Renaissance and Reformation
Renaissance et Réforme

LeBlanc, Zoe, editor and technical lead; Sarah Melton, managing editor; Ian Milligan, ombudsperson; Anna-Maria Sichani, editor, technical team; Nabeel Sidiqui, editor; Alex Wermer-Colan, editor, technical team. Founded by William J. Turkel and Alan MacEachern. The Programming Historian. Other

Rachel White

Volume 44, Number 4, Fall 2021

URI: https://id.erudit.org/iderudit/1089365ar
DOI: https://doi.org/10.33137/rr.v44i4.38658

See table of contents

Publisher(s)

Iter Press

ISSN

0034-429X (print)
2293-7374 (digital)

Explore this journal

Cite this review

White, R. (2021). Review of [LeBlanc, Zoe, editor and technical lead; Sarah Melton, managing editor; Ian Milligan, ombudsperson; Anna-Maria Sichani, editor, technical team; Nabeel Sidiqui, editor; Alex Wermer-Colan, editor, technical team. Founded by William J. Turkel and Alan MacEachern. The Programming Historian. Other]. *Renaissance and Reformation / Renaissance et Réforme*, *44*(4), 254–257. https://doi.org/10.33137/rr.v44i4.38658

under which the item has been tagged. The levels of access that researchers can gain to digital versions of original documents through a facility of this kind promises to be a game changer for this field in the coming years, presuming that Ancestry.com LLC has no immediate desire to change its business model.

LAURIE JOHNSON
University of Southern Queensland
https://doi.org/10.33137/rr.v44i4.38657

LeBlanc, Zoe, editor and technical lead; Sarah Melton, managing editor; Ian Milligan, ombudsperson; Anna-Maria Sichani, editor, technical team; Nabeel Sidiqui, editor; Alex Wermer-Colan, editor, technical team. Founded by William J. Turkel and Alan MacEachern.
The Programming Historian. Other.
Accessed 19 July 2021.
programminghistorian.org.

As a novice digital humanist, it can be difficult to know where to begin gaining digital skills beyond job-related training. It can also be daunting to undertake hours of self-teaching and training in digital skills when even understanding the vocabulary and acronyms can seem like a major hurdle. For people trying to learn digital skills for humanities-based work, The Programming Historian offers a potential solution.

Founded in 2008 by Willam J. Turkel and Alan MacEachern, The Programming Historian now offers eighty-six lessons in English and "is now a proudly multi-lingual project" with lessons in Spanish, French, and Portuguese.[1] Since its foundation, The Programming Historian's editorial board has expanded and there is a strong emphasis on diversity and inclusion. Indeed, members of the editorial board have published articles on the community's active participation in diversity and inclusion. In 2015, the editorial board believed that the community was gender neutral, as "they had not consciously constructed gender barriers," but, realizing this was not the case, they sought to find out what they could do to improve via open discussion online and a survey.[2]

[1] Sichani, Baker, Llach, and Walsh, 3.

[2] Crymble, 49.

Simply knowing that the editorial board of this resource is reflective about and reactive to its own principles of diversity and inclusivity is encouraging for those wanting to participate in its community as learners or as teachers. Indeed, for those interested in contributing a lesson, the process is not like that of traditional journals in which a submission is simply accepted or rejected. Instead, the contributor will receive mentoring from the editors to make the essay/lesson "as clear and as useful as possible" ("Contribute"), making the process one of collaboration rather than gatekeeping.

Originally, Turkel and MacEachern focused on Python, which now comprises twenty of the eighty-six available lessons. The remaining lessons introduce a broad range of digital skills including using network analysis in R, using and analyzing Twitter data, georeferencing, and data mining. These lessons are described as "novice-friendly, peer-reviewed tutorials that help humanists learn a wide range of digital tools, techniques, and workflows to facilitate research and teaching" (programminghistorian.org/en/). Lessons are arranged in groups, both by phases of research ("Acquire," "Transform," "Analyse," "Present," and "Sustain") and by topic or program. Each lesson has been peer-reviewed and each has a difficulty rating; most lessons are low or medium difficulty. The lessons use open-source languages and programs where possible, which further improves accessibility by removing financial barriers. The website also offers the opportunity for feedback via GitHub or email, meaning that any issues with a lesson can be pointed out and rectified.

The lessons are written in straightforward language with clear aims. In the introduction, it is made clear whether that particular lesson relies on the student having completed another one, and hyperlinks are used throughout to take the user to previous lessons, sites for software, datasets, or further reading, which makes it easy to navigate. Lessons often include screenshots—which enables the user to check progress, particularly when learning individually. The lessons are set out in a simple way: users scroll down as they work through the lesson, and lessons are typically short enough to avoid the problems this format might create; the user is thus spared the confusion of clicking through pages while trying to remember a previous step. Moreover, the simple interface is unlikely to cause problems for users with poorer Internet quality. While links to sample datasets are included with each lesson, it would be possible for a user to repeat the lesson using their own data.

The lesson “The Sound of Data (a gentle introduction to sonification for historians)” by Shawn Graham (programminghistorian.org/en/lessons/sonification) provides a new way of thinking about the historical data we generate—by converting it into sound. The aim of Graham’s lesson is to “hear something of the past” rather than to see or visualize it. A short way into the lesson, steps are taken to change the data into an audible representation with a single voice. The lesson builds on this to demonstrate that data can be layered to include multiple voices that represent different strands of data; the steps of the lesson lead to a clear and, in this case, audible output. The learner produces something and is further challenged to think about how to process their own data in new and exciting ways; the lesson contains links to other projects to demonstrate the possibilities. Another lesson, “Basic Text Processing in R” by Taylor Arnold and Lauren Tilton (programminghistorian.org/en/lessons/basic-text-processing-in-r), is part of a series in which learners gain skills used in R. As with other lessons, the aims are stated at the start and each step is explained with clear instructions. At the completion of each step, learners know what should be on the screen, which is useful for checking for errors. The lesson is also adaptable; learners can replace the dataset provided with their own and follow the same steps to begin some simple stylometric analysis of that data. Not all lessons are practical tutorials, however. For example, James Baker’s lesson, “Preserving Your Research Data” (programminghistorian.org/en/lessons/preserving-your-research-data), is a useful discussion about the usability of project data and outputs, now and in the future. This lesson offers advice from the naming of files to the best formats in which to save work to ensure it can be used and understood for years to come—with clear explanations for each suggestion, links to different schemas that could be adapted, and a list of further reading. This format, which deliberately avoids prescriptive directions, allows fledgling digital humanists to make their own informed choices about preserving their data while acknowledging—with a humorous reference to some of Baker’s own files labelled without a clear system—that becoming adept at digital-based research is an iterative process, with mistakes to be made and learned from along the way.

The Programming Historian challenges its users to think about how they approach their own research data using the digital skills they’ve gained through the lessons. The range of lessons available continues to grow. Learners can work methodically through a set of lessons, such as the series on Python or R, or try individual lessons, such as working with Twitter data. Any lesson that requires prerequisite knowledge provides links to lessons that teach the relevant foundational skills, such as using the Command Line.

The lessons provided by The Programming Historian are suitable for both individual use and group teaching. For individuals looking to expand their data skills, the lessons are clearly written with links to suitable free software and practice datasets, as well as further reading. From a teaching perspective, The Programming Historian appears to be a reliable resource with lessons that could be taught to small and large groups. Having experienced The Programming Historian both in a group and in an effort to self-teach, I have found the guidance of a teacher invaluable—as small mistakes (such as in writing code) or the quirks of operating systems and software can be very difficult to find or troubleshoot without help.

As a "volunteer, community-driven resource," and with its conscious principles of diversity, inclusivity, and accessibility, The Programming Historian sets a bar to which other projects should aim, both in and beyond the digital humanities.

RACHEL WHITE
Durham University
https://doi.org/10.33137/rr.v44i4.38658

Works Cited

Crymble, Adam. 2016. "Identifying and Removing Gender Barriers in Open Learning Communities: *The Programming Historian*." *Blended Learning in Practice* (Autumn): 49–60.

Sichani, Anna-Maria, James Baker, Maria José Afanador Llach, and Brandon Walsh. 2019. "Diversity and Inclusion in Digital Scholarship and Pedagogy: The Case of *The Programming Historian*." *Insights* 32 (1):3.