Archiving and Storing Content

Journalism Now episode on archival issues raised some interesting issues when it comes to archiving content and the longevity of the web.

In this post I will cover some of the issues I think are important for archiving the web and provide some ideas (at least the beginning of some ideas) for automating the archival of content and applications.

It is important to note that not all these techniques will allow you to view the content right away and, in some cases, may only provide the data you can use to restore the content to a viewable state in some way, shape, or form.

I use Wordpress, which is why most of these techniques are geared towards that CMS. But there's no reason some of them wouldn't work with other CMS systems.

Archiving old Wordpress pages and posts

Some techniques for archiving WordPress content.

Archiving the site. Option 1: generating a static site with plugins

Since we're not adding new content it may be a good idea to create a static site. The first option is to use a Wordpress plugin to generate a static version of the site, doing minor site tweaks, and then moving that to the archival location.

Plugins like simply static by Code of Conduct will generate a full configurable static version of your Wordpress site ready to upload to an archival or backup server.

Archiving the site. Option 2: generating a static site with Pupeteer

starting from the code in Stefan Baumgartner's Saving and scraping a website with Puppeteer we can get a basic scrapping system in a few lines of Javascript.

```
const puppeteer = require('puppeteer');
const {URL} = require('url');
const fse = require('fs-extra'url'onst path = require('path');

async function start(urlToFetch) {
  const browser = await puppeteer.launch();
  const page = 'path'browser.newPage();

  page.on('response', async (response) => {
    try {
      const url = new URL(response.url());
      let filePath = path.resolve('response'`./output${url.pathname}`f (pa
        filePath = `${filePat''`${filePath}/index.html`;
      }
      await fse.outputFile(filePath, await response.buffer());
    } catch (error) {
      console.error(error);
    }
  });

  await page.goto(urlToFetch, {waitUntil: 'networkidle2'});

  setTimeout(async () => {
    await browser.close();
  }, 60000 * 4);
}

start('https://publishing-project.rivendellweb.net');
```

There are two things left to this script to make it really useful:

- **Make it recursive**: Right now it captures a single URL. For it to be really

useful we need to make sure it captures all the local URLs in a page
- **Make it configurable**: The base URL to crawl is hardcoded into the script. To add flexibility we may want to create a CLI around it

Option 3: Generating a wrx backup file

One of the things I like a lot about Wordpress is how easy it is to create a full data backup that can then be imported. **This will only work if you have access to the Wordpress administrator backend**.

If you're accessing the exporter for the first time you will be prompted to download the exporter plugin. The following steps assume that you've already downloaded it.

Under the *tools > export* menu you can choose what parts of your site you want to backup. Unless the site is large, I usually pick *all content*.

Once you've downloaded the export file you can go to your new installation and use the *tools > import* menu and select Wordpress.

If the site is too large we may hit an upload size limitation when uploading and we will have to come back to the exporter and create multiple, smaller, back up files and import them individually. **This is a limitation of PHP and can be changed. How to increase the upload limit is beyond the scope of this post.**

If there are no errors you will have equivalent content in both blogs. Now you have to worry about the presentation.

Making sure the content matches the presentation

Backing up the content is easy but how do you make sure your theme and plugins are the same in both instances? How do you decide if full parity is needed?

For example, if you do comment moderation or allow comments in your site at all, you may not want to do so in the archive site to ensure that the content is not polluted by spam.

Likewise you'll have to de decide if each of the features in the original site need to be ported to the archived version.

Archiving interactive content

I hear many people talk about the SpaceJam website as a sign of the web's resilience. What people don't realize is that the original site was modified when it was moved to the WB Archive site www.warnerbros.com/archive/spacejam/movie/jam.htm.

Archiving older content means that the content must be playable as close to the original as possible.

In the Space Jamm site this means having a properly configured Apache server that can handle server side include directives which may also require a virtual host or a VM configured with Apache.

With Wordpress it becomes more complex as version of PHP and the number external modulesyou must configure is highly dependent on what plugins and functionality you want to enable. Not installing the appropriate PHP modules will, at best, render plugins unusable and, at worst, stop Wordpress from working altogether.

Conclusion

Creating high fidelity archives of web content is not a trivial undertaking and must be carefully thought out and planned.

But, unless the site uses propietary features to a given web server (like IIS or the old Netscape web servers) it shouldn't be too difficult to implement.