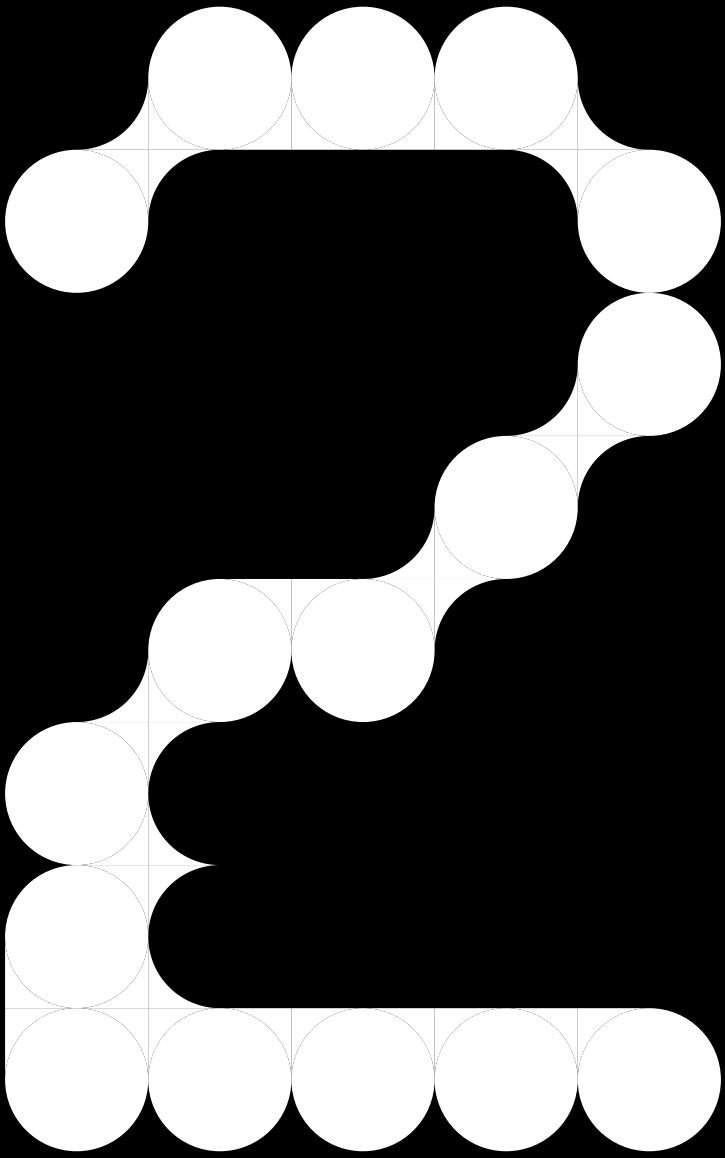
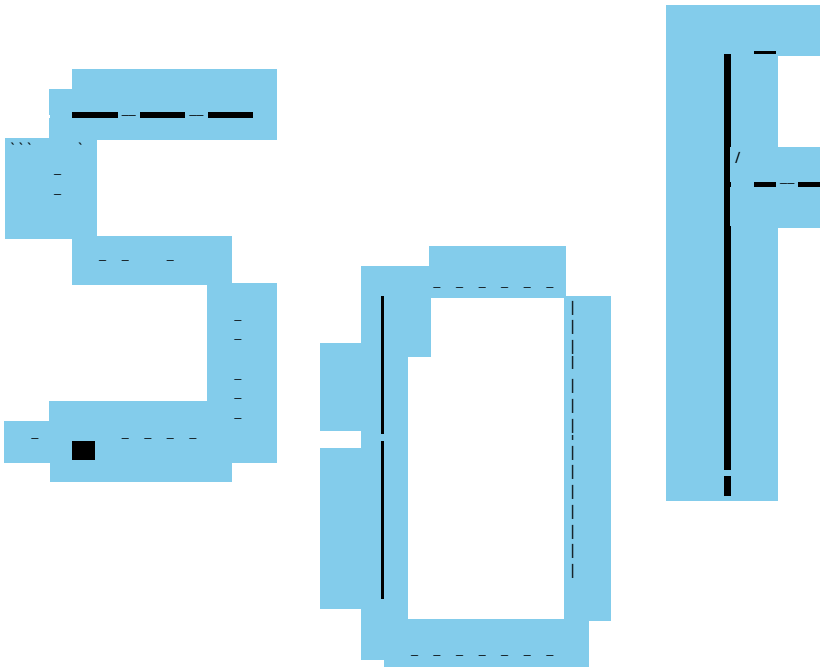


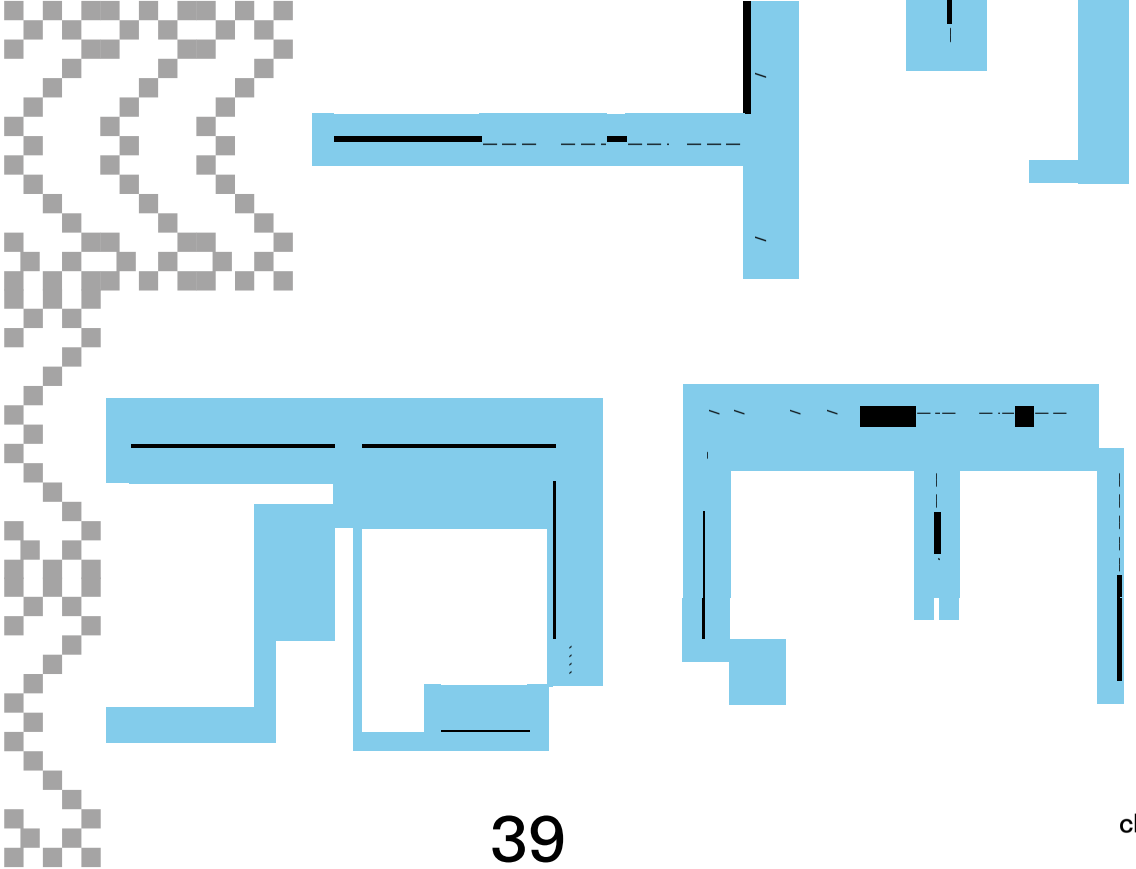
Chapter 2

Softer computing is a design attitude and practice that values friction, care, and participatory agency over seamlessness, automation, and invisible control. Unlike open source or DIY, which often focus on technical transparency or independence, softer computing is fundamentally about relational engagement—between maker, tool, and context.

INTRODUCING SOFTER COMPUTING



Softer computing is not a system, nor is it a fixed set of techniques. Rather, it is an attitude—a critical, relational approach to design and technology. At its core, softer computing resists the dominant logics of efficiency, flattening, and automation that underpin today's digital landscape. It proposes an alternative value system: one that prioritizes friction, transparency, constraint, modularity, and the concept of ***"slow time"*** over seamlessness, opacity, and relentless acceleration.



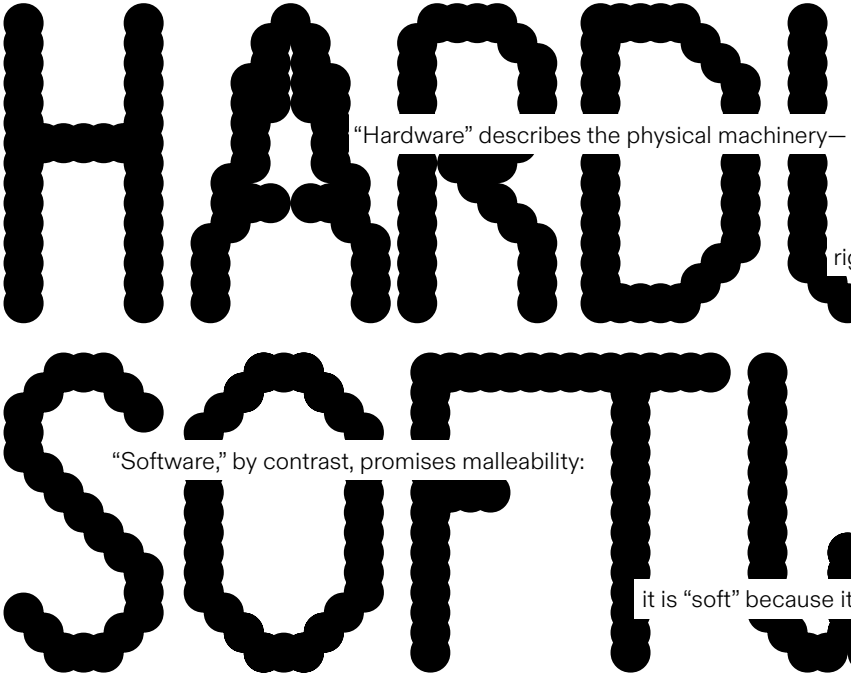
WHY NOW? SOFTER COMPUTING AS A RESPONSE TO DIGITAL MAXIMALISM

William Powers, Hamlet's BlackBerry, 2011

The need for a softer approach has never felt more urgent. The rapid proliferation of generative AI, the dominance of algorithmic curation, and the "***digital maximalism***" that Powers (2011) critiques have together created a climate of design overload. The culture of tech, shaped by capitalist imperatives of constant growth and efficiency, increasingly demands that designers and users alike submit to automated workflows, default templates, and invisible infrastructures. In response, a growing movement for "***degrowth computing***" has emerged, seeking to decouple digital technology from the "growth-focused imperatives of capitalist societies" (Neil, 2022).

Neil, What might degrowth computing look like?, 2022

For me, the very idea of "softer" computing comes from reflecting on the language of technology itself.

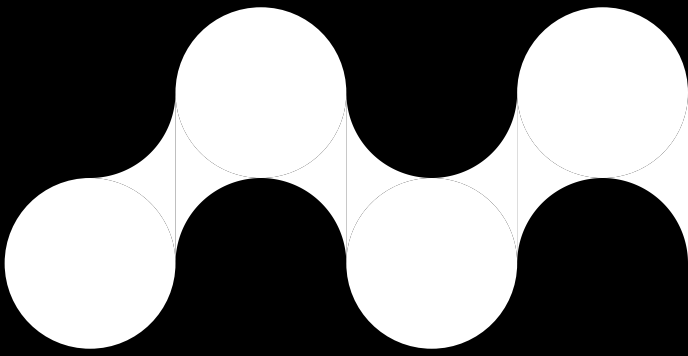


Yet this is a misleading comfort. In practice, contemporary software has become so complex, so deeply layered with proprietary code and abstracted interfaces, that even developers find it difficult to intervene. Far from soft, today's software is often rigid, inaccessible, and resistant to meaningful modification.

As a result, the notion that digital environments are open to intervention by ordinary users has become largely illusory. The smooth, seamless structures of modern interfaces conceal the underlying code, hiding opportunities for agency and making it easy to mistake constraint for choice. The GUI—a "graphical user interface"—may feel approachable, but it often functions as a form of abstraction that distances us from the real workings of technology (Emerson, 2014).

Emerson, Lori, Reading Writing Interfaces: From the Digital to the Bookbound, 2014

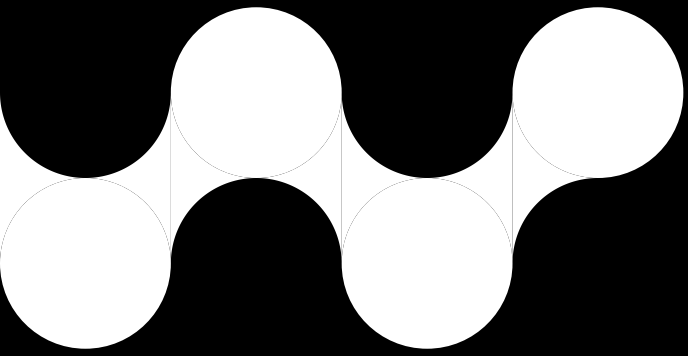
HARDWARE



Wikipedia

Computer hardware includes the physical parts of a computer, such as the central processing unit (CPU), random-access memory (RAM), motherboard, computer data storage, graphics card, sound card, and computer case. It includes external devices such as a monitor, mouse, keyboard, and speakers.

SOFTWARE



Wikipedia

Software consists of computer programs that instruct the execution of a computer. Software also includes design documents and specifications.
Software can generally be categorized into two main types:
operating systems, which manage hardware resources and provide 1. services for applications 2. application software, which performs specific tasks for users

THE DECEPTIVE PROMISE OF "SOFTNESS"

The characterization of software as "soft" obscures the reality of contemporary computing. Most people today interact with technology through platforms designed for immediate clarity and efficiency.

We are discouraged from "wasting time" on things that do not offer instant, frictionless communication. This dynamic is especially pronounced in the design of websites, which have become so streamlined and standardized that the development process itself mirrors software production: modular, automated, and largely uncustomizable for the average user. The web is no longer "soft"; it has become, as Zhang (2019) writes, a built environment and a social space that is monopolized, commodified, and colonized by corporate interests.

Gary Zhexi Zhang, Reenvisioning the Internet: Create Tools that Reveal its Ideological Infrastructures, 2019

LOST VERNACULAR WEBS AND FREEDOM

Gary Zhexi Zhang, Reenvisioning the Internet: Create Tools that Reveal its Ideological Infrastructures, 2019

Looking back, the amateur web of the mid-1990s, as described by Olia Lialina (2015), offers a powerful contrast to the present. This was a time when the web was unregulated, personal, and always "under construction." The web, in Lialina's terms, was communal: every page was an act of care, each contribution a unique addition to a shared space. Lialina calls this the "vernacular web," a term that may seem strange now but precisely captures the ethos of the era—a web made by its users, not just for them.

GeoCities, founded in 1994, stands as a testament to this spirit of digital amateurism. At its peak, it hosted 38 million personal websites—spaces for self-representation and connection, unpolished but fiercely individual. While GeoCities is now gone, this ethos survives in platforms like Neocities, which archives and extends the legacy of amateur web-building. In just the last few years, Neocities has grown to host nearly a million sites, providing free space for people to make the web their own once again.