




KIỂM THỬ NÂNG CAO

BÀI 8: CÁC TÍNH NĂNG MỞ RỘNG CỦA TESTNG

- ☐ TestNG Result To Excel
- ☒ So sánh TestNg và Junit



Phần I: TestNG Result To Excel

-  Báo cáo bằng excel

-  Cấu hình xuất báo cáo ra excel

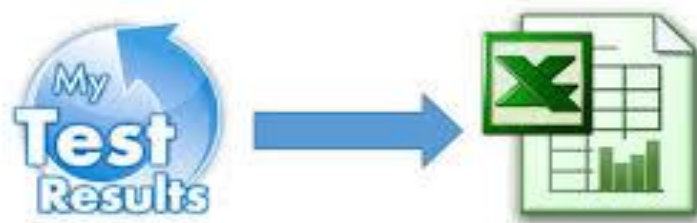
Phần II: So sánh TestNg và Junit

-  Ưu khuyết điểm

-  Các điểm khác nhau giữa junit và testng



- ❑ Xem kết quả thực thi test case trực quan, thân thiện
- ❑ Lựa chọn các tiêu chí, kết quả mong muốn khi xem báo cáo
- ❑ Dễ dàng tiếp cận, quản lý và tích hợp giữa các hệ thống



- ❑ Dùng thư viện "Selenium Webdriver standalone" (có thể download tại Seleniumhq.org)
- ❑ Dùng thư viện "Apache POI library" (có thể download tại <https://poi.apache.org/download.html>)

Selenium WebDriver



❑ Thêm thư viện vào Project kiểm thử

The screenshot shows the Eclipse IDE interface. The Package Explorer on the left displays the project structure: TestNGSeleniumDemo, src, com.techbeamers.testng, and SaveTestNGResultToExcel.java. The Build Path context menu is open, and the 'Add External Archives...' option is selected. The 'Referenced Libraries' section shows the added jars: selenium-server-standalone-2.53.0.jar and poi-3.14-20160307.jar. The main editor shows the SaveTestNGResultToExcel.java file with the following code:

```
package com.techbeamers.testng;

import static org.testng.AssertJUnit.assertEquals;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.Date;
import java.util.HashMap;
import java.util.Map;
import java.util.concurrent.TimeUnit;

import org.apache.poi.hssf.usermodel.HSSFSheet;
import org.apache.poi.hssf.usermodel.HSSFWorkbook;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Test;

public class SaveTestNGResultToExcel {

    public WebDriver driver;
    public UIMap uimap;
    public UIMap datafile;
    public String workingDir;

    // Declare An Excel Work Book
    HSSFWorkbook workbook;
    // Declare An Excel Work Sheet
    HSSFSheet sheet;
    // Declare A Map Object To Hold TestNG Results
    Map<String, Object[]> TestNGResults;
```

- ❑ Tạo tập tin lưu trữ các thông tin Test Data sẽ sử dụng, ví dụ tập tin có tên "datafile.properties" chứa các thông tin về địa chỉ website, user name và password...

```
url=http://phptravels.net/login  
username=fpoly@fpt.edu.vn.com  
password=demouser
```

- ❑ Tạo lớp kiểm thử sử dụng các test data đã được cung cấp, ví dụ cần kiểm thử đăng nhập thành công hay thất bại thì ta tạo 3 test case
 - ❖ Sử dụng test case truy xuất tới website cần login

```
@Test(description = "Opens the TestNG Demo Website for Login Test")
public void LaunchWebsite() throws Exception {

    try {
        driver.get("http://phptravels.net/login");
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(50,
TimeUnit.SECONDS);
        TestNGResults.put("2", new Object[] { 1d,
            "Navigate to demo website", "Site gets
opened", "Pass" });
    } catch (Exception e) {
        TestNGResults.put("2", new Object[] { 1d,
            "Navigate to demo website", "Site gets
opened", "Fail" });
    }
}
```


- ❑ Tạo lớp kiểm thử sử dụng các test data đã được cung cấp, ví dụ cần kiểm thử đăng nhập thành công hay thất bại thì ta tạo 3 test case
 - ❖ Cung cấp các thông tin login từ tập tin test data

```
@Test(description = "Fill the Login Details")
public void FillLoginDetails() throws Exception {

    try {

        // Get the username element
        WebElement username = driver.findElement(uimap
            .getLocator("Username_field"));
        username.sendKeys(datafile.getData("username"));

        // Get the password element
        WebElement password = driver.findElement(uimap
            .getLocator("Password_field"));
        password.sendKeys(datafile.getData("password"));

        TestNGResults.put("3", new Object[] { 2d,
            "Fill Login form data
(Username/Password)",
            "Login details gets filled", "Pass" });

    } catch (Exception e) {
        TestNGResults.put("3", new Object[] { 2d,
            "Fill Login form data
(Username/Password)",
            "Login form gets filled", "Fail" });
    }
}
```

- ❑ Tạo lớp kiểm thử sử dụng các test data đã được cung cấp, ví dụ cần kiểm thử đăng nhập thành công hay thất bại thì ta tạo 3 test case
 - ❖ Tiến hành chạy test case và cho ra kết quả

```
@Test(description = "Perform Login")
public void DoLogin() throws Exception {

    try {
        // Click on the Login button
        WebElement login = driver.findElement(uimap
            .getLocator("Login_button"));
        login.click();

        Thread.sleep(3000);
        // Assert the user login by checking the Online user
        WebElement onlineuser = driver.findElement(uimap
            .getLocator("online_user"));
        AssertJUnit.assertEquals("Hi, John Smith",
            onlineuser.getText());
        TestNGResults.put("4", new Object[] { 3d,
            "Click Login and verify welcome
            message", "Login success",
            "Pass" });
    } catch (Exception e) {
        TestNGResults.put("4", new Object[] { 3d,
            "Click Login and verify welcome
            message", "Login success",
            "Fail" });
    }
}
```

- ❑ Tạo phương thức xuất kết quả ra tập tin excel
- ❑ Sử dụng annotation <@AfterClass> kết hợp với phương thức TearDown()

```
@AfterClass
public void suiteTearDown() {
    // write excel file and file name is SaveTestNGResultToExcel.xls
    Set<String> keyset = TestNGResults.keySet();
    int rownum = 0;
    for (String key : keyset) {
        Row row = sheet.createRow(rownum++);
        Object[] objArr = TestNGResults.get(key);
        int cellnum = 0;
        for (Object obj : objArr) {
            Cell cell = row.createCell(cellnum++);
            if (obj instanceof Date)
                cell.setCellValue((Date) obj);
            else if (obj instanceof Boolean)
                cell.setCellValue((Boolean) obj);
            else if (obj instanceof String)
                cell.setCellValue((String) obj);
            else if (obj instanceof Double)
                cell.setCellValue((Double) obj);
        }
    }
}
```

❑ Xác định đường dẫn và tên file excel

```
        try {  
            FileOutputStream out = new FileOutputStream(new  
File("SaveTestNGResultToExcel.xls"));  
            workbook.write(out);  
            out.close();  
            System.out.println("Successfully saved Selenium WebDriver  
TestNG result to Excel File!!!");  
        } catch (FileNotFoundException e) {  
            e.printStackTrace();  
        } catch (IOException e) {  
            e.printStackTrace();  
        }  
    }
```

❑ Chạy phương thức kiểm thử và tạo file testing.xml

The screenshot illustrates the steps to generate a TestNG XML file from a Java class. The 'Refactoring' dialog is open, and the 'Generate testng.xml' checkbox is checked. The 'Location' is set to '/TestNGSeleniumDemo/testng.xml', the 'Suite name' is 'Suite', and the 'Test name' is 'Test'. The 'Preview' section shows the generated XML code, which includes the class name 'com.techbeamers.testng.SaveTestNGResultToExcel'.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Suite" parallel="none">
  <test name="Test">
    <classes>
      <class name="com.techbeamers.testng.SaveTestNGResultToExcel" />
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

❑ Cấu hình chạy lớp kiểm thử từ file testing.xml

```
File." ><?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://fpoly.edu.vn/testng-1.0.dtd">
<suite name="SaveTestNGResultToExcel" parallel="false">
  <test name="SaveTestNGResultToExcel">
    <classes>
      <class name="fpoly.edu.vn.testng.SaveTestNGResultToExcel">
        <methods>
          <include name="LaunchWebsite"></include>
          <include name="FillLoginDetails"></include>
          <include name="DoLogin"></include></methods>
        </class>
      </classes>
    </test> <!-- SaveTestNGResultToExcel -->
  </suite> <!-- SaveTestNGResultToExcel -->
```

❑ Thực thi và kiểm tra kết quả từ tập tin excel

The screenshot illustrates the process of running a TestNG suite and viewing the results in an Excel spreadsheet. In the IDE, the 'testng.xml' file is selected under 'test-output', and the 'Run As' button is highlighted. The 'TestNG Suite' is also highlighted in the 'Run As' dropdown menu. The 'SaveTestNGResultToExcel.xls' file is highlighted in the 'Resources' folder. The 'testng.xml' file content is shown, with imports for Selenium WebDriver and TestNG annotations. The Excel spreadsheet displays the test results, with the 'Test Step No.', 'Action', 'Expected Output', and 'Actual Output' columns highlighted. The test results show three steps, all of which passed.

Test Step No.	Action	Expected Output	Actual Output
1	Navigate to demo website	Site gets opened	Pass
2	Fill Login form data (Username/Password)	Login details gets filled	Pass
3	Click Login and verify welcome message	Login success	Pass



DEMO

Demo chạy TestNG xuất kết quả ra
Tập tin Excel





KIỂM THỬ NÂNG CAO

BÀI 8: CÁC TÍNH NĂNG MỞ RỘNG CỦA TESTNG (P2)

- ❑ JUnit 4 và TestNG đều là framework mạnh cho việc kiểm thử unit test
- ❑ Cả 2 đều có các tính năng tương tự nhau như annotation, para, group...

Is there any chance
JUnit 5 beat TestNG?



□ Các tính năng tương đồng

	JUnit4	TestNG
Annotation Support	✓	✓
Suite Test	✓	✓
Ignore Test	✓	✓
Exception Test	✓	✓
Timeout	✓	✓
Parameterized Test	✓	✓
Dependency Test	✗	✓

❑ Sử dụng các annotation đa dạng

- ❖ TestNG dùng @BeforeMethod , @AfterMethod giống như cách mà Junit dùng @Before , @After
- ❖ TestNG và Junit4 dùng @Test(timeout = 1000) cho thời gian timeout
- ❖ JUnit 4, khai báo “@BeforeClass” và “@AfterClass” là phương thức static, TestNG thì không

```
@BeforeClass
public static void oneTimeSetUp() {
    // one-time initialization code
    System.out.println("@BeforeClass - oneTimeSetUp");
}
```

```
@BeforeClass
public void oneTimeSetUp() {
    // one-time initialization code
    System.out.println("@BeforeClass - oneTimeSetUp");
}
```

- ❑ JUnit sử dụng @RunWith và @Suite để chạy các suite
- ❑ TestNG sử dụng file XML để định nghĩa test case sẽ chạy cùng nhau

```
package fpoly.junit;

import org.junit.runner.RunWith;
import org.junit.runners.Suite;

@RunWith(Suite.class)
@Suite.SuiteClasses({
    SuiteTest1.class,
    SuiteTest2.class,
})

public class JunitTest {
    // This class remains empty, it is used only as a holder for the above
    // annotations
}
```

```
<!DOCTYPE suite SYSTEM "http://fpply.edu.vn.com/testng"
<suite name="My test suite">
    <test name="testing">
        <classes>
            <class name="com.trangvt.TestNGTest1" />
            <class name="com.trangvt.TestNGTest2" />
        </classes>
    </test>
</suite>
```

❑ Ignore Test

❖ JUnit 4

```
@Ignore("Not Ready to Run")
@Test
public void divisionWithException() {
    System.out.println("Method is not ready yet");
}
```

❖ TestNG

```
@Test(enabled=false)
public void divisionWithException() {
    System.out.println("Method is not ready yet");
}
```

❑ Time Test

❖ *JUnit 4*

```
@Test(timeout = 1000)
public void infinity() {
    while (true);
}
```

❖ *TestNG*

```
@Test(timeOut = 1000)
public void infinity() {
    while (true);
}
```

❑ Parameterized Test

❖ *JUnit 4 sử dụng @RunWith và @Parameter*

```
@RunWith(value = Parameterized.class)
public class JunitTest6 {

    private int number;

    public JunitTest6(int number) {
        this.number = number;
    }

    @Parameters
    public static Collection<Object[]> data() {
        Object[][] data = new Object[][] { { 1 }, { 2 }, { 3 }, { 4 } };
        return Arrays.asList(data);
    }
}
```

❖ *TestNG sử dụng file XML hoặc @DataProvider*

```
public class Test1 {

    @Test
    @Parameters(value="number")
    public void parameterTest(int number)
    {
        System.out.println("Parameterized Number is : " + number);
    }

}
```


❑ Exception Test

❖ *JUnit 4 sử dụng @RunWith và @Parameter*

```
@Test(expected = ArithmeticException.class)
public void divideByZero()
{
    Int i = 1/0;
}
```

❖ *TestNG sử dụng file XML hoặc @DataProvider*

```
@Test(expectedExceptions = ArithmeticException.class)
public void divideByZero()
{
    Int i = 1/0;
}
```



DEMO

Demo phân biệt Junit và TestNG



Tổng kết bài học

📖 Phần I: TestNG Result To Excel

- 📖 Báo cáo bằng excel

- 📖 Cấu hình xuất báo cáo ra excel

📖 Phần II: So sánh TestNg và Junit

- 📖 Ưu khuyết điểm

- 📖 Các điểm khác nhau giữa junit và testng





KẾT THÚC