AROUND FOR DECADES, GONE IN A FLASH

How we dealt with Flash objects at the National Archives of the Netherlands

Remco van Veenendaal
National Archives of the Netherlands
Netherlands
remco.van.veenendaal@nationaalarchief.nl
0000-0002-2351-1677

Lotte Wijsman
National Archives of the Netherlands
Netherlands
lotte.wijsman@nationaalarchief.nl

Jacob Takema
National Archives of the Netherlands
Netherlands
jacob.takema@nationaalarchief.nl

Marin Rappard
National Archives of the Netherlands
Netherlands
marin.rappard@nationaalarchief.nl

Abstract – In 2020, Adobe announced that they would end support for Adobe Flash Player. Initially, we (the preservation team at the National Archives of the Netherlands) assumed we had only a few or no Flash objects in our digital repository, but this assumption turned out to be incorrect. The discovery of Flash objects in our holding led to the start of a research project to answer several questions. Through a series of dedicated meetings, we formulated a strategy focused on preserving ongoing accessibility to our Flash objects through emulation. We were curious to find out *if* we had Flash objects, *where* they were located, and *which solution* would help us render these objects. This was done with the use of the three preservation functions (Watch, Action, and Planning). After locating the Flash objects, we were able to test potential solutions. The results were then applied to our situation at the National Archives. This led to the development of conclusions and several pieces of advice accompanying those.

Keywords – Flash, emulation, migration
Conference Topics – From Theory to Practice

I. INTRODUCTION

At the National Archives of the Netherlands (NANETH), we have implemented the three important preservation functions, namely: Preservation- Watch, Planning, and Action. Team preservation NANETH uses Watch to undertake an in- and external risk assessment. These risks can be changes in the technical environment, the user community, and organization (e.g., budget cuts). Planning then allows us to develop advice for previously identified risks. This can be done in collaboration with potential stakeholders. Our advice will then be transferred to the collections department, who are responsible for its implementation at NANETH. [1]

Although rumors about Adobe Flash's impending End of Live status had been circulating for a while, it was in September 2020 that our team discovered the news that Adobe would definitively end support of Adobe Flash Player on December 31st, 2020. Adobe would also block Flash objects from running. [2] Initially, we thought the impact to our holdings to be minimal, expecting no or only a few information objects containing Flash to be present in the collection. However, a quick scan of our holdings showed that we did have Flash objects in our digital repository. This led us to change our risk assessment (Watch) of Flash from 'no risk' to 'potential risk'. Before starting the Planning function, we formulated several research questions concerning the subject of Flash content in our archives:

- How many Flash objects do we have in our digital repository?

iPRES 2023: The 19th International Conference on Digital Preservation, Champaign-Urbana, IL, US.

https://hdl.handle.net/2142/121095

- Is the assumption correct that the Flash content can mainly be found in web archives/websites?
- What is the impact of the Flash content, and can Flash be rendered in pywb[1] or another viewer?
- What are possible strategies for keeping Flash sustainably accessible?

Our goal was to identify the magnitude of the problem, the potential solutions, and selecting which one would suit our organization best. Through a series of meetings dedicated to researching Flash, we eventually formulated a strategy or advice for preserving our Flash objects in a way that ensures their ongoing accessibility through emulation. Setting up several dedicated meetings, we ensure within our team that there is opportunity to work on these extended projects with the entire team. This way, we can learn from and with each other while also working toward a final product (e.g., an advice, research report).

II. What is Adobe Flash?

Adobe Flash is a software platform that allows for animations, web videos, and web application (e.g., games and websites) to be created. It was primarily used to design websites and advertisements on websites, also known as banners. Subsequently, Adobe Flash Player is the viewer that could be used to view the content that we created with Adobe Flash. Flash had an immense user base for creating interactive websites at first. However, with the introduction of HTML5 this decreased. Moreover, security issues were identified, which led Adobe to transfer to the Adobe Air platform. Flash Player was eventually deprecated in 2017 and became end-of-life in 2020 for all users outside of China and the non-enterprise users. [3]

III. Finding Flash

We used several methods to answer the question of how many Flash objects are present in our digital repository. We conducted our initial search for Flash objects by extension. We had already found out that, while there are more options for Flash objects, the extensions .fla, .swf, and .flv were most relevant to our holdings.

- The Macromedia Flash FLA Project File Format with .fla extension is the 'authoring' format for the application software. It's a proprietary format and therefore only able to be created and edited in Adobe Animate and Adobe Flash Pro. Objects in this format contain the original, uncompressed source files for Flash animations and applications and are used to store vector graphics, pictures, text, animation timelines, and other components necessary to make a Flash project. They also include metadata such as project settings and scripting code required to provide interactivity and other project abilities.
- For distribution, the 'final result' of these FLA project files is typically exported to a Shockwave Flash file, the compiled format for sending Flash content over the internet. SWF files are formed by assembling and compressing the FLA file's assets and elements. The assembled SWF (pronounced 'Swiff') file includes all of the information required to show and interact with the content, such as the timeline and stage attributes. SWF files can include complex features such as scripting, vector graphics, and multimedia playback in addition to animations and interactive content. The SWF files are compressed and optimized to reduce their size, which results in them not being easily modified or edited.
- Alternatively, FLA projects can also be exported to Flash Video or .flv files. This is a video container that supports a variety of video codecs and several audio codecs. These files can still be opened with software such as Adobe Animate (multiplatform), Media Player Classic (Windows), VideoLAN VLC media player (multiplatform) and individual objects in this format, depending on the codecs used, might therefore be less 'at risk' than previously mentioned formats.[2]

Unfortunately, simply searching by these extensions was not foolproof. It resulted in giving us false hits in

[1] Pywb is a Python web archiving toolkit for replaying web archives. From: https://github.com/webrecorder/pywb.

[2] See PRONOM and https://www.loc.gov/preservation/digital/formats/fdd/fdd000132.shtml.

addition to giving us valid results. This was due to the fact that our digital repository, obviously, also considered text containing our search terms as a hit. By using an added filter, we were able to fill in the search terms in the File name field. This gave us exclusively Flash objects. However, we are aware that extension is not a solid guarantee for finding file formats. For this reason, we use PRONOM and Digital Record Object Identification (DROID) in our digital repository. Using the PRONOM Unique ID (PUID), we assembled a list to search NANETH's digital repository for Flash objects. This resulted in the following search query:

"x-fmt/382" OR "fmt/507" OR "fmt/757" OR "fmt/758" OR "fmt/759" OR "fmt/760" OR "fmt/671" OR "fmt/762" OR "fmt/763" OR "fmt/764" OR "fmt/765" OR "fmt/766" OR "fmt/767" OR "fmt/768" OR "fmt/769" OR "fmt/770" OR "fmt/771" OR "fmt/772" OR "fmt/773" OR "fmt/775" OR "fmt/776" OR "fmt/505" OR "fmt/506" OR "fmt/104" OR "fmt/105" OR "fmt/106" OR "fmt/107" OR "fmt/108" OR "fmt/109" OR "fmt/110"

After this advanced search, we discovered that we do have Flash objects in our digital repository, at two levels: as single objects, in a folder structure of a website, and as objects in ZIPs or WARCs, as part of a harvested website. The search yielded nine results:

- Four separate objects
- One ZIP-file
- Five WARCs

With the ZIP-file and the WARCs, we had now discovered they contained Flash objects, but not how many and that they contained. Further research outside our digital repository resulted in figuring out there were three Flash objects in the ZIP-file and a total of nine in the WARCs. In addition to not being able to directly query our digital repository to find out how many Flash objects we have, we also didn't know the exact location of the Flash objects within those containers. To figure this out, you have to look inside the containers, by unzipping them (ZIP), for example, or creating indexes for them (WARC). Therefore, we downloaded the ZIP-files to our laptops to look inside the map structure. With the WARC-files, we downloaded them to our laptops so we could open the files with Notepad++ (a source code editor). [4] Opening the WARC files in Notepad++ allowed us to look at the entire WARC and the building blocks within it. By using Ctrl + F we could search for the extensions previously identified (.fla, .swf, and .flv). This will show us where the Flash content is present within the website and gives a slight indication to what it is about.

In addition to not being able to directly query our digital repository to find out how many Flash objects we have, we also didn't know the exact location of the Flash objects within those containers. To figure this out, you have to look inside the containers, by unzipping them (ZIP), for example, or creating indexes for them (WARC).

In total, we found four individual objects, three objects in a ZIP-file, and five WARCs, bringing us to a total of nine objects. Among the Flash objects were several interactive maps. We also found audio files and headers that were loaded into an interactive Flash object. In absolute terms this may not sound like a big problem, but at the time of this search there were about ten ZIPs and 25 WARCs in the digital repository. Relatively speaking, a tenth of our ZIPs and a fifth of our archived websites were in danger of information loss. Since governmental organizations have a period of 20 years to transfer archival records not selected for destruction to NANETH, we can only expect these numbers to grow in the coming years. This idea was further strengthened after our more specific research into the Flash objects in our repository showed that one of the objects was merely a reference to another website. The web page, part of the website of the minister of the Interior and Kingdom Relations of Netherlands, shows a small article that warns against bicycle theft (a very important issue in a country with more bicycles than people). Accompanying the article is a hyperlink to a video (the Flash object in question). However, this video is not present on the harvested website, but on the website of another ministry. This ministry has not yet transferred their web archive to us, so we can expect this video in the next couple of years in our digital repository.

Our second question during this stage was if the assumption was correct that our Flash content could solely be found in web archives/websites. After our search, this assumption was found to be correct. The separate Flash objects are located in the folder structure of an archived website, while the ZIP files are a compressed website. It is still possible that future transferred archival records with Flash objects will not be limited to websites. They could for example be cd-roms with Flash animations in

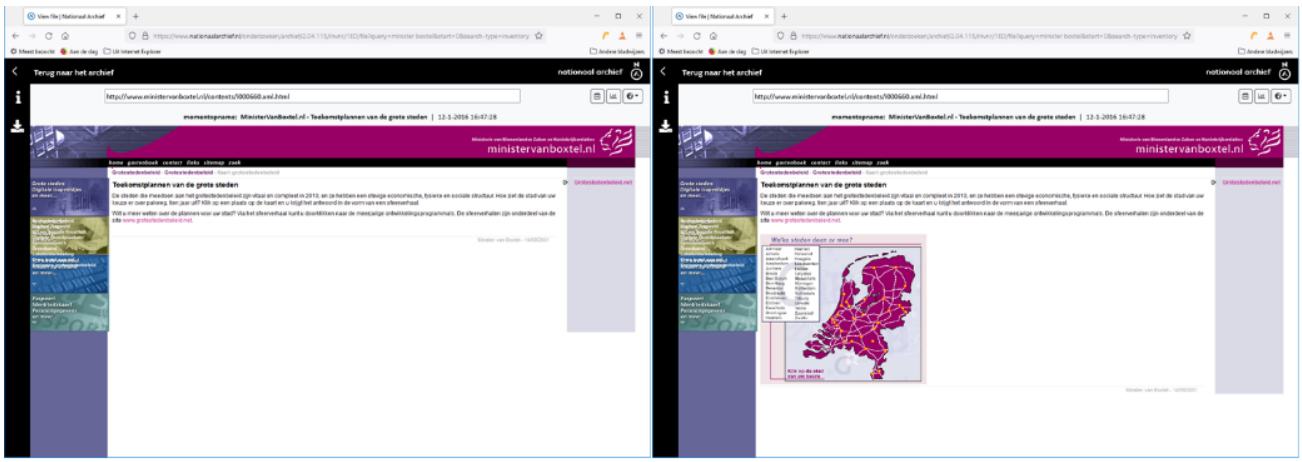

Figure 1. Without the Flash content (left) and with Flash content (right).

government campaigns, raising the public's awareness of some topic. However, the current situation and short-term predicted situation is limited to websites.

Having identified the Flash content within our digital repository, we were able to establish the impact further by trying to view the content. A good example was an archived webpage of the website of the Minister of Metropolitan- and Integration Policy, Roger van Boxtel. The website was archived in 2016 from servers, the date on the website is August 22nd, 2002.[3] The Flash object on this page is a map of the Netherlands wherein larger cities can be selected. When selecting one of the cities, the user is then sent to a story about future plans for that particular city.[4] Fig. 1 shows the impact of not being able to view the Flash content on the left. On the right, the user is able to see the website as a whole, including Flash content. As you can see in fig. 1, we did eventually succeed in being able to see the Flash content. In chapter III we will elaborate on this more.

Users can of course use (website) viewers at home. However, these are not equipped to show Flash content by default. Moreover, our users do not have the information that we have Flash content in our web archive, how much there is, where they are situated exactly, and what the best solution is to view these objects. This gives us two possible solutions:

- NANETH-side solution: implementation on the side of NANETH. This allows our users to view our web archive without investing time to research how to view it and subsequently installing that solution.
- User-side solution: implementation by the user. We find this to be less ideal since it expects a certain degree of research and effort on the side of the user. At the National Archives we are committed to and stand for low-threshold sustainable accessibility. If you expect your users to install viewers, you create barriers. This is also why we find a client-side solution for Flash in Web sites/web archives undesirable.

IV. Dealing With Prejudice

The two preservation strategies considered for keeping Flash content accessible were file format migration and emulation. With migration you migrate the information from an older or less durable file format to a more modern or durable file format. Emulation allows you to mimic the old hardware and software environments in a modern hardware and software environment. [5] Our preservation policy doesn't explicitly state a preferred choice between the two. However, our daily practice shows a clear inclination towards migration. This is due to, for example, the technical and legal challenges included in emulation. This led

[3] https://www.nationaalarchief.nl/onderzoeken/archief/2.04.115/invnr/1ED/file?eadID=2.04.115&unitID=1ED

[4] The website the map links to, www.grotestedenbeleid.net, is no longer online. However, it has been archived by the Internet Archive. Thanks to the map's link, users can find, for example, https://web.archive.org/web/20030516013940/https://www.grotestedenbeleid.net/www/sfeermenu/sfeer/amsterdam/index.html (accessed 6-3-2023) there. Without the map, without Flash, users of the archived website of Minister van Boxtel miss the link between the cities of the map and the atmospheric stories of the metropolitan policy.

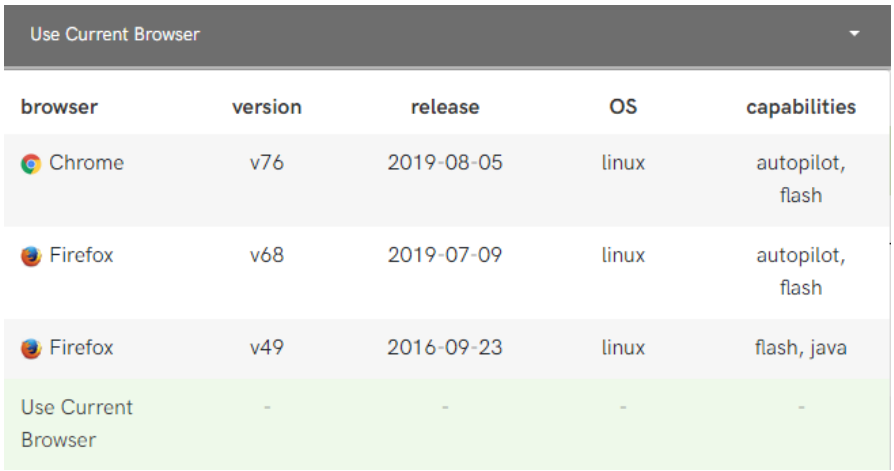

Figure 2. Conifer gives the user the option of multiple browser-versions. As seen in the figure, both Google Chrome v76 and Firefox v68 have the capability to render Flash objects.

us to start with investigating migration as a potential solution. However, to be able to properly assess potential solutions, we first needed to see the Flash content rendered to know exactly what we are dealing with.

A. Rendering

To render the Flash content, we tested three ways:

- Conifer [6]
- Ruffle [7]
- Browsers with older version of Flash Player [8]

Previously, we mentioned our inclination toward migration. However, these three ways are all emulation-based. Our initial searches did not yield any migration-based solutions. At this time, we started to realize that our inclination towards migration would be based on the previously mentioned outdated prejudice that surrounds emulation.

In addition to using our own collection for these tests, we also used a collected corpus of Flash objects. This corpus was compiled with, among others, the use of the Internet Archive, the UK Web Archive, and the Apache Software Foundations test sets. [9]

Conifer allowed us to launch an environment containing an emulated older browser with Flash support. Opening the website of Minister van Boxtel in that browser shows the interactive map, as seen in fig. 1. The benefits of using Conifer are that it is open-source, allows the use of multiple browser-versions (see fig. 2), and is free to use. The disadvantages are that you need to register as a user, and the limit set on the amount of concurrent users, which results in waiting times. With Conifer being an online service, this solution would be a user-sided solution. Furthermore, it is unclear to us to what extent an emulated browser passes the security risks associated with Flash Player to the user's computer. However, as long as the emulated browser is offered only for trusted Flash objects from our collection, this risk will be negligible.

Using the Ruffle website, we were able to download a Flash emulator. This standalone version allows the user to render loose Flash objects, while the browser plugin shows the objects in the websites themselves. Using the plugin, we were able to render the interactive map. As with Conifer, Ruffle is open-source and free to use. Ruffle also has the benefit of

Solution	Open source	Installation required	Client- or server-sided
Conifer	Yes	No	Client
Ruffle	Yes	Yes	Client
Browsers with older version of Flash Player	No	Yes	Client

Table 1. Overview of the results of the three solutions tested.

being available as either a standalone version as a browser plugin. However, both of these need to be installed by the user. Therefore, it is a user-sided solution, which is not preferred by us.

Using the Internet Archive, you can download versions of both Mozilla Firefox and Google Chrome that have an older version of Flash Player installed. This third option, like the previous two, also provided us with a rendering of the interactive map. This solution allows the user to view the Flash objects in their 'original environment' using a free download. Nonetheless, as with Ruffle, it is a user-sided solution. Moreover, the risks associated with Flash Player are brought in.

The three solutions each have their own pros and cons connected to them. Table 1 shows an overview of a few findings connected to these solutions.

B. File Format Migration

When we investigated file format migration as a strategy, it quickly became apparent that there is little to no open-source tooling available to migrate Flash objects. NANETH's digital repository offers no tooling for it, and the leading registries in our field such as PRONOM [10], WikiData for Digital Preservation [11], and the Community Owned digital Preservation Tool Registry (COPTR) [12] also don't mention any Flash migration tools. There used to be Google Swiffy, a Flash to HTML5 converter, but this tool was more specifically intended to work on banner advertisements and has since been taken down. [13] There are other commercial providers, but these are expensive and we discovered other reasons why migration does not seem to be the best approach in NANETH's case.

Desk research led us to conclude that there are two main approaches for migrating Flash objects. Interactive Flash objects are often migrated to HTML5. Flash movies can also be migrated to MP4. [14] The Flash objects in our digital repository are primarily .swf files. These are the distributable "compiled" versions of the Flash objects, not the "source code" underlying them. Since there are no known conversion paths from SWF to HTML5, the objects would usually have to be redeveloped from scratch. As Maheswari and Reddy show in their article, the time this takes per object varies from 1 hour to 51 hours. We would assume few organizations will have the resources to allow this as a preservation strategy, depending on the scale of Flash content present. [15] However, even then, it would require specific skills to completely rebuild the look and feel of these objects. Maheshwari and Reddy argue this is because;

- "Recreation of Flash assets like images, vector graphics and animations while adhering to all the aesthetic details is a resource intensive effort.
- Developers may often lack the domain knowledge required for the particular animation. Therefore, they will have to perform the additional step of enumerating all the animation states, before rewriting the entire logic in JavaScript which in itself is a huge task." [14]

An additional complicating factor with migration is that we found many of our Flash objects within archived websites: inside a ZIP-file or WARC. Even if we did manage to migrate those objects into different formats, we would have to unpack the containers, modify all references to the migrated Flash objects, and then repackage the containers. This is a laborious process, requiring us to modify not only the Flash object, but also the container in which it is packaged.

Eventually we came to the conclusion, to our surprise, that migrating our Flash objects did not seem to be the most suitable strategy. This was affirmed during our research of tools to display WARCs that contain Flash objects, where we came across interesting alternatives.

C. *Emulation*

We found that there were ways to render Flash objects available in the open-source domain and often emulation-based. Thereby, solutions are available, such as Ruffle, which do not have the same security problems as Flash Player does. According to Ruffle, it is even possible to offer Ruffle as a server-side solution and embed Ruffle into Web pages containing Flash objects via JavaScript. [16] The user then does not have to install anything themselves. A test with the website of Minister van Boxtel on a test web server showed that this can indeed be realized with little work. However, this did require modifying the web page that contains the Flash object.

A solution that seems to fit our infrastructure even better is related to the Conifer solution previously mentioned. At NANETH we connected the Webrecorder Python wayback [17] web archiving toolkit to our digital repository for the playback of web archives on our website. In 2020 the first incrementally harvested web archives were transferred to NANETH. This led to the first (Agile) user stories calling for a web archive viewer with support for this type of web archives. After some research, we chose to implement pywb. The older browser emulation functionality that has been built into Conifer is available to install in pywb environments and is called pywb remote browsers [18]. This solution allows you to provide an emulated older browser version with Flash support. [19]

An additional advantage of being able to provide emulated older browser versions is, that it allows us to display other archived websites, without Flash objects, in a browser that was common when the website still existed. Developments in browser technology and Internet standards can cause modern browsers to display older websites differently than older browsers, whereby the rendering of the older websites in the "natural habitat" of the older browser may yield a more authentic result.

V. TAKING ACTION

As shown in the previous chapter, emulation, for us, is the preferred preservation strategy for Flash objects. Emulation solutions are readily available and available in the open-source domain. They are even available for our pywb infrastructure.

Our advice for action to be taken is threefold. The first advice is a prerequisite for the other two. Without it, the others cannot be realized.

1. We need to document and/or create metadata in which web archives Flash objects are present.[5] That will allow us at NANETH to inform our users about it, while activating server-side solutions for these archives with Flash objects.
2. Short-term advice: inform the user that they are viewing an archived website that contains Flash objects. Additionally, provide instructions on the actions they would need to take to view the content, like downloading an emulator or plug-in such as Ruffle. This is a user-sided solution, since it requires time and effort from our users.
3. Long-term advice: A server-sided emulation solution would need to be integrated into our infrastructure, so users are able to view our Flash objects without investing their own time and effort. Pywb remote browsers is an example of this server-sided emulation solution. While we can get this started in the immediate future, necessary prioritization and lead time will mean it will take longer to realize than the second advice.

These three pieces of advice have been forwarded to our collections department, together with our research report, so they can implement the most suitable solution. Team preservation is part of another department that serves as a sort of consultancy branch. This means we have preservation advisors present in our team, not acquirers or custodians of our collection. For this

[5] Currently we only have Flash content in our web archives. In time, we can of course receive other Flash content that lies outside our web archives.

reason we have forwarded our advice, so that they can implement a solution.

While our collections department can implement the first two solutions mostly on their own, with possibly some help needed for adjustments to our website, the third advice needs further work that involves multiple departments. User stories will need to be created that work into our continuous project concerning web archiving. Subsequently, our IT department needs to implement this solution.

In chapter III, we briefly mentioned the security risks associated with Flash Player. In both types of solutions (user- and NANETH-sided) this has to be considered at all stages. We as the National Archives, after all, cannot afford to have our users install an unsafe plugin, or send potentially unsafe content to the user's browser. Therefore, we have explicitly stated this in our report.

VI. Conclusion

Flash has been phased out and is no longer supported by default. We have only a few Flash objects in our collection at the moment, but relatively speaking, a tenth of our ZIPs and a fifth of our WARCs are at risk of information loss. We expect that our collection of web archives is going to and will continue to grow substantially. However, the phasing out of Flash is a significant risk for sustainable accessibility, especially when transferring legacy websites. In post-2020 websites, we expect to find little to no Flash.

We found that emulation is a better strategy than migration for rendering Flash objects. Our research resulted in three pieces of advice, which can be realized in stages: document the presence of Flash objects in our collection, inform the user about the presence of Flash and solutions to render Flash objects, and develop a server-side solution for displaying Flash objects.

Each realized advice reduces the risk of information loss. After the realization of our long-term advice (no. 3), web archives with Flash objects can be authentically rendered. The choice of pywb remote browsers allows other Web archives to be displayed in older browsers, which can also benefit their authentic display.

This research taught us a lot about Flash and the object present in our digital repository. This knowledge will help our organization to not only deal with the objects already present, but also with potential future Flash objects transferred to our digital repository. As mentioned previously, our team is part of the 'consultancy branch' at NANETH. We will also use our Flash research to give advice to other governmental organizations and archival institutions.

1. REFERENCES

[1] Pepijn Lucker, Remco van Veenendaal, Marcel Ras, and Barbara Sierman, "Preservation Watch at the National Archives of The Netherlands," iPRES 2018: https://doi.org/10.17605/OSF.IO/KQ6HM.

[2] https://www.adobe.com/products/flashplayer/end-of-life.html. Adobe. "Adobe Flash Player EOL General Information Page." Last modified January 13th, 2021.

[3] https://en.wikipedia.org/wiki/Adobe_Flash. Wikipedia. "Adobe Flash." Accessed February 28th, 2023; https://en.wikipedia.org/wiki/Adobe_Flash_Player. Wikipedia. "Adobe Flash Player." Accessed February 28th, 2023.

[4] https://notepad-plus-plus.org/. Notepad++. Accessed June 25th, 2023.

[5] For the definition of file format migration we used: https://www.archives.govt.nz/manage-information/how-to-manage-your-information/digital/file-format-migration. Te Rua Mahara o te Kāwanatanga Archives New Zealand. "File format migration." Accessed March 3rd, 2023; For the definition of emulation we used: https://www.dpconline.org/handbook/glossary. Digital Preservation Coalition. "Glossary." Accessed March 3rd, 2023.

[6] https://conifer.rhizome.org/. Conifer. Accessed March 6th, 2023.

[7] https://ruffle.rs/. Ruffle. Accessed March 6th, 2023.

[8] https://archive.org/details/Firefox_Chrome_Adobe_Flash. Internet Archive. "Firefox and Google Chrome with Flash Player." Accessed (for this paper) March 6th, 2023.

[9] https://archive.org/. Internet Archive. Accessed March 6th, 2023; https://www.webarchive.org.uk/en/ukwa/. The UK Web Archive. Accessed March 7th, 2023; https://svn.apache.org/. The Apache Software Foundation. Accessed March 7th, 2023.

[10] https://www.nationalarchives.gov.uk/PRONOM/Default.aspx . PRONOM. Accessed February 28th, 2023.

[11] https://wikidp.org/search?q=Adobe%20Flash. WikiData for Digital Preservation. "Adobe Flash." Accessed February 28th, 2023.

[12] https://coptr.digipres.org/index.php/File_Formats_and_Metadata_Formats. Community Owned digital Preservation Tool Registry (COPTR). "File Formats and Metadata Formats." Accessed March 7th, 2023.

[13] P. A. Senster. "The design and implementation of Google Swiffy: A Flash to HTML5 converter." Master's thesis, 2012: oai:tudelft.nl:uuid:cab4b862-d662-432a-afa4-45ccb725177f.

[14] Yogesh Maheshwari and Y. Raghu Reddy. 2017. A study on Migrating Flash files to HTML5/JavaScript. In Proceedings of the 10th Innovations in Software Engineering Conference (ISEC '17). Association for Computing Machinery, New York, NY, USA, 112–116. ¡https://doi.org/10.1145/3021460.3021472 ; https://www.hurix.com/convert-flash-based-websites-html5/ and https://pixelplex.io/blog/tools-to-convert-flash-

to-html5/ are examples of websites that offer more information on migration concerning Flash objects.

[15] Anna Mladentseva (2022) Responding to obsolescence in Flash-based net art: a case study on migrating Sinae Kim's Genesis, Journal of the Institute of Conservation, 45:1, 52-68, DOI: 10.1080/19455224.2021.2007412.

[16] https://ruffle.rs/#usage. Ruffle. "Usage." Accessed March 7th, 2023.

[17] https://github.com/webrecorder/pywb. Github. "pywb." Accessed June 25th, 2023.

[18] A video from the International Internet Preservation Consortium's 2021 Web Archiving Conference (IIPC WAC) demonstrates how this works: https://www.youtube.com/watch?v=XvBt31KgeSk. Youtube. "Not Gone in a Flash: Keeping Flash Accessible in Web Archives (IIPC WAC 2021 Presentation)." Accessed March 7th, 2023.

[19] The source code for this solution is part of the Webrecorder repository on Github. https://github.com/webrecorder/pywb-remote-browsers. GitHub. "pywb-remote-browsers." Accessed March 7th, 2023.