实验 07：使用 Ajax 实现异步数据请求

一、实验目的

1、掌握 AJAX 的核心原理，理解其对“页面无刷新交互”的关键作用；

2、学会使用 XHR 和 Fetch 发送 GET/POST 请求，处理响应数据；

3、结合 DOM 操作，实现无刷新数据交互功能，提升用户体验。

二、实验学时

2 学时

三、实验类型

综合性

四、实验需求

1、硬件

每人配备计算机 1 台，建议优先使用个人计算机开展实验。

2、软件

安装 Visual Studio Code，以及 Edge 浏览器。

3、网络

本地主机能够访问互联网和实验中心网络。

4、工具

无。

五、实验任务

1、使用 XHR 获取远程用户数据并渲染列表

2、使用 Fetch 实现用户名异步验证

3、调用互联网接口实现天气预报

六、实验内容及步骤

1、使用 XHR 获取远程用户数据并渲染列表

步骤 1：创建 HTML 结构

XML/HTML

```
<h3>远程用户列表（XHR获取）</h3>
<ul id="userList" class="user-list"></ul>
```

步骤 2：实现 CSS 样式，示例如下。

CSS

```
.user-list {
  list-style: none; /* 去掉默认列表符号 */
  padding: 0;
  margin-top: 10px;
}
.user-list li {
  padding: 8px 12px;
  border-bottom: 1px solid #eee; /* 底部边框分隔 */
  font-size: 14px;
}
.user-list li:last-child {
  border-bottom: none; /* 最后一项去掉边框 */
}
```

步骤 3：实现 JavaScript 代码，示例如下。

JavaScript

```javascript
// 1. 获取DOM容器
const userList = document.getElementById('userList');

// 2. 创建XHR对象
const xhr = new XMLHttpRequest();

// 3. 配置请求：GET方法 + 远程用户接口（JSONPlaceholder提供免费测试数据）
xhr.open('GET', 'https://jsonplaceholder.typicode.com/users');

// 4. 监听请求完成事件（状态码2xx表示成功）
xhr.onload = function() {
  if (xhr.status >= 200 && xhr.status < 300) {
    // 解析JSON响应数据
    const users = JSON.parse(xhr.responseText);
    // 渲染用户列表
    renderUserList(users);
  } else {
    console.error('请求失败:', xhr.statusText);
    userList.innerHTML = '<li>数据加载失败，请重试</li>';
  }
};

// 5. 监听请求错误事件（如网络中断）
xhr.onerror = function() {
  console.error('网络请求出错');
  userList.innerHTML = '<li>网络连接失败</li>';
};

// 6. 发送GET请求（无请求体，参数填null）
xhr.send();

// 辅助函数：渲染用户列表
function renderUserList(users) {
  // 清空容器原有内容
  userList.innerHTML = '';
  // 遍历用户数据，创建列表项
  users.slice(0, 5).forEach(user => { // 只渲染前5个用户
    const li = document.createElement('li'); // 创建<li>节点
    li.textContent = `${user.name} - ${user.email}`; // 设置文本内容（姓名+邮箱）
    userList.appendChild(li); // 将<li>添加到<ul>中
```

```
41   });
42 }
```

2、使用 Fetch 实现用户名异步验证

步骤 1：创建 HTML 结构

XML/HTML

```
<h3>用户名异步验证（Fetch）</h3>
<form id="usernameForm" class="form">
  <div class="form-group">
    <label for="username">用户名：</label>
    <input type="text" id="username" placeholder="请输入用户名（≥3位）" required>
    <span id="usernameError" class="error-msg hidden">用户名已存在</span>
  </div>
  <button type="submit" class="btn">检查可用性</button>
</form>
```

步骤 2：定义 CSS 样式

CSS

```css
.form {
  margin-top: 20px;
  padding: 15px;
  border: 1px solid #eee;
  border-radius: 8px;
}
.form-group {
  margin-bottom: 10px;
}
.error-msg {
  color: red;
  font-size: 12px;
  margin-left: 10px;
}
.hidden {
  display: none;
}
.btn {
  padding: 6px 12px;
  cursor: pointer;
  background: #2196F3;
  color: white;
  border: none;
  border-radius: 4px;
}
```

步骤 3：实现 JavaScript 逻辑

JavaScript

```javascript
// 1. 获取DOM元素
const form = document.getElementById('usernameForm');
const usernameInput = document.getElementById('username');
const errorSpan = document.getElementById('usernameError');

// 2. 模拟远程用户列表（实际可替换为真实API）
// 注意：此处用JSONPlaceholder的用户接口，检查用户名是否存在
const checkUsernameUrl = 'https://jsonplaceholder.typicode.com/users?username=';

// 3. 监听表单提交事件
form.addEventListener('submit', async (e) => {
  e.preventDefault(); // 阻止表单默认提交
  const username = usernameInput.value.trim();

  if (!username) {
    errorSpan.textContent = '请输入用户名';
    errorSpan.classList.remove('hidden');
    return;
  }

  try {
    // 发送GET请求，检查用户名是否存在
    const response = await fetch(`${checkUsernameUrl}${username}`);
    const users = await response.json(); // 解析JSON响应

    if (users.length > 0) {
      // 用户名已存在
      errorSpan.textContent = '用户名已存在';
      errorSpan.style.color = 'red';
      errorSpan.classList.remove('hidden');
    } else {
      // 用户名可用
      errorSpan.textContent = '用户名可用';
      errorSpan.style.color = 'green';
      errorSpan.classList.remove('hidden');
    }
  } catch (err) {
    console.error('验证失败:', err);
    errorSpan.textContent = '验证出错，请重试';
    errorSpan.style.color = 'red';
    errorSpan.classList.remove('hidden');
```

```
42    }
43 });
```

3、调用互联网接口实现天气预报

步骤 1：注册和风天气

和风天气是国内常用的天气 API 服务，提供免费的天气数据接口（需注册获取 API Key）。以下是接入步骤：

（1）注册与获取 API Key

访问和风天气开发者平台，注册账号并登录；

进入「控制台」→「创建应用」，填写应用信息（如“天气预报实验”），获取 API Key（后续请求需用到）；

查看「API 文档」→「实时天气」接口（链接），了解请求参数与返回格式。

（2）选择目标城市

和风天气通过 Location ID 标识城市（如北京：101010100、上海：101020100）。可从和风天气 Location 列表获取目标城市的 ID。

步骤 2：HTML 结构搭建

XML/HTML

```
1 <h3>实时天气预报（和风天气API）</h3>
2 <div id="weatherInfo" class="weather-card">
3   <p class="loading">正在加载天气数据...</p>
4 </div>
```

步骤 3：定义 CSS 样式

CSS

```css
.weather-card {
  margin-top: 20px;
  padding: 20px;
  border: 1px solid #eee;
  border-radius: 8px;
  text-align: center;
  font-size: 14px;
  min-height: 150px; /* 避免内容为空时容器塌陷 */
}
.weather-card .loading {
  color: #666;
}
.weather-card img {
  width: 60px;
  height: 60px;
  margin: 10px auto;
  display: block;
}
.error-msg {
  color: #ff4d4f;
  font-size: 14px;
}
```

步骤 4：JavaScript 逻辑实现

JavaScript

```javascript
// 1. 配置参数（替换为你的API Key和目标城市Location ID）
const API_KEY = '你的和风天气API Key'; // ⚠ 必须替换为自己的Key
const LOCATION_ID = '101010100'; // 北京的Location ID（可替换为其他城市）
const NOW_WEATHER_URL = `https://api.qweather.com/v7/weather/now?location=${LOCATION_ID}&key=${API_KEY}`;

// 2. 获取DOM元素
const weatherInfo = document.getElementById('weatherInfo');

// 3. 发送Fetch请求获取天气数据
fetch(NOW_WEATHER_URL)
  .then(response => {
    if (!response.ok) { // 检查HTTP状态码（非2xx表示失败）
      throw new Error(`请求失败: ${response.status}`);
    }
    return response.json(); // 解析JSON响应
  })
  .then(data => {
    // 检查API返回的状态（和风天气用code字段表示请求结果）
    if (data.code !== '200') {
      throw new Error(`天气数据获取失败: ${data.message}`);
    }
    // 渲染天气数据
    renderWeather(data);
  })
  .catch(err => {
    console.error('天气请求出错: ', err);
    // 显示错误信息
    weatherInfo.innerHTML = `<p class="error-msg">${err.message || '获取天气失败，请重试'}</p>`;
  });

// 4. 渲染天气信息的辅助函数
function renderWeather(data) {
  // 解析数据（参考和风天气实时天气接口返回格式）
  const city = data.location.name; // 城市名
  const temp = data.now.temp + '°C'; // 温度（°C）
  const weatherText = data.now.text; // 天气状况（如“晴”）
  const iconCode = data.now.icon; // 天气图标代码（如"101"）
  const iconUrl = `https://a.hecdn.net/img/common/icon/202306d/${iconCode}.png`; // 图标URL

```

```
40   // 拼接HTML渲染
41   weatherInfo.innerHTML = `
42     <p>城市：${city}</p>
43     <p>温度：${temp}</p>
44     <p>天气：${weatherText}</p>
45     <img src="${iconUrl}" alt="${weatherText}">
46   `;
47 }
```

七、实验考核

本实验考核采用【实验随堂查】方式开展。

每个实验完成后，在实验课上通过现场演示的方式向实验指导教师进行汇报，并完成现场问答交流。

每个实验考核满分 100 分，其中实验成果汇报 60 分，现场提问交流 40 分。

实验考核流程：

（1）学生演示汇报实验内容的完成情况，实验指导老师现场打分。

（2）指导老师结合实验内容进行提问，每位学生提问 2-3 个问题，根据回答的情况现场打分。

（3）实验考核结束后，进行公布成绩。