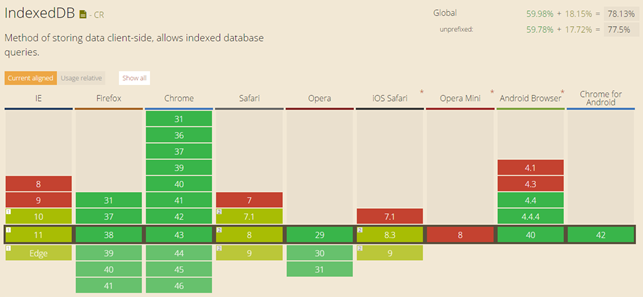
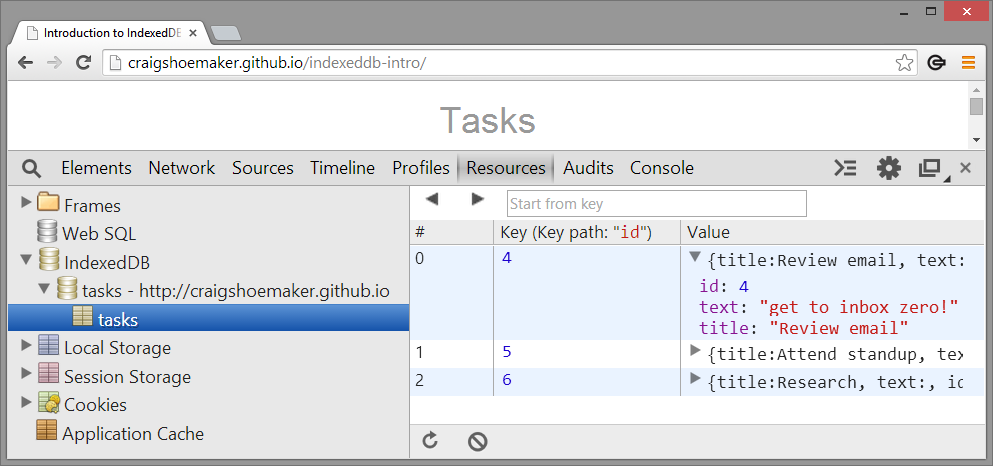
IndexDB, WebSQL

IndexDB



IndexDB – документ-ориентированная база данных, которая существует на стороне клиента.

Данные хранятся как отдельные объекты.



Объектно-ориентированное данные сохраняются в так называемых object stores и все действия выполняются по запросу в пределах определенной транзакции.

***Object Stores***

Object Stores эквивалентны таблицам в реляционных базах данных.

Object Stores включают в себя 1 или несколько индексов которые служат в качестве ключа и предоставляют более быстрый способ доступа к данным.

Уникальный идентификатор может браться из данных, а может генерироваться искусственно.

На картинке 1 task data хранится в tasks объекте и имеет in-line key – поле id

***Executed within a transaction scope.***

Транзакция может включать 1 или несколько Object Stores, это обозначается при ее определении.

Драфт W3C определяет четыре типа трансакций:

READ\_ONLY,

READ\_WRITE,

SNAPSHOT\_READ

VERSION\_CHANGED.   
READ\_ONLY – служит, как следует из названия, для чтения. Блокирует транзакции других типов.

READ\_WRITE – служит для изменения данных, дожидается завершения всех конкурирующих транзакций над выбранным объектом, блокирует все прочие транзакции и выполняется.

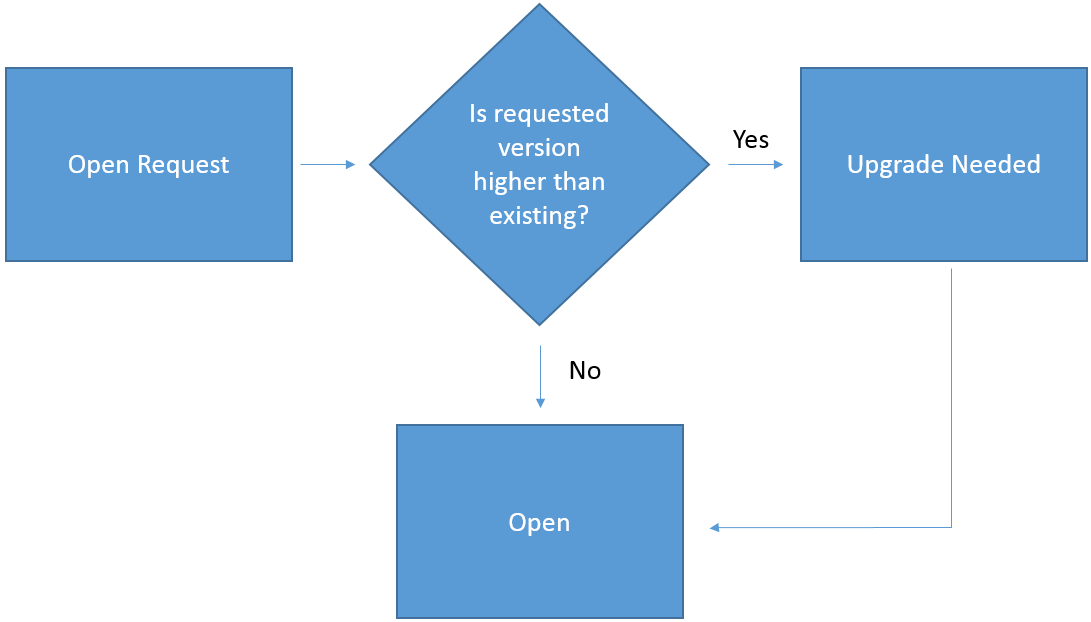
VERSION\_CHANGE – транзакция, которая дожидается завершения всех прочих трансакций, блокирует доступ к объектам данных для всех и выполняется. Только в этой трансакции можно создавать, удалять или изменять объекты данных.

***Request-Based***

Для выполнения каждой операции с базой данных, сначала нужно создать запрос на эту операцию. Когда запрос выполнился, можно обрабатывать события и ошибки, которые появляются в результате запроса.

IndexedDB API выполняются асинхронно.

На рисунке изображен жизненный цикл open request



IndexedDB следует использовать когда:

- большая часть пользователей использует браузер, который поддерживает IndexedDB;

- нужно хранить большие объемы информации;

- нужно обеспечить быстрый поиск по данным;

- требуется поддержка транзакций.

Для того чтоб обеспечить доступ к IndexedDB нужно установить значения переменных window.indexedDB, window.IDBTransaction и window.IDBKeyRange для разных браузеров:

window.indexedDB = window.indexedDB || window.mozIndexedDB || window.webkitIndexedDB || window.msIndexedDB;

window.IDBTransaction = window.IDBTransaction || window.webkitIDBTransaction || window.msIDBTransaction;

window.IDBKeyRange = window.IDBKeyRange || window.webkitIDBKeyRange ||

window.msIDBKeyRange;

IndexedDB доступны как online так и offline. Эта база данных хранится локально и содержит уникальное в пределах домена имя, и не доступна с других доменов.

***Opening***

При открытии базы данных следует указывать ее название и версию.

|  |  |
| --- | --- |
|  | var openRequest = indexedDB.open("test",1); |

Открытие базы данных является асинхронной операцией, для обработки результата ее срабатывания, есть такие обработчики:

* success
* error
* upgradeneeded
* blocked

Событие upgradeneeded срабатывает когда пользователь впервые открывает базу данных, либо когда его версия – устарела.

В обработчике success объект события, который передается параметром коллбека содержит базу данных в поле target.result

Создать object stores можно только с помощью события upgradeneeded

Важно помнить, что для обновления уже существующего object stores, следует обновить версию базы данных.

Получить список записанных в базу данных object stores можно с помощью свойства e.target.result - objectStoreNames

## Что можно сделать с объектами данных: Создать – createObjectStore() Удалить – deleteObjectStore() Назначить трансакцию – transaction()

|  |  |
| --- | --- |
|  | var idbSupported = false;  var db;    document.addEventListener("DOMContentLoaded", function(){        if("indexedDB" in window) {          idbSupported = true;      }        if(idbSupported) {          var openRequest = indexedDB.open("test\_v2",1);            openRequest.onupgradeneeded = function(e) {              console.log("running onupgradeneeded");              var thisDB = e.target.result;                if(!thisDB.objectStoreNames.contains("firstOS")) {                  thisDB.createObjectStore("firstOS");              }            }            openRequest.onsuccess = function(e) {              console.log("Success!");              db = e.target.result;          }            openRequest.onerror = function(e) {              console.log("Error");              console.dir(e);          }        }    },false); |

***Keys***

Каждый object store должен иметь ключ.

IndexedDB содержит несколько типов ключей (создавать в ручную, генерировать уникальный, использовать уникальные поля в качестве ключей keypath).

## *Adding Data*

## IndexedDB хранит объекты.

## Работа с данными требует использования транзакций.

При создании транзакции участвует 2 аргумента:

- список таблиц, к которым будет обращение

- тип операции (readonly, readwrite

|  |  |
| --- | --- |
|  | //Assume db is a database variable opened earlier  var transaction = db.transaction(["people"],"readwrite"); |
|  | var store = transaction.objectStore("people"); |
|  | //Define a person  var person = {      name:name,      email:email,      created:new Date()  }    //Perform the add  var request = store.add(person,1); |

***Reading Data***

|  |  |
| --- | --- |
|  | var transaction = db.transaction(["test"], "readonly");  var objectStore = transaction.objectStore("test");    //x is some value  var ob = objectStore.get(x);    ob.onsuccess = function(e) {    } |
|  | | db.transaction(["test"], "readonly").objectStore("test").get(X).onsuccess = function(e) {} | |

***Reading More Data***

Для чтения большого объема данных используются курсоры

|  |  |
| --- | --- |
|  | var transaction = db.transaction(["test"], "readonly");  var objectStore = transaction.objectStore("test");    var cursor = objectStore.openCursor();    cursor.onsuccess = function(e) {      var res = e.target.result;      if(res) {          console.log("Key", res.key);          console.dir("Data", res.value);          res.continue();      }  } |

***Индексы***

Индексы помогают извлекать данные, основываясь на их значения, или их уникальность.

|  |  |
| --- | --- |
|  | var objectStore = thisDb.createObjectStore("people", { autoIncrement:true });  //first arg is name of index, second is the path (col);  objectStore.createIndex("name","name", {unique:false});  objectStore.createIndex("email","email", {unique:true}); |

|  |  |
| --- | --- |
|  | var transaction = db.transaction(["people"],"readonly");  var store = transaction.objectStore("people");  var index = store.index("name");    //name is some value  var request = index.get(name); |

Основное отличие IndexedDB от LocalStorage в том, что IndexedDB хранит непосредственно объекты, в то время как LocalStorage – строки

IndexedDB имеет асинхронные API, а в LocalStorage – синхронные

IndexedDB может хранить 50MB, LocalStorage 5000 Кб.

WebSQL



WebSQL – больше не поддерживается

База данных хранится в виде реляционных таблиц, для запроса к данным используется SQL

Среди основных методов работы с этой базой данных выделяют: openDatabase, transaction, executeSql

Links

<http://blogs.shephertz.com/2014/01/14/html5-learn-how-to-use-indexeddb/>

<http://www.codemag.com/Article/1411041>

<http://www.girliemac.com/blog/2014/07/03/indexeddb/>

http://blog.teamtreehouse.com/create-your-own-to-do-app-with-html5-and-indexeddb