Introduction

There's been a huge amount of interest among librarians in learning to code. Numerous library conferences—including ALA, LITA Forum, Texas Library Association, Access, Code4Lib, and the Ontario Library Association—have featured programming tutorials or hackathons in recent years. These short workshops are wonderful for exposing people to fundamental concepts and creating positive experiences around code, but students don't necessarily know what to do next. For the learning to stick, people need real-world projects to which they can apply their newfound skills, which raises the question: how do librarians use code in their everyday work? This issue of *Library Technology Reports* is an answer to that question.

For this report, I reached out to LITA-L, the Code4Lib and LibTechWomen mailing lists, and my own network to survey librarians on how they use code in their jobs. I particularly looked for people who are not primarily developers and who could share examples of short scripts (under a hundred or so lines) that they'd written. While there are some wonderful large-scale code projects in libraries, I wanted to write about small, useful scripts that don't require expert-level coding skill: projects that are within reach even for fairly new coders. If you've been looking for reasons to learn code, ways to apply developing coding skills, or concrete examples to help you justify professional development support, this report is for you. If you manage aspiring coders you want to support, this report is for you, too.

Who This Report Is For

This report has three main audiences:

- librarians who are considering learning how to code
- librarians who have done some introductory programming study and are looking for next steps
- managers of librarians who code

If you're considering learning how to code, this report will give you lots of ideas about what you can do with code, what might be most helpful to learn, and how you can make your case to management. You'll probably want to skip ahead first to chapter 6, which summarizes survey respondents' advice on learning how to code. Then read chapters 2–4 to see real-world things your peers have done with code. That will help you narrow down your options.

Unless you're aiming to be in a pure development role, it's not practical to learn all the software topics that might be of interest; by contrast, mastering the fundamentals of programming and learning how to make one language do a few useful things is an achievable goal that will give you powerful, flexible new ways to approach your everyday librarian work. The middle chapters describe programs written by librarians with a variety of job descriptions, in different institutions, addressing diverse use cases. Look for projects that would be helpful in the context of your work, and use those to guide what you need to learn.

If you've already learned programming fundamentals, you may also find chapter 6 helpful, since it includes intermediate learning resources as well as options for review. However, start with chapters 2–4. Writing your own version of one of the programs in these chapters will be an excellent way to practice your skills and find out what you need to learn next. Most of them are under a hundred lines; I explicitly asked for short programs so that they'll be manageable for new programmers. Many of them have source code online; these links are provided in each chapter. Modifying others' code for your own use case is also a great way to improve your skills. Indeed, many library coders get a great deal of mileage out of modifying others' code and rarely if ever write their own from scratch. Finally, chapter 5, which details the political challenges and benefits of library code, will be helpful to you as you start to deploy code at work. Many library coders find that navigating buy-in and advocating for code are every bit as integral to their work as the actual programming. Avoid reinventing the wheel; learn about the problems they've encountered and techniques for handling those problems. And look for inspiration in the ways that people have positively impacted themselves and their coworkers through code.

If you're a manager, whether you code or not, you may be looking for ways to support your technically inclined supervisees. Chapter 6 outlines the support that librarians have received at work for learning to code. Some survey respondents are managers as well as coders, and I also asked them what kinds of support they provide; this is covered in chapter 6 as well. Chapter 5, about the political impacts of code, is also relevant to you. Finally, if you're not a coder, you may be interested in reading chapters 2–4 simply to see what others have done with code in libraries and how similar work might benefit your own library. Of course, if you are a coder, you may be interested in reading these chapters to get ideas for your own projects.

What You'll Find Here

After the introduction, there are three main sections. Chapters 2–4 cover programs that librarians have written to get things done better in their libraries, organized by common use cases: data cleanup, import, and export; reporting; and patron-facing services (mostly, but not always, through the website). Chapter 5 covers the political and social impact of library code, which came up so often in survey responses that it demanded a chapter of its own. Chapter 6 covers resources and advice for learning to code.

Chapters 2–4 each follow the same internal structure. After an introduction outlining the general use case of the code in the chapter, there's an examples section with brief overviews of a half dozen or so scripts. Each overview notes the script's author, language, and purpose. Where available, links to source code are provided, and you're encouraged to consult that code alongside the text.

After the overviews, each of chapters 2–4 has a deep dive into one script. Think of this as a code-reading group we're doing together, aimed at novice programmers. I'll walk you through the scripts line by line, showing you how they're organized and what each part accomplishes. Each of these scripts is online, so please pull up the code in your web browser and follow along. Additionally, the deep dives offer commentary on best practices and suggestions for how to expand or modify each program in case you want concrete ideas for practicing your programming skills.

These deep dives intentionally address different use cases and focus on programs written in different languages (PHP, Python, and JavaScript, all of which are in widespread use in libraries). I want to maximize the chances that one of them is relevant to you, no matter what library role you're in or what programming languages you may speak. I'm also deliberately agnostic throughout this report about what language you should learn. All of them incorporate the same fundamental programming concepts, like variables, control flow, and functions; all of them can be used to tackle a wide variety of problems. While many programmers have strong feelings about what language is best, the best language *for you* depends on which languages have good tools for solving problems you care about; which ones your coworkers, friends, or local community are already using; which have good learning resources you can access readily; and which appeal to your own idiosyncratic sense of aesthetics. I feel strongly that learning to program can be intellectually stimulating, personally empowering, and professionally useful, and you can realize those benefits regardless of what programming language you start with.

A note on mechanics: like many books on programming, this report follows the convention that text written in `monospaced font` represents code. When I am directly quoting lines of code or words from programming languages, I'll use `Courier`. For the most part, however, I refer to code samples online rather than reproducing them in the report.

Finally, there are a lot of survey responses that couldn't fit in this report, and ultimately the web is a more natural home for code than a report. For more information—including both scripts as of this writing, and any changes they may have undergone since—consult the companion website.

Companion website
https://thatandromeda.github.io/ltr

Survey

I'm writing a Library Technology Report *for ALA TechSource on lightweight, useful programs librarians & archivists have written to get their jobs done better. You need not be a library developer to answer these questions—in fact, I want to hear from people with all sorts of library roles! You need not be an expert coder or have a perfect code sample available, either. If you wrote some code that got something done, no matter how hackish or how elegant, I want to hear from you.*

How much of your job is about coding? Do you have any formal code responsibilities, or is this simply a skill that you bring to your formal responsibilities? *

Have you had support from your employer in learning to code/spending your time on coding? *

If you're a manager, have you provided support to employees who want to learn to code? If so, what sort? If not, why not?

What would you recommend to someone who wanted to learn to write code?

For the following questions, please pick a short program (less than a hundred-ish lines) that you wrote to do some library/archives task. If you have more than one you'd like to talk about, feel free to have multiple answers per question (just make it clear which answer goes with which script).

What language was this code written in? *

What problem did this code solve? *

What was the impact of the code? (time saved, new/improved service to patrons, etc.) *

If the code is publicly available, please link it here.

What did you learn from implementing this code? *

Survey

In the box above are questions and explanatory text from the survey. Asterisks indicate a required question. Not shown are requests for name, contact information, and referrals.

Acknowledgments

Thank you to all of my survey respondents, who shared so generously of your time, insight, and code. You are the giants on whose shoulders I stand.

Hillel Arnold (Lead Digital Archivist, Rockefeller Archive Center); Jason Bengtson (Head of Library Computing and Information Systems, University of Oklahoma); John Blyberg (Assistant Director for Innovation and User Experience, Darien Library); Terry Brady (Applications Programmer Analyst, Georgetown University); Matthew S. Collins (Director of the Library and Associate Professor, Louisville Presbyterian Theological Seminary); Esme Cowles (Lead Product Developer, UC San Diego Library); Jeremy Darrington (Politics Librarian, Princeton University Library); Robin Camille Davis (Emerging Technologies & Distance Services Librarian, John Jay College of Criminal Justice [CUNY]); nina de jesus (Digital Projects Librarian, York University); Misty De Meo (Digital Collections Technician, Canadian Museum for Human Rights); Rachel Donahue (Digital Projects Librarian, Special Collections National Agricultural Library); Mike Drake (Deputy Director, Tulare County Library); Shaun Ellis (User Interface Developer, Princeton University Library); Genny Engel (Webmaster, Sonoma County Library); Chris Fitzpatrick (ArchivesSpace Developer, Lyrasis); Angela Galvan (Digital Reformatting Specialist and Head, Document Delivery, The Ohio State University Health Sciences Library); Mike Giarlo (Digital Library Architect, Penn State University); Ron Gilmour (Web Services Librarian, Ithaca College Library); Annie Glerum (Head of Complex Cataloging, Florida State University Libraries); Jason Griffey (Chief Technology Strategist, University of Tennessee at Chattanooga); Thomas Guignard (École polytechnique fédérale de Lausanne); Cindy Harper (Electronic Services and Serials Librarian, Virginia Theological Seminary); Michael Holt (Reference Librarian/Marketing Coordinator, Valdosta State University Odum Library); Ken Irwin (Reference Librarian, Wittenberg University); Deborah Kaplan (Digital Resources Archivist, Tufts University Digital Collections and Archives); Francis Kayiwa (Assistant Professor and Systems Librarian, Colgate University Libraries); Bohyun Kim (Associate Director for Library Applications and Knowledge Systems, University of Maryland, Baltimore); Sam Kome (Director, Collection Services and Scholarly Communication, Claremont Colleges Library); Tricia Lampron (Metadata Service, University of Illinois at Urbana-Champaign); Joel Marchesoni (Tech Support Analyst, Hunter Library at Western Carolina University); Joe Montibello (Library Systems Manager, Dartmouth College); Christine Moulen (Library Systems Manager, MIT); Chad Nelson (Developer, CollectionSpace); Jeremy Nelson (Metadata and Systems Librarian, Tutt Library, Colorado College); Joy Nelson (Director of Migrations, ByWater Solutions); Eric Phetteplace (Emerging Technologies Librarian, Chesapeake College); Dot Porter (Curator, Digital Research Services, University of Pennsylvania); Carrie Preston (Head of Web Services, Ohio University Libraries); Matthew Reidsma (Web Services Librarian, Grand Valley State University); Sibyl Schaefer (Assistant Director, Head of Digital Programs, Rockefeller Archive Center); Michael Schofield (Librarian of Web Services, Alvin Sherman Library, Research, and

Information Technology Center); Coral Sheldon-Hess (Web Services Librarian, University of Alaska Anchorage); Jason Simon (Technology & Serials Librarian, Fitchburg State University); Owen Stephens (Independent Consultant); Ruth Szpunar (Reference and Instruction Librarian, DePauw University); Scott Turnbull (Lead Software & Systems Engineer, University of Virginia); Esther Verreau (Web Developer, Skokie Public Library); Matt Weaver (Web Librarian, Westlake Porter Public Library); Evviva Weinraub Lajoie (Director, Emerging Technologies & Services, Oregon State University Libraries & Press); Josh Westgard (Graduate Assistant, Digital Programs and Initiatives, University of Maryland Libraries); Amy Wharton (Research & Emerging Technologies Librarian, University of Virginia School of Law Library); Erin White (Web Systems Librarian, Virginia Commonwealth University); and Stuart Yeates (Technical Specialist, Victoria University of Wellington).

I'd like to thank two people particularly for not only responding to the survey but also for giving me feedback on the rough draft: Sarah Simpkin (GIS and Geography Librarian, University of Ottawa) and Becky Yoose (Discovery and Integrated Systems Librarian/Assistant Professor, Grinnell College). Additionally, thank you to the following people for last-minute beta reading: Natalie DeJonghe, Jordan Hale, David Saunders, Padraic Stack, Anne Sticksel, and Deidre Winterhalter.

The listed institutions and titles were accurate when respondents answered the survey; some have since changed jobs. Any other errors are, of course, my own.