Day05笔记

|  |
| --- |
| 1. Web存储与通信  1.1 Web存储  Web存储是HTML5新增的一项功能，它允许在用户浏览器中存储数据，主要分为两种类型：  localStorage: 用于存储持久化的数据，数据不会过期，除非手动删除。  sessionStorage: 用于存储会话级的数据，当页面会话结束（浏览器关闭）后，数据会被清除。  存储数据:  localStorage.setItem('key', 'value');  sessionStorage.setItem('key', 'value');  读取数据:    let data = localStorage.getItem('key');  let data = sessionStorage.getItem('key');    删除数据:    localStorage.removeItem('key');  sessionStorage.removeItem('key');    1.2 Web通信  Web通信技术允许在客户端与服务器之间建立持续的连接，以便实时发送和接收数据。  1.2.1 Ajax  Ajax是一种在不重新加载整个页面的情况下，通过与服务器进行少量数据交换来更新页面的技术。  用法：  let xhr = new XMLHttpRequest();  xhr.open('GET', 'url', true);  xhr.onreadystatechange = function() {  if (xhr.readyState === 4 && xhr.status === 200) {  console.log(xhr.responseText);  }  };  xhr.send();  1.2.2 Fetch API  Fetch API是现代替代Ajax的标准，它使用Promise处理网络请求。  用法：  fetch('url')  .then(response => response.json())  .then(data => console.log(data))  .catch(error => console.error('Error:', error));  2. WebSocket  WebSocket是一种在单个TCP连接上进行全双工通信的协议，它允许服务端主动向客户端推送数据。适用于聊天应用、实时数据更新等场景。  用法：  let socket = new WebSocket('ws://url');  socket.onopen = function() {  console.log('WebSocket 已经建立连接');  socket.send('Hello Server!');  };  socket.onmessage = function(event) {  console.log('Message from server:', event.data);  };  socket.onclose = function() {  console.log('WebSocket connection closed');  };  socket.onerror = function(error) {  console.error('WebSocket error:', error);  };  3. JS地理位置  JavaScript提供了`Geolocation API`，允许获取用户的地理位置信息。  用法：  if (navigator.geolocation) {  navigator.geolocation.getCurrentPosition(  position => {  console.log('Latitude:', position.coords.latitude);  console.log('Longitude:', position.coords.longitude);  },  error => console.error('Error Code:', error.code, 'Message:', error.message)  );  } else {  console.error('Geolocation is not supported by this browser.');  }  4. Bootstrap及可视化  Bootstrap是一个流行的前端框架，用于快速开发响应式网站和Web应用。  4.1 基本结构  Bootstrap基于12列栅格系统，提供了多种响应式布局和组件。  <div class="container">  <div class="row">  <div class="col-md-6">Column 1</div>  <div class="col-md-6">Column 2</div>  </div>  </div>  4.2 常用组件  按钮:  <button type="button" class="btn btn-primary">Primary Button</button>  导航栏:  <nav class="navbar navbar-expand-lg navbar-light bg-light">  <a class="navbar-brand" href="">Navbar</a>  </nav>  4.3 Bootstrap与可视化  Bootstrap与数据可视化库（如ECharts）可以结合使用，提供美观的UI和交互。  5. ECharts  ECharts是一个开源的、基于JavaScript的数据可视化库，支持丰富的图表类型，如折线图、柱状图、散点图等。  用法：  <div id="main" style="width: 600px;height:400px;"></div>  <script>  var myChart = echarts.init(document.getElementById('main'));  var option = {  title: { text: 'ECharts example' },  tooltip: {},  xAxis: { data: ['A', 'B', 'C', 'D'] },  yAxis: {},  series: [{  type: 'bar',  data: [5, 20, 36, 10]  }]  };  myChart.setOption(option);  </script>  6. Ant Design (Antd)  Ant Design是一个用于开发企业级UI的React组件库，提供了一套丰富的组件，支持快速构建用户界面。  用法：  import { Button, DatePicker } from 'antd';  import 'antd/dist/antd.css';  function App() {  return (  <div>  <Button type="primary">Primary Button</Button>  <DatePicker />  </div>  );  }  export default App;  6.1 常用组件  按钮:  <Button type="primary">Primary Button</Button>  表格:  import { Table } from 'antd';  const columns = [{ title: 'Name', dataIndex: 'name', key: 'name' }];  const data = [{ key: '1', name: 'John Brown' }];  <Table columns={columns} dataSource={data} />  日期2024.8.30 |