

# دستور کار آزمایشگاه مهندسی نرم افزار

دانشگاه صنعتی شریف

دانشکده مهندسی کامپیوتر

## عنوان آزمایش: پوشش تست (Test Coverage)

### • اهداف

- آشنایی دانشجویان با مفهوم پوشش تست با کمک یک برنامه ساده

### • نیازمندی‌ها

- آشنایی با یک زبان برنامه‌نویسی مانند جاوا

### • ابزارهای مورد استفاده

- IntelliJ IDEA (ترجیحاً نسخه ۲۰۱۸ به بعد)
- JUnit
- Cucumber

### • روال انجام آزمایش

این آزمایش به مفاهیم پوشش تست (Test Coverage) می‌پردازد. به طور کلی در این آزمایش منظور از پوشش تست آن است که مشخص کنیم موارد آزمون (Test Cases) تا چه اندازه کد اصلی برنامه را مورد تست قرار می‌دهند و در واقع چند درصد از کد برنامه توسط این موارد، اجرا و تست می‌شوند. پوشش تست می‌تواند توسط معیارهایی نظیر تعداد کلاس‌ها، متدها و خطوط بررسی شود و به صورت خودکار از طریق ابزارهایی قابل محاسبه است.

۱. پروژه json-simple را در محیط IntelliJ باز (import) کنید. در صورتی که فایل‌های برنامه با خطا مواجه شدند و کتابخانه Test را شناسایی نکرده‌اند باید JUnit4 را اضافه کرد. SDK هم از قبل باید نصب شده باشد. ساختار پروژه شامل دو پکیج اصلی main و test است که حاوی کدهای برنامه و کد کلاس تست می‌باشد.

۲. کلاس تست TestJson شامل متدهایی جهت تست کلاس‌های اصلی است ولی این متدها همه کلاس‌ها را تست نمی‌کنند. برای یافتن اینکه چند درصد از کلاس‌ها، متدها و خطوط برنامه اصلی توسط TestJson مورد تست قرار می‌گیرند بر روی کلاس تست، کلیک راست کرده و گزینه Run TestJson with Coverage را انتخاب می‌کنیم. (اگر درصدهای Coverage نشان داده نشده است باید قسمت Edit Configuration را از پنل Coverage کلیک کرد و سپس در قسمت مربوط به Coverage گزینه مربوط به ذخیره خروجی Coverage را انتخاب کرد.)

The screenshot shows the IntelliJ IDEA interface with the 'Run TestJson with Coverage' option selected in the context menu. The coverage report for the package 'org.json.simple' is displayed on the right.

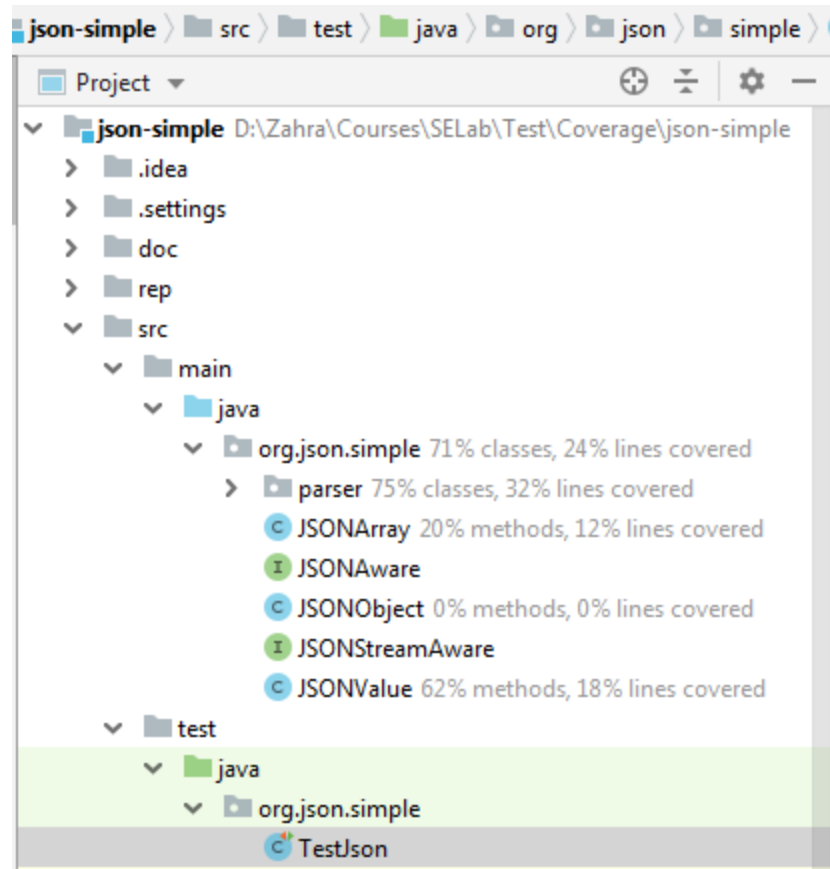
Element	Class, %	Method, %	Line, %
parser	75% (3/4)	40% (20/49)	32% (172/531)
JSONArray	100% (1/1)	20% (5/25)	12% (22/175)
JSONObject	0% (0/1)	0% (0/9)	0% (0/41)
JSONValue	100% (1/1)	62% (5/8)	18% (21/114)

Tests passed: 2 of 2 tests - 78 ms

```

78 ms include patterns:
31 ms org\.json\.simple\.*
47 ms exclude patterns:=====decode=====
=====the 2nd element of array=====
10

Class transformation time: 0.072974771s for 499 classes or 1.462420260521042E-4s per class
Process finished with exit code 0
  
```



Coverage: TestJson x

↑

Ant Build

Maven

Database

71% classes, 24% lines covered in 'all classes in scope'

Element	Class, %	Method, %	Line, %
<div> <div>org.json.simple</div> </div>	71% (5/7)	32% (30/91)	24% (215/861)

Coverage: TestJson x

71% classes, 24% lines covered in package 'org.json.simple'

Element	Class, %	Method, %	Line, %
parser	75% (3/4)	40% (20/49)	32% (172/531)
JSONArray	100% (1/1)	20% (5/25)	12% (22/175)
JSONObject	0% (0/1)	0% (0/9)	0% (0/41)
JSONValue	100% (1/1)	62% (5/8)	18% (21/114)

Ant Build  
Maven  
Database

برای اینکه با روال کار بهتر آشنا شوید، می‌توانید با گذاشتن breakpoint بر روی یکی از متدهای تست، روند debug آن و پیمایش خطوط برنامه را مشاهده کنید. با استفاده از گزینه Generate Coverage Report (فلش زردرنگ در تصویر فوق) می‌توان گزارشی به فرمت html حاوی خلاصه درصدهای پوشش، خطوط پوشش داده شده (با رنگ سبز) و خطوطی که پوشش داده نشده (خطوط قرمز) را مشاهده کرد.

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

Project: D:\Zahra\Courses\SELab\Test\Coverage

TestJson.java

```

20 Object obj=JSONValue.parse(s);
21 JSONArray array=(JSONArray)obj;
22 System.out.println("=====the 2nd element of");
23 System.out.println(array.get(1));
24 System.out.println();
25 assertEquals( expected: "10",array.get(1).toString());
26 }
27
28 @Test
29 public void testJSONArrayCollection() {
30     final ArrayList<String> testList = new ArrayList<>();
31     testList.add("First item");
32     testList.add("Second item");
33     final JSONArray array = new JSONArray(testList);
34     assertEquals("JSONArray Collection", array.get(1), "Second item");
35 }
36
37 }
38

```

Generate Coverage Report...

Output directory:  
TestCoverage\json-simple\rep

☒ Open generated HTML in browser

Save Cancel

Run: TestJson x

Tests passed: 2 of 2 tests - 78 ms

TestJson (org.json.simple)

testJSONArrayCollection 31 ms

testDecode 47 ms

include patterns:  
org\.json\.simple\.\*

exclude patterns:=====decode=====

=====the 2nd element of array=====

10

Class transformation time: 0.072974771s for 499 classes or 1.462420260521042E-4s per class

Process finished with exit code 0

Coverage Report :: Summary - Windows Internet Explorer

D:\Zahra\Courses\SELab\Test\Coverage\json-simple\rep\index.html

Search Bing

Favorites Suggested Sites Web Slice Gallery

Coverage Report :: Summary

[ all classes ]

### Overall Coverage Summary

Package	Class, %	Method, %	Line, %
all classes	71.4% (5/ 7)	33.3% (31/ 93)	25.1% (215/ 857)

### Coverage Breakdown

Package	Class, %	Method, %	Line, %
org.json.simple	66.7% (2/ 3)	23.3% (10/ 43)	13.2% (43/ 326)
org.json.simple.parser	75% (3/ 4)	42% (21/ 50)	32.4% (172/ 531)

generated on 2019-05-12 10:56

Computer Protected Mode: Off 105%

با کلیک بر روی لینک هر یک از فایل‌های گزارش، خطوط پوشش داده شده (به رنگ سبز) و نشده (به رنگ قرمز) نمایش داده می‌شوند.

```

    public static void writeJSONString(Collection collection, Writer out) throws IOException{
        if(collection == null){
            out.write("null");
            return;
        }

        boolean first = true;
        Iterator iter=collection.iterator();

        out.write('[');
        while(iter.hasNext()){
            if(first)
                first = false;
            else
                out.write(',');

            Object value=iter.next();
            if(value == null){
                out.write("null");
                continue;
            }

            JSONValue.writeJSONString(value, out);
        }
        out.write(']');
    }

    public void writeJSONString(Writer out) throws IOException{
        writeJSONString(this, out);
    }

```

### تمرین:

با افزودن بخش‌هایی به کد تست، اعداد پوشش تست را در مورد تمامی شش کلاس موجود در برنامه بهبود دهید. (برخی از کلاس‌ها در پوشه **parser** قرار گرفته‌اند). درصد افزایش اعداد پوشش تست مهم نیست ولی بخش‌هایی که به کد تست اضافه می‌شوند باید معنادار باشند و صرفاً یک فراخوانی ساده کلاس یا متد، بدون استفاده در بخش‌های دیگر کد کافی نیست.

همراه با ارسال کل پروژه، لازم است گزارشی در مورد بخش‌هایی که به کد تست اضافه کرده‌اید و میزان افزایشی که در اعداد پوشش تست به دنبال این تغییرات حاصل شده است را نیز تهیه و ارسال نمایید.

### • نحوه ارسال پروژه:

- ارسال کل پروژه به همراه گزارش از طریق سامانه CW