Day29笔记

|  |
| --- |
| AJAX学习笔记  1. 什么是AJAX？  AJAX（Asynchronous JavaScript And XML）是一种用于创建交互式网页应用的开发技术，旨在通过异步数据传输（HTTP请求）来增强网页与服务器之间的数据交换。其特点是能够在不重新加载整个页面的情况下，对页面的部分内容进行更新。  AJAX技术是现代Web开发中的核心部分。它的异步特性使得用户能够在不感知页面刷新的情况下进行操作，提高了用户体验。尽管名称中包含XML，但实际应用中JSON更为普遍，因为它更轻量且易于解析。  2. HTTP请求与响应  HTTP（超文本传输协议）是Web浏览器与服务器之间通信的基础。AJAX通过JavaScript对HTTP的请求和响应进行操作，从而实现数据的动态加载。  3. XMLHttpRequest对象  `XMLHttpRequest`对象是实现AJAX功能的关键，它用于在后台与服务器交换数据。常用的方法包括：  `open(method, url, async)`：设置请求类型（如GET或POST）、目标URL以及是否异步。  `setRequestHeader(header, value)`：设置HTTP请求头。  `send(data)`：发送请求数据。  var request = new XMLHttpRequest();  request.open('GET', 'https://api.example.com/data', true);  request.send();  request.onreadystatechange = function() {  if (request.readyState === 4 && request.status === 200) {  console.log(request.responseText);  }  };  `XMLHttpRequest`提供了一个简洁的接口用于与服务器进行通信，尤其是在处理异步请求时，它避免了页面的阻塞。然而，随着Fetch API的引入，`XMLHttpRequest`的使用逐渐减少，但它仍然是理解AJAX的基础。  4. AJAX的封装  为了简化AJAX的使用，通常会对其进行封装。  var myAjax = {  get: function(url, params, success, error) {  var httpRequest = new XMLHttpRequest();  httpRequest.open('GET', url + "?" + new URLSearchParams(params).toString(), true);  httpRequest.send();  httpRequest.onreadystatechange = function() {  if (httpRequest.readyState === 4 && httpRequest.status === 200) {  success(JSON.parse(httpRequest.responseText));  } else if (httpRequest.readyState === 4) {  error(httpRequest.responseText);  }  };  }  };  封装AJAX有助于提高代码的可复用性和可维护性。通过封装，我们可以统一处理请求的细节，减少冗余代码，同时也可以更容易地应对跨浏览器的兼容性问题。  5. 实战操作：GET与POST请求  GET请求  var request = new XMLHttpRequest();  request.open('GET', 'https://api.example.com/data?param=value', true);  request.send();  POST请求  var request = new XMLHttpRequest();  request.open('POST', 'https://api.example.com/data', true);  request.setRequestHeader('Content-type', 'application/json;charset=UTF-8');  request.send(JSON.stringify({key: 'value'}));  GET请求适用于获取数据且不会改变服务器状态的操作，POST则更适合提交数据。理解两者的区别并合理选择，可以提高请求的效率与安全性。  总结  AJAX通过异步数据传输大大提升了Web应用的交互性，理解并熟练掌握其基本原理和常用操作是前端开发的重要技能。通过对`XMLHttpRequest`的深入理解和灵活应用，可以实现更加高效和友好的用户体验。  日期2024.8.23 |