# HTML5 Offline Applications


- Kushal Likhi
- Intelligrape Software

# Agenda

- HTML 5 Offline Apps?
- Demonstration Of The Magic
- Type Of Caches
- The Manifest file
- Browser Events
- Data Storage
- Optimization Tool For Existing Apps
- Design And Code Analysis Of The Examples
- Browser Compatibility



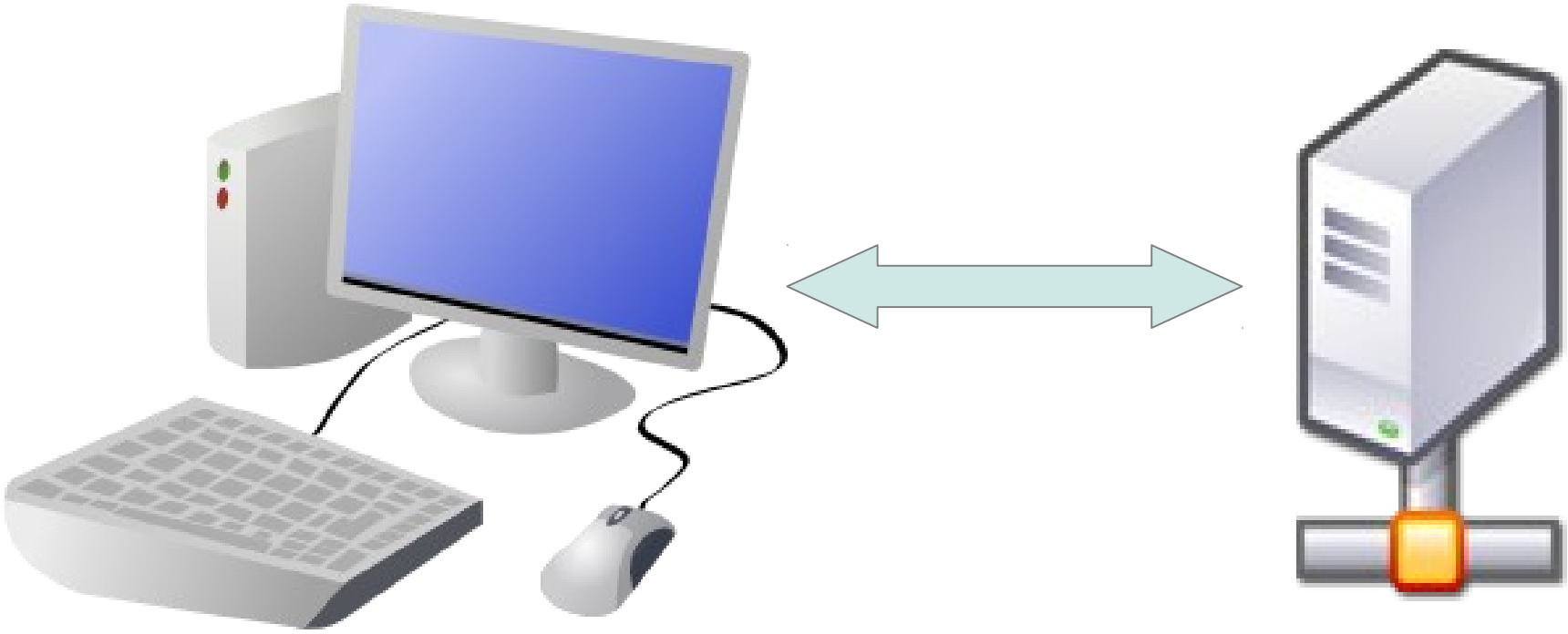
# The Evolution

## When it all started... We built

DESKTOP APPLICATIONS

# The Evolution Contd...

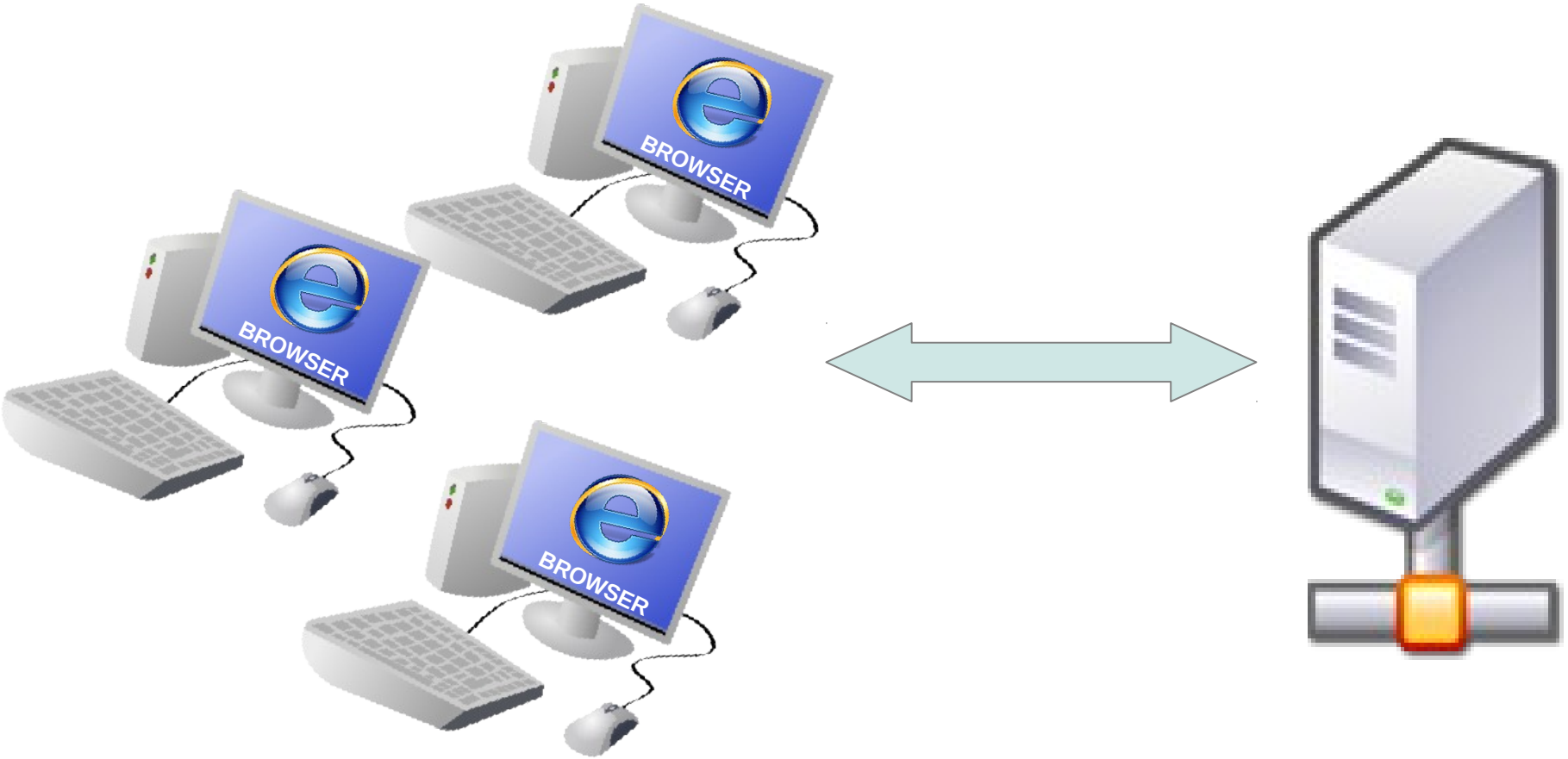
## Then came The....



CLIENT-SERVER APPLICATIONS

# The Evolution Contd...

## Then came The Era Of....




WEB APPLICATIONS

# Desktop Vs Web(HTML 4)

| | Desktop | HTML 4 |
|---|---|---|
| Easily Accessible | NO | YES |
| Fast | YES | NO |
| Unity Upgrades | NO | YES |
| Local Data Storage | YES | NO |
| Internet Connection Mandatory | NO | YES |
| MVC Client And Data Server | YES | NO |



# Desktop Vs HTML4 Vs HTML5

| | Desktop | HTML 4 | HTML 5 |
|---|---|---|---|
| Easily Accessible | NO | YES | YES |
| Fast | YES | NO | YES |
| Unity Upgrades | NO | YES | YES |
| Local Data Storage | YES | NO | YES |
| Internet Connection Mandatory | NO | YES | YES |
| MVC Client And Data Server | YES | NO | YES |

HTML 5 Offline Applications Are The Application Which Can Function Even When Offline And Also Provides Us the Ability To Store Data On The Local/User's System.

# Time To See The Magic

Three working examples:

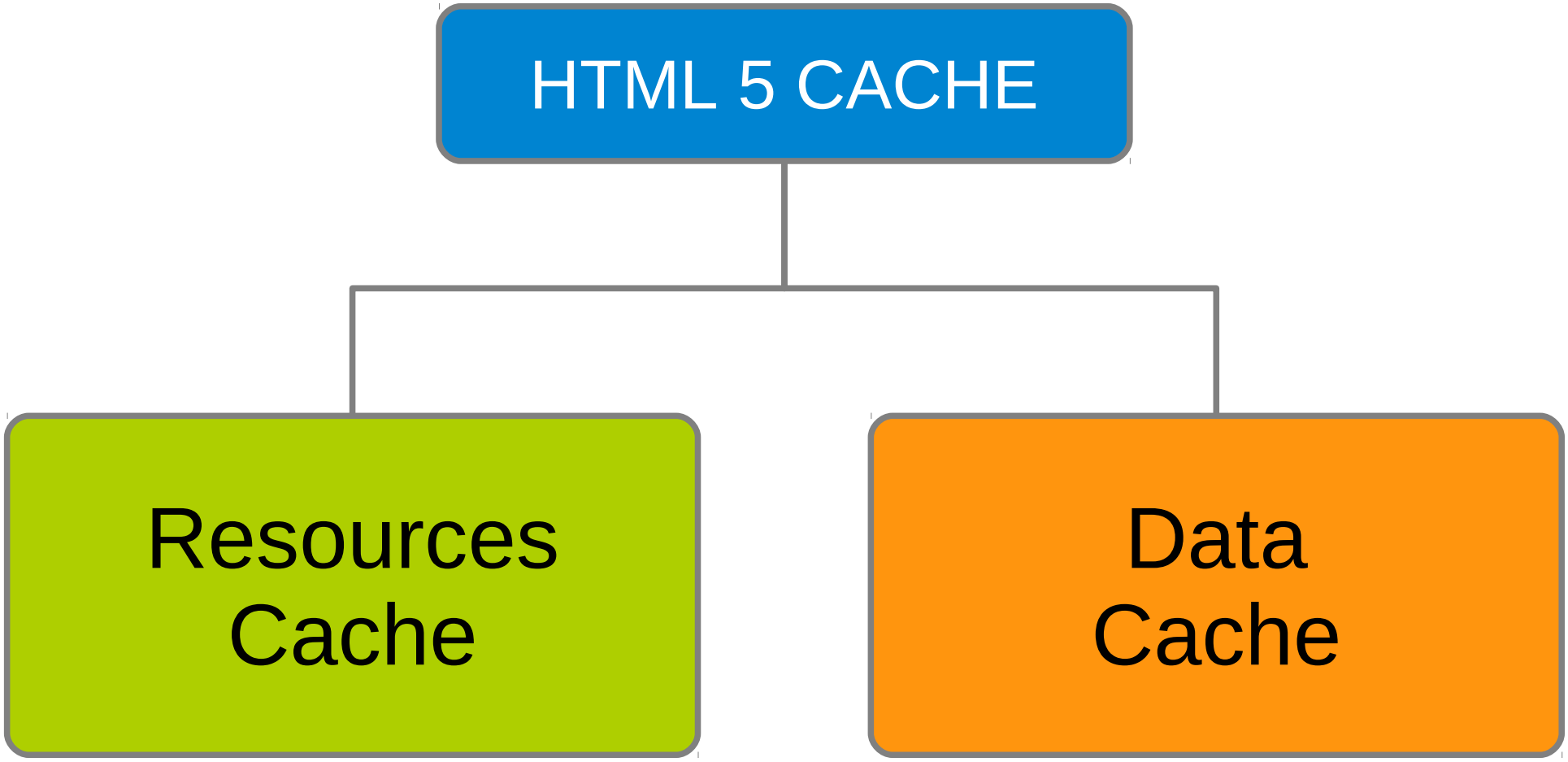
1)A Simple **static site** which can be viewed offline.

2)A Simple Browser based **Calculator** application which can work offline.

3)An **Email Client** which can work offline.



# Type Of Caches



# The Manifest File

The mechanism for ensuring Web applications are available even when the user is not connected to their network is the **manifest** attribute on the html element.

```
<!DOCTYPE html>
<html lang="en" manifest="/offline.appcache">
  // your html document
</html>
```



# The Manifest File

## What That Attribute Takes

The attribute takes a URI to a manifest, which specifies which files are to be cached.

## Content Type For Manifest

The manifest has a ***text/cache-manifest*** MIME type.

# The Manifest File(Format)

**CACHE MANIFEST**

# This is a comment

**CACHE:**

Files entries here

**FALLBACK:**

<request URL> <cached page>

**NETWORK:**

Files entries here

# The Manifest File(Labels)

**CACHE**

A list of explicit URLs to request and store

**FALLBACK**

What to do when an offline user attempts to access an uncached file

**NETWORK**

Which resources are available only while online



# Example Of A Manifest File

**CACHE MANIFEST**
# build 1

**CACHE:**
/css/screen.css
/css/offline.css
http://example.com/css/styles.css

**FALLBACK:**
/ /offline.html
/images/profile/display_pic/ /images/default/dp.png

**NETWORK:**
/feed/onlineUsers.php
*



# Invalidating The Manifest

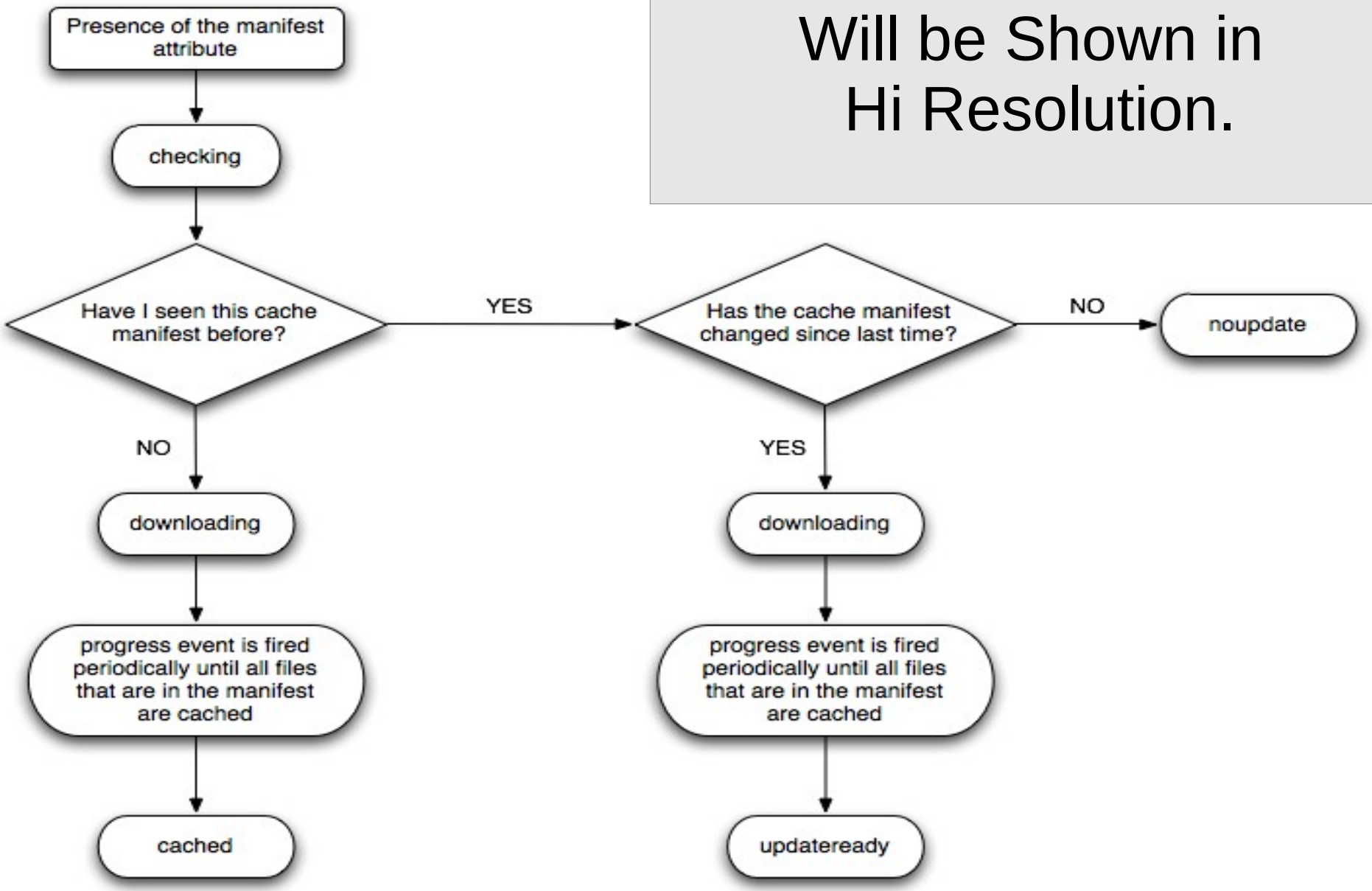
- Using the Comments *(Version Comment as the best practice)*

**Firefox Issue:**

Firefox caches the manifest itself and never requests it from the server. Hence manifest should be invalidated using the appropiate headers.

```
<IfModule mod_expires.c>
 ExpiresActive On
 ExpiresByType text/cache-manifest "access plus 0 seconds"
</IfModule>
```

# DOM Events



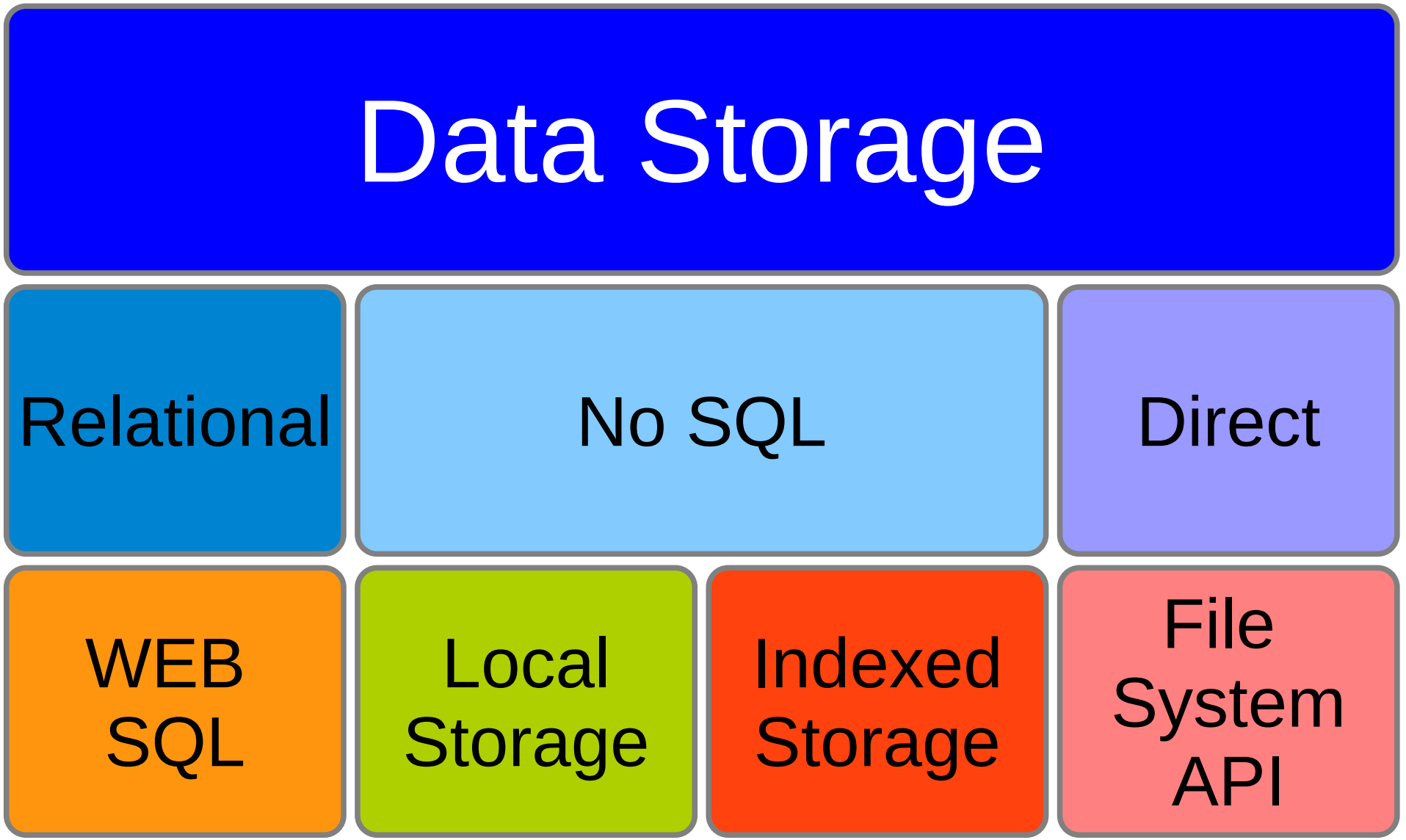

Will be Shown in Hi Resolution.

# Demo DOM Events

Show events getting fired



# Data Storage



# Local storage

Is a Key-Value No-Sql datastore.

**Ex:**

***Set value:***

localStorage.foo = “bar”

***Fetch value:***

Var value = localStorage.foo

# Session storage

Is a Key-Value volatile datastore.

Has a session scope and exists till browser window is closed.

**Ex:**

***Set value:***

sessionStorage.foo = “bar”

***Fetch value:***

Var value = sessionStorage.foo

# Web SQL (Deprecated)

It gives a Structured relational datastore to save our data.

**Deprecated**. Is **not** be supported on IE and Firefox, and will probably be phased out from the other browsers at some stage.

Hence we will Ignore this datastore.



# Indexed Storage (Incubator)

- An Indexed Database is a collection of "object stores" which you can just drop objects into.
- Asynchronous API
- Indexing on basis of keys possible.
- Fast and a good **future** alternative to Web SQL.

**Cons:**

- Complex API
- Still not implemented by majority of browsers.

# File System API (Still a baby)

- We can Read/Write large text/binary files on the user's hard drive.
- Cross browser data sharing.
- Brings Browser based applications a step closer to desktop applications

**Cons:**

- Only implemented in chrome yet.
- In a very early development phase.

# The Winner in Data Storage (Today)

Local Storage / Session Storage



# Optimization Tool For Existing Apps

**Question)** How can the knowledge we learnt can be used to optimise current web based applications.

**Solution)**

Discuss with the Audience!!!

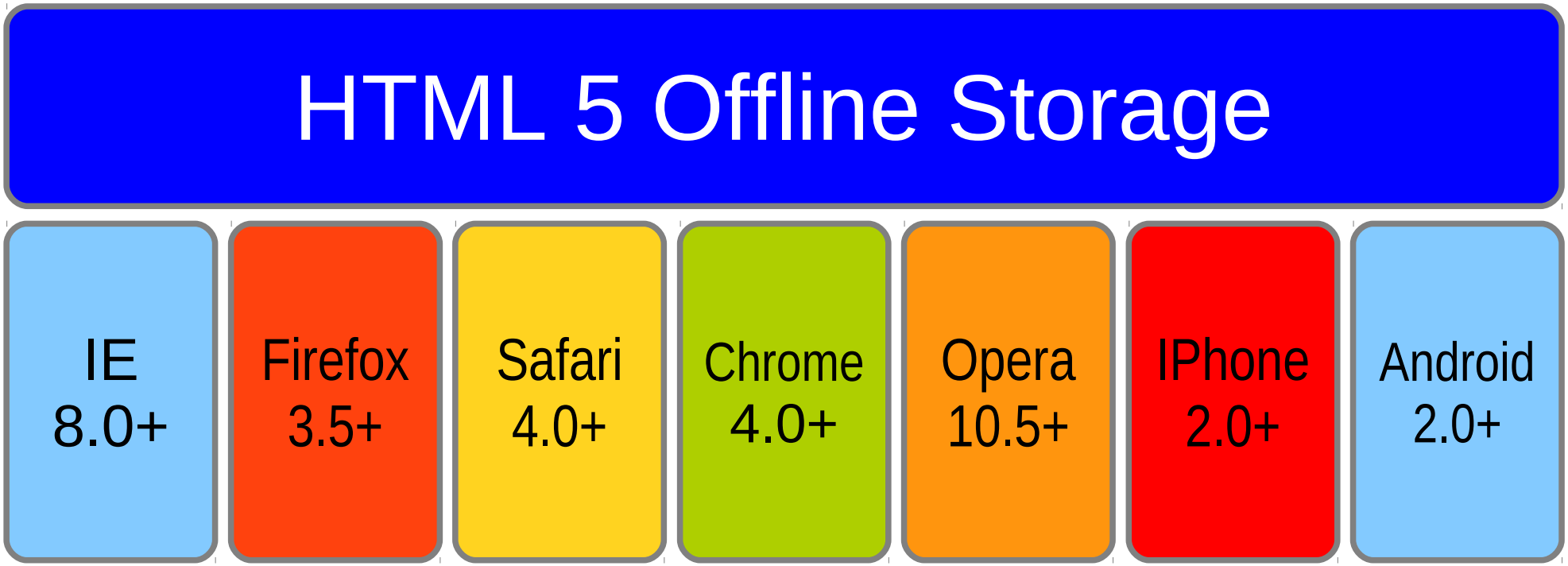


# Analysis Of The Examples

Let us take all the three examples one by one, and audience can guess how they are implemented. Let us discuss and come to the implementation solution.



# Browser Compatibility



Thankyou