模拟用户访问网页-技术预研报告

广东宜通世纪科技股份有限公司
刘勇(http://atian25.iteye.com) 2011年02月

我们遇到了什么问题?

- 移动互联网时代，随着运营商的快速发展，相应的业务服务网站越来越多。
- 网站之间的差异性越来越大。页面流程越来越复杂，不再仅仅是访问单个页面。
- 一个典型的业务流程：

目的

- 用户通过WLAN接入CMCC
- 打开浏览器，访问百度首页

认证

- 被重定向至CMCC的Portal登录页面
- 用户输入帐号和密码，点击登陆
- 登录成功后系统自动跳转之前的页面，即百度首页

访问

- 用户在百度搜索框输入关键词，点击查询
- 用户查看结果页面，统计符合条件的结果数
- 用户点击某个链接下载文件，或截图。

传统方案

- **使用wget/curl模拟HTTP协议**
  -- 基于C语言，相关开源类库太少。
  -- 自己造轮子，底层指令需要做很多封装来实现上层应用。
  -- 不支持Socket v5代理等特性。

- **使用正则表达式分析页面**
  -- 非主流方式，正则式不适合于分析复杂的页面。

- **业务流程逻辑固化在程序里**
  -- 耦合性太大开发效率低，响应速度慢，面对复杂流程业务力不从心。
  （之前一个新的portal页面登录需要5人/日！！！）

方案1: Selenium

- 主流的Web自动集成测试工具。
  - http://seleniumhq.org
  - http://code.google.com/p/selenium/
  - 支持多种操作系统：Windows，Linux，Mac，Android…
    - Android版预研，具体参见刘伟杰的预研报告。
    - http://code.google.com/p/selenium/wiki/AndroidDriver
  - 支持模拟多种浏览器：IE，Firefox，Chrome，Android…

- 目前提供2个版本:
  - Selenium IDE:

    Firefox插件, 可以在录制用户操作脚本, 运行测试，脚本可以导出。

  - Selenium RC:
    - 可以用具体语言来写业务测试流程。
    - **支持多种语言：java，c#，python，ruby，php，perl，js**
    - **可以直接使用IDE导出的脚本！！！**

Selenium代码示例

- http://code.google.com/p/selenium/wiki/GettingStarted

```java
package org.openqa.selenium.example;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.htmlunit.HtmlUnitDriver;

public class Example  {
    public static void main(String[] args) {
        // Create a new instance of the html unit driver
        // Notice that the remainder of the code relies on the interface,
        // not the implementation.
        WebDriver driver = new HtmlUnitDriver();

        // And now use this to visit Google
        driver.get("http://www.google.com");

        // Find the text input element by its name
        WebElement element = driver.findElement(By.name("q"));

        // Enter something to search for
        element.sendKeys("Cheese!");

        // Now submit the form. WebDriver will find the form for us from the element
        element.submit();

        // Check the title of the page
        System.out.println("Page title is: " + driver.getTitle());
    }
}
```

方案2: CasperJS

- 基于QTWebKit，类库为C++，开发语言使用JS。
  - http://casperjs.org/index.html
  - http://www.phantomjs.org/
- 优点：
  - 我们可以把它看作是无界面的webkit浏览器，它速度相当快。
  - 原生支持多种web标准，如DOM，CSS选择符，JSON，SVG…
  - 支持页面截图，网络爬虫，页面登陆等高级别的操作。
  - 事件机制，编程简单，程序高效。
- 缺点:
  - 目前还是需要安装xvfb。
  - 下一个版本（预计3月中旬），将升级为QT4.8，届时不需要xvfb。

CasperJS代码示例

- 访问禅道
- 自动跳转登录页
- 登录验证
- 下载文件，截图
- 统计耗时

```
//未登录的时候访问指定页面
casper.echo('目标页面:'+targetUrl);
casper.start(targetUrl, function(){
    casper.viewport(1024, 768);
    //Step1: 验证是否会自动跳转到登录页面
    if(this.getTitle().indexOf('用户登录')==-1){
        this.die('未跳转到登录界面,实际访问页面:'+this.getCurrentUrl());
    }else{
        this.echo('成功跳转到登录界面:'+this.getCurrentUrl());
        this.echo(utils.format('Portal跳转时延:%dms', (new Date().getTime() - startTime)));
    }
    this.capture('step1.png');//截图

    //Step2: 自动登录
    this.echo(utils.format('使用[%s / %s]开始登录...', account, pwd));
    this.fill('form', {
        'account': account,
        'password': pwd
    }, true);
    this.click('input#submit');
    this.capture('step2.png');//截图

    //Step3: 判断登录是否成功
    var timeout = 3000;
    this.echo(utils.format('检测中...等待登陆后跳转页面,期望标题:%s.', targetTitleRegex));
    this.waitFor(function() {   //等待页面提交完毕
        return this.getTitle().indexOf(targetTitleRegex)!=-1;
    }, function() {
        this.echo('成功登录,并跳转到目标界面:'+this.getCurrentUrl());
        this.capture('step3_loginSuc.png');
    }, function() {    //超时处理
        this.capture('step3_loginTimeout.png');
        this.die(utils.format('等待页面载入超时:%dms', timeout));
    }, timeout);
```

其他方案

- 类库还有很多
  - node.io
  - jsdom
  - request
  - needle
  - httpclient (java)
  - …
- 特点：
  - 这些类库基本上都只是实现了HTTP协议，或多或少支持多个URL并发，或cookie等特性，仅有一些部分支持JS。
  - 缺点是页面跳转（如304或window.location）需要开发者自己判断实现，相关的资源（img，css，js）也要开发者自行发起下载请求，仅有一些部分支持JS，对于重JS的页面（EXTJS）模拟访问很麻烦。

目前在WLAN项目中的使用情况

- 使用Selenium HtmlUnitDriver
  - 基于命令行下运行的考虑。
  - ChromeDriver等需要弹出浏览器窗口。
  - **注：经预研，通过xvfb可以解决该问题。**
    - xvfb（在命令行下实现对X-server的模拟，渲染图形进行缓存）-在没有安装X-Server的环境下提供图像渲染
    - http://www.labelmedia.co.uk/blog/posts/setting-up-selenium-server-on-a-headless-jenkins-ci-build-machine.html
    - http://www.alittlemadness.com/2008/03/05/running-selenium-headless/
- 现存的问题点：
  - 页面的跳转需要程序判断，不能自动化。无法适应复杂业务。
  - 暂未找到监听资源下载进度的接口，需人为设置等待时间，不科学。
  - 无法设置指令超时时间。

总结

- Selenium -- 业界主流方案
  - 建议更换HtmlUnitDriver为ChromeDriver/IEDriver等驱动,以便有更丰富的API接口，并可以自动跳转。
  - 使用xvfb界面浏览器弹出问题。（非基于QTWebkit，不能用QT4.8）
  - 需进一步预研资源下载监控的接口。
  - 应对多个Portal页面/多个业务网站：
    - 使用病毒库+学习的机制
    - 维护人员使用Selenium IDE录制脚本，导入脚本库，后台直接调用。
- CasperJS – 热门新宠儿
  - 基于QTWebkit，编程脚本使用javascript。
  - 提供的接口比较人性化，提供丰富的事件通知。
  - 可以提供HTTP协议接口或通过命令行，供JAVA程序调用。
  - 可以配置NodeJS等新兴技术，爬虫必备。
- **鉴于目前的架构已使用Selenium，建议采用并进一步预研。**

感谢～

您的聆听！