Offline strategies for HTML5 web applications

Stephan Hochdörfer, bitExpert AG
Offline strategies for HTML5 web applications

About me

- Stephan Hochdörfer
- Head of IT at bitExpert AG, Germany
- enjoying PHP since 1999
- S.Hochdoerfer@bitExpert.de
- @shochdoerfer
Offline strategies for HTML5 web applications

Storing data on the client? Seriously?
Offline strategies for HTML5 web applications

How did we solve these issues in the past?
Offline strategies for HTML5 web applications

Cookies are tasty, but not awesome!
Offline strategies for HTML5 web applications

IE DHTML behaviours, not sweet!
Offline strategies for HTML5 web applications

Flash Cookies are yummie!
Offline strategies for HTML5 web applications

Google Gears made it right. Sort of.
Offline strategies for HTML5 web applications

Can I haz alternative?
Offline strategies for HTML5 web applications
to the rescue!
Offline strategies for HTML5 web applications

[...] we take the next step, announcing 2014 as the target for Recommendation.

Jeff Jaffe, Chief Executive Officer, World Wide Web Consortium
Offline strategies for HTML5 web applications

Mark Zuckerberg: Our Biggest Mistake Was Betting Too Much On HTML5

DREW OLANOFF

Tuesday, September 11th, 2012 75 Comments

Today, Mark Zuckerberg revealed that Facebook’s mobile strategy relied too much on HTML5, rather than native applications.

Not only was this a big mistake with mobile, but Zuckerberg says that its biggest mistake period was the focus on HTML5. This is the first time that the Facebook CEO has openly admitted this, but things are looking good for the new iOS native app. According to Zuckerberg, people are consuming twice as many feed stories since the update to the new iOS app, which is great.

The first half year has been a little bit slow on product, but for the next six months I expect a lot of really cool stuff.

This “really cool stuff” will probably have monetization in mind, as it’s very clear that mobile is the path to ad revenue for the company.

It’s extremely difficult for a company to nose-dive into an adoption of a particular set of tools and then quickly change course. I suspect that this is exactly what happened with Facebook and things are at least looking up.

Last year, Facebook’s CTO at the time, Bret Taylor, discussed the future of mobile, which at the time very much included HTML5. Here’s a snippet from that piece, where Taylor discusses the platform:

Does that mean an evolution away from Flash? After all, Flash dominates the market for the types of HTML5 games that Facebook is talking about. “Well it’s hard,” Taylor said about Flash specifically. When I laughed and noted he was giving the diplomatic answer, he assured me that it is something they think about a lot. “We want to be ahead of the curve and fill in the gaps when possible,” is how he ended up putting it.
Offline strategies for HTML5 web applications

When we started what became Sencha, we made a bet on the web - a bet that modern application development didn’t need anything except the browser, a great set of frameworks and a great set of tools. With those three weapons in hand, we knew developers could build applications that would delight users. The advent of HTML5 upped the game and it gave developers even more tools to let them treat the browser as an application development platform and not just a page rendering engine. Developers sprang at the opportunity and unleashed a torrent of apps — on both desktop and mobile — that leveraged the new HTML5 capabilities to build amazing applications using web standards.

So, when Mark Zuckerberg said HTML5 wasn’t ready, we took a little offense to the comment.
What does „offline“ mean?
What does „offline“ mean?

Application Cache vs. Offline Storage
App Cache for caching static resources

HTML Page:

```html
<!DOCTYPE html>
<html lang="en">
```
Offline strategies for HTML5 web applications

App Cache for caching static resources

HTML Page:

```html
<!DOCTYPE html>
<html lang="en" manifest="cache.manifest">

cache.manifest (served with Content-Type: text/cache-manifest):

CACHE MANIFEST

js/app.js
css/app.css
favicon.ico
http://someotherdomain.com/image.png
```
Offline strategies for HTML5 web applications

App Cache for caching static resources

CACHE MANIFEST
# 2013-07-25

NETWORK:
data.php

CACHE:
/main/home
/main/app.js
/settings/home
/settings/app.js
http://myhost/logo.png
http://myhost/check.png
http://myhost/cross.png
Offline strategies for HTML5 web applications

App Cache for caching static resources

CACHE MANIFEST
# 2013-07-25

FALLBACK:
/ /offline.html

NETWORK:
*
Offline strategies for HTML5 web applications

App Cache – Some gotchas!
App Cache – Some gotchas!

1. Files are always(!) served from the application cache.
App Cache – Some gotchas!

2. The application cache only updates if the content of the manifest itself has changed!
App Cache – Some gotchas!

3. If any of the files listed in the CACHE section can't be retrieved, the entire cache will be disregarded.
App Cache – Some gotchas!

4. If the manifest file itself can't be retrieved, the cache will be ignored!
App Cache – Some gotchas!

5. Non-cached resources will not load on a cached page!
App Cache – Some gotchas!

6. The page needs to be reloaded, otherwise the new resources do not show up!
App Cache – Some gotchas!

7. To avoid the risk of caching manifest files set expires headers!
Offline strategies for HTML5 web applications

App Cache – What to cache?

Yes:
- Fonts
- Splash image
- App icon
- Entry page
- Fallback bootstrap

No:
- CSS
- HTML
- Javascript
Offline strategies for HTML5 web applications

Storing dynamic data locally (in HTML5)
Offline strategies for HTML5 web applications

Example: Todolist application
Offline strategies for HTML5 web applications

Storing dynamic data locally (in HTML5)

Find the sources here:
github.com/bitExpert/html5-offline
Offline strategies for HTML5 web applications

Storing dynamic data locally (in HTML5)

Web Storage, Web SQL Database, IndexedDB, File API
Offline strategies for HTML5 web applications

Web Storage
Web Storage

Very convenient form of offline storage: simple key-value store
Web Storage: 2 different types

localStorage vs. sessionStorage
function add(item) {
    try {
        // for a new item set id
        if((typeof item.id === "undefined")
            || (null == item.id) || ("" == item.id)) {
            item.id = get_lastIndex() + 1;
        }

        // store object as string
        localStorage.setItem(item.id,
            JSON.stringify(item)
        );

        // update the index
        set_lastIndex(item.id);
    } catch(ex) {
        console.log(ex);
    }
}
function modify(item) {
  try {
    // store object as string
    localStorage.setItem(item.id, JSON.stringify(item));
  }
  catch(ex) {
    console.log(ex);
  }
}
Offline strategies for HTML5 web applications

Web Storage: Remove item

```javascript
function remove (id) {
    try {
        localStorage.removeItem(id);
    }
    catch(ex) {
        console.log(ex);
    }
}
```
function read() {
    try {
        var lastIdx = get_lastIndex();
        for (var i = 1; i <= lastIdx; i++) {
            if (null !== localStorage.getItem(i)) {
                // parse and render item
                var item = JSON.parse(localStorage.getItem(i));
            }
        }
    }
    catch (ex) {
        console.log(ex);
    }
}
Offline strategies for HTML5 web applications

Web Storage: Don`t like method calls?
Web Storage: Don`t like method calls?

```javascript
var value = "my value";

// method call
localStorage.setItem("key", value);

// Array accessor
localStorage[key] = value;

// Property accessor
localStorage.key = value;
```
Web Storage: Pro

Most compatible format up to now.
Web Storage: Con

The data is not structured.
Web Storage: Con

No transaction support!
Web Storage: Con

Lack of automatically expiring storage.
Web Storage: Con

Inadequate information about storage quota.
Offline strategies for HTML5 web applications

Web SQL Database
Offline strategies for HTML5 web applications

Web SQL Database

An offline SQL database based on SQLite, an general-purpose SQL engine.
Web SQL Database: Callback methods

```javascript
var onError = function(tx, ex) {
    alert("Error: " + ex.message);
};

var onSuccess = function(tx, results) {
    var len = results.rows.length;

    for(var i = 0; i < len; i++) {
        // render found todo item
        render(results.rows.item(i));
    }
};
```
Offline strategies for HTML5 web applications

Web SQL Database: Setup Database

// initialize the database connection

var db = openDatabase('todo', '1.0', 'Todo Database',
5 * 1024 * 1024);

db.transaction(function (tx) {
    tx.executeSql(
        'CREATE TABLE IF NOT EXISTS todo ' +
        '(id INTEGER PRIMARY KEY ASC, todo TEXT)',
        [],
        onSuccess,
        onError
    );
});
function add(item) {
    db.transaction(function(tx) {
        tx.executeSql(
            'INSERT INTO todo (todo) VALUES (?)',
            [item.todo],
            onSuccess,
            onError
        );
    });
}
Offline strategies for HTML5 web applications

Web SQL Database: Modify item

```javascript
function modify(item) {
    db.transaction(function(tx) {
        tx.executeSql('UPDATE todo SET todo = ? WHERE id = ?', 
                      [item.todo, item.id], 
                      onSuccess, 
                      onError
    );
});
}
```
function remove(id) {
    db.transaction(function (tx) {
        tx.executeSql(
            'DELETE FROM todo WHERE id = ?',
            [id],
            onSuccess,
            onError
        );
    });
}
Web SQL Database: Read items

```javascript
function read() {
    db.transaction(function (tx) {
        tx.executeSql('SELECT * FROM todo', [], onSuccess, onError);
    });
}
```
Web SQL Database: Pro

It's a SQL database within the browser!
Offline strategies for HTML5 web applications

Web SQL Database: Con

It's a SQL database within the browser!
Web SQL Database: Con

SQLite is sloooooow!
Web SQL Database: Con

The specification is no longer part of HTML5!
Offline strategies for HTML5 web applications

IndexedDB
IndexedDB

A nice compromise between Web Storage and Web SQL Database giving you the best of both worlds.
Offline strategies for HTML5 web applications

IndexedDB: Preparation

// different browsers, different naming conventions
var indexedDB = window.indexedDB ||
    window.webkitIndexedDB || window.mozIndexedDB ||
    window.msIndexedDB;

var IDBTransaction = window.IDBTransaction ||
    window.webkitIDBTransaction;

var IDBKeyRange = window.IDBKeyRange ||
    window.webkitIDBKeyRange;
IndexedDB: Create object store

```javascript
var db = null;
var request = indexedDB.open("todo");
request.onfailure = onError;
request.onsuccess = function(e) {
    db = request.result;
    var v = "1.0";
    if(v != db.version) {
        var verRequest = db.setVersion(v);
        verRequest.onfailure = onError;
        verRequest.onsuccess = function(e) {
            var store = db.createObjectStore(
                "todo",
                {
                    keyPath: "id",
                    autoIncrement: true
                }
            );
            e.target.transaction.oncomplete = function() {};
        }
    }
};
```
IndexedDB: Add item

```javascript
function add(item) {
    try {
        var trans = db.transaction(["todo"],
                                   IDBTransaction.READ_WRITE);

        var store = trans.objectStore("todo");
        var request = store.put({
            "todo": item.todo,
        });
    }
    catch(ex) {
        onError(ex);
    }
}
```
IndexedDB: Modify item

```javascript
function modify(item) {
  try {
    var trans = db.transaction(['todo'], IDBTransaction.READ_WRITE);
    var store = trans.objectStore('todo');
    var request = store.put(item);
  }
  catch (ex) {
    onError(ex);
  }
}
```
IndexedDB: Remove item

```javascript
function remove(id) {
  try {
    var trans = db.transaction(["todo"],
      IDBTransaction.READ_WRITE);

    var store = trans.objectStore("todo");
    var request = store.delete(id);
  }
  catch(ex) {
    onError(ex);
  }
}
```
function read () {
    try {
        var trans = db.transaction(['todo'],
            IDBTransaction.READ);

        var store = trans.objectStore('todo');
        var keyRange = IDBKeyRange.lowerBound(0);
        var cursorRequest = store.openCursor(keyRange);

        cursorRequest.onsuccess = function(e) {
            var result = e.target.result;
            if (!!result == false) {
                return;
            }
            // @TODO: render result.value
            result.continue();
        };
    } catch (ex) {
        onError(ex);
    }
}
Offline strategies for HTML5 web applications

File API
Offline strategies for HTML5 web applications

File API

FileReader API and FileWriter API
File API: Preparations

```javascript
var onError = function(e) {
    var msg = '';

    switch(e.code) {
        case FileError.QUOTA_EXCEEDED_ERR:
            msg = 'QUOTA_EXCEEDED_ERR'; break;
        case FileError.NOT_FOUND_ERR:
            msg = 'NOT_FOUND_ERR'; break;
        case FileError.SECURITY_ERR:
            msg = 'SECURITY_ERR'; break;
        case FileError.INVALID_MODIFICATION_ERR:
            msg = 'INVALID_MODIFICATION_ERR'; break;
        case FileError.INVALID_STATE_ERR:
            msg = 'INVALID_STATE_ERR'; break;
        default:
            msg = 'Unknown Error'; break;
    }

    alert("Error: " + msg);
};
```
Offline strategies for HTML5 web applications

File API: Preparations

// File system has been prefixed as of Google Chrome 12
window.requestFileSystem = window.requestFileSystem ||
    window.webkitRequestFileSystem;

window.BlobBuilder = window.BlobBuilder ||
    window.WebKitBlobBuilder;

var size = 5 * 1024*1024; // 5MB
Offline strategies for HTML5 web applications

File API: Requesting quota

```
// request quota for persistent store
window.webkitStorageInfo.requestQuota(
  PERSISTENT,
  size,
  function(grantedBytes) {
    window.requestFileSystem(
      PERSISTENT,
      grantedBytes,
      function(fs) {
        // @TODO: access filesystem
      }
    )
  }
)}
```
Offline strategies for HTML5 web applications
Offline strategies for HTML5 web applications

File API: Add item

```javascript
function add(item) {
  window.webkitStorageInfo.requestQuota(
    PERSISTENT,
    size,
    function( grantedBytes ) {
      window.requestFileSystem( PERSISTENT, 
                                grantedBytes,
                                function( fs ) {
                                  writeToFile( fs, item );
                                },
                                onError
                            );
    },
    function( e ) {
      onError( e );
    });
}
```
Offline strategies for HTML5 web applications

File API: Add item

```javascript
function writeToFile(fs, item) {
    fs.root.getFile('todo.txt',
        { create: true },
        function(fileEntry) {
            fileEntry.createWriter(function(fileWriter) {
                var bb = new window.BlobBuilder();
                bb.append(JSON.stringify(item) + "\n");

                fileWriter.seek(fileWriter.length);
                fileWriter.write(bb.getBlob('text/plain'));
            }, onError);
        }, onError);
}
```
function writeToFile(fs, item) {
    fs.root.getFile('todo.txt',
    {
        create: true
    },
    function(fileEntry) {
        fileEntry.createWriter(function(fileWriter) {
            var bb = new window.BlobBuilder();
            bb.append(JSON.stringify(item) + "\n");

            fileWriter.seek(fileWriter.length);
            fileWriter.write(bb.getBlob('text/plain'));
        }, onError)
    }, onError
    );
}
function writeToFile(fs, item) {
    fs.root.getFile('todo.txt',
        {
            create: true
        },
        function(fileEntry) {
            fileEntry.createWriter(
                function(fileWriter) {
                    var blob = new Blob([JSON.stringify(item)+'
                        ]); 
                    fileWriter.seek(fileWriter.length);
                    fileWriter.write(blob);
                }, onError
            ), onError
        }, onError
    );
}
Offline strategies for HTML5 web applications

File API: Read items

```javascript
function read() {
    window.webkitStorageInfo.requestQuota(PERSISTENT, size, function(GrantedBytes) {
        window.requestFileSystem(PERSISTENT, GrantedBytes, function(fs) {
            readFromFile(fs);
        }, onError);
    }, onError);
}
```
Offline strategies for HTML5 web applications

File API: Read items

```javascript
function readFromFile(fs) {
    fs.root.getFile('todo.txt', {
        create: true
    },
    function(fileEntry) {
        fileEntry.file(function(file) {
            var reader = new FileReader();
            reader.onloadend = function(e) {
                if (evt.target.readyState == FileReader.DONE) {
                    // process this.result
                }
            };
            reader.readAsText(file);
        });
    }, onError
);
}
```
Offline strategies for HTML5 web applications

Browser support?
Offline strategies for HTML5 web applications

## Browser support?

<table>
<thead>
<tr>
<th></th>
<th>App Cache</th>
<th>Web Storage</th>
<th>WebSQL</th>
<th>IndexedDB</th>
<th>File API</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE</td>
<td>10.0</td>
<td>8.0</td>
<td>10.0</td>
<td>10.0</td>
<td>-</td>
</tr>
<tr>
<td>Firefox</td>
<td>11.0</td>
<td>11.0</td>
<td>11.0</td>
<td>11.0</td>
<td>19.0</td>
</tr>
<tr>
<td>Chrome</td>
<td>18.0</td>
<td>18.0</td>
<td>18.0</td>
<td>18.0</td>
<td>18.0</td>
</tr>
<tr>
<td>Safari</td>
<td>5.1</td>
<td>5.1</td>
<td>5.1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Opera</td>
<td>12.1</td>
<td>12.1</td>
<td>12.1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>iOS Safari</td>
<td>3.2</td>
<td>3.2</td>
<td>3.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Android</td>
<td>2.1</td>
<td>2.1</td>
<td>2.1</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Source: http://caniuse.com
Offline strategies for HTML5 web applications

Storage limitations?
Offline strategies for HTML5 web applications

Storage limitations?

All storage technologies are limited by quotas. Be aware of what you do!
## Offline strategies for HTML5 web applications

### Storage limitations?

<table>
<thead>
<tr>
<th></th>
<th>App Cache</th>
<th>Web Storage</th>
<th>WebSQL</th>
<th>IndexedDB</th>
<th>File API</th>
</tr>
</thead>
<tbody>
<tr>
<td>iOS 5.1</td>
<td>10 MB</td>
<td>5 MB</td>
<td>5 MB</td>
<td>5 MB</td>
<td></td>
</tr>
<tr>
<td>Android 4</td>
<td>unlimited</td>
<td>5 MB</td>
<td>?</td>
<td>?</td>
<td></td>
</tr>
<tr>
<td>Safari 5.2</td>
<td>unlimited</td>
<td>5 MB</td>
<td>5 MB</td>
<td>5 MB</td>
<td></td>
</tr>
<tr>
<td>Chrome 18</td>
<td>5 MB</td>
<td>5 MB</td>
<td>unlimited</td>
<td>unlimited</td>
<td>unlimited</td>
</tr>
<tr>
<td>IE 10</td>
<td>50 MB</td>
<td>10 MB</td>
<td>500 MB</td>
<td>500 MB</td>
<td></td>
</tr>
<tr>
<td>Opera 11</td>
<td>50 MB</td>
<td>5 MB</td>
<td>5 MB</td>
<td>5 MB</td>
<td></td>
</tr>
<tr>
<td>Firefox 11</td>
<td>unlimited</td>
<td>10 MB</td>
<td>50 MB</td>
<td>50 MB</td>
<td></td>
</tr>
</tbody>
</table>
Thank you!