DHQ: Digital Humanities Quarterly

2025
Volume 19 Number 3

Rondo: A Minimal Single Page Application for Digital Exhibits

Nick Szydlowski <nick_dot_szydlowski_at_sjsu_dot_edu>, San José State University https://orcid.org/0000-0002-6851-7490

DOI: https://doi.org/10.63744/6gthf728hcfj

Abstract

Minimal computing is a promising conceptual framework for digital humanities infrastructure, but the static site architecture most commonly associated with minimal computing can present a steep learning curve, particularly in a workshop or classroom context. This article introduces Rondo, a new minimal framework for digital exhibits which requires no software installation or command line interaction. Rondo differentiates itself from static site tools by adopting a single page application architecture with data stored in a Google Sheet, but it includes tools to disconnect from Google Sheets and create a static site with no external dependencies. Rondo's reliance on Google Sheets reflects an approach to computing which draws inspiration from Agnès Varda's film *The Gleaners and I*. Gleaning, the practice of collecting food or other resources left behind by commercial enterprises like farming, is proposed as a productive framework for exploring the relationship between digital humanities and the technology industry. Rondo's integration with the Digital Public Library of America presents another way to explore the possibilities of small acts of curation, criticism, and juxtaposition using resources gleaned from larger institutions and corporations.

> "Gleaning is the spirit of years past. My mother would say, 'Pick up what's left so nothing's wasted.' But sadly we no longer do, because machines are so efficient nowadays." From *The Gleaners and I* by Agnès Varda (2000)

Introduction

While minimal computing is strongly associated with the use of static site generators as a mode of web development [Dombrowski 2022], the principles of minimal computing can also be applied to other website architectures. Rondo is a tool for creating digital exhibits according to minimal computing principles, but – in contrast to existing tools – it is built as a single page application. [1] Rondo sites are primarily built using a templated Google Spreadsheet that stores data about exhibit items, pages, and site configuration. The Rondo site connects to the Google Sheet via an application programming interface (API) and uses the data from the spreadsheet to configure the site and display its content. Once the site is completed, Rondo has tools to disconnect from Google Sheets and run on local data files. 1

This project is intended for practical use, but it is also an exploration of what it might mean to apply the principles of minimal computing to web architectures other than static sites, and, in particular, what approaches prove most welcoming for users with less web development experience. As much as its architecture, Rondo's reliance on Google Sheets is a departure from typical minimal computing practice. This dependency raises the question of what relationship, if any, minimal computing practitioners have to the corporations that make up the technology industry. At the same time, this architecture allows for an approach that requires both less technical background and less infrastructure than static site generator approaches. 2

As we pursue strategies for web development that are both more inclusive and less resource-intensive, I want to propose a model for thinking about the relationship between individual practitioners and the tech industry. In Agnès 3

Varda's film *Les Glaneurs et La Glaneuse* [Varda 2000], she explores the practice of gleaning or gathering food or other resources that have been thrown out or left behind. In the most traditional form of gleaning, gleaners follow the harvest, collecting food that farmers and their employees leave behind in the field. In addition to this agricultural form of gleaning, which is legally protected in France, Varda's documentary reveals a host of parallel practices in more urbanized settings and goes on to posit the filmmaker herself as a gleaner, collecting left behind bits of experience and beauty and transforming them into a way of life [King 2007].

As minimal computing practitioners negotiate their relationship with powerful technology companies, I would argue that Varda's re-imagination of gleaning as a cultural as well as economic practice is an illuminating model. Rather than becoming fully enmeshed in, and dependent on, the environments created by large corporations, perhaps we can instead glean just what we need from what they leave behind. As gleaners, we would know that the free food or free server space is likely to be in a different place tomorrow, so we wouldn't allow ourselves to become too dependent on any one tool or platform. At the same time, we would acknowledge that we don't live in a world or environment that we can fully control, and that rather than trying to own everything we use, we can find a way in the world by gleaning what we need from those who have more than us. In this way, Rondo enables users to glean from Google Sheets and GitHub Pages, but also to create sites that can stand on their own when those free resources are no longer available. 4

In describing the impetus behind Rondo, this gleaning orientation, or what [Croft 2018] calls "the gleaner's gaze," pops up again and again. It is especially present in Rondo's integration with the Digital Public Library of America, a feature which encourages small projects that make creative reuse of existing digital collections. Particularly when combined with the principles of minimal computing, the gleaner's gaze reveals possibilities in the cracks of our existing technical and scholarly infrastructure, and it is up to us to exploit those cracks and look for ways of working inside them. 5

Precedents and Inspirations

Rondo is very much inspired by the joys and challenges of attempting to practice minimal computing in a digital humanities and library context. In many ways, it is an homage and possible companion to Wax, the popular framework for creating digital exhibits as static sites. [Nyrop and Gil n.d.], or analogous tools like Collection Builder, created using University of Idaho's Lib-STATIC framework [Wilke and Williamson 2024]. In this section, I will describe how its development and approach were shaped by this particular context and the constraints it imposes. 6

As a librarian tasked with creating and supporting new digital humanities projects and digital exhibits, I eagerly adopted Wax, and I was part of teams that created two projects hosted on Wax: *Betita Martínez: A Memorial Exhibit* [Blackmer Reyes and Szydlowski 2022] and *The Unionist Unified: Connecticut's First Immediate Abolitionist Newspaper* [Rycenga et al. 2023]. In production, on projects with small teams and where I was the sole web developer, Wax was a fantastic solution. Hosting Wax sites on GitHub pages solved our most significant potential problem, which was that neither of these projects had any financial resources to devote to web hosting, either in the short or long term. 7

The challenges we would face with Wax became more apparent when trying to design workshops on how to use Wax. It proved challenging to install development environments for Wax on university-owned hardware. In an environment where users are not administrators on their own hardware, it was quite challenging to get the Wax environment up and running. This raised the first question that Rondo tries to solve: Is it possible to do minimal computing in a way that is more compatible with the restrictions that many academic users are likely to face in the workplace? This would require a framework where the site creator is not required to install software in order to create the site. [Dombrowski 2022] and [Brooks 2021] report similar challenges in teaching with static site generators. 8

[Risam and Gil 2022] write that "minimal computing connotes digital humanities work undertaken in the context of some set of constraints." The inability to install software is a frequent constraint in the workplace both inside and outside academia. Content management systems that can be administered within a web browser are a ubiquitous solution to this constraint, and once installation or hosting is in place, platforms like Omeka provide a fully-browser based solution for digital exhibits. However, dedicated software requires significant IT and network resources – what if you could build a digital exhibit without installing software, and without having access to a dedicated platform? 9

The next key point in the imagining of Rondo came from observing how collaborators who were new to digital projects embraced the spreadsheet-based workflow of Wax and applied that workflow even in unanticipated parts of the project. In Wax, exhibit items are created by processing a CSV or other data file, and so the workflow typically begins with a metadata spreadsheet. I noticed that my collaborator then created a spreadsheet for the pages that would appear on the site, and added the text and other information for each of the site's pages there. It was easy enough for me to then build those pages in markdown files for the Wax site, but this also struck me as a key insight on what users might expect from this kind of workflow – if items are created via spreadsheet, creating pages the same way starts to make a lot of sense. 10

The third inspiration for Rondo came during the planning of King Library's Digital Humanities Center. This center is a collaboration between SJSU's College of Humanities and the Arts, SJSU's University Library, and the San José Public Library. In imagining how we might support projects from community groups and individuals that are users of the San José Public Library, the need for tools that addressed a wider variety of constraints became apparent. At this point we had moved the SJSU Library's work on digital exhibits to an Omeka S installation that allowed us to involve more library staff and faculty, and to support student-led exhibits inside and outside the classroom. But in planning for work with outside groups, we anticipated a need to create projects that were not tied to university IT infrastructure, and could be maintained long term by smaller organizations [King Library's Digital Humanities Center 2024]. 11

These then were the three primary constraints that informed the technical decisions behind Rondo: 12

- The architecture supports site creators and developers who cannot install specific software
- As much as possible, the entire site can be created via spreadsheet
- The resulting site is not tied to a specific hosting environment – like static sites, it can easily be moved from one server to another

These requirements informed the choice to design Rondo as a single page application, and influenced many of the smaller decisions that would arise in the development process. 13

Single Page Applications

The single page application (SPA) has proved to be a durable architecture for web services, used in a wide variety of contexts. In the single page model, an entire site or service is delivered from one HTML document. This single page retrieves and displays content based on the context and the user's actions, typically via JavaScript. Familiar single page applications include Google's Gmail, Maps, and Drive services, along with Facebook, Twitter, and Netflix [W3 Lab 2022]. 14

Single page applications are not inherently minimal, but they are also not inherently "maximal" or resource-intensive. In general, the resources that SPAs require can be divided between resources required within the browser, or on the user's side, and resources required to manage and provide the data, or on the server side. On the browser side, SPAs generally require the user to have JavaScript enabled in their web browser. Beyond this basic requirement, individual applications may demand more or less processing power from the user's device – it is equally possible to design single page applications that require very little processing power as it is to design applications that are so demanding that they lag or perform poorly on older or less powerful devices [Gavrilă et al. 2019] [Mukhiya and Hung 2018] [Scott 2015]. 15

Similarly, the server side of an SPA may demand either a high or low level of resources, as well as a high or low level of technical maintenance and expertise. Many SPAs retrieve their data from one or more APIs, and the systems that provide these interfaces can be complex or simple. 16

The server side of an SPA can consist of any configuration, as long as that configuration can provide the data that the browser requests. In Rondo, that data can either be provided by the Google Visualization API [Google n.d.] or by a small set of local data files. The Visualization API is used because, unlike other options for retrieving data from Google Sheets, it does not require any API keys or special configuration when working with public spreadsheets. 17

When using local data files, the suggested workflow is to first build the site using a connected Google Sheet, and then

to separate the site from Google Sheets using Rondo's built in localization tools. This results in a static single page application which retrieves its data from three locally-stored files. These files store the site data as JSON. This arrangement aims to bring some of the advantages of static site generators like Jekyll to the single page architecture. A localized Rondo site has no database connection and can run locally on any simple web server, or on GitHub Pages. The localized site no longer depends on Google Sheets or any external API. Not having to rely on a database or back-end is one thing that makes static sites so attractive from a maintenance and preservation perspective, and a static single page application shares this advantage. However, this approach adds challenges for sites that require regular updates. These trade-offs and possible approaches to address them will be discussed in more detail below. 18

On the browser side, Rondo is designed to provide the basic functionality required for digital exhibits while placing relatively minimal demands on the browser. The single page architecture tends to require a bit more from the browser as the page loads, with the trade-off that subsequent interactions such as searches or page changes require few resources. For sites with a very large number of items, there may be a noticeable delay the first time a user loads the site, but once the site is loaded it should respond quickly to navigation and new requests. 19

Rondo's browser interactions and functionality rely on a small set of open-source resources. Rondo uses the minimal CSS framework Pico [Larroche n.d.]. It relies on the ubiquitous and well-established JavaScript library jQuery [OpenJS Foundation n.d.], and its search functionality is provided by the jQuery plugin DataTables [SpryMedia n.d.]. Another JavaScript library, Markdown it! [markdown-it 2024] enables the use of Markdown when building pages in Rondo. 20

Rondo itself is intentionally very small. The JavaScript code is currently fewer than 700 lines, and the local CSS is fewer than 200. This is just enough to provide the basic functionality of a digital exhibit – to organize and search collection materials, to build pages and curate items thematically, and to include items within pages. The next stage in developing Rondo will most likely be determining the best ways to add functionality such as mapping, timelines, and other data visualizations while retaining both ease of use and a small footprint. 21

Constraints We Take for Granted

By encouraging us to think about computing under constraints, the impetus to minimal computing can also help to highlight those constraints that we have come to take for granted. This can especially occur when practices adopted under constraints become second nature - when over time our work-arounds become our workflows. 22

As an example, in a recent conversation with a student employee who was working on a different project, the student asked me why the library would rely on jQuery and DataTables as the architecture for a project rather than choosing an application development framework like React or Next.js. I replied that the main reason we were working this way is that we needed the component to be embedded in multiple content management systems, rather than to function as its own standalone application. While it's possible to build that kind of widget in a contemporary framework like React, it isn't necessarily the kind of task those frameworks are built to accomplish. Because we are working in an educational environment where we already have to integrate a lot of different third-party tools and platforms, the less structured approach tends to work for us. 23

While not entirely satisfying, this was probably a good-enough response in that moment. The question did make me reflect, though, on just how specific and in some cases constrained the development environments that many of us work in are. I started using jQuery and DataTables as a solution to a variety of problems in a situation where our library had no real access to server resources outside of the university's content management system. We developed a journal website, a federated search for prospective students, and a handful of smaller widgets, all with the same basic framework, and all sitting on top of a standard webpage in the university's website. So this was how I learned to do a kind of web development in a quite resource-constrained environment, and it is a way of working that responds to a category of constraints that I think are present, in some form or other, in most educational environments, especially for staff and faculty who are not full-time web developers. 24

This student would probably have the same question about Rondo – why doesn't it use a framework? – and in this case it would be an even better question, and I would have to come up with a different self-justification, since Rondo is a 25

freestanding application and doesn't have to be compatible with a variety of content management systems. After all, frameworks like React and Angular were developed specifically to make it easier to build and develop single page applications like Rondo.

On the other hand, not using a framework may be exactly what makes Rondo a *minimal* single page application. The impacts of using or not using a framework are primarily or entirely on the development and maintenance process – the end user and site creator are not likely to know or care how the code was assembled. In development and maintenance, using a framework tends to create a higher barrier of entry to contribute or modify the code. To work in JavaScript, you need to know JavaScript. To work in React, you need to know JavaScript and React. In a community where many web developers are also educators or students, this lower barrier to entry can make a meaningful difference in who can use, maintain, and contribute to a project. 26

In practice, minimal computing may encourage us to assess our assumptions about what a web developer skill set looks like. Tools that are designed for full-time developers working within industry may actually be a poor fit for digital humanities practitioners, or even for web developers working in education more generally. Coding languages, development frameworks, and other development tools are designed with the needs of specific communities and work environments in mind. Their benefits, affordances, and shortcomings can be opaque to those who come to them from outside their assumed context. It is no coincidence that React and Angular were developed by Google and Meta for the use of their own employees – these are tools for web development at an industrial scale. 27

Some may respond that the web itself has evolved into a commercial or industrial environment, and it is hard to argue against that point of view. In that context, Varga's embrace of gleaning, of surviving on things that have been left behind, of subsisting rather than competing, is even more of a model. 28

But Is This Really Minimal Computing?

Rondo's most glaring departure from established minimal computing practices is its dependence on Google Sheets. While it is possible to disconnect a Rondo site from Google's infrastructure, the initial site creation process assumes a user who is willing and able to use Google Sheets. 29

On one hand, a work-around for this constraint might be relatively straightforward. Since Rondo is configured to work with data in the JSON format output by Google's Visualization API, any workflow that converts between a more conventional CSV format and this JSON format would allow Rondo to work with other spreadsheet programs, though not with the same immediacy as users experience with Google Sheets. However, it may be more interesting to pause with this contradiction as an example of the persistent materiality of computing, a constraint that minimal computing foregrounds but does not remove. Servers must exist somewhere, and they are never actually free to buy, operate, or maintain. This may be a situation where adopting a gleaning approach frees us to take advantage of available infrastructure while strategically avoiding over-dependence on resources that we cannot own or control. 30

It is possible to imagine and even build a computing environment constructed entirely from software that is both open source and not controlled by the technology industry[2]. Such an environment is not likely, however, to be compatible with the additional constraints that are present in most academic environments, or other workplaces, such as restrictions that prevent end users from installing software. In the end, we all have to choose which parts, if any, of the corporate computing environment we wish to embrace and take advantage of. While static site generators like Wax are not tied to any particular environment, in practice they are often configured to take advantage of Microsoft's continued provision of free server space and Jekyll integration on GitHub Pages. 31

Here, adopting the gleaner's gaze might reveal a different side of the situation. It is certainly possible to develop Rondo so as to eliminate its reliance on Google Sheets. However, it is worth asking whether that would be a worthwhile use of scarce technical resources. Perhaps there is something not just acceptable, but positive about gleaning value - in this case a back end database for our simple digital exhibits - from the free offerings and other left-behind resources of large corporations. 32

In adopting this approach, Rondo draws inspiration from the Knight Lab's popular tools, in particular TimelineJS [Knight Lab n.d.], which similarly relies on Google Sheets as its data source. The interface for creating a new TimelineJS embed code or link also serves as the model for Rondo's Easy Builder feature, which is being developed to make the process of creating a first site in Rondo even simpler. 33

Rondo's reliance on Google Sheets for data storage can also be seen as a practice developed in response to common constraints found in a higher education or academic library environment. As discussed briefly above, many of the specific methods used in Rondo have precedents in projects I was part of while working for Boston College Law Library. That context provided a set of technical constraints that required creating some complex user interactions purely in the front end – our small team created many components that sat on a page of a content management system, but accessed data from Google Sheets or local JSON files. Like Rondo, most of these tools use the Datatables jQuery plugin to provide search and other interactions. Some of these projects have been stable in production for almost a decade, arguing for the stability and reliability of these approaches. These projects are not unusual in their use of these tools, but I mention them to demonstrate that the methods used in Rondo arise from the specific sets of constraints that I have encountered practicing digital humanities and web development more generally in an educational setting. Example projects still in production at Boston College Law School include Fusion Search, a federated search intended for prospective students [Boston College Law School n.d.] and the library's new books list [Boston College Law Library n.d.]. The basic structure of Rondo, and in particular the way items are handled, is drawn from these projects, and I want to make this progression explicit in order to suggest that methods developed under constraints in other domains may be fruitfully applied in a digital humanities context as well. Additional examples might include Andrew Lison's exploration of the Gemini protocol [Lison 2022] and the adoption of Tor by some public libraries [Lund and Beckstrom 2019]. Practices like zine making, developed under constraints in environments far outside higher education [Duncombe 1997] have also become widely accepted as pedagogical practices [Creasap 2014] [Scheper 2023]. 34

Lastly, the idea of building a simple website with Google Sheets is not as new or radical as it might sound. As two examples, [Davenport 2019] describes a process for building a site using the Google Sheets API, and [Bradley 2023] reports on a library-based project with a similar structure. 35

Small Acts of Curation

The Digital Public Library of America (DPLA) is a metadata aggregator which brings together digitized collections from United States libraries, archives, and other cultural heritage institutions. The DPLA harvests descriptive records of digital collections from state, regional, or subject hubs. These hubs in turn collect and aggregate descriptive metadata from individual institutions. In addition to serving metadata to DPLA, many hubs share their collections on their own websites [Sandy and Freeland 2016]. 36

Rondo's features include support for a workflow or recipe for building small, curated sites based partly or entirely on items selected from the DPLA. The DPLA allows users to search over 50 million items, and to assign individual items to personalized lists, a feature DPLA calls My Lists. 37

Interestingly, this My Lists function is implemented without requiring users to log in or provide personal information. Instead, it relies only on cookies, and so is tied to a specific browser on a specific device. This is a much more minimal approach to site personalization than the one pursued by most projects, especially larger projects like DPLA. Importantly, this approach is fully functional from the user perspective, but it does not provide the site owner with the same opportunities to gather commercially valuable data about users that a more conventional approach that requires users to log in would create. 38

Users of this feature can export a CSV file with basic metadata for each item on a list, including a link to a thumbnail image. Rondo's spreadsheet is arranged so that users can copy and paste this metadata directly into the spreadsheet and begin working with a set of items from DPLA, complete with thumbnails and basic descriptions.[3] 39

This workflow is intended to suggest a wide variety of possible classroom or workshop assignments, inviting students to engage with, organize, and write about the primary materials in DPLA. Outside the classroom this recipe suggests the 40

possibility for digital humanities projects to be small acts of curation or criticism.

Placing value on small projects is a different kind of intervention than the one proposed through minimal computing, though the two ideas can complement one another. Minimal computing is strongly associated with the adoption or acceptance of constraints [Risam and Gil 2022]. This framing is valuable, but it is also suggestive. The foregrounding of constraints suggests that some person, group, or project is being constrained – that the creator wants to go farther and do more, but is held back in some way. The language of minimal computing can suggest grand scholarly and curatorial ambitions which have to be reined in or redirected. And what is it that constrains these ambitions? 41

I think it is fair to say that, collectively, the ambitions of digital humanities as a field of activity are heavily constrained by a lack of financial resources. Compounding this challenge, funders have historically favored projects which focus on White men over women or people of color, increasing the gap between ambition and resources for many valuable projects [Martin III and Runyon 2016]. Even past their creation, projects require maintenance and preservation, an activity for which few sources of funding or labor currently exist [VandeCreek 2022]. If minimal computing attempts to address this imbalance, it is by intervening on the resources side of the equation. Practitioners who aspire to compute minimally have developed methods that are intended to support digital humanities projects in established forms for longer periods of time with fewer resources. 42

What would it mean to address the ambition side of this imbalance instead? I don't want to make an argument about what digital humanists should or shouldn't do, but I do want to suggest that there are possibilities in small-ness – in small projects, small forms, and small collections – that have not yet been explored. We have many DH projects that are large, comprehensive, and seemingly poised to grow indefinitely. Do we have enough projects that are small, concise, and bounded? Do we have enough projects whose ambition is interpretive, expressive, and individual, whose value rests on insight and criticism, and not on completeness or scale? Aislin O'Donnell suggests that we take "the scavengers and gleaners of Agnès Varda's films" [O'Donnell 2017, 20] as anti-heroic models, and that gleaning can be a valuable orientation in all aspects of our work, not only the those that are more technical. This question of scale is present in minimal computing as well. [Gil 2015] asks scholars, "What is enough? What's your finished stairway," raising the possibility of a recalibration with regards to ambition. We should be cautious of minimalisms that intervene only on the technical side, without also questioning the way we scope and define the goals of our projects. 43

In that spirit, Rondo's DPLA workflow is intended to inspire small projects, experimental projects, free-associative acts of curation. It is an invitation to select a few objects and write about them. There are things that need saying, insights and perspectives that need sharing. Our tools and the expressive forms we adopt for our projects should make all of that easier, not more difficult. 44

A Walk-Through of Rondo

Having unpacked some of the inspirations for Rondo, I want to offer a critical walk-through of Rondo's code from a technical point of view, focusing on the JavaScript that runs in the browser when a user loads a Rondo site. The purpose of this exercise is not primarily to explain how Rondo works, but instead to examine the ways in which its specific implementation does or does not correspond to the goals of minimal computing and ease of use for new site builders. The goal, then, is to get deeper into the question of what a minimal approach to single page application development might look like and to expose the challenges of putting theory into practice in this area. 45

Creating a Rondo Site

Rondo's primary JavaScript file is called rondo.js. Site creators can build a new Rondo site by forking the Rondo GitHub repository and editing the rondo.js file to point to their spreadsheet.[4] 46

```
//spreadsheet variables – edit these to point to your spreadsheet
var spreadsheetID = "1iwIx8sqiBTqbtAI7TX4vDGDVee3dPz0wa3MazBv2fzQ";
var siteSheet = '1569296108';
var pagesSheet = '28080804';
```

```
var itemsSheet = '0';
```

This step is likely to be one of the more uncomfortable or unfamiliar aspects of the site building process for many users, and it comes right at the beginning of their process[5]. Site creators must identify not only the overall ID of their Google Sheet, but the IDs of each individual sheet. However, as we will see later in this section, storing all four of these IDs allows us to use the Google Visualization API to access the spreadsheet. It would also be possible to design an application like Rondo to use a different Google API – the Google Sheets API – which would only require only the spreadsheet ID. That API would make it possible to retrieve the sheet IDs programmatically, but it would also require the user to set up an account with Google Cloud and use an API key to access the data. The Visualization API's greatest advantage is that it is free and doesn't require any key or account to access public spreadsheet data.

This inflection point is a good example of the challenges involved in designing a minimal application, because the choice of API to connect to the spreadsheet strongly impacts Rondo's ease of use at the very beginning of the site-building process. Both the Google Visualization API and the Google Sheets API create obstacles for new users. It is quite possible to imagine tools that would help a new user create a Rondo site in the proverbial one click, but those tools themselves would demand significant additional resources. Within the current constraints – a small application distributed via GitHub – the best we can do may be to design workflows that minimize these speed bumps for new users, and to document the processes in great detail in order to facilitate successful installation. 47

Loading Site Data

One of the first things that Rondo does once the HTML page loads is to check the URL for queries. Rondo's queries do one simple thing - they tell the application which page content to load. So on the demo site, the URL for the "Build Your Exhibit" page is https://sjsu-library.github.io/rondo/?page=build. The function that retrieves queries is: 48

```
//function to get queries from the URL and get the URL slug for the
requested page
function getQueries() {
   var queryString = window.location.search;//get queries from current URL
   if(queryString) {
        var urlParams = new URLSearchParams(queryString);//get queries as
search parameters
           openPage = urlParams.get('page').replace('%20',' ');//set the
openPage variable to the value from the current URL
   }
};
```

This stores the page's URL slug in the variable "openPage." At this point, Rondo has not yet loaded the data from the spreadsheet, but once that happens, it will refer to that variable to open the correct page.

This is an example of the steps involved in coding a single page application without a framework. Similar code handles tasks like navigating from one page to another, or ensuring that the back and forward buttons in the browser work in the way the user would expect. JavaScript frameworks typically have their own structures for accomplishing these tasks, but working without a framework reveals that, for a smaller application, these structures need not be overly complicated. 49

When Rondo retrieves the data, it looks to the variable "remote" to determine if it will access data from Google Sheets (remote = TRUE) or from local data (remote = FALSE). It retrieves the data in a specific order – first it retrieves the site configuration data, then the pages data, and finally the items data. 50

The Google Visualization API returns a datatable which is parsed into JSON. That JSON data is first added to the Rondo Tools section of the page, where it is available for use in the localization process or for data sharing. Next, the function "siteConfig()" runs to begin building the site based on the configuration data. The structure of loading the pages 51

and items data is very similar. After loading the pages data, the function “pages()” runs to build the content for all pages. After the pages, the items are loaded, and the function “itemsDataTable()” uses the Datatables jQuery plugin to create a searchable table of the items.

```
//Connect to Google Visualization API and query the Site sheet to retrieve
basic site configuration
var                    query                    =                    new
google.visualization.Query('https://docs.google.com/spreadsheets/d/'+spread
sheetID+'/gviz/tq?gid='+siteSheet'&headers=1');
query.send(function (response) {
        //error handling
        if (response.isError()) {
        console.log('Error in query: ' + response.getMessage() + ' ' +
response());
      Return;
        };
   var siteDT = response.getDataTable();//get datatable from response
   var siteJsonData = siteDT.toJSON();//convert datatable to JSON
   siteJsonData = JSON.parse(siteJsonData);//parse JSON
   siteConfig(siteJsonData);//configure site using data from spreadsheet
   //end of site configuration
```

Localization

When Rondo is localized, the process of loading the data is simpler, but the same functions are run to build the site from the local data. The localized code makes it easier to read the structure of this part of the code. JQuery is used to load the local JSON files (“$.getJSON”) and each subsequent request is nested within the callback function of the request before it. This ensures that the three functions are run sequentially, and don’t start running before the data is loaded. This same structure is used when running remotely. 52

```
//run Rondo page as static page using local data
$.getJSON("json/site.json", function(siteJsonData) {//get site data
   siteConfig(siteJsonData);//configure site using site data
   $.getJSON("json/pages.json", function(pagesJsonData) {//get pages data
      pages(pagesJsonData);//create pages
         $.getJSON("json/items.json", function(itemsJsonData) {//get items
data
         itemsDataTable(itemsJsonData);//build the items table
      });//end of items
   });//end of pages
});//end of configuration
```

The goal of the localization process is to recreate key benefits of a static site system within the single page application. The localized data allows the site to run without a connection to any external API. When localized, the site could be considered a static single page application. As with a static site, there is no back end or database, but only a set of local files stored in a way that the web browser can easily interpret. 53

It is important to note, however, that Rondo does not yet include dedicated tools for editing localized sites. Editing the JSON files directly is possible but relatively cumbersome; tools like JSON Editor Online [De Jong n.d.] may provide the best experience for simple edits. Given the current tools the localization feature is most appropriate for sites which need to be preserved in their current state. Developing better workflows for editing localized sites is a priority for improving 54

Rondo's functionality when disconnected from Google Sheets. One approach might be to develop a tool that converts data back and forth between CSV format and the JSON format that Rondo expects, enabling users to edit localized sites in any spreadsheet software they choose.

Another consideration is that the static single page application relies on the durability and stability of the JavaScript that tells the browser how to construct the site. The use of JavaScript in artifacts intended for long-term preservation is not new, particularly in domains like digital art [Fino Radin 2011]. Both additional research and the passage of time will probably be required in order to identify specific practices that will lead to stable and supportable JavaScript over the longer term. At this point, practices like the use of well-established libraries, the packaging of required libraries along with the project, and well-documented and commented code, may provide a starting point for developers working in this area. JavaScript is too ubiquitous to be written off as a tool or format that is not suitable for long-term preservation – if we can't preserve JavaScript, we can't preserve very much of the web at all. Additionally, web archiving technologies are capable of preserving sites that use JavaScript [Brunelle et al. 2016]. Importantly, the workflow for localization allows Rondo sites to outlive the Google Visualization API without requiring any changes to Rondo itself. 55

The question of whether to disconnect a Rondo site from Google Sheets is left up to the site owner. Under current limitations, disconnecting the site is primarily intended as a way to archive completed sites which no longer need updates, and as a way to ensure that Rondo sites can be preserved if Google Sheets or the Google Visualization API is no longer available. Even if we are able to create better tools, the workflow for updating localized sites is likely to be more cumbersome than for sites connected directly to the spreadsheet. On the other hand, localized sites will be more durable and have fewer fragile external dependencies. 56

In any web architecture there are likely to be some trade-offs between long-term stability on the one hand and flexibility and ease of use on the other. These trade-offs often exist for technical reasons, but they may have parallels in the social domain; imagine a well-resourced and stable research platform that places restrictions on the content or perspectives that can be expressed on its servers. One of the contributions of minimal computing has been to make these trade-offs legible to practitioners and site creators by foregrounding the specific costs and benefits of large-scale infrastructure. Alan Liu captures this tension in his description of infrastructure as "the social-cum-technological milieu that at once enables the fulfillment of human experience and enforces constraints on that experience" [Liu 2016]. It may not be possible to engineer or theorize our way past the contradictions inherent in using the infrastructure of the internet, with all that comes with it, for humanistic research. Giving researchers concrete choices, none of them necessarily better than the others, is one way to foreground the enabling and constraining push and pull of contemporary infrastructure. 57

Displaying Pages and Items

On a Rondo site, all pages are always present in the Document Object Model (DOM), but only the present page is displayed. This is a common method for providing JavaScript-based interactions like tabbed content or carousels, and it is common in single page application structures. The HTML for all of the pages is created as part of loading the site, and any conversion from Markdown to HTML occurs during that process. In Rondo this process is typically not particularly demanding of the local device's resources, but it does illustrate the trade-offs present in working with the SPA model. The initial load of the site can be slower, and is more demanding of resources. However, once the site is loaded, few network resources are required and performance is typically fast. 58

The items are displayed using Datatables, which provides both simple and advanced searching via the Search Builder extension. Individual items are displayed in what web developers refer to as a modal – a box that floats above the standard interface, like a benevolent pop-up ad. This approach favors simplicity while still providing the basic functionality needed for an exhibit. In this model, items do not have their own pages with specific URLs – that treatment is reserved for pages, which would typically include interpretive content. Rondo includes a simple method for embedding items on pages, and there too clicking on the item opens the item's modal. This is not an inherent limitation of the SPA structure – it would be just as feasible to handle the items as pages rather than pop-up modals – but rather a design decision in favor of simplicity and directing the user to focus on the material contained in a site's pages. All of the 59

information about each item is available in the modal, but after closing the modal, the user is returned directly to their place in the narrative or interpretive material represented in the pages.

Unlike the pages, the content of the modal is not stored on the page but built from data stored in the Datatable of items. This implementation takes advantage of the ability to store data in the Datatable that is not displayed – for each item on the table, only a few fields are displayed, but all are available for search and use in building the modal. When a new item is selected, the modal is emptied and then rebuilt from the new item's data. This model ensures that the full-size image for each item is only loaded when the user interacts with the item. This is a basic but key performance consideration, as loading all full-sized images when the site loads would likely cause slow loading for many users. 60

Styling

The modal interaction is provided by the Pico CSS library, as are all of the site's styling and basic structure. As its name suggests, Pico itself is a minimal tool, intended to provide for the needs of small to medium sites with a very small package of CSS. Pico relies heavily on semantic HTML, meaning that, for example, Pico interprets the HTML element <dialog> as a modal dialog window, without the need for any additional CSS classes. Pico's reliance on semantic HTML helps to keep Rondo's HTML quite straightforward, and users can customize their Rondo site using standard CSS. Pico's extensive use of CSS variables can help to simplify customization via CSS. 61

Within Rondo's spreadsheet template, a site configuration sheet allows users to define the title and header image of their site. This sheet also includes columns which set a small number of key CSS variables, giving users some control over the site's look and feel without coding. Even small decisions like how many CSS variables should be exposed via the spreadsheet reveal the tensions between simplicity and flexibility when designing tools for minimal computing. Pico has over 130 variables, and only five of those are currently implemented in the spreadsheet. Of those five, four control the accent colors used in the site and one assigns a single font to be used for all text. 62

The level of styling customization available from the spreadsheet is currently quite limited. It would be possible to add more variables to the spreadsheet in order to allow further customization without writing CSS, but adding those variables would also make the site configuration spreadsheet more complicated and possibly intimidating. Customization via CSS is also preferable from a performance point of view. This is an area where further usability testing with site creators might be beneficial in order to identify the most appropriate balance. 63

Discussion and Conclusion

Having described some of the specific technical choices that establish the functionality and constraints of the application, I want to return to the concept of gleaning as a framework for approaching minimal computing and digital humanities work more generally. Imaging the researcher or student as a gleaner, collecting small, forgotten bits of code, technology, and server space, left behind primarily by the companies that make up the tech industry suggests a way of moving beyond the valorization of resource-intensive projects, while also avoiding overly moralistic or theoretical interpretations of technical choices. The image of the gleaner breaks down the dichotomy between the world-beating grant winner and the ascetic minimalist. 64

Kate Ince's (2013) reading of Varda's films is helpful here in expanding the possible resonances and associations of gleaning. Ince regards Varda as a filmmaker of the "lived body and women's embodied experience" [Ince 2013, 607], as exemplified by her self-identification with the experience of gleaning. [Croft 2018] emphasizes that "according to Agnès Varda, the tilting of the body and eyeline down towards the ground is what most emphatically characterises the gleaner" and that this attention to the ground, to the place where left-behind things are to be found, constitutes "the gleaner's gaze." This gaze, for which both Ince and Croft describe as haptic or focused on texture, is capable of identifying and recontextualizing overlooked objects and possibilities. As an orientation to the world, gleaning is rooted in the experience of women and specifically in poor and working class women. As an image of our relationship to technology, this is a welcome departure from portrayals which valorize the ways that powerful men relate to technology. It may go without saying, but we need alternatives to the perception that the executives of technology companies always have the most progressive and forward-thinking relationships to technology. 65

Adopting the gleaner's gaze can provide a certain clarity in conceptualizing the relationship between digital humanities and the tech industry. Industry narratives shape our sense of what is new and innovative in technology, and the tech industry's habitual hype cycles are more and more part of our everyday world and culture. The tech industry shapes the tools and services that are available to researchers and students. In trending or hyped areas, free resources and fresh code abound, left behind by the flows of venture capital. In an economy where even large technology companies can take years to become profitable, and where providing free or subsidized access to computing resources is a viable strategy to attract interest from users and investors, opportunities for creative gleaning abound. As a very visible example, recent reports suggest that OpenAI does not expect to be a profitable business until 2029 [Isaac and Griffith 2024]. However, the strategy of "money burning" is one employed in both large and small technology companies [Rachmad_2022]. 66

The gleaning approach also creates space for productive ambivalence about our approaches and tools. We don't have to be enthusiastic about Google Sheets, or about Google as a corporation, in order to glean some value from what they provide us, or what our campus IT budgets pay them for. Gleaning is not aspirational – it is above all about seizing what is attainable. If technology companies are going to burn money with or without us, is it wrong to stand near the fire to keep warm? 67

As an approach, gleaning complements minimal computing, but helps to expand both the positionalities and the technical approaches that might fit under the minimal umbrella. Single page applications, with their reliance on browser-side computation using JavaScript, might seem diametrically opposed to the static site approaches that are closely associated with minimal computing. Rather than a collection of static HTML pages, the single page approach produces one dynamic page. Both results are minimal in one dimension, and maximal in another. Similarly, gleaning expands the images and metaphors that we might use to explore and develop minimal approaches. The gleaner is not exactly the "user" we imagine when designing technology, but more of a "re-user" or even "mis-user" of technology. 68

The metaphor or framework of digital gleaning may prove more useful with further development, and in particular theoretical investigation that places pressure on the ways that this form of gleaning differs from the long tradition of agricultural gleaning. There is admittedly something strange about comparing impeccably polished commercial products like Google Sheets or Github Pages to ears of wheat left behind after a harvest or groceries retrieved from a dumpster. One way of reconciling this dissonance may be to attend to the material aspect of the digital. In this framing, we are not gleaning "Google Sheets" as a piece of technological infrastructure. We are gleaning one Google Sheet, as if it were an object that could fall off a truck. Or more literally, we are gleaning a few megabytes of space in one of Google's many massive data centers. We can't hold our left-behind resource but, perhaps strangely, we can access it remotely, from any internet-connected device. From another angle, maybe what we glean from Google is more like discarded packaged food than agricultural produce: It comes to us with the shiny wrapper still on, but we may have to get our hands dirty to pick it out and find our uses for it. Wherever we land in this metaphor, it is important to remember how Varda's film expands the notion of gleaning beyond its traditional meaning to include artistic and creative reuses of discarded material, and searches for these other ways and contexts in which gleaning may emerge. As she states in one of the film's transitional scenes, "we continue to film people who spend time around trash cans, yet they all have different reasons, and it's a different experience for each" [Varda 2000]. 69

Returning to Rondo, the project attempts to put this image of the gleaner into practice. It takes advantage of free server space and resources from Google, via Google Sheets, and Microsoft, via GitHub Pages. Through its integration with DPLA, Rondo also tries to enable gleaning in the curatorial or interpretive aspects of digital projects, facilitating a workflow based on re-use, interpretation, juxtaposition, and contextualization of existing digital collections. 70

From a technical point of view, this project proposes that the single page application model, when implemented with the goals of simplicity and ease of use in mind, is a promising complement to static site models for minimal computing platforms. In particular, SPAs have the potential to be easier for new users to adopt because they do not require the site creator to install software or use the command line. The project is also an experiment in minimizing the ambition, and maximizing the simplicity, of the tool or platform itself. Rondo strives to provide just what is needed for a minimal digital exhibit, and to avoid cluttering its interface with non-essential features. 71

The success of minimal computing as an intervention need not be derailed by the well-documented challenges associated with using static site generators, particularly in a pedagogical environment [Brooks 2021] [Dombrowski 2022]. This project aims to suggest that the gleaning approach as exemplified by an SPA which appropriates Google Sheets for data storage, may be an alternate but complementary path towards a more minimal, less resource-intensive digital humanities practice.

Notes

[1] Code for Rondo is available at https://github.com/sjsu-library/rondo, and the demonstration and tutorial site for the project is available at https://sjsu-library.github.io/rondo/.

[2] Some of the most popular open-source frameworks were developed within technology companies. For instance, Android, Angular, and Kubernetes from Google; and React, PyTorch, and Llama from Meta.

[3] Other widely-used databases that offer similar functionality include Hathi Trust and WorldCat, and it would be reasonably straightforward using a spreadsheet to transform metadata exported from those systems into the format expected by Rondo. It might also be fruitful to explore possible integrations with open APIs like those offered by the Metropolitan Museum of Art and Europeana, among others. Working with these APIs would require a different workflow; one approach might be to create tools for harvesting metadata from these APIs into a Google Spreadsheet using Google Apps Scripts.

[4] The code in this walkthrough is simplified and formatted for easier readability. Variable names are in bold.

[5] In a workshop or classroom setting, this might be a step that instructors consider taking ahead of time, providing students with a new site already connected to a spreadsheet.

Works Cited

Blackmer Reyes and Szydlowski 2022 Blackmer Reyes, K and Szydlowski, N. (2022). *Betita Martínez: A Memorial Exhibit*. Available at https://sjsu-library.github.io/betita/. (Accessed: 14 October 2024).

Boston College Law Library n.d. Boston College Law Library (n.d.) *New Books*. Available at: https://www.bc.edu/bc-web/schools/law/sites/students/library/using/new-books.html (Accessed: 12 October 2024).

Boston College Law School n.d. Boston College Law School (n.d.) *Fusion Search*. Available at: https://www.bc.edu/bc-web/schools/law/academics-faculty/fusion-search.html (Accessed: 12 October 2024).

Bradley 2023 Bradley, J. (2023) “Using Scalable Vector Graphics (SVG) and Google Sheets to build a visual tool location web app”, *code4lib*, 58. Available at https://journal.code4lib.org/articles/17697.

Brooks 2021 Brooks, M. (2021) “Real talk on static sites and digital literacies”, *Mackenzie K. Brooks*. Available at: https://mackenziekbrooks.info/update/2021/04/14/real-talk-static-sites.html (Accessed: 12 October 2024).

Brunelle et al. 2016 Brunelle, J.F. et al. (2016) “The impact of JavaScript on archivability”, *International Journal on Digital Libraries*, 17(2), pp. 95–117. Available at:https://doi.org/10.1007/s00799-015-0140-8.

Creasap 2014 Creasap, K. (2014) “Zine-making as feminist pedagogy”, *Feminist Teacher*, 24(3), pp. 155–168. Available at: https://doi.org/10.5406/femteacher.24.3.0155

Croft 2018 Croft, J. (2018) “Gleaning and dreaming on Car Park Beach”, *Humanities* 7(2), 33. Available at: https://doi.org/10.3390/h7020033.

Davenport 2019 Davenport, M. [@calebmackdaven] (2019) “Integrating Google Sheets as a backend” [Medium] 20 March. Available at: https://medium.com/@calebmackdaven/integrating-google-sheets-as-a-backend-7fc6a214418d (Accessed: 12 October 2024).

De Jong n.d. De Jong, J. (n.d.) *JSON Editor Online*. Available at: https://jsoneditoronline.org/. (Accessed 23 July 2025).

Dombrowski 2022 Dombrowski, Q. (2022) “Minimizing computing maximizes labor”, *Digital Humanities Quarterly*, 16(2). Available at: https://dhq.digitalhumanities.org/vol/16/2/000594/000594.html

Duncombe 1997 Duncombe, S. (1997) *Notes from Underground: Zines and the Politics of Alternative Culture*. London: Verso.

Fino Radin 2011 Fino-Radin, B. (2011) “Digital preservation practices and the Rhizome ArtBase”, *Rhizome ArtBase.* Available at: https://media.rhizome.org/artbase/documents/Digital-Preservation-Practices-and-the-Rhizome-ArtBase.pdf (Accessed: 12 October 2024).

Gavrilă et al. 2019 Gavrilă, V. et al. (2019) “Modern single page application architecture: A case study”, *Studies in*

Informatics and Control, 28(2). Available at:https://doi.org/10.24846/v28i2y201911.

Gil 2015 Gil, A. (2015). “The user, the learner and the machines we make”, *Minimal Computing*. Available at:https://go-dh.github.io/mincomp/thoughts/2015/05/21/user-vs-learner/ (Accessed: 12 October 2024).

Google n.d. Google (n.d.) *Google Visualization API Reference | Charts | Google for Developers*. Available at:https://developers.google.com/chart/interactive/docs/reference (Accessed: 12 October 2024).

Ince 2013 Ince, K. (2013) “Feminist phenomenology and the film world of Agnès Varda”, *Hypatia*, 28(3), pp. 602–617. Available at: https://doi.org/10.1111/j.1527-2001.2012.01303.x.

Isaac and Griffith 2024 Isaac, M. and Griffith, E. (2024) “OpenAI is growing fast and burning through piles of money”, *The New York Times*, 27 September. Available at:https://www.nytimes.com/2024/09/27/technology/openai-chatgpt-investors-funding.html (Accessed: 12 October 2024).

King 2007 King, H. (2007) “Matter, time, and the digital: Varda’s *The Gleaners and I*”, *Quarterly Review of Film and Video*, 24(5), pp. 421–429. Available at:https://doi.org/10.1080/10509200500536322.

King Library’s Digital Humanities Center 2024 King Library’s Digital Humanities Center (2024). *Digital Humanities Center Community Conversations*. Available at: https://library.sjsu.edu/ld.php?content_id=80034869 (Accessed: 26 July 2025).

Knight Lab n.d. Knight Lab (n.d.) *TimelineJS*. Available at: https://timeline.knightlab.com/ (Accessed: 12 October 2024).

Larroche n.d. Larroche, L. (n.d.) *Pico CSS • Minimal CSS Framework for semantic HTML*. Available at:https://v2.picocss.com/ (Accessed: 12 October 2024).

Lison 2022 Lison, A. (2022) “Minimal computing”, *COUNTER-N,* 2022. Available at: http://dx.doi.org/10.18452/25607.

Liu 2016 Liu, A. (2016) “Drafts for against the cultural singularity (book in progress)” Available at: https://liu.english.ucsb.edu/drafts-for-against-the-cultural-singularity/ (Accessed: 26 July 2025).

Lund and Beckstrom 2019 Lund, B. and Beckstrom, M. (2019) “The integration of Tor into library services: An appeal to the core mission and values of libraries”, *Public Library Quarterly*, 40(1), pp. 60–76. Available at:https://doi.org/10.1080/01616846.2019.1696078.

markdown-it 2024 markdown-it (2024). *markdown-it*. Available at:https://github.com/markdown-it/markdown-it (Accessed: 12 October 2024).

Martin III and Runyon 2016 Martin III, J.D. and Runyon, C. (2016) “Digital humanities, digital hegemony: Exploring funding practices and unequal access in the digital humanities”, *ACM SIGCAS Computers and Society* 46(1). pp. 20-26. Available at https://doi.org/10.1145/2908216.2908219.

Mukhiya and Hung 2018 Mukhiya, S.K. and Hung, H.K. (2018) “An architectural style for single page scalable modern web application”, *International Journal of Recent Research Aspects* 5(4), pp. 6–13. Available at: https://www.ijrra.net/Vol5issue4/IJRRA-05-04-02.pdf

Nyrop and Gil n.d. Nyrop, M. and Gil, A. (n.d.) *Wax*. Available at:https://minicomp.github.io/wax/ (Accessed: 12 October 2024).

O'Donnell 2017 O’Donnell, A. (2017) “For the scavengers and the gleaners: An anti-heroic vision of education” . *Other Education*, 5(2). pp. 1738. Available at: https://mural.maynoothuniversity.ie/8603/.

OpenJS Foundation n.d. OpenJS Foundation. (n.d.) *jQuery*. Available at:https://jquery.com/ (Accessed: 12 October 2024).

Rachmad_2022 Rachmad, Y.E. (2022) “The influence and impact of the money burning strategy on the future of startups,” *Proceedings of the 1st Adpebi International Conference on Management, Education, Social Science, Economics and Technology* (AICMEST), Jakarta, July 26, 2022. Available at: https://adpebipublishing.com/index.php/AICMEST/article/view/180 (Accessed: 12 October 2024).

Risam and Gil 2022 Risam, R. and Gil, A. (2022) “Introduction: The questions of minimal computing”, *Digital Humanities Quarterly*, 16(2). Available at:https://dhq.digitalhumanities.org/vol/16/2/000646/000646.html.

Rycenga et al. 2023 Rycenga, J. et al. (2023) *The Unionist Unified: Connecticut's First Immediate Abolitionist Newspaper*. Available at https://sjsu-library.github.io/unionist/ (Accessed: 14 October 2024).

Sandy and Freeland 2016 Sandy, H.M. and Freeland, C. (2016) “The importance of interoperability: Lessons from the digital public library of America”, *The International Information & Library Review*, 48(1), pp. 45–50. Available

at:https://doi.org/10.1080/10572317.2016.1146041.

Scheper 2023 Scheper, J. (2023) “Zine pedagogies: Students as critical makers”, *The Radical Teacher*, (125), pp. 20–32. Available at: https://doi.org/10.5195/rt.2023.963.

Scott 2015 Scott Jr., E.A. (2015) *SPA Design and Architecture: Understanding Single-Page Web Applications*. Shelter Island, NY: Manning.

SpryMedia n.d. SpryMedia. (n.d.). *DataTables | JavaScript table library*. Available at: https://datatables.net/ (Accessed: 12 October 2024).

VandeCreek 2022 VandeCreek, D. (2022) “Where are they now? The 2020 status of early (1996–2003) online digital humanities projects and an analysis of institutional factors correlated to their survival”, *Preservation, Digital Technology & Culture*, 51(3), pp. 91–109. Available at: https://doi.org/10.1515/pdtc-2022-0011.

Varda 2000 *The Gleaners and I* (2000) Directed by Agnès Varda. Kanopy. Available at: https://www.kanopy.com/en/product/10913169 (Accessed: 12 October 2024).

Wilke and Williamson 2024 Wikle, O.M. and Williamson, E.P. (2024) “Exploring Static web in the digital humanities classroom: The learn-static initiative”, *IDEAH*, 4(2). Available at: https://doi.org/10.21428/f1f23564.f88a989c.

W3 Lab 2022 W3 Lab (2022) “The Popularity of Single-Page Applications in 2022”, *W3 Lab*. Available at: https://w3-lab.com/single-page-applications-2022/ (Accessed: 12 October 2024).

Recommendations

DHQ is testing out three new article recommendation methods! Please explore the links below to find articles that are related in different ways to the one you just read. We are interested in how these methods work for readers—if you would like to share feedback with us, please complete our short evaluation survey. You can also visit our documentation for these recommendation methods to learn more.

SPECTER Recommendations

Below are article recommendations generated by the SPECTER model:

1. Minimizing Computing Maximizes Labor, 2022, Quinn Dombrowski, Stanford University
2. Open, Equitable, and Minimal: Teaching Digital Scholarly Editing North and South, 2022, Raffaele Viglianti, Maryland Institute for Technology in the Humanities, University of Maryland, USA; Gimena del Rio Riande, National Council on Scientific and Technical Research, Argentina; Nidia Hernández, National Council on Scientific and Technical Research, Argentina; Romina De Léon, CONICET (National Council on Scientific and Technical Research, Argentina)
3. Organic and Locally Sourced: Growing a Digital Humanities Lab with an Eye Towards Sustainability, 2020, Rebekah Cummings, University of Utah; David S. Roh, University of Utah; Elizabeth Callaway, University of Utah
4. Minimal Computing from the Labor Perspective, 2022, Tiffany Chan, University of Victoria Libraries; Jentery Sayers, University of Victoria
5. The In/Visible, In/Audible Labor of Digitizing the Public Domain, 2019, Amelia Chesley, Northwestern State University of Louisiana

DHQ Keyword Recommendations

Below are article recommendations generated by DHQ Keywords:

1. Social Networks and Archival Context Project: A Case Study of Emerging Cyberinfrastructure, 2014, Tom J. Lynch, Computer Sciences Corporation
2. Archival Liveness: Designing with Collections Before and During Cataloguing and Digitization, 2015, Tom Schofield, Culture Lab, Newcastle University; David Kirk, Digital Interactions Group, Newcastle University; Telmo Amaral, Digital Interactions Group, Newcastle University; Marian Dörk, Potsdam University of Applied Sciences, Institute for Urban Futures; Mitchell Whitelaw, Faculty of Arts and Design, University of Canberra; Guy Schofield, Digital Interactions Group, Newcastle University; Thomas Ploetz, Digital Interactions Group, Newcastle University
3. Digital Humanities on Reserve: From Reading Room to Laboratory at Yale University Library, 2020, Catherine DeRose, Yale University; Peter Leonard, Yale University
4. From Tamagotchis to Pet Rocks: On Learning to Love Simplicity through the Endings Principles, 2023, Martin Holmes, University of Victoria Humanities Computing and Media Centre; Joey Takeda, Digital Humanities Innovation Lab, Simon Fraser University
5. Continuous Integration and Unit Testing of Digital Editions, 2018, Bridget Almas, The Alpheios Project, Ltd.; Thibault Clérice, Centre Jean-Mabillon (École des chartes) - PSL

TF-IDF Recommendations

Below are article recommendations generated by the TF-IDF Model:

1. Minimal Computing with Progressive Web Apps, 2022, Chris Diaz, Northwestern University
2. From Tamagotchis to Pet Rocks: On Learning to Love Simplicity through the Endings Principles, 2023, Martin Holmes, University of Victoria Humanities Computing and Media Centre; Joey Takeda, Digital Humanities Innovation Lab, Simon Fraser University
3. Healing the Gap: Digital Humanities Methods for the Virtual Reunification of Split Media and Paper Collections, 2021, Stephanie Sapienza, Maryland Institute for Technology in the Humanities; Eric Hoyt, University of Wisconsin-Madison; Matt St. John, University of Wisconsin-Madison; Ed Summers, Maryland Institute for Technology in the Humanities; JJ Bersch, University of Wisconsin-Madison
4. Open Arabic Periodical Editions: A Framework for Bootstrapped Scholarly Editions Outside the Global North, 2022, Till Grallert, Orient-Institut Beirut
5. Minimal Computing from the Labor Perspective, 2022, Tiffany Chan, University of Victoria Libraries; Jentery Sayers, University of Victoria