Client-Side Storage in HTML5

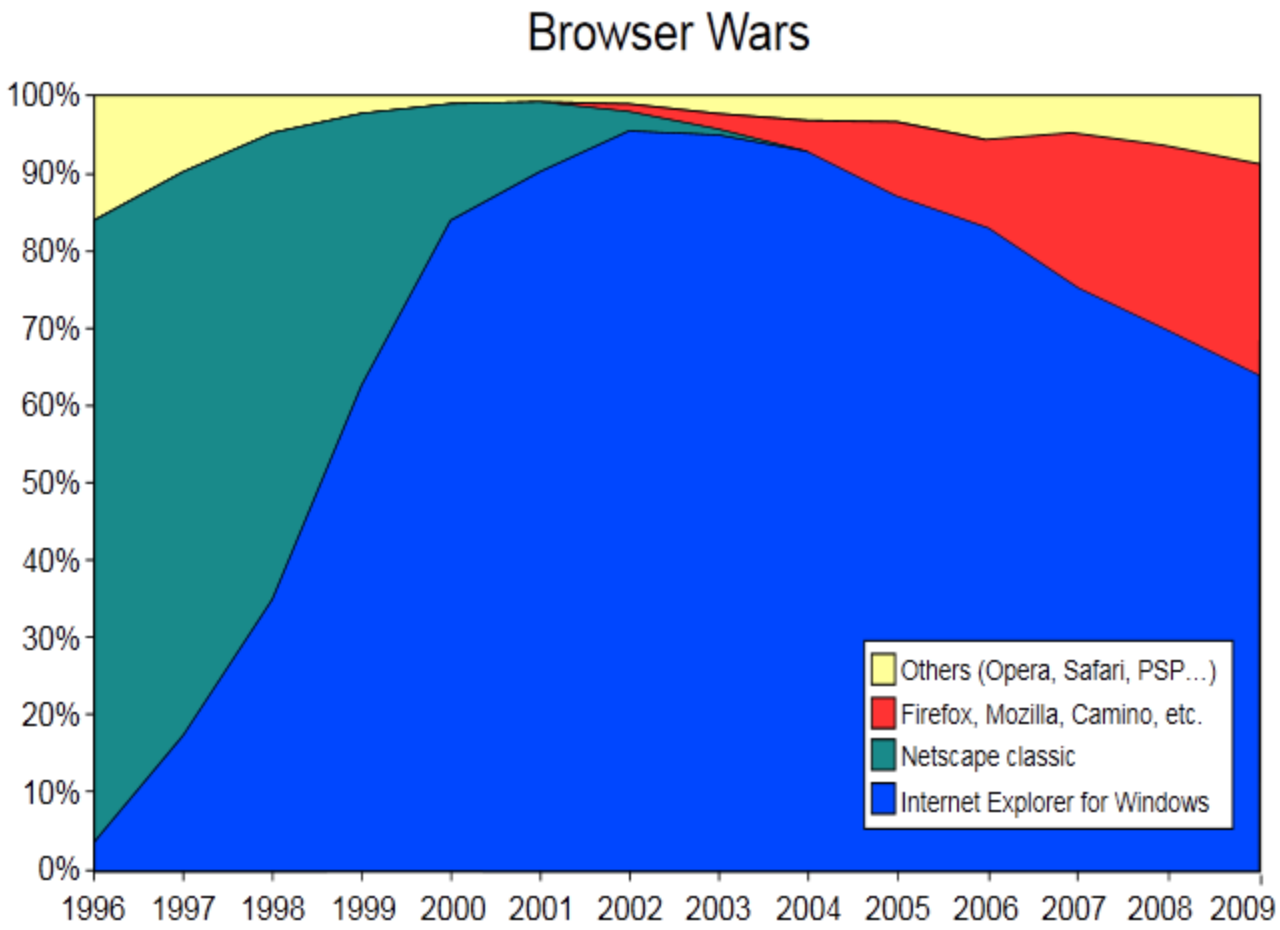
작성자: 전용우(mixed@nhn.com)

작성년월일: 2011/02/22

목차

1. Storage History

1.1 The First Browser War(95')



1.2 userData


(도메인 마다 64 KB)

```
<html>

<head>
<style type="text/css">
.storeuserData {
    behavior: url(#default#userData);
}
</style>
<script type="text/javascript">
function fnSaveInput(){
   var oPersist=oPersistForm.oPersistInput;
   oPersist.setAttribute("sPersist",oPersist.value);
   oPersist.save("oXMLBranch");
}
function fnLoadInput(){
   var oPersist=oPersistForm.oPersistInput;
   oPersist.load("oXMLBranch");
   oPersist.value=oPersist.getAttribute("sPersist");
}
</script>
</head>

<body>

<form id="oPersistForm">
    <input class="storeuserData" type="text" id="oPersistInput">
    <input type="button" value="Load" onclick="fnLoadInput()">
    <input type="button" value="Save" onclick="fnSaveInput()">
</form>

</body>

</html>
```

1.3 Local Shared Objects(2002’ Flash6)

(도메인 마다 100 KB)

1.3 Google Gear(2007')

SQLite을 기반으로 하는 SqlDB
데이터 제한 없음.

Web Storage
Web Database

2. Web Storage(aka DOM Storage)

1.사용하기 쉽다.

2.용량이 크다.(5kb/5mb)

3.서버에 데이터를 전송하지 않는다.

4.유효기간이 없다.

Internet Explorer 8

Firefox 3.5

Safari 4

Google Chrome 4

Opera 10.50

Iphone 2.0

Android 2.0

sessionStorage는 브라우저를 닫으면 사라지고

localStorage는 지속적으로 남음.

2.3. localStorage메서드

```
localStorage.setItem("bar",”foo”); //값 저장
var foo = localStorage.getItem("bar"); //값 반환

var foo = localStorage["bar"];
localStorage["bar"] = foo;

localStorage.removeItem("bar"); // 값 삭제
localStorage.key();//  마지막 key값
localStorage.key(1);//  마지막에서 두 번째 key값
localStorage.key(2);//  없으면 null
localStorage.clear(); // 모두 삭제
localStorage.length ; // 저장된 수
```

3. Web Database

SQLLite을 기반으로 하는 브라우저에 내장된 DB.

3.2. 지원현황

Safari 4.0

Chrome 4.0

Opera 10.50

Iphone 3.0

Android 2.0

3.2. Database 생성

```
openDB : function(){
        this.db =
            openDatabase('AddressBook', '1.0', '데이터 베이스', (5 * 1024 * 1024));
}
```

3.3. Table 생성.

```
createTable : function(){
  this.db.transaction(function(tx){
        tx.executeSql('CREATE TABLE contacts
                    (id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
                     name TEXT, email TEXT, cell TEXT)');
  });
},
```

3.4. Row 생성.

```
insertRow : function(){
    this.db.transaction(function(tx){
        tx.executeSql('INSERT INTO contacts (name, email, cell)
                        VALUES (?, ?, ?)',
                       ["mixed","mixed@nhn.com","000000"],
                       function(transaction, resultSet){
                           console.log("create row. id = %s",resultSet.insertId);
                            //생성된 id
                       });
    });
},
```

3.4. Row Fetch.

```
selectRow : function(){
        this.db.transaction(function(tx){
                tx.executeSql('SELECT * FROM contacts',
                        [],
                        function(transaction, resultSet){
                            for (var i = 0; i < resultSet.rows.length; i++) {
                                console.log(resultSet.rows.item(i));
                            }
                        });
        });
},
```

3.5. Row 수정.

```
updateRow : function(){
        this.db.transaction(function(tx){
                tx.executeSql('UPDATE contacts set cell=? Where id=?;',
                        ["aaaaaa","3"],
                        function(transaction, resultSet){
                            console.log resultSet.rowsAffected);
                            //변경된 row수
                        });
        });
},
```

3.6. Row 삭제.

```
deleteRow : function(){
    this.db.transaction(function(tx){
        tx.executeSql('DELETE from contacts where id=?;',
                      ["3"],
                      function(transaction, resultSet){
                          console.log("delete row");
                      });
    });
}
```

3.7. WebDB의 문제점

1. SQLLite는 표준 SQL92를 대부분 지키긴 했으나 중요한 일부분을 빼먹었음.
2. SQLLite는 표준이 아니라 HTML5의 표준정책에 위반됨.
3. SQLLite가 Upgrade시 문제가 있음.

4. IndexedDB(aka Web Simple DB)

클라이언트에 대용량 데이터를 저장하고

인덱스를 이용하여 보다 빠르게 찾는 API

localStorage는 용량이 작고 구조화된 데이터를 다루기엔 부족.

IE (플러그인)

Firefox4 beta10

Chrome 11

Safari, Opera 미 구현

동기접근(미지원-WebWork을 이용)

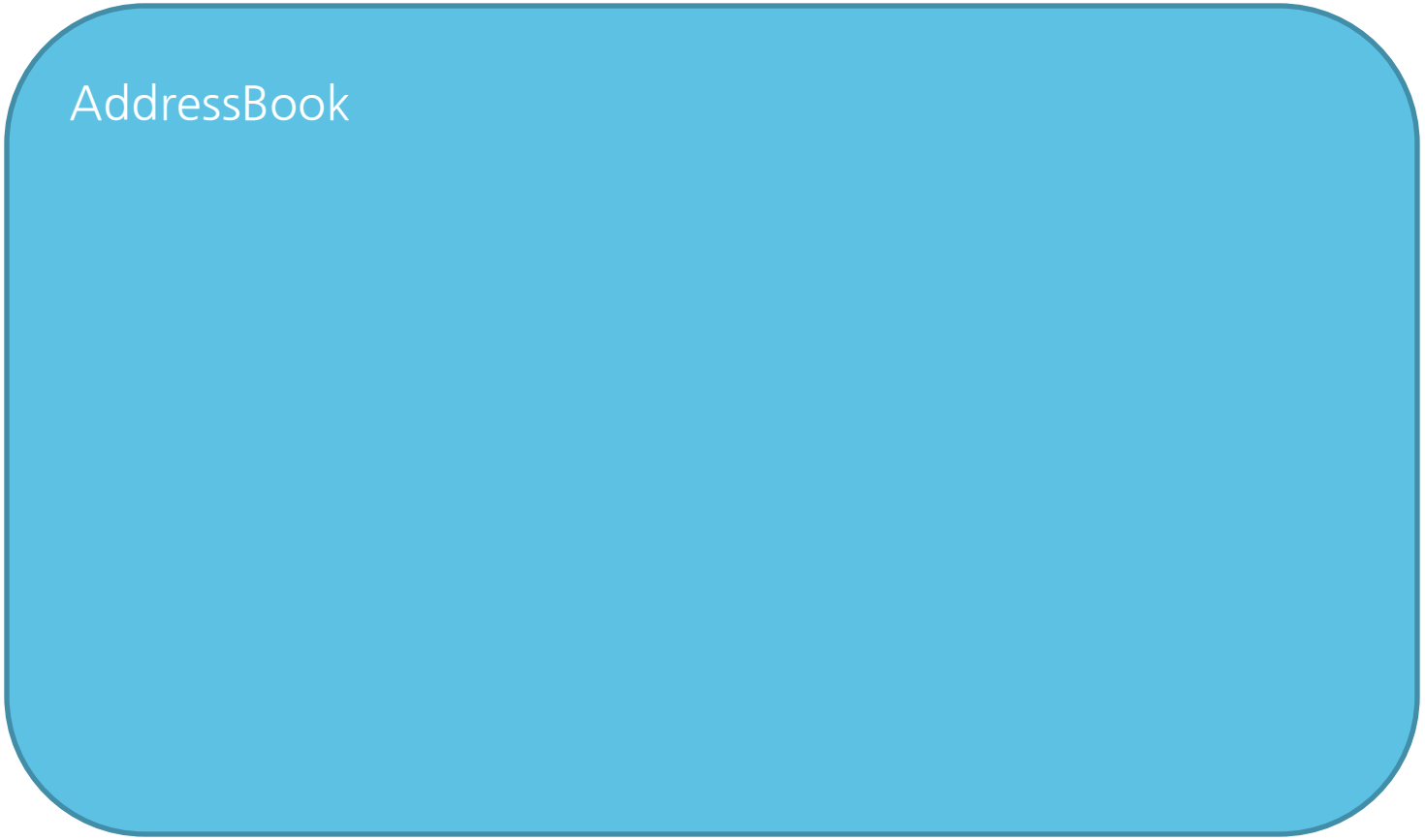
비동기 접근(지원 firefox4.b8부터)

- 각자 실행되고 callback로 대답한다.

API설명

DB생성

4.5. IndexedDB의 API - DB생성

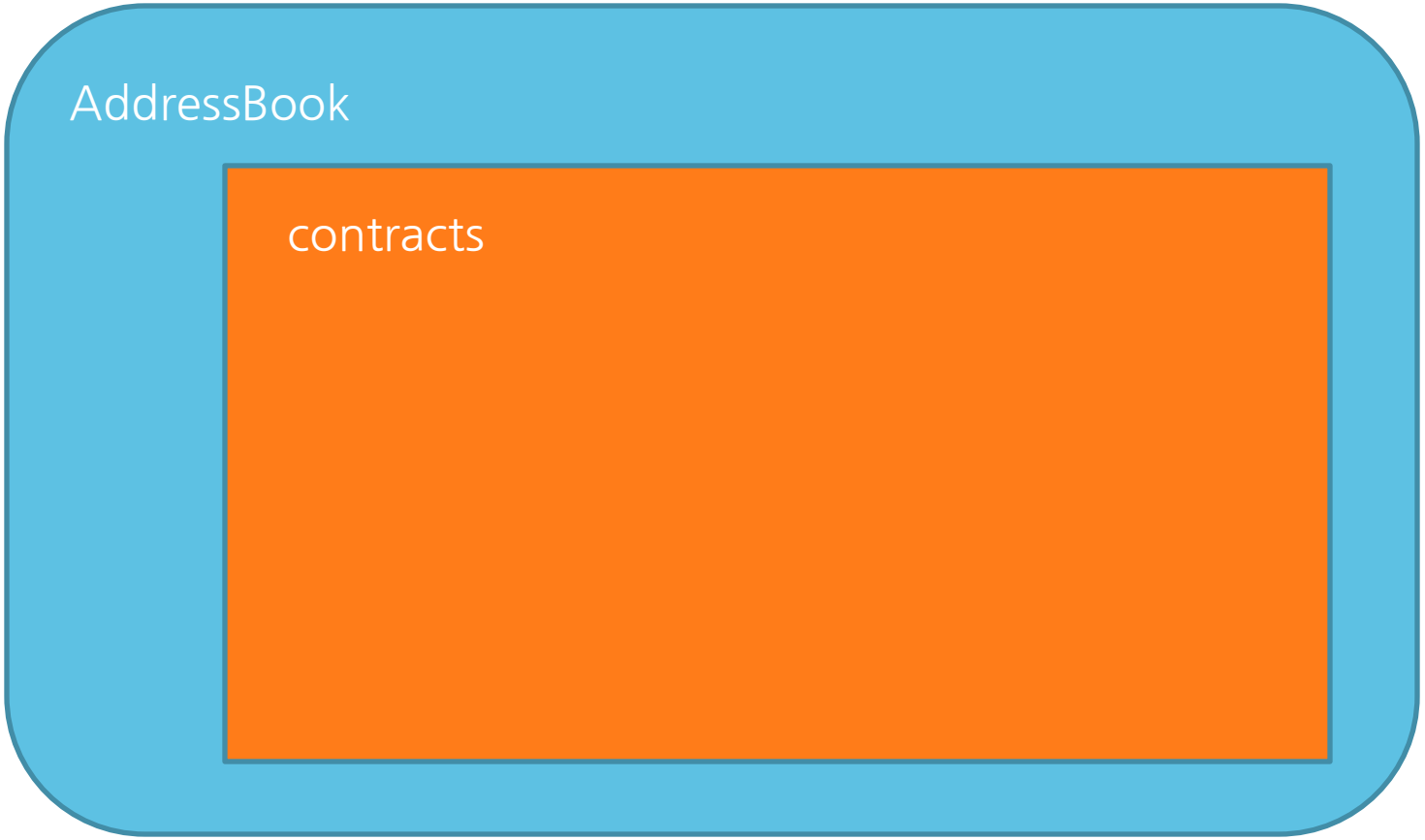


4.5. IndexedDB의 API - DB생성

```
this.db =
window.mozIndexedDB.open(
        "AddressBook",//이름
        "책 관련 데이터 베이스“//설명
);
```

Object Store

4.6. IndexedDB의 API - Object Store생성



```
//반드시 버전이 변경되어야만 ObjectStore생성 및 인덱스 등을
변경할 수 있음.
var store = this.resultIDBDatabase.createObjectStore(
    "contacts", //이름
    {"keyPath": "id"} //pk
    // in-line keys - 자동생성, out-of-line keys - 직접 입력
    //false  //자동 증가. firefox4 default false, w3c true
);
```

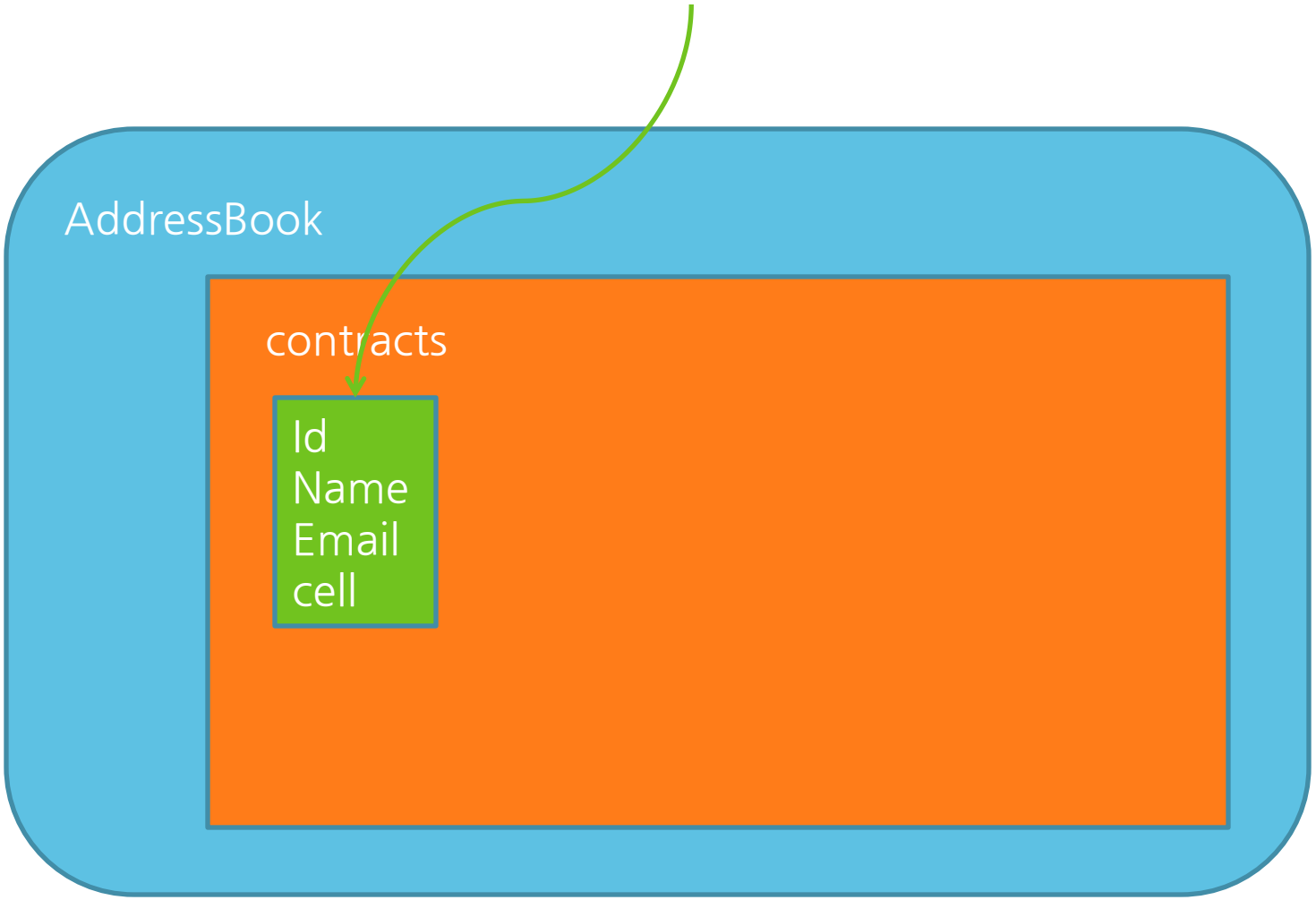
4.6. IndexedDB의 API - Object Store생성

```
var request = this.resultIDBDatabase.setVersion(
                    parseInt(Math.random() * 100));
request.onsuccess = function(){
    var store = this.resultIDBDatabase.createObjectStore(
        "contacts",
        {"keyPath": "id"}
    );
}
```

```
var request = this.resultIDBDatabase.setVersion(
                    parseInt(Math.random() * 100));
request.onsuccess = function(){
    var store = request.result.objectStore("contacts");
     store.createIndex(
                    "CellPhone", // 인덱스명
                    "cell"      // 인덱스 속성
                    false        // 중복 여부
    );
}.bind(this);
```

Data manipulate

4.7. IndexedDB의 API - Data manipulate

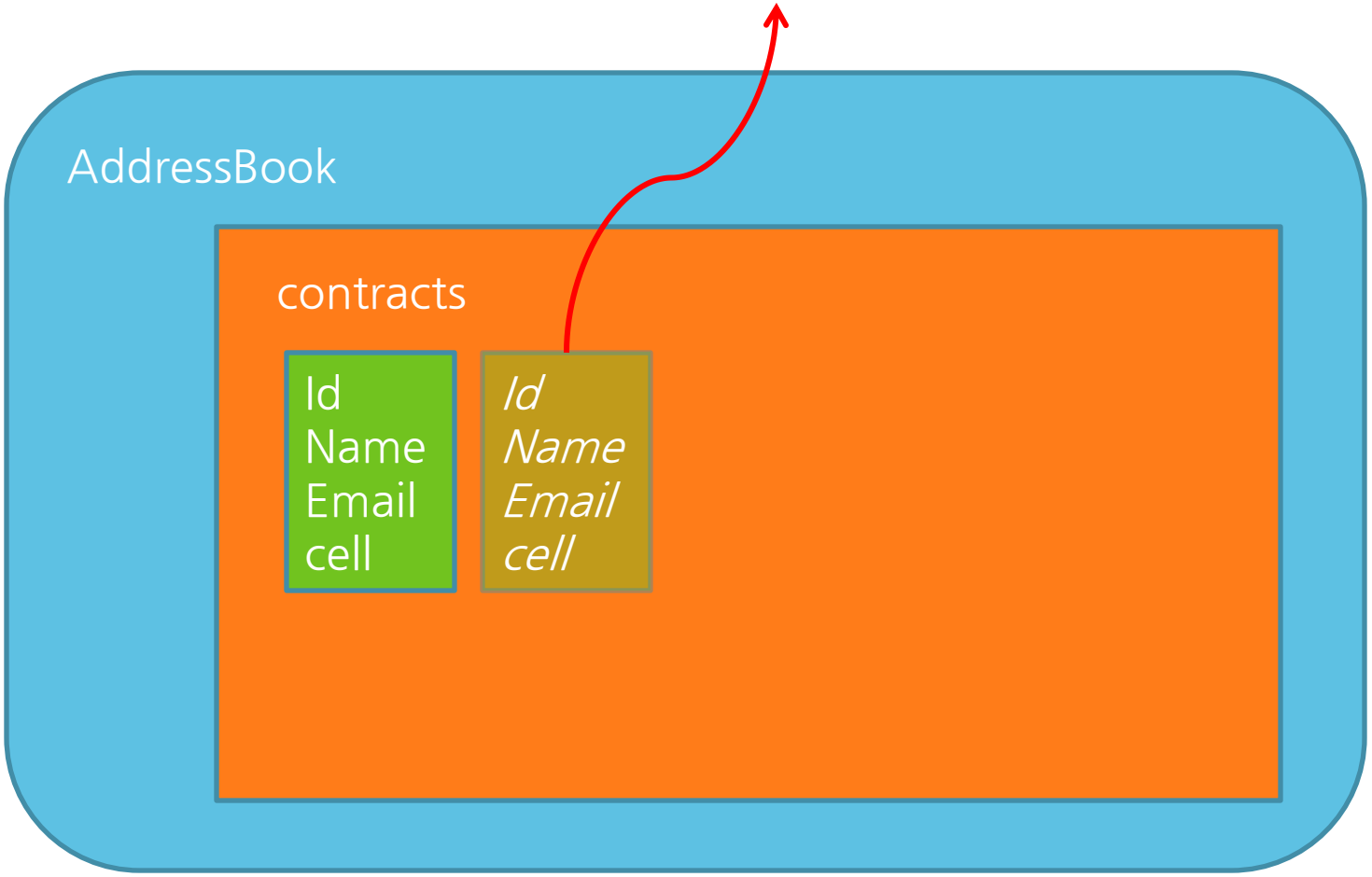


```
var writeTransaction = this.resultIDBDatabase.transaction(
                                [ "contacts" ],//잠글 object store명
                                IDBTransaction.READ_WRITE
                                // Lock type (READ_ONLY, READ_WRITE)
                                //,30 timeout 밀리세컨드
);
var store = writeTransaction.objectStore( "contacts" );
var id = parseInt(Math.random() * 100);
var writeRequest = store.add({
    "name" : "mixed", "email" : "mixed@nhn.com", "cell": id, "id" : id
});
```

4.7. IndexedDB의 API - Data manipulate

```
writeRequest.onsuccess = function(e){
    console.log (writeRequest.result);//id
}.bind(this);
writeRequest.ontimeout = function ( e ) {
    alert("timeout");
   writeTransaction.abort();
};
```

4.7. IndexedDB의 API - Data manipulate



4.7. IndexedDB의 API - Data manipulate

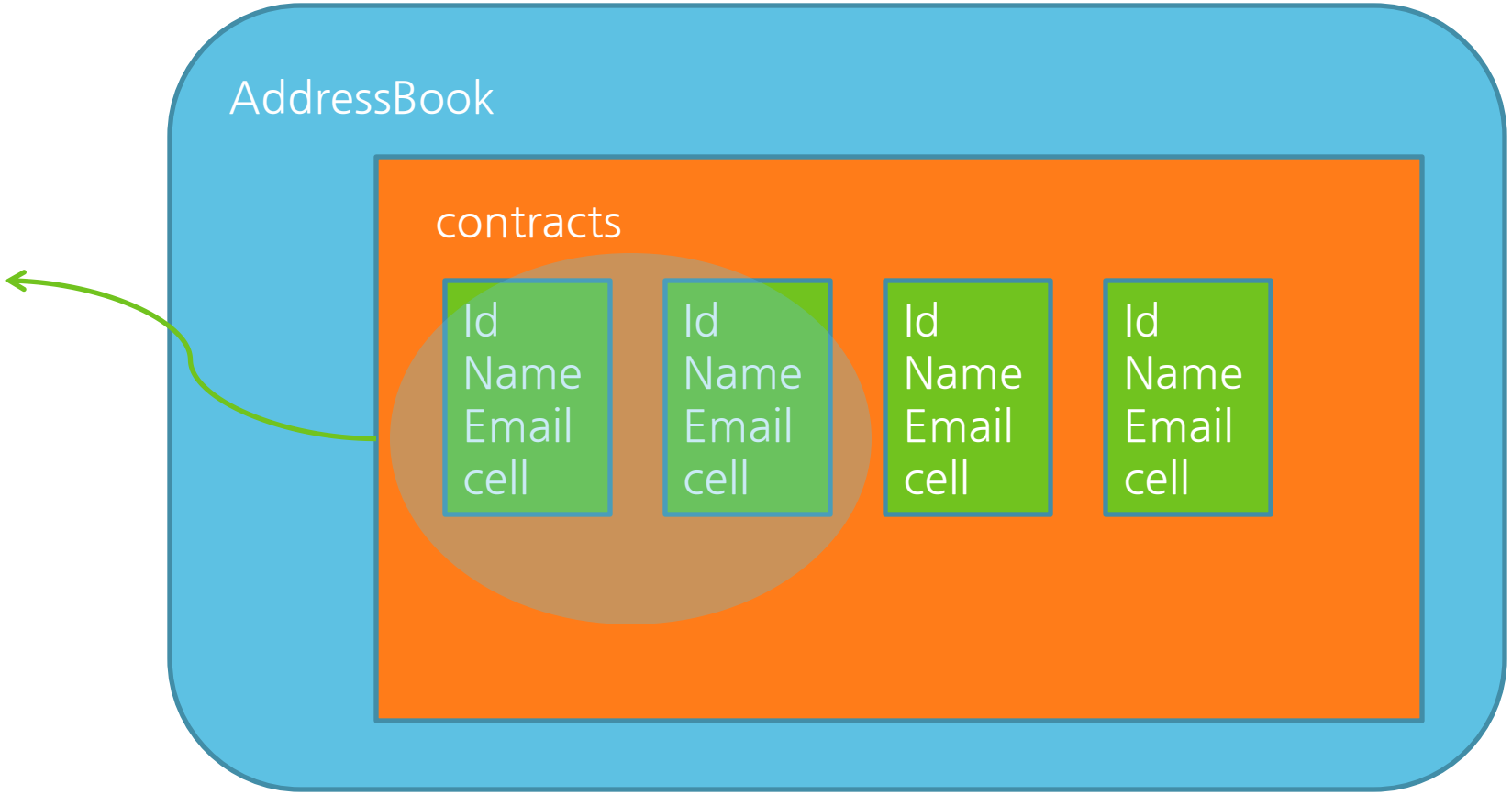
```
var writeTransaction = this.resultIDBDatabase.transaction(
                                [ "contacts" ],
                                IDBTransaction.READ_WRITE  );

var store = writeTransaction.objectStore( "contacts" );

var writeRequest = store.delete(id);
```

4.7. IndexedDB의 API - Data manipulate

```
var request = store.get(id);
request.onsuccess = function(event){
    request.result.email = "i.nevalose@gmail.com";
    var request2 = store.put(request.result);
    request2.onsuccess = function(){
        console.log("modified");
    }
}.bind(this);
```

Data Fetch

4.8. IndexedDB의 API - Data Fetch



4.8. IndexedDB의 API - Data Fetch

```
var store = readTransaction.objectStore( "contacts" );
var request = store.get(this.id);

request.onsuccess = function(event){
    console.log(request.result); //데이터
};
```

4.8. IndexedDB의 API - Data Fetch

```
var store = readTransaction.objectStore( "contacts" );
var request = store.getAll();

request.onsuccess = function(){
    console.log(request.result);  //data 배열
}
```

4.8. IndexedDB의 API - Data Fetch

```
var store = readTransaction.objectStore( "contacts" );
var readCursor = store.openCursor();

readCursor.onsuccess = function ( e ) {
  if ( readCursor.result ) {
    console.log( readCursor.result); //데이터
    readCursor.result["continue"]();
  }else{
     console.log("end");
  }
};
```

4.8. IndexedDB의 API - Data Fetch

```
var store = readTransaction.objectStore( "contacts" );
var bounds = new IDBKeyRange.bound(
  0,  // 시작
  10,  // 끝
  false, // 시작 포함 여부 근데 반대로 되어 있음.
  true //  끝 포함 여부
);
var readCursor = store.openCursor( bounds );
//IDBCursor.NEXT  - 정방향
//IDBCursor.NEXT_NO_DUPLICATE - 정방향 이면서 중복 제외
//IDBCursor.PREV - 역방향
//IDBCursor.PREV_NO_DUPLICATE- 역방향 이면서 중복 제외
```

4.8. IndexedDB의 API - Data Fetch

```
var index = store.index("CellPhone");
var bounds = new IDBKeyRange.bound(10, 90, false, false);
var readCursor = index.openCursor( bounds );
```

4.8. IndexedDB의 API - Data Fetch

```
var manipulateCursor = store.openCursor();
manipulateCursor.onsuccess = function ( e ) {
  if ( manipulateCursor.result ) {
     if(manipulateCursor.result.value.id%2 == 0){
            manipulateCursor.result.delete();
     }else{
            manipulateCursor.result.value.name = "전용우";
            manipulateCursor.result.update(
                  manipulateCursor.result.value);
     }
    manipulateCursor.result["continue"]();
  }else{  console.log("end"); }
};
```

4.9. IndexedDB의 장점

1. WebDB는 명시적으로 transaction을 해야 하지만 IndexedDB는 오브젝트 단위로 자동 transaction.
2. IndexedDB는 오브젝트를 넣을 수 있음.
3. cursor을 이용하여 보다 빠르게 얻어올수 있음.
4. *lower* layer라 사용자에게 더 많은 옵션을 선택할 수 있게 한다.

Reference

http://www.w3.org/TR/IndexedDB/
http://diveintohtml5.org/storage.html
http://en.wikipedia.org/wiki/Browser_wars#The_first_browser_war
http://msdn.microsoft.com/en-us/library/ms531424(VS.85).aspx
http://hacks.mozilla.org/2010/06/beyond-html5-database-apis-and-the-road-to-indexeddb/
http://hacks.mozilla.org/2010/06/comparing-indexeddb-and-webdatabase/
http://en.wikipedia.org/wiki/Web_Storage
http://code.google.com/p/indexeddb/
http://www.hotcleaner.com/web_storage.html
http://antmicro.com/blog/2010/07/async-indexeddb-js-api-test/
http://sites.google.com/site/usingindexeddb/
https://developer.mozilla.org/en/IndexedDB
http://mxr.mozilla.org/mozilla-central/source/dom/indexedDB/test/test_key_requirements.html?force=1#33
http://nparashuram.com/trialtool/index.html#example=/ttd/IndexedDB/moz_indexedDB.html&selected=#db&
http://html5doctor.com/introducing-web-sql-databases/
http://uxebu.com/blog/2011/01/17/indexeddb-updates-ff4-09b/
http://news.cnet.com/8301-30685_3-20000376-264.html?tag=mncol;title
http://blogs.msdn.com/b/interoperability/archive/2010/12/21/indexeddb-prototype-available-for-internet-explorer.aspx
http://blogs.msdn.com/b/interoperability/archive/2011/02/03/the-indexeddb-prototype-gets-an-update.aspx
http://mikewest.org/2010/12/intro-to-indexeddb
http://blog.submitmy.info/2010/04/html5-client-side-storage/
https://developer.mozilla.org/en/IndexedDB/IndexedDB_primer

Thank you.