OPEN
WORLD
FORUM

THINKCODE
EXPERIMENT

OPEN WORLD FORUM | THINKCODE EXPERIMENT

Feedback on offline web apps

Retour d'expérience sur le web offline

OPEN WORLD FORUM

Whoami

- Thomas Cataldo
  - Architecte chez Blue Mind, éditeur de logiciel de messagerie collaborative
  - thomas.cataldo@blue-mind.net
  - @tcataldo
- Blue Mind
  - **http://www.blue-mind.net**
  - Logiciel **sorti** en novembre 2011 au JRES
  - Par l'équipe qui a mené quelques grands projets de messagerie
  - Repensé pour le web d'aujourd'hui

OPEN WORLD FORUM

Today's topic

- Le web offline qu'est ce que c'est ?
  - Une application dans un navigateur web
  - Pas (ou peu) de connexion réseau
- Pourquoi ?
  - Blue Mind fait de la messagerie
  - L'argument numéro 1 pour l'utilisation de clients lourds :
    - L'utilisation sans connexion réseau
    - La disponibilité des données en local, plus rapidement
- Chez Blue Mind nous refusons de voir la messagerie web comme une utilisation dégradée du client lourd

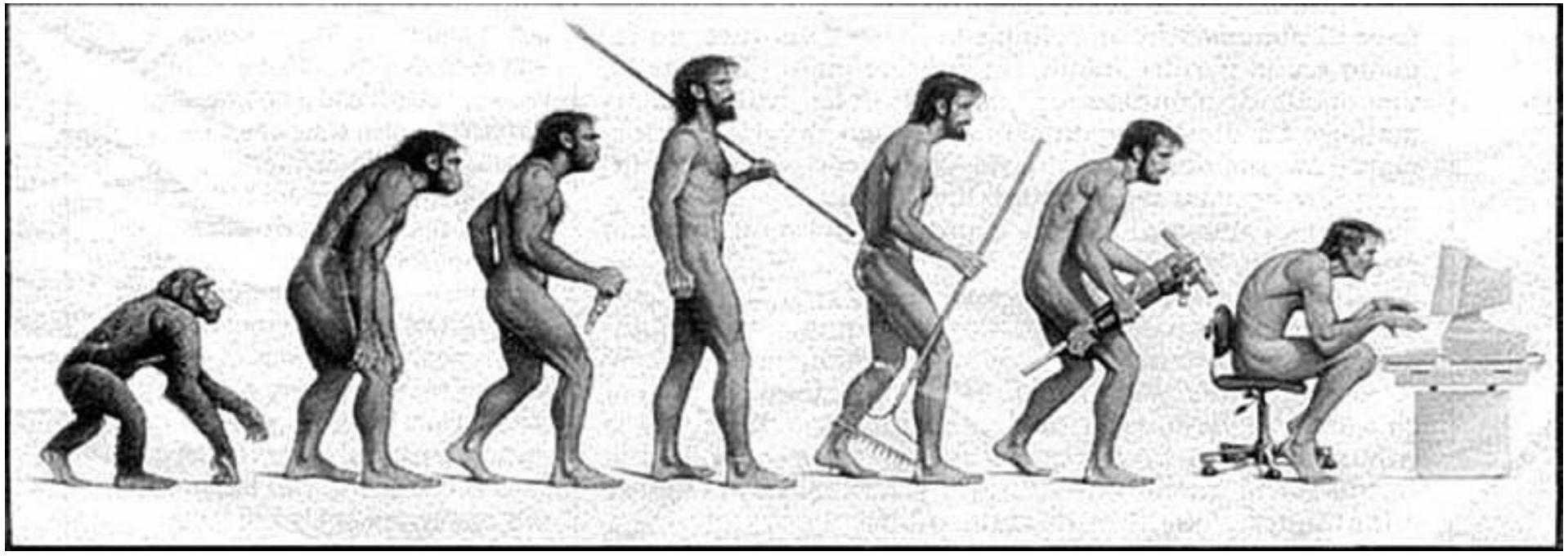
Offline webapps

- Nos applications ont le support du mode offline
- Les données sont stockées en local
  - Données d'agenda
  - Base de contact
  - Messagerie (**Q3 2014**)
- L'application est stockée en local

- Démo rapide

OPEN WORLD FORUM

Architecture evolution



Architecture evolution

- Web dynamique 101
- http://myapp/
  - Page HTML générée par le serveur
- http://myapp/product.php?id=33
  - Génération complète côté serveur

- Rendre cette application accessible sans réseau serait... difficile

OPEN WORLD FORUM

Evolution – Ajax/JS

- Avec l'arrivée des XMLHttpRequest, le serveur ne génère plus des pages entières
    - Seulement de la donnée
        - Ok, parfois un morceau de HTML injecté dans une div
    - Applications « mono-page »
    - Rôle du serveur plus limité

- Le serveur est moins sollicité mais toujours indispensable

Full JS apps

- La logique est intégralement dans le javascript, donc dans le navigateur
- Le serveur n'est plus qu'un fournisseur de données :
  - Envoie de JSON / XML
  - Jamais de données pré-formatées pour l'affichage

- Maintenant supposons que les données envoyées par le serveur sont conservées localement
- Quand une donnée est nécessaire :
  - Si dispo localement, utilisation de la donnée locale
  - Si non dispo :
    - Online : appel ajax puis mise en cache local
    - Offline : window.alert('sorry bro')

OPEN WORLD FORUM

Synchronization

- Rapatrier et **mettre en cache** uniquement à la demande n'est pas très efficace
    - Je ne peux consulter que ce que j'ai déjà vu une fois...
- On va plutôt faire de la synchronisation de masse :
    - Rapatriement d'un maximum de données
    - Re-synchro périodiques et incrémentales

OPEN WORLD FORUM

Massive sync

- Le type de données est important
- Agenda :
  - 1 mois dans le passé, 6 mois dans le futur
- Contact :
  - Anniversaire à -1mois, +1mois ? :-)
  - Le corpus de données est faible, donc on synchronise tout
    - Pas toujours possible
- Mails :
  - Plus compliqué
  - Les 7 derniers jours ?
  - Sans les pièces jointes ?
  - Des dossiers prioritaires tels que les modèles ?

Application Store

- AppCache
  - Pour forcer le navigateur à conserver le code de l'application
  - Et les ressources liées (images, css)
- Au moins, une seule techno pour tous les navigateurs
  - Suffisamment rare pour le signaler !
- Pour la conservation des données, les choses se compliquent...

AppCache - example

```
<html manifest='appcache.jsp'>
  …
</html>
```

- Et l'appcache lui même :

```
CACHE MANIFEST
# Current Language : FR
# Version : 2.0

CACHE:
# tous les fichiers statiques de la webapp
```

AppCache - js

```
window.applicationCache.addEventListener('updateready', function(e)
{
  if (window.applicationCache.status ==
window.applicationCache.UPDATEREADY) {
    window.applicationCache.swapCache();
    if (confirm('A new version of this site is available. Load it?')) {
      window.location.reload();
    }
  }
}, false);
```

Data Stores

- Web Storage (localStorage)
  - Old school
  - Blue Mind 1.0 et 2.0
- IndexedDB
  - NoSQL style
  - Blue Mind 2.0
- WebSQL
  - Sqlite style
  - Blue Mind 2.0
- FileAPI
  - simple

Compatibility

	Chrome	IE	Firefox	Safari
WebStorage	4	8	3.5	4
IndexedDB	24	10	16	
WebSQL	4			3.1
FileAPI	13	10	3.6	6

Today...

- WebStorage comme seule solution « universelle »
  - Généralement limité à 5Mo
  - Sans possibilité d'extension
  - Et comme les chaines javascript sont utf-16, c'est 2.5M de caractères, pas 5.... (seulement dans certains navigateurs)
- Suffisant... parfois
  - Pour des données de calendrier (quelques milliers d'événements)
  - Déjà limite pour les données contact quand 1500+ utilisateurs dans la structure
- Inutile d'espérer remplacer Outlook et des PST de quelques gigas par du webstorage

Webstorage - example

```
// use localStorage for persistent storage
// use sessionStorage for per tab storage
saveButton.addEventListener('click', function () {
  window.localStorage.setItem('value', area.value);
  window.localStorage.setItem('timestamp', (new Date()).getTime());
}, false);
textarea.value = window.localStorage.getItem('value');
```

IndexedDB - example

```
var idbRequest = window.indexedDB.open('Database Name');
idbRequest.onsuccess = function(event) {
  var db = event.srcElement.result;
  var transaction = db.transaction([], IDBTransaction.READ_ONLY);
  var curRequest = transaction.objectStore('ObjectStore Name').openCursor();
  curRequest.onsuccess = ...;
};
```

WebSQL – example

```
var db = window.openDatabase("DBName", "1.0", "description", 5*1024*1024); //5MB
db.transaction(function(tx) {
  tx.executeSql("SELECT * FROM test", [], successCallback, errorCallback);
});
```

FileAPI - example

```
window.requestFileSystem(window.TEMPORARY, 1024 * 1024,
function(fs) {
  // fs.root is a DirectoryEntry object.
  fs.root.getFile('log.txt', {create: true}, function(fileEntry) {
    fileEntry.createWriter(function(writer) {  // FileWriter object.
        writer.onwrite = function(e) { ... };
        writer.onerror = function(e) { ... };
        var bb = new BlobBuilder();
        bb.append('Hello World!');
        writer.write(bb.getBlob('text/plain'));
    }, errorHandler);
  }
}, errorHandler);
```

Improving the situation

- Adaptation au navigateur
- Utilisation de la meilleure techno disponible
    - Pas d'autre choix pour dépasser la limite de taille

- Utilisation de ydn-db
    - Une API tierce mais unique quelque-soit le backend de stockage utilisé
    - Supporte webstorage, indexeddb et websql
    - Activement maintenu
    - Libre
        - Contributions de Blue Mind

OPEN WORLD FORUM

API roundup

- Chacune des APIs est différente
- Certaines sont synchrones, d'autres asynchrones
- Modèles de données différents
- Des librairies existent pour faire abstraction de la couche stockage dans la navigateur
    - Ydn-db

Ydn - example

```
var db = new ydn.db.Storage('db name');
db.put('store1', {test: 'Hello World!'}, 123);
db.get('store1', 123).done(function(value) {
  console.log(value);
}
```

- Stockage d'objet dans store1 avec la clef 123
- Intégré avec la librairie closure de Google
    - Une des 2 technos d'UI utilisée par Blue Mind
        - L'autre est GWT

Added benefits

- Meilleure performances
  - Design par synchro périodiques (ou déclenchées par le serveur)
  - Le navigateur n'a pas réellement besoin du serveur et c'est lui qui execute l'application, plus le serveur
- D'un point de vue serveur, 1 gros transfert est bien plus facile à gérer que N petits et rapprochés
- Possibilité pour le serveur d'indiquer au client quand faire la prochaine resync

OPEN WORLD FORUM

Merci pour votre attention

Retrouvez Blue Mind stand C2