you.can?(:read)

Joris J. van Zundert
Huygens Institute for the History of the Netherlands
Royal Netherlands Academy of Arts and Sciences
joris.van.zundert@huygens.knaw.nl

In his seminal 1959 lecture "The Two Cultures" C.P. Snow characterized the distance between scientific and humanistic understanding by considering what would happen if he asked a simple question: "What do you mean by mass, or acceleration"? Snow regarded this as "the scientific equivalent of saying, Can you read?" (Snow 1998:87; original emphasis), and he contended that "not more than one in ten of the highly educated would have felt that I was speaking the same language".

I think it is urgent that we humanities scholars ask ourselves a similar question with regard to the ability to read, understand, and critique software code. Given a few decades of softwarization of society and scholarship (Hockey 2004, Zundert and Haentjens Dekker 2015) code illiteracy is a barrier to deep examination and questioning of the software that is shaping culture and society (Jones 2014, Berry 2014). But arguably fewer than one in fifty humanities researchers would be able to answer the question "What do you mean by a closure, or a class object?", which is the programming equivalent of saying "Can you read?"

Currently the skill of digital reading and writing is far from evenly distributed over the scientific community, which creates an inequality in analytical and critical capabilities. I shall argue that this skill is not—as is often assumed—a mere methodological service that can be obtained through the involvement of programmers in humanities research projects. The inability to read and write digitally should be compared rather to the inability to read and write one's natural language, in other words to illiteracy. This new illiteracy impairs the ability of scholars to communicate on an equal footing with the creators of digital objects and those users for whom it is first and second nature to use such objects that produce reality in present day society, culture, scholarship, and science.

In my paper I shall argue that we need a New Literacy that creates affordances for humanities researchers to understand and critique digital objects. This New Literacy departs from Kittler's understanding that there is no such thing as software, only the continuation of writing with different means (Kittler 1993). New Literacy is in this sense different than the more usual understandings of new literacies, which amount to the ability to negotiate and utilize digital objects such as graphical interfaces and their affordances that arise from the execution of software (cf. Coiro et al. 2008). In my paper I will explain that New Literacy is also different from coding to produce digital objects as scholarly output (such as via markup or code, cf. e.g. Burnard 2014), and from analyzing such objects through reductive empirical means (as in e.g. Goldstone and Underwood 2012). I currently understand this New Literacy as the ability to hermeneutically read and write digital objects through the application of code.

My argument however does not result in the contention that all humanists should become programmers. Rather, I argue for a series of experiments to take digital reading and writing closer to a form that answers to the hermeneutic needs of humanities scholars—thus exploring a New Literacy that will allow humanists to conduct scholarship in a softwarized society and culture. Donald Knuth provided a model for such experimentation through his concept and implementation of 'literate programming' (Knuth 1984). Knuth argued that code should be considered as works of literature; in a similar fashion Mark Marino argued that code should be read and explicated as text (Marino 2006), and Florian Cramer argued that code is "loaded with meaning" (Cramer 2005). Knuth and Marino thus argue that we should understand code-as-text just as we understand text-as-text. This amounts to the application of current scholarly literacy to code. I contend that an inverse direction of understanding is needed for New Literacy: as scholars we should also be able to understand code-through-code and text-through-code.

In the second half of my presentation I will unpack and explain a first experiment towards exploring the possible forms of this New Literacy. In this particular experiment Ruby programming language is used to read text through code. I try to read and understand a (small fragment) of a literary work through a meticulous object oriented modeling of the text. The modeling encompasses both semantic, temporal, and narrative elements, and the relations between them as perceived by me, the scholar. Three ground rules guide the modeling. The first is that only direct and indirect speech may be represented as string instances. The second rule demands that there be no 'ghost' objects or methods (unused program code) during the execution of the program. The third rule stipulates that the resulting program should execute without producing Ruby exceptions or runtime errors. The code that results is a new form of close reading and scholarly criticism through code, which is reproducible and open to a computational discursive. The experiment serves thus to demonstrate how code can be used to read text in a hermeneutic and scholarly fashion. I will argue that this approach is complementary to the widely applied empirical approaches that pioneered distant reading, examples of which are found in Goldstone and Underwood 2012 and Rybicki, Hoover and Kestemont 2014.

I will conclude my paper by positing that only a New Literacy will allow us to advance scholarly method beyond a mode of applying interactional expertise (Gorman 2010) to the investigation of any digital object, the understanding of which requires a tacit ability to read and write digitally.

References

Berry, D. M. (2014) *Critical Theory and the Digital*, English, Critical Theory and Contemporary Society, New York, London, New Delhi etc.: Bloomsbury Academic.

Burnard, L. (2014) *What is the Text Encoding Initiative?*, English, Marseille: OpenEdition Press, URL: http://books.openedition.org/oep/426 (visited on 12/02/2016).

Coiro, J. et al., eds. (2008) *Handbook of Research on New Literacies*, English, New York, London: Taylor & Francis Group, LLC.

Cramer, F. (2005) *Words Made Flesh: Code, Culture, Imagination*, English, Rotterdam: Piet Zwart Institute, URL: http://cramer.pleintekst.nl/00-recent/words_made_flesh/html/words_made_flesh.html (visited on 12/02/2016).

Goldstone, A. and Underwood, T. (2012) 'What Can Topic Models of PMLA Teach Us About the History of Literary Scholarship?', English, *Journal of Digital Humanities* 2.1, URL: http://journalofdigitalhumanities.org/2-1/what-can-topic-models-of-pmla-teach-us-by-ted-underwood-and-andrew-goldstone/ (visited on 12/02/2016).

Gorman, M. E., ed. (2010) *Trading Zones and Interactional Expertise*, Cambridge Mass.: MIT Press.

Hockey, S. (2004) 'The History of Humanities Computing', English, *A Companion to Digital Humanities*, ed. by S. Schreibman, R. Siemens and J. Unsworth, Oxford, UK: Blackwell, URL: http://www.digitalhumanities.org/companion/.

Jones, S. E. (Aug. 2014) *The Emergence of the Digital Humanities*, English, New York, London: Routledge.

Kittler, F. (1993) ‘Es gibt keine Software’, Deutsch, *Draculas Vermächtmis*, Leipzig: Reclam Verlag, pp. 225–242.

Knuth, D. E. (1984) ‘Literate Programming’, English, *The Computer Journal* 27.1, pp. 97–111, URL: http://comjnl.oxfordjournals.org/content/27/2/97.full.pdf+html.

Marino, M. C. (Dec. 2006) ‘Critical Code Studies’, English, *Electronic Book Review*, URL: http://www.electronicbookreview.com/thread/electropoetics/codology (visited on 16/01/2015).

Rybicki, J., Hoover, D. and Kestemont, M. (2014) ‘Collaborative authorship: Conrad, Ford and Rolling Delta’, English, *Literary and Linguistic Computing* 29.3, pp. 422–431, URL: http://llc.oxfordjournals.org/content/29/3/422.abstract (visited on 12/02/2016).

Snow, C. (1998) *The Two Cultures*, English, Part 1 first published 1959, Cambridge, New York, etc.: Cambridge University Press.

Zundert, J. J. v. and Haentjens Dekker, R. (2015) ‘Code, Scholarship, and Criticism: When is Coding Scholarship and When is it Not?’, English, *pre-print*, URL: http://jorisvanzundert.net/etc. (visited on 12/02/2016).