RESEARCH-ARTICLE

The Dilemma of Building Do-It-Yourself (DIY) Solutions For Workplace Accessibility

YOONHA CHA, University of California, Irvine, Irvine, CA, United States

VICTORIA JACKSON, University of California, Irvine, Irvine, CA, United States

KARINA KOHL, Federal University of Rio Grande do Sul, Porto Alegre, RS, Brazil

RAFAEL PRIKLADNICKI, Pontifical Catholic University of Rio Grande do Sul, Porto Alegre, RS, Brazil

ANDRÉ VAN DER HOEK, University of California, Irvine, Irvine, CA, United States

STACY MARIE BRANHAM, University of California, Irvine, Irvine, CA, United States

The Dilemma of Building Do-It-Yourself (DIY) Solutions For Workplace Accessibility

Yoonha Cha
Informatics
University of California, Irvine
Irvine, California, USA
yoonha.cha@uci.edu

Victoria Jackson
Informatics
University of California, Irvine
Irvine, California, USA
vfjackso@uci.edu

Karina Kohl
Institute of Informatics
UFRGS
Porto Alegre, RS, Brazil
karina.kohl@inf.ufrgs.br

Rafael Prikladnicki
School of Technology
PUC-RS
Porto Alegre, RS, Brazil
rafael.prikladnicki@pucrs.br

André van der Hoek
Informatics
University of California Irvine
Irvine, California, USA
andre@ics.uci.edu

Stacy Branham
Informatics
University of California, Irvine
Irvine, California, USA
sbranham@uci.edu

Abstract

Existing commercial and in-house software development tools are often inaccessible to blind and low vision software professionals (BLVSPs), hindering their participation and career growth at work. Building on existing research on Do-It-Yourself (DIY) assistive technologies and customized tools made by programmers, we shed light on the currently unexplored intersection of how DIY tools built and used by BLVSPs support accessible software development. Through semi-structured interviews with 30 BLVSPs, we found that such tools serve many different purposes and are driven by motivations such as desiring to maintain a professional image and a sense of dignity at work. These tools had significant impacts on workplace accessibility and revealed a need for a more centralized community for sharing tools, tips, and tricks. Based on our findings, we introduce the "Double Hacker Dilemma" and highlight a need for developing more effective peer and organizational platforms that support DIY tool sharing.

CCS Concepts

• **Human-centered computing** → **Empirical studies in HCI**; **Empirical studies in accessibility**; • **Software and its engineering** → **Programming teams**.

Keywords

software development, accessibility, hacking, Do-It-Yourself (DIY), workplace accessibility, blind and low vision

1 Introduction

Companies in the software development industry have increasingly focused on diversity, equity, and inclusion (DEI) with respect to their workforce. In line with the 154 Fortune 500 companies that published DEI data in 2024 [102], tech companies such as Google and Microsoft now include disability data as part of their DEI reporting, revealing that 6.7% and 8.8% of employees identify as having a disability, respectively [39, 66]. Actions have been taken to make the software industry more diverse and inclusive, some of these legally binding like the Americans with Disabilities Act (ADA) [76] and the Web Content Accessibility Guidelines (WCAG) [106] in the United States, the European Accessibility Act [27, 80] in European countries, and the Quotas Law for People with Disabilities in Brazil [28, 77].

For blind and low vision software professionals (BLVSPs) specifically, where there has been an increase in representation year after year [96], the workplace still poses barriers in the form of inaccessible, unusable tools and tasks [20, 21]. Many tools used for programming, such as integrated development environments (IDEs), have been found to have accessibility bugs [4, 5, 11, 12]. Other workplace activities such as meetings [20] and project management [35, 47, 78], are also riddled with accessibility challenges. As a result, BLVSPs may experience reduced career mobility [21], in part due to the additional access labor they must perform in the workplace. Often, this labor takes the form of finding and implementing applicable off-the-shelf technologies that may help [63], asking for sighted assistance [4, 20, 78], and leveraging "conversational nudges" [20]. However, these strategies are often suboptimal and may only partially mitigate the accessibility problems BLVSPs face at work [20].

The concept of Do-It-Yourself (DIY) is well known among both software professionals and people with disabilities (PWD), including blind and low vision (BLV) individuals. Software developers are known to create DIY tools in the workplace to save time, help others, or because there is no known solution to the problem they are facing [94]. At the same time, DIY-ing is a necessary practice of many PWD, who are considered the "original lifehackers" [52] that spend their lives cultivating intuitive creativity, to navigate a society that is not designed for them [52]. PWD tend to DIY their own

solutions for accessibility (DIY assistive technology), which widely ranges from using existing solutions for different purposes [51] and combining multiple existing solutions together [44], to building solutions from scratch [20, 50]. While existing research has investigated DIY solutions for accessibility [44, 51] and customized DIY tools developed by programmers to meet their needs [84, 94], the intersection — how BLVSPs develop and build DIY tools for accessibility at the workplace — has yet to be explored.

To address this gap, we ask the following research questions:

- RQ1. What DIY tools do BLVSPs build and use in the workplace, and why?
- RQ2. How do these DIY tools impact BLVSPs?

We conducted in-depth, semi-structured interviews with 30 BLVSPs who identified as being blind or having low vision, are working or have worked in a software development position either in a corporate setting or as a freelancer, and have at least one year of work experience. Thematic analysis [19] was utilized for data analysis, and we identified four main themes: (1) key insights about the DIY tools BLVSPs build and use, (2) motivations for building and using DIY tools in the workplace, (3) impacts of DIY tools on BLVSPs' work, and (4) impacts of DIY tools on the entire BLV community. Based on our findings, we discuss the "Double Hacker Dilemma" that BLVSPs face: BLVSPs' dual identities of being a life hacker and a software professional (hacker, in the positive sense of the word) creates the dilemma between waiting months or even longer for the company to better support their work situation or taking immediate action because they know how to, yet at the expense of their own additional access labor. Finally, we highlight the need for a community platform for BLVSPs to share DIY tools and knowledge, and offer actionable insights for companies and technology designers.

Through this research, we contribute the following:

- A list of exemplary DIY tools that BLVSPs have built and used in the workplace.
- An understanding of BLVSPs' motivations behind building DIY tools and the impact these tools have on their workplace, the BLVSP community, and, in some cases, even the greater BLV community.
- An articulation of the "Double Hacker Dilemma" that BLVSPs face when they recognize their situation can and should be better, but they either need to wait or do it themselves.
- A set of suggestions for companies to better support DIY tool creation and sharing internally, and for the broader BLVSP community to share their DIY innovations and knowledge more broadly.

2 Related Work

2.1 Workplace Accessibility of BLVSPs

Software development requires both social, collaborative work (e.g., meetings [65, 87, 99]) and technical work (e.g., programming [65]. Research has identified several challenges faced by BLV workers in the workplace, typically resulting in additional access labor [24]. The root cause of many of the challenges is inaccessible technologies and tools [24, 47, 63] and ableist attitudes from colleagues embedded in organizations processes [21]. Collectively, the challenges

that BLVSPs face in both programming and non-programming tasks hinder their full participation [20], which may hamper their career mobility [21].

Social, collaborative work introduces accessibility barriers for BLVSPs. For example, the unstructured content of "stickies" on popular digital whiteboards (e.g., Miro [68]) used for collaborative ideation adds difficulty for screen readers to parse content [25], thus BLVSPs have to perform workarounds such as sending ideas to sighted colleagues over chat for them to paste it onto the whiteboards. Project management tools including Jira are also inaccessible for BLVSPs [20, 35, 47]. Although software professionals spend significant time in meetings [65], meeting settings, including when online, create significant hurdles for BLVSPs, leading them to perform significant access labor and in some cases even to stop participation altogether [3, 18, 20].

Most studies focusing on BLVSPs have investigated the accessibility of technical, programming-related tasks [4, 11, 13, 32, 72]. Researchers have found that the visually-oriented nature of integrated development environments (IDEs) such as Visual Studio Code [67] hinder code comprehension [12], navigation [47], and debugging [5]. For example, glanceability of code [83] is hindered as screen readers linearly navigate line-by-line through the codebase [83]. Pair programming also poses accessibility challenges for BLVSPs [35, 47, 78], as AT is usually not installed on their sighted colleagues' machines [11, 82, 83]. In addition to integrated development environments, BLVSPs commonly use web-browsers and command line interfaces (CLIs) to accomplish tasks (e.g., information seeking, compiling and running code). While command line interfaces are essential for local development and accessing cloud-based services, they pose accessibility challenges for BLVSPs due to the unstructured outputs and poor navigability with a screen reader [89].

Researchers have proposed a variety of tools for accessible software development, including replacements for standard tools [31, 69], command line interface scripts [31], browser plugins [48], and integrated development environment plugins [11, 83]. Some examples include screen reader plugins that help BLVSPs navigate a codebase non-linearly [5], a custom application integrated with screen readers to replace the standard command line interface bundled with the Windows operating system [69], a command line interface script to aid debugging code [88], a screen reader friendly Chrome browser plugin that aids BLVSPs in building visual websites [48], and a Visual Studio Code plugin to improve collaborative programming between a BLVSP and a sighted professional [82]. It is unclear how many of these tools are used "in the wild", but their existence shows the demand for improved developer tooling for BLVSPs.

2.2 Maker Culture, DIY-ing, Hacking

The term "maker movement" was originally coined by Dougherty [30] to refer to people who engage with objects in ways that make them more than just consumers [30]. Maker culture is driven by a shared belief in the possibilities of creation and innovation through technology, with members as active participants in shaping technology's development and application [10, 58, 59, 64]. While some

view the maker movement as promoting innovation and democratizing technology, research suggests that making is often perceived as a personal lifestyle or leisure activity rather than a political or economic endeavor [26, 101, 107].

In 1950, the term *"hack"* was coined at the Massachusetts Institute of Technology to describe innovative, unconventional approaches to technical issues [86]. Subsequently, the application of "*hacks*" was extended to computing and everyday life [86]. Since then, the idea of "life hacking" has gradually evolved and, today, the term covers the intersection of technology, culture, and larger concerns about work, wealth, health, relationships, and meaning [86].

2.2.1 DIY in the Workplace. While studies in Human-Computer Interaction (HCI) and Computer-Supported Cooperative Work (CSCW) have increasingly focused on the convergence of DIY, hacking, and craft practices (e.g., [44, 101]), literature in this field about hacking in the workplace is sparse. A few articles in the organizational and management literature discuss workplace hacking as a form of "smart working" [62] and developing "shortcuts" [15] that goes beyond programming tasks. This entails discovering, inventing, and applying tricks for better productivity [53, 97] and well-being [15], including individual- and team-level hacking activities [94]. Bloom et al. [15] identify this as another form of labor — "working to work" — which inadvertently reinforces capitalist systems.

Although building tools is considered typical of expert software design work [81] and recognized as commonly being undertaken by software engineers [94], workplace hacking in the form of DIY-ing tools has rarely been documented in existing literature, with two notable exceptions [22, 94]. In a study on the development and use of DIY tools within a company [94], various motivators for software engineers to build DIY tools included desires for efficiency and automation, and because there was no known solution. Some of these tools may be developed clandestinely without the knowledge or approval of a developer's management, and are therefore not tracked in official bug trackers nor stored in official source repositories [94]. Other times, the tools are shared across the organization. For example, the tool CodeFlow began as a DIY tool to improve code reviews but is now widely adopted in the organization [22].

Outside of the workplace, the open-source software movement [17] provides a platform for developers to DIY tools. Open-source software was initially an opportunity for developers to "scratch a personal itch" by creating their own projects [85]. An early study identified that key motivators for contributing to open-source included the intellectual stimulation of writing code, and that the code was needed either for work or non-work purposes [56]. Moreover, own-use value was categorized as a form of internalized extrinsic motivator in a 2010 literature review that identified ten categories of motivators [105]. Altruism was also considered a motivator. Investigation into motivation continued with a more recent study [37] identifying that motivating factors had changed since the early days of open-source software, with the importance of own-use declining.

2.2.2 DIY Assistive Technology. While specialized assistive technologies (ATs) exist to address different challenges BLV individuals face in life, they often provide generic solutions that do not consider individual differences among users [43]. Consequently, BLV individuals have become experts at customizing and "hacking" ATs to

suit their unique needs, instead of attempting – often pointlessly so – to convince corporations of the needed features and their market viability [6, 44].

Do-It-Yourself assistive technology (DIY-AT) refers to the development and adaptation of AT devices by non-professionals [16, 46]. DIY-AT addresses issues of abandonment and low acceptance rates of commercial ATs by considering user opinions, improving device performance, and adapting to changing needs [51, 64]. There are different motivations to creating DIY-AT, including increased control over design elements, passion, and cost-effectiveness. The emergence of rapid prototyping tools and online communities has further empowered individuals to create custom AT solutions [51, 57, 74]. Makerspace culture, with its emphasis on bespoke creativity, iterative design, and personal customization, is a promising match for the needs of disabled people, especially for the design of customized assistive technology [6] at low cost [46, 50, 51]. Open design in particular paves the way for the development of customized, affordable ATs [42].

Some screen readers (e.g., Jaws [91] and NVDA [1]) have purposefully been designed to allow the use of plug-ins. Such plug-ins help to make applications more accessible or enhance the capabilities of the screen reader [71]. A few of these plug-ins are geared to assisting BLVSPs in coding activities (e.g., [70]). While the plugins are beneficial, challenges remain for BLV(SP)s in identifying suitable plugins due to a lack of a centralized repository, concerns about the security of the plugins, and a lack of financial incentives to develop such plugins [70].

2.3 Sharing Knowledge and Tools

2.3.1 Sharing among Software Developers. To successfully engage communities and to share knowledge requires platforms. For example, Google at some point distributed a weekly, one-page printed newsletter to restrooms to increase awareness and adoption of internal tools [73]. Within open-source software, an active X (formerly Twitter) community is important as it attracts contributors, impacts popularity, facilitates sharing of resources, and provides a platform for engaging in technical discussions [33, 34, 92, 93]. Mailing lists and social Q&A communities are also important hubs for project discussions [41, 104]. Social Q&A sites in particular function as learning communities and have become valuable repositories of knowledge [2, 10, 41, 104].

2.3.2 Sharing among PWD. PWD also use online platforms such as Facebook and X to share knowledge, join communities, and provide social support to one another, especially to those with a similar disability [100]. Online communities also provide a venue for PWD to co-create knowledge with researchers and others, leading to product and service improvements that benefit PWD [8]. PWD and non-disabled colleagues are also known to use these communities to engage in joint efforts to create accessible environments, showing interdependent support [14]. Specifically for people within the BLV community with an interest in programming (either as a hobby or profession), the Program-L [36] mailing list provides practical assistance on using tools and approaches to aid BLV programmers [54, 79]. Additionally, an analysis of the use of X reveals an active accessibility community advocating for inclusion and discussing accessibility practices, challenges, and potential solutions [49].

2.4 Research Gap

While research has examined the development of DIY tools by software developers (primarily within the open-source software community) and also studied DIY-AT tools by PWD, to the best of our knowledge, this study is the first to explore the intersection of work hacking that is realized through DIY tools built by BLVSPs, especially for accessible work. Our study explores this unexplored area to reveal BLVSPs' experiences with and perspectives on developing DIY tools to tackle workplace inaccessibility.

3 Methods

This section describes the recruitment process, participants, interviews, and data analysis undertaken.

3.1 Participants

We recruited 30 people identifying as being blind or having low vision, who are working or have worked in a software development position (e.g., software engineer, tester, accessibility designer, product manager), either in a corporate setting or as a freelancer, and with at least one year of experience working in the role. Participants were recruited through professional contacts, mailing lists such as Program-L (an online discussion group catered to programmers with visual disabilities) [36], and snowball sampling. Most of our participants were located in the United States of America (18), with other participants in Europe (6), India (4), and Brazil (2).

To preserve the confidentiality of our participants, we removed identifiable information, including name, age, and job titles. Specific job titles were categorized into generic job categories since titles vary between organizations and thus may be identifiable. For example, Accessibility Specialists include a range of positions such as accessibility tester and consultant. In addition, we report ages in ranges and categorize the type of organization in which our participants most recently worked in the manner of Pandey et al. [78]. Participant ages ranged from 19 to 59, and years of work experience ranged from 2 to 40 years. We selectively omit participant IDs in places where disclosure may lead to participant deanonymization, and indicate with a footnote. Participants self-reported their vision status: 60% (n = 18) as being totally or completely blind with little or no light perception, and 40% (n = 12) a range of visual abilities. Refer to Table 1 for more detailed information on participants.

3.2 Procedure

To answer our research questions, we conducted audio-recorded, semi-structured interviews with 30 participants over Zoom between May 2024 and July 2024. The interviews ranged from 43 minutes to 182 minutes with an average of 73 minutes. Prior to the interviews, we emailed participants a study information sheet for review and acquired verbal consent at the beginning of the interview. Participants were also asked to complete a pre-interview survey to collect demographic information. Questions during the interview covered topics such as current accessibility problems participants faced at work and off-the-shelf workarounds they used, followed by questions about tools participants built and their experiences sharing the tools with others. Participants were compensated at a rate of $40 per hour via either an Amazon gift card or PayPal. This study was approved by the researchers' Institutional Review Board (IRB).

3.3 Data Analysis

All 30 audio-recorded interviews were transcribed and checked for quality and accuracy by the researchers. 28 interviews were conducted in English and two in Portuguese (as preferred by these participants) by an author who is a native Portuguese speaker and proficient English speaker. This author translated these two transcripts into English. Thematic analysis [19] was used for data analysis. The first three authors first analyzed the same seven randomly selected transcripts, performed open coding, and met frequently to identify and refine codes, discuss and refine themes constructed by the researchers, and reach consensus over time. This led to a codebook that the first author more formally documented, which included codes such as "altruistic motivations" and "maintaining professional image." After developing the codebook, the first seven transcripts were re-analyzed by the first author, and the remaining 23 were coded individually by two authors, who met frequently to discuss findings, newly identified codes, and possible new or refined themes. When researchers constructed new themes, the authors re-analyzed prior interviews. Throughout the data analysis process, the entire research team met periodically, which led to a refining a few themes and identifying a few new themes. After the first external review of the paper, the authors revisited the themes and combined some lower-level themes for more coherency. The headings and subheadings of the Findings section map to the higher-level themes and detailed themes resulting from our analysis (see Table 2), with the exception of subsection 4.1 where the types of tools used were explicitly identified and categorized.

3.4 Researcher Subjectivity, Positionality, and Reflexivity

The research team consists of researchers in the fields of HCI and Software Engineering, with diverse backgrounds and varying demographics. Researchers are from four different continents, with four researchers being women and two being men. While none of the researchers are BLV themselves, two researchers are highly experienced in accessibility research and engaging with the BLV community; four researchers have extensive experience working in or with software teams in industry; and several researchers have experience in both fields. This diversity of the team enabled the development of the study protocol sensitive to workplace culture and social and technical knowledge. The varied positionalities affected analysis. For example, the accessibility researchers introduced asset-based interpretations [109] while the software engineering researchers understood the value of various technical tools in getting work done. To acknowledge and include these varied perspectives, at least three researchers were involved in each data analysis session.

4 Findings

In each of the below subsections, we address one of the following high-level themes identified in Table 2 above.

4.1 DIY Tools Built and Used

As BLVSPs faced accessibility and usability challenges with tools and processes provided by their employers, participants used numerous DIY tools to assist them in the workplace, both for technical and non-technical work. 67% (n = 20) of our participants explained

ID	Job Position	Exp. (yrs)	Org. Type	Org. Size	Self-Reported Visual Ability	Gender
P1	Software Engineer	6 to 10	IT	5,000 +	totally blind	M
P2	Software Engineer	1 to 5	IT	5,000 +	low vision	M
P3	Software Engineer	1 to 5	Aerospace	500 to 4,999	20/200, limited field vision	M
P4	Freelancer	1 to 5	N/A	1 to 9	light perception	M
P5	Software Engineer	1 to 5	Finance	5,000 +	light perception	M
P6	Software Engineer	1 to 5	IT	10 to 99	totally blind	NB*
P7	Accessibility Specialist	1 to 5	Finance	5,000 +	totally blind	M
P8	Software Engineer	16+	IT	10 to 99	poor orientation	M
P9	Technical Lead	6 to 10	IT	5,000 +	legally blind	M
P10	Software Engineer	11 to 15	Outsourcing	100 to 499	totally blind	M
P11	Accessibility Specialist	11 to 15	Non-profit	100 to 499	legally blind	M
P12	Software Engineer	6 to 10	IT	5,000 +	totally blind	M
P13	Freelancer	1 to 5	N/A	500 to 4,999	completely blind	M
P14	Freelancer	6 to 10	N/A	1 to 9	light / color perception	W
P15	Software Engineer	6 to 10	IT	10 to 99	totally blind	M
P16	Software Engineer	1 to 5	IT	100 to 499	2% useful vision	M
P17	Accessibility Specialist	16+	Education	5,000 +	totally blind	M
P18	Software Engineer	1 to 5	IT	5,000 +	low vision	W
P19	Software Engineer	6 to 10	IT	5,000 +	legally blind w/ tunnel vision	M
P20	Software Engineer	1 to 5	Education	500 to 4,999	totally blind	M
P21	Data Scientist	1 to 5	Cosmetics	5,000 +	totally blind	M
P22	Software Engineer	16+	IT	5,000 +	low vision	M
P23	Technical Executive	1 to 5	Non-profit	1 to 9	low vision	W
P24	Software Engineer	6 to 10	Media	5,000 +	totally blind	M
P25	Technical Executive	16+	Finance	5,000 +	blind / low vision	M
P26	Software Engineer	6 to 10	IT	5,000 +	totally blind	M
P27	Accessibility Specialist	6 to 10	IT	5,000 +	light / color perception	M
P28	Software Engineer	1 to 5	IT	5,000 +	totally blind	M
P29	Software Architect	16+	IT	5,000 +	20/80 corrected	M
P30	Software Engineer	16+	Finance	100 to 499	light perception	M

Table 1: Detailed information of participants. For participant anonymity, all participant names were replaced with IDs, age is reported in ranges, and job titles do not include specific position information. "Exp." stands for years of professional experience working in software development. "Org." stands for organization, and the unit of "org. size" column is the number of people. *: NB stands for Non-Binary.

High-Level Themes	Themes
Motivations for DIY-ing	Inaccessibility and Indignity of Existing Tools
	Intrinsic and Extrinsic Joys of Hacking
	Efficiency and Autonomy in Relation to Colleagues
Impact of DIY Tools on Work	Increased Accessibility, Confidence, and Equity
	Spending Extra Time Building DIY Tools
	Organizational (Dis)Approval
Impact of DIY Tools on the BLV Community	Sharing DIY Tools
	Collaborating, Advocating, and Offloading
	Uptake of DIY Tools
	Need for Internal Community
	Need for Centralized External Community

Table 2: The headings and subheadings of the Findings section map to the higher-level themes and detailed themes of our analysis.

either building or contributing to DIY tools, while others used existing DIY tools. Tables 3 and 4 summarize the 55 different DIY tools that our 30 BLVSPs mentioned, 37 (67.2%) of which were built by our participants and 18 (32.8%) were adopted from elsewhere. Adoption rates of the home-built tools varied, with most only used by the author (e.g., "AI Content Describer"), some being used by multiple of our participants (e.g., "Log Output Accessibilizer"), and others in widespread use in the community (e.g., "Can You See Me").

4.1.1 Purposes of DIY Tools. We categorized the DIY tools at the workplace as follows:

- **Text/Image Interpretation Support (T & I):** These tools helped BLVSPs interpret inaccessible information. Some helped describe images (e.g., "AI Content Describer", "VOCR", "Monitor and Read"), others helped to better present code through screen readers (e.g., "IndentNav", "Cleaner for Swagger"), yet others were built to solve inaccessibility in internal corporate tools (e.g., "Statistics Retrieval").
- **Collaboration Support (CS):** These tools were designed to aid collaborative tasks such as meetings and presentations (e.g., "Can You See Me", "Volume Docker", "PPTX2MD").
- **Development Support (DS):** These tools were designed to make specific software development tasks more accessible. "SQLite Preview Navigator", "SQL Manager Lite", and "MongoDB Navigator", for instance, support easier access to specific databases. As two other examples, "Log Filtering Tool" and "NVDA Auto-read" identify changes in files and monitor log files in an accessible manner, respectively.
- **Productivity Support (PS):** These tools boosted BLVSPs' productivity by making keyboard usage more efficient with the screen reader (e.g., "Golden Cursor").
- **Screen Reader Optimization Support (SRO):** These tools facilitated better interaction of participants with their screen readers. For example, "AccessiNVDA" provides accurate pronunciation of Greek symbols that used to be incorrectly pronounced.

Notably, only a handful of the DIY tools were company specific, for instance to access otherwise inaccessible internal company data (e.g., "Dashboard Accessibilizer"). The vast majority, however, were well-suited for any workplace environment. A subset of these tools, especially several related to text/image interpretation support and collaboration support, are helpful beyond BLVSPs to BLV information workers more generally. Finally, many of the DIY tools were built and used to assist BLVSPs in programming, reinforcing the essential role DIY tools have in supporting accessible software development.

Some DIY tools started off as a solution to address an issue in a participant's personal life but later became essential for work. P7 was unable to read memes sent by their friends, so they developed a tool which they realized could actually help them in the workplace: *"So after I wrote it... it spiraled into... a transformative type of technology that we really, really, desperately need in not just my industry, but any blind or visually impaired individual who uses the computer needs something like this."* Companies thus benefit from developers' willingness to tinker in their spare time.

Interestingly, in compiling tables 3 and 4, we found duplicate efforts put into some of the DIY tools built by our participants. "Dashboard Accessibilizer", "Log Output Accessibilizer" , and "CLI Pathname Simplifier" were each built by different participants, yet effectively performed the same function, sometimes even for the identical software. It appears that our participants were either unaware of the existence of similar tools or decided to build their own regardless.

4.1.2 Types of Tools Built/Used. BLVSPs were flexible in the types of DIY tools they built or adopted, as the technology—ranging anywhere from small scripts to standalone, interactive applications—*"depends on the task"* (P12).

Among the 55 DIY tools built or used by BLVSPs, the most common type of tool was scripts (n = 20) followed by NVDA add-ons (n = 14), and applications (n = 13).

Many participants described the development of DIY CLI scripts (n = 20). P16 noted having developed as many as 132 scripts (only some of which we touched upon in the interview; those are reflected in Tables 3 and 4). The complexity of scripts ranged widely, from just simplifying pathnames to leveraging available Application Programming Interfaces (APIs) to build the textual equivalent of otherwise graphical standard user interfaces (UIs). P12 built a script that enabled them to access information previously inaccessible on a corporate dashboard: *"Anything you can do using a visual UI, you could essentially do using a bunch of APIs as well. For me it was basically making those thread calls to specific actions, whether a 'get' or 'post', and getting kind of those things done."*

For the next most commonly mentioned tools, NVDA add-ons, participants appreciated that NVDA allows writing add-ons: *"The fact that NVDA has add-ons, and the fact that there are people with skills who have shared them [DIY tools], it's pretty nice"* (P15). The flexible architectures of browsers were equally leveraged by some BLVSPs to develop plugins to make platforms (e.g., websites) accessible, which many participants reported frequently doing so: *"Usually the problem with these... is the the buttons are not labeled... So, I have them labeled through JavaScript and stuff. I do that, a lot."*

Less frequently developed were screen readers (n = 3), libraries to be incorporated into applications (n = 2), and an accessible version of the Linux Distribution ("TalkingArch"). The reduced number perhaps reflects the complexity of these more full-fledged endeavors in comparison to the typically easier effort of developing plug-ins and scripts, as P24 commented: *"It's worth drawing a distinction between tools and scripts, right? So when I say a 'script,' I mean something that isn't as fully featured."*

4.2 Motivations for DIY-ing

Our participants had a wide range of motivations for building and using DIY tools at the workplace.

4.2.1 Inaccessibility and Indignity of Existing Tools. BLVSPs were motivated to build and use DIY tools because companies failed to provide adequate accommodations, and suboptimal workarounds were indignifying. Taking matters into their own hands was an alternative to submitting tickets for accessibility troubleshooting and waiting indefinitely. P7 expressed bitterness about the *"sad state"* of accessibility request turnaround times. Although he reached out to

Type	Purpose	Tool Name	Description
Add-On	T & I	AI Content Describer	Descriptions for images and UI controls, leveraging multiple GenAI models
Add-On	DS	NVDA Auto-read	Detects new text in the terminal window to notify of changes
Add-On	PS	Text Information	Provides screen reader in-line definition of selected text
Add-On	SRO	AccessiNVDA	Provides correct pronunciation of Greek symbols
Add-On	SRO	Speech History	Reviews the last 500 strings of speech synthesizer output
Add-On	PS	Remote Support	Controls a computer running NVDA from another computer running NVDA
Add-On	T & I	IndentNav	Enables code navigation between different indentation levels
Add-On	SRO	CrashReporter	Attempts recoveries when NVDA crashes, preserving context
Add-On	T & I	Audio Screen	Enables aural interpretation of images via touch on Windows 8+ screens
Add-On	T & I	NVDA OCR	Optical Character Recognition (OCR) to extract text from inaccessible objects
Add-On	SRO	Phonetic Punctuation	Converts punctuation and regular expression signs into audio sounds
Add-On	DS	FileZILLA	Labels buttons, adds keyboard shortcuts to FTP (File Transfer Protocol) apps
Add-On	PS	Golden Cursor	Moves mouses with key presses and saves mouse positions for applications
Add-On	T & I	SVG to PNG	Converts images in SVG format to PNG format to allow NVDA access
Script	DS	InclusiAI	Transparent window reading inaccessible terminal outputs for screen reader
Script	DS	MongoDB Navigator	Allows easier navigation and editing of MongoDB data on notepad
Script	DS	Visual Studio Autocomplete	Allows autocomplete pop-ups on Visual Studio with screen readers
Script	DS	CLI Pathname Simplifier	Reads out only the last backslashed path from an entire path in the CLI
Script	T & I	JSON Viewer	CLI to read websites, transform to JSON, and see content in a tree view
Script	DS	GitHub Project Manager	Clones Github repositories and enables operations with fewer key presses
Script	PS	Chromium Operations	Launches website and performs operations using a headless browser
Script	DS	Code Review System	CLI that identifies code changes in the code review system
Script	T & I	Cleaner for Swagger	Cleans C# code generated by Swagger, for intuitive use with screen reader
Script	DS	Log Output Accessibilizer	Parses output of code execution, reading necessary lines only
Script	DS	Accessible Code Signer	Makes it easier to use a code signing certificate to sign exe and dll files
Script	PS	XBindKeys	Binds commands to certain keys or key combinations
Script	DS	Notifications Monitor	Manages notifications in Linux distributions and tells user
Script	CS	PPTX2MD	Converts slide decks into screen reader friendly Markdowns
Script	DS	Dashboard Accessibilizer	Retrieves visual information in dashboards through API calls
Script	T & I	Monitor and Read	Takes partial screenshot of screen and OCRs only that part
Script	CS	Volume Docker	Decreases meeting volume when screen reader speaks
Script	DS	SQLite Preview Navigator	Quickly navigates to SQLite preview with screen reader
Script	PS	Data Monitoring System	Pulls data from company products and presents it in an accessible way
Script	T & I	Statistics Retrieval System	Directly pulls data from API to bypass inaccessible internal statistics system
App	CS	Can You See Me	Offers guidance on position within camera for independent positioning
App	PS	Win My AI	Computer-Android emulator bridge to run Be My Eyes on Windows Desktop
App	DS	GPTCMD	Command line interface wrapper tool to help interface with OpenAI models
App	DS	Text-To-Speech in C#	TTS for Linux, Android, and Javascript which works without internet
App	PS	HTML Accessibilizer	Converts inaccessible PDF including formulas, etc. to an accessible HTML
App	DS	Android Client	TCP socket to make phone calls and access texts through a computer
App	T & I	VOCR	AI-powered OCR that integrates with VoiceOver
App	DS	SQL Manager Lite	Accessible, light version of Microsoft SQL Manager
App	DS	Log Filtering Tool	Pastes log output into text editor and filters out logs according to need
App	PS	Graphics Color Inverter	Inverts the color scheme only on the local machine on the graphics level
App	DS	Hex Editor	Screen reader accessible hex editor
App	PS	Brightness App	Sets display brightness either automatically or to predefined values
App	PS	Magnifier Glass	Enables zooming in on parts of the display on Windows 10

Table 3: DIY tools that BLVSPs built or used. The authors of this paper assigned temporary names to some tools without official names for reference purposes. "Add-On" refers to NVDA add-ons, "T & I" stands for text and image interpretation, "DS" for development support, "CS" for collaboration support, "PS" for productivity support, and "SRO" for screen reader optimization.

Type	Purpose	Tool Name	Description
Plugin	T & I	Button Labeler	Javascript extension to label unlabeled buttons
Plugin	PS	Accessibility Agent	Internal AI chat tool searching accessibility documents and provides information
Library	T & I	Web Accessibilizer	Web framework to help make HTML pages more accessible
Library	T & I	MathCat	Converts MathML to speech and braille
Screen Reader	T & I	Eloud	Less verbose screen reader for Emacs using a speech synthesizer
Screen Reader	T & I	Chatterbox	Text-to-speech solution in Swedish or English, for small pieces of text
Screen Reader	T & I	TDSR	Two Day Screen Reader. Command line interface screen reader for MacOS and Linux
Linux Distribution	DS	TalkingArch	Accessible version of Arch Linux live ISO image

Table 4: DIY tools that BLVSPs built or used (continued from Table 3).

a vendor *"months, and months, and months ago,"* asking *"Hey, can you just make this one code change?",* he never heard back, concluding that *"sitting on a long-haul flight and coding out a solution"* was easier. This sense that companies simply would not do the work was pervasive. For example, P22 stated *"I just don't think anyone is going to do it for us,"* and P24 explained, *"[my need] was never going to be solved by a mainstream company, so I rolled up my sleeves and did it."* Even when BLVSPs proactively wrote the code fix and sent it to the company, the issue remained unsolved: *"It kind of gets to me on a primal level, like if I'm sending you the code to fix it... It's still going to sit in the backlog forever and ever, and ever"* (P7).

Some participants chose to try various off-the-shelf workarounds before resorting to DIY-ing. However, these were often suboptimal and ultimately drove participants to create their own solutions. P22, tired of struggling with unreliable tools, shared: *"It is awful, it steals my dignity and makes me feel like an oaf... For a while I struggled with it. I was like, 'Is it me? It's not me. It's the tools. I'm going to build the tools. I'm perfectly capable.'"*

The involuntary DIY-ing we see here contradicts the mainstream notion of DIY-ing as a voluntary, recreational activity.

4.2.2 Intrinsic and Extrinsic Joys of Hacking.
Our participants identified hacking as a result of being intrinsic problem solvers and eager to help fellow BLV individuals with their DIY tools.

Several participants described themselves as intrinsic problem solvers who happened to have the domain expertise in software development, which motivated them to build DIY tools. Some BLVSPs believed that their *"brain is probably wired that way"* (P8), saying: *"I want to resolve this problem. Sometimes it's not worth the time, but often, it's fun"* (P8). P24 also described that *"[DIY-ing]'s happened quite a few times now at work,"* as his life being a blind person has been *"one massive life hack all the way through, constantly having to come up with workarounds from a really, really young age."*

Participants also had altruistic motivations as members of the BLV community, namely to help other BLV individuals. P15 described his three-fold motivation: *"It's part tinkering, part trying to help other people, and part just because I can."* P13 shared the satisfaction that comes from helping other people: *"It's also very satisfying when other people also find it helpful."* Similarly, P24 shared: *"I like the idea of putting stuff out there to help people."*

We see how building and sharing DIY tools are activities motivated by one's software professional identity as well as one's blindness identity.

4.2.3 Efficiency and Autonomy in Relation to Colleagues.
Participants desired to achieve higher efficiency and autonomy to maintain a professional image through DIY tools, which sometimes was not feasible with existing workarounds such as off-the-shelf technology and sighted assistance.

Just like any other software professional, our participants leveraged their domain expertise to get work done more efficiently: *"I'm not good at Excel. So if someone gives me a CSV... if I need to do any kind of analysis on it... my perception is that, 'You know what? It'll be quicker to just write some quick and dirty code to do it'"* (P24). Participants also referred to their *"laziness"* (P15): *"The stuff was accessible anyway, but instead of it being a six-step process, it's a one-step process because I wrote an add-on for something to do it"* (P28).

Unlike sighted software professionals, however, BLVSPs had an additional need to be efficient in the face of accessibility and usability barriers. When software is technically accessible but is not designed for screen reader use, the interaction becomes tedious and slow–a phenomenon we refer to as the "accessibility tax." What is often one click away for a sighted professional involves many key presses for BLVSPs. P16 described:

> *"I guess as a good blind person, you're supposed to start hitting the tab key 4,000 times. And then hitting enter when you hit the right thing... and it's only because you're forced to use this sighted paradigm. If you were going to build a blind-first interface, it wouldn't be a GUI [Graphical User Interface]."*

To circumvent this, P16 built non-GUI interfaces to *"just interact with the API"* and avoid the tax. Similarly, P8 described building his own tool to navigate JSON files, which *"washed away the unnecessary crap, and placed it in a tree view, so you can expand and collapse. So it was much easier to navigate the JSON output [with fewer keystrokes]."*

Participants were particularly concerned about how the accessibility tax would affect their professional image. P24 explained: *"I care a lot about the pace that I can do things at. It's not to do with my technical knowledge, but I am slower at a lot of things. And that can have a bit of an effect in terms of, 'Well, are people going to want to pair with me?'"* P22 shared his frustration when he lost access to tools after losing his vision mid-career:

> *"It really hurts me that I don't have the tools to [work fast] anymore. I also feel wasted, because I know what*

I can do. Other people don't. What they see when I use the tools that are available is someone who's fumbling around and struggling... And that feedback causes me crazy anxiety. It is horrifying... I hate that."

Through creating their own tools, participants sought to *"have the same speed as everybody else"* (P1) and *"compete effectively, and be someone who's valuable"* (P22).

In some cases, BLVSPs responded to the accessibility tax by *"fall[ing] back on sighted assistance"* (P28) from colleagues. Participants described that *"sometimes, using five different tech options is not an efficient use of time"* (P27), and *"the best option is just getting someone sighted to help you out"* (P12). Similarly, P9 found asking for sighted assistance *"a pain, but it works."*

However, many others worried about negative implications sighted assistance could have on their careers as well as their colleagues'. P24 had to ask *"whether I was in shot of my webcam or not"* in every meeting, which made it difficult to maintain a professional image: *"a sighted person might think, 'Oh, wow! They can't even tell whether his face is in shot of his webcam or not. That's really, really easy! If he can't do that, then what else can't he do?'"* Relatedly, P6 was concerned colleagues would become resentful: *"Asking for other people's time, and potentially having them delay their work might show adverse effects on [their colleagues' careers]. So that is one of the scary parts about reaching out and getting assistance."*

For these participants, their motivation to DIY-ing was *"primarily personal autonomy and independence. Not having to depend on sighted people for me to do my work"* (P12). DIY-ing was particularly appealing for recurring tasks: *"If it's something that I need to do very routinely, maybe more than once every couple months, I like to create a process for it, so I don't have to ask people"* (P7).

BLVSPs were faced with two unappealing options: risking peers' disrespect and resentment by seeking assistance, or risking low work productivity by declining it. DIY-ing became the only way out.

4.3 Impact of DIY Tools on Work

The DIY tools that BLVSPs built and used enhanced workplace accessibility, confidence, and equity, but such benefits were counterbalanced by significant access labor and friction with organizational policies.

4.3.1 Increased Accessibility, Confidence, and Equity. As described above, BLVSPs were motivated to DIY to increase accessibility, restore dignity, and enhance independence and efficiency; in large part, their tools delivered. In some cases the DIY tool was so critical that, without it, the BLVSP may not have been able to continue working in the field. P16 recounted the anxiety of setting up an accessible Linux environment upon starting a new position, noting that failure would have rendered him *"unable to start the job."* Similarly, P22 reported: *"I would not have been able to continue programming [after losing my vision] without something like Eloud."* An important theme in this regard is independence: the use of DIY tools did not just make them able to work as software professionals, but on many an occasion enabled them to perform work on their own. For example, P7 was able to independently perform accessibility testing with a DIY tool, whereas previously he *"had to always ask*

my colleagues" to verify aspects such as zooming, contrast, and underlined links.

With DIY tools, increased independence translated to increased speed and parity with sighted colleagues. For example, P26 credited "PPTX2MD" for *"drastically increas[ing] my speed"* in reviewing and preparing accessible software development presentations, making it *"invaluable to my time, and everyone else's time."* As a result, building DIY tools, *"level[s] the playing field inasmuch as it might make me quicker at my job"* (P24). In fact, DIY tools sometimes allowed BLVSPs to surpass their sighted colleagues. The set of strategies and technologies P17 developed for work *"made me actually more efficient than my sighted counterparts,"* as the DIY tools allowed him to bypass the traditional GUI paradigm, or *"the way that sighted people did it."*

As a result, BLVSPs became more confident at work, especially in the presence of sighted colleagues. P24 felt *"a lot more relaxed"* during videoconferencing meetings, as "Can You See Me" ensured he was properly *"in shot"*. For P22, the additional streamlining offered by DIY tools was key: *"I feel like I'm finally able to use a computer with dignity again... I don't have to do all these stupid workarounds while people are looking over my shoulder going, 'What the fuck is he doing?'"*

Sometimes, BLVSPs' efforts were rewarded with unexpected accolades. One participant[1] wrote DIY software to replace an inaccessible video playback system, which *"actually worked really well for the rest of the team as well."* The team *"ended up shelving the commercial tool,"* and he got promoted from an associate to a junior software engineer. Another BLVSP[1] was acknowledged as an expert on a specific API after using the API to create a DIY workaround to an inaccessible analytics system: *"Every now and then, [coworkers' response to DIY-ing] does surprise me, and something good will come from it."*

Overall, many participants voiced that DIY tools *"absolutely increased the quality of my work life"* (P12). They felt more productive, confident, and on equal footing with sighted colleagues, and sometimes even got promotions.

4.3.2 Spending Extra Time Building DIY Tools. While some BLVSPs received company support to build DIY accessibility tools during work hours, many spent personal time *"engineering my way around accessibility"* (P24). Some participants resented using personal time, some accepted it, and some enjoyed or learned to enjoy this process.

Some participants were allowed to build DIY tools during their work hours. P12 praised his company for letting him *"spend three weeks in a year designing and developing for accessibility"* that he can use *"for the remaining 49 weeks."* Others negotiated development time on a per-tool basis by reframing accessibility problems as usability problems that *"benefit the whole team if it's fixed"*:

> *"If you phrase it as just an accessibility problem, then realistically, you're either doing that in your own time, or during lunch... If you take the problem and rephrase it,... you can usually find the time to do it in the BAU [Business As Usual] time."* (P24)

More frequently, however, BLVSPs built DIY tools *"in [their] personal time"* (P25), although *"it's not really something that I should*

---

[1]We omit specific details in this example to reduce risk of deanonymization.

have to do outside of company hours" (P12). The time spent building their own solutions ranged anywhere from 30 minutes to several sleepless week nights and unrestful weekends. P24 shared how *"it's affecting my career negatively"*:

> *"Sometimes I've had to really draw on my energy reserves... As a blind software engineer, you might have Monday, where literally everything that you do just doesn't work, because it's all inaccessible. And then, you have to spend your evening engineering your way around these accessibility problems, and you can't go to bed until you've got that done. Not only that, but you have to have enough energy to go in on Tuesday, with a smile on your face, as if nothing's happened. Everyone else got a good night's sleep, but I only got two hours, right?"*

As a result, many participants expressed resentment towards having to build DIY tools outside of work hours. BLVSPs felt strongly against being *"used as beta testers"* (P25), and that *"blind people should not be used as solutions providers for products at work that should be accessible"* (P25). The time spent on DIY accessibility workarounds infringed on their ability to *"take a break from being blind"* (P16) and spend time *"work[ing] on other interesting [software] problems"* (P16) that are *"purely for myself"* (P13). Even P24, who earned a promotion based on his DIY work, did not perceive DIY-ing as career development and was keenly aware of the inequity: *"You could argue that working on accessibility software is career development. Because it's going to enable me to progress. But I just don't see it like that... because taking care of accessibility is not my job. And no one else has to do it."*

Several BLVSPs resignedly accepted the necessity of DIY development effort: *"it's annoying... But what's the alternative?"* (P24). For P7, it was simply *"inevitable"*, as he described: *"It's not ideal. But I don't know another way... as a blind individual, there's extra work that you're going to have to put in."* Similarly, P13 expressed reluctant acceptance: *"Now, it's just life. It's less annoying now, just part of work, even though it's not part of work."*

Despite feeling resentment and resignation, some participants enjoyed DIY-ing or tried to focus on its positive aspects.

Those who enjoyed DIY-ing alluded to their love of problem solving. For P25, it is a *"pretty clear win-win"* since *"I get to write code, and I get to make things easier for myself."* Yet, for others, finding enjoyment required a bit more effort: *"I'd much rather not have to... But there's almost always a silver lining"* (P24). P13 explained his coping mechanism of *"trying to figure out how I could enjoy this more and think of it less as work"*, identifying DIY solutions at the intersection of what he *wanted* to do and what *needs* to be done. Ultimately, feelings of resentment tended to overshadow enjoyment. P22 believed that there was a *"double-edged sword to that extra work that blind people have to do... Work outside of work that benefits work."* Similarly, reflecting on his experiences, P24 shared: *"There are good points around having to [DIY], but to be clear, I don't think they make up for the negatives."*

4.3.3 Organizational (Dis)Approval. Organizations had different policies on the use of DIY tools and open source assistive technologies, which varied widely depending on the company's size and flexibility. Restrictions compelled BLVSPs to keep DIY tools hidden to maintain their productivity and job security.

Some organizations—mostly smaller-sized companies—took a flexible approach to permitting DIY tools. For instance, one participant noted, *"it's a 200-person company, things are more relaxed and they don't enforce very strong policies on us"* (P10). Conversely, larger corporations imposed stricter regulations and required excessively complex approval processes, which were cumbersome and time-consuming. One participant[1] who had worked for both a startup and a large corporation explained the dichotomy: *"It's kind of night and day [between two organizations]... The smaller company doesn't care, as long as you don't leak important data,"* while the large company enforced *"a lot of policies. We are bound by so many different restrictions."* This contrast is well illustrated by the usage of DIY tool *"AI Content Describer"*; while the small companies that employ P10 and P15 permitted its use, the larger companies that employed P24 and P25 did not.

Companies' approval processes to review BLVSPs' DIY tools and authorize them for use at work were lengthy, required additional access labor, and were unlikely to return a positive result. P7 described needing to weigh productivity against compliance: *"You have two options: do it and shut up about it... or request permission... and wait a few months for them to maybe get back to you."* Even after months of waiting, participants faced the extra labor of educating the compliance team about accessibility, only to be denied use of the DIY tool:

> *"Someone gets back to you, and has, [with] all due respect, no idea what they're talking about... then I have to segue into the 'this is why accessibility is important' conversation. And we have to start from the total basics, which is... that cuts down on productivity, costs the company money, etcetera, etcetera."* (P7)

In the face of these options, some BLVSPs chose not to seek formal permission and to simply desist from using their DIY tool: *"[I'd] rather not deal with it [approval process], and not use it [DIY tool]"* (P7). On the other hand, to maintain productivity, some participants[2] described not notifying employers about their tools or being *"willing to bend the rules, just a little bit"*. One participant commented: *"Nah, he [employer] doesn't even know,"* while another BLVSP candidly admitted: *"the hacker part of me would probably just carry on using it anyway."*

Some BLVSPs questioned *why* DIY tools for accessibility required formal review in the first place, when sighted colleagues' DIY tools for efficiency did not. P1 argued: *"I think [my DIY tools] don't need to be [approved]... It's just some scripts that summarize data, or just present things in a different way."* Similarly, P24 insisted: *"Most developers write little scripts to automate something [and don't seek approval]...I don't believe I'm doing anything wrong."*

Some companies not only disallowed DIY tools but also strictly prohibited open-source software solutions found on platforms like GitHub, along with commercial workarounds. Even basic ATs and add-ons were frequently denied. P9 reported that *"we need security permission to install tools, and they normally will say no to any tool which an individual has built and no security checks have been done on."* P24 highlighted the necessity to sometimes install open-source

---

[2]We omit specific details in the next examples to reduce risk of deanonymization.

software screen readers on servers, which is *"just a huge no for a lot of [employers]."*

Disapproval could have severe professional consequences for our participants, such as losing their jobs. P10, who is currently allowed to use DIY tools for work, imagined being *"fired off"* without the tools, *"because my speed would have certainly decreased."* P13 expressed resentment over terminated contracts and lost income due to a company's refusal to either address accessibility issues or authorize DIY solutions: *"As long as I'm getting the work done, I don't know why they're not willing to let me come up with my own workarounds, because it's hard, extra work that I shouldn't have to do, and I'm taking the extra time, still willing to do it."* (P13)

4.4 Impact of DIY Tools on the BLV Community

A number of participants shared their DIY tools with the broader BLVSP and sometimes even the BLV community, leading to collaborative tool enhancements and farther-reaching advocacy, tinged with both gratification and regret. Deficiencies in current sharing networks revealed a need for better internal and external support for dissemination.

4.4.1 Sharing DIY Tools. Participants alluded to the BLV community's *"culture of sharing"* (P12) when explaining their drive to share DIY tools, tips, and tricks through internal and external social networks. Several participants shared their internal DIY tools with other BLV(SP) employees within the company: *"I put it in an internal repository... it's still [there] and some colleagues have used it"* (P15). They sometimes received *"tips and tricks"* (P1) from other BLV employees at the company and actively wrote to the internal mailing list about workplace accessibility problems to *"collaborate and brainstorm possible solutions"* (P12). Sometimes, a useful tool they found in the internal community became *"a part of my tools as well"* (P12).

External BLVSP communities that participants engaged with included Program-L—which was described as *"really the biggest one"* (P4), PythonVis, AccessComputing, Lime Connect, Programme Nationale (Portuguese), Ciegos Programadores (Portuguese), the NVDA users mailing list, Access India, and blindcoders.com. BLVSPs saw value in having a mailing list specifically tailored to BLVSPs, since *"it doesn't get as cluttered"* (P4) as other lists and *"everyone knows at least something about programming"* (P4). In cases where participants were unable to get help from sighted peers, P5 described leveraging external communities as an additional source of support.

4.4.2 Collaborating, Advocating, and Offloading. BLVSPs discovered DIY tools through online communities, as they were *"always looking for tools to make my life easier"* (P4). P1 explained that web searches often led him to results *"on a GitHub page or a mailing list."* Similarly, P10 highlighted the recommendations from other BLVSPs on Program-L: *"JAWS scripts, also some old scripts for Visual Studio 2008... This DBeaver thing I told you about, it was also recommended by Program-L."*

It was not uncommon for the participants to then contribute to some of the tools they found, ranging from suggesting issues to address *"so a lot of the time, if there's something that I think is obvious, that an add-on's not doing, I'll submit an issue about it"* (P7),

to small code fixes, to modifying another BLVSP's tool when *"they aren't kept up to date"* (P10).

Conversely, when they had published tools, some participants also received contributions from others. P7 recounted receiving pull requests for his DIY tools: *"I was just looking through my messages, and sure enough, a guy from three months ago wanted to do something with [DIY tool]."* P16 reported another BLVSP creating *"basically a next version"* of his DIY tool: *"He uses my [DIY tool] for everything, basically... And he's rewriting some of it... He's built stuff on it. And he's trying to contribute some of it back."*

The online BLVSP community also fostered *"community-based"* (P7), purposefully organized DIY-ing, as participants collaborated with others to addressed shared challenges: *"A few of these [DIY tools] were developed because we were complaining about something in our professional lives that didn't work well... then, Remote Support came about"* (P7).

Meanwhile, participation in these efforts was not always limited to BLVSPs. Indeed, participants described reducing the collective labor of DIY-ing by coordinating accessibility advocacy on mainstream sharing platforms. For example, sometimes BLVSPs would share a DIY proof of concept to nudge sighted developers to integrate accessibility features into typical software:

> *"There's a certain inaccessible tool where [BLVSPs] will just come up with an accessible alternative, and sometimes companies just basically say that 'This cannot be made accessible; it's just not possible,' which is not true, at all. Then [BLVSPs] will just write prototypes to prove that this is actually doable."* (P13)

In other cases, BLVSPs strategically sidestepped DIY labor altogether, by rallying around accessibility feature requests:

> *"Jupyter Notebook was not being accessible. Through Program-L, we all got together, and one of us decided to open up a GitHub issue, and then everybody would contribute to it, so that it would stay near the top, and be active. And [sighted] developers would see it. And eventually, [they] ... replied to the issue, and now it's a lot more accessible."* (P13)

4.4.3 Uptake of DIY Tools. BLVSPs took pride in the success and impact of their DIY tools on the BLV(SP) community. Some tools *"went viral"* (P24) through the community and were used by many other BLV individuals: *"I just woke up one morning, and it had like 30-something stars on GitHub"* (P7). P13 described: *"It's pretty exhilarating... also very satisfying when other people also find it helpful,"* while P7 had a *"huge learning experience"* as his tool was used in ways he had not anticipated.

Constructive, sometimes *"overwhelmingly positive"* (P7) feedback from the community motivated BLVSPs to improve their DIY tools, as others were now able to *"integrate [DIY tool] with their existing workflow"* (P7). P13 recounted a time where *"people were able to suggest more things to add,"* which helped him extend the tool's capability and *"just have one script that would work with three different platforms."*

Sometimes, BLVSPs' DIY tools even became inspiration for commercial implementations by companies. One participant[1] shared

how a company implemented and released a feature of a paid product, which *"feels incredibly similar"* to his *free*, open-source DIY tool: *"[Company] that made [product]... I'm pretty sure they took a lot of inspiration [for the feature] from [my DIY tool]. Even some of the wording is literally exactly the same as what I wrote."*

Although tools becoming integrated into mainstream technologies was *"cool, because now more people get access to it,"* it left the BLVSP with a bittersweet reminder that their valuable labor was uncompensated and unacknowledged: *"I did it first, just saying. But that's a sign of success, right?"*

4.4.4 Need for Internal Community. Many BLVSPs highly valued internal mailing lists as *"one of the best places to get [work-]specific help"* (P1): *"For more specific enterprise questions, internal [list] tends to be much more effective than external"* (P12). This was particularly because *"[company] has a bunch of technology that only exists at [company]"* (P25) and they could not discuss this with external BLVSPs.

Yet, only 40% of our participants' organizations (n = 11) had internal mailing lists for BLV employees: *"Honestly, there is nothing that I can reach out to internally"* (P5). Most BLVSPs who did not have internal mailing lists strongly desired them, saying that it *"would be extremely valuable"* (P7) for sharing *"different tricks and hacks and stuff that would be useful to get your work done, since you guys are all in the same company, completing similar work, or working towards the same goal"* (P23). P20 projected that such internal communities would *"be great to build confidence in yourself, 'Oh, there's somebody working in a similar field as me in the company.'"* Moreover, they might *"give everybody a bigger voice"* (P25), especially when pushing for accessibility fixes in inaccessible internal tools to ultimately influence company culture: *"If we are able to collaborate and bring out issues collectively... It would help us push for better accessibility as a group and may well open doors for more visually impaired users down the line."* (P9)

In addition to wanting a mailing list, participants emphasized the need for the company to publicize its existence, making it easier to find. This was partly because they did not want to disclose their disability while seeking such a community. P18 explained, *"I definitely felt like sometimes, some colleagues judged... so I was hesitant to even talk to some people,"* fearing that and that disclosure to seek community may *"bite me later"* in performance reviews. P7, who was unsure whether community existed at his company, argued for active publicizing, *"so that people do know that they have that resource to take advantage of"* and not *"suffer in silence."*

4.4.5 Need for Centralized External Community. BLVSPs expressed how they *"definitely need a place where we can find each other"* (P22), and desired for a centralized platform for BLVSPs to share and discover DIY tools and knowledge. Participants lamented the dissolution of Twitter (currently X), which used to be *"the place"* (P7) for BLVSPs to connect and share resources. Now, the BLV(SP) community is *"very much scattered"* (P7), *"segregated"* (P16) across many different platforms, and therefore *"hard to find"* (P22). For example, while Program-L is *"probably the largest resource that's out there"* (P17), three participants *"didn't know they existed"* (P29).

Difficulties finding community led to difficulties finding and sharing tools. P24 described how *"it's harder to find tools now"*, requiring *"either painstaking research, having a pint with other blind people, mailing lists, and Reddit, I guess. It's pretty sad"* (P16). On the flip side, some tools were unable to be shared as BLVSPs *"didn't know where to share... there's not a centralized place to share JAWS scripts"* (P10).

In addition to having a place for all their tools, participants wanted a specific, dedicated community where members have a *"shared understanding"* (P7) and could answer *"not just programming questions... [but] visual impairment related programming questions"* (P13), along with strategies and tools to overcome accessibility barriers at work. They also wanted *"a booklet or public repositories"* (P10) that documented *"some common tips and tricks for this kind of stuff... Like a Wiki... a place where people are encouraged to contribute and just describe, 'this is how a blind person may interact with a diff tool'"* (P1). P16 added how newcomers to the BLVSP identity should be equipped with a handbook: *"someone goes blind on Tuesday, and then 10 years later they find out that there's 10 mailing lists [for BLVSPs]. People need to be handed a book of all the things that exist for blind people on day one"* (P16).

Participants differentiated sharing on mainstream platforms with relatively less BLV engagement (e.g., GitHub) versus lesser known, highly specialized communities (e.g., Program-L), and each comes with its own limitations. Mainstream platforms have a large audience but may hinder discoverability of "blindness tools" (P24):

> *"There's a website called Hacker News, which is for techy people, and that will occasionally recommend tools to me, which is good, but they're not blindness tools... The blindness tools are never going to get promoted on that [website], so I definitely feel like I've lost something now that I don't get those recommendations [from Twitter]."* (P24)

Specialized communities, on the other hand, are harder to locate, if they even exist (e.g., P10 described the absence of a JAWS scripts platform, and P7 noted that the NVDA Add-On store was only recently created), and reach fewer people. Participants described the email-based nature of Program-L as *"overwhelming"* (P7): *"It's really ridiculous that mailing lists are the main way"* (P16).

5 Discussion

HCI researchers have been investigating how BLV individuals employ life hacks to address inaccessibility in their everyday lives [44], and Software Engineering researchers have studied how programmers create customized DIY solutions to fit their work needs [94]. To the best of our knowledge, this is the first study to explore the experience of BLVSPs, who have dual identities of (1) the BLV hacker and (2) the software professional hacker, about their experiences in building and using DIY solutions for the workplace.

5.1 The Double Hacker Dilemma

The type of hacking we found BLVSPs engaging in stems from the fact they exhibited two intersecting identities: one as a software professional hacker and the other one as a BLV hacker. BLVSPs, just like other software professionals, built their own tools to streamline tasks at work [9, 40, 60]. In the process, they equally acted as creative problem solvers and enjoyed coding solutions to address problems and needs. Moreover, in line with prior research on work

hacking [53, 97] and software developer DIY-ing [94], our participants were motivated by the desire to be more efficient and help others, and their tools served purposes like general automation and monitoring.

At the same time, however, our participants' BLV identities necessitated hacking not just for convenience, but for overcoming accessibility barriers. While much research has observed such hacking outside the workplace [23, 51, 70, 74], our study shows that BLVSPs had critical additional motivations to DIY at work. This included wanting to be independent and seeking greater efficiency so as to shave off the "accessibility tax" of being a screen reader user–all in service of maintaining their sense of dignity and a professional image while staying apace with sighted colleagues.

Building DIY tools for workplace accessibility was a double-edged sword for our participants. Through building DIY tools, participants not only made their work lives better, but at times were recognized by management and in some cases even earned promotions. Yet, this did not necessarily offset the intense labor of struggling through inaccessible, clunky tools, trying to use existing workarounds and failing, building DIY solutions, and going through release, maintenance, and corporate compliance processes. Here, we see how BLVSPs' visible work on the job was only made possible by the uncompensated, invisible access labor [20, 78] they did behind the scenes.

To capture the tension that exists behind this double-edged sword, we introduce the notion of the "Double Hacker Dilemma" to explain the difficult situation that BLVSPs repeatedly face: request organizational troubleshooting by submitting accessibility support tickets that may never be addressed, or take matters into their own hands and leverage their expertise to build DIY solutions? This dilemma is uniquely experienced by BLVSPs. It is distinct from the situation for non-disabled software professional hackers, because they can choose not to DIY, yet still perform their work as normally without 'standing out' as incapable amidst their co-workers. It is equally distinct from the situation for non-software professional BLV workers, who, lacking the technical expertise to create solutions, necessarily are dependent on others (co-workers, organizational troubleshooting, Program-L advice) to provide solutions. BLVSPs live at the intersection: they have the ability to solve their own problems, but should they?

When our participants submitted support tickets as BLV employees, requests for support typically languished in the backlog for months, sometimes never being resolved. Our BLVSPs, equipped with their additional software professional hacker identities, thus often chose to resolve the problem by leveraging their expertise and DIY-ing solutions such as the command line interface tool connecting to the code review system and identifies the code changes for review or the accessible, light version of Microsoft SQL Manager, to name two examples among the many in Table 3. Doing so, however, typically and unfortunately came at the expense of uncompensated labor, time, and effort. Further compounding the dilemma, even when they built a successful solution, external companies could reject their fix or internal compliance barriers could prevent use within their workplace.

Our participants' experiences challenged the notion of DIY as a leisure activity [26]. Individuals—including our BLVSPs—found hacking enjoyable and personally fulfilling [26, 44], and sometimes found that it enhanced task productivity [15, 53, 62, 97]. However, DIY-ing out of necessity to address accessibility shortcomings at work led to these positive feelings frequently being usurped by a sense of resentment being token solution providers, having to spend personal time, and their tools being singled out and sanctioned, compared to their sighted colleagues' DIY tools that were built for higher productivity and efficiency. The double hacker dilemma thus adds to the well-known systemic accessibility and accommodation issues across the tech industry [45]. Even when employers provide BLVSPs with accessible setups, these turn out to be incomplete and not as effective as they could be, which is why our participants rolled up their sleeves to build DIY solutions to address their needs and bring themselves to a more equitable playing field. Two thirds of our participants necessarily engaged in DIY, a number that we conjecture far outpaces that of non-BLVSP colleagues, and is especially poignant because BLVSP DIY-ing concerns essential needs rather than conveniences.

5.2 Towards a Company Culture of Interdependence

Our findings revealed contrasting challenges for BLVSPs working in companies of different sizes. In larger corporations, restrictive policies and complex approval processes—a well-known phenomenon in tech companies [75, 95]—hindered DIY tool use, while infrastructure for internal community support was mostly—though not always—present. In contrast, smaller companies offered flexibility with regards to policies but lacked internal resources to facilitate the discovery and dissemination of DIY tools. With the exception of two participants, BLVSPs did not have access to both a robust internal BLV community and adequate support by the company towards using DIY tools.

Companies are attempting to make their workplaces more accessible [39, 66]. We argue that this has to go beyond providing accessible technologies, instead requiring a culture shift (in line with, e.g., [90, 98]). Some of this culture shift needs to happen organizationally, especially in larger companies that typically are not as nimble as startups in adopting new tools [7, 75, 95, 108]. Important first steps include differentiating approval processes for DIY tools requested by BLVSPs from other tools that developers ask to bring into the workplace, streamlining the approval process, procurement mandating accessibility in tools being acquired, and creating affinity or resource groups [38].

We also contend that, in addition to procedural changes, this cultural shift must include changes in colleagues' understanding and attitudes towards BLVSPs and accessible (DIY) tools. Indeed, efficient authorization of DIY tools does not necessarily translate to seamless integration into workflows. For example, the new tool may require sighted colleagues to alter their individual and collaborative work processes. A select few of our BLVSPs were successful in bringing their sighted colleagues into the conversation. As those colleagues come to understand access needs of BLVSPs, they may join them in adopting, advocating for, or even collaborating on developing accessible internal DIY tools. When coworkers acknowledge and enact this interdependence, the culture becomes one that can disrupt ability-based hierarchies by revealing the contributions of people with disabilities [14].

5.3 Towards a Centralized Platform for Sharing DIY Tools and Knowledge

Our findings point to a need for a global, accessible platform for sharing, finding, discussing, and improving DIY tools outside of the workplace. Participants expressed frustration about the currently fragmented BLVSP community following the disintegration of Twitter that once served as a centralized hub for BLVSPs where they could at least discover and share tools. Today, DIY tools are scattered across different subcommunities across many mailing lists, subreddits, GitHub repositories, and Discord servers. This led to a lack of awareness of available tools (aligning with Momotaz et al.'s observation [70]), participants looking for certain tools but not finding them, and multiple participants building DIY tools serving the same purpose.

The desire for a platform was not only about the DIY tools, however. BLVSPs overwhelmingly expressed a need for a support network beyond their company's BLVSP community, if their company even had one. Especially for those at smaller companies, but often equally so for BLVSPs at larger companies, they wished to meet fellow BLVSPs, feel a sense of belonging and validation, share knowledge [100], and have the ability to actively discuss, make suggestions for, and in some cases actively contribute to the ecosystem of DIY tools for work. For companies that commercially develop ATs and software tools, such a platform can equally help them connect to the BLVSP community, understand its needs, and thereby co-create improved accessible technology [8].

Ultimately, a platform such as described here could bring together and seamlessly integrate the best of existing platforms such as GitHub, Twitter (X), Reddit, and Program-L, but re-imagined in a highly accessible manner. For example, the platform could include a searchable listing of available DIY tools and accessibility ratings provide by other BLV users. It may additionally provide a space for BLVSPs to find, communicate with, and provide mutual support (e.g., mentorship) for one another. Today, individual companies mostly benefit from the DIY labor of BLVSPs. The fruits of BLVSPs' largely uncompensated labor, however, ought to be returned to the greater BLV(SP) community and a global DIY-centered platform could make progress toward that vision.

6 Limitations and Future Work

Our sample size (30) is more than double the median(13) reported in accessibility research focusing on BLV individuals [61]. However, most of our participants are located in the United States of America (18), with other participants in Europe (6), India (4), and Brazil (2); therefore, our findings do not represent the global experiences of BLVSPs. In addition, our participants come from various countries with different cultures and legislations around disability and accommodations. For instance, the Americans with Disabilities Act (ADA) [76], a landmark legislation in the United States, was enacted in 1990, while a comparable law in Brazil [28] was only recently passed in 2015. We did not collect data regarding how different disability legislations and policies in each country shape BLVSPs' experiences DIY-ing for workplace accessibility.

Our participants reported diverse visual abilities. 60% of participants (n =1 8) were totally or completely blind with little or no light perception, and 30% (n = 12) had varying degrees of vision.

However, we acknowledge that this sample distribution may not fully represent real-world demographics.

The gender demographics of our participants were skewed towards men (26 men, 3 women, 1 non-binary), which is a recognized limitation in software engineering research [29, 55, 103]. Furthermore, we did not explore how intersecting identities, such as race, ethnicity, other disabilities, or sexual orientation, may impact participants' work experiences.

This study did not cover every role in the software development process. However, we gathered experiences and perspectives of a much wider range of job positions in software development compared to many pieces of prior literature. Additionally, our study made sure to include BLVSPs with a wide range of experience, seniority, and positions.

Future work should follow up on the assorted suggestions we make, firstly in terms of investigating how to best design and institute effective, interdependent corporate practices. Based on our findings, future work could also investigate features that BLVSPs desire in a centralized, accessible community (e.g., the list of available DIY tools and their descriptions, an ability to request features, integration with GitHub, success stories around introducing tools): toward this, we encourage designing the community platform with BLVSP users to best implement, deploy, and maintain an effective global BLVSP community platform. Lastly, how different social (e.g., advocacy, culture) and legistlative settings around disability across countries impact workplace accessibility is a warranted direction for future work.

7 Conclusion

In this study, we presented our findings from semi-structured interviews with 30 BLVSPs about building and using DIY tools to support accessible software development. We identified four novel themes, each with numerous insights underneath: (1) the DIY tools they built and used, (2) motivations for building and using DIY tools in the workplace, (3) impacts of DIY tools on BLVSPs' work, and (4) impacts of DIY tools on the broader BLV(SP) community. BLVSPs found, built, and maintained a wide range of DIY tools they used daily. Yet, the "Double Hacker Dilemma" they faced each time they considered DIY-ing for workplace accessibility highlights the need for companies to significantly improve how they support and value BLVSPs' DIY work. Moreover, the lack of an adequate global BLVSP DIY platform means dissemination and discourse are meager, and notably hinders the emergence of a thriving community.

Acknowledgments

References

[1] NV Access. 2017. About NV Access. https://www.nvaccess.org/about-nv-access/
[2] June Ahn, Brian S. Butler, Cindy Weng, and Sarah Webster. 2013. Learning to be a better Q'er in social Q&A sites: social norms and information artifacts. In

Proceedings of the 76th ASIS&T Annual Meeting: Beyond the Cloud: Rethinking Information Boundaries (Montreal, Quebec, Canada) *(ASIST '13)*. American Society for Information Science, USA, Article 4, 10 pages.

[3] Taslima Akter, Yoonha Cha, Isabela Figueira, Stacy M. Branham, and Anne Marie Piper. 2023. "If I'm supposed to be the facilitator, I should be the host": Understanding the Accessibility of Videoconferencing for Blind and Low Vision Meeting Facilitators. In *Proceedings of the 25th International ACM SIGACCESS Conference on Computers and Accessibility* (New York, NY, USA) *(ASSETS '23)*. Association for Computing Machinery, New York, NY, USA, Article 45, 14 pages. https://doi.org/10.1145/3597638.3608420

[4] Khaled Albusays and Stephanie Ludi. 2016. Eliciting Programming Challenges Faced by Developers with Visual Impairments: Exploratory Study. In *Proceedings of the 9th International Workshop on Cooperative and Human Aspects of Software Engineering* (Austin, Texas) *(CHASE '16)*. Association for Computing Machinery, New York, NY, USA, 82–85. https://doi.org/10.1145/2897586.2897616

[5] Khaled Albusays, Stephanie Ludi, and Matt Huenerfauth. 2017. Interviews and Observation of Blind Software Developers at Work to Understand Code Navigation Challenges. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility* (Baltimore, Maryland, USA) *(ASSETS '17)*. Association for Computing Machinery, New York, NY, USA, 91–100. https://doi.org/10.1145/3132525.3132550

[6] Katherine H. Allen, Audrey K. Balaska, Reuben M. Aronson, Chris Rogers, and Elaine Schaertl Short. 2023. Barriers and Benefits: The Path to Accessible Makerspaces. In *Proceedings of the 25th International ACM SIGACCESS Conference on Computers and Accessibility* (New York, NY, USA) *(ASSETS '23)*. Association for Computing Machinery, New York, NY, USA, Article 7, 14 pages. https://doi.org/10.1145/3597638.3608414

[7] Daniel A Almeida, Gail C Murphy, Greg Wilson, and Michael Hoye. 2019. Investigating whether and how software developers understand open source software licensing. *Empirical Software Engineering* 24 (2019), 211–239.

[8] Julia Amann and Sara Rubinelli. 2017. Views of Community Managers on Knowledge Co-creation in Online Communities for People With Disabilities: Qualitative Study. *J Med Internet Res* 19, 10 (10 Oct 2017), e320. https://doi.org/10.2196/jmir.7406

[9] Mahta Amini, Jay Olson, and Zohreh Sharafi. 2024. Coding with a Creative Twist: Investigating the Link Between Creativity Scores and problem-solving Strategies. In *Proceedings of the 2024 ACM/IEEE 44th International Conference on Software Engineering* (Lisbon, Portugal) *(ICSE-NIER'24)*. Association for Computing Machinery, New York, NY, USA, 21–25. https://doi.org/10.1145/3639476.3639766

[10] Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. 2012. Discovering value from community activity on focused question answering sites: a case study of stack overflow. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Beijing, China) *(KDD '12)*. Association for Computing Machinery, New York, NY, USA, 850–858. https://doi.org/10.1145/2339530.2339665

[11] Ameer Armaly, Paige Rodeghero, and Collin McMillan. 2018. AudioHighlight: Code Skimming for Blind Programmers. In *2018 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, Madrid, Spain, 206–216. https://doi.org/10.1109/ICSME.2018.00030

[12] Ameer Armaly, Paige Rodeghero, and Collin McMillan. 2018. A Comparison of Program Comprehension Strategies by Blind and Sighted Programmers. *IEEE Transactions on Software Engineering* 44, 8 (2018), 712–724. https://doi.org/10.1109/TSE.2017.2729548

[13] Catherine M. Baker, Lauren R. Milne, and Richard E. Ladner. 2015. StructJumper: A Tool to Help Blind Programmers Navigate and Understand the Structure of Code. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 3043–3052. https://doi.org/10.1145/2702123.2702589

[14] Cynthia L. Bennett, Erin Brady, and Stacy M. Branham. 2018. Interdependence as a Frame for Assistive Technology Research and Design. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility* (Galway, Ireland) *(ASSETS '18)*. Association for Computing Machinery, New York, NY, USA, 161–173. https://doi.org/10.1145/3234695.3236348

[15] Peter Bloom and Martyna Śliwa. 2022. Hacking work: Critically examining the implications of the new discourse and practices of hacking for work intensification and organisational control. *Human Relations* 75, 5 (2022), 795–816. https://doi.org/10.1177/0018726721996763 arXiv:https://doi.org/10.1177/0018726721996763

[16] Yash Bohre, Purba Joshi, and Rowan Page. 2023. A Review of the Potential and Path to the Large-Scale Adaptation of DIY in Assistive Technology. In *Design in the Era of Industry 4.0, Volume 1*, Amaresh Chakrabarti and Vishal Singh (Eds.). Springer Nature Singapore, Singapore, 1067–1079.

[17] Andrea Bonaccorsi and Cristina Rossi. 2003. Why Open Source software can succeed. *Research Policy* 32, 7 (2003), 1243–1258. https://doi.org/10.1016/S0048-7333(03)00051-9 Open Source Software Development.

[18] Stacy M. Branham and Shaun K. Kane. 2015. The Invisible Work of Accessibility: How Blind Employees Manage Accessibility in Mixed-Ability Workplaces. In *Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility* (Lisbon, Portugal) *(ASSETS '15)*. Association for Computing Machinery, New York, NY, USA, 163–171. https://doi.org/10.1145/2700648.2809864

[19] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative research in psychology* 3, 2 (2006), 77–101. https://doi.org/10.1191/1478088706QP063OA

[20] Yoonha Cha, Isabela Figueira, Jessy Ayala, Emory James Edwards, Joshua Garcia, André van der Hoek, and Stacy Marie Branham. 2024. "Do You Want Me to Participate or Not?": Investigating the Accessibility of Software Development Meetings for Blind and Low Vision Professionals. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 933, 17 pages. https://doi.org/10.1145/3613904.3642130

[21] Yoonha Cha, Victoria Jackson, Isabela Figueira, Stacy Marie Branham, and André van der Hoek. 2024. Understanding the Career Mobility of Blind and Low Vision Software Professionals. In *Proceedings of the 2024 IEEE/ACM 17th International Conference on Cooperative and Human Aspects of Software Engineering* (Lisbon, Portugal) *(CHASE '24)*. Association for Computing Machinery, New York, NY, USA, 170–181. https://doi.org/10.1145/3641822.3641872

[22] Jacek Czerwonka, Michaela Greiler, Christian Bird, Lucas Panjer, and Terry Coatta. 2018. Codeflow: Improving the code review process at microsoft: A discussion with jacek czerwonka, michaela greiler, christian bird, lucas panjer, and terry coatta. *Queue* 16, 5 (2018), 81–100.

[23] Maitraye Das, Katya Borgos-Rodriguez, and Anne Marie Piper. 2020. Weaving by Touch: A Case Analysis of Accessible Making. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–15. https://doi.org/10.1145/3313831.3376477

[24] Maitraye Das, Darren Gergle, and Anne Marie Piper. 2019. "It doesn't win you friends": Understanding Accessibility in Collaborative Writing for People with Vision Impairments. *Proceedings of the ACM on Human-Computer Interaction* 3, CSCW (Nov. 2019), 1–26. https://doi.org/10.1145/3359293

[25] Maitraye Das, Abigale Stangl, and Leah Findlater. 2024. " That comes with a huge career cost:" Understanding Collaborative Ideation Experiences of Disabled Professionals. *Proceedings of the ACM on Human-Computer Interaction* 8, CSCW1 (2024), 1–28.

[26] Sarah R. Davies. 2018. Characterizing Hacking: Mundane Engagement in US Hacker and Makerspaces*. *Science, Technology, & Human Values* 43, 2 (2018), 171–197. https://doi.org/10.1177/0162243917703464

[27] Ana Cléssia Pereira Lima De Araújo, Maria Analice D. Santos Sampaio, Edward Martins Costa, Ahmad Saeed Khan, Guilherme Irffi, and Rayssa Alexandre Costa. 2022. The quotas law for people with disabilities in Brazil: is it a guarantee of employment? *International Review of Applied Economics* 36, 4 (July 2022), 496–525. https://doi.org/10.1080/02692171.2021.1962257

[28] Disability:IN. 2024. Brazil. https://disabilityin.org/country/brazil/

[29] Juliana Carvalho Silva do Outão, Luiz Alexandre Martins da Costa, Rodrigo Pereira dos Santos, and Alexander Serebrenik. 2024. Investigating the Barriers that Women Face in Software Development Teams Focusing on the Context of Proprietary Software Ecosystems. In *Software Business*, Sami Hyrynsalmi, Jürgen Münch, Kari Smolander, and Jorge Melegati (Eds.). Springer Nature Switzerland, Cham, 164–170.

[30] Dale Dougherty. 2012. The Maker Movement. *Innovations: Technology, Governance, Globalization* 7, 3 (07 2012), 11–14. https://doi.org/10.1162/INOV_a_00135 arXiv:https://direct.mit.edu/itgg/article-pdf/7/3/11/704919/inov_a_00135.pdf

[31] Md Ehtesham-Ul-Haque, Syed Mostofa Monsur, and Syed Masum Billah. 2022. Grid-Coding: An Accessible, Efficient, and Structured Coding Paradigm for Blind and Low-Vision Programmers. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2022-10-28) *(UIST '22)*. Association for Computing Machinery, New York, NY, USA, 1–21. https://doi.org/10.1145/3526113.3545620

[32] Olutayo Falase, Alexa F. Siu, and Sean Follmer. 2019. Tactile Code Skimmer: A Tool to Help Blind Programmers Feel the Structure of Code. In *Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility* (Pittsburgh, PA, USA) *(ASSETS '19)*. Association for Computing Machinery, New York, NY, USA, 536–538. https://doi.org/10.1145/3308561.3354616

[33] Youmei Fan, Tao Xiao, Hideaki Hata, Christoph Treude, and Kenichi Matsumoto. 2024. "My GitHub Sponsors profile is live!" Investigating the Impact of Twitter/X Mentions on GitHub Sponsors. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering* (Lisbon, Portugal) *(ICSE '24)*. Association for Computing Machinery, New York, NY, USA, Article 191, 12 pages. https://doi.org/10.1145/3597503.3639127

[34] Hongbo Fang, Hemank Lamba, James Herbsleb, and Bogdan Vasilescu. 2022. "This is damn slick!": estimating the impact of tweets on open source project

popularity and new contributors. In *Proceedings of the 44th International Conference on Software Engineering* (Pittsburgh, Pennsylvania) *(ICSE '22)*. Association for Computing Machinery, New York, NY, USA, 2116–2129. https://doi.org/10.1145/3510003.3510121

[35] Avelino Ferreira Gomes Filho and Rodrigo De Toledo. 2015. Visual Management and Blind Software Developers. In *2015 Agile Conference*. IEEE, Washington, DC, 31–39. https://doi.org/10.1109/Agile.2015.14

[36] Freelists. 2024. Program-L. https://www.freelists.org/list/program-l. Online; accessed 10-Sept-2024.

[37] Marco Gerosa, Igor Wiese, Bianca Trinkenreich, Georg Link, Gregorio Robles, Christoph Treude, Igor Steinmacher, and Anita Sarma. 2021. The Shifting Sands of Motivation: Revisiting What Drives Contributors in Open Source. In *Proceedings of the 43rd International Conference on Software Engineering* (Madrid, Spain) *(ICSE '21)*. IEEE Press, Madrid, Spain, 1046–1058. https://doi.org/10.1109/ICSE43902.2021.00098

[38] Aaron M Glassman and Myron Glassman. 2017. The use of affinity groups by fortune 100 firms. *Journal of Business Diversity* 17, 2 (2017), 104–114.

[39] Google. 2024. Google Diversity Annual Report 2024.

[40] Wouter Groeneveld, Lynn Van den Broeck, Joost Vennekens, and Kris Aerts. 2022. Self-Assessing Creative Problem Solving for Aspiring Software Developers: A Pilot Study. In *Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education Vol. 1* (Dublin, Ireland) *(ITiCSE '22)*. Association for Computing Machinery, New York, NY, USA, 5–11. https://doi.org/10.1145/3502718.3524759

[41] A. Guzzi, A. Bacchelli, M. Lanza, M. Pinzger, and A. van Deursen. 2013. Communication in open source software development mailing lists. In *2013 10th IEEE Working Conference on Mining Software Repositories (MSR 2013)*. IEEE Computer Society, Los Alamitos, CA, USA, 277–286. https://doi.org/10.1109/MSR.2013.6624039

[42] Foad Hamidi, Melanie Baljko, Toni Kunic, and Ray Feraday. 2014. Do-It-Yourself (DIY) Assistive Technology: A Communication Board Case Study. In *Computers Helping People with Special Needs*, Klaus Miesenberger, Deborah Fels, Dominique Archambault, Petr Peňáz, and Wolfgang Zagler (Eds.). Springer International Publishing, Cham, 287–294.

[43] Liwen He, Yifan Li, Mingming Fan, Liang He, and Yuhang Zhao. 2023. A Multimodal Toolkit to Support DIY Assistive Technology Creation for Blind and Low Vision People. In *Adjunct Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (San Francisco, CA, USA) *(UIST '23 Adjunct)*. Association for Computing Machinery, New York, NY, USA, Article 3, 3 pages. https://doi.org/10.1145/3586182.3616646

[44] Jaylin Herskovitz, Andi Xu, Rahaf Alharbi, and Anhong Guo. 2023. Hacking, Switching, Combining: Understanding and Supporting DIY Assistive Technology Design by Blind People. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 57, 17 pages. https://doi.org/10.1145/3544548.3581249

[45] Matthew J. Hill, Nicole Maestas, and Kathleen J. Mullen. 2016. Employer accommodation and labor supply of disabled workers. *Labour Economics* 41 (2016), 291–303. https://doi.org/10.1016/j.labeco.2016.05.013 SOLE/EALE conference issue 2015.

[46] Jonathan Hook, Sanne Verbaan, Abigail Durrant, Patrick Olivier, and Peter Wright. 2014. A study of the challenges related to DIY assistive technology in the context of children with disabilities. In *Proceedings of the 2014 Conference on Designing Interactive Systems* (Vancouver, BC, Canada) *(DIS '14)*. Association for Computing Machinery, New York, NY, USA, 597–606. https://doi.org/10.1145/2598510.2598530

[47] Earl W. Huff, Kwajo Boateng, Makayla Moster, Paige Rodeghero, and Julian Brinkley. 2020. Examining The Work Experience of Programmers with Visual Impairments. In *2020 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, Adelaide, SA, Australia, 707–711. https://doi.org/10.1109/ICSME46990.2020.00077

[48] Mina Huh and Amy Pavel. 2024. *DesignChecker: Visual Design Support for Blind and Low Vision Web Developers*. arXiv. https://doi.org/10.1145/3654777.3676369 arXiv:2407.17681 [cs]

[49] Syed Fatiul Huq, Abdulaziz Alshayban, Ziyao He, and Sam Malek. 2023. #A11yDev: Understanding Contemporary Software Accessibility Practices from Twitter Conversations. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 217, 18 pages. https://doi.org/10.1145/3544548.3581455

[50] Amy Hurst and Shaun Kane. 2013. Making "making" accessible. In *Proceedings of the 12th International Conference on Interaction Design and Children* (New York, New York, USA) *(IDC '13)*. Association for Computing Machinery, New York, NY, USA, 635–638. https://doi.org/10.1145/2485760.2485883

[51] Amy Hurst and Jasmine Tobias. 2011. Empowering individuals with do-it-yourself assistive technology. In *The Proceedings of the 13th International ACM SIGACCESS Conference on Computers and Accessibility* (Dundee, Scotland, UK) *(ASSETS '11)*. Association for Computing Machinery, New York, NY, USA, 11–18.

[52] Liz Jackson. 2018. We Are the Original Lifehackers. https://www.nytimes.com/2018/05/30/opinion/disability-design-lifehacks.html. Accessed: 2024-08-04.

[53] Saj Jetha. 2019. *The Smarts: Big Little Hacks to Take You a Long Way at Work*. Penguin Random House, United Kingdom.

[54] Jazette Johnson, Andrew Begel, Richard Ladner, and Denae Ford. 2022. Program-L: Online Help Seeking Behaviors by Blind and Low Vision Programmers. In *2022 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, Roma, Italy, 1–6. https://doi.org/10.1109/VL/HCC53370.2022.9833106

[55] Karina Kohl and Rafael Prikladnicki. 2022. Benefits and Difficulties of Gender Diversity on Software Development Teams: A Qualitative Study. In *Proceedings of the XXXVI Brazilian Symposium on Software Engineering* (Virtual Event, Brazil) *(SBES '22)*. Association for Computing Machinery, New York, NY, USA, 21–30. https://doi.org/10.1145/3555228.3555253

[56] Karim R. Lakhani and Robert G. Wolf. 2005. Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects. In *Perspectives on Free and Open Source Software*. The MIT Press, Cambridge, MA, USA. https://doi.org/10.7551/mitpress/5326.003.0005 arXiv:https://direct.mit.edu/book/chapter-pdf/2297407/9780262256124_caa.pdf

[57] Franklin Mingzhe Li, Di Laura Chen, Mingming Fan, and Khai N. Truong. 2021. "I Choose Assistive Devices That Save My Face": A Study on Perceptions of Accessibility and Assistive Technology Use Conducted in China. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 372, 14 pages. https://doi.org/10.1145/3411764.3445321

[58] Silvia Lindtner. 2015. Hacking with Chinese Characteristics: The Promises of the Maker Movement against China's Manufacturing Culture. *Science, Technology, & Human Values* 40, 5 (2015), 854–879. https://doi.org/10.1177/0162243915590861

[59] Silvia Lindtner and David Li. 2012. Created in China: the makings of China's hackerspace community. *Interactions* 19, 6 (nov 2012), 18–22. https://doi.org/10.1145/2377783.2377789

[60] Ming-Chou Liu and Hsi-Feng Lu. 2002. A STUDY ON THE CREATIVE PROBLEM-SOLVING PROCESS IN COMPUTER PROGRAMMING. , 6 pages.

[61] Kelly Mack, Emma McDonnell, Dhruv Jain, Lucy Lu Wang, Jon E. Froehlich, and Leah Findlater. 2021. What Do We Mean by "Accessibility Research"? A Literature Survey of Accessibility Papers in CHI and ASSETS from 1994 to 2019. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 371, 18 pages. https://doi.org/10.1145/3411764.3445412

[62] Anne Marie McEwan. 2016. *Smart working: Creating the next wave*. Routledge, 711 Third Avenue, New York, NY 10017, USA.

[63] Sean Mealin and Emerson Murphy-Hill. 2012. An exploratory study of blind software developers. In *2012 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, Innsbruck, Austria, 71–74. https://doi.org/10.1109/VLHCC.2012.6344485

[64] Janis Lena Meissner, John Vines, Janice McLaughlin, Thomas Nappey, Jekaterina Maksimova, and Peter Wright. 2017. Do-It-Yourself Empowerment as Experienced by Novice Makers with Disabilities. In *Proceedings of the 2017 Conference on Designing Interactive Systems* (Edinburgh, United Kingdom) *(DIS '17)*. Association for Computing Machinery, New York, NY, USA, 1053–1065. https://doi.org/10.1145/3064663.3064674

[65] Andre N. Meyer, Laura E. Barton, Gail C. Murphy, Thomas Zimmermann, and Thomas Fritz. 2017. The Work Life of Developers: Activities, Switches and Perceived Productivity. *IEEE Transactions on Software Engineering* 43, 12 (2017), 1178–1193. https://doi.org/10.1109/TSE.2017.2656886

[66] Microsoft. 2023. Diversity & Inclusion Report | Global Diversity and Inclusion at Microsoft. https://www.microsoft.com/en-us/diversity/inside-microsoft/annual-report

[67] Microsoft. 2024. Visual Studio Code - Code editing. https://code.visualstudio.com/

[68] MIRO. 2024. MIRO | The Innovation Workspace. https://miro.com/

[69] Takahiro Miura, Hiroki Fujii, Ryuki Yamazaki, Masaki Matsuo, Masatsugu Sakajiri, and Junji Onishi. 2024. Accessible Terminal Application for Visually Impaired Users Utilizing Screen Readers. , 13 pages.

[70] Farhani Momotaz, Md Ehtesham-Ul-Haque, and Syed Masum Billah. 2023. Understanding the Usages, Lifecycle, and Opportunities of Screen Readers' Plugins. *ACM Trans. Access. Comput.* 16, 2, Article 17 (jul 2023), 35 pages. https://doi.org/10.1145/3582697

[71] Farhani Momotaz, Md Touhidul Islam, Md Ehtesham-Ul-Haque, and Syed Masum Billah. 2021. Understanding screen readers' plugins. In *Proceedings of the 23rd International ACM SIGACCESS Conference on Computers and Accessibility*. Association for Computing Machinery, New York, NY, USA, 1–10.

[72] Aboubakar Mountapmbeme, Obianuju Okafor, and Stephanie Ludi. 2022. Addressing Accessibility Barriers in Programming for People with Visual Impairments: A Literature Review. *ACM Transactions on Accessible Computing* 15, 1 (March 2022), 7:1–7:26. https://doi.org/10.1145/3507469

[73] Emerson Murphy-Hill, Edward K. Smith, Caitlin Sadowski, Ciera Jaspan, Collin Winter, Matthew Jorde, Andrea Knight, Andrew Trenk, and Steve Gross. 2019. Do Developers Discover New Tools On The Toilet?. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. IEEE, New York City, NY, USA, 465–475. https://doi.org/10.1109/ICSE.2019.00059

[74] Eri n Buehler, Stacy Branham, Abdullah Ali, Jeremy J. Chang, Megan Kelly Hofmann, Amy Hurst, and Shaun K. Kane. 2015. Sharing is caring: Assistive technology designs on thingiverse. In *CHI 2015 - Proceedings of the 33rd Annual CHI Conference on Human Factors in Computing Systems* (NY, USA) *(Conference on Human Factors in Computing Systems - Proceedings)*. Association for Computing Machinery, NY, USA, 525–534. https://doi.org/10.1145/2702123.2702525 Publisher Copyright: © Copyright 2015 ACM.; 33rd Annual CHI Conference on Human Factors in Computing Systems, CHI 2015 ; Conference date: 18-04-2015 Through 23-04-2015.

[75] Del Nagy, Areej M. Yassin, and Anol Bhattacherjee. 2010. Organizational adoption of open source software: barriers and remedies. *Commun. ACM* 53, 3 (mar 2010), 148–151. https://doi.org/10.1145/1666420.1666457

[76] U.S Department of Justice Civil Rights Division. 2024. The Americans with Disabilities Act. https://www.ada.gov/

[77] Presidency of the Republic of Brazil. 1991. L8213consol. https://www.planalto.gov.br/ccivil_03/leis/l8213cons.htm

[78] Maulishree Pandey, Vaishnav Kameswaran, Hrishikesh V. Rao, Sile O'Modhrain, and Steve Oney. 2021. Understanding Accessibility and Collaboration in Programming for People with Visual Impairments. *Proceedings of the ACM on Human-Computer Interaction* 5, CSCW1 (April 2021), 129:1–129:30. https://doi.org/10.1145/3449203

[79] Jaihyun Park, JooYoung Seo, and Jae Lee. 2023. Exploring an Online Community of Blind Programmers by Using Topic Modeling and Network Analysis. *Proceedings of the Association for Information Science and Technology* 60 (10 2023), 1096–1098. https://doi.org/10.1002/pra2.956

[80] THE EUROPEAN PARLIAMENT and THE COUNCIL OF THE EUROPEAN UNION. 2019. Directive (EU) 2019/882 of the European Parliament and of the Council of 17 April 2019 on the accessibility requirements for products and services (Text with EEA relevance). http://data.europa.eu/eli/dir/2019/882/oj/eng

[81] Marian Petre and André van der Hoek. 2016. *Software Design Decoded: 66 Ways Experts Think*. MIT Press, Cambridge, MA, USA. Google-Books-ID: EVE4DQAAQBAJ.

[82] Venkatesh Potluri, Maulishree Pandey, Andrew Begel, Michael Barnett, and Scott Reitherman. 2022. CodeWalk: Facilitating Shared Awareness in Mixed-Ability Collaborative Software Development. In *Proceedings of the 24th International ACM SIGACCESS Conference on Computers and Accessibility* (Athens, Greece) *(ASSETS '22)*. Association for Computing Machinery, New York, NY, USA, Article 20, 16 pages. https://doi.org/10.1145/3517428.3544812

[83] Venkatesh Potluri, Priyan Vaithilingam, Suresh Iyengar, Y. Vidya, Manohar Swaminathan, and Gopal Srinivasa. 2018. CodeTalk: Improving Programming Environment Accessibility for Visually Impaired Developers. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2018-04-21) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3173574.3174192

[84] Rudolf Ramler and Werner Putschögl. 2013. A Retrospection on Building a Custom Tool for Automated System Testing. In *Proceedings of the 2013 IEEE 37th Annual Computer Software and Applications Conference (COMPSAC '13)*. IEEE Computer Society, USA, 820–821. https://doi.org/10.1109/COMPSAC.2013.135

[85] Eric Raymond. 1999. The Cathedral and the Bazaar. *Knowledge, Technology & Policy* 12, 3 (1999), 23–49. https://doi.org/10.1007/s12130-999-1026-0

[86] Joseph Reagle. 2019. *Chapter 1: Introduction* (1 ed.). MIT, Cambridge, MA, USA. https://hackinglife.mitpress.mit.edu/pub/lq22zt1x.

[87] N.C. Romano and J.F. Nunamaker. 2001. Meeting analysis: findings from research and practice. In *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*. IEEE, Maui, HI, USA, 13 pp.–. https://doi.org/10.1109/HICSS.2001.926253

[88] Clark Saben and Prashant Chandrasekar. 2024. Enabling BLV Developers with LLM-driven Code Debugging. https://doi.org/10.48550/arXiv.2401.16654 arXiv:2401.16654 [cs]

[89] Harini Sampath, Alice Merrick, and Andrew Macvean. 2021. Accessibility of Command Line Interfaces. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 489, 10 pages. https://doi.org/10.1145/3411764.3445544

[90] Lisa Schur, Douglas Kruse, and Peter Blanck. 2005. Corporate culture and the employment of persons with disabilities. *Behavioral Sciences & the Law* 23, 1 (2005), 3–20. https://doi.org/10.1002/bsl.624 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/bsl.624

[91] Freedom Scientific. 2024. JAWS® – Freedom Scientific. https://www.freedomscientific.com/products/software/jaws/

[92] Abhishek Sharma, Yuan Tian, Agus Sulistya, Dinusha Wijedasa, and David Lo. 2018. Recommending Who to Follow in the Software Engineering Twitter

[93] Space. *ACM Trans. Softw. Eng. Methodol.* 27, 4, Article 16 (oct 2018), 33 pages. https://doi.org/10.1145/3266426

[93] Naomichi Shimada, Tao Xiao, Hideaki Hata, Christoph Treude, and Kenichi Matsumoto. 2022. GitHub Sponsors: Exploring a New Way to Contribute to Open Source. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE) (ICSE '22)*. Association for Computing Machinery, New York, NY, USA, 1058–1069. https://api.semanticscholar.org/CorpusID:246822794

[94] Edward K. Smith, Christian Bird, and Thomas Zimmermann. 2015. Build It Yourself! Homegrown Tools in a Large Software Company. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, Vol. 1. IEEE, NY, USA, 369–379. https://doi.org/10.1109/ICSE.2015.56

[95] Diomidis Spinellis and Vaggelis Giannikas. 2012. Organizational adoption of open source software. *Journal of Systems and Software* 85, 3 (2012), 666–682. https://doi.org/10.1016/j.jss.2011.09.037 Novel approaches in the design and implementation of systems/software architecture.

[96] StackOverflow. 2024. Stack Overflow Insights - Developer Hiring, Marketing, and User Research. https://survey.stackoverflow.co/

[97] Scott Stein. 2022. *Leadership Hacks: Clever Shortcuts to Boost Your Impact and Results*. John Wiley & Sons, Queensland, Australia.

[98] Maria Stone, Frank Bentley, Brooke White, and Mike Shebanek. 2016. Embedding User Understanding in the Corporate Culture: UX Research and Accessibility at Yahoo. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI EA '16)*. Association for Computing Machinery, New York, NY, USA, 823–832. https://doi.org/10.1145/2851581.2851592

[99] Viktoria Stray and Nils Brede Moe. 2020. Understanding coordination in global software engineering: A mixed-methods study on the use of meetings and Slack. *Journal of Systems and Software* 170 (Dec. 2020), 110717. https://doi.org/10.1016/j.jss.2020.110717

[100] Kayla S Sweet, Jennifer K LeBlanc, Laura M Stough, and Noelle W Sweany. 2020. Community building and knowledge sharing by individuals with disabilities using social media. *Journal of computer assisted learning* 36, 1 (2020), 1–11.

[101] Theresa Jean Tanenbaum, Amanda M. Williams, Audrey Desjardins, and Karen Tanenbaum. 2013. Democratizing technology: pleasure, utility and expressiveness in DIY and maker practice. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) *(CHI '13)*. Association for Computing Machinery, New York, NY, USA, 2603–2612. https://doi.org/10.1145/2470654.2481360

[102] Ruth Umoh. 2024. Fortune 500 diversity data reveals DEI trends. https://fortune.com/2024/03/13/fortune-500-diversity-data-promotions-representation-legal/

[103] Bogdan Vasilescu, Daryl Posnett, Baishakhi Ray, Mark G.J. van den Brand, Alexander Serebrenik, Premkumar Devanbu, and Vladimir Filkov. 2015. Gender and Tenure Diversity in GitHub Teams. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 3789–3798. https://doi.org/10.1145/2702123.2702549

[104] Bogdan Vasilescu, Alexander Serebrenik, Prem Devanbu, and Vladimir Filkov. 2014. How social Q&A sites are changing knowledge sharing in open source software communities. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing* (Baltimore, Maryland, USA) *(CSCW '14)*. Association for Computing Machinery, New York, NY, USA, 342–354. https://doi.org/10.1145/2531602.2531659

[105] Georg Von Krogh, Stefan Haefliger, Sebastian Spaeth, and Martin W Wallin. 2012. Carrots and rainbows: Motivation and social practice in open source software development. *MIS quarterly* 36, 2 (2012), 649–676.

[106] W3C. 2024. Web Content Accessibility Guidelines: Conformance Resources. https://www.wcag.com/

[107] Tricia Wang and Joseph 'Jofish' Kaye. 2011. Inventive leisure practices: understanding hacking communities as sites of sharing and innovation. In *CHI '11 Extended Abstracts on Human Factors in Computing Systems* (Vancouver, BC, Canada) *(CHI EA '11)*. Association for Computing Machinery, New York, NY, USA, 263–272. https://doi.org/10.1145/1979742.1979615

[108] Nathan Wintersgill, Trevor Stalnaker, Laura A. Heymann, Oscar Chaparro, and Denys Poshyvanyk. 2024. "The Law Doesn't Work Like a Computer": Exploring Software Licensing Issues Faced by Legal Practitioners. *Proc. ACM Softw. Eng.* 1, FSE, Article 40 (jul 2024), 24 pages. https://doi.org/10.1145/3643766

[109] Marisol Wong-Villacres, Carl DiSalvo, Neha Kumar, and Betsy DiSalvo. 2020. Culture in Action: Unpacking Capacities to Inform Assets-Based Design. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3313831.3376329