無料電子ブック

学習

testng

Free unaffiliated eBook created from
Stack Overflow contributors.

#testng

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: testng

It is an unofficial and free testng ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official testng.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

1: testngをいめる

このセクションでは、testngのと、がそれをいたいについてします。

また、testngのきなテーマについてもし、するトピックにリンクするがあります。 testngのドキュメンテーションはしいものなので、それらのトピックのバージョンをするがあります。

バージョン

バージョン	
1.0	2017-06-07

Examples

インストールまたはセットアップ

TestNGは、JDK 7をするがあります。

http://testng.org/doc/download.htmlによると、testngをインストールするには、あなたのmaven pom.xmlまたはgradle build.gradleファイルにtestngをするがあります

Maven

```
<repositories>
  <repository>
    <id>jcenter</id>
    <name>bintray</name>
    <url>http://jcenter.bintray.com</url>
  </repository>
</repositories>

<dependency>
  <groupId>org.testng</groupId>
  <artifactId>testng</artifactId>
  <version>6.9.12</version>
  <scope>test</scope>
</dependency>
```

Gradle

```
repositories {
    jcenter()
}

dependencies {
    testCompile 'org.testng:testng:6.9.12'
}
```

よりくのオプションはページでつけることができます 。

TestNGをったクイックプログラム

```
package example;

import org.testng.annotations.*; // using TestNG annotations

public class Test {

 @BeforeClass
 public void setUp() {
   // code that will be invoked when this test is instantiated
 }

 @Test(groups = { "fast" })
 public void aFastTest() {
   System.out.println("Fast test");
 }

 @Test(groups = { "slow" })
 public void aSlowTest() {
    System.out.println("Slow test");
 }

}
```

メソッド`setUp()`は、テストクラスがビルドされ、テストメソッドがされるにびされます。このでは、グループをでするので、 `aFastTest()`はスキップされますが、 `aSlowTest()`がびされます。

TestNG Hello Worldの

シンプルな`TestNG`プログラムのとは、に3ステップのプロセスです。

1. コード - テストのビジネスロジックをし、 TestNGでをける
2. - `testng.xml`または`build.xml`テストのをする
3. TestNGをする - コマンドライン、ANT、EclipseのようなIDE、IntelliJのIDEAからびすことができる

のなテストするがあるもの

たちは、4のPINをし`int`としてすメソッド`generateFourDigitPin`をつ`RandomNumberGenerator`クラスをっています。ここでは、そのが4のかどうかをテストしたいとします。はコードです

テストするクラス

```
package example.helloworld;

public class RandomNumberGenerator {

public int generateFourDigitPin(){
    return (int)(Math.random() * 10000);
}
```

```
}
```

TestNGテストクラス

```
package example.helloworld;

import org.testng.Assert;
import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Test;

public class TestRandomNumberGenerator {

    RandomNumberGenerator rng = null;

    @BeforeClass
    public void deSetup(){
        rng = new RandomNumberGenerator();
    }

    @Test
    public void testGenerateFourDigitPin(){
        int randomNumber = rng.generateFourDigitPin();
        Assert.assertEquals(4, String.valueOf(randomNumber).length());
    }

    @AfterClass
    public void doCleanup(){
        //cleanup stuff goes here
    }
}
```

Ther testng.xml

```
<suite name="Hello World">
    <test name="Random Number Generator Test">
        <classes>
            <class name="example.helloworld.TestRandomNumberGenerator" />
        </classes>
    </test>
</suite>
```

GradleでTestNGスイートをする

サンプル`build.gradle`ファイル

```
plugin: 'java'

repositories {
    mavenLocal()
    mavenCentral()
    jcenter()
}

dependencies {
    compile "org.testng:testng:6.9.12"
```

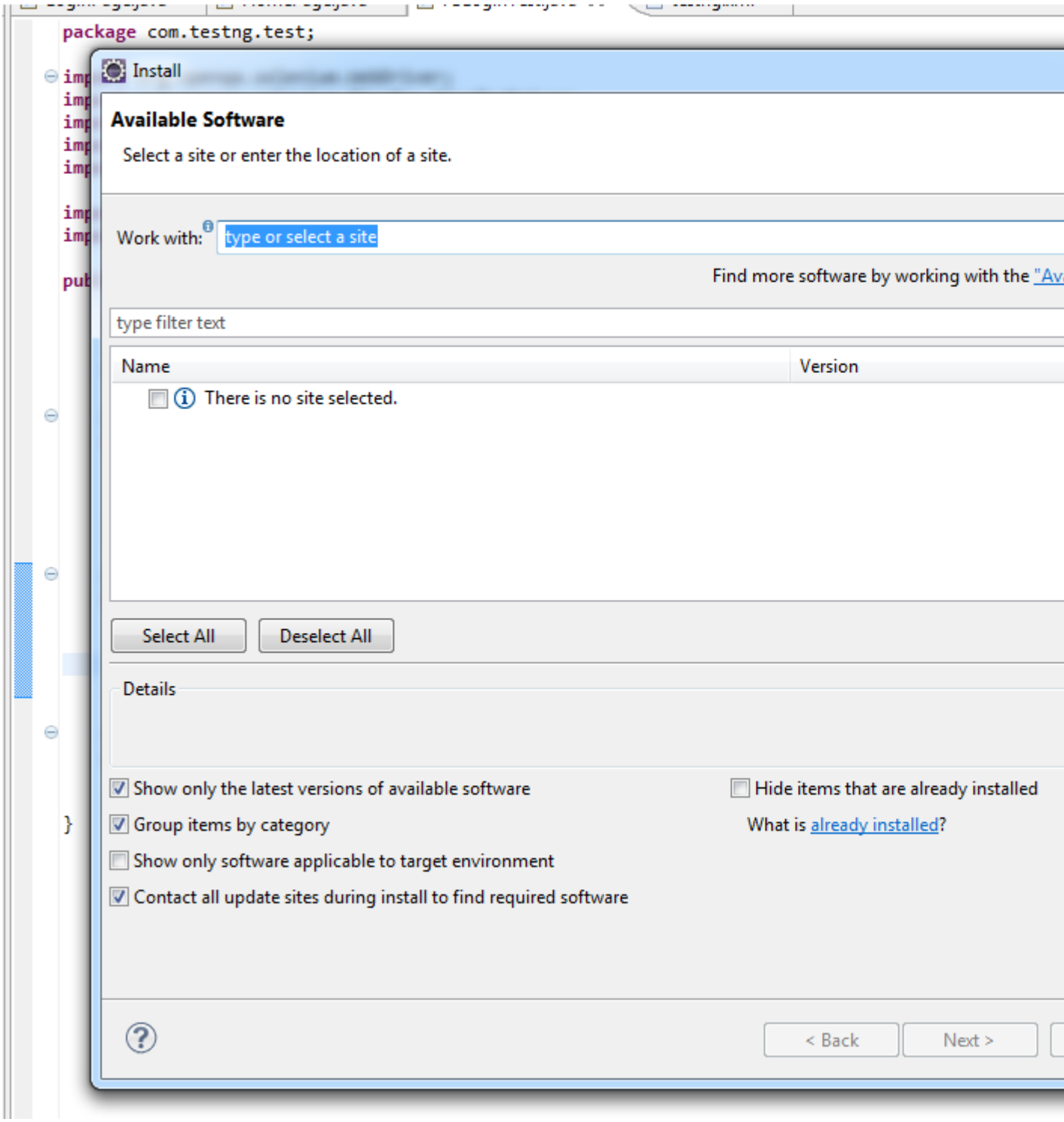
```
}

test {
    useTestNG() {
    suiteXmlBuilder().suite(name: 'Sample Suite') {
        test(name : 'Sample Test') {
            classes('') {
                'class'(name: 'your.sample.TestClass')
            }
        }
    }
}
```

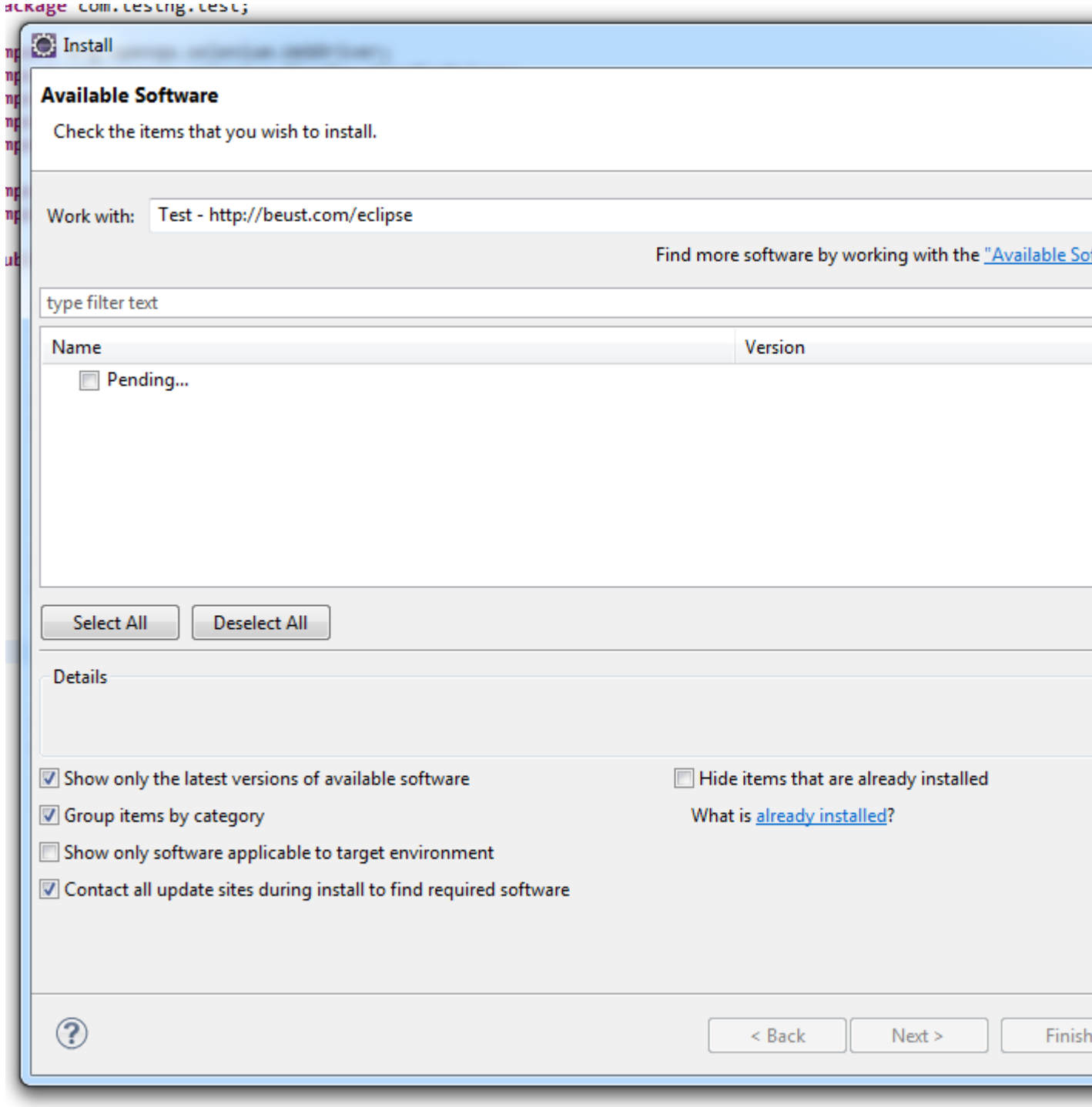
EclipseでTestNGをするとxmlをってテストをする

TestNGをEclipseにインストールする

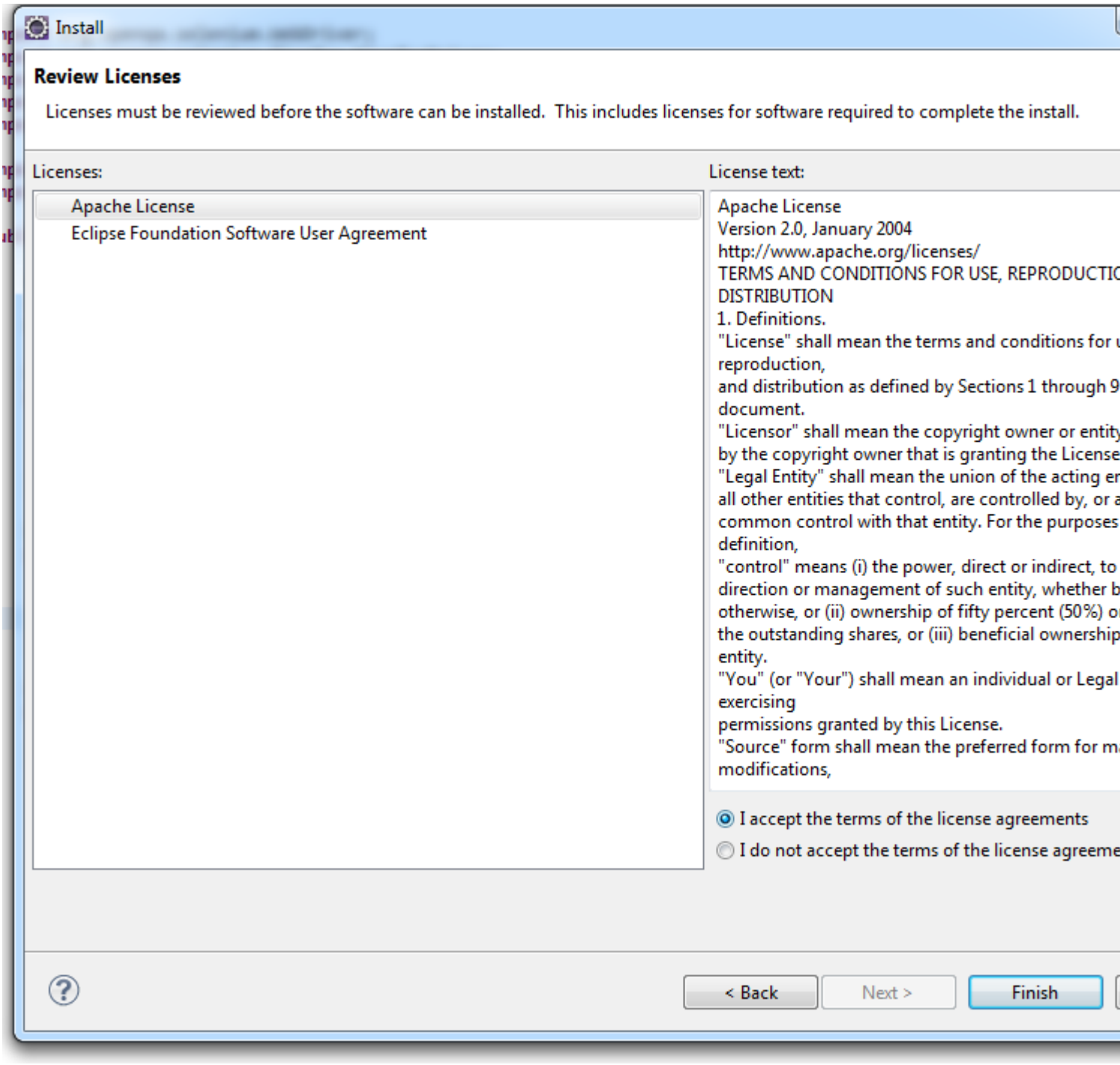
1. いた
2. [ヘルプ]> [しいソフトウエアをインストールする]をクリックします。



3. をクリックします
4. とURLをしてください - http://beast.com/eclipse



5. TestNGを
6. へをクリックします



7. Finishをクリックします。
8. TestNGをインストールするにはがかかります

インストールしてからEclipseをしてください。

TestNGプロジェクトをできます

1. ファイル>> Javaプロジェクト>をしてをクリック
2. TestNGClassとしてクラスをする
3. のクラスをする

1.LoginPage.class

2.HomePage.class

3.FBLoginTest.class

コードはのようになります。

LoginPageクラス

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class LoginPage {

    @FindBy(id = "email")
    private WebElement username;

    @FindBy(id = "pass")
    private WebElement password;

    @FindBy(xpath = ".//input[@data-testid='royal_login_button']")
    private WebElement login;

    WebDriver driver;

    public LoginPage(WebDriver driver){
        this.driver = driver;
         PageFactory.initElements(driver, this);
    }
    public void enterUserName(String name){
        username.clear();
        username.sendKeys(name);
    }

    public void enterPassword(String passwrd){
        password.clear();
        password.sendKeys(passwrd);
    }


    public HomePage clickLoginButton(){
        login.click();
        return new HomePage(driver);
    }
}
```

HomePageクラス。

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class HomePage {
```

```
    @FindBy(id = "userNavigationLabel")
    private WebElement userDropdown;

    WebDriver driver;

    public HomePage(WebDriver driver){
        this.driver = driver;
        PageFactory.initElements(driver, this);
    }

    public boolean isUserLoggedIn(){
        return userDropdown.isDisplayed();
    }

}
```

***FBLoginTest*クラス**

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

import com.testng.pages.HomePage;
import com.testng.pages.LoginPage;

public class FBLoginTest {

    WebDriver driver;
    LoginPage loginPage;
    HomePage homePage;

    @BeforeClass
    public void openFBPage(){
        driver = new FirefoxDriver();
        driver.get("https://www.facebook.com/");
        loginPage = new LoginPage(driver);
    }

    @Test
    public void loginToFB(){
        loginPage.enterUserName("");
        loginPage.enterPassword("");
        homePage = loginPage.clickLoginButton();
        Assert.assertTrue(homePage.isUserLoggedIn());
    }

    @AfterClass
    public void closeBrowser(){
        driver.quit();
    }

}
```

ここではtestng xmlがありますプロジェクトをクリックしてxmlファイルをし、このをコピーしてりけます。

```
<?xml version="1.0" encoding="UTF-8"?>
<suite name="Suite">
  <test name="Test">
    <classes>
      <class name="com.testng.FBLoginTest"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

セレンスタンドアロンジャーをする
のselenium standalone jarをダウンロードし、プロジェクトのビルドパスにします。

1. [プロジェクト]> [ビルドパス]> [ビルドパスの]> [ライブラリの]> [Jarsの]をクリックします
   。

TestNG xmlをするには xml> Run As> TestNGSuiteをクリックします。

ハッピーコーディング:)

オンラインでtestngをいめるをむ https://riptutorial.com/ja/testng/topic/5393/testngをいめる

2: @Test Annotation

- @テスト
- @Testattribute1 = attributeValue、atrribute2 = attributeValueなど

パラ メーター

パラ メーター	
alwaysRun	trueにすると、このテストメソッドは、したメソッドにしていてもにされます。
dataProvider	このテストメソッドのデータプロバイダの。
dataProviderClass	クラスは、データプロバイダをすです。されていない、データプロバイダは、のテストメソッドまたはそのクラスの1つのクラスをします。このをする、データプロバイダメソッドはされたクラスでであるがあります。
dependsOnGroups	このメソッドがするグループのリスト。
dependsOnMethods	このメソッドがするメソッドのリスト。
	このメソッドの。
	このクラス/メソッドのメソッドがかどうかをします。
expectedExceptions	テストメソッドがスローするとされるのリスト。このリストのまたはのがスローされない、このテストはとマークされます。
グループ	このクラス/メソッドがするグループのリスト。
invocationCount	このメソッドをびす。
invocationTimeOut	このテストがすべてのびしカウントのにするミリ。 invocationCountがされていない、このはされます。
	このテストメソッドの。いがにスケジュールされます。
	このでされる
singleThreaded	trueにすると、テストが`parallel="methods"`でされ`parallel="methods"`いても、このテストクラスのすべてのメソッドがじスレッドでされることがされます。このはクラスレベルでのみでき、メソッドレベルでされるはされます。 このは、びされていましたはされていません。

パラメータ	
タイムアウト	このテストでなミリ。
threadPoolSize	このメソッドのスレッドプールのサイズ。このメソッドは、invocationCountでされたのスレッドからびされます。invocationCountがされていない、このはされます。

Examples

@Testアノテーションのな

`@Test`アノテーションは、どのクラスやメソッドにもできます。このは、クラスまたはメソッドをテストのとしてマークします。

1. `@Test`メソッドレベルでテストする - きメソッドをテストメソッドとしてマークする
2. `@Test`クラスレベルでの`@Test`
   - クラスレベルの`@Test`アノテーションのは、クラスのすべてのパブリックメソッドををけなくてもテストメソッドにすることです。
   - `@Test`アノテーションは、のをするにもメソッドにしてりすことができます。

メソッドレベルでの**`@Test`**

```
import org.testng.annotations.Test;

public class TestClass1 {
    public void notTestMethod() {
    }

    @Test
    public void testMethod() {
    }
}
```

`@Test`クラスレベルでの

```
import org.testng.annotations.Test;

@Test
public class TestClass2 {
    public void testMethod1() {
    }

    @Test
    public void testMethod2() {
    }
}
```

オンラインで@Test Annotationをむ https://riptutorial.com/ja/testng/topic/6716/-test-annotation

3: TestNG -

Examples

TestNGテストAPIメソッドの

```
public class TestngAnnotation {
   // test case 1
   @Test
   public void testCase1() {
      System.out.println("in test case 1");
   }

   // test case 2
   @Test
   public void testCase2() {
      System.out.println("in test case 2");
   }

   @BeforeMethod
   public void beforeMethod() {
      System.out.println("in beforeMethod");
   }

   @AfterMethod
   public void afterMethod() {
      System.out.println("in afterMethod");
   }

   @BeforeClass
   public void beforeClass() {
      System.out.println("in beforeClass");
   }

   @AfterClass
   public void afterClass() {
      System.out.println("in afterClass");
   }

   @BeforeTest
   public void beforeTest() {
      System.out.println("in beforeTest");
   }

   @AfterTest
   public void afterTest() {
      System.out.println("in afterTest");
   }

   @BeforeSuite
   public void beforeSuite() {
      System.out.println("in beforeSuite");
   }

   @AfterSuite
   public void afterSuite() {
```

```
      System.out.println("in afterSuite");
   }

}
```

をするには、C> WORKSPACEにtestng.xmlというファイルをしてみましょう。

```
<suite name="Suite1">
  <test name="test1">
    <classes>
       <class name="TestngAnnotation"/>
    </classes>
  </test>
</suite>
```

C\ WORKSPACE> javac TestngAnnotation.java

に、testng.xmlをします。これは、されたTest Caseクラスでされたテストケースをします。

```
in beforeSuite
in beforeTest
in beforeClass
in beforeMethod
in test case 1
in afterMethod
in beforeMethod
in test case 2
in afterMethod
in afterClass
in afterTest
in afterSuite

===============================================
Suite
Total tests run: 2, Failures: 0, Skips: 0
===============================================
```

はのとおりです。

1. まず、 **beforeSuite**メソッドは1だけされます。
2. に、 **afterSuite**メソッドは1だけされます。
3. **beforeTest** 、 **beforeClass** 、 **afterClass** 、および**afterTest**メソッドもだけされます。
4. **beforeMethod**メソッドは、テストケースごとにされますが、テストケースをするにされます。
5. **afterMethod**メソッドは、テストケースごとにされますが、テストケースをしたにされます。
6. **beforeMethod**と**afterMethod**ので 、それぞれのテストケースがされます。

オンラインでTestNG - をむ https://riptutorial.com/ja/testng/topic/7889/testng----

4: TestNG グループ

- @Testgroups = {"group1"、 "group.regression"}、dependsOnGroups = {"group2"、"group3"}

Examples

TestNG グループのとな

グループは、 `testng.xml` `Suite`および/または`Test`でできます。 `tesng.xml`まれているとマークされたすべてのグループはとみなされ、された`tesng.xml`はされます。 `@Test`ののグループがされているのは、のグループをっており、それらのグループから`testng.xml`という`@Test`メソッドがされません。

のグループの`Test`レベルでのな`testng.xml`をに`testng.xml`ます。

```
<suite name="Suite World">
<test name="Test Name">
  <groups>
    <run>
      <include name="functest" />
      <exclude name="regtest" />
    </run>
  </groups>
  <classes>
    <class name="example.group.GroupTest"/>
  </classes>
</test>
</suite>
```

そして、これはテストクラスのようになります

```
package example.group;

import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Test;

public class GroupTest {

    @BeforeClass
    public void deSetup(){
        //do configuration stuff here
    }

    @Test(groups = { "functest", "regtest" })
    public void testMethod1() {
    }

    @Test(groups = {"functest", "regtest"} )
    public void testMethod2() {
    }
```

```
    @Test(groups = { "functest" })
    public void testMethod3() {
    }

    @AfterClass
    public void cleanUp(){
        //do resource release and cleanup stuff here
    }
}
```

このGroupTest TestNGクラスをすると、 testMethod3()のみがされます。

- <include name="functest" /> functestグループのすべてのテストメソッドは、のグループからされていなければです。
- <exclude name="regtest" /> regtestグループのテストメソッドはregtestできません。
- testMethod1()とtestMethod2()はregtestグループにあるため、されません。
- testMethod3()はregtestグループにあり、されます。

TestNG メタ グループ - グループの グループ

TestNGでは、のグループをむグループをすることができます。メタグループは、1つまたはのグループをにし、それらのグループにする@Testメソッドのをします。

のでは、なるグループにするさまざまな@Testメソッドがあります。のスタックにのものはほとんどなく、テストとけれテストはほとんどありません。ここでメタグループをできます。 2つのなメタ グループをんでみましょう

1. allstack - liux.jboss.oracleとaix.was.db2のグループをみ、これらのグループのいずれかにするすべてのテストメソッドをにできるようにします。
2. systemtest - allstack 、 regressionおよびacceptanceグループをみ、これらのグループのいずれかにするすべてのテストメソッドをにできるようにします。

testng.xml

```
<suite name="Groups of Groups">
    <test name="MetaGroups Test">
        <groups>
            <!-- allstack group includes both liux.jboss.oracle and aix.was.db2 groups -->
            <define name="allstack">
                <include name="liux.jboss.oracle" />
                <include name="aix.was.db2" />
            </define>

            <!-- systemtest group includes all groups allstack, regression and acceptance -->
            <define name="systemtest">
                <include name="allstack" />
                <include name="regression" />
                <include name="acceptance" />
            </define>

            <run>
```

```
                <include name="systemtest" />
            </run>
        </groups>

        <classes>
            <class name="example.group.MetaGroupsTest" />
        </classes>
    </test>

</suite>
```

MetaGroupsTestクラス

```
package example.group;

import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

public class MetaGroupsTest {

    @BeforeMethod
    public void beforeMethod(){
        //before method stuffs - setup
    }

    @Test(groups = { "liux.jboss.oracle", "acceptance" })
    public void testOnLinuxJbossOracleStack() {
        //your test logic goes here
    }

    @Test(groups = {"aix.was.db2", "regression"} )
    public void testOnAixWasDb2Stack() {
        //your test logic goes here
    }

    @Test(groups = "acceptance")
    public void testAcceptance() {
        //your test logic goes here
    }

    @Test(groups = "regression")
    public void testRegression(){
        //your test logic goes here
    }

    @AfterMethod
    public void afterMthod(){
        //after method stuffs - cleanup
    }
}
```

オンラインでTestNGグループをむ https://riptutorial.com/ja/testng/topic/5821/testngグループ

5: パラメータされたテスト

Examples

データプロバイダ

データプロバイダをすると、のテストをしてテストでできます。がしくされていることをするテストをえてみましょう。データプロバイダがすメソッドをするいずれかの`Object[][]`または`Iterator<Object[]>`ではテストのをにする`@DataProvider`プロパティと、アノテーション`name`するのでありますプロバイダ。

```
import org.testng.annotations.DataProvider;

public class DoublingDataProvider {
    public final static String DOUBLING_DATA_PROVIDER = "doublingDataProvider";

    @DataProvider(name = DOUBLING_DATA_PROVIDER)
    public static Object[][] doubling() {
        return new Object[][]{
                new Object[]{1, 2},
                new Object[]{2, 4},
                new Object[]{3, 6}
        };
    }
}
```

の、`Object[]`は1つのテストケースのデータセットをします。ここではを、いて2したのをします。

データプロバイダをするには、テストの`dataProvider`プロパティにプロバイダのをします。プロバイダメソッドがテストクラスまたはそのクラスのでされているは、`dataProviderClass`プロパティもするがあります。テストメソッドは、テストケースのにするパラメータをるがあります。ここでは2つのintです。

```
import org.testng.annotations.Test;

import static org.testng.Assert.assertEquals;

public class DoublingTest {

    @Test(dataProvider = DoublingDataProvider.DOUBLING_DATA_PROVIDER, dataProviderClass =
DoublingDataProvider.class)
    public void testDoubling(int number, int expectedResult) {
        assertEquals(number * 2, expectedResult);
    }
}
```

オンラインでパラメータされたテストをむ https://riptutorial.com/ja/testng/topic/5684/パラメータされたテスト

クレジット

S. No		Contributors
1	testngをいめる	Atul Dwivedi, Community, Idos, mackowski, RocketRaccoon, Sudha Velan
2	@Test Annotation	Atul Dwivedi, Benoit
3	TestNG -	Shrikant
4	TestNGグループ	Atul Dwivedi
5	パラメータされたテスト	Benoit, mszymborski