# HTML5 Manifest Demo Application

It's becoming increasingly important for web-based applications to be accessible offline. HTML5 cache manifest can be very useful to serve this purpos. But before we cache all the pages and images in user’s browser we need to think which website we are going to cache and is caching will be usefull at all for that website. In cases where data chane frequently we may not want to cache those, also we may have some page which we don’t not to be cached. Using Manifest file we can configure all this. Let me explain with an example.

In this application I am trying to explain the usefulness of caching using HTML5 Manifest apporach.

I will be creating a Node server and from that will be serving the static files and the appcache file since for appcache to work content-type is mandetory.

Prerequisite:

1. Nodejs need to be installed
2. Express (to create server)
3. Fs (to read appcache file)
4. Chrome

Server:

Creating a simple express server on 8082 port and serving the app folder as static.

const express = require('express');

const fs = require('fs');

const app = express();

app.use(express.static('app'));

const server = app.listen(8082, function () {

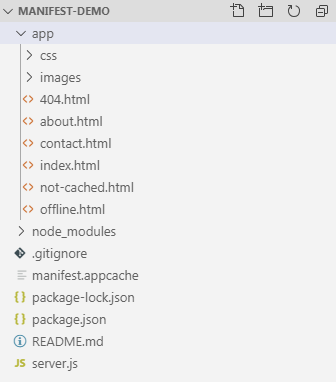
    const host = server.address().address

    const port = server.address().port

    console.log("Example app listening at http://%s:%s", host, port)

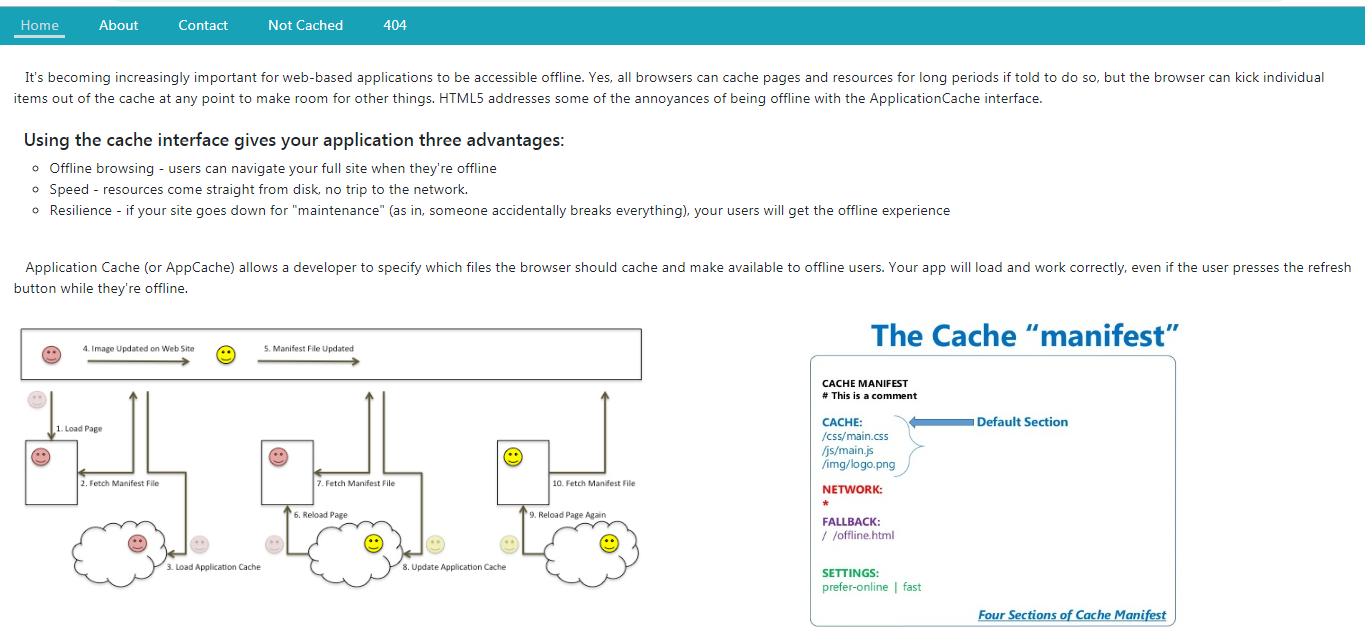
});

Following is the folder structure:



## Index.html

Create a simple HTML file as index.html and add external css files to style the application.





Here the important thing is the manifest=”appcache” in html tag, this will use the manifest file and cache the content as per the manifest.

Appcache:

To serve the appcache from node server I create an endpoint

app.get('/appcache', (req, res) =>{

    res.setHeader('content-type', 'text/cache-manifest');

    const content = fs.readFileSync('manifest.appcache', 'utf8');

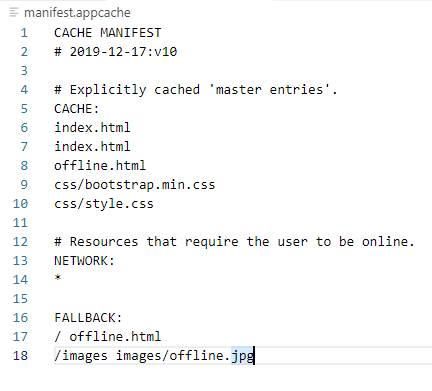
    res.end(content);

})

This will set the header content-type as text/text-manifest this is important that the file content to be served with this content type. This will tell which file to cache and which not.

## Manifest file:

Lets check what we have in manifest file.



First the date aand version is important. Once browser cached the files it will not update even you change the file content and user will see the older one untill you change this version.

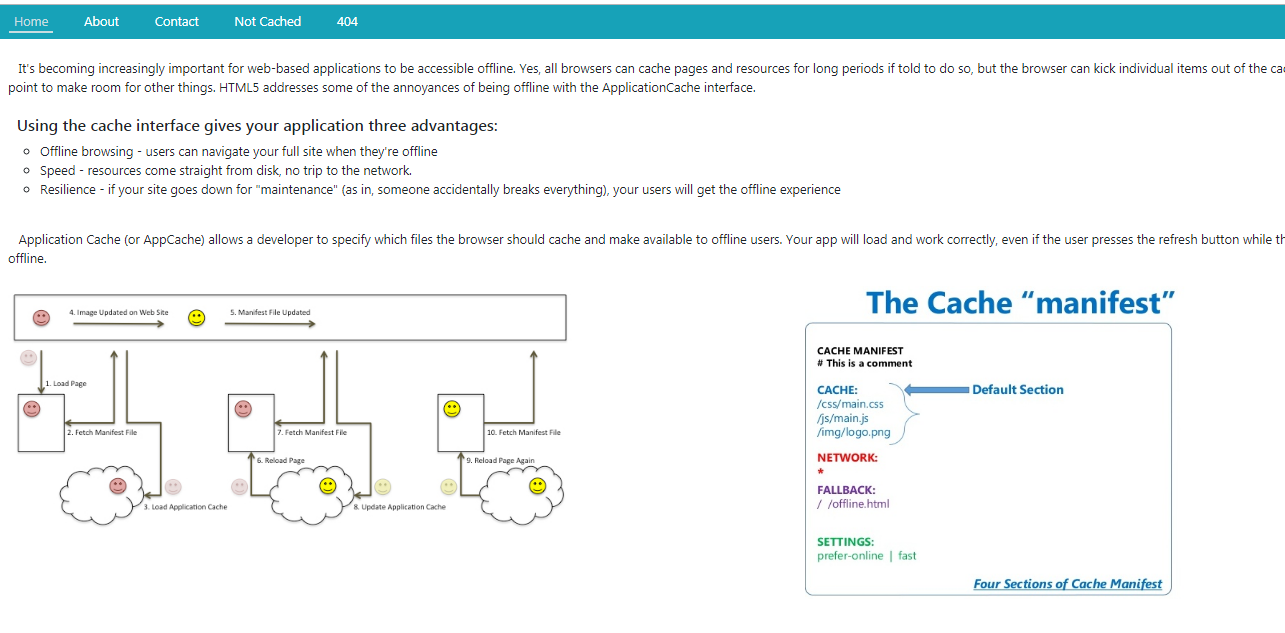
Cache: In this section we tell which files to cache. This is the default section for entries. Files listed under this header will be explicitly cached after they're downloaded for the first time.

Network: Files listed in this section may come from the network if they aren't in the cache, otherwise the network isn't used, even if the user is online. You can white-list specific URLs here, or simply "\*", which allows all URLs. Most sites need "\*".

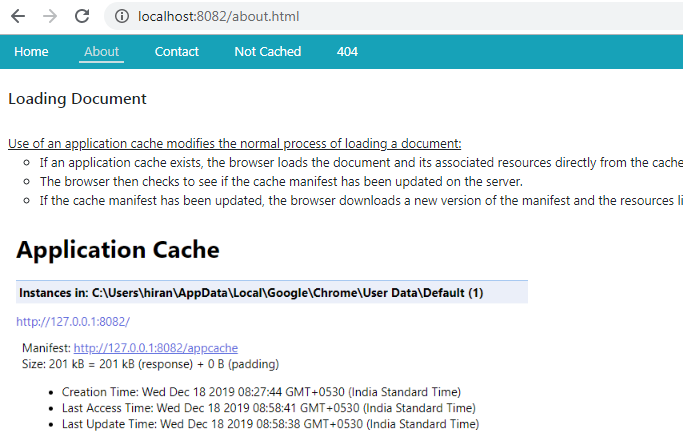
Fallback: Here we tell if you are offline and if any file is not cached then which fill to be shown as fallback. Here in above example if anything is not found it will show the offline.html page and if any image inside /images folder to be shown as offline.jpg.

Lets check in action:

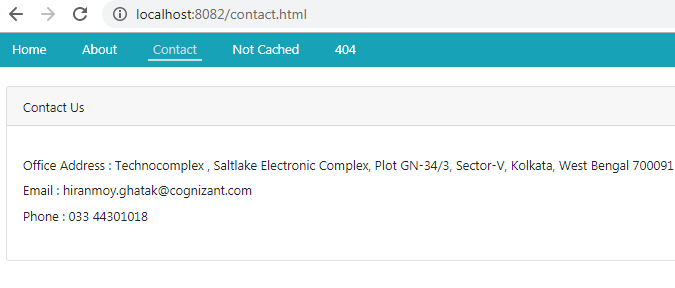
In cmd run: node server.js to run the server then open chrome and hit: <http://localhost:8082/> and navigate through all the pages.



About:



Contact:



Now now stop the server and try again to navigate the application and see the difference:

HOME:

As you can asee in below screens the images are replased with offline.jpg.

|  |  |
| --- | --- |
| **Online** | **Offline** |
|  |  |

About:

The file is available offline but the image is not.

|  |  |
| --- | --- |
| **Online** | **Offline** |
|  |  |

Contact:

Page is available offline.

|  |  |
| --- | --- |
| **Online** | **Offline** |
|  |  |

### Not Cached:

This file is not cached offline, so it is not available.

|  |  |
| --- | --- |
| **Online** | **Offline** |
|  |  |

404:

|  |  |
| --- | --- |
| **Online** | **Offline** |
|  |  |

## Clear all cache:

Open chrome and hit [chrome://appcache-internals/#](chrome://appcache-internals/)

To see all the cached files and you can remove all.