|  |
| --- |
|  |



Use Case Document

W3C Interest Group Offline Use Cases

W3C\_03 v0.1

19/07/2013

**Security Classification: This document contains GSMA Confidential Information**

Access to and distribution of this document is restricted to the persons listed under the heading Security Classification Category. This document is confidential to the Association and is subject to copyright protection. This document is to be used only for the purposes for which it has been supplied and information contained in it must not be disclosed or in any other way made available, in whole or in part, to persons other than those listed under Security Classification Category without the prior written approval of the Association. The GSM Association (“Association”) makes no representation, warranty or undertaking (express or implied) with respect to and does not accept any responsibility for, and hereby disclaims liability for the accuracy or completeness or timeliness of the information contained in this document. The information contained in this document may be subject to change without prior notice.

|  |  |  |
| --- | --- | --- |
| **Security Classification – CONFIDENTIAL GSMA Material** | | **Can be distributed to:** |
| Confidential | GSMA Full Members |  |
| Confidential | GSMA Associate Members |  |
| Confidential | GSMA Rapporteur Members |  |
| Confidential | GSMA Parent Company Members |  |

**Copyright Notice**

Copyright © 2012 GSM Association

**Antitrust Notice**

The information contained herein is in full compliance with the GSM Association’s antitrust compliance policy.

Use Case Document: Offline Use Cases

**Authors**

Natasha Rooney, GSMA

nrooney@gsma.com

+44 7730 219 765

W3C Interest Group Members

**Document Authorisation**

| Name | Position | Date |
| --- | --- | --- |
|  |  |  |

**Document Change History**

| Version | Date | Brief Description of Changes | Author | Status |
| --- | --- | --- | --- | --- |
| 0.1 |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Reviewer**

| Name | Position | Date |
| --- | --- | --- |
|  |  |  |
|  |  |  |

**Action Points**

| Name | Description | Date |
| --- | --- | --- |
| Bryan Sullivan | Provide an update on W3C workshop on social standards and his discussion on offline use cases |  |
|  |  |  |

**To-do**

Please include action points for this document here.

* + - 1. Decide whether we want separate cases around “allowing web apps to act as if they are native apps” by adopting some kind of app cache model, or forcing everything to use a web model (which is better in most cases as content is always fresh if available).
      2. Add CDN and Roaming Use Cases.
      3. Add cases from FT labs.
      4. Add cases where it says what a developer should tell the app. For example; ‘cache these pages’ or ‘cache these assets’, then what the developer expects the cache to do.
      5. Caching normal websites, comes back to definition of web app and web site
      6. Does user tell browser a website is “allowed” to cache more data?
      7. Does the user tell the browser to cache a site, or does it tell the site?
      8. Add some trust based Use Cases. “Caching of a site should be reliant on how trustworthy it is, and if based on the user’s trust then perhaps this will grow over time”
      9. Do devs just want to load a “shell” of an application and data will come from server? App cache was built to cache the shell, db should be on server or local storage. You cache 2 things: shell, content. These should be treated independently.
      10. Add Fallback Use Cases. Fallback timeouts – app cache had fallbacks but they were too long
      11. Synchronisation (dealt with this a bit already)
      12. Just downloading data – rather then downloading shell and data – so dev just says “cache shell, get data”
      13. Make sue we have both “offline app” use cases as well as “offline for performance” use cases (although Jake said http cache is for performance)
      14. Could we have a priority system to prioritise items to cache?
      15. Privilege vs priority – if you get more space you give up something else like sandboxing (what Alex was telling me last time)
      16. Lookup “futures” or “promises” for dom, Alex Russell
      17. See if we can hunt down Jonas and ask him about Mozilla alternative proposal to navigation controller

**Contents**

Overview 4

Background 4

W3C: Existing Work 4

W3C: New Work 4

States 4

Use Cases 4

Actors 4

Key Use Cases 5

Caching Assets 5

Use Case: Download and Go 5

Use Case: Cache based on Personalisation 6

Use Case: Cache based on Generalisation if Personalisation is not available 6

Use Case: User Can Alter Personalisation within the Settings 6

Use Case: User selects assets to download 6

Use Case: Caching Maps 7

Use Case: Caching Books 7

Use Case: Caching Websites 8

Online / Offline State Change Behaviour 8

Use Case: Only Display Available Options 8

Use Case: Do Not Track 8

Use Case: Update Web App Assets 8

Use Case: Working with the Network Information API 8

Use Case: Moving from Offline to Online 9

Use Case: Caching when working with Subscriptions 9

Use Case: Smart Cache based on Bandwidth 9

Use Case: Smart Cache based on an Intermittent Connection 10

“Installed” Web Apps 10

Use Case: “Installed” Web Apps Can Function 10

Use Case: Is “Installed” 10

CDN Use Cases 11

Use Case: Descriptive Title 11

Roaming Use Cases 11

Use Case: Descriptive Title 11

Other Use Cases 11

Use Case 2: Watch and Record 11

Out of Scope / Not Supported 11

Scenarios 12

Use Cases 12

Security and Risk Assessment 12

Security 12

Privacy 12

Appendix A: Reading 12

Appendix B: Natasha’s Notes 12

# Overview

## Background

There has been a lot of varied work being done within the W3C on offline behaviour. A number of issues were existing with these implementations, which have caused the W3C Interest Group to analyse the situation to see what can be done. It was decided to start by documenting Use Cases. These can then be taken to the W3C to contribute to the larger discussion in the correct working groups.

## W3C: Existing Work

Existing offline supporting APIs are given below.

* [Application Cache](http://www.w3.org/html/wg/drafts/html/master/browsers.html#appcache) – analysed as broken and discredited, mainly for the reasons detailed in Jake Archibald’s blog post “[Application Cache is a Douchebag](http://alistapart.com/article/application-cache-is-a-douchebag)”
* [Local Storage (or Web Storage)](http://www.w3.org/TR/webstorage/)
* [Web SQL](http://www.w3.org/TR/webdatabase/) – now inactive
* [Indexed Database](http://www.w3.org/TR/IndexedDB/)
* [Online/offline events](http://www.w3.org/TR/2012/WD-netinfo-api-20121129/#event-handlers) – as part of the Network Information API

## W3C: New Work

Mainly due to the insufficiencies of App Cache some new proposals are being developed and will be proposed to the W3C. Below are links and descriptions of these new proposals.

* [NavigationController](https://github.com/slightlyoff/NavigationController/) – the [explainer document](https://github.com/slightlyoff/NavigationController/blob/master/explainer.md) is the best way to gain an understanding of this proposal; below is a quick copy/paste from this doc:

“A Navigation Controller is a bit of script that manages content caches and decides what content to display when a URL is requested.

In our video example, one cache might be built/managed to help make sure that the shell of the application is available offline. Another might be built to represent the downloaded videos. Yet another might be built to keep a local inventory of ads or trailers to show before movies play. Each of these caches are effectively independent bits of content, joined at runtime by the application -- and Navigation Controllers mediate how applications come into being.”

* [Web Manifest](https://github.com/w3c/manifest) specification being worked on by the Web Applications Working Group.

## States

Throughout the document we talk about such things as “offline state” and “online state”. Here is a better description of these phrases:

* OFFLINE STATE – the user has no connectivity to the web.
* ONLINE STATE – the user can access the web via WiFi or via the operator network.

# Use Cases

## Actors

| Name | Description | Parent |
| --- | --- | --- |
| USER | The User is a consumer of the web application or web site and the end user of the system. |  |
| DEVELOPER | The Developer is the creator of the system and consumes APIs as defined by the W3C and browser vendors |  |
| DEVICE | Device in which the user is using to view the web site or web app (via a User Agent) |  |
| USER AGENT | The client or application which the user views to access the web app, either by sending a request to a web server or by accessing packaged web-apps on the device’s storage, local storage, cache or another caching mechanism. |  |
| WEB APP | Web App or site being viewed by the USER on the USER AGENT |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Key Use Cases

Use Cases are shown below.

### Caching Assets

Use cases which involve caching assets which is triggered by the USER or the WEB-APP.

#### Use Case: Download and Go

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | Using a web browser, a user downloads a series of videos onto a tablet, intending to use them to entertain the kids on a road trip. |
| **Need/Justification** | Accessing live video streams may not be possible (e.g. no network, or per content provider policy) or desirable (e.g. due to data usage or low QoE) when mobile or away from home. Nonetheless with video-capable devices, there will be a desire to be able to watch videos even in those circumstances. |
| **Requirements** | Download   * A method of referencing video sources for download, e.g. using anchor elements with [the download attribute](http://www.w3.org/TR/html5/single-page.html#downloading-resources).   Content protection   * Ability to store video content in a protected format, as applicable. * Ability to view previously stored protected video content, e.g. via the [HTML5 Encrypted Media Extensions](https://dvcs.w3.org/hg/html-media/raw-file/tip/encrypted-media/encrypted-media.html) * Ability to transfer media in a single operation (not copy-then-delete) * Ability to account & verify the number of coexisting instances of the media   Storage   * An adequately-sized storage medium; at least enough to store several full movies, e.g. 32GB * A method of accessing the storage medium to save videos, e.g.   + For local filesystem storage, the [File Writer API](http://dev.w3.org/2009/dap/file-system/file-writer.html)   + For browser-internal storage, the [IndexedDB API](http://www.w3.org/TR/IndexedDB/) * A method of specifying the validity duration of the content in offline storage * A method of clearing the content in offline storage when its validity period has expired   Playback   * A method of accessing the storage medium for video playback, e.g.   + The [File API](http://www.w3.org/TR/FileAPI/)   + [IndexedDB API](http://www.w3.org/TR/IndexedDB/) * Ability to view previously stored video content, e.g. via   + The [HTML5 video element](http://www.w3.org/TR/html5/single-page.html#the-video-element) using a reference to a locally-stored file, provided through the [File API](http://www.w3.org/TR/FileAPI/)   + The [HTML5 video element](http://www.w3.org/TR/html5/single-page.html#the-video-element) using a reference to a video stored in browser-internal storage, and accessed via the [IndexedDB API](http://www.w3.org/TR/IndexedDB/) * Ability to view previously stored protected video content, e.g. via the [HTML5 Encrypted Media Extensions](https://dvcs.w3.org/hg/html-media/raw-file/tip/encrypted-media/encrypted-media.html) |
| **Note** | Existing Use Case taken from: <http://www.w3.org/2011/webtv/wiki/Download#Download_and_Go> |

#### Use Case: Cache based on Personalisation

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | A WEB APP should be able to cache a set of assets based on the USER’s history |
| **Need/Justification** | A WEB APP may want to cache some assets for the user so they have a smooth UI when offline or on low connectivity networks. Large WEB APPs have the issue where the number and total size of their assets is too large (e.g. a music library). In this case the WEB APP should be able to cache assets to the USER based on the USER’s history. |
| **Requirements** |  |
| **Note** | Need to think of how this will work with “Do Not Track” |

#### Use Case: Cache based on Generalisation if Personalisation is not available

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | A WEB APP should be able to cache a set of assets based on general data collected from all users of which the WEB APP has data |
| **Need/Justification** | A WEB APP may want to cache some assets for the user so they have a smooth UI when offline or on low connectivity networks. Large WEB APPs have the issue where the number and total size of their assets is too large (e.g. a music library). In this case the WEB APP should be able to cache assets to the USER based on the USER’s history; but when this is not available the WEB APP should cache assets based on the general data collected from all users, rather then caching the total collection of assets. |
| **Requirements** |  |
| **Note** |  |

#### Use Case: User Can Alter Personalisation within the Settings

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | When a WEB APP caches a set of assets based on the USER’s history the USER should be able to alter these personalisation parameters in the “settings”. |
| **Need/Justification** | USER’s often search for items which they do not wish to be stored within their personalised history. This could be because; they searched for something out of curiosity but did not actually want to save it; they searched for something as a “gift” for someone else and this is not indicative of their preference; or they searched for something as a mistake. In these cases USER’s should be able to alter the personalisation parameters (e.g. like Amazon do for recommendations; although perhaps not this method!) so caching gives them the desired results. For example; I search for “Greatest Football Injuries” for my Dad but I have no interest in football, I wish to turn “football” off in the settings so football assets aren’t cached on my PC. |
| **Requirements** |  |
| **Note** |  |

#### Use Case: User selects assets to download

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | User “selects” the items to download before they enter into an offline situation. |
| **Need/Justification** | Before a user goes into an offline situation; e.g. going into an underground metro or onto a flight, they may wish to “download” a series of items which they can still interact with whilst in this offline mode. A common example would include such things as levels of a video game, episodes from a TV series, paint brushes for a painting app, articles from a newspaper or emails from an email client. |
| **Requirements** | Assets   * Not only will the selected assets need to be downloaded (e.g. the levels of a game) but all the assets needed to run the app in an offline state will also be needed. * This includes all the HTML files, CSS files, scripts, image, sounds etc. that may be needed to maintain a smooth level of UI between online and offline states   + This may require some from of “asset library” which holds the necessary assets associated with a type of downloadable content   + E.g. if the downloadable content is a “movie” the “asset library” may hold such things as the HTML and CSS files, scripts and codecs. |
| **Note** |  |

#### Use Case: Caching Maps

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | USER downloads/caches maps for later use |
| **Need/Justification** | USERs often use mapping web apps when on the move. In many cases they use these apps when they are in areas of bad or no connection. In this case a USER would be able to download/cache a section of a map (maybe when they are at on their home wifi before leaving) and use this map later. |
| **Requirements** | Zoom Levels   * All zoom levels of the map need to be downloaded including the zoom level the user is on.   + It makes more sense that all zoom levels from the most granular the user is using and up is downloaded, rather then all levels.   + EXMAPLE: (zoom levels 1-10) This means that if a user is viewing zoom level 5 (medium level of granularity) and they can see the whole of the UK then levels 5-10 will be downloaded (10 will be the whole Earth view). The user will not be able to zoom to levels 4,3,2,1 when offline as they did not visit these online.   + This method helps to make sure in the instance when a user views the whole earth they also don’t accidently download all maps at all zoom levels   Assets   * Assets such as supporting HTML/CSS/Script files are outside scope of this use case |
| **Note** | Offline or Low Connectivity?  Distinction needs to be made between offline and low connectivity. If a USER caches a map and accesses it on low connectivity does the app attempt to get a new version or use the cache? It would be sensible for the app to use the cache when on low connectivity and grab the latest version from the server when on a better connection. In this case the levels for “low” and “high” connectivity would have to be set, *or* this gets left to the developer and the API just allows them to make the decision and code against it. |

#### Use Case: Caching Books

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | USER downloads/caches books for later use |
| **Need/Justification** | USERs would like to view their books when offline or going in and out of areas with low connectivity. |
| **Requirements** |  |
| **Note** |  |

#### Use Case: Dynamic Content Delivery for larger files such as videos

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, DEVICE, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** |  |
| **Need/Justification** |  |
| **Requirements** |  |
| **Note** | DCD can be trickier for larger files (such as Videos), this use case should cover this (although this may be covered by “Download and Go”). |

#### Use Case: Caching Websites

Do we need a Use Case for caching standard websites? I assume this is covered by “[Cache based on Personalisation](#_Use_Case_1:)” and “[Cache based on Generalisation](#_Use_Case_1:_1)” but this may be a naive view.

### Online / Offline State Change Behaviour

When moving from offline to online states a web app should exemplify certain behaviours. These are given below.

#### Use Case: Only Display Available Options

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | When a USER is offline they should only see the options that are actually available to them. |
| **Need/Justification** | When USERs go offline and they continue to see links and interactive options on a web app they may feel that these are still available to them to use. These should be made unavailable (not greyed out) so the USER knows these are not available anymore. |
| **Requirements** | * DEVELOPER would need an easy way to make a link or interactive feature disappear in the instance of a USER going offline |
| **Note** |  |

#### Use Case: Do Not Track

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | When a USER has selected “Do No Track” then no data should be taken which means personalisation cannot occur. |
| **Need/Justification** | Standardisation around “Do Not Track” will mean that personalisation based on USER’s data cannot occur. In this case generalisation can be used to select the assets to be downloaded (as mentioned above). The major element of this Use Case is that if a USER selects “Do Not Track” then they will not be tracked and no data will be taken of their history or preferences unless they specifically enter these themselves within the settings. |
| **Requirements** |  |
| **Note** |  |

#### Use Case: Update Web App Assets

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | A DEVELOPER should be able to trigger the WEB-APP to “download” new HTML, CSS, JavaScript, images etc. when these are uploaded to the server |
| **Need/Justification** | One of the benefits of the web is the fact that the USER always sees the most up to date version of any web-app or web-site. When we start introducing complex caching methods the issue arises of when a web-app’s assets have been updated then how does a “cached” web app know this has happened? |
| **Requirements** |  |
| **Note** | Not possible without polling or push. |

#### Use Case: Working with the Network Information API

|  |  |
| --- | --- |
| **Actors** | DEVELOPER, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | Offline capabilities should be able to work with the Network Information API so the triggering of certain events (such as “user goes offline”, or “user goes online”) can mesh with offline capable APIs. |
| **Need/Justification** | The Network Information API is the best method a developer has to detecting the USER’s online status and bandwidth. Whatever offline capable solutions we standardise we must make sure it is possible for the DEVELOPER to code a solution which works with the Network Information API. |
| **Requirements** |  |
| **Note** |  |

#### Use Case: Moving from Offline to Online

|  |  |
| --- | --- |
| **Actors** | DEVELOPER, WEB-APP, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | When a USER moves from offline onto online, there must be a way for the WEB-APP to notify the server to tell the server what assets the USER consumed, and which assets it now needs. |
| **Need/Justification** | When a USER goes offline the WEB-APP can request assets be cached to allow the USER to access them whilst offline. However, when this USER moves back to an online state the WEB-APP should be able to notify the server to say “I am back online, here is what I did and what I now need”. For instance, if a USER is watching a US sitcom series of 24 episodes, they may download episodes “6-10” for their 2 hour plane flight. After the flight, the app will need to tell the server “update this USER’s history to say they watched episodes 6-10, and display this in their history, and remove them from the recommended list as they have been watched”. In certain cases the WEB-APP could also serve new “episodes” or other large assets, although it may be recommended that the DEVELOPER tests the USER’s bandwidth before downloading large assets. |
| **Requirements** |  |
| **Note** |  |

#### Use Case: Caching when working with Subscriptions

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | When a USER is on a subscription plan or a “pay limit” plan (e.g. they are only allowed to view X number of articles) then the USER should only be able to cache the number of items relevant to their pay plan, AND once the USER is back online, the WEB-APP should notify the server as to how many “pay-plan assets” were consumed. |
| **Need/Justification** | Some web apps are run using subscription or pay-plan models which allow USER’s to consume certain content but not everything. Considering this; when a USER is about to go into an offline state the WEB-APP should only cache assets that the USER is allowed to consume according to their pay-plan. Moreover, when the USER goes back into an online state the WEB-APP will need to notify the server of which assets where consumed so it can update such items as the USER’s history, play-list etc. and make sure the user cannot consume more assets then their pay-plan limit. |
| **Requirements** |  |
| **Note** |  |

#### Use Case: Smart Cache based on Bandwidth

|  |  |
| --- | --- |
| **Actors** | DEVELOPER, WEB-APP, DEVICE, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | A “Smart Cache” could cache assets only when a USER has acceptable bandwidth, and not attempt cache them when the USER has an unacceptable bandwidth. |
| **Need/Justification** | If a WEB-APP allows a user to cache assets based on offline/online state this may cause issues when a USER is online but on a “bad” or “unacceptable” bandwidth. Using the Network Information API the WEBAPP could cache assets only when it is on an “acceptable” bandwidth. |
| **Requirements** |  |
| **Note** | There is an issue here based on the term “acceptable bandwidth”. What is acceptable for caching movies is not the same as what is acceptable in caching news articles. Further definition on “acceptable bandwidth” will need to occur, or this level can be left to the developer. |

#### Use Case: Smart Cache based on an Intermittent Connection

|  |  |
| --- | --- |
| **Actors** | DEVELOPER, WEB-APP, DEVICE, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | In the cases of an intermittent connection (as DEVICE is moving from and offline to online state) the Smart Cache should be allowed to start it's caching but pause it when going offline. |
| **Need/Justification** | Smart caches could cache assets based on the bandwidth a USER has, however, this causes issues when a USER jumps onto an “acceptable” connection but only remains connected for a very short period of time. A good example here is the London Underground; stations have good connectivity but a USER may only be at a station for 30 seconds. If at the point of connectivity a movie starts downloading and the USER moves into an offline state then the Smart Cache should pause the cache, “remembering” how much of the current asset it has cached, as well as what has been cached and what is left to go. At the next point the USER has connectivity, it should resume caching from the point it left off. |
| **Requirements** |  |
| **Note** | The same should be true is the DEVICE is moving from an “acceptable” bandwidth to an “unacceptable” bandwidth as of the Use Case: Smart Cache based on Bandwidth. |

### Privileges and Priorities

#### Use Case: User can give space privileges

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, DEVICE, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | A USER should be able to grant a WEB-APP that they trust with greater privileges related to space. Say (for example) a web-app has access to 10MB cache space but it really needs 20MB. The current caching solutions don’t allow for 20MB and for lots of reasons this is a good idea, as when a USER navigates to a new web-app they do not yet know how much they trust it. However, on using the web-app they can determine that they **do** trust the app and they want it to have access to extra storage to cache files for offline use or for performance gains (the objective is irrelevant). On the USER’s request the browser grants further access to storage to the web-app. |
| **Need/Justification** |  |
| **Requirements** |  |
| **Note** | Should the user or browser be able to revoke these privileges? |

### “Installed” Web Apps

In this case “Installed” means a web app which is downloaded to the device **and** coded using web technologies. The code lives on the device, rather then living on a remote web server.

#### Use Case: “Installed” Web Apps Can Function

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, DEVICE, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | “Installed” web apps can function even when the user is not online |
| **Need/Justification** | Packaged web apps, such as those installed on FirefoxOS devices, are coded using Web technologies and web standards but either [a] do not need to be online to function or [b] need to function at all times including offline. A perfect example will be utility style applications including “alarms” and “calendars”. In these cases the apps must continue to run even without a connection. |
| **Requirements** | Manifest file   * This use case requires some sort of manifest file to hold that assets that are needed to run offline * Can hold assets such as: HTML files, CSS files, scripts, image, sounds, SVG |
| **Note** |  |

#### Use Case: Is “Installed”

|  |  |
| --- | --- |
| **Actors** | WEB-APP, DEVICE, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | The WEB APP should be able to know if it is "installed" and being able to promote the user to "install" it if it is not already installed. |
| **Need/Justification** |  |
| **Requirements** |  |
| **Note** |  |

#### Use Case: Installed apps should be able to request new content

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, DEVICE, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | The code for packaged web apps lives on the device rather then on a central web server. Normally when a developer wishes to add content to their web app they can just do this by pushing the update to the web server. In packaged apps this is harder. This use case states that the app should be able to know (or be told) when there’s new content, and request that the new content be downloaded to the app. |
| **Need/Justification** |  |
| **Requirements** |  |
| **Note** | “Content” in this case refers to items such as new news articles, videos, game levels etc. and NOT the shell of an application |

#### Use Case: Installed apps should be able to request new update

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, DEVICE, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | The code (or “shell”) for packaged web apps lives on the device rather then on a central web server. Normally when a developer wishes to update their web app they can just do this by pushing the update to the web server. In packaged apps this is harder, as the app needs to request that the user downloads a new update. This use case states that the app should be able to know (or be told) when there’s a new update, and request that the user downloads the new update. |
| **Need/Justification** |  |
| **Requirements** |  |
| **Note** | “Shell” in this case refers to the code of an application that makes up the whole of the application with exception of the actual content. For example in “Netflix” the shell would be the library layout |

### CDN Use Cases

Use cases associated with Content Delivery Networks.

#### Use Case: Descriptive Title

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, DEVICE, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** |  |
| **Need/Justification** |  |
| **Requirements** |  |
| **Note** |  |

### Roaming Use Cases

#### Use Case: Descriptive Title

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, DEVICE, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** |  |
| **Need/Justification** |  |
| **Requirements** |  |
| **Note** |  |

### Other Use Cases

Other use cases which don’t fit into one of the above categories.

#### Use Case 2: Watch and Record

|  |  |
| --- | --- |
| **Actors** | USER, DEVELOPER, WEB-APP, DEVICE, USER-AGENT |
| **Pre-condition** |  |
| **Post-condition** |  |
| **Description** | Using a web browser, a user watches a video and records it at the same time. |
| **Need/Justification** | Ability to record a video while watching is a basic thing that many users will expect. |
| **Requirements** | Recording while watching   * Ability to store video content accessed via the [HTML5 video element](http://www.w3.org/TR/html5/single-page.html#the-video-element), while the video is being presented in the browser.   Content protection   * Ability to store video content in a protected format, as applicable.   Storage   * An adequately-sized storage medium; at least enough to store several full movies, e.g. 32GB * A method of accessing the storage medium to save videos, e.g.   + For local filesystem storage, the [File Writer API](http://dev.w3.org/2009/dap/file-system/file-writer.html)   + For browser-internal storage, the [IndexedDB API](http://www.w3.org/TR/IndexedDB/) |
| **Note** | Existing Use Case taken from: <http://www.w3.org/2011/webtv/wiki/Download#Watch_and_Record> |

## Out of Scope / Not Supported

For clarity, out of scope items are listed below:

| Name | Description |
| --- | --- |
|  |  |

# Scenarios

Text.

## Use Cases

Sequence diagrams and flow for each use case.

# Security and Risk Assessment

## Security

[EXAMPLE] Security risks are detailed below:

## Privacy

[EXAMPLE] Privacy risks are detailed below:

# Appendix A: Reading

Here are some helpful links to help everyone understand the current situation with offline and the work that has been done previously:

* [AppCache Wows, Vows & Woes, Position Paper for the W3C Workshop on The Future of Offline Web Applications](http://www.w3.org/2011/web-apps-ws/papers/Facebook.html). *Author:* Tobie Langel, Software Engineer, Facebook.
* [W3C Workshop on The Future of Offline Web Applications](http://www.w3.org/2011/web-apps-ws/Report). 5 November 2011, Redwood City, CA, USA

# Appendic B: Glossary

|  |  |
| --- | --- |
| Content |  |
| Shell |  |
|  |  |
|  |  |
|  |  |

# Appendix C: Natasha’s Notes

Notes I wish to keep which may be added to the core paper at some stage.

* applications that are still available to the user when the network is no longer available (provided the user knows how to access them)
* applications that are “installed” by the user — associated with a greater sense of ownership
* applications that are granted additional privileges (for instance, via an “installation” step)

The link between a Web app and the availability/visibility of a browser chrome was also discussed, exposing different approaches that have been experimented with: running Web app as a pinned tab, or running it in a separate process in a chromeless browser. It was established that users expect different behaviors inside a browser chrome vs in a chromeless Web app: for instance, in terms of following links, or in the presence of ads.

**JAKE:**Detecting connectivity doesn’t work. There is a JavaScript API called online, navigate online, but it’s pretty useless. It’s rubbish. The reason is that what is offline? If I’ve got no signal on my mobile phone, am I offline? You’d probably say yes. But if I’m connected to a Wi-Fi router, but that router isn’t connected to the internet, am I offline? I probably am. But what if I’m just accessing the website and there’s a proxy error in between, a DNS failure so I can’t actually get your site. To that website, I am offline and the only way of testing that is to try and make the connection. That makes it slow. You can’t do that for a whole site, especially if you’ve got a middling mobile signal. That can take minutes until it finally gives up and goes, oh yeah, I’m probably offline.

Anything that native can do web can do – Jonas Sicking