How an Academic Companion Website Makes Media-Specific Arguments

Hannah Ackermans

Abstract

In this digital project review, I discuss the companion website CriticalCodeStudies.com in relation to Mark C. Marino's book *Critical Code Studies* (2020). Over the past decades, companion websites have become a small but persistently growing genre in academe, with products ranging from paratextual records to publications in their own right. The *Critical Code Studies* companion website makes excellent use of content and design to make media-specific arguments that interrogate the research subject, foregrounding a method that oscillates between close reading and contextual reading as well as promotes personal and communal reading practices. The combination of book and companion website successfully makes intellectual interventions not only into the case studies but also into our conception of source code in general. I review how the companion website reflects, amplifies, and contradicts the arguments made in the book.

Differences in media materialities produce different content and meaning. N. Katherine Hayles's concept of "media-specific analysis" has become a staple in the study of creative digital artifacts, which she posits to "explore the dynamic interaction between the artifactual characteristics and the interpretation that materiality embodies."[1] Hayles conceives of media-specific analysis to interrogate the materiality of literary hypertext. Its print and digital manifestations make it ideal for her to analyze the specificity of the electronic hypertext.

Although increasingly commonplace in the study of creative works, a parallel for the study of multimodal academic texts is still in the evolution phase. Comparable to hypertext's print and digital manifestations, print books can have digital companion websites as counterparts. Unlike hypertexts, these print and digital counterparts usually complement each other in one project. In reviewing academic texts, we take media-specific analysis to a higher level by investigating the effects of media-specificity on the intellectual intervention of the book project. Just as Hayles argues that part of the meaning of a hypertext is in the links between textual nodes, so can a companion website make arguments by its content as well as its structure. For this review, then, my overarching interrogation is, how does the companion website reflect, amplify, and contradict the arguments made in the book?

Book

With his conception of critical code studies (CCS), Marino aims to persuade people of the significance of source code within cultural analysis. His 2006 essay "Critical Code Studies" in *Electronic Book Review* started the development of CCS in a reflective, theoretically grounded manner.[2] Since then, the field has grown, with more applied publications such as *Reading Project: A Collaborative Analysis of William Poundstone's Project for Tachistoscope {Bottomless Pit}*.[3] Marino and Jeremy Douglass have also organized biennial Critical Code Studies Working Groups since 2010. Through various themes, participants have collaborated on analyzing various case studies. This has led, for example, to the book *10 PRINT CHR$ (205.5+RND(1)); : GOTO 10*,[4] in which ten researchers analyze the same one-line Commodore 64 BASIC program.

Now there is *Critical Code Studies*.[5] Despite having fifteen years of publications before it, this book is likely to become the main source of reference for the field for its successful combination of a proclamation of the importance of source code and an application of analysis to the source code. The first chapter introduces small case studies, familiarizing the reader with code as well as contextual sources. The introduction creates a feeling of excitement. Like a treasure hunt, it motivates the reader to delve deeper into the code to find all the hidden gems within. Chapter 2 is a revised version of the initial 2006 essay, now with the new subtitle “a manifesto.” This manifesto contains surprisingly little source code; rather, it engages with prior research on source code to claim its position as a field. In the rest of the chapters, this changes significantly. Through more case studies, Marino demonstrates the creativities and ambiguities present in source code. Chapter 3 analyzes the hacktivist art project Transborder Immigrant Tool as a simulation of ideologies through code. Code, then, is immediately positioned as a political signifier. Chapter 4 reflects on the so-called Climategate, in which a piece of leaked code was used as supposed evidence of a climate change hoax. This chapter almost reads like a detective novel, as the twists and turns of the analysis of the code and surrounding sources uncover the true story around “Climategate.” Chapter 5 turns to FLOW-MATIC to highlight the connection and tension between coding language and natural language. Although every chapter takes an engaged approach, this chapter’s perspectives especially demonstrate a sensitivity to the nuances of language in relation to gender, colonialism, and labor. In chapter 6, Friedrich Kittler’s code is shown to exemplify his theoretical positions. A true humanities chapter, this analysis demonstrates the relation between theory and artistic practice consolidated in code. Finally, chapter 7 considers generative code by analyzing Nick Montfort’s computer poem *Taroko Gorge* in relation to its remixes by various authors. This chapter expresses all Marino’s excitement over remix practices and contributory creative work. Finally, after the case study chapters, the conclusions take a bird’s-eye perspective of critical code studies, outlining its potential venues, developments, and futures.

There are not that many books about source code that are a joy to read, until now. In *Critical Code Studies*, Marino combines academic rigor with vivid storytelling. Regardless of prior knowledge of code, this book has something to offer to all readers.

Website

Over the past decades, there has been a growing practice to create companion websites by well-known academic publishers such as Sage and MIT Press. Some individual researchers also provide extra resources for their books without calling them companion websites. It is unsurprising, then, that this translates into different approaches to companion websites. These websites can be born-digital academic publications themselves, such as *Structure and Interpretation of Computer Programs*.[6] This early example from the nineties contains the entire book as well as code, instructor’s manual, and sample assignments. Some other companion websites might be best described as a paratextual record, such as the companion site to *Software Design Decoded*,[7] which contains an annotated bibliography and various sorts of promotional material. Timothy J. Miles-Board et al. review companion websites existing at the time of the article's publication,[8] based on their textual relation to the book, hypertext features, student resources, and instructor resources. Companion websites accompany print publications of various sorts. The term *companion website* covers a broad range of genres, referring to newspaper and magazine websites,[9] as well as websites of books, from fiction to scholarly monographs and from edited collections to student textbooks.

Although companion websites exist for books across disciplines, Marino's born-digital subject adds an extra urgency to the development of a companion site. One might even wonder why a project like this is a print book at all, instead of an elaborate digital publication. Yet print books hold prestige, especially in the humanities, which validates critical code studies in current academic structures even if there were no practical benefits. Reading a print book also provides a set of conventions that invites people to read with a particular focus,[10] and the combination of reading the print book beside the companion website without having to toggle between windows works smoothly and effectively. This combination of print and digital publication offers a good infrastructure to invite people into the field.

The book itself contains large sections of source code, between two and twenty pages, at the start of each case study chapter. The companion website has a section of all the source code in the book as embedded view from Marino's GitHub account. Organized by chapter, the source code is easy to find for book readers who wish to read the book alongside the digital code. Although the case study approach might feel like a type of canonization at first glance, Marino argues throughout the book that *any* source code can be read for cultural significance. Marino even reminds readers about his stance in the source code of the companion website:

> <!-- Yes, just as you expected, there are messages hidden within the code of this page. However, Critical Code Studies is not just a hunt for Easter Eggs. It is instead predicated on the belief that code is a meaningful medium of communciation [*sic*], different from other sign systems because of its status on machines. You can read this note, but there is meaningful [*sic*] in even the code you think is not meant to address you.[11]

In addition to source code of the case studies itself, each page includes some metadata, such as authorship and publication year, and links to interesting, relevant hyperlinks to webpages and documents relating to the code.

When you think of source code, chances are you consider it in terms of practical execution, not as texts to actually *read*. Fortunately, the combination of print book and website facilitates this shift. The inclusion of code as a text to read in the book uses familiarity of text that is meant to be read, while the code on the websites uses convention of code in the editor, adding the layer of making the code more alive, something readers can tinker with. The code text is text; however, it is not text like any other, and the book's and the website's presentations of code reflect that. The book uses the Courier New font for cited source code, and the website presents the source code in embedded view. Through typography and platform, readers are aware that code is readable yet different from natural language. This reflects arguments made most explicitly in chapter 3 on FLOW-MATIC, which resembles English, but "despite the familiar appearance of the tokens, programmers must still learn unique, unambiguous references, as they would with any other tokens."[12] Although one of the more simple source codes, FLOW-MATIC became possibly my favorite chapter for its deep dive into the readability of code, a topic already on readers' minds, especially if they read the book alongside the website.

At the same time, this does not diminish the prominence of contextual reading. The book cites Barbara Marino: "Reading requires taking something's meaning in context."[13] Here the relationship between book argument and website argument becomes more complex. The code itself is not a stand-alone source, as it also contains Mark Marino's notes for various lines. Although they could be more integrated on the website (rather than as footnotes just as in the book), they provide something to hold on to for the reader just starting out. The webpages of individual chapters contain relevant sources that Marino often also uses in the book, but there is no context for these sources; the reader can interpret them in relation to the source. Listing the sources legitimizes their value even if the book actively disproves them. This brings up foundational questions on the definition and service of the companion site. Literally defined as companion, the author might be justified to assume that every visitor to the website has the book. At the same time, the companion site does not place the chapter pages in a central place on the website. As the site encompasses other elements of critical code studies as well, we assume that people come across the chapters' pages out of a general interest in critical code studies. Combined with the notion that websites make media-specific arguments, it is necessary to be critical of how code is presented.

For example, the first source code that Marino analyzes in his introduction is a piece of code (or codework) on a protest sign by a young Indian woman that has an algorithm stating that protests will continue until the Jan Lokpal Bill is passed, which will lead to a "corruption free India" (fig. 1). In addition to analyzing the code on the sign itself, Marino give a contextual analysis that includes the comments on the photo on the social platform Reddit. Redditors comment on the photo by critiquing the code's technical validity in a disdainful manner that often includes sexist and racist remarks.

[Insert fig. 1]

Marino pushes back against this line of criticism that distracts from the message of the protester, and he names their comments "encoded chauvinism." On the companion website, readers can find a fragment of the picture of the woman with the sign in the header "protest code" before clicking on the page (fig. 2). Once on the "protest sign" page, it provides only a raw view version of the code (fig. 3), just like all the other case studies in the book. In this particular case, however, that approach inadvertently sends the wrong message. Providing the protest sign *as code* that can be copy-pasted into an editor legitimizes syntax critique without the nuanced discussion in the book. Of course, displaying only the picture would fall short, as it would be inaccessible to blind people. I would suggest changing this page to include the full photograph with an image description that includes the text on the sign.

[Insert figures 2, 3]

We need to think through these subtleties to ensure that the book and companion site are well aligned. This particular example seems corrective, but we can harness its potential in a more speculative framework to further the objective of the website, both as a full companion to the book and as an exposition of the field. Marino takes different approaches in each chapter, almost as if they were different genres. The companion website, on the other hand, takes a standard approach to all chapters, providing contextual links, metadata, source code, and footnotes on lines of source code. This is useful to the reader, but how can we reimagine the different types of genres into the presentation of the source code? After positioning

FLOW-MATIC as an operation similar to present-day Microsoft Excel, Marino criticizes the "hierarchy in an economy in which certain programming skills are valued more than others and in which the more priestly class of programmers maintain a degree of status that those working in data-entry environments, such as Excel, even while programming them, do not."[14] To incorporate this argument on the companion website, it would be interesting if, for example, the actual FLOW-MATIC code was accompanied by an embedded (or hyperlinked) view of the same operations as formulas in Microsoft Excel / Google Sheets cells. Additionally, this would give readers different ways to familiarize themselves with code. In thinking through each chapter from this perspective, the companion website itself could be used as a space for more speculative, generative praxis.

Beyond accompanying the book, the website also becomes a website for the entire field. Marino's approach puts into practice what he argues in the book: CCS is a field for amateurs and professionals of many different fields, as long as they are interested in the cultural reading of source code. In doing so, he provides solutions to the pleas of such scholars as Kathleen Fitzpatrick: "If scholars hope to promote a vision of scholarly work, even at its most critical, as being rooted in and working toward the public good, we need to find ways to engage broader publics with that work."[15] The website is a place for action more than history. Marino set up the website as a platform for a community of people interested in analyzing code. This performative act both shows the scope of the field and builds this field further on the platform. Through the Tools page, Marino literally provides people with the tools to create a personal reading practice when it comes to reading source code. At the same time, the working groups facilitate communal reading practices. The website links to the biennial online working groups of CCS that Marino organizes with Douglass. Readers of the book and companion site who are new to the field can take a look at all the different practices by past participants of the working groups. This open social scholarship approach sets up the site as a conversation about code, highlighting the layers and ambiguity of any piece of code. The working groups include personal readings of participants but add up to a greater role of generative interaction between participants.

Rather than a one-time publication, the CCS companion website is an ongoing project that Marino plans to keep updating. Updatability of websites is a major strength of the companion website compared with the print publication. Some companion websites are hosted by publishers, whereas others are stand-alone websites, which gives authors more freedom to develop the content further. The other side of the coin of this freedom and updatability is the labor involved in doing so. The website requires the author to spend time creating and updating the companion website, generally without support for editing and proofreading that are customary in the print book. Depending on the author's institutional support and position, this may or may not be feasible. Doing this "free labor" is at once expected and denied official value.[16] Companion websites, then, can have various roles in shaping a book project, in terms of content and readership as well as timeline of engagement that is expected from the author.

Concluding Remarks

The companion website of *Critical Code Studies* is a necessary and productive addition to the book. The simple WordPress design manages to be inviting to readers, and the content reflects many of the book's arguments. Through the provided data, the website encourages close readings and contextual readings. In some cases, as I pointed out, the website inadvertently contradicts some of the arguments in its design. A rethinking of the position of the code on the website could provide possibilities to present code in new and interesting ways that align with the role of code in the book project. Through the tools and working group pages, the website facilitates both personal and communal readings. An overarching intellectual intervention, then, is the guiding of readers' attitudes and approaches to source code. This is an essential practice to further the field of critical code studies as well as one way to connect the academy to the public good. At the same time, these novel approaches need to be continuously scrutinized as an academic practice to ensure both its internal structure and its external labor are accessible and inclusive. A review like this is necessarily based on a snapshot of the *Critical Code Studies* companion website, as it is in continuous development. This media-specific project review highlights the companion website as an intellectual intervention through structure and content with potential for growth.

Notes

I would like to thank Inge van de Ven at Tilburg University for her feedback on this review.

1. N. Katherine Hayles, "Print Is Flat, Code Is Deep: The Importance of Media-Specific Analysis," *Poetics Today* 25.1 (2004): 67–90.
2. Mark C. Marino, "Critical Code Studies," *Electronic Book Review* 4 (2006), electronicbookreview.com/essay/critical-code-studies.
3. Jessica Pressman, Mark C. Marino, and Jeremy Douglass, *Reading Project: A Collaborative Analysis of William Poundstone's Project for Tachistoscope {Bottomless Pit}* (Iowa City: University of Iowa Press, 2015).

4. Nick Montfort, Patsy Baudoin, John Bell, Ian Bogost, Jeremy Douglass, Mark C. Marino, Michael Mateas, Casey Reas, Mark Sample, and Noah Vawter, *10 PRINT CHR$ (205.5+ RND(1)); : GOTO 10* (Cambridge, MA: MIT Press, 2012).
5. Mark C. Marino, *Critical Code Studies* (Cambridge, MA: MIT Press, 2020).
6. Harold Abelson and Gerald Jay Sussman, "Structure and Interpretation of Computer Programs (Companion Website)," accessed April 24, 2020, mitpress.mit.edu/sites/default/files/sicp/index.html.
7. Marian Petre, André Van der Hoek, and Yen Quach, "Software Design Decoded (Companion Website)," accessed April 24, 2020, softwaredesigndecoded.wordpress.com/.
8. Timothy J. Miles-Board, Christopher P. Bailey, Wendy Hall, and Leslie A. Carr, "Building a Companion Website in the Semantic Web," *Proceedings of the Thirteenth International Conference on World Wide Web*, May 2004, 365–73.
9. Ulrich Kaiser and Hans Christian Kongsted, "Magazine 'Companion Websites' and the Demand for Newsstand Sales and Subscriptions," *Journal of Media Economics* 25.4 (2012): 184–97, doi.org/10.1080/08997764.2012.729545.
10. Anne Mangen and Adriaan Van der Weel, "The Evolution of Reading in the Age of Digitisation: An Integrative Framework for Reading Research," *Literacy* 50.3 (2016): 116–24.
11. Mark Marino, "Critical Code Studies," n.d., accessed April 30, 2020, criticalcodestudies.com/.
12. Marino, *Critical Code Studies*, 139.
13. Marino, 145.
14. Marino, 133–34.
15. Kathleen Fitzpatrick, *Generous Thinking: A Radical Approach to Saving the University* (Baltimore: Johns Hopkins University Press, 2019).
16. Tiziana Terranova, "Free Labor," *Digital Labor: The Internet as Playground and Factory* (2012): 41–65.