VillageTelco

Digital Library

Preparing Library Content

Ver 1.0

Draft

Note: This document is intended to be read in conjunction with Digital Library Ver 1.0 firmware.

Table of Contents

1. Introduction

One of the most important aspects of setting up a Digital Library system is assembling the library content.

The content must be relevant to the intended audience at the site where the Digital Library is to be deployed, it must be easy for users to navigate, and it must be delivered to the client devices quickly and correctly.

There are many ways in which to organise the files that make up the Library content, ranging from simply storing a few files on a USB/SD Memory device, through to building an elaborate HTML based web page to navigate the Library.

This document will describe several of these approaches.

1.1 Digital Library Basic Operation

The Digital Library essentially operates as a micro web server i.e. it will respond to HTTP requests and return the requested files to client devices, which generally will be running a web browser application.

The Digital Library does no “server side” processing; it simply sends a copy of the requested files as they are stored in its memory, along with some header information that the client Web Browser can use to work out what to do with the files.

The Digital Library web server sends some header information that identifies the MIME Type based on the extension of the file e.g. ".jpg" => "image/jpeg" or ".mp4" => "video/mp4".
(Note that in DL Ver 1 software, this header information is limited to a small number of common file types. DL Ver 2 software has an extensive mapping of file types.)

The sort of file types that you might store in the library and then request from the client Web Browser would include:

- Simple text files
- PDF document files
- HTML files, including JavaScript
- Picture files e.g. JPEG, PNG
- Video Files e.g. MPEG-4, WebM

All of these file types should be able to be directly handled by a modern Web Browser such as FireFox or Chromium/Chrome.

Where a client Web Browser can not directly handle the file /MIME type, it may simply download the file and store it on local disk, or it may launch another application and pass the file on to that application to run.

Note that if there is no USB/SD Memory installed in the Digital Library device, then a Home Page showing the default Digital Library content will be shown, but no content will be accessible.

2. Organising Library Content Files

The following sections describe various methods of organising the library content files.

A Simple Collection of Files

The simplest way to deploy library content is to copy some files onto a USB/SD Memory device and install it into the Digital Library device, then apply power.

When the client web browser accesses the DL device, it will show a listing of the files and directories stored on the memory device.

Clicking on a file name will open the file in the Web Browser, or download it if the Browser can not handle the file type directly.

Clicking on a directory name will open a list of files and sub-directories, and so on.
The parent directory is shown as **../** at the top of the listing.

So a user can navigate through the file system on the memory device.

An example directory listing is shown below:

../

modified: Sat, 16 May 2020 18:32:23 GMT
directory - 4.00 kbyte

1.**Directory/**
modified: Tue, 31 Dec 2019 14:40:45 GMT
directory - 4.00 kbyte

2.Digital-Library-Hardware-Ver-1.0.pdf
modified: Tue, 31 Dec 2019 14:07:58 GMT
application/pdf - 259.12 kbyte

3.Digital-Library-Intro.pdf
modified: Tue, 31 Dec 2019 14:07:47 GMT
application/pdf - 35.58 kbyte

4.Digital-Library-Manual-Ver-1.0.pdf
modified: Tue, 31 Dec 2019 14:07:21 GMT
application/pdf - 874.66 kbyte

Note that if there is no USB/SD Memory installed in the Digital Library device, then a Home Page showing the default Digital Library content will be shown, but no content will be accessible.

Where you want to logically group files together, you can place associated files into a set of sub-directories. You can build up a hierarchy of directories that represent a logical way to navigate through the content.

2.1 Using HTML Index Files

Adding an ***index.html*** file to the USB/SD memory device will cause the Digital Library to send this as the Home page when the client Web Browser accesses the URL for the DL
(e.g. http://digital-library)

If your ***index.html*** file looks like this:

```
<html>
        Hello World!
</html>
```

Then what you will see on the Home page will be: ***Hello World!***

You can create an ***index.html*** file that is as complex as you wish, and include JavaScript and similar code that the client web browser can use to render a complex page, or set of pages.

The Home Page created by the ***index.html*** file is obviously a way to provide a top level index to all the files stored on the USB/SD memory by embedding links to the files.

Using Sub Directories

Where you want to logically group files together, you can place associated files into a set of sub-directories, and so you can build up a hierarchy of directories that represent a logical way to navigate through the content.

Each sub directory can contain its own ***index.html*** file, and the top level ***index.html*** file can contain links to each of these in order to provide a simple web page based navigation scheme.

3. Default Digital Library File Layout

The Default Library consists of the following:

- A top level ***index.html*** file
- Supporting files e.g. CSS files
- Directories containing supporting files e.g. ***art*** and ***css*** directories
- A ***local*** directory
- A ***modules*** directory which contains a number of sub directories, one for each major component of the library content e.g. Wikipedia for Schools, PHET, Khan Academy etc

When the client Web Browser accesses the Digital Library URL, the top level ***index.html*** provides the Digital Library Home page which lists all the major modules in the library, as well as a Menu bar which links to other functions.

Each library module has a corresponding sub directory within the ***modules*** directory, and each of these sub directories contains its own ***index.html*** file, which is a ***Home*** page for that module.

When a user clicks on a module name on the Home page, the page changes to the “Home” page for the module.

3.1 Editing the Home Page

Adding and removing items from the Digital Library default Home Page is fairly straightforward.

The top level ***index.html*** file is organised in sections corresponding to the major library modules, with a link to the ***index.html*** file for each of the modules.

A sample portion of the file is shown below, containing sections for the African Stories, Blockly Games, and BookDash library modules.

In the sample, you can see that the ***<!-- position:21 -->*** section in the file refers to the ***African Stories*** module which is stored in the ***en-afristory*** subdirectory.

The contents of this section are as follows:

- <***a href...***> tags point to the index.html file for the module ("modules/en-afristory/index.html")
- <***img...***> tags contains reference to an image file to display the thumbnail picture
- <***h2***> tags contain the title of the item on the page
- <***p***> tags contain the text describing the item

To remove this item from the Home page, you could delete all the lines in the ***Position 21*** section, or temporarily comment them out by surrounding them with comment tags <!-- … -->

Similarly, to add a new item to the Library, create a new sub directory in the ***modules*** directory to contain the relevant files (including the ***index.html*** file for the module), then copy the layout of an existing section in the top level ***index.html*** file and edit the file and directory references to point to the new ***modules*** sub directory.

Sample Home Page index.html file:

```
<!-- position:21 -->
<div class="indexmodule">
   <a href="modules/en-afristory/index.html" target="content"><img src="modules/en-afristory/af.png" alt="African
Story Books"></a>
   <h2><a href="modules/en-afristory/index.html" target="content">African Story Books</a></h2>
   <p>  Open access to picture storybooks in the languages of Africa. For children’s literacy, enjoyment and imagination
   </p>
</div>
<!-- position:22 -->
<div class="indexmodule">
   <a href="modules/blockly-games/index.html" target="content"><img
        src="modules/blockly-games/en/index/title.png" alt="Blockly Games"></a>
   <h2>  <a href="modules/blockly-games/index.html" target="content">Blockly Games</a></h2>
   <p>  <b>Blockly Games</b> is a series of educational games that teach programming. It is designed for children
who have not had prior experience with computer programming. By the end of these games, players are ready to use
conventional text-based languages.    </p>
</div>
<!-- position:23 -->
<div class="indexmodule">
   <a href="modules/en-bookdash/index.html" target="content"><img src="modules/en-bookdash/logo.png" alt="Book
Dash"></a>
   <h2><a href="modules/en-bookdash/index.html" target="content">Book Dash</a></h2>
   <p>  <b>Book Dash</b> gathers creative volunteers to create new African storybooks that anyone can freely print,
translate and distribute.    </p>
</div>
```

3.2 Assembling the Default Digital Library Content

Details for how to assemble the default Digital Library content on a USB/SD memory device are given in the ***Digital Library User Guide***.

Essentially, the process involves downloading and copying the files from the Library Template onto the memory device, and then downloading and copying the individual library modules into their respective sub-directories in the ***modules*** directory.

Most of the library modules may be downloaded from the ***OER2GO*** website (oer2go.org).

You can also copy the library content from an existing Digital library USB/SD memory device.

Note that for storing the large amount of data and files in the library, you will need to format the memory devices appropriately with a 4kB block size.

Recommended formats are ***ext2*** (readable on Linux systems) and ***exFAT*** (readable on Linux, Windows, Apple systems). Devices formatted as FAT32 will work on the Digital Library, but are recommended only for small numbers of files, and not for the main library.

4. Video Files

Video files need particular consideration for use in the Digital Library.

These files are generally quite large, and so can consume a lot of network and CPU resources in sending them to client devices. And if the whole file has to be sent first, there will be a considerable delay before the video can start playing.

A typical MPEG-4 file for a one hour video might be 100MB in size, so the average data transfer rate for each client accessing the file is (100 x 10 x 10E6 bits) / (60 x 60 seconds) ~= 3Mbps

The WiFi sub system can typically transfer data at upwards of 100Mbps so can easily keep up with the data demand from multiple clients, provided that the data can be sent in relatively small chunks, rather than as the whole file at once. However if the video

Video files can be easily optimised for use with a web server. Tests with a typical Digital Library device show that it can readily stream optimised videos to 20 or more client devices concurrently.

4.1 Optimising Video Files

Refs: https://medium.com/canal-tech/how-video-streaming-works-on-the-web-an-introduction-7919739f7e1

https://www.keycdn.com/blog/video-optimization

https://blog.dareboost.com/en/2018/01/optimize-your-mp4-video-for-better-performance/

The above references cover several aspects of video file optimisation including appropriate scaling for the intended screen size, choosing the right format, and configuring the file for best streaming operation.

If video files are configured correctly for Internet use, they will be compatible with mainstream web browsers, display with good resolution, load and run quickly. The video data will be transferred in small chunks rather than as a complete file, thus maximising the network efficiency of the server.

Our primary focus here is on MPEG-4 files as they are an earlier standard and the most common, and are compatible with a wide range of web browsers.

However the basic principles apply to other file types.

For future use, the Webm file type (derived from .mkv video file type) will likely become more common as it has been designed specifically for Internet use.

Scaling

Scaling is key to getting a reasonable file size for your video, and for presenting it with good video quality and resolution.

Utilities such as FFMPEG, HandBrake and Blazemp provide facilities for re-scaling video files. Smaller files will obviously require less data transfer and will load faster. Choose a scale that preserves good video quality for the types of screen you are targetting.

For the Digital Library application, client devices will typically include PCs, Tablets and Smartphones, so you need to ensure that the scale you choose displays the video at a good level of quality on the largest screen, typically a PC screen.

MP4 Metadata

For MP4 video files, there is a metadata block (called the "***moov atom***") which may be placed before or after the main video data block (called the "mdata atom").

For "fast start" or "web optimized" MPEG-4 video files, the ***moov atom*** needs to be placed at the start of the file, ahead of the ***mdata atom,*** so that the browser can get the information it needs to start playing the file immediately.

Various utilities can be used to process the video files in order to optimise them e.g. FFMPEG, HandBrake, VLC and AVCONV.

For FFMPEG, a typical command to optimise an MPEG-4 file for ***faststart*** is:

```
  $ ffmpeg -i origin.mp4 -acodec copy -vcodec copy -movflags faststart
output.mp4
```

Where: ***origin.mp4*** is the input file; ***output.mp4*** is the optimised output file.

4.2 Embedding Video in HTML Page

HTML5 includes the ***video*** tag specifically for including a video in a web page.

Some simple examples are shown below. These will produce a blank page with a video window including controls for start, pause and progress. Additional page attributes can be added with additional HTML code.

Example 1

```
<html>

<video controls="controls">

    <source src="/video.mp4" type="video/mp4" />

    Your browser does not support the video tag.  <!-- Alternate text -->

</video>

</html>
```

Example 2

```
<html>
  <head>
    <meta charset="UTF-8">
    <title>My Video</title>
  </head>
  <body>
    <video src="some_video.mp4" width="1280px" height="720px" />
  </body>
</html>
```

A common technique used to create a web page to navigate a collection of videos is to create a simple HTML file as above for each video, and provide a link to each of these on the main web page.