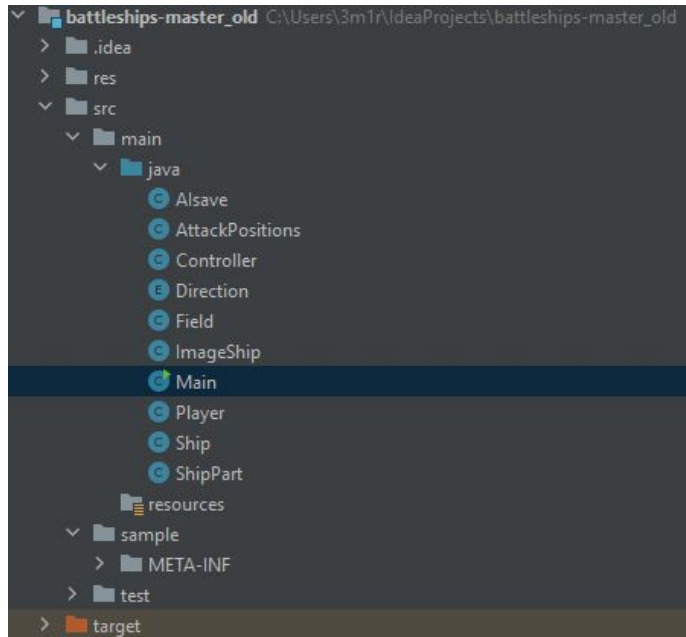


# Battleship

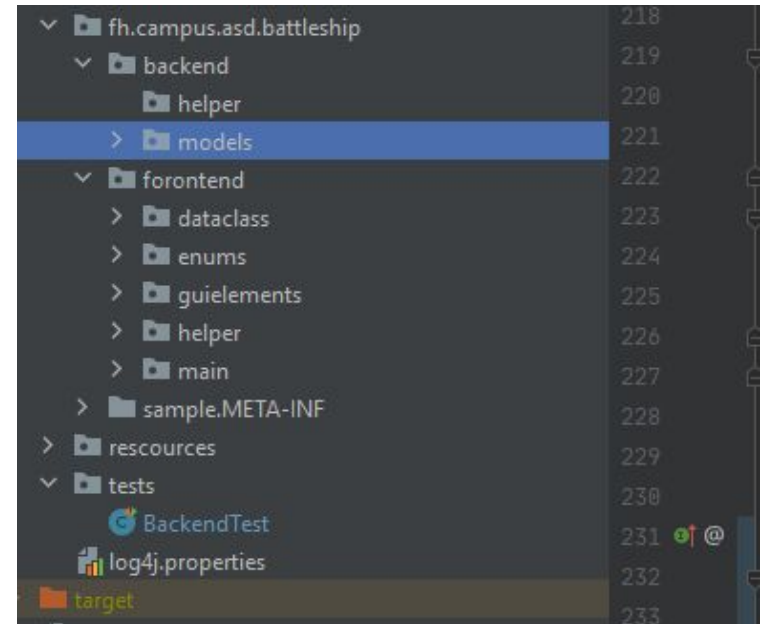
Gruppe2: Addo, Cajlakovic, Lulzim,

# Paketstruktur

Alt



Neu



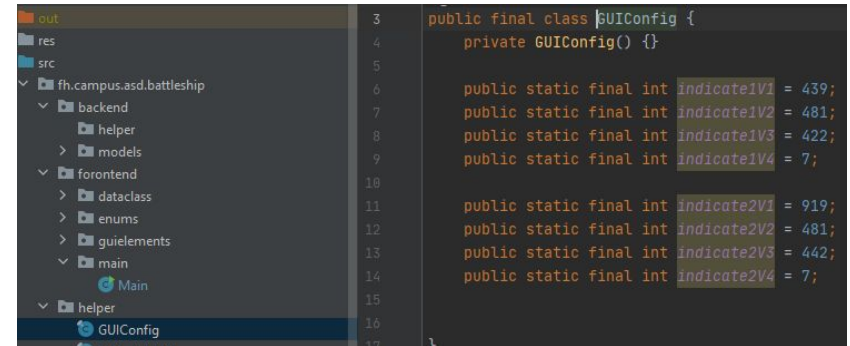
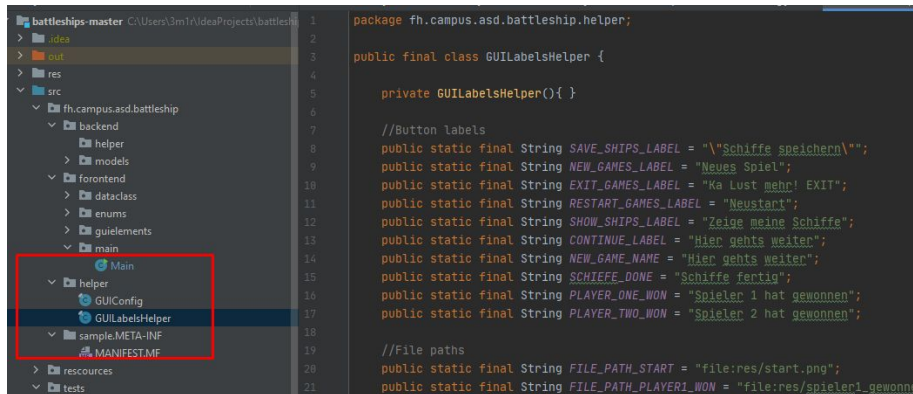
# GUI Labels und Konfigurationsparameter zentralisiert

```
public class Main extends Application
{
    private Player player1 = new Player( isHuman: true);
    private Player player2 = new Player( isHuman: true);
    private double pressedX, pressedY;
    private int gameround = 1;
    private boolean shipscomplete = false; //zu testzwecken auf true später muss auf false

    private Button buttonSaveShipsLeft = new Button( s: "Schiffe speichern");
    private Button buttonSaveShipsRight = new Button( s: "Schiffe Speichern");
    private Button newGame = new Button( s: "Neues Spiel");
    private Button exit = new Button( s: "Ka Lust mehr! EXIT");
    private Button reset = new Button( s: "Neustart");
    private Button seeShips1 = new Button( s: "Zeige meine Schiffe");
    private Button seeShips2 = new Button( s: "Zeige meine Schiffe");
    private Button cont = new Button( s: "Hier gehts weiter");
}
```

Alt

Neu



# Code vereinfachung

```
    {  
        a = calculateXY(x, y, p1x: 440 + 40, p1y: 40 + 40, p2x: 440 + 440, p2y: 440 + 40);  
  
        if (a.length > 0)  
        {  
            if (player1.attackPossible(a[0], a[1]))
```

```
buttonSaveShipsLeft.setLayoutX(1800 - 1520 - 3 * 40);  
buttonSaveShipsLeft.setLayoutY(500);  
buttonSaveShipsLeft.setPrefSize(120, 10);  
  
buttonSaveShipsLeft.setOnAction(event -> {  
    saveShips(imageShip0, player1, p1x: 440 + 40, p1y: 40 + 440 + 40 + 40, p2x: 440 + 440, p2y: 40 + 920);  
    shipsComplete();  
});  
  
buttonSaveShipsRight.setLayoutX(1520);  
buttonSaveShipsRight.setLayoutY(500);  
buttonSaveShipsRight.setPrefSize(120, 10);  
buttonSaveShipsRight.setOnAction(  
    event -> {  
        saveShips(imageShip1, player2, p1x: 2 * 440 + 40 + 40, p1y: 40 + 440 + 40 + 40, p2x: 440 + 440 + 40 + 440, p2y: 920 + 40);  
        shipsComplete();
```

# Ein Logger wurde hinzugefügt

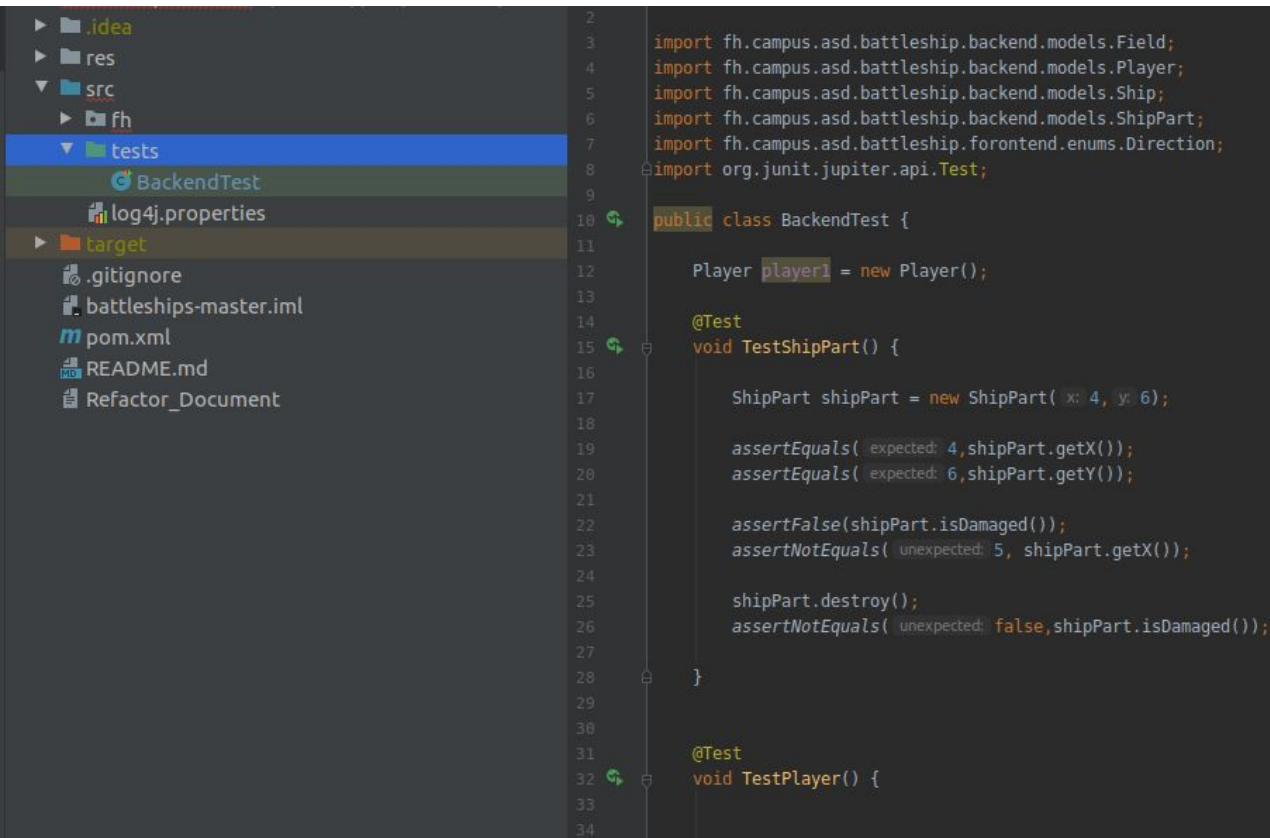
Alt:

```
{  
    if (shipscomplete)  
    {  
        System.out.println("Schiffe fertig");  
        if (gameround % 2 == 1)  
        {
```

Neu:

```
{  
    if (shipscomplete)  
    {  
        Log.debug(GUILabelsHelper.SCHIEFE_DONE);  
        if (gameround % 2 == 1)  
        {
```

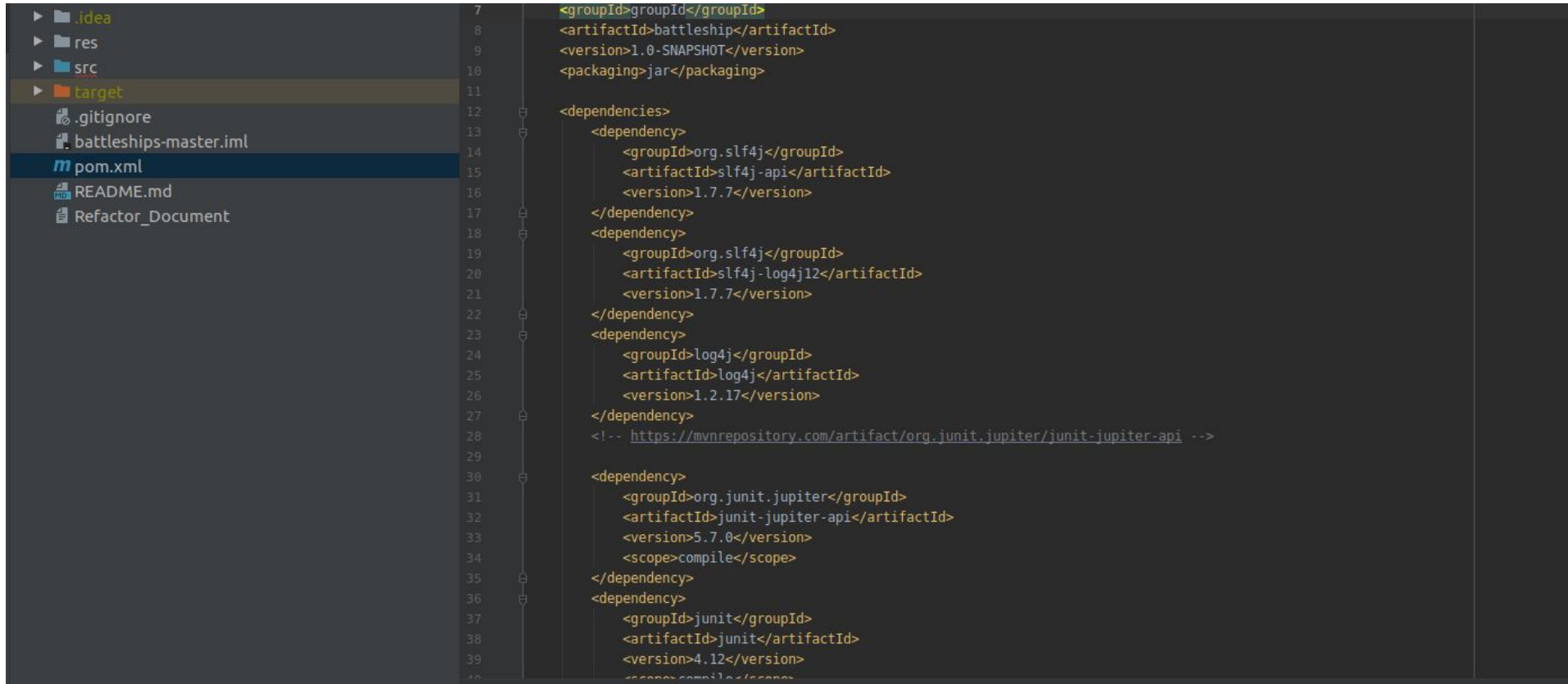
# Tests hinzugefügt



The screenshot shows an IDE with a project structure on the left and Java code on the right. The project structure includes a 'tests' directory containing 'BackendTest', 'log4j.properties', and a 'target' directory. The code on the right is for 'BackendTest' and includes imports for models and enums, followed by two test methods: 'TestShipPart()' and 'TestPlayer()'.

```
2
3 import fh.campus.asd.battleship.backend.models.Field;
4 import fh.campus.asd.battleship.backend.models.Player;
5 import fh.campus.asd.battleship.backend.models.Ship;
6 import fh.campus.asd.battleship.backend.models.ShipPart;
7 import fh.campus.asd.battleship.forontend.enums.Direction;
8 import org.junit.jupiter.api.Test;
9
10 public class BackendTest {
11
12     Player player1 = new Player();
13
14     @Test
15     void TestShipPart() {
16
17         ShipPart shipPart = new ShipPart( x: 4, y: 6);
18
19         assertEquals( expected: 4, shipPart.getX());
20         assertEquals( expected: 6, shipPart.getY());
21
22         assertFalse(shipPart.isDamaged());
23         assertEquals( unexpected: 5, shipPart.getX());
24
25         shipPart.destroy();
26         assertEquals( unexpected: false, shipPart.isDamaged());
27
28     }
29
30
31     @Test
32     void TestPlayer() {
33
34
```

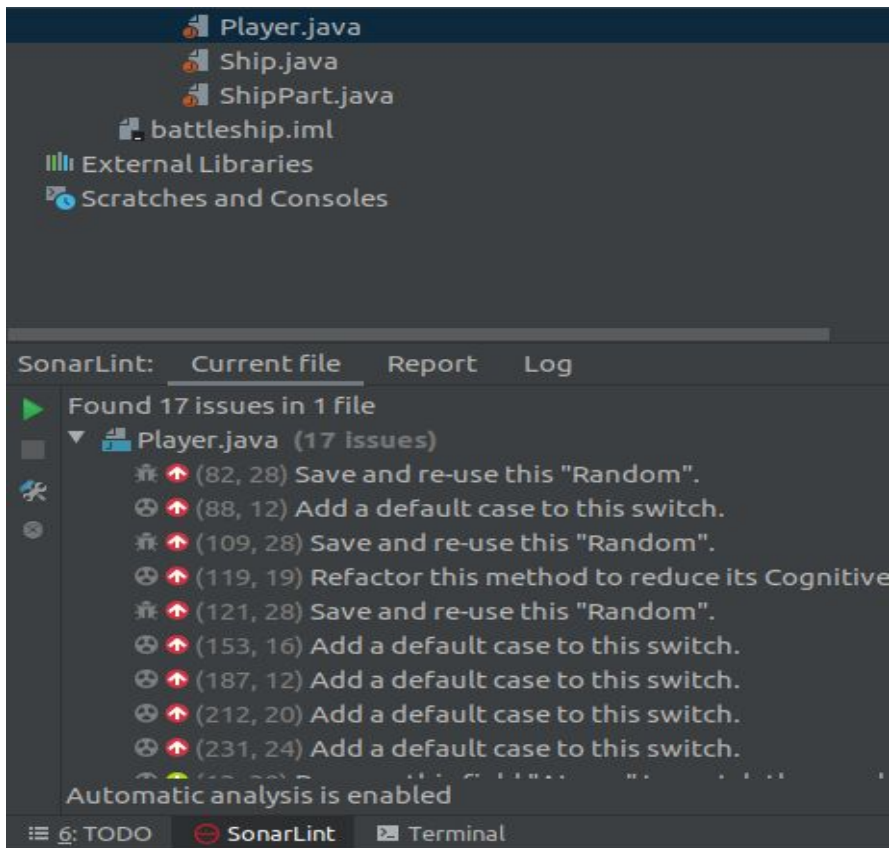
# Dependency manager und Readme hinzugefügt



The screenshot shows an IDE interface with a project structure on the left and a code editor on the right. The project structure includes folders like .idea, res, src, and target, and files like .gitignore, battleships-master.iml, pom.xml, README.md, and Refactor\_Document. The code editor displays the content of pom.xml, which is a Maven POM file. It defines a project with groupId 'groupId', artifactId 'battleship', and version '1.0-SNAPSHOT'. It also lists several dependencies: slf4j-api (version 1.7.7), slf4j-log4j12 (version 1.7.7), log4j (version 1.2.17), junit-jupiter-api (version 5.7.0, scope compile), and junit (version 4.12, scope compile).

```
7 <groupId>groupId</groupId>
8 <artifactId>battleship</artifactId>
9 <version>1.0-SNAPSHOT</version>
10 <packaging>jar</packaging>
11
12 <dependencies>
13   <dependency>
14     <groupId>org.slf4j</groupId>
15     <artifactId>slf4j-api</artifactId>
16     <version>1.7.7</version>
17   </dependency>
18   <dependency>
19     <groupId>org.slf4j</groupId>
20     <artifactId>slf4j-log4j12</artifactId>
21     <version>1.7.7</version>
22   </dependency>
23   <dependency>
24     <groupId>log4j</groupId>
25     <artifactId>log4j</artifactId>
26     <version>1.2.17</version>
27   </dependency>
28   <!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->
29
30   <dependency>
31     <groupId>org.junit.jupiter</groupId>
32     <artifactId>junit-jupiter-api</artifactId>
33     <version>5.7.0</version>
34     <scope>compile</scope>
35   </dependency>
36   <dependency>
37     <groupId>junit</groupId>
38     <artifactId>junit</artifactId>
39     <version>4.12</version>
40     <scope>compile</scope>
```

# Sonarlint fixes

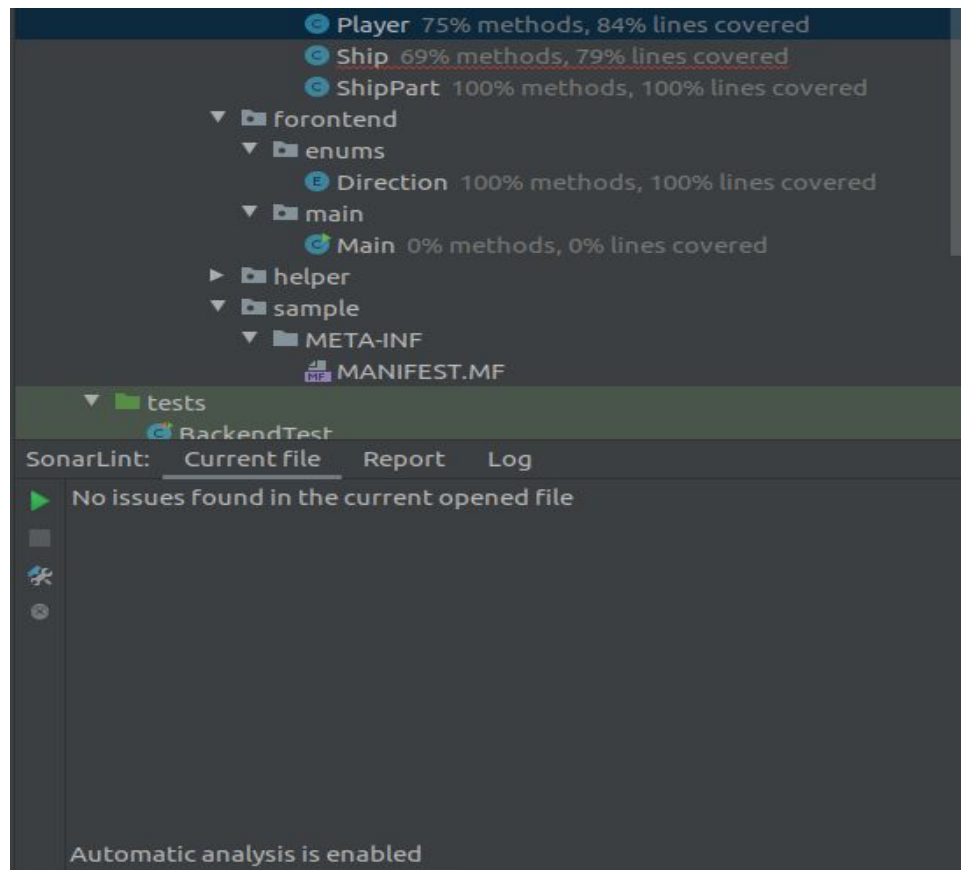


This screenshot shows the IntelliJ IDEA interface with the SonarLint plugin. The left sidebar displays the project structure with files `Player.java`, `Ship.java`, `ShipPart.java`, and `battleship.iml`. The main editor area shows the SonarLint tab for `Player.java`, which has 17 issues. The issues list includes:

- Found 17 issues in 1 file
- Player.java (17 issues)
- (82, 28) Save and re-use this "Random".
- (88, 12) Add a default case to this switch.
- (109, 28) Save and re-use this "Random".
- (119, 19) Refactor this method to reduce its Cognitive
- (121, 28) Save and re-use this "Random".
- (153, 16) Add a default case to this switch.
- (187, 12) Add a default case to this switch.
- (212, 20) Add a default case to this switch.
- (231, 24) Add a default case to this switch.

Automatic analysis is enabled

At the bottom, the status bar shows tabs for `6: TODO`, `SonarLint`, and `Terminal`.



This screenshot shows the IntelliJ IDEA interface with the SonarLint plugin, displaying coverage information for the entire project. The left sidebar shows the project structure with folders `forontend`, `enums`, `main`, `helper`, `sample`, `META-INF`, and `tests`. The main editor area shows the SonarLint tab for the project, which displays coverage information for the following files:

- Player 75% methods, 84% lines covered
- Ship 69% methods, 79% lines covered
- ShipPart 100% methods, 100% lines covered
- Direction 100% methods, 100% lines covered
- Main 0% methods, 0% lines covered
- MANIFEST.MF
- BackendTest

No issues found in the current opened file

Automatic analysis is enabled