PHP Selenium 使用教程

wangking717

目　录

快速开始

> Selenium是用于自动化测试工具，它是在Apache许可证2.0许可的开放源代码工具。Selenium是一套工具，它有助于自动化Web应用程序测试。
> > 框架底层使用JavaScript模拟真实用户对浏览器进行操作。测试脚本执行时，浏览器自动按照脚本代码做出点击，输入，打开，验证等操作，就像真实用户所做的一样，从终端用户的角度测试应用程序。
> > 使浏览器兼容性测试自动化成为可能，尽管在不同的浏览器上依然有细微的差别。
> >* 使用简单，可使用Java，Python等多种语言编写用例脚本。

1.通过composer安装Selenium：

```
composer require facebook/webdriver
```

2.下载Selenium Server并启动：

```
//到http://www.seleniumhq.org/download/ 下找到Selenium Standalone Server并下载，或到https://chromedriver.storage.googleapis.com/index.html 上面下载。
//到命令提示符里启动以下命令（前提需要安装jdk，确保java命令能够运行）
java -jar selenium-server-standalone-2.42.2.jar
```

3.运行测试代码

> 另启命令提示符，运行php test.php 命令
> 示例脚本test.php：

```
<?php
namespace Facebook\WebDriver;
use Facebook\WebDriver\Remote\DesiredCapabilities;
use Facebook\WebDriver\Remote\RemoteWebDriver;
require_once('vendor/autoload.php');
```

header("Content-Type: text/html; charset=UTF-8");
// start Firefox with 5 second timeout
$waitSeconds = 15; //需等待加载的时间，一般加载时间在0-15秒，如果超过15秒，报错。
$host = 'http://localhost:4444/wd/hub'; // this is the default
//这里使用的是chrome浏览器进行测试，需到http://www.seleniumhq.org/download/上下载对应的浏

览器测试插件
//我这里下载的是win32 Google Chrome Driver 2.25
版：https://chromedriver.storage.googleapis.com/index.html?path=2.25/
$capabilities = DesiredCapabilities::chrome();
$driver = RemoteWebDriver::create($host, $capabilities, 5000);
// navigate to 'http://docs.seleniumhq.org/'
$driver->get('https://www.baidu.com/');

echo iconv("UTF-8","GB2312",'标题1')." : " . $driver->getTitle() . "\n"; //cmd.exe中文乱码，所以需转码

$driver->findElement(WebDriverBy::id('kw'))->sendKeys('wwe')->submit();

// 等待新的页面加载完成....
$driver->wait($waitSeconds)->until(

```
WebDriverExpectedCondition::visibilityOfElementLocated(
    WebDriverBy::partialLinkText('100shuai')
)
```

);
$driver->findElement(WebDriverBy::partialLinkText('100shuai'))->sendKeys('xxx')->click(); //一般点击链接的时候，担心因为失去焦点而抛异常，则可以先调用一下sendKeys，再click

switchToEndWindow($driver); //切换至最后一个window

// 等待加载....
$driver->wait($waitSeconds)->until(

```
WebDriverExpectedCondition::visibilityOfElementLocated(
    WebDriverBy::partialLinkText('SmackDown收视率创历史新低')
)
```

);
echo iconv("UTF-8","GB2312",'标题2')." : " . $driver->getTitle() . "\n"; //cmd.exe中文乱码，所以需转码

$driver->findElement(WebDriverBy::partialLinkText('SmackDown收视率创历史新低'))->click();

switchToEndWindow($driver); //切换至最后一个window

// 等待加载....
$driver->wait($waitSeconds)->until(

```
WebDriverExpectedCondition::titleContains('SmackDown收视率创历史新低')
```

);
echo iconv("UTF-8","GB2312",'标题3')." : " . $driver->getTitle() . "\n"; //cmd.exe中文乱码，所以需转码

//关闭浏览器
$driver->quit();

//切换至最后一个window
//因为有些网站的链接点击过去带有target="_blank"属性，就新开了一个TAB，而selenium还是定位到老的TAB上，如果要实时定位到新的TAB，则需要调用此方法，切换到最后一个window
function switchToEndWindow($driver){

```
$arr = $driver->getWindowHandles();
foreach ($arr as $k=>$v){
    if($k == (count($arr)-1)){
        $driver->switchTo()->window($v);
    }
}
```

}

?>

```
下载资料：
1.http://selenium-release.storage.googleapis.com/index.html (selenium 下载地址)
2.https://chromedriver.storage.googleapis.com/index.html (chrome driver 下载地址)
3.http://phantomjs.org/download.html (PhantomJS Driver 下载地址)
4.http://www.cnbeta.com/articles/soft/563605.htm (chrome 下载地址，建议使用这个版本或者以下版本，其他最新版本，浏览器识别了是否为测试软件，对于个别用途的软件需要注意，如果仅仅是为了做测试，那就无所谓了。)
5.http://blog.csdn.net/huilan_same/article/details/51896672 （chromedriver.exe版本对应的chrome版本）


参考文档：
1.https://github.com/facebook/php-webdriver （里面有example.php以及 tests文件下的案例文档共参考）
2.https://github.com/facebook/php-webdriver/wiki 快速开始教程
3.http://facebook.github.io/php-webdriver/namespaces/default.html API文档
4.http://www.yiibai.com/selenium/ 易百教程
6.https://github.com/chibimagic/WebDriver-PHP/
7.https://code.google.com/archive/p/php-webdriver-bindings/
8.https://github.com/Element-34/php-webdriver
9.https://github.com/Nearsoft/php-selenium-client
10.http://pan.baidu.com/s/1eR31pM6 selenium_webdriver(python)第一版.pdf
```

元素定位

> 在使用selenium webdriver进行元素定位时，通常使用findElement或findElements方法结合By类返回的元素句柄来定位元素。其中By类的常用定位方式共八种，现分别介绍如下。

1. WebDriverBy::id()

> 通过ID属性定位元素

```
//HTML代码
//<input id="kw" name="wd" class="s_ipt" maxlength="255" autocomplete="off" />
```

<?php
$driver = RemoteWebDriver::create($host, DesiredCapabilities::chrome(), 5000);
$driver->get('https://www.baidu.com/');
$element = $driver->findElement(WebDriverBy::id('kw'));
?>

```
#### 2. WebDriverBy::name()
> 通过name属性定位元素。如果查找多个，则使用findElements进行定位
```

//HTML代码
// 

<?php
$driver = RemoteWebDriver::create($host, DesiredCapabilities::chrome(), 5000);
$driver->get('https://www.baidu.com/');
$element = $driver->findElement(WebDriverBy::name('wd'));
foreach($elements as $elem){ //有时候多个元素时，想找出某个特定元素，可根据attribute或text进行判断过滤

```
echo $elem->getAttribute('type');
echo $elem->getText();
```

}
?>

```
#### 3. WebDriverBy::tagName()
> 通过标签进行定位元素。如果查找多个，则使用findElements进行定位，具体参考WebDriverBy::name()的代码
```

```
//HTML代码
//
// 
//.....

<?php
$driver = RemoteWebDriver::create($host, DesiredCapabilities::chrome(), 5000);
$driver->get('https://www.baidu.com/');
$element = $driver->findElement(WebDriverBy::tagName('input'));
?>
```

```
#### 4. WebDriverBy::className()
> 通过css类进行定位元素。如果查找多个，则使用findElements进行定位，具体参考WebDriverBy::name()的代码
```

```
//HTML代码
// 

<?php
$driver = RemoteWebDriver::create($host, DesiredCapabilities::chrome(), 5000);
$driver->get('https://www.baidu.com/');
$element = $driver->findElement(WebDriverBy::className('s_ipt'));
?>
```

```
#### 5. WebDriverBy::linkText()
> 通过超文本链接上的文字信息来定位元素。如果查找多个，则使用findElements进行定位，具体参考WebDriverBy::name()的代码
```

```
//HTML代码
//新闻

<?php
$driver = RemoteWebDriver::create($host, DesiredCapabilities::chrome(), 5000);
$driver->get('https://www.baidu.com/');
$element = $driver->findElement(WebDriverBy::linkText('新闻'));
?>
```

```
#### 6. WebDriverBy::partialLinkText()
> 这个方法是上一个方法的扩展。当你不能准确知道超链接上的文本信息或者只想通过一些关键字进行匹配时，可以使用这个方法来通过部分链接文字进行匹配。如果查找多个，则使用findElements进行定位，具体参考WebDriverBy::name()的代码
```

//HTML代码
//新闻

```
<?php
$driver = RemoteWebDriver::create($host, DesiredCapabilities::chrome(), 5000);
$driver->get('https://www.baidu.com/');
$element = $driver->findElement(WebDriverBy::partialLinkText('新'));
?>
```

```
#### 7. WebDriverBy::xpath()
> 这个方法是非常强大的元素查找方式，使用这种方法几乎可以定位到页面上的任意元素。在正式开始使用XPath进行定位前，我们先了解下什么是XPath。XPath是XML Path的简称，由于HTML文档本身就是一个标准的XML页面，所以我们可以使用XPath的语法来定位页面元素。更多详细的定位资料，参考页脚链接地址。

XPath语法如下：
```

```
<?php
$element = $driver->findElement(WebDriverBy::xpath("//*
[@id='J_login_form']/dl/dt/input[@id='J_password']"));
?>
```

```
#### 8. WebDriverBy::cssSelector()
> cssSelector这种元素定位方式跟xpath比较类似，但执行速度较快，而且各种浏览器对它的支持都相当到位，所以功能比较强大。
下面是一些常见的cssSelector的定位方式：
定位id为flrs的div元素，可以写成：#flrs      注：相当于xpath语法的//div[@id='flrs']
定位id为flrs下的a元素，可以写成 #flrs > a  注：相当于xpath语法的//div[@id='flrs']/a
定位id为flrs下的href属性值为/forexample/about.html的元素，可以写成： #flrs > a[href="/forexample/about.html"]
如果需要指定多个属性值时，可以逐一加在后面，如#flrs > input[name="username"][type="text"]。

明白基本语法后，我们来尝试用cssSelector方式来层级关系定位，代码如下：
```

//HTML代码
//新闻

```
<?php
$element = $driver-
>findElement(WebDriverBy::cssSelector("#J_login_form>dl>dt>input[id='J_password']"));
?>
```

```
cssSelector还有一个用处是定位使用了复合样式表的元素，之前在第4种方式className里面提到过。现在我们就来看看如何通过cssSelector来引用到第4种方式中提到的那个button。button代码如下：
```

//HTML代码
// 登录

<?php
$element = $driver->findElement(WebDriverBy::cssSelector("button.btn.btn_big.btn_submit"));
?>

```
此外，cssSelector还有一些高级用法，如果熟练后可以更加方便地帮助我们定位元素，如我们可以利用^用于匹配一个前缀，$用于匹配一个后缀，*用于匹配任意字符。例如：
>匹配一个有id属性，并且id属性是以”id_prefix_”开头的超链接元素：a[id^='id_prefix_']
匹配一个有id属性，并且id属性是以”_id_sufix”结尾的超链接元素：a[id$='_id_sufix']
匹配一个有id属性，并且id属性中包含”id_pattern”字符的超链接元素：a[id*='id_pattern']


#### 9. 判断定位的元素是否存在
> selenium找元素时，如果超出timeout时间后，还未找到元素，则会报异常，我们则可以利用这样的方式进行判断元素是否存在。不过这种方式对于及时性要求不高的情况下使用，如果要求及时性的话，建议还是获取到page source后，用htmlparser或正则判断即可。
```

<?php

if(isElementExsit($driver, WebDriverBy::linkText('新闻'))){

```
echo '找到元素啦';
```

}else{

```
echo '没有找到元素';
```

}

/**

- 判断元素是否存在
- @param WebDriver $driver
- @param WebDriverBy $locator
  */
  function isElementExsit($driver,$locator){
  try {

```
$nextbtn = $driver->findElement($locator);
return true;
```

} catch (\Exception $e) {

```
echo 'element is not found!';
return false;
```

}
}
?>

```

```

更多参考资料：
1.https://saucelabs.com/resources/articles/selenium-tips-css-selectors

元素操作

元素操作

> 前面讲到了不少知识都是定位元素，定位只是第一步，定位之后需要对这个原素进行操作。鼠标点击呢还是键盘输入，这要取决于我们定位的是按钮还输入框。
> 一般来说，webdriver 中比较常用的操作对象的方法有下面几个
> > click 点击对象
> > send_keys 在对象上模拟按键输入
> > clear 清除对象的内容，如果可以的话
> > submit 清除对象的内容，如果可以的话

1.输入框

```
$element = $driver->findElement(WebDriverBy::id('wd'));
$element->sendKeys("wwe");      //在输入框中输入内容
$element->clear();      //将输入框清空
$element->getAttribute('value');        //获取输入框的文本内容
```

2.单选框

```
$radio = $driver->findElement(WebDriverBy::id('BookMode'));
$radio->click();        //选择某个单选项
$radio->isSelected();   //判断某个单选项是否已经被选择
```

3.多选框

```
$checkbox = $driver->findElement(WebDriverBy::id('myCheckbox'));
$checkbox->click();     //选择某选项
$checkbox->isSelected();        //判断该选项是否已经被选择
$checkbox->isEnabled(); //判断是否enable
```

4.下拉框

> 下拉框是我们最常见的一种页面元素，对于一般的元素，我们只需要一次就定位，但下拉框里的内容需要进行两次定位，先定位到下拉框，再定位到下拉框内里的选项。

```
$select = $driver->findElement(WebDriverBy::id('myselect'));
$select->findElement(WebDriverBy::xpath("//option[@value='100']"))->click();    //找到值为100的选项进行选中
```

5.按钮

```
$btn = $driver->findElement(WebDriverBy::id('btn'));
$btn->click();  //点击按钮
$btn->isEnabled();      //判断是否enable
```

6.文件上传

```
$upload = $driver->findElement(WebDriverBy::id('img-upload'));
$filePath = "C:\test\\uploadfile\\media_ads\\test.jpg";
$upload->sendKeys($filePath);
```

7.指定元素点击

> 因为有时候因为种种原因（如元素上有蒙版层或者因为一些浮动的div导致坐标变化或者不可点击），而不能使用$ele.click()时，采用通用的方式，就是用JS执行点击事件。而且点击事件是可被监听的，可在元素上写onclick事件进行监听。

```
//用webdriver获取对象，再进行JS操作
$ele = $driver->findElement(WebDriverBy::id('haha'));
$driver->executeScript("arguments[0].click();",[$ele]);
```

//用JS获取对象进行操作
$js = <<<js
var ele = document.getElementById('haha');
ele.click();
js;
$driver->executeScript($js);

```
#### 8.滚动到指定元素
> 用JS的方式滚动到指定的元素。
```

$ele = $driver->findElement(WebDriverBy::id('haha2'));
$driver->executeScript("arguments[0].scrollIntoView();",[$ele]);

```
#### 9.移动到指定元素
> 移动的时候，鼠标不会真正移动，但实际上已移动，这个可以在haha2元素上写一个onmouseover事件进行监听即可测试。
```

```
$ele = $driver->findElement(WebDriverBy::id('haha2'));
$driver->getMouse()->mouseMove($ele->getCoordinates());
```

元素等待

> web的自动化测试中，我们经常会遇到这样一种情况：当我们的程序执行时需要页面某个元素，而此时这个元素还未加载完成，这时我们的程序就会报错。怎么办？等待。等待元素出现后再进行对这个元素的操作。
> 在selenium-webdriver中我们用两种方式进行等待：明确的等待和隐性的等待。

明确的等待

>明确的等待是指在代码进行下一步操作之前等待某一个条件的发生。最不好的情况是使用sleep()去设置一段确认的时间去等待。但为什么说最不好呢？因为一个元素的加载时间有长有短，你在设置sleep的时间之前要自己把握长短，太短容易超时，太长浪费时间。selenium webdriver提供了一些方法帮助我们等待正好需要等待的时间，比如以下例子中我们设置的最长等待时间为15秒。

```
<?php
namespace Facebook\WebDriver;
use Facebook\WebDriver\Remote\DesiredCapabilities;
use Facebook\WebDriver\Remote\RemoteWebDriver;
require_once('vendor/autoload.php');

header("Content-Type: text/html; charset=UTF-8");
// start Firefox with 5 second timeout
$host = 'http://localhost:4444/wd/hub'; // this is the default
$waitSeconds = 15;  //需等待加载的时间，一般加载时间在0-15秒，如果超过15秒，报错。
$capabilities = DesiredCapabilities::chrome();
$driver = RemoteWebDriver::create($host, $capabilities, 5000);
$driver->get('https://www.baidu.com/');

//由于下拉框是通过点击“搜索设置”按钮触发JS动态生成的DOM，所以这里使用Wait for new element to appear方式，不然直接调用查找元素会报错，说找不到元素
$driver->wait($waitSeconds)->until(
    WebDriverExpectedCondition::visibilityOfElementLocated(
        WebDriverBy::id('kw')
    )
);
$driver->findElement(WebDriverBy::id('kw'))->sendKeys('wwe');
echo 'done!';
//关闭浏览器
//$driver->quit();

?>
```

隐性等待

> 隐性等待是指当要查找元素，而这个元素没有马上出现时，告诉WebDriver查询Dom一定时间。默认值是0,但是设置之后，这个时间将在WebDriver对象实例整个生命周期都起作用。上面的代码就变成

> 了这样：
>
> ```
> <?php
> namespace Facebook\WebDriver;
> use Facebook\WebDriver\Remote\DesiredCapabilities;
> use Facebook\WebDriver\Remote\RemoteWebDriver;
> require_once('vendor/autoload.php');
> ```

```
header("Content-Type: text/html; charset=UTF-8");
// start Firefox with 5 second timeout
$host = 'http://localhost:4444/wd/hub'; // this is the default
$capabilities = DesiredCapabilities::chrome();
$driver = RemoteWebDriver::create($host, $capabilities, 5000);
$driver->manage()->timeouts()->implicitlyWait(15); //隐性设置15秒
$driver->get('https://www.baidu.com/');

$driver->findElement(WebDriverBy::id('kw'))->sendKeys('wwe');
echo 'done!';

//关闭浏览器
//$driver->quit();

?>
```

>两者选其一，第二种看起来比较一劳永逸。
如果一个无素没有出现都会默认等待你所设定的时间，直到超时或者元素出现

JS调用

JS调用

```
<?php
namespace Facebook\WebDriver;
use Facebook\WebDriver\Remote\DesiredCapabilities;
use Facebook\WebDriver\Remote\RemoteWebDriver;
require_once('vendor/autoload.php');

header("Content-Type: text/html; charset=UTF-8");


// start Firefox with 5 second timeout
$host = 'http://localhost:4444/wd/hub'; // this is the default
$capabilities = DesiredCapabilities::chrome();
$driver = RemoteWebDriver::create($host, $capabilities, 5000);
$driver->get('https://www.baidu.com/');

$element = $driver->findElement(WebDriverBy::className('s_ipt'));

$js = <<<js
        document.getElementById("kw").style.border = '2px solid red';
        var button = document.getElementById("su");
        button.setAttribute('type','button');
        button.setAttribute('onclick','document.getElementById("kw").style.border = "2px solid blue";al
ert("hello,dear wangkun!");');
js;
$driver->executeScript($js);
echo 'done!';

//关闭浏览器
//$driver->quit();

?>
```

验证码识别

> 有些网站打开后，有验证码需要填写，而我们又不能直接获取该验证码的远程地址，然后下载，这样的话就相当于又请求了一次，那么验证码又被更新了，就达不到我们识别验证码的效果。
> 而我们的方案是通过网页截图，然后找到验证码的具体位置，然后再截图方式把验证码图片获取到。
> 废话不多说，直接开始！

示例脚本：

```
<?php
namespace Facebook\WebDriver;
use Facebook\WebDriver\Remote\DesiredCapabilities;
use Facebook\WebDriver\Remote\RemoteWebDriver;
require_once('vendor/autoload.php');

header("Content-Type: text/html; charset=UTF-8");
const vcodeDst = 'f://vcode.png';   //验证码存放地址

// start Firefox with 5 second timeout
$host = 'http://localhost:4444/wd/hub'; // this is the default
$capabilities = DesiredCapabilities::chrome();
$driver = RemoteWebDriver::create($host, $capabilities, 5000);
$driver->get('http://www.yimuhe.com/');

$driver->manage()->window()->maximize();    //将浏览器最大化
$driver->takeScreenshot(vcodeDst);  //截取当前网页，该网页有我们需要的验证码
$element = $driver->findElement(WebDriverBy::id('vcode_img'));
generateVcodeIMG($element->getLocation(), $element->getSize(),vcodeDst);

echo 'done!';

//关闭浏览器
$driver->quit();

/**
 * 生成验证码图片
 * @param $location 验证码x,y轴坐标
 * @param $size 验证码的长宽
 */
function generateVcodeIMG($location,$size,$src_img){
    $width = $size->getWidth();
    $height = $size->getHeight();
    $x = $location->getX();
    $y = $location->getY();

    $src = imagecreatefrompng($src_img);
    $dst = imagecreatetruecolor($width,$height);
    imagecopyresampled($dst,$src,0,0,$x,$y,$width,$height,$width,$height);
    imagejpeg($dst,$src_img);
    chmod($src_img,0777);
    imagedestroy($src);
    imagedestroy($dst);
}
?>
```

> 备注：当我们已经把正确的验证码图片下载到了本地后，不管是用自己写的OCR程序进行识别还是用第三方程序进行识别都可以，这个就比较简单了，就不在这里进行陈述。
> 这里推荐一个比较准确的第三方验证码识别的程序，集成一下就可以了。
> https://www.juhe.cn/docs/api/id/60/aid/344

frame与弹窗的控制

弹窗与frame的定位与控制

> 对于web应用，经常会出现框架（frame）或窗口（window）的应用，这也就给我们的定位带来了一个难题。有时候我们定位一个元素，定位器没有问题，但一直定位不了，这时候就要检查这个元素是否在一个frame中，seelnium webdriver 提供了这样的方法，可以很轻松的来解决这个问题。
> > $driver->switchTo()->frame("id")
> > $driver->switchTo()->window("id")
> >* $driver->switchTo()->alert()

frame示例脚本

```
<?php
namespace Facebook\WebDriver;
use Facebook\WebDriver\Remote\DesiredCapabilities;
use Facebook\WebDriver\Remote\RemoteWebDriver;
require_once('vendor/autoload.php');

header("Content-Type: text/html; charset=UTF-8");


// start Firefox with 5 second timeout
$host = 'http://localhost:4444/wd/hub'; // this is the default
$capabilities = DesiredCapabilities::chrome();
$driver = RemoteWebDriver::create($host, $capabilities, 5000);
$driver->manage()->timeouts()->implicitlyWait(15);    //隐性设置15秒
$driver->get('https://v.qq.com/x/cover/e7hi6lep1yc51ca.html?vid=h0018p9ihom');

echo $driver->getCurrentURL().'\r\n';


//将页面滚动条拖到底部
//因为这个页面默认打开的时候，"评论区"的iframe没有渲染到DOM里，腾讯做的处理是拖动到底部的时候用JS动态渲染，所以我们需要控制浏览器滚动至底部
$js = "window.scrollBy(0,100000000);";
$driver->executeScript($js);
sleep(3);

#再找到其下面的 iframe(id=commentIframe)
$driver->switchTo()->frame("commentIframe");
$str = $driver->getPageSource();

//将获取到的影评数据保存再本地，再测试是否正确。
$myfile = fopen("d://newfile.html", "w") or die("Unable to open file!");
fwrite($myfile, $str);
fclose($myfile);
```

```
echo 'done!';

//关闭浏览器
$driver->quit();

?>
```

alert示例脚本

```
<?php
//~~~以上代码省略...
$driver->switchTo()->alert()->accept(); //获取到confim alert, 并且点击同意
$driver->switchTo()->alert()->dismiss();        //取消
$driver->switchTo()->alert()->getText();        //获取alert弹出的提示内容
?>
```

> 我们以百度的页面为例来演示一下功能。
> 1.点击百度的设置
> 2.选择"每天显示多少条"的下拉框 (这个顺便把“元素操作”文章中的“select”元素章节也演示了)
> 3.点击保存设置按钮：会弹出一个alert弹窗
> 4.执行点击alert的确定按钮
> 5.试一下搜索的结果是否变化，操作完毕
>
> ```
> <?php
> namespace Facebook\WebDriver;
> use Facebook\WebDriver\Remote\DesiredCapabilities;
> use Facebook\WebDriver\Remote\RemoteWebDriver;
> require_once('vendor/autoload.php');
> ```

header("Content-Type: text/html; charset=UTF-8");

// start Firefox with 5 second timeout
$host = 'http://localhost:4444/wd/hub'; // this is the default
$waitSeconds = 15; //需等待加载的时间，一般加载时间在0-15秒，如果超过15秒，报错。
$capabilities = DesiredCapabilities::chrome();
$driver = RemoteWebDriver::create($host, $capabilities, 5000);
$driver->get('https://www.baidu.com/');

$driver->findElement(WebDriverBy::linkText('设置'))->click();

$driver->findElement(WebDriverBy::linkText('搜索设置'))->click();

$warpper = $driver->findElement(WebDriverBy::id('wrapper'));

//由于下拉框是通过点击“搜索设置”按钮触发JS动态生成的DOM，所以这里使用Wait for new

element to appear方式，不然直接调用查找元素会报错，说找不到元素
$driver->wait($waitSeconds)->until(

```
WebDriverExpectedCondition::visibilityOfElementLocated(
    WebDriverBy::id('nr')
)
```

);

$selectDom = $warpper->findElement(WebDriverBy::id('nr'));
$select = new WebDriverSelect($selectDom);
$select->selectByValue(10);

//由于下拉框是通过点击“搜索设置”按钮触发JS动态生成的DOM，所以这里使用Wait for new element to appear方式，不然直接调用查找元素会报错，说找不到元素
$driver->wait($waitSeconds)->until(

```
WebDriverExpectedCondition::visibilityOfElementLocated(
    WebDriverBy::linkText('保存设置')
)
```

);

$driver->findElement(WebDriverBy::linkText('保存设置'))->click();

sleep(2);

$driver->switchTo()->alert()->accept();

$driver->findElement(WebDriverBy::id("kw"))->sendKeys('wwe');
$driver->findElement(WebDriverBy::id("su"))->click();

echo 'done!';

//关闭浏览器
//$driver->quit();

?>

```
#### window弹窗 示例脚本
```

//$driver->switchTo()->window("id")，用法与 frame 相同。

AJAX分页数据获取

> 有时候有些列表页使用的是滚动条到最底部才加载下一页的数据，这个时候就需要用到selenium来操作这样的数据。我们拿letv的分页来做测试。

```
<?php
namespace Facebook\WebDriver;
use Facebook\WebDriver\Remote\DesiredCapabilities;
use Facebook\WebDriver\Remote\RemoteWebDriver;
require_once('vendor/autoload.php');

header("Content-Type: text/html; charset=UTF-8");
// start Firefox with 5 second timeout
$host = 'http://localhost:4444/wd/hub'; // this is the default
$capabilities = DesiredCapabilities::chrome();
$driver = RemoteWebDriver::create($host, $capabilities, 5000);
$driver->manage()->timeouts()->implicitlyWait(15);    //隐性设置15秒
$driver->get('http://list.le.com/listn/c1_t-1_a50071_y-1_s1_lg-1_ph-1_md_o4_d1_p.html');

//翻一页
js = "window.scrollTo(0,document.body.scrollHeight)"    //滚动至底部
//$js = "window.scrollBy(0,100000000);";  //也可以把值设大一点，达到底部的效果
$driver->executeScript($js);

echo 'sleep...';
sleep(6);

//再翻一页
js = "window.scrollTo(0,document.body.scrollHeight)";
$driver->executeScript($js);

echo 'sleep...';
sleep(6);

//再翻一页
js = "window.scrollTo(0,document.body.scrollHeight)";
$driver->executeScript($js);

echo 'done!';

//关闭浏览器
//$driver->quit();

?>
```

PhantomJS

> 用浏览器驱动的方式，很方便我们在测试阶段调试代码正确性，但是由于浏览器要启动，解析DOM、JS、下载图片等，使得程序跑起来的效率并不高，这个时候我们就需要用到Phantomjs，以后台的形式运行程序，大大的提升运行的性能。

1.安装Phantomjs

> 到http://phantomjs.org/download.html 上面去下载对应的版本，我这里下载的是windows版本的。将解压包中的phantomjs.exe放到PHP程序根目录，或将该exe加入到本机的环境变量中都行。

2.使用Phantomjs

> 拿的是“验证码识别”那篇文章的代码，只改了一处，$capabilities = DesiredCapabilities::phantomjs();这一行。
> 运行后，可以看到“验证码识别”程序不再启动chrome浏览器，而是后台执行，速度也快了很多。程序顺利跑出了我们想要的结果~ 大家可以休息一下咯~

```
<?php
namespace Facebook\WebDriver;
use Facebook\WebDriver\Remote\DesiredCapabilities;
use Facebook\WebDriver\Remote\RemoteWebDriver;
require_once('vendor/autoload.php');
```

```
header("Content-Type: text/html; charset=UTF-8");
const vcodeDst = 'f://vcode.png'; //验证码存放地址

// start Firefox with 5 second timeout
$host = 'http://localhost:4444/wd/hub'; // this is the default
$capabilities = DesiredCapabilities::phantomjs();
$driver = RemoteWebDriver::create($host, $capabilities, 5000);
$driver->get('http://www.yimuhe.com/');

$driver->manage()->window()->maximize(); //将浏览器最大化
$driver->takeScreenshot(vcodeDst); //截取当前网页，该网页有我们需要的验证码
$element = $driver->findElement(WebDriverBy::id('vcode_img'));
generateVcodeIMG($element->getLocation(), $element->getSize(),vcodeDst);

echo 'done!';

//关闭浏览器
```

```
$driver->quit();

/**
```

- 生成验证码图片
- @param $location 验证码x,y轴坐标
- @param $size 验证码的长宽

```
*/
function generateVcodeIMG($location,$size,$src_img){
$width = $size->getWidth();
$height = $size->getHeight();
$x = $location->getX();
$y = $location->getY();

$src = imagecreatefrompng($src_img);
$dst = imagecreatetruecolor($width,$height);
imagecopyresampled($dst,$src,0,0,$x,$y,$width,$height,$width,$height);
imagejpeg($dst,$src_img);
chmod($src_img,0777);
imagedestroy($src);
imagedestroy($dst);
}
?>
```

屏蔽图片

> 共有两种方法，一种设置chrome配置，一种直接使用插件达到相应的目的。其中使用插件来屏蔽图片可转到“使用扩展插件”章节。这里就介绍chrome配置。

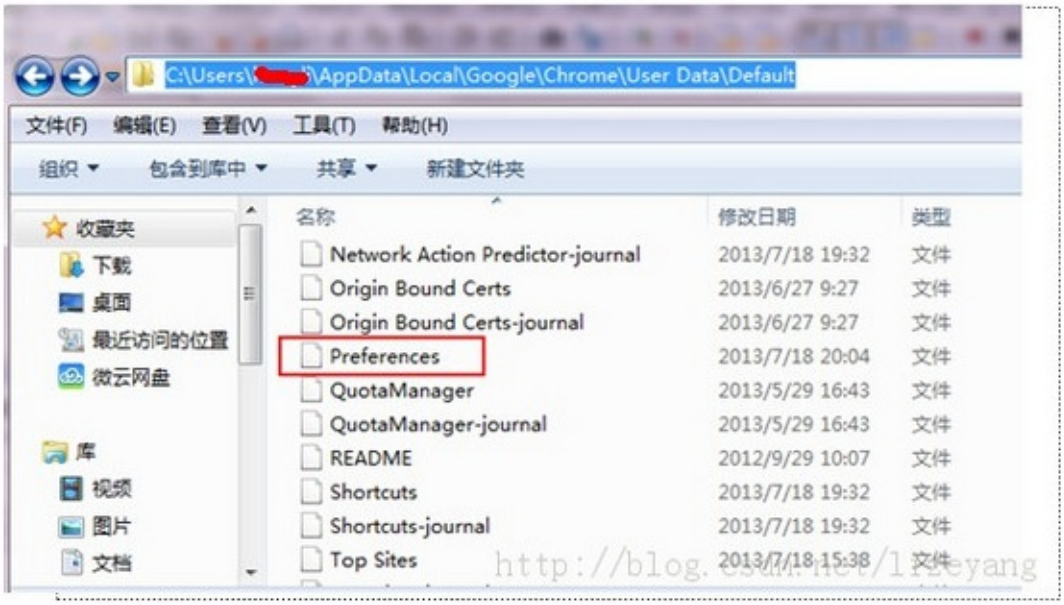
1. 查看chrome支持的浏览器属性

> chrome driver的官方文档(https://sites.google.com/a/chromium.org/chromedriver/capabilities)，可以看到，chrome driver可以支持的自定义属性

Name	Type	Default	Description
args	list of strings		List of command-line arguments to use when starting Chrome. Arguments with an associated value should be separated by a '=' sign (e.g., ['start-maximized', 'user-data-dir=/tmp/temp_profile']). See here for a list of Chrome arguments.
binary	string		Path to the Chrome executable to use (on Mac OS X, this should be the actual binary, not just the app. e.g., '/Applications/Google Chrome.app/Contents/MacOS/Google Chrome')
extensions	list of strings		A list of Chrome extensions to install on startup. Each item in the list should be a base-64 encoded packed Chrome extension (.crx)
localState	dictionary		A dictionary with each entry consisting of the name of the preference and its value. These preferences are applied to the Local State file in the user data folder.
prefs	dictionary		A dictionary with each entry consisting of the name of the preference and its value. These preferences are only applied to the user profile in use. See the 'Preferences' file in Chrome's user data directory for examples.
detach	boolean	false	If false, Chrome will be quit when ChromeDriver is killed, regardless of whether the session is quit. If true, Chrome will only be quit if the session is quit (or closed). Note, if true, and the session is not quit, ChromeDriver cannot clean up the temporary user data directory that the running Chrome instance is using.

2. 查看本地配置的参数值

> 可以使用自己的chrome浏览器进行配置，配置好了后，查看“Preferences文件”里的值就可以了。一般路径都为“用户文件夹\AppData\Local\Google\Chrome\User Data\Default”。


C:\Users\[redacted]\AppData\Local\Google\Chrome\User Data\Default
文件(F) 编辑(E) 查看(V) 工具(T) 帮助(H)
组织 包含到库中 共享 新建文件夹
收藏夹
下载
桌面
最近访问的位置
微云网盘
库
视频
图片
文档
名称 修改日期 类型
Network Action Predictor-journal 2013/7/18 19:32 文件
Origin Bound Certs 2013/6/27 9:27 文件
Origin Bound Certs-journal 2013/6/27 9:27 文件
Preferences 2013/7/18 20:04 文件
QuotaManager 2013/5/29 16:43 文件
QuotaManager-journal 2013/5/29 16:43 文件
README 2012/9/29 10:07 文件
Shortcuts 2013/7/18 19:32 文件
Shortcuts-journal 2013/7/18 19:32 文件
Top Sites 2013/7/18 15:38 文件
http://blog.csdn.net/liuyang


设置 - 内容设置
chrome://settings/content
Chrome
设置
内容设置
Cookie
允许设置本地数据（推荐）
只在我退出浏览器之前保留本地数据
阻止网站设置任何数据
阻止第三方 Cookie 和网站数据
管理例外情况...
所有 Cookie 和网站数据...
图片
显示所有图片（推荐）
不显示任何图片

对比设置前后Preferences的区别

了这个修改后，会发现 Preferences多了红框这几行，因此这几行配置，对应了“不显示图片”这配置

```
"profile": {
   "avatar_index": 0,
   "block_third_party_cookies": false,
   "content_settings": {
      "clear_on_exit_migrated": true,
      "pattern_pairs": {
         "[*.]x.rdm.3g.qq.com,*": {
            "cookies": 1
         },
         "file:///Users/harryli/Desktop/ITC/iTunes%20Connect1.htm,*": {
            "images": 1
         },
         "file:///Users/harryli/Desktop/ITC/iTunes%20Connect2.htm,*": {
            "images": 1
         },
         "http://profile.pengyou.com:80,*": {
            "notifications": 1
         }
      },
      "pref_version": 1
   },
   "default_content_settings": {
      "images": 2
   },
   "exit_type": "Crashed",
   "exited_cleanly": true,
   "is_managed": false,
   "name": "\u7B2C 1 \u4F4D\u7528\u6237",
   "per_host_zoom_levels": {
      "bbs.csdn.net": -0.5778829455375671,
      "write.blog.csdn.net": -0.5778829455375671,
```

3. 代码DEMO

> 找到了屏蔽图片对应的参数后，我们就可以进行测试了
>
> ```
> $host = 'http://localhost:4444/wd/hub'; // this is the default
> $capabilities = DesiredCapabilities::chrome();
> $options = new ChromeOptions();
> $value = ['profile.managed_default_content_settings.images'=>2];
> $options->setExperimentalOption('prefs', $value);
> $capabilities->setCapability(ChromeOptions::CAPABILITY, $options);
> ```

$driver = RemoteWebDriver::create($host, $capabilities, 5000);
$driver->manage()->timeouts()->implicitlyWait(15); //隐性设置15秒

$driver->get('http://www.baidu.com/');

echo 'done';

```
#### 4. 换个思路，变得更简洁
> 上面提到的都是设置具体的参数，而要找到对应设置的参数比较繁杂，当如果不想那么麻烦时候，完全可以在已有的chrome浏览器先设置好，然后把“用户文件夹\AppData\Local\Google\Chrome\User Data”加载到自己的应用中。
```

```
$host = 'http://localhost:4444/wd/hub'; // this is the default
$capabilities = DesiredCapabilities::chrome();
$options = new ChromeOptions();
$options->addArguments(["--user-data-dir=d:/xampp/test/User Data"]);
$capabilities->setCapability(ChromeOptions::CAPABILITY, $options);

$driver = RemoteWebDriver::create($host, $capabilities, 5000);
$driver->manage()->timeouts()->implicitlyWait(15); //隐性设置15秒

$driver->get('http://www.baidu.com/');

echo 'done';
```

> 总结：这里虽然只是介绍的屏蔽图片，但是同理的chrome其他设置也可以用这样的方式进行，如禁用JS等等之类的。

使用扩展插件

> 有时候我们需要使用浏览器的扩展插件来帮助我们达到某种特定的效果，比如“Block Image”插件可使得selenium使用过程中不显示图片，从而加快访问的速度和性能。再比如使用“adsafe”插件可使得访问网站的时候禁用相应的广告以及弹窗广告等。

>这里我们使用"Block Image"插件的来展示下selenium怎么加载插件。这里使用的是chrome driver方式进行测试，其他浏览器的测试代码请自行百度。

1.下载Block Image插件

> 到http://www.cnplugins.com/ 上面搜索Block Image，下载下来，存放到自己的程序根目录。

2.代码测试

```
<?php
namespace Facebook\WebDriver;
use Facebook\WebDriver\Remote\DesiredCapabilities;
use Facebook\WebDriver\Remote\RemoteWebDriver;
use Facebook\WebDriver\Chrome\ChromeOptions;
require_once('vendor/autoload.php');

header("Content-Type: text/html; charset=UTF-8");

// start Firefox with 5 second timeout
$host = 'http://localhost:4444/wd/hub'; // this is the default
$capabilities = DesiredCapabilities::chrome();
$options = new ChromeOptions();
$options->addExtensions(['Block-image.crx']);   //这一句则为加载我们下载好的插件
$capabilities->setCapability(ChromeOptions::CAPABILITY, $options);
$driver = RemoteWebDriver::create($host, $capabilities, 15000);
$driver->manage()->timeouts()->implicitlyWait(15);    //隐性设置15秒
$driver->get('http://www.tudou.com/list/ach4a-2b-2c-2d-2e-2f1689g-2h-2i-2j-2k-2l-2m-2n-2sort1.html');
```

代理设置

> 有时候我们需要修改浏览器的一些User Agent信息或代理IP，从而达到更好的兼容性测试。这里我们就用一个在线UserAgent分析工具进行测试。

1.修改User Agent
>Chrome 示例

> 这里提供user-agent大全供参考http://www.cnblogs.com/hykun/p/Ua.html

```
<?php
namespace Facebook\WebDriver;
use Facebook\WebDriver\Remote\DesiredCapabilities;
use Facebook\WebDriver\Remote\RemoteWebDriver;
use Facebook\WebDriver\Chrome\ChromeOptions;
require_once('vendor/autoload.php');
```

header("Content-Type: text/html; charset=UTF-8");

// start Firefox with 5 second timeout
$host = 'http://localhost:4444/wd/hub'; // this is the default
$capabilities = DesiredCapabilities::chrome();
$useragent = 'Mozilla/5.0 (Linux; U; Android 2.3.7; en-us; Nexus One Build/FRF91) AppleWebKit/533.1 (KHTML, like Gecko) Version/4.0 Mobile Safari/533.1';
$options = new ChromeOptions();
$options->addArguments(["user-agent={$useragent}"]);
$capabilities->setCapability(ChromeOptions::CAPABILITY, $options);

$driver = RemoteWebDriver::create($host, $capabilities, 5000);
$driver->manage()->timeouts()->implicitlyWait(15); //隐性设置15秒
$driver->get('http://www.atool.org/useragent.php');
var_dump($capabilities->getCapability(ChromeOptions::CAPABILITY));
echo 'done!';

//关闭浏览器
//$driver->quit();
?>

```
>Phantomjs 示例
```

```
<?php
namespace Facebook\WebDriver;
use Facebook\WebDriver\Remote\DesiredCapabilities;
use Facebook\WebDriver\Remote\RemoteWebDriver;
use Facebook\WebDriver\Chrome\ChromeOptions;
require_once('vendor/autoload.php');

header("Content-Type: text/html; charset=UTF-8");

// start Firefox with 5 second timeout
$host = 'http://localhost:4444/wd/hub'; // this is the default
$capabilities = DesiredCapabilities::phantomjs();
$useragent = 'Mozilla/5.0 (Linux; U; Android 2.3.7; en-us; Nexus One Build/FRF91)
AppleWebKit/533.1 (KHTML, like Gecko) Version/4.0 Mobile Safari/533.1';
$capabilities->setCapability("phantomjs.page.settings.userAgent", $useragent);

$driver = RemoteWebDriver::create($host, $capabilities, 5000);
$driver->manage()->timeouts()->implicitlyWait(15); //隐性设置15秒
$driver->get('http://www.atool.org/useragent.php');
echo $driver->findElement(WebDriverBy::id('ua_code'))->getAttribute('value');
echo 'done!';

//关闭浏览器
$driver->quit();
?>
```

```
#### 2.修改代理IP
```

```
<?php
namespace Facebook\WebDriver;
use Facebook\WebDriver\Remote\RemoteWebDriver;
use Facebook\WebDriver\Remote\WebDriverCapabilityType;
require_once('vendor/autoload.php');

header("Content-Type: text/html; charset=UTF-8");
```

$host = 'http://localhost:4444/wd/hub'; // this is the default
$ip = '115.225.2.3:8998'; //设置代理IP
$capabilities = array(WebDriverCapabilityType::BROWSER_NAME => 'chrome',

```
                    WebDriverCapabilityType::PROXY => array('proxyType' => 'manual',
                    'httpProxy' => $ip, 'sslProxy' => $ip));
```

$driver = RemoteWebDriver::create($host, $capabilities, 5000);
$driver->manage()->timeouts()->implicitlyWait(15); //隐性设置15秒
$driver->get('https://www.baidu.com/');
$driver->findElement(WebDriverBy::id('kw'))->sendKeys('ip');
$driver->findElement(WebDriverBy::id('su'))->click();
echo 'done!';
//关闭浏览器
//$driver->quit();

?>

使用总结

> 使用selenium时会遇到很多的坑，这里分享一些经验给大家。

1.wait ~ until的使用

> 要勤使用wait~until，由于我们打开的网页有时候会比较慢时，查找调用某个元素时就比较容易报NoSuchElementException异常。或者有时候一些JS是动态生成的，也需要用到wait~until，不然也会报错。

2.switchToEndWindow勤使用

> switchToEndWindow是我们自定义的一个方法，由于我们使用selenium访问某个网页，然后又点击了其中一个链接，谁知道这个链接是否含有target="_blank"呢。做爬虫类程序尤为如此。所以我们增加一个方法，每次点击了某个链接跳转后，执行一下switchToEndWindow方法。

```
<?php
//切换至最后一个window
function switchToEndWindow($driver){
```

```
$arr = $driver->getWindowHandles();
foreach ($arr as $k=>$v){
    if($k == (count($arr)-1)){
        $driver->switchTo()->window($v);
    }
}
```

}
?>

```
#### 3.异常处理
> 使用selenium过程中，会碰到各种意想不到的报错。比如做爬虫过程中，都要用try包含起来处理异常，这样可以防止一旦异常报错后，终止了程序的执行。那么在测试的每个节点的健康状态，可在数据库中记录查询。

#### 4.定位异常解决
> 元素在网页第一次加载后，就会确定他的坐标，当我们进行了某种操作，改变了宽度或高度，则很容易引起相关元素的坐标改变，从而报错。
> 报错信息：Element is not clickable at point (284, 11).
> 解决保存就是在调用click或submit方式之前先调用sendKeys方法，让其重绘坐标
```

$elemA->sendKeys('xxx')->click();
$elemB->sendKeys('xxx')->submit();

```
> 有时候，以上使用sendKeys方式还是没办法解决问题时，则很可能是因为网页中含有浮动DIV，导致各个元素定位变化了。
> 这个时候只需要找到那个浮动的DIV，隐藏掉就可以了。
```

$js = <<<js
var nav = document.getElementsByClassName("nav_m");
nav[0].style.display = 'none';
js;
$driver->executeScript($js);

```
>如果以上两种方式结合都还有问题的情况下，就建议用JS来解决。
```

//设置屏幕滚动到当前元素
$elems = $driver->findElements(WebDriverBy::className('n'));
foreach ($elems as $elem){

```
if(CommonUtil::contain($elem->getText(), '下一页')){
        $elem->sendKeys('xxx'); //设置焦点
}
```

}

$js = <<<js
var next = document.getElementsByClassName('n');
for (i = 0; i < next.length; i++) {
if(next[i].innerHTML == '下一页>'){
//next[i].click();
next[i].style.backgroundColor = "red";
}
}
js;

$driver->executeScript($js); //JS执行点击下一页

```
#### 5.执行时间的设置
> 使用自动化测试或爬虫程序，往往程序执行的时间会比较长。为了防止timeout，我们需要设置如下
```

```
set_time_limit(0);
ignore_user_abort(true);
```