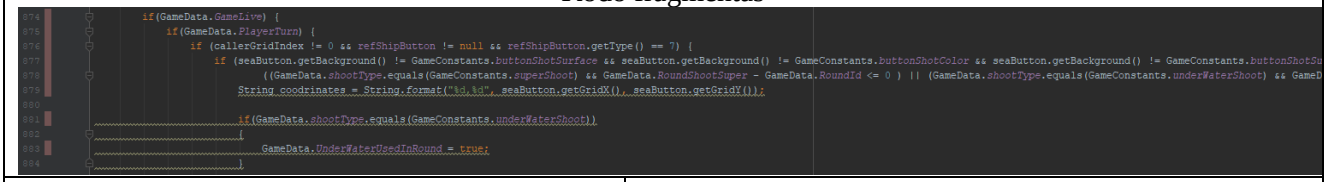
Failo pavadinimas	Game.java

```java
        resultSet = statement.executeQuery( sql: "SELECT COUNT(DISTINCT fk_player_id) as playerCnt FROM ship
                "INNER JOIN players ON players.id = ships.fk_player_id " +
                "WHERE fk_game_id = " + gameId);
        resultSet.next();
        if (resultSet.getInt( columnLabel: "playerCnt") < 2) {
            jsonObject.append("GameStatus", "Waiting ships");
            jsonObject.append("RoundNumber", 0);
            return jsonObject;
        }

        resultSet = statement.executeQuery( sql: "SELECT * FROM games WHERE id=" + gameId);
        resultSet.next();
        int currentRound = resultSet.getInt( columnLabel: "round");

        resultSet = statement.executeQuery( sql: "SELECT DISTINCT fk_player_id as playerId, username FROM sh
                "INNER JOIN ship_parts ON ship_parts.fk_ship = ships.id " +
                "INNER JOIN players ON players.id = fk_player_id " +
                "WHERE destroyed = 0 AND fk_game_id = " + gameId);
        resultSet.next();
```

Kodo eilutė faile	64-100
Klaidos kategorija	Vizualinis
Klaidos apibūdinimas	Visi SQL yra labai negražiai hard codinti ir blogas skaitomumas
Galimas sprendimas	Išsikelti visus SQL į atskirą klasę

Failo pavadinimas	EntityMaker.java
	
Kodo eilutė faile	110-136 eilutės
Klaidos kategorija	Architektūrinis
Klaidos apibūdinimas	Visi setterių metodai yra naudingi tik testavimui, kad užsimockinti objektus, kitose vietose šie metodai iš viso nėra naudojami.
Galimas sprendimas	Susikurti atskirą klasę kuri paveldėtu šią klasę ir galbūt į ją susikelti tuos metodus

Failo pavadinimas	Context.java
	
Kodo eilutė faile	7
Klaidos kategorija	Sintaksinė
Klaidos apibūdinimas	Strategy kintamasis yra rekomenduojama, kad būtu final
Galimas sprendimas	Pakeisti property į final

Failo pavadinimas	GameController.java
Kodo fragmentas	
	
Kodo eilutė faile	874,875
Klaidos kategorija	Kodo gerosios praktikos
Klaidos apibūdinimas	Bereikalingai išskirtos if formuluotės
Galimas sprendimas	Galima padaryti if formuluote su and kad būtų: if (GameData.GameLive && GameData.PlayerTurn)

Failo pavadinimas	ConnectionManager.java
Kodo fragmentas	

```
185         // TODO: OLD Logic
186         //  for (int partId = 0; partId < shipPartList.size(); partId++) {
187         //      String coordinate = String.format("%d,%d",shipPartList[partId].getX(), shipPartList[partId].getY());
188         //      shipParts.put(coordinate);
189         //  }
```

Kodo eilutė faile	185-189
Klaidos kategorija	Kodo gerosios praktikos
Klaidos apibūdinimas	Neaiškus komentaras
Galimas sprendimas	Ištrinti nereikalingą komentarą arba jį pataisyti į aiškesnį.

Failo pavadinimas	ClientPlayer.java
Kodo fragmentas	

```
44      public void appendShips(ClientShip ship) {
45          Collection parts = new ShipPartCollection(ship);
46          Iterator iterator = parts.createIterator();
47          while (iterator.hasNext()) {
48              ClientShipPart shipPart = (ClientShipPart) iterator.next();
49              int x = shipPart.getX();
50              int y = shipPart.getY();
51
52              ClientCoordinate coordinate = map.getCoordinate(x, y);
53              coordinate.setType(ClientCoordinateType.SHIP);
54          }
55
56  //          for (ClientShipPart newShip: ship.shipParts) {
57  //              int x = newShip.getX();
58  //              int y = newShip.getY();
59  //
60  //              ClientCoordinate coordinate = map.getCoordinate(x, y);
61  //              coordinate.setType(ClientCoordinateType.SHIP);
62  //          }
63          ships.add(ship);
64      }
```

Kodo eilutė faile	56-62
Klaidos kategorija	Kodo gerosios praktikos
Klaidos apibūdinimas	Paliktas užkomentuotas ciklas
Galimas sprendimas	Ištrinti šį komentarą

Failo pavadinimas	PlaceShipOnCommand.java
Kodo fragmentas	

```
13      private ConnectionManager conManager;
14      private SeaButton[][][] seaButtons;
15      private SeaButton[][][] undoSeaButtons;
16      private SeaButton seaButton;
17      private SeaButton undoSeaButton;
18      private ShipButton refShipButton;
19      private ShipButton undoRefShipButton;
20      private ArrayList<ClientShipPartInterface> newShipParts;
21      private ArrayList<ClientShipPartInterface> undoNewShipParts;
22      private String rotation;
23      private String undoRotation;
24      private ClientShip originalShip;
25      private ClientShip undoOriginalShip;
26      private int callerGridIndex;
27      private int undoCallerGridIndex;
28      private int shipLength;
29      private int undoShipLength;
30      private int shipsLeft;
31      private int undoShipsLeft;
```

Kodo eilutė faile	18,23,24
Klaidos kategorija	Kodo gerosios praktikos
Klaidos apibūdinimas	Yra aprašyti nenaudojami kintamieji
Galimas sprendimas	Juos panaikinti

Failo pavadinimas	MenuPanel.java

Kodo fragmentas

```
33      // Unused buttons section
34      private ShipButton unusedButton1;
35      private ShipButton unusedButton2;
36      private ShipButton unusedButton3;
37      private ShipButton unusedButton4;
38      private ShipButton unusedButton5;
39      private ShipButton unusedButton6;
```

Kodo eilutė faile	34-39
Klaidos kategorija	Kodo gerosios praktikos
Klaidos apibūdinimas	Nekorektiški kintamųjų pavadinimai
Galimas sprendimas	Refaktorinti pagal į labiau tinkamus pavadinimus

Failo pavadinimas	MenuPanel.java
Kodo fragmentas	

```
373         @Override
374 @       public void actionPerformed(ActionEvent e) {
375             Object sender = e.getSource();
376
377             if (sender.equals(createGameButton)) {
378                 // Create Game button pressed
379                 delegate.createGameButtonPressed( menuPanel: this, createGameButton);
380             }else if (sender.equals(gameLevelOptions)){
381                 // User selected gameLevel
382                 delegate.gameLevelSelected( menuPanel: this, gameLevelOptions);
383             }else if (sender.equals(joinGameButton)){
384                 // Join game button pressed
385                 delegate.joinGameButtonPressed( menuPanel: this, joinGameButton);
386             }else if (sender.equals(unusedButton1)){
387                 delegate.unusedButtonPressed( menuPanel: this, unusedButton1);
388                 unusedButton1.setActive(true);
389                 unusedButton2.setActive(false);
390                 unusedButton3.setActive(false);
391                 unusedButton4.setActive(false);
392                 unusedButton5.setActive(false);
393                 unusedButton6.setActive(false);
```

Kodo eilutė faile	380, 383, 386
Klaidos kategorija	Kodo formatavimas
Klaidos apibūdinimas	Netolygūs skliaustų atitraukimai tarp sąlygų
Galimas sprendimas	Suvienodinti tarpus

Failo pavadinimas	FloatingShipConfigurator.java

Kodo fragmentas

```
7   import client.Controllers.GameController;
8   import client.GameConstants.GameConstants;
9   import client.GameConstants.ShipParameters;
10  import client.Models.ClientFloatingShip;
11  import client.Models.ClientShip;
12  import client.Models.ClientShipPart;
13  import client.Models.Decorator.ClientShipPartArmour;
14  import client.Models.Decorator.ClientShipPartBase;
15  import client.Models.Decorator.ClientShipPartBaseShip;
16  import client.Models.Decorator.ClientShipPartInterface;
17  import client.Models.GameData;
18  import client.Views.SeaButton;
19
20  import javax.imageio.ImageIO;
21  import javax.swing.*;
22  import java.awt.*;
23  import java.io.FileInputStream;
24  import java.io.IOException;
25  import java.util.ArrayList;
```

Kodo eilutė faile	7, 8, 18, 20, 21, 22, 23, 24
Klaidos kategorija	Kodo gerosios praktikos
Klaidos apibūdinimas	Nenaudojami importai
Galimas sprendimas	Panaikinti nenaudojamus importus

Failo pavadinimas	ShootShipOnCommand.java

Kodo fragmentas	

<pre>3 import client.Models.Bridge.*;
4 import client.Models.ConnectionManager;
5 import client.Models.GameData;
6 import org.json.JSONObject;</pre>	

Kodo eilutė faile	3, 5, 6
Klaidos kategorija	Kodo gerosios praktikos
Klaidos apibūdinimas	Nenaudojami importai
Galimas sprendimas	Panaikinti nenaudojamus importus

5. Statinė kodo analizė naudojant įrankį

Statinės kodo analizės įrankiu yra testuojama aplikacija, kuri yra padaryta su Java programavimo kalba, todėl šiam testavimui naudosime Intellij PMD statinės kodo analizės įskiepį, kuris leidžia atlikti šių kategorijų testus:

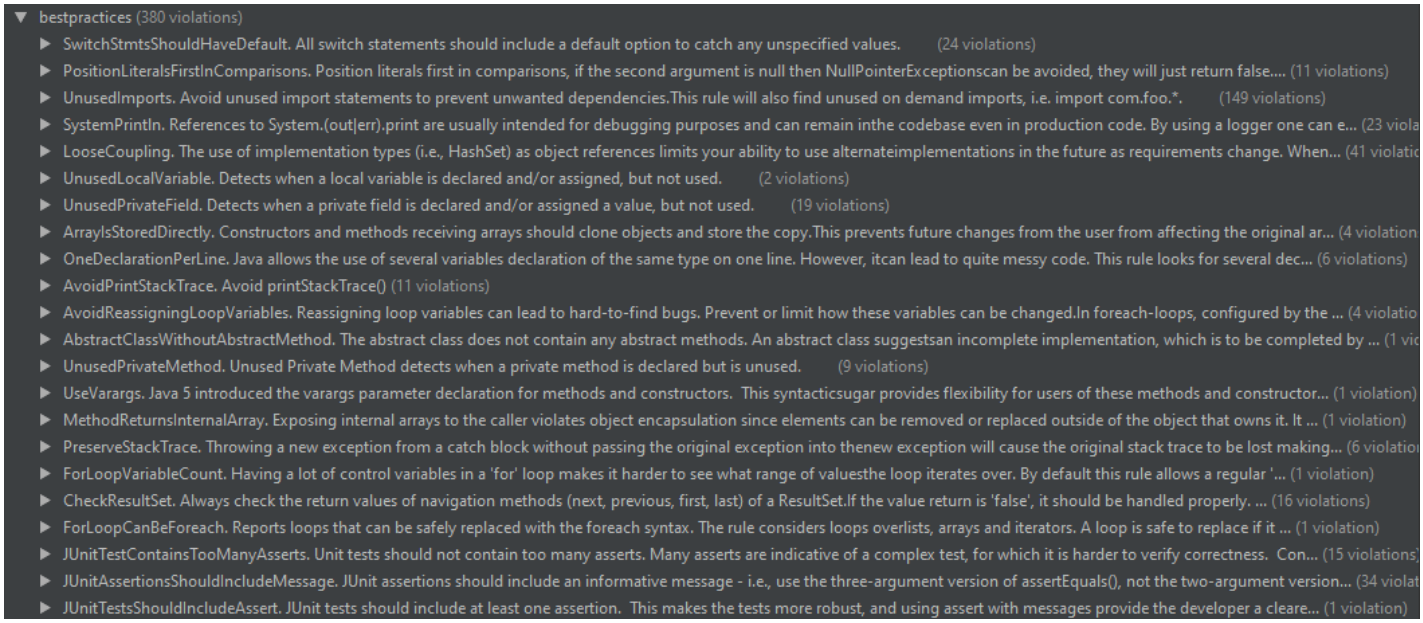
- Geriausių kodo praktikų
- Kodo stiliaus
- Dizaino
- Dokumentacijos
- Kodo linkusio į klaidas
- Daugelio gijų
- Našumo

Taip pat pasirašysime savo taisyklę, kuri bus tikrins ar „if“ sąlyga turi skliaustus, dėl patogesnio skaitomumo.

Atlikę statinę kodo analizę naudodami Intellij PMD įskiepį gavome tokius rezultatus:

Pav. 1 PMD statinės analizės rezultatai

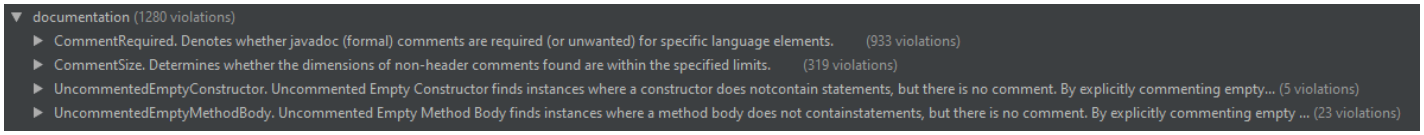
Pav. 2 Geriausių praktikų statinės kodo analizės rezultatai



Pav. 3 Kodo stiliaus statinės analizės rezultatai

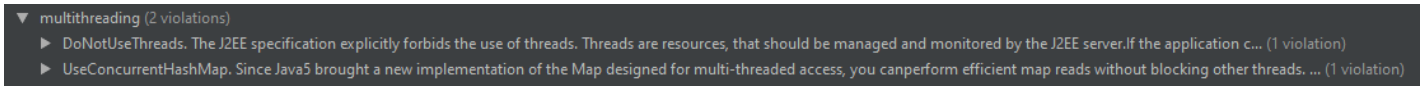
Pav. 4 Dizaino statinės kodo analizės rezultatai

Pav. 5 Dokumentacijos statinės kodo analizės rezultatai

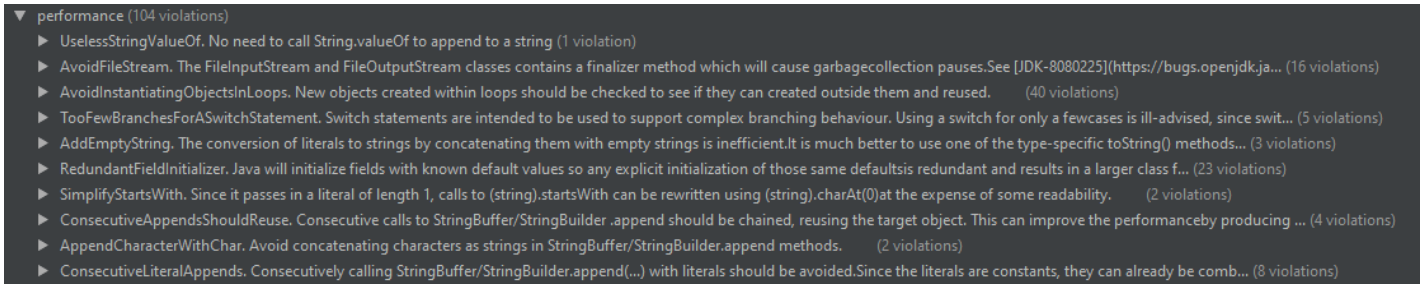


Pav. 6 Linkimo į klaidas kodo statinės analizės rezultatai

Pav. 7 Daugelio gijų statinės kodo analizės rezultatai



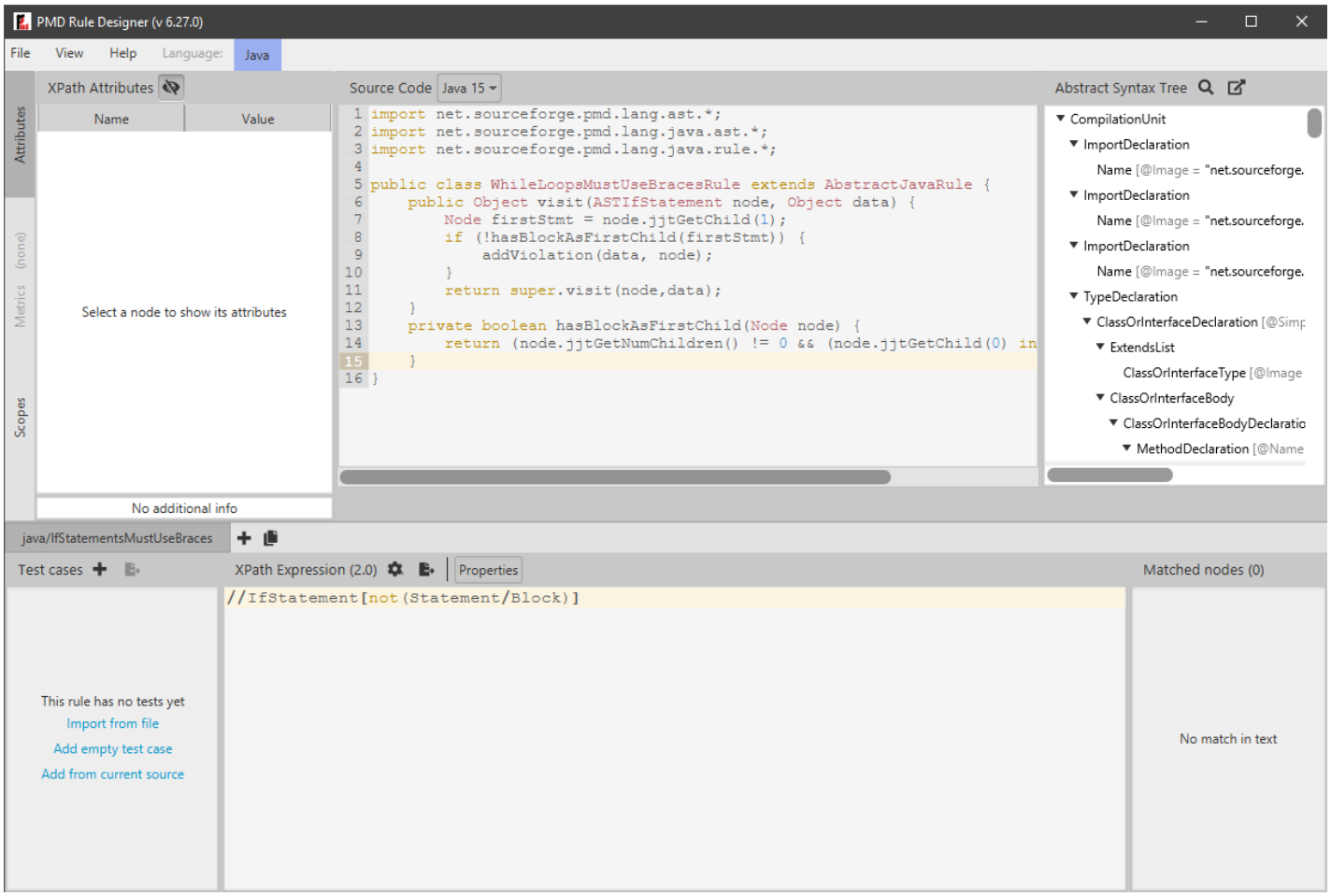
Pav. 8 Našumo statinės kodo analizės rezultatai



Kaip matome iš rezultatų, yra padaryta daug pažeidimų įvairiose statinės kodo analizės kategorijose. Jeigu šį įrankį būtume naudoję nuo programos kūrimo pradžios, būtume galėją nesunkiai šių klaidų išvengti reguliariai atlikdami statinę kodo analizę, tačiau dabar tai atlikti yra gan sudėtinga, kadangi yra daug programinio kodo kurį reikėtų keisti.

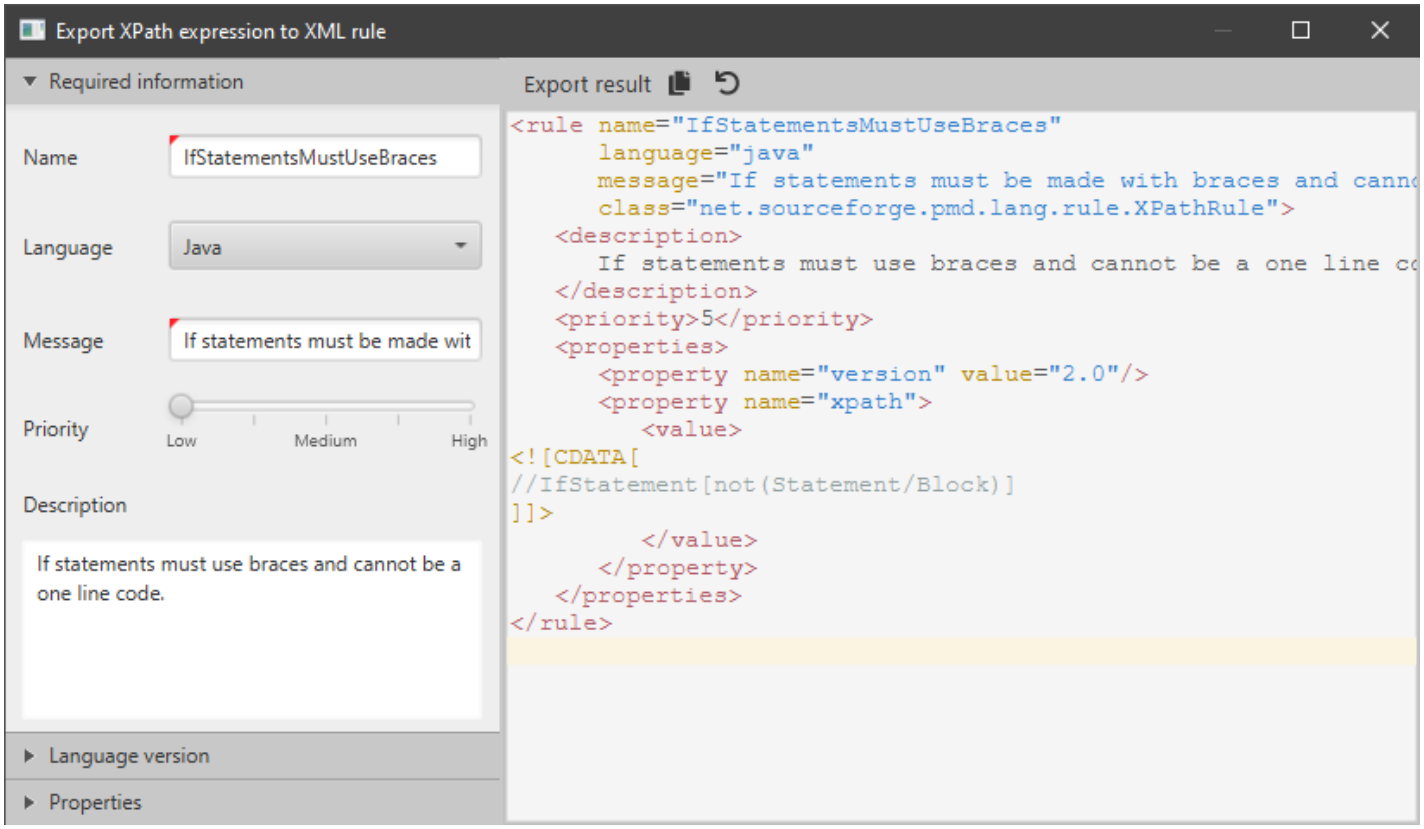
Naudodami PDM Designer sukūrėme savo taisyklę, kuri tikrina ar „if“ sąlyga turi skliaustus ar ne:

Pav. 9 PMD Designer parašyta taisyklė



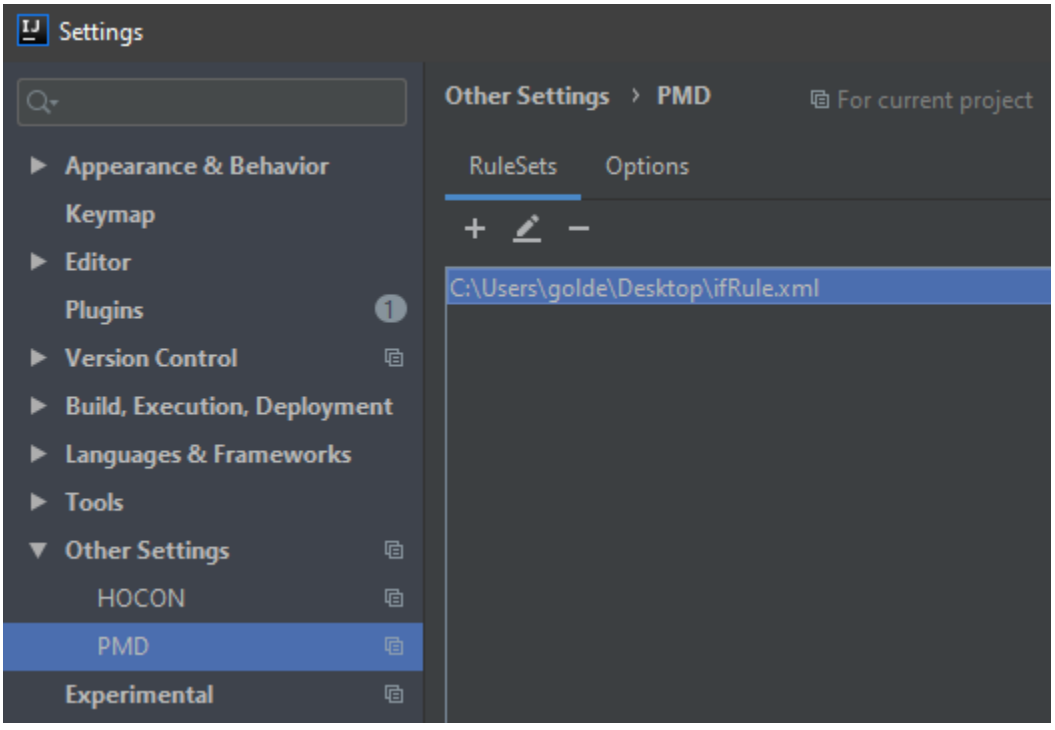
Toliau aprašome taisyklę ir eksportuojame šią taisyklę kaip xml failą:

Pav. 10 Eksportuota taisyklė į XML



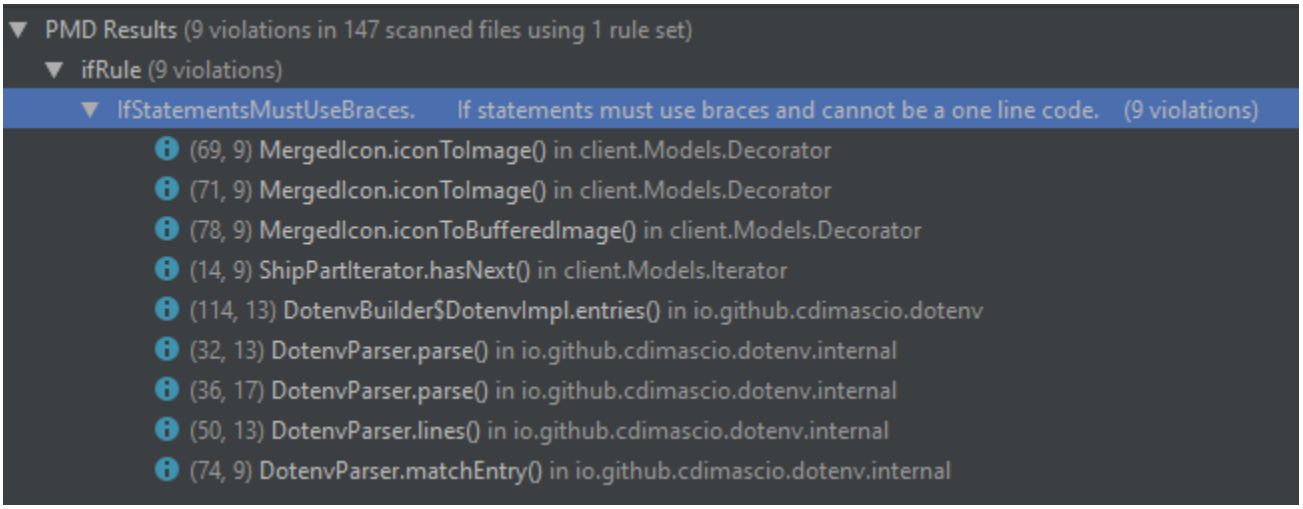
Pridėjome šią taisyklę Intellij:

Pav. 11 Pridėta taisyklė



Paleidome šią taisyklę su PDM ir gavome šiuos rezultatus:

Pav. 12 Gauti mūsų taisyklės rezultatai



Patikriname programos kode ar taisyklė yra tikrai pažeista:

Lentelė 2 Pažeista mūsų sukurta statinės kodo analizės taisyklė

Failo pavadinimas	MergedIcon.java

Kodo fragmentas
`@ public static Image iconToImage(Icon icon) {`
`if (icon == null)`
`return null;`
`if (icon instanceof ImageIcon)`
`return ((ImageIcon) icon).getImage();`
`return iconToBufferedImage(icon);`
`}`	

Kodo eilutė faile	69,71
Klaidos kategorija	Kodo formatavimas
Klaidos apibūdinimas	If sąlygoje nėra skliaustelių
Galimas sprendimas	Uždėti skliaustelius šiose if sąlygose

6. Išvados

Pasidarėme kodo analizę tiesiog peržvelgiant kodą, radome klaidų. Taip pat įsidiegėme papildomą IDEA aplinkos pluginą PMD kuris mums leido atlikti statinę kodo analizė. Jos pagalba, paaiškėjo, kad turime daug klaidų. To pasėkoje, suvokėme, kad jau nuo projekto pradžios būtų buvę naudinga ir tikslinga naudoti tokį įrankį kaip PMD, kad šios klaidos būtų išvengtos iškart. Taip pat laboratorinio darbo metu išmokome susikurti papildomų taisyklių PMD įrankiui. .nuo projekto kūrimo pradžios. Taip pat išmokome susikurti papildomų taisyklių, kadangi, taip pat išmokome susikurti papildomų taisyklių.