接口测试自动化

引入测试框架

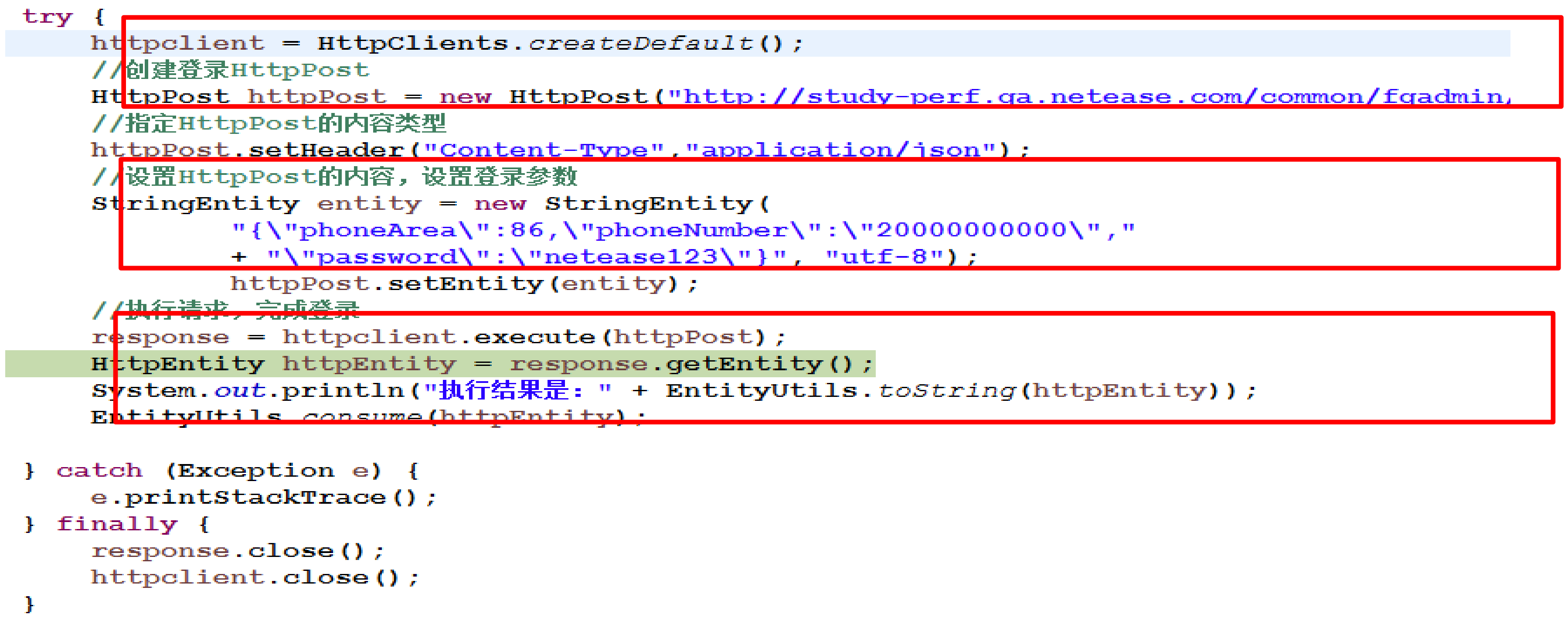
本章大纲

- 接口自动化概述
- HttpClient
- 引入测试框架
- 接口测试结果验证
- 面向场景的接口自动化测试

目前完成的测试用例

用例标题
Get请求的用例
Post请求的用例
带登录信息的Post请求用例

目前完成的测试用例



```
try {
    httpclient = HttpClients.createDefault();
    //创建登录HttpPost
    HttpPost httpPost = new HttpPost("http://study-perf.qa.netease.com/common/fgadmin,
    //指定HttpPost的内容类型
    httpPost.setHeader("Content-Type","application/json");
    //设置HttpPost的内容，设置登录参数
    StringEntity entity = new StringEntity(
            "{\"phoneArea\":86,\"phoneNumber\":\"20000000000\","
            + "\"password\":\"netease123\"}", "utf-8");
            httpPost.setEntity(entity);
    //执行请求，完成登录
    response = httpclient.execute(httpPost);
    HttpEntity httpEntity = response.getEntity();
    System.out.println("执行结果是： " + EntityUtils.toString(httpEntity));
    EntityUtils.consume(httpEntity);

} catch (Exception e) {
    e.printStackTrace();
} finally {
    response.close();
    httpclient.close();
}
```

距离真正的自动化测试还缺少什么？

- 测试验证点

  目前的用例只是自动化操作的积累，没有测试验证

- 用例集的组织

  目前的用例只能单个、手动触发执行

- 测试报告

  目前测试结果只能以无错误、错误异常堆栈日志形式展示

距离真正的自动化测试还缺少什么？

- 测试验证点
- 用例集的组织
- 测试报告

测试框架：TestNG

TestNG简介

- The next generation of unit testing
- Cedric Beust
- http://testng.org/doc/
- 基于Junit、Nunit并支持注解、数据驱动、多线程执行等特性的Java测试框架

TestNG基础：注解（Annotation）

- JDK5引入JAVA，TestNG用以方便的标注测试方法和组件

@Test 标注测试方法

@BeforeTest 标注全部测试方法执行前需要执行的方法

@AfterClass 标注测试类全部方法执行之后需执行的方法

@DataProvider 数据驱动

注意：TestNG执行测试方法之前，都会重新实例化测试类

```
......@Test
      public void testMethod() {
                  System.out.println("hello");
      }
```

TestNG基础：断言（Assert）

org.testng.Assert

fail 直接失败测试用例

assertTrue	判断是否为true
assertNull	判断是否为null
assertEquals	判断是否相等

```
@Test
public void testMethod() {
        System.out.println("hello");
        int result =new Caculator().add(1, 2);
        Assert.assertEquals(result, 3);
}
```

TestNG基础：Test属性使用

- 给测试方法增加分组属性

```
public class Test1 {
  @Test(groups = { "functest", "checkintest" })
  public void testMethod1() {
  }

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

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

- 给测试方法增加依赖关系

```
@Test(dependsOnMethods = { "serverStartedOk" })
public void method1() {
}
```

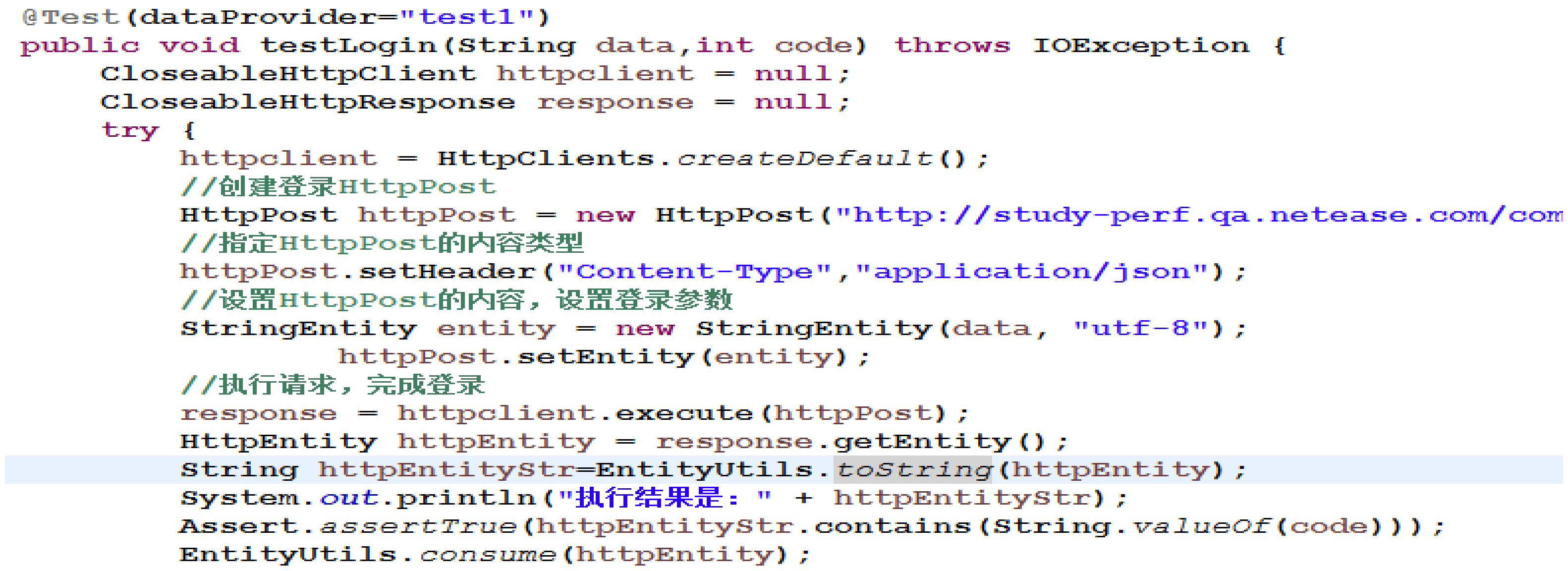
TestNG基础：@DataProvider

- 使用数据驱动，复用测试方法

```
@DataProvider(name = "test1")
public Object[][] createData1() {
 return new Object[][] {
        {"{\"phoneArea\":\"86\",\"phoneNumber\":\"20000000000\","
      + "\"password\":\"netease123\"}", 200},
        { "{\"phoneArea\":86,\"phoneNumber\":\"20000000000\","
      + "\"password\":\"netease123\"}", 400},
        { "{\"phoneArea\":\"86\",\"phoneNumber\":\"12345\","
      + "\"password\":\"netease123\"}", 400}};
}
```

TestNG基础：DataProvider属性使用

- 使用数据驱动，复用测试方法



```
@Test(dataProvider="test1")
public void testLogin(String data,int code) throws IOException {
    CloseableHttpClient httpclient = null;
    CloseableHttpResponse response = null;
    try {
        httpclient = HttpClients.createDefault();
        //创建登录HttpPost
        HttpPost httpPost = new HttpPost("http://study-perf.qa.netease.com/com
        //指定HttpPost的内容类型
        httpPost.setHeader("Content-Type","application/json");
        //设置HttpPost的内容，设置登录参数
        StringEntity entity = new StringEntity(data, "utf-8");
                httpPost.setEntity(entity);
        //执行请求，完成登录
        response = httpclient.execute(httpPost);
        HttpEntity httpEntity = response.getEntity();
        String httpEntityStr=EntityUtils.toString(httpEntity);
        System.out.println("执行结果是： " + httpEntityStr);
        Assert.assertTrue(httpEntityStr.contains(String.valueOf(code)));
        EntityUtils.consume(httpEntity);
```

TestNG基础：testng.xml用例集

- suite-test-class/groups-class-method

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >
  <suite name="Suite1" verbose="1" >
  <test name="Nopackage" >
    <classes>
       <class name="NoPackageTest" />
    </classes>
  </test>
   <test name="Regression1">
    <classes>
      <class name="test.sample.ParameterSample"/>
      <class name="test.sample.ParameterTest"/>
    </classes>
  </test>
</suite>
```

TestNG基础：testng.xml用例集

- 指定测试类

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >
  <suite name="Suite1" verbose="1" >
  <test name="Nopackage" >
    <classes>
       <class name="NoPackageTest" />
    </classes>
  </test>

  </suite>
```

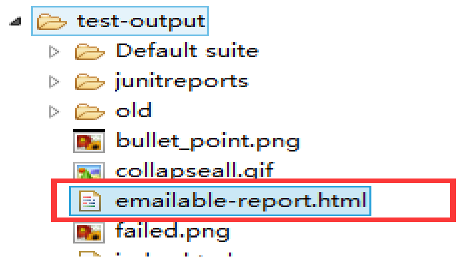
TestNG基础：testng.xml用例集

- 执行某个分组的全部测试

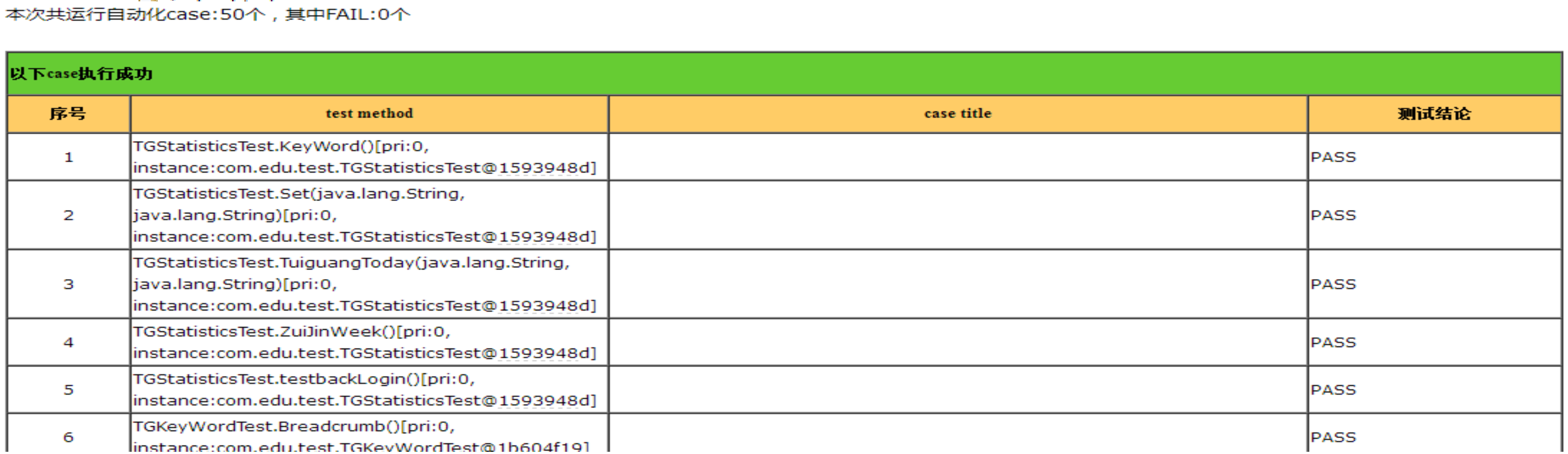
```
 <test name="Regression1">
 <groups>
   <run>
     <exclude name="brokenTests"  />
     <include name="checkinTests"  />
   </run>
 </groups>

 <classes>
   <class name="test.IndividualMethodsTest">
     <methods>
       <include name="testMethod" />
     </methods>
   </class>
 </classes>
</test>
```

TestNG基础：测试报告

建议使用FreeMarker来生成新的测试报告



```
◢ test-output
  ▷ Default suite
  ▷ junitreports
  ▷ old
    bullet_point.png
    collapseall.gif
    emailable-report.html
    failed.png
```



本次共运行自动化case:50个，其中FAIL:0个

以下case执行成功

序号	test method	case title	测试结论
1	TGStatisticsTest.KeyWord()[pri:0, instance:com.edu.test.TGStatisticsTest@1593948d]		PASS
2	TGStatisticsTest.Set(java.lang.String, java.lang.String)[pri:0, instance:com.edu.test.TGStatisticsTest@1593948d]		PASS
3	TGStatisticsTest.TuiguangToday(java.lang.String, java.lang.String)[pri:0, instance:com.edu.test.TGStatisticsTest@1593948d]		PASS
4	TGStatisticsTest.ZuiJinWeek()[pri:0, instance:com.edu.test.TGStatisticsTest@1593948d]		PASS
5	TGStatisticsTest.testbackLogin()[pri:0, instance:com.edu.test.TGStatisticsTest@1593948d]		PASS
6	TGKeyWordTest.Breadcrumb()[pri:0, instance:com.edu.test.TGKeyWordTest@1b604f19]		PASS