Mink: Integrating the Live and Archived Web Viewing Experience Using Web Browsers and Memento

Mat Kelly, Michael L. Nelson, and Michele C. Weigle
Old Dominion University
Department of Computer Science
Norfolk, Virginia 23529 USA
{mkelly,mln,mweigle}@cs.odu.edu

ABSTRACT

We describe Mink, a new web browser extension that provides a different model for integration of the live and archived web. While a user browses the live web, Mink actively queries the archives and reports other instances of the page in the archives without requiring active querying by the user. Further, by querying the archives dynamically and asynchronously, a user can view the extent to which the currently viewed page on the live web has been archived and proactively submit a request to various archives using an overlay on the live web page and a simple interface.

Categories and Subject Descriptors

H.3.7 [**Online Information Services**]: Digital Libraries and Archives

1. INTRODUCTION

To better integrate the past and live web, implementations of the Memento framework [1] provide the facilities to query the archives (using URI and HTTP Accept-Datetime headers as parameters) to provide resources on the past web called *mementos*. Several Memento clients exist including browser-based plugins for Mozilla Firefox [2] and Google Chrome[1] as well as native app-based mobile clients for both iOS and Android [3]. While Memento support for mobile is already problematic [3], requiring a user to change the contexts of their native browsers gives an *ad hoc* feel of requiring a separate client (e.g., an app). Retaining use to the client normally used to view the live web (i.e., a web browser) is more fluid to the user. The memento browser extensions that exist, however, use either a modal approach (MementoFox sticks in either live or time travel mode) or requires the user to either re-opt to the past with each page visited or rely on the server for selection of the past/live web. We have developed a new browser extension, Mink[2], that instead uses an unobtrusive alert model to remind the user about the past. This model allows the user to quickly poll through the mementos available while maintaining the paradigm of relying on what is returned by the server to determine whether the user stays in the past or returns to the present. The additional feature of allowing the user to seamlessly jump from the past to the present while maintaining a quick return to the past makes Mink's approach unique.

2. ANOTHER APPROACH

The browser-based model of accessing archives is preferable to that of mobile apps. Bombarding the Memento proxies and archives with frequent requests for content negotiation is computationally expensive. We have implemented another approach[3] that utilizes the Memento TimeGate and TimeMap capabilities to provide all references to a URI, which reduces the negotiation complexity and still provides a more integrative model between the live web and the archived web using the user's web browser.

We chose the Google Chrome browser extension environment due to the browser's popularity, but the logic is simple enough to be ported to other browsers. When a user loads a web page with Mink enabled, the extension queries a memento aggregator with the URI as a parameter and expects a Memento TimeMap in return. While processing the request, the extension's icon is present at the bottom right of the browser viewport and provides a "spinning" animation until the TimeMap is received (Figure 1). If the TimeMap is paginated with a reference to a subsequent TimeMap, a button is provided to the user to invoke the iterative fetching of all TimeMaps from the aggregator (which is temporally expensive) or to stop iterating at a number of times set in the extension's preferences, customizable by the user. Regardless of whether the iterative procedure is executed, a badge is set atop the extension icon indicating how many mementos are available for direct access to the extension. This facility allows a user to browse around the web to observe how well pages are archived without needing to commit to browsing the archived web nor to proactively submit a request to the archives to receive this archival metadata about the live web.

Once a user has accessed an archived page using Mink, the interface provides an additional button that allows the user to return to the live web with a single click for easy compar-

[1] `https://chrome.google.com/webstore/detail/memento/jgbfpjledahoajcppakbgilmojkaghgm`

[2] Named for Minkowski Space

[3] Available at `https://github.com/machawk1/mink`

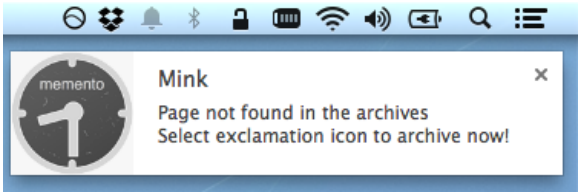
978-1-4799-5569-5/14/$31.00

Figure 1: Mink's basic user interface for mementos that do not require multiple queries to the aggregator.

ison with the archived version. This is accomplished using a combination of HTML5 localStorage referencing and utilizing the native Memento `rel="original"` constructs from the retrieved TimeMaps. Utilizing a hybrid approach makes the extension more robust for discrepancies in the data retrieved from TimeMaps and allows for decoupling from the TimeMap dependency in the cases where this value is not present.

3. ENCOURAGING USERS TO ARCHIVE

If the aggregator reports that no mementos exist for the URI, the extension displays an alert icon using the same overlay method and an unobtrusive system notification (Figure 2(a)). If the user interacts with the alert icon, the UI normally used to browse which mementos were retrieved from the aggregator is instead replaced with Mink's one-click "Archive Now" interface (Figure 2(b)). Selecting a button, each labeled with a web archive's name, sends a request for the page to be archived by Internet Archive (archive.org), Archive.today, WebCite (webcitation.org), or Perma.cc, with an additional option to specify a custom service. Further, an "All" button sends a request for the page to be archived by all services available. The archive request procedure is accomplished by sending an Ajax request to each of the archives' web accessible forms, which would normally require manual interaction with the respective archive's interface. The code to accomplish this is extensible and will support any service that has a web-based submission process.



(a) Notification that page is not in archives

(b) Interface showing one-click options to archive page

Figure 2: Mink's Archive Now interface

4. DOES IT SCALE?

When a user submits a URI to be archived using the web services available, there exists latency in both the Memento aggregator and the Memento interface to the archive (i.e., querying the archives using Memento) as to when the memento exists in generated TimeMaps. To account for this, Mink requests the memento using the TimeGate interface of the aggregator instead of simply the TimeMap. Doing this overcomes the issue at the aggregator level, though other issues with caching TimeMaps exist [4].

One of the primary reasons for content negotiation to the archives in the Memento Framework is the large quantity of content contained within. Even for URIs with a small number of mementos, displaying an interface to allow the user to select a memento was initially problematic, so we explored other approaches. While this is handled in other Memento clients using a Memento-Datetime slider and date-based content negotiation, we did not want to introduce the latter's increased computational complexity requirement. Instead, we experimented with other approaches of drill-down date selection and an adaptation of IA's "bubble" interface using a more common date selection model and a less busy density encoding (Figure 3) using Miller columns, a paradigm commonly utilized in modern operating system and thus likely familiar to the user.

Figure 3: For a large number of mementos (shown here for CNN.com), Miller columns provide an alternative to simple drop down menu selection, which would be taxing for browsers because of the large number of entries in the select box.

A further issue is the temporal complexity required to retrieve a comprehensive TimeMap from the Memento aggregator for a URI, which is dependent on the speed of response from the aggregator. This is especially noticeable for well-archived sites, which require multiple iterations of querying a Memento aggregator to be comprehensive of all mementos available for the URI. We are hoping that further experimentation and the inevitable streamlining of the aggregators will help to resolve this issue.

5. REFERENCES

[1] H. Van de Sompel *et al.*, "HTTP Framework for Time-Based Access to Resource States – Memento," IETF RFC 7089, December 2013.

[2] R. Sanderson *et al.*, "Implementing Time Travel for the Web," *Code4Lib Journal*, April 2011.

[3] H. Tweedy *et al.*, "A Memento Web Browser for iOS," in *Proc. ACM/IEEE Joint Conference on Digital Libraries*, 2013, pp. 371–372.

[4] J. F. Brunelle and M. L. Nelson, "An Evaluation of Caching Policies for Memento TimeMaps," in *Proc. ACM/IEEE Joint Conference on Digital Libraries*, 2013, pp. 267–276.