Offline Web applications

**Introduction**

In order to enable users to continue interacting with Web applications and documents even when their network connection is unavailable — for instance, because they are traveling outside of their ISP's coverage area — authors can provide a manifest which lists the files that are needed for the Web application to work offline and which causes the user's browser to keep a copy of the files for use offline.

With HTML5 it is easy to make an offline version of a web application, by creating a cache manifest file

## What is Application Cache?

HTML5 introduces application cache, which means that a web application is cached, and accessible without an internet connection.

Application cache gives an application three advantages:

1. Offline browsing - users can use the application when they're offline
2. Speed - cached resources load faster
3. Reduced server load - the browser will only download updated/changed resources from the server

Internet Explorer 10, Firefox, Chrome, Safari and Opera support Application cache.

To enable application cache, include **the manifest attribute** in the document's <html> tag:

The recommended file extension for manifest files is**: ".appcache"**

<!DOCTYPE HTML>  
<html manifest="demo.appcache">  
<body>  
The content of the document......  
</body>  
</html>

## The Manifest File

The manifest file is a simple text file, which tells the browser what to cache (and what to never cache).

The manifest file has three sections:

* **CACHE MANIFEST** - Files listed under this header will be cached after they are downloaded for the first time
* **NETWORK** - Files listed under this header require a connection to the server, and will never be cached
* **FALLBACK** - Files listed under this header specifies fallback pages if a page is inaccessible

### CACHE MANIFEST

The first line, CACHE MANIFEST, is required:

CACHE MANIFEST  
/theme.css  
/logo.gif  
/main.js

The manifest file above lists three resources: a CSS file, a GIF image, and a JavaScript file. When the manifest file is loaded, the browser will download the three files from the root directory of the web site. Then, whenever the user is not connected to the internet, the resources will still be available.

### NETWORK

The NETWORK section below specifies that the file "login.asp" should never be cached, and will not be available offline:

NETWORK:  
login.asp

An asterisk can be used to indicate that all other resources/files require an internet connection:

NETWORK:  
\*

### FALLBACK

The FALLBACK section below specifies that "offline.html" will be served in place of all files in the /html/ catalog, in case an internet connection cannot be established:

FALLBACK:  
/html/ /offline.html

**Note:** The first URI is the resource, the second is the fallback.

## The Manifest File

The manifest file is a simple text file, which tells the browser what to cache (and what to never cache).

The manifest file has three sections:

* **CACHE MANIFEST** - Files listed under this header will be cached after they are downloaded for the first time
* **NETWORK** - Files listed under this header require a connection to the server, and will never be cached
* **FALLBACK** - Files listed under this header specifies fallback pages if a page is inaccessible

### CACHE MANIFEST

The first line, CACHE MANIFEST, is required:

CACHE MANIFEST  
/theme.css  
/logo.gif  
/main.js

The manifest file above lists three resources: a CSS file, a GIF image, and a JavaScript file. When the manifest file is loaded, the browser will download the three files from the root directory of the web site. Then, whenever the user is not connected to the internet, the resources will still be available.

### NETWORK

The NETWORK section below specifies that the file "login.asp" should never be cached, and will not be available offline:

NETWORK:  
login.asp

An asterisk can be used to indicate that all other resources/files require an internet connection:

NETWORK:  
\*

### FALLBACK

The FALLBACK section below specifies that "offline.html" will be served in place of all files in the /html/ catalog, in case an internet connection cannot be established:

FALLBACK:  
/html/ /offline.html

**Note:** The first URI is the resource, the second is the fallback.

To illustrate this, consider a simple clock applet consisting of an HTML page "clock.html", a CSS style sheet "clock.css", and a JavaScript script "clock.js".

Before adding the manifest, these three files might look like this:

<!-- clock.html -->

<!DOCTYPE HTML>

<html>

<head>

<title>Clock</title>

<script src="clock.js"></script>

<link rel="stylesheet" href="clock.css">

</head>

<body>

<p>The time is: <output id="clock"></output></p>

</body>

</html>

/\* clock.css \*/

output { font: 2em sans-serif; }

/\* clock.js \*/

setInterval(function () {

document.getElementById('clock').value = new Date();

}, 1000);

If the user tries to open the "clock.html" page while offline, though, the user agent (unless it happens to have it still in the local cache) will fail with an error.

The author can instead provide a manifest of the three files, say "clock.appcache":

CACHE MANIFEST

clock.html

clock.css

clock.js

With a small change to the HTML file, the manifest (served as [text/cache-manifest](http://developers.whatwg.org/iana.html#text/cache-manifest)) is linked to the application:

<!-- clock.html -->

<!DOCTYPE HTML>

<html manifest="clock.appcache">

<head>

<title>Clock</title>

<script src="clock.js"></script>

<link rel="stylesheet" href="clock.css">

</head>

<body>

<p>The time is: <output id="clock"></output></p>

</body>

</html>

Now, if the user goes to the page, the browser will cache the files and make them available even when the user is offline.

Authors are encouraged to include the main page in the manifest also, but in practice the page that referenced the manifest is automatically cached even if it isn't explicitly mentioned.

#### The cache manifest syntax

##### 6.7.2.1 Some sample manifests

*This example manifest requires two images and a style sheet to be cached and whitelists a CGI script.*

CACHE MANIFEST

# the above line is required

# this is a comment

# there can be as many of these anywhere in the file

# they are all ignored

# comments can have spaces before them

# but must be alone on the line

# blank lines are ignored too

# these are files that need to be cached they can either be listed

# first, or a "CACHE:" header could be put before them, as is done

# lower down.

images/sound-icon.png

images/background.png

# note that each file has to be put on its own line

# here is a file for the online whitelist -- it isn't cached, and

# references to this file will bypass the cache, always hitting the

# network (or trying to, if the user is offline).

NETWORK:

comm.cgi

# here is another set of files to cache, this time just the CSS file.

CACHE:

style/default.css

*It could equally well be written as follows:*

CACHE MANIFEST

NETWORK:

comm.cgi

CACHE:

style/default.css

images/sound-icon.png

images/background.png