

本文主要介绍如何针对指定的接口进行巡检封装

## 接口巡检

response事件监听可以对任意响应数据进行监控,

### 1-工具创建

新建: src/utis/Patrol.js

⚠: 此时判断条件时状态为200, 真正进行监听时, 可以设置为不为200

```
module.exports = {  
  //接口响应巡检  
  patrolResp:async function(response, apis) {  
    const url_list = apis instanceof Array ? apis : [apis];  
    if (url_list.indexOf(response.url()) !== -1 && response.status() === 200) {  
      response.text().then(text => {  
        console.log(response.request().resourceType() + '=' + response.url() +  
",返回值=" + text);  
      });  
    }  
  }  
}
```

### 2-巡检接口

test/automation/cases/Login/LoginBlogTest.js

引入依赖

```
const patrol = require('../../../../src/utis/Patrol');
```

登录后进行接口巡检:

```
//接口巡检  
const api1="https://a1.cnblogs.com/group/B1";  
const api2="https://www.cnblogs.com/ajax/wechatshare/getconfig?  
url=https%3A%2F%2Fw.cnblogs.com%2F";  
const list=[api1,api2];  
await page.on('response', async response => {  
  await patrol.patrolResp(response,list);  
});
```

完整代码LoginBlogTest.js:

```
const {describe,it,before,after,afterEach}=require('mocha');  
const Base=require('../../../../Base');
```

```

const {assert}=require('chai');
const patrol = require('../../../../src/utils/Patrol');

describe('博客园',function () {
  before(Base.before);
  // after(Base.after);
  afterEach(Base.afterEach);
  this.timeout(40000);
  it('Case1-登陆', async function () {
    //1. 登陆: 博客园
    await page.goto("https://account.cnblogs.com/signin");
    //2. 点击: 密码登陆
    await page.waitForSelector('#mat-tab-label-0-0 > div',
{timeout:2*1000}).then(ele=>{ele.click()}).catch(e=>{console.log(e)})
    //3. 输入: 用户名
    await page.focus('#mat-input-0');
    await page.type('#mat-input-0','UI自动化');
    //4. 输入: 密码
    await page.focus('#mat-input-1');
    await page.type('#mat-input-1','520bokeyuan');
    //5. 勾选: 记住我 (可选)
    // await page.click('#mat-checkbox-1 > label > span.mat-checkbox-inner-
container');
    //6. 点击: 登陆
    await page.waitForSelector('body > app-root > app-sign-in-layout > div > div >
app-sign-in > app-content-container > div > div > div > form > div > button',
{timeout:2*1000}).then(ele=>{ele.click()}).catch(e=>{console.log(e)})
    //接口巡检
    const api1="https://a1.cnblogs.com/group/B1";
    const api2="https://www.cnblogs.com/ajax/wechatshare/getconfig?
url=https%3A%2F%2Fw.cnblogs.com%2F";
    const list=[api1,api2];
    await page.on('response', async response => {
      await patrol.patrolResp(response,list);
    });
    //等待响应返回200
    await page.waitForResponse(response => response.status() === 200);
    await page.waitForTimeout(2000);
    //7. 进入: 我的
    await page.goto("https://home.cnblogs.com/u/2910916");
    await page.waitForTimeout(2000);
    //8. 用户头像, 并截图
    const avatar=await page.$('#user_profile_block > table > tbody > tr > td:nth-
child(1) > div > img');
    await avatar.screenshot({
      path: './test/report/screenshot/avatar.png',
      type: 'png'
    })
    //9. 验证园号=2910916

```

```

    const name = await page.$eval('#user_profile > li:nth-child(2) > span:nth-child(2)', el => el.textContent);
    assert.equal(name, '2910916');
  })
})

```

运行后：

The screenshot shows an IDE with a project named 'OpenPatrol'. The file explorer on the left shows a directory structure including 'test', 'automation', 'cases', 'Login', and 'LoginBlogTest.js'. The main editor displays the content of 'LoginBlogTest.js', which contains a series of Cypress commands for testing a login page. The commands include waiting for the page to load, typing a username and password, clicking a login button, and asserting the response. The test is named 'Case1-登陆' (Case1-Login). The bottom panel shows the test results, indicating that the test passed successfully. The output of the test is visible in the console, showing the response from the login endpoint.

```

18  await page.type('#mat-input-0', 'UI自动化');
19  //4. 输入: 密码
20  await page.focus('#mat-input-1');
21  await page.type('#mat-input-1', '520bokeyuan');
22  //5. 勾选: 记住我 (可选)
23  // await page.click('#mat-checkbox-1 > label > span.mat-checkbox-inner-container');
24  //6. 点击: 登陆
25  await page.waitForSelector('body > app-root > app-sign-in-layout > div > div > app-sign-in > app-content-container > div');
26  //接口巡检
27  const api1="https://a1.cnblogs.com/group/B1";
28  const api2="https://www.cnblogs.com/ajax/wechatshare/getconfig?url=https%3A%2F%2Fw.cnblogs.com%2F";
29  const list=[api1,api2];
30  await page.on('response', async response => {
31    await patrol.patrolResp(response, list);
32  });
33  //等待响应返回200
34  await page.waitForResponse( urlOrPredicate: response => response.status() === 200);
35  await page.waitForTimeout( milliseconds: 2000);
36  //7. 进入: 我的
37  await page.goto("https://home.cnblogs.com/u/2910916");
38  await page.waitForTimeout( milliseconds: 2000);

```

Run: LoginBase x 博客园 Case1-登陆 x Tests passed: 1 of 1 test - 11s 144ms

Test Results 11s 144ms

博客园 Case1-登陆 11s 144ms

Case1-登陆 11s 144ms

博客园 Case1-登陆 before running

xhr=https://a1.cnblogs.com/group/B1,返回值={"B1":""}

xhr=https://www.cnblogs.com/ajax/wechatshare/getconfig?url=https%3A%2F%2Fw.cnblogs.com%2F,返回值={"appId":"wxa033f083bd791"}

博客园 Case1-登陆 afterEach running

说明：接口的定义可以通过文件的形式统一管理，后续UI自动化的PO设计模式时会进行优化，这里仅做例子。