HTML5存储

课程介绍

1. 本地存储 - Web Storage

2. 本地存储 - IndexedDB

3. 总结

分析存储需求

1. 移动端网络环境因素

- 数据响应慢，体验下降

- 节省流量=节省金钱

分析存储需求

2. 追求原生体验

- 离线使用应用

- 存储应用相关资源、数据

Cookie 可行否？

Cookie 的劣势

-存储大小限制，仅 4kb 左右

-单个域名下的数量限制，50个左右

-污染请求头，浪费流量

本地存储 - Web Storage

认识 localStorage 和 sessionStorage

localStorage 和 sessionStorage

1. 相同的使用方法

2. 不同的存储时效

API

1. 使用 setItem 方法设置存储内容

2. 使用 getItem 方法获取存储内容

3. 使用 removeItem 方法删除存储内容

API

4. 使用 clear 方法清除存储内容

5. 使用 length 属性获取存储内容个数

6. 使用 key 方法获取存储内容

不同的存储时效

localStorage 存储会持久化

sessionStorage 存储会在网页会话结束后失效

不同的存储容量

localStorage 容量一般在 2 – 5Mb 左右

sessionStorage 存储容量不一，部分浏览器不设限

使用 Storage 时的注意点

使用 Storage 时的注意点

1. 存储容量超出限制

- 抛出 QuotaExceededError 异常

存储值时应使用 try catch 避免异常未捕获

使用 Storage 时的注意点

2. 存储类型的限制

- 仅能存储字符串

- 注意类型转换

使用 Storage 时的注意点

3. sessionStorage 失效机制

- 刷新页面并不能使 sessionStorage 失效

- 相同 URL 不同标签页不能共享 sessionStorage

实现一个游戏小案例

实现一个带有过期机制的 localStorage

实现一个带有过期机制的 localStorage

1. 实现的功能需求

- 可以设置数据的存储时间

- 失效后清除数据

Web Storage 的优化

性能与存储容量大小无关，与读取次数有关

- 减少读取 item 次数

- 单个 item 中尽可能多的存储数据

如何创建数据库和“表”

如何创建数据库和“表”

1. indexedDB.open 创建数据库

2. indexedDB.createObjectStore 创建“表”

设置主键的两种方法

1. 设置自增主键 - {autoIncrement: true}

2. 取数据中字段作为主键 - {keyPath: 字段名}

关于“表”的增删查改

如何使用事务获取表

调用 IDBDatabase.transaction 方法会返回一个 IDBTransaction 对象，它含有一个 objectStore 方法可以让用户通过指定模式操作数据库中的“表”

indexedDB -> transaction -> objectStore

事务的模式

1. 读写模式 - readwrite

2. 只读模式（默认） - readonly

关于“表”的增删查改的相关方法

1. 增加数据 – IDBObjectStore.add

2. 获取数据 – IDBObjectStore.get

3. 获取所有数据 – IDBObjectStore.getAll

关于“表”的增删查改的相关方法

4. 修改数据 – IDBObjectStore.put

5. 删除数据 – IDBObjectStore.delete

6. 清除所有数据 – IDBObjectStore.clear

IDBRequest 对象

1. 使用 IDBRequest.onsuccess 执行查询完成回调

2. 使用 IDBRequest.result 获取查询结果

3. 使用 IDBRequest.onerror 执行查询失败回调

关于索引的使用

如何创建索引

IDBObjectStore.createIndex

- indexName: 索引名称

- keyPath: 索引字段，可以为空或者数组（type array）

- optionParameters: 索引配置参数

使用索引的好处

1. 可以使用存储记录中的值进行检索

2. 索引自动更新

3. 索引数据自动排序

关于游标的使用

如何创建游标

IDBObjectStore/IDBIndex.openCursor

- range: 指定游标范围

- direction: 游标的方向

IDBRange 对象

key range 取值表

Range	Code
All keys ≤ x	upperBound(x)
All keys < x	upperBound(x, true)
All keys ≥ y	lowerBound(y)
All keys > y	lowerBound(y, true)
The key = z	only(z)

IDBRange 对象

key range 取值表

Range	Code
All keys ≥ x && ≤ y	bound(x, y)
All keys > x && < y	bound(x, y, true, true)
All keys > x && ≤ y	bound(x, y, true, false)
All keys ≥ x && < y	bound(x, y, false, true)

设置游标的 direction

游标的 direction 取值

- next: 顺序查询
- nextunique: 顺序唯一查询
- prev: 逆序查询
- prevunique: 逆序唯一查询

使用游标进行数据查询

使用游标带来的好处

1. 可以查询指定数据集范围

2. 拥有逆序遍历能力

索引和游标的结合使用

索引和游标结合带来的好处

1. 索引按值搜索+游标范围遍历

2. 索引排序+游标按序遍历

IndexedDB 的优劣

IndexedDB 与 Web Storage 比较

优势

- 存储类型更加丰富
- 可以在 Worker 中使用
- 条件搜索优势明显
- 存储容量更大

IndexedDB 与 Web Storage 比较

劣势

- 学习曲线略陡峭

- 兼容性问题略严重

IndexedDB 的兼容性问题

1. IOS8 & 9 中 webview 不支持 indexedDB

2. Firefox 单次存储 Blob 数据超 50Mb 会抛出异常

3. IE 10 & 11 有部分子功能未实现

跨域存储限制

知识点回顾

知识点回顾

1. localStorage 和 sessionStorage

- 存取次数影响性能

- 只能存储字符串类型数据

- 注意存储至上限时的异常处理

知识点回顾

2. indexedDB

- 存储容量大

- 需要考虑兼容性

- 适合处理复杂数据查询

再会