

به نام خدا

گزارش آزمایش ششم

آزمایشگاه مهندسی نرم افزار

محمدعلی نادمی ۹۶۱۰۹۸۹۹

علی قاسمی ۹۷۱۰۶۲۰۵

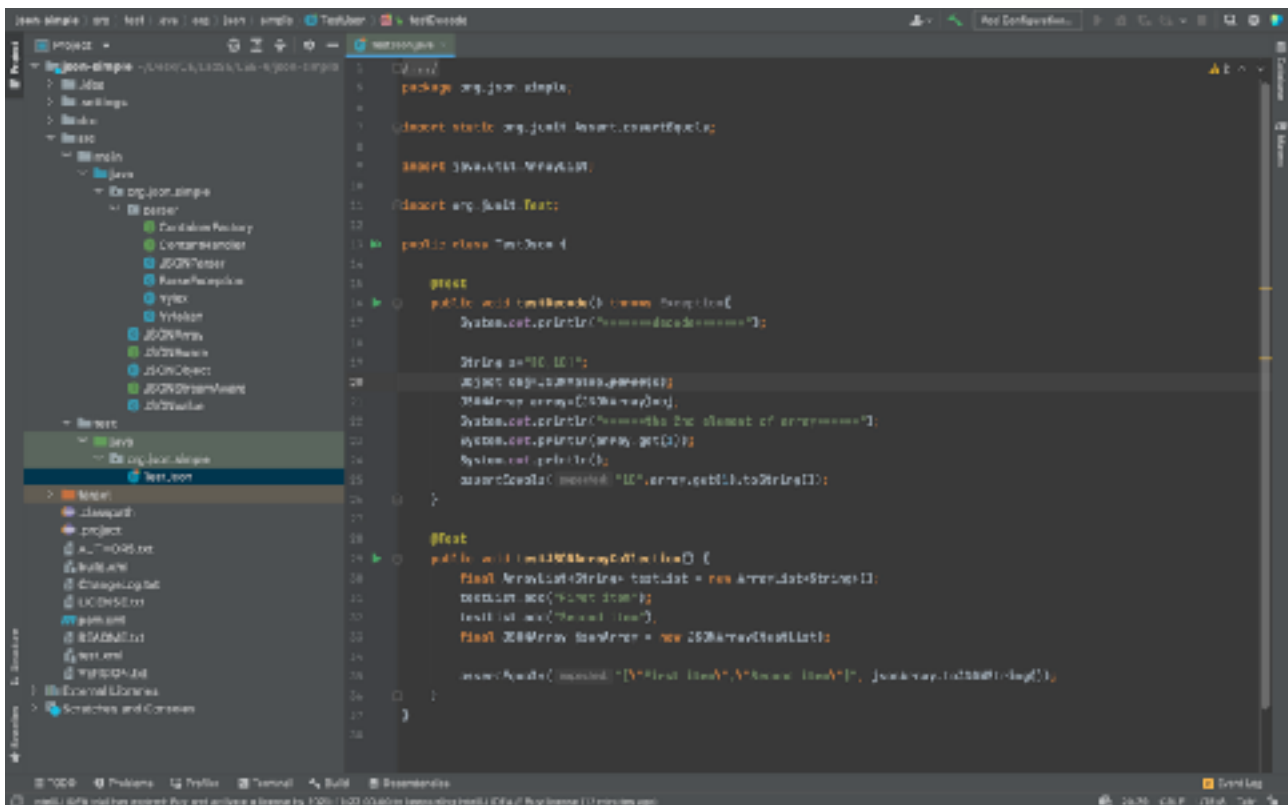
پاییز ۱۴۰۰

بخش اول: json-simple

طبق مستند دستور کار آزمایش، بخش روال انجام آزمایش، تمام مراحل را به درستی طی کردیم و مواردی که از ذکر شده بود اسکرین شات گرفتیم تا روند طی شده را بتوانید مشاهده فرمایید. گزارش Coverage Report در فولدر rep قابل مشاهده می باشد.

گیت برای این بخش جداگانه ساخته شده و آدرس ریپازیتوری آن به شرح زیر می باشد:

https://github.com/marinade۱۹۲۸/SElab۶_۱



بخش دوم: تمرین

IDE showing a Java project 'org.json.simple' with a coverage report. The report shows 100% class and line coverage for the package. The code in the center shows a public void method 'writeJSONString' that takes a JSONArray and a Writer as arguments.

Current scope: all classes | org.json.simple

Coverage Summary for Package: org.json.simple

| Package | Class, % | Method, % | Line, % |
|-----------------|------------|------------|------------|
| org.json.simple | 100% (1/1) | 100% (1/1) | 100% (1/1) |
| Class | Class, % | Method, % | Line, % |
| JSONWriter | 100% (1/1) | 100% (1/1) | 100% (1/1) |
| JSONObject | 0% (0/0) | 0% (0/0) | 0% (0/0) |
| JSONValue | 100% (1/1) | 100% (1/1) | 100% (1/1) |

generated at 2020-10-19 17:20

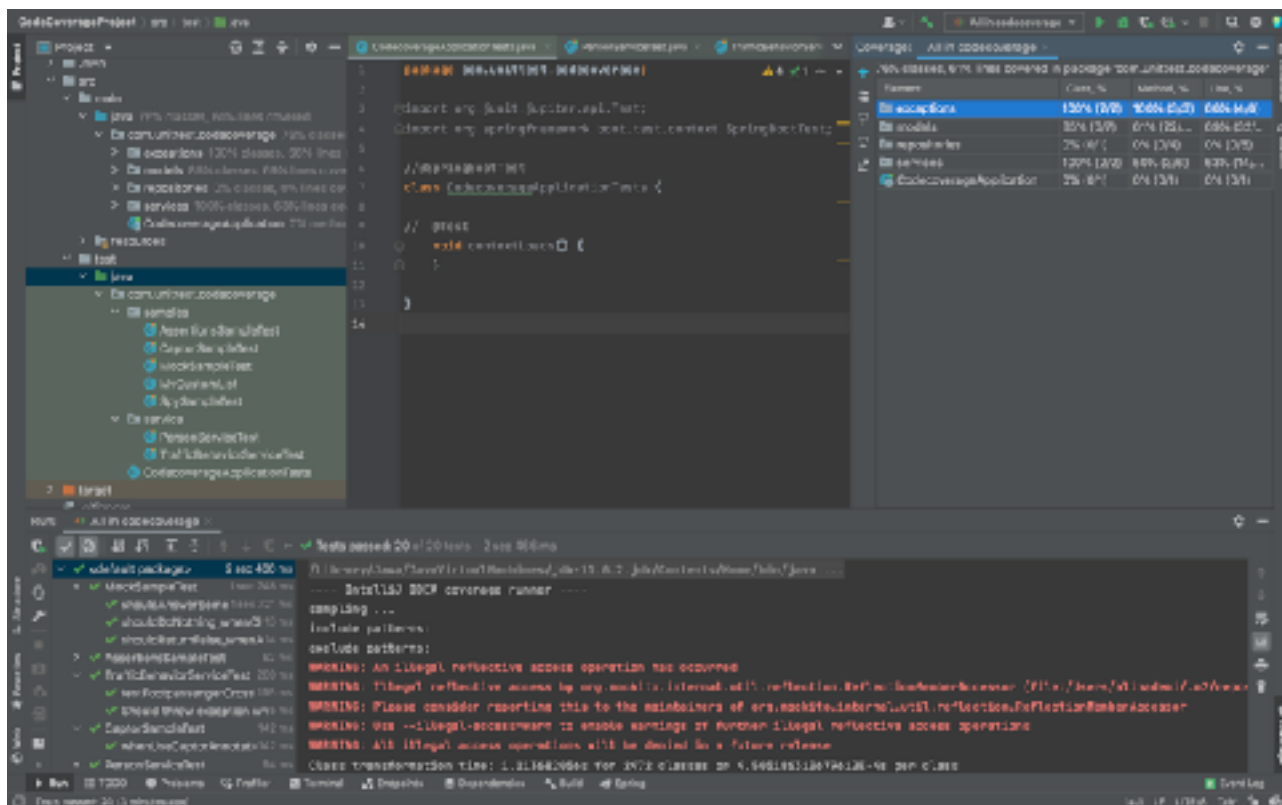
Source code of the 'writeJSONString' method. The code is highlighted with green and red colors, indicating coverage. The code is a public void method that takes a JSONArray and a Writer as arguments. It uses a switch statement to handle different JSON types: JSONArray, JSONObject, and String.

روالی که برای بخش اول طی شده را باید برای CodeCoverageProject نیز انجام دهیم.

گیت برای این بخش جداگانه ساخته شده و آدرس ریپازیتوری آن به شرح زیر می‌باشد:

<https://github.com/marinade1928/SELab6>

۱- اعداد پوشش آزمون:



گزارش Coverage Report در فولدر rep قابل مشاهده می‌باشد.

During scope: all classes

Overall Coverage Summary

| Package | Class, % | Method, % | Line, % |
|-------------|-------------|-------------|---------------|
| all classes | 78.8% (2/2) | 87.0% (2/2) | 83.2% (10/12) |

Coverage Breakdown

| Package | Class, % | Method, % | Line, % |
|---|------------|------------|--------------|
| com.unittest.codecoverage | 100% (2/2) | 100% (2/2) | 100% (14/14) |
| com.unittest.codecoverage.exceptions | 100% (1/1) | 100% (1/1) | 100% (1/1) |
| com.unittest.codecoverage.exceptions.exceptions | 100% (1/1) | 100% (1/1) | 100% (1/1) |
| com.unittest.codecoverage.exceptions.exceptions.exceptions | 100% (1/1) | 100% (1/1) | 100% (1/1) |
| com.unittest.codecoverage.exceptions.exceptions.exceptions.exceptions | 100% (1/1) | 100% (1/1) | 100% (1/1) |

generated on 2025-11-28 11:54

۲- کلاس‌هایی که میزان کاور آن نیاز به بهبود دارند، به شرح زیر می‌باشند: (بخش Line باید کاور خطوط بالا رود و بخش Methode باید کاور متدها بالا رود)

Line:

۱. PersonException

Methode:

۲. PersonValidator

۳. Footpassenger

۴. Person

۵. Traffic

۶. PersonRepository

۷. PersonServiceImpl

در کد تست، بخش‌هایی اضافه شده که منجر به افزایش نرخ کاور تمامی این کلاس‌ها شود. با توجه به لیست بالا مشاهده می‌کنیم که کلاس‌های با کاور ۱۰۰ درصد اصلاً ذکر نشده‌اند زیرا نیازی به بهبودی میزان کاور ندارند.

۲. PersonValidator & ۴. Person:

در کلاس Person، متد getAge و در کلاس PersonValidator متد requiredName را می‌خواهیم استفاده کنیم. (در بخش PersonServiceTest متد testGetAge_Person_and_testRequiredName_PersonValidator را برای کاور این دو متد ساخته‌ایم) کد این بخش به این صورت می‌باشد:

```
@Test
public void testGetAge_Person_and_testRequiredName_PersonValidator() {
    Person p1 = new Person();
    p1.setName("TestNameForP1");
    p1.setAge(23);
    Person p2 = new Person();
    p2.setName("TestNameForP2");
    p2.setAge(23);

    PersonValidator personValidator = new PersonValidator();
    personValidator.requiredName(p1.getName());
    personValidator.requiredName(p2.getName());

    assertEquals(p1.getAge(), p2.getAge());
}
```

۶. PersonRepository & ۱. PersonException:

در کلاس `PersonRepository`، متد `insert` و `get` را می‌خواهیم استفاده کنیم و در کلاس `PersonException` متد `PersonServiceTest` را استفاده کنیم. (در بخش `PersonServiceTest` متد `testInsert_PersonRepository_initialize_PersonException` را برای کاور این دو متد ساخته‌ایم) کد این بخش به این صورت می‌باشد:

```
@Test
public void testInsert_PersonRepository(){
    Person p1 = new Person();
    p1.setName("TestNameForP1");
    Person p2 = new Person();
    p2.setName("TestNameForP2");

    PersonRepository personRepository = new PersonRepository();
    personRepository.insert(p1);
    personRepository.insert(p2);

    PersonException personException = new PersonException("Err");

    assertEquals(personRepository.get(p1.getName()), personRepository.get(p2.getName()));
}
```

۵. Traffic:

در کلاس `Traffic`، متد `setCurrentTrafficLight` و `getCurrentTrafficLight` را می‌خواهیم استفاده کنیم. (در بخش `TrafficBehaviorServiceTest` متد `setAndGet_Traffic` را برای کاور این دو متد ساخته‌ایم) کد این بخش به این صورت می‌باشد:

```
@Test
public void setAndGet_Traffic(){
    Traffic traffic = new Traffic();
    traffic.setCurrentTrafficLight(TrafficLigth.GREEN);

    assertEquals(traffic.getCurrentTrafficLight(), TrafficLigth.GREEN);
}
```

۳. Footpassenger:

در کلاس `Footpassenger`، متد `setCrossedTheCrosswalk` و `crossedTheCrosswalk` را می‌خواهیم استفاده کنیم. (در بخش `TrafficBehaviorServiceTest` متد `setAndGet_Traffic` را برای کاور این دو متد ساخته‌ایم)
 کد این بخش به این صورت می‌باشد:

```
@Test
public void setAndGet_Footpassenger(){
    Footpassenger footpassenger = new Footpassenger();
    footpassenger.setCrossedTheCrosswalk(true);

    assertEquals(footpassenger.crossedTheCrosswalk(), actual: true);
}
```

V. PersonServiceImpl



در کلاس `PersonServiceImpl`، متد `get` را می‌خواهیم استفاده کنیم. (در بخش `PersonServiceTest` متد `testGet_PersonServiceImpl` را برای کاور این متد ساخته‌ایم)
 کد این بخش به این صورت می‌باشد:








```
@Test
public void testGet_PersonServiceImpl(){
    Person p1 = new Person();
    p1.setName("TestNameForP1");
    p1.setGender(Gender.F);
    p1.setAge(58);
    Person p2 = new Person();
    p2.setName("TestNameForP2");
    p2.setGender(Gender.M);

    PersonServiceImpl personServiceImpl = new PersonServiceImpl();
    personServiceImpl.insert(p1);
    personServiceImpl.insert(p2);

    assertEquals(personServiceImpl.get(p1.getName()), personServiceImpl.get(p2.getName()));
}
```

| Element | Class, % | Method, % | Line, % |
|--------------|------------|-------------|-------------|
| exceptions | 100% (2/2) | 100% (3/3) | 100% (6/6) |
| models | 85% (6/7) | 77% (28/... | 82% (38/... |
| repositories | 100% (1/1) | 50% (2/4) | 71% (5/7) |
| services | 100% (2/2) | 50% (3/6) | 63% (14/... |

| Element | Class, % | Method, % | Line, % |
|--|------------|------------|------------|
|  BehaviorException | 100% (1/1) | 100% (1/1) | 100% (1/1) |
|  PersonException | 100% (1/1) | 100% (2/2) | 100% (5/5) |

| Element | Class, % | Method, % | Line, % |
|--|------------|-------------|-------------|
|  validators | 100% (1/1) | 100% (4/4) | 100% (11... |
|  Footpassenger | 100% (1/1) | 100% (10... | 100% (11... |
|  Gender | 100% (1/1) | 100% (2/2) | 100% (2/2) |
|  Person | 100% (1/1) | 100% (6/6) | 100% (7/7) |
|  StreetDirectionFlow | 0% (0/1) | 0% (0/2) | 0% (0/2) |
|  Traffic | 100% (1/1) | 40% (4/10) | 45% (5/11) |
|  TrafficLigth | 100% (1/1) | 100% (2/2) | 100% (2/2) |

| Element | Class, % | Method, % | Line, % |
|--|------------|-----------|-----------|
|  PersonRepository | 100% (1/1) | 50% (2/4) | 71% (5/7) |

| Element | Class, % | Method, % | Line, % |
|---|------------|-----------|-------------|
|  impl | 100% (2/2) | 50% (3/6) | 63% (14/... |