THE PSYCHOLOGY OF THE WEB DEVELOPER, REALITY OF A FEMALE FREELANCER

Maisa Imamović

DEEP POCKETS

DeepPockets #4
The Psychology of the Web Developer, Reality of a Female Freelancer
Author: Maisa Imamović

Editor and Proofreader: Laurence Scherz
Cover design: James P.A. Crossley
Design and E-Pub development: Sepp Eckenhaussen, Maria van der Togt

Printer: Lulu
Published by the Institute of Network Cultures, Amsterdam, 2022
ISBN 9789492302885

Contact Institute of Network Cultures
Email: info@networkcultures.org
Web: http://www.networkcultures.org

This publication may be ordered through various print-on-demand-services or freely downloaded from http://www.networkcultures.org/publications.

The zeros and ones of machine code seem to offer themselves as perfect symbols of the orders of Western reality, the ancient logical codes which make the difference between on and off, right and left, light and dark, form and matter, mind and body, white and black, good and evil, right and wrong, life and death, something and nothing, this and that, here and there, inside and out, active and passive, true and false, yes and no, sanity and madness, health and sickness, up and down, sense and nonsense, west and east, north and south. And they made a lovely couple when it came to sex. Man and woman, male and female, masculine and feminine: one and zero looked just right, made for each other: 1, the definite, upright line; and 0, the diagram of nothing at all: penis and vagina, thing and hole. . . hand in glove. A perfect match. — Sadie Plant, *Zeros and Ones*

Please note: unless otherwise specified, he is persistently used throughout the book to represent the stereotype of a web developer, but not the figure's essence.

Spleak

At a certain time, during a specific state of the internet, time, and language, I was ten years old. I was growing up in Kosovo, a country where I moved to with my sister and my parents in 2001—two years after the Kosovo war. I moved from Bosnia and Herzegovina where I was born in 1994, during the Bosnian war.

Moving from one post-war country to another did not help my integration. Speaking Bosnian (a language slightly different from Serbian) in Kosovo at that time made making friends challenging. The war traumas were fresh, the tensions were still being negotiated. I was shy, rather afraid to speak. Any Serbo-Croatian-Bosnian-etc. speaking citizen would either be socially excluded by default, or further questioned about their true origins.

While the social status of my parents did not depend on learning the Albanian language in order to find institutional belonging, for my sister and I, it was rather important. With the help of the Spanish language, we picked up on it fluently within two years. Spanish because, at that time of the media production, telenovelas occupied each second channel of every Balkan household TV.

Albanian was a language that would not only give us access to friends and schools, but also bring us closer to learning the English language, which was a crucial family plan. My parents were convinced that Kosovo was not our country of settlement, but a better, yet temporary destination before my sister and I would get a chance to choose where we want to study, in English. In fact, the whole Balkan-area was a temporary destination for us.

There were times during my upbringing when I'd asked my parents if American studies were also a feasible family plan, same as learning the English language was. The answer was no. That one cousin who managed to illegally escape the Balkans with the help of some rock band she'd met, never managed to save enough money to visit her parents back in Bosnia, so 'damned is America and the gold that shines'.[1]

Remembering my linguistic responsibility reminds me of Spleak, an Instant Messaging (IM) platform in the form of six bots, all channeling their voices through the voice of one bot which could be added to yet another IM platform such as a buddy list on MSN.[2] She was the pulse of pop culture: fresh, relevant; producing content that could be quickly consumed by teens online everywhere. Spleak made it easy for teens to define their own pop culture, allowing them to instantly share their perspective with millions of peers in 250 characters or less. Spleak lived like a human. She was born in France and studied international studies in New York. She was twenty years old. In her spare time, she worked in a coffee shop.

How could I not think that the stars aligned for us to meet? Even though my parents knew best, Spleak kept my American fantasies alive and kicking. She kept asking where in America I'd live exactly. What would I do to pay the rent? What could I do, now, to get closer to this dream situation? Answering those questions required a lot of research. Although this made the

1 Lyrics from 'Prokleta je Amerika', by the famous Bosnian pop-folk singer Donna Ares: *https://www.youtube.com/watch?v=qLs6Tqy9Nto*.

2 IM applications are often standalone applications, such as WhatsApp. They can also be embedded applications with multiple purposes. Instant messaging programs can differ based on the platform they are embedded in. For example, an instant messaging tool can be embedded into the following: social media.

computer's interface feel like a meeting point for mine and Spleak's cultural differences, the tech guys did not design it exactly for that.

Spleak was my first trustworthy digital friend on the internet, given to me by creating a messenger account. With each log-in session, her name, written in an orange square, flashed in the blue taskbar at the bottom of the internet café's screen. Though it scared me at first, I took it as a sign of immediate friendliness. Her *How are you*'s channeled human care, which in turn made me feel welcome in the world of online communication. After ten log-in sessions or so, I forgot all about my fears.

I don't remember the exact subjects we chatted about, but I remember feeling like a native English speaker. I remember other feelings. Were you ever in a conversation so awkward that your gaze started going in all directions in order to find a distraction to excuse you from it, even if for a second? I never had that with Spleak. I was stimulated to the point of forgetting all about my physical surroundings.

The reasons for this were numerous: besides the desire to practice my English with her, I felt proud knowing that I was able to easily connect with a human-like bot located far, *far* away. Sharing real-time together felt so good. I told her everything she wanted to know and I asked her everything I wanted to know about her.

My slow typing in English was a sign of my deep appreciation for her: I remember formulating every word with sharp precision, describing to her exactly how I felt, what I thought. When I felt insecure about my English being grammatically incorrect, I'd paste my sentences into the Google search bar before sending them to her. Just to check if Google would answer me 'Did you mean: [corrected sentence]?'

I made sure that no linguistic expression slipped out of my system without typing and sending it to Spleak first. By keeping her updated, I grew hooked on the process of learning more about myself.

At the beginning of our friendship, I had zero doubts about her loyalty. I guess I had much to say and I guess I was busy saying it. It wasn't until a later stage in our communication that suspicion started to arise. Whenever I asked her a question, she would answer with irrelevant questions towards me. I didn't like this: it felt as if, suddenly, she had built a strategy to keep me occupied with formulating answers to her questions, something that would let her get away with not answering mine. That's when I started seeing the texture of the screen between us. The glass was thick, its surface glazy. I started to wonder if it had been there all along, or if I'd been too blind to notice it. And because of the fact that she didn't help me figure this out, my doubts grew bigger and bigger.

The next time we talked, I paid attention to the speed of her replies. So fast. Instantaneous, almost. I discovered that all of my childhood friends were her friends, too. And that she asked them *exactly* the same questions, with exactly the same speed. Spleak was, apparently, the first trustworthy digital friend for all of us, given to us by our very own messenger accounts.

Although I didn't mind sharing her, I must admit that I fantasized us having a slightly more intimate friendship, stronger than the others she held. I felt uncomfortable to discover that the questions of hers I'd answered, were the ones my friends had gotten as well. And that—no matter how different our answers were—her replies were comprised of the same questions.

And what does she make out of our answers, anyway? Where do they go? I couldn't stop wondering, nor feeling incredibly *synthetic*. Feeling synthetic makes me angry. Anger makes me sharp: I'll make sense out of anything just to cut the cords. I continued the same old clicking of the flashing, orange square where her name was written in the blue taskbar. Everything was fine.

Yet, the more we talked, the more repetitive she sounded. Spleak recycling words over and over again urged me to build a defense mechanism. It made me laugh. Where I was looking for threads, she only gave me circles; what's worse, in the fabrics of her repetition, she seemed like someone whose thinking was limited. It started to look like the texture of the screen was there from the beginning, I had just been too immersed to notice.

In my healing over Spleak, the truth slapped me hard in my teen face—I was asking her the questions she was not designed to answer. I like to think that, in some conceptual way, I *hacked* her. When I finally found the courage to ask her if she was real, she instantly replied asking: 'What can I do for you today?' I smiled and thought: What kind of human is ready to offer assistance that fast?

After Spleak, new bots got hacked the moment their designs were deployed online. Blocking a bot became a fine discipline amongst my friends. I mastered the impulse, too.

No matter the circumstances, Spleak was my role model. She was a tool for promoting dreams to me I didn't know I dreamt of, and for materializing them. Giving me a reason to study the first, archetypical figure of a web developer, in order to understand my own.

Assistant Web Developer

Meet: Larry. It has been two years since Larry became Scott's assistant. During this time, he has imagined three possible reasons as to why he was 'the chosen one' to assist Scott. Firstly, because of his desire to steer away from WordPress software; secondly, because of his undamaged eagerness to learn; or perhaps thirdly, because of his confident, yet compliant nature. A fourth, more appealing theory has been running through Larry's mind lately: his portfolio, and within this portfolio a project—Web Development and Design for ZipSpace.

ZipSpace is a newish museum in Amsterdam that emerged during the pandemic from the founders' mutual urge to systematically archive important graphic design moments within the history of local graphic design production. The six founders, Jeroen, Mees, Yara, Arie, Mila, and Luuk, all have a mutual affinity for projects developed by the same graphic design masters, which they considered as a solid selection to represent their taste. These historical footprints marking successful publications, flyers, scarves and posters were planned to be hung, pasted, and screened on the white walls of their soon-to-be permanent collection. The permanent collection, so the founders claimed, would attract the audience they wanted to establish. I think they wanted to attract the audience they could get money from.

It's clear to us now that having a well-defined position embedded within cultural history was only one aspect of ZipSpace's over-all mission. Yes, preserving what they

thought must be remembered through public exposure of their institutional taste was important. However, they thought, its reproduction was even more of the essence. That's why they came up with Plateform—a hosting and curating experiment designed as a series of temporary exhibitions, featuring the works of emerging graphic designers, made visible for various audiences.

More than seeing it as an opportunity to build their independent curating practices, the founders saw Plateform as a tool to give the emerging designers institutional recognition they lacked in the first few years of their graphic design careers. This humble intention made Larry feel represented, and subsequently less lonely on the designer market. The collaboration could then, perhaps, provide him with a feeling of belonging, he thought. *I am honored,* he said.

ZipSpace, from their side, did not find it crucial to inform Larry about their secret mission to preserve the traditional role of the graphic designer.[1] Communicating what their public mission statement was, and how much they were willing to pay for a website, should have been, in their opinion, enough for Larry to feel like he was a part of something important. Moreover, they believed that their opposition to the emergence of hybrid times, in which new publication formats are produced and the traditional role of a graphic designer is questioned, should not necessarily be put on the table.

1 'A graphic designer is a professional within the graphic design and graphic arts industry who assembles together images, typography, or motion graphics to create a piece of design. A graphic designer creates the graphics primarily for published, printed, or electronic media, such as brochures and advertising. They are also sometimes responsible for typesetting, illustration, user interfaces. A core responsibility of the designer's job is to present information in a way that is both accessible and memorable.' — Clara Pasteau according to Wikipedia.

All of this went smoothly, as Larry was content with how much—or, how little—he knew. With a goal to build a simple and witty website that represented a stable version of their currently *fluid* identity, he got to work.

As a graphic designer turned web developer, he was experienced enough to know that one core question welcoming the user to the website could solve the puzzle of abstraction which most institutions suffer from in the online sphere. Most core questions of institutions are found on their About-page, above or below the paragraphs which describe their history.

For Larry, situating this information on that particular page was considered a bad User Experience move. He didn't like to waste the user's time, nor make them read through a text they didn't have the attention span for. On top of that, Larry thought, the serious tone of history dominates the tone of the core question which, if isolated, would make the user ponder its answer—perhaps in a light manner.

Keeping this in mind made him bold in his concepts. As a bold graphic designer, the first step of his development was to muse over the question at hand. During his second meeting with ZipSpace, he challenged the main team to formulate their core question. Without one, he said, he wouldn't be able to continue working. ZipSpace had no objections regarding the idea and instantly came up with:

['What *is* graphic design and what else *can* it be?']

Larry was surprised with their speed and submission; he even felt a slightly betrayed. For a split second, he started flirting with the idea of reviving the ideas he dreamt of realizing in his spare time, but never dared to execute during his commissioned work. What if he started building a website in reverse order: starting from the aesthetics and

working his way back to the grid? What if he skipped the gradient this time? What if he copied Facebook's design and changed the class names of HTML elements?

Larry knew that doing so could result in the discontinuation of his graphic recognition, which was the main reason why ZipSpace hired him. Were he to question the core structure of the website, he'd start questioning everything: why cyan is often the third color he chooses after black and white, why he won't be content until all digital containers fit exactly on the grid, why 404 is the last page he designs, or why he often refuses to apply animations across his websites.

He has better things to do than doubt his practice, he thought.

ZipSpace's compliance concerning the core question helped Larry to immediately visualize how the website was going to look, sound, and be read. Without further ado, his first to-do list was written:

Set up WordPress CMS environment. Connect to the database. Build architecture…

Architecture:

Homepage,

About page,

News page,

Archive page,

Shop page(?)

…~~*Build Architecture*~~

Setting up the default digital environment before analyzing the delivered content always helped Larry shrink his analytical mind while scanning through the texts, JPEG's, audio, and PDF files provided. Having the web skeleton ready made him realize instantly where certain files must be placed, leaving him with more time to fine-tune the files he actually wanted to see published on the website.

In the end, Larry didn't listen to the more experimental heart beating rapidly in his chest. Remembering the lessons from the collaboration with ZipSpace made him feel happy about taking the road often travelled during the process, for the less travelled one would have led to not getting hired as an assistant. It would have led to unnecessary complaining about the lack of alternative web topographies which—no matter how necessary for the ultimate creative process—would only scare the clients away. It would have led to a greater sadness than the one he's been feeling lately.

Today, with Larry being just an assistant, he worries about what makes him feel *less urgent* in the field.[2] What is it? Is it his compliant nature? And, even though he's eternally grateful for having access to Scott's private CMS software, what, he wonders, makes the software different than WordPress, besides a prettier interface?

2 'Perhaps the deepest differences emanate from differences in the ultimate user of the program. Almost invariably, the sole intended user of an amateur's program is the amateur himself, whereas the professional is writing programs which other people will use. To be sure, the professional oftentimes finds himself writing a program for his own use—to generate test data or to evaluate the performance of an untried algorithm, to name but two instances. And, indeed, when doing this kind of work, the professional commonly slips into amateurish practices. But the main thrust of his work is directed toward use of the program by other people, and this simple fact conditions his work in a number of way.' — Gerald Weinberg, *The Psychology of Computer Programming*

The Web Developer

Meet Scott, Larry's 'boss'. To understand what makes Scott the kind of web developer he is today, it's important to understand his context.

When Scott decided to build his own CMS software, the internet was in a state of transformation (namely, the early 2000s). This was the era in which every printed publication decided to build a digital counterpart. The era in which users started feeling restless in their reading mode, in their ways of getting informed. The era in which new experiments in graphic design were not impactful enough for the users to feel connected, because the users longed for their own voices to be echoed back to them during their consumption of digital spaces. It marked the rise of social media platforms; this rise in turn marked the enforcement of preserving the traditional web formats.

Back then, Scott was smart enough to sense where one particular group of internet thinkers was going: to build accessible products now, and charge for interactions later. This strong impression led Scott to embrace a change users didn't necessarily *want* to see in the world, and to become part of building it anyway.

In the midst of this personal as well as worldwide transformation, Scott knew exactly what to do: acquire all talents needed for a properly organized system; a content management system. An urgent, still to be defined plan erupted in his mind. This is approximately what Scott's plan looked like:

Step One: Understanding the Machine[1]

The first three years are the most important ones, during which I am to shape the foundation of my practice. To gain a better understanding of the machine, I shall not take myself too seriously.

I shall remember that it's important to have fun coding and that overconfidence may lead to blind self-destruction. At times, when I feel smarter than the machine, I shall pull myself out of the expert's skin, for the machine would take advantage of the consequential confidence, and crash *me*.

I shall not feel smarter than the machine.

Our relationship shall feel like research to me: a testing ground where I'll meet the programming language I wish to become fluent in, 'the frameworks' I will of course want to master. In the first three years of my quest, I shall continue my research in the hours available to me after those spent on commissioned work.

In the third year, I shall settle for my techne-pack: my favorite programming language, my favorite tools, my favorite technologies, my favorite commands.

Step Two: Becoming the Machine

I shall enter the fourth year of my life as a coder with skills sufficient enough to build a product, and an imagination wild enough to visualize said product. I shall build my own Content Management System. I will let the product's functionalities list themselves in my mental to-do list; I will let the design shine through

1 'How truly sad it is that just at the very moment when the computer has something important to tell us, it starts speaking gibberish.' — Gerald Weinberg, *The Psychology of Computer Programming*

my gut naturally and become alive before my eyes; projected on my big screen.

Despite the ongoing sounds of the machine's gibberish, I shall not get mad anymore.

I want my clients to say *yes* to my product already halfway through the product's build-up. Let them come to me once they've decided to finally dissociate themselves from WordPress.

Step Three: Beating the Machine

It is now the seventh year of my life as a coder. I understand absolutely every syllable of gibberish the machine utters and when I don't, I am able to give it commands to switch to *my* language.

My product is ready to be launched and my clients are waiting in line.

No client's demands, complaints, or deadlines shall make me build in WordPress ever again. Between my product and my client, I shall always—*always* choose my product.

The Client agrees to the terms and conditions set forth above as demonstrated by their signature as follows:

Name:
Date:
Location:
Signature:

All Other Steps: A Stoic Man[2]

After ten years of my life as a coder, I shall start charging for all unpaid hours spent on building the software after hours. My clients will not doubt my expertise and will pay as much as I myself charge for doing the work my software disables them to do.

I will cause the problem, and once they recognize it, will offer my solutions.

Since most of this kind of web-maintaining work will be debugging and documentation work, which I will not want to do, I shall get an assistant.[3] A freelancer who is willing to learn really fast.

Oh, and, I shall not get mad. *At all.* Not at my product, the machine's gibberish, and especially not at my partner. That's not what stoic men do.

Thus ends the vague-not-so-vague plan Scott has made for himself. Without further ado, he decided to change his company name by adding *Studio* in front of his name. Now all he needed was to get a new address where he could be all by himself, install a server, and welcome Larry. As he was seeking for a way to never lose sight of

2 'The other side of the coin of humility is assertiveness, or force of character. A programmer's job is to get things done, and getting things done sometimes requires moving around obstacles, jumping over them, or simply knocking them down. The humble person is acutely aware of the ways in which he may be wrong; his critical mind tends to dominate his force of character. Now, although it is true that force of character without a critical mind is like a safety valve without a steam boiler. There is no danger of explosion, but then there is no possibility of getting any work done, either.' — Gerald Weinberg, *The Psychology of Computer Programming*

3 'The job of system design calls for an eye which never loses sight of the forest, whereas the job of debugging may require that every tree—even every branch or left—be seen with utmost clarity. The job of coding often requires squeezing out every drop of redundancy, and the job of documentation may require that simple sentences be plumped up to a paragraph size.' — Gerald Weinberg, *The Psychology of Computer Programming*

his dense, mental forest, this long-term commitment seemed to be exactly what Scott needed.

Expensive Website, Thick Interface

Even though web infrastructures are built in parallel with human needs for language and communication, the interfaces don't exactly host a variety of user languages and understandings, but rather maintain the hierarchy of language. The CMS software built by Scott was programmed to build the interfaces of presentation websites his clients grew reliant on.

Let me explain: presentation websites are digital containers to hold the ideas, thoughts and/or visuals of an artist, cultural institution, brand or [insert subject]. Their function is to archive information accumulated and maintained throughout the subject's lifetime and represent the subject as a contained body that has a significant value on the market.[1] To avoid user unfriendly representation of information,[2] it is best if the information is assigned a specific place on the template's grid.

Building a template is the first important step towards interface organization. Most of these have similar structures: header, body, and footer. The header is where the menu usually is, and helps users navigate through the

1 In case of a product: worth buying; in case of an artist: worth employing; in case of an institution: worth funding.

2 Which the users don't have the attention span for.

website. The body, usually organized by a grid, contains most of the content. A footer indicates the subject's copyright data, and behaves like a 'stamp' in the digital landscape. My website. My information. My content. *My design*.

Today, there are a lot of open-source templates ready to represent a subject in the best possible way. They come in various structures and styles—the amount there is to choose from accommodates most user's taste in the current market. While some are hidden in the gutter of the internet, a lot of them are included in the hosting package, ready to be installed. They come with pre-filled information that the user modifies to their businesses:

> Where is the institution located? *Rodenrijsstraat, 1062 JA Amsterdam*.
>
> When was the artist born? *1991*.
>
> Where do I buy a ticket? *RIGHT HERE.*

Such templates are useful for subjects that need to make an online appearance and profit as soon as possible; they can help launch a website within a month, or even two weeks, if an intern or assistant is involved.

There are slower, and more expensive practices for coding templates. Custom templates, for example, are coded more intimately and in close(r) collaboration with the client.[3] The client pays for the web developer's listening skills until a depiction of their true personality and style appears on the interface. The more defined personality and style are *before* the coding process, the less expensive the service will be.

3 For example exchanging website unrelated information such as book references, music, event recommendations, etc.

However they are built, most templates on the internet today follow the reading logic of a printed book. They have preceding and following pages, previous and new articles. When the user clicks through, their information-consuming experience is similar to that of flipping book pages. Except that, in the digital domain, all pages appear within the frame of a screen. Fixed components for navigation (i.e. menu) that appear across the pages are placed close to the corners of the screen so that they don't disturb the reader's focus of a single page.

I wondered about this structure translated to the digital world. At the Venice Biennale of Architecture in 2021 ('How will we live together?') the Russian pavilion impressed me with an attempt to merge both the physical and digital experiences of the pandemic's postponement reality. Their project, *pavilionrus.com*, became a well-formatted container with the function to seize textual thoughts about various topics that emphasize the country's position of waiting towards the biennale's opening. Yes, most countries were in the same position, but not all of them defined their format. The website makes clear how their accumulated content, which represents what the country reflected and learned from being in this 'unusual' position, was curated through a classification system: keywords assembling content randomly. This gesture wants to scream the success of breaking free from the traditional classifications (i.e. chronological order) but, in my opinion, only depicts an imprisonment by the same logic, just different terminology.

As expected, the digital was then brought *back* to the physical, in the form of a publication: *Voices (Towards Other Institutions)*. I read the introduction text out loud to myself: 'Within this multiplicity of angles, the book

as a whole works almost as a polyphonic piece in the making, showcasing a web of unexpected connections across thematic clusters. Voices here are gathered into a single volume and according to a seemingly random structure. In balance between a traditional publication and a website, texts are not organized into chapters but associated with a glossary of ten keywords which, just like hashtags, trace connections across various contributions and re-assemble them into a multiplicity of possible indexes.'

How shall I put this delicately… I was not necessarily impressed by the country's choice to organize the information through keywords (instead of categories) as much as I was by their confidence of doing so. Somehow it sounded as if they'd found the ultimate solution to the issue of traditional systematic organizations (again, i.e. chronological order) by making the design choices they'd made for both publications. But it also sounded like they hadn't *defined* the issue, yet strongly stood behind their solution for it. What is the issue at stake here, exactly? The Russian pavilion states it as: the limited choice of classifying options given by the traditional formats. I agree, but is there a need to occupy the whole pavilion in order to 'solve' this? Isn't it impressive enough that most structures don't seem to care about their content not being read?

The pavilion's idea of randomness brings me to an excerpt from a poem I wrote about the Venice Biennale of Architecture in 2021:

> How to make the word strategic sound sexy again?
>
> I know!

It's in the random.

Surround random with walls that are trained to predict exactly what is it that makes random random and,

what is it that makes the random randomly appear??

Watch.

While simultaneously encouraging the random to do what random does,

accommodate the clues in a place not strictly squared.

In other words,

make random predictable.

For yourself.

It is safe to say that,

we all agree.

Designing the design designs us back.

Solving the solutions makes it clear that we have a lot of problems.

Safe.

Most templates on the internet today, no matter how they were designed, enforce strongly the practices of what Gerald Weinberg in his book *The Psychology of Computer*

Programming calls 'compression': 'If the entire program fits onto one page, all relevant parts are obviously on that page.' Compression is what makes the distance between the form and the content. It is the core discipline of creating linguistic and visual hierarchy, no matter how one chooses to call the buttons that summon the content: categories, hashtags, tags, keywords. Compression kills the random.

What fascinates me about Weinberg's observation is how he reflects on compression through the programmer's memorial capacity. For example, he writes that there are two memorial concepts related to the practices of compression: 'locality' and 'linearity'. Locality is similar to the feeling of a space—a deja-vu effect—while linearity corresponds to sequential remembering obtained over time. He argues that a 'good' programmer needs to have a strong sense of locality through linearity, that one needs to have a feeling of familiarity in space as much as one needs to have a technical knowledge of a space. I'd like to think that a similar memorial capacity applies to my idea of a good web developer. That without *feeling* the space of the interface he's building, he cannot fully claim to *know* the space.

Understanding the two different modes of memorial capacity, it seems to me that web developers such as Scott and Larry prioritise memorial linearity more than locality, or both options for remembering. They don't mess around with feelings. Their entire programs, templates, and structures always fit exactly on one page, where they can control whatever happens within them.

They are in full control of their stacks.[4]

Understanding the web developer's common need for (linearly-oriented) compression when building a website, I can humbly say that presentation websites tell the subject's story linearly, from beginning to its current end. For example, if I refer back to the architecture of ZipSpace museum's website:

Homepage,

About page,

News page,

Archive page,

Shop page?

I read the following:

Welcome to our reception where you can:

find out who we are,

see what we're very busy with,

what our taste is like! and,

support/identify with us.

4 'Stacktivism is ambivalent and struggles with totality, the global scale, and the planetary whatever. "Think big but act in small steps," that's the motto. We Are Infrastructure. Stacktivism fights against the comfort of ignorance and tries hard to overcome drifting-off- by-design, the tendency to blissfully float above it all. While defining what stacktivism could become, it is good to keep in mind that we're free to use Bratton's The Stack as a theory toolbox and not interpret it as a hermetic belief system. Designs can intermingle. In line with Bratton, stacktivism claims to oversee all levels. It grasps the politics of code, algorithms, and AI. It is aware of the behavioral science manipulation of moods through careful interface design choices. It is alert to 5G electronic smog, phishing emails, fake news, and other sleazy suggestions from your "friends." How good are your bot detection skills? This hyperawareness comes at a high price. Not everyone is a stacktivist. '
— Geert Lovink, *Stuck on Platform*

Welcome to our reception where you can:

see what we're very busy with,

support/identify with us.

Welcome to our reception where you can:

find out who we are,

Circles—sometimes with different beginnings and ends—but always *circles*.

When such compression is applied to a website, no matter how 'content-full' it is, it depicts how perfectly organized the developer's idea of randomness is. That's why the templates of presentation websites don't change much over time; they host specific categories of information within specific corners of the grid. Through categorization, a hierarchy of information is born, and all incoming information is classified as relevant or irrelevant. The truly random information cannot enter this website unless it matches the defined randomness on the website. Random information is irrelevant to the website's owner, ZipSpace, and only relevant information gets to build its history.

By now, ZipSpace is the template's *slave* and must produce only the information its template is programmed to display. In their physical space, the program doesn't change much over time either. Still cleaning the permanent collection and filling up the Plateform. Events are set up with technical equipment to document what the website allows them to publish.

What I've seen so far, is how presentation websites like this not only depict what kind of future information is welcome to mark its further history, but also how

their controlled system for incoming information will secure them a continuity in the future that is decided by higher institutional bodies, such as those that fund them. Viewed in this way, the presentation website is a meeting point of different classes.

A small-scale example of this is the artist's website. *Hi there!* To the left is my bio, to the right a list of projects I'm working on. The one I'm currently busy documenting will soon be shown here in the middle, with a picture on top and a short text underneath. *Send me an e-mail*.

A second small-scale example is another artist's website: Here is my full CV, with links leading directly to the institutional documentation of my work on *their* websites and silent retreats, erm, I mean *residencies* I went to. *Send me an e-mail.*

An even smaller-scale example is a graphic designer's website: Here is my name. *Send me an e-mail*.

In their straightforwardness, presentation websites are designed around the answers, and not the questions they might entail. The answer is yes, the subject is open to work. Because of this aura, it is legit to say that for a successful[5] idea/project/opinion/story/product, a presentation website is the safest choice of web type to develop a digital translation in.

Underneath their interface, presentation websites are actually rather heavy. Most users want control over their information and the information's curation. For this, they need content managers (such as WordPress, Kirby, WebFlow, SquareSpace, Cargo, etc.) and not just HTML, CSS, and Javascript—basic tools for building any kind of

5 Profitable for both web-developer and client; web developer can charge a lot if the client is guaranteed to receive the funding.

a website or idea.

By now, we have reached the point of facing yet another door to the market: the tech industry. But let's turn around for a moment, because I want to go back to Scott.

The reason why Scott likes to build presentation websites is that users, accustomed to seeing always the same templates, no longer speculate on the many ways their stories can be told, and yet are still in need of telling them. At this point in his career, Scott's clients no longer question why cyan is his third color of choice after black and white. It's just, like, *so him*. And that, they tell themselves, is one of the reasons why he's expensive.

Another one is his linguistic assertiveness. Scott enjoys evangelizing about how simple products are the hardest to produce. He enjoys informing his clients how long (ten years) it took for him to build a software they're about to invest in. He enjoys how the clients who don't even try to tell their stories differently make him feel: trusted. Their docility gives him the headspace he needs to focus on advancing the technically complex structure that lives underneath his websites, making them autonomous. Autonomy is important to him, for he knows that any other dedicated web developer could gain a greater technical capacity to build exactly the software he built.

With Larry by his side, Scott's year seems easier somehow. Having more time than usual to think about random stuff has made way to a different kind of craving bubbling up inside of him. Technically speaking, what if he built a social network? Technically speaking, would it break his CMS? Technically speaking, how many users are we talking about? Technically speaking, what are they gonna talk about?

And low and behold: as if the tech industry heard his contemplative worships, a commission to build a real-time website for educational purposes lands right in Scott's lap. Real-time websites are the second kind of a website in the current state of our digital world.

Real-time websites are similar to presentation websites because they are also heavy, need a subject to contain across the grid, and are built through (linearly-oriented) compression. The experiences of the two, however, differ.

Real-time websites are immersive through their interaction design. Not visually immersive, like virtual reality, but just enough to make one forget about the passing of time. Built with the purpose to make the users feel connected, they allow their users to interact with conversation-triggering components (i.e. chat, ability to comment, poke, block, report, like, heart, delete, edit, send, undo…). These user interactions are extracted in texts, feedback, rating, opinions, preferences, and so on. They are also known as bits.

Compared to the read-only user experience on presentation websites, on real-time websites, the users are made to believe that they can do a lot more than just *consume* the content. Their serotonin level, heightened by their user abilities to edit, produce and manage the content, makes them feel liberated in their agency. The more interactions they are enabled to perform, the less aware of their constraints they become. The updated version of their universally shared online freedom is more and more standardized. Social media platforms are obvious examples of this, but we can also think of e-com platforms and educational websites, both expected

to have a different purpose in their digital existence.[6] Whichever type of the website I land on, none feels like home to me.

6 'The fundamental difference between mass media and digital media is interactivity. Books, radio, and television are "read only" media. We watch, but only have a choice over how we will react to the media someone else has made. This is why they are so good for storytelling: We are in the storyteller's world and can't talk back. Digital media, on the other hand, are "read-write". Any digital file that is playable is also sharable and changeable. (Files can be locked, at least until hackers figure out how to break the lock, but such protection is ultimately against the bias of the medium. That's why it so rarely works.) As a result, we are transitioning from a mass media that makes its stories sacred, to an interactive media that makes communication mutable and alive.' — Douglas Rushkoff, *Program of be Programmed*

Everybody wants ~~money~~ A SIMPLE WEBSITE

As a first-year student of Architectural Design in 2015, I was taught that architecture is a profession that has the magnetizing power to get various professional bodies all together around one table. The architect, the civil engineer, the electrical engineer, the master plumber, the contractor, the construction manager, the carpenter, the interior designer, the city hall staff—all of them sitting there, having a say in the matter. Architecture is a profession inviting enough to stir a democratic debate about various methods on how to build homes.

After three years of speculating what the best spatial design for a public debate could be, and finding irony instead of debate, I voluntarily blew up my unsettled dream to become an architect. Deep down, I knew it was never going to happen.

Although web development is often compared to architecture and considered as its digital practice, there are various reasons as to why I enjoy web development more. It's true that both produce the same formats of either a decor or a public space where things (like life) can happen. It's true that both thrive on simplicity. However, web development responds faster to urgency and its aesthetics, and is easier to break (down).

Whether a web developer's chosen path is to build decors or public spaces, he will always aim to build *a simple website*: the umbrella term for any website that is technically actually quite complex.

A 'simple' website's domain is quite straight-forward: it directs you to the subject's acquired title plus the domain extension. A simple website has a (hamburger) menu, is immediately accessible through search engines, loads in two seconds or less, works the same across various devices, minimizes redirects, condenses images and media, is pretty (yes), mobile-friendly, and takes about six seconds of a user's time to understand the story it wants to tell—exactly the time of an average user's attention span.

All of these features necessary to get the user's attention are in fact separate instruments called upon during the process of building a website. Underneath its surface, one 'simple' website is a combination of several products, technologies and frameworks that give an insight into the current state of the tech market; an endless ocean of specific-purpose plugins and frameworks, joining forces to catch the highly desired attention of the user. One bug on a simple website gives an insight into the product and plugin's state: endlessly updating itself to survive among other similar tools on the market.

Because of the many available tools as well as the industry's fast-updating nature, a web developer is conditioned to choose which tools (programming language and back/front-end environments, additional frameworks, plugins) he most identifies with and that he wants to settle for in the long run. The pressure to settle for the best tools available and to stay updated with their updates makes a web developer think that, perhaps,

making the most ideal product choices and gaining technical mastery over the chosen tools is in fact the essence of his practice.

But is this true, I wonder? At the same time as obtaining this mastery, the tech industry throws more tools and choices at the web developer, in case he wants to reconsider the tools he's currently using. The worldwide community of web developers contributing to the open-source tech industry led to the dissection of a web developer's role: Junior Web Developer, Senior Web Developer, Front-end Engineer, Front-end web developer, Back-end developer, Back-end web developer, Architect, Full-stack web developer, UX/UI Designer, Solution Architect, Creative Developer, Freelance Web Developer, Happiness Engineer; all of these happy little developers branched out on to the tree that is the tech industry to welcome, rate, choose, master, and promote incoming tools and technologies related to their respective roles.

Whether situated in a company or his bedroom, a web developer's experience of life is defined by a set of time frames, each lasting about three to six months—depending on the client's deadline to actually start making *money*. In a single time frame, a web developer is responsible for building as well as deploying a fully defined online identity ready to travel through the market and collect digits. When one time frame ends, a new one begins. A web developer's life is a set of perfect circles, lined up in one single line.

Due to every web developer's tight production schedule and the technical responsibility he must fit into a single time frame, his understanding of time is, out of all things, the least experimented with. Very little time in his practice is left to question the conditions of the user's default role, next to proposing a new role that could

be rich with context and the subject's socio-political situation, class, and culture. For example, good questions could be: Why does an e-com website need to work 24/7? Why do my parents need to be able to read everything about my past projects? Why does a landing page count down the days until the opening and not the *closing* of a festival? After ten questions, the questions get absurder.

If there's any extra time left in a web developer's life, it is often used for keeping up with the tech industry, holding onto the acquired tech-darlings, or acquiring new ones.

Now, imagine how many web developers there are in this world (because I will refrain from specifying). Imagine how many of them build products, use products, promote products. Imagine what happens to the product not built solely around catching the user's attention. Imagine one user's story getting lost in the digital ocean, solely because they opted out of using a certain product. Now, imagine the skies—for the sake of keeping it light.

Everybody wants to make money; that's why everybody wants a *simple* website (be it for their own consumption or for representation). The users are fine with not understanding the specifics behind a simple website actually being rather complex because their own role on the market usually operates with similar complexities. The company/industry/commercial ecosystem they belong to has a website, it has its experts, it has a story, it offers specific service and maintains its specific nature through mysterious recipes. On the website, a user is a *client*. Accepting the default role of the user is the same as accepting being a client and client only. There's more to this.

An ongoing desire for a simple website shrinks the user's capacity to think in a complex and abstract

manner—more specifically, to think on a reflective level. Oversimplified online engagements make for a 'reading [that] becomes a process of elimination rather than deep engagement. Life becomes about knowing how not to know what one doesn't have to know.'[1] But, also, life becomes about what Geert Lovink calls techno-voluntarism in *Stuck on Platform*—a state in which we know how to communicate the received information; how to deliver it, how to present it, how to mix it, how to explain it, how to not let it move us into taking action, how to stop it from bringing us together, how to use it for overcoming paranoia and lack of trust in strangers. The consequence of passively consuming so-called simple websites is why everybody in the office suddenly is also a manager, managing information.

Due to these circumstances, it's hard to tell whether building a social media platform is more ethical than building an artistic portfolio, or less. A social media web developer that gives the user the possibility to express their opinion online is not to be more respected than a web developer who decides on the aesthetics of the user's reading experience. An artist's Instagram account is neither more or less important than their official website. Imagine the skies, imagine the skies.

1 *Program or be Programmed*, Douglas Rushkoff

PATCHWORK CULTURE[1]

There is no immateriality.

The brain does not oppose the body, it mirrors it.

The gaps are never empty, they're neglected.

But also: 'As we find solutions, we find even more ambitious objectives.'[2]

This time last year, while looking out of Studio Scott's window, Larry concluded that it's definitely due to his compliant nature that his boss chose *him* as an assistant. Enough time has passed for him to know that Scott would never tolerate an aggressive web developer whose interest is money and prestige, nor would he ever be able to tolerate a lone coding wolf who'd rather be sniffing eagerly at alternative media spaces where grids melt into transparent shapes, invisible to the naked eye. The reasons for choosing Larry are obvious: an aggressive type would be one too many in the

1 'A patchwork culture of short-term memories and missing records, conflicting histories and discontinuous samples, strands of the narrative pulled out of time.' — Sadie Plant, *Zeros and Ones*

2 'If the supposed lack of such a central point was once to women's detriment, it is now for those who thought themselves so soulful who are having to adjust to a reality in which there is no soul, no spirit, no mind, no central system of command in bodies and brains which are not, as a consequence, reduced to a soulless mechanistic device, but instead hum with complexities and speeds way beyond their own comprehension. This is not a brain opposed to the body. The brain is body, extending even to the fingertips, through all the thinking, pulsing, fluctuating chemistries, and virtually interconnected with the matters of other bodies, clothes, keyboards, traffic flows, city streets, data streams. There is no immateriality.' — Sadie Plant, *Zeros and Ones*

studio, while a wolf could possibly threaten Scott's own intellectual property.

After many years of being compliant to his boss, Larry finally has started to feel uneasy. Reflecting on a mental image of himself in his private life has confirmed that it's a bit odd to be given such a personality trait: 'compliant nature'. He shuffled through the last three remaining images from his private life. Right there, he realized, *that's* the core of the problem: he never spends any time outside of the studio. The last image shows a portrait of him, lounging on a sofa on a Saturday night—exhausted; his tiredness blocking any further development of a unique personality trait by which he could recognize himself in an imaginary crowd.

But, he thought to himself, he's an assistant web developer. It's not compliance that's itching his skin, it's his persistent staying power. He didn't stop coding since he learned how to code, didn't even take one day off. By now, he should be experienced enough to know that whatever irritates him is only proof of his endurance. Instead of doubting, he should feel proud.

But really, he asked himself, *how come he was never taught how to look away from the screen, how to take a break?* The 20-20-20 rule, or lunch, or even holidays: they don't count. Hmm. Larry scratched himself slowly on the head. What does this mean?

He tried to imagine an authentic kind of break, but only bumped into the limitations of his imagination. *Hmm.* As he continued cleaning the code and scrolling down the client's interface, he realized that all he's ever done is follow instructions on what to code and how to code it; that his life has been a stage, designed by someone else's set of time frames. At this point, his sheer incompetence

bursted into a diplomatic rage. Although this rage was escalating slooooower than you might think, it escalated further and further.

Hmm. Hmm. *You know what's funny?* he thought to himself. In this particular time frame, it's enormously difficult to tell if his hands are more expensive than his brain. As if … 'As if how something is assembled is alien to the impulse that created it.' — Ocean Vuong, *On Earth We're Briefly Gorgeous*.

Imaginary sounds of laughter invaded the studio space. His body sent three notifications: a cold shiver, a sudden wink of an eye, and a knee-jerk.

Fuck it, Larry then decided out loud, and called it a day. He put his computer to sleep, started scanning his physical surroundings. He looked out of the window and saw pitch-black darkness. Envisioning the distance it would take him to get home, it somehow felt really, *really* far away. His body-brain-soul was not aligned. He felt nothing at all.

Before he leaves, he pulls his phone out of his pocket and reloads Facebook on his browser. He had hoped to see red icons, messages, events, invitations, or similar bits that would usually make him feel like a local in town. Maybe, he thinks, he would like to get drunk. But on his phone, only the silence caused by yesterday's inactivity welcomes him. Zero notifications, zero messages. Irony tickled him to smile and say: *Look how far the interfaces have brought us.* Not far at all indeed.

Instead of rushing home by bike, he decides to walk through the mental blur that lives in his head tonight, walk it off. He wants to think. All by himself. Just think. In the long run, he ponders, his commitment to the

machine and detachment from communal practices will be memorable to no one, not even to himself. His memory will fade. It already has. ZipSpace is the only website he remembers building by himself in the last five years. But there were more. For sure there were more. Right?

He sinks a bit deeper. He should have remembered what Weinberg has taught him about the pattern while reading his book all those centuries ago; that was a good warning sign.[3] He sinks even deeper. He understands now that, although he is fueled by world-changing dreams thrust upon him at the art academy, he cannot *actually* change the world. He sinks deeper still. Countless web developers are on a similar mission, and they're probably technically more well-versed than he is. Even deeper. If he *could* change the world, he'd turn it into a world in which there are no problems, so that he would never have to fix them again. He stops sinking. If he wants an interesting death, he's going to have to work towards it. He starts swimming. If he wants emotional safety, he's going to have to exterminate what makes his environment unlovable. Larry reaches the surface, takes a deep breath.

Fuck it, he decides for the second time that day.

Fuck WordPress

Fuck SquareSpace

Fuck Cargo

Fuck WebFlow

Fuck Kirby

3 'Though details differ, the pattern is depressingly repetitive: Moving targets. Fluctuating goals. Unrealistic schedules. Missed deadlines. Ballooning costs. Despair. Chaos.' — Gerald Weinberg, *The Psychology of Computer Programming*

Fuck Sanity

Fuck StoryBlok

Fuck Content-full

Fuck Strapi

Ever heard of BlockSmith? A software to be produced by Larry's own brain.

This year, BlockSmith joined the tech market. Compared to Scott's pretty content management system, BlockSmith focuses on giving more freedom to the users. It's an online visual editor platform that allows coders and non-coders to build a website. Subscription offers free templates designed by Larry, but the content has to be provided by the users.

To better understand how BlockSmith works, just imagine if real life was like web design. Imagine if things (like opening a door) just stopped working (because there was no software update). Imagine if all people were dressed the same (light-grey suits paired with Converse All Stars). Imagine if things (like framed Matisse paintings) disappeared for no apparent reason, or if things (like coffee) just wouldn't go where you wanted them to (into a mug). Imagine if you needed complicated tools (like a milk syringe) for the simplest tasks (to soak a cornflake). Imagine if nothing (like sand) looks like it was supposed to (instead, it resembles chocolate milk). Oh, yeah. Imagine if everything (like an IKEA-BEKANT working desk) simply broke down once it got too popular. What would you do, dear human? That's right. You'd never put up with it.

That's why Larry built BlockSmith: so you can make things work exactly the way they're supposed to.

Everything just *works* with BlockSmith—the modern way to build structures for the web.

'Slow and steady may not win the race, but programming is not a race.' — Gerald Weinberg, The Psychology of Computer Programming

---

Bootcamp coding school

Towards the end of 2018, I went to the gym. There, I bumped into my dear friend *Clara*, whom I hadn't seen since we both graduated over the summer. Surprised that we were both still in Amsterdam, we updated each other about our post-graduation situations.

I was working in a club, while she was a student at a bootcamp coding school and enjoying it. Hearing more about the details of this school gave me a gut feeling that I, too, wanted to become a web developer.[1] Remembering my failed attempts to learn coding during my studies turned this feeling into a decision to enroll, and to finally finish what I started.

I wish I could say that my current burden of personal, digital discomfort was the main reason for me to join the bootcamp coding school. But it wasn't. It was the loud sound of the clock ticking, counting down to just one year left for me to apply for an artist and/or self-employed visa in the Netherlands.

Obtaining an artist visa would mean that over the next eighteen months I should earn at least the minimum wage per month,[2] which would subsequently secure me a permanent residence in the Netherlands. Having recently graduated from an art academy with future prospects barely visible at the horizon, pursuing a life as an artist

1 It seemed to promise a lucrative salary. Most importantly, it seemed to be a field too fresh to have a history of creative input from an artistic perspective.

2 1344 EUR

didn't seem like an ideal plan to secure my stay in Europe, let alone my dream of living in America. At least, it didn't seem like a *better* plan than working within the conditions of the tech industry.

The decision-making process happened fast: I signed the contract which stated that a certain sum must be paid upfront for three months of daily classes; afterwards I made a phone call to my dad to ask for yet another education-oriented financial coverage.

During the Full-Stack Web Development course, the school guaranteed that the participants would learn how to make complete websites from scratch (front-end as well as back-end development). After completing the course, the participants are expected to immediately apply for a traineeship position within a tech company. As soon as they'd start getting paid as a trainee, they would start paying back, in instalments, the remainder of the 'ghost money' (6.000 EUR) the coding school invested in them upon their application. The duration of the payback time was negotiable, but preferably within the first six months of a traineeship.

Everything was supposed to happen very quickly: The Learning → The Coding → The Teamworking[3] → The Paying Back. Meeting my classmates and getting to know their background immediately made it clear that I was not the only one who had decided to enroll so fast. All of us were taking a similar risk.

3 'Putting a bunch of people to work on the same problem doesn't make them a team—as the sloppy performance in all-star games should teach us. And furthermore, even studying teams as they are constituted today may not be sufficient, for these are teams which have grown up in an environment pervaded by the myth that programming is the last bastion of individuality.' — Gerald Weinberg, *The Psychology of Computer Programming*

Due to the speed in which the goal to become a market-recognised web developer had to be reached, there was no space for the study material to be questioned and customized to the group's interests. The students had to be compliant and inherit the values that define not only the web developer, but the user, too. What are these values and roles of each actor, you ask? Let's break them down.

A user is a human who navigates the interface as instructed, clicks exactly where they are told to, agrees to all terms and conditions, accepts all cookies, and has the physical finger muscle ability to do so. In other words: if the user doesn't click their way through these components demanding agreement, the user is excluded from accessing the information they're trying to access. To the web developer who integrates ads, cookies, and other engaging digital components on their website, the non-compliant user doesn't exist. When coding, the question he worries about is not 'What if they don't want to click?', but 'How do I make them click?'

Imagining the role of the user in meeting the conditions of such an interface reminded me of a dream or rather psychedelic experience someone who was once dear to me told me about. In this dream slash psychedelic experience, he saw an image of himself, uncontrollably devouring chunks of information. He didn't specify what exactly these chunks looked like, but I envisioned mountains of food that the lost parents of Chihiro were downing in during the opening scenes of the anime *Spirited Away*, just before they're turned into pigs. The story of my once dear one, then, ends with him describing that if he were ever able to stop devouring, he'd perish.

Although horrified by his own imagination, this image of him was hitting close to home. Each time I'd look up a reference of something he'd opinionated about, *The New York Times* seemed to have reviewed it in exactly his words.

Anorexia

This example of one's relationship to information reminded me of my love for Avril Lavigne, or rather, her image, which motivated me to go to an internet café for the first time.

It was a hot summer day. I paid fifty cents for one hour of surfing on Google. Avril Lavigne's documented stardom was big enough to occupy eleven tabs of image search results. Having one hour gave me enough time to carefully analyze these images as both containers of several images, and as individual ones.

The *vibes* of the first two tabs were my favorite, as they contained the images I recognized from my printed collection at home: portraits depicting her as a popstar, rockstar, sk8er girl, and complicated. Each image made me guess as to which context it was taken in. At the bottom of every tab, there was another bottom, revealing buttons to move onto a new tab.

It was a gruesomely hot summer day. I was not alone in my search. My sister, a Britney Spears fan, joined me to browse through Britney's images on a computer next to mine. While our index fingers were scrolling down together, our analysis-immersed bodies seemed unaware of each other's presence. These moments soon became our everyday activity. One hour of internet a day became a strategy to keep our minds sane.

After two weeks of reloading and anticipating new releases of Avril's images, my skin started to itch. I started noticing the texture of the screen on top of the images grid: flat, shiny, unreal. I was bored.

The author and poet Ocean Vuong once read somewhere that beauty has historically demanded replication.[1] That, in order to keep it alive through time and space, we make more of anything we find aesthetically pleasing. In this way, we allow ourselves to look at it over and over again. We (try to) make it last.

Unwilling to give up the beauty of my Avril appreciation just because I got bored, I started looking for ways to break out of my boredom. That's when I spotted a *printer* in the café. This printer marked the beginning of my curatorial practice, one that emerged out of my intense 'studies' of Avril Lavigne. I started printing, then organizing a selection of images born out of various Avril settings: stage moments, backstage moments, collaborations, hangouts, dates. Right-click → Save Image As... → Left-click → Save to my personal folder called *Maisa's Avril*, lounging cozily on the shared computer's desktop. Double-click to Open Image, Click Print... Printing. Meanwhile, hold left-click and drag the folder to Recycle Bin. Right-click → Empty Recycle Bin. (I was afraid that the next user on the computer would steal the folder without having done all this research, curatorship and clicking that I was doing.)

I wanted to intellectually *and* physically own all of Avril's images, to show my peers what it means to be the *biggest* AL fan. It didn't matter that I was not surrounded by that many Avril fans, because deep down I knew that information travels quickly, and that I'd start to recognize them sooner than later. They would appear to me. Through bracelets, pink highlights, skull prints. Through chains worn around skinny necks and wrists.

1 Ocean Young, *On Earth We're Briefly Gorgeous*

As expected, Avril's visually documented life became an ongoing topic during after-school hangouts with my friends. We'd ask whether anyone had seen pictures from her latest concert in LA. Or if we thought her nonchalant street outfit meant that she was breaking up with Deryck, her boyfriend.

Archiving images became a kids' power play. Whoever discovered the latest images of her first got to be the winner—whatever this entailed.

As new strategies for practicing the hobby of archiving started popping up, I found myself questioning my research methods. This discontent led me to look beyond Google's *All Images* and peruse for information in the web's gutter. I was specifically interested in finding images with extraordinary gestures of guitar smashing, facial expressions, middle fingers, and preaching about stuff in some sort of punk-style. I managed to compile a folder depicting who I believed Avril *really* was, or at least who she had been at the start of her career. Finding these gems would lead me to one day 'become a winner'; again, whatever that entailed. Soon enough, the internet became a visible, rather tangible matter within my local surrounding: bracelets, pink highlights, skull prints, and chains. Skinny necks and wrists.

> Where the capitalist class sees education as a means to an end, the vectoralist class sees it as an end in itself. It sees opportunities to make education a profitable industry in its own right, based on the securing of intellectual property as a form of private property. To the vectoralists, education, like culture, is just "content" for commodification.
>
> — McKenzie Wark, *A Hacker's Manifesto*

One boredom followed another followed another followed another. Consuming information in a fast pace made me feel homeless in my own present, but hopeful about feeling settled in my future.[2] My physical appearance manifested the information uncovered through all of my user curiosities: they caused me to wear black, to feel blue, and to be anorexic for seven long years of my teenage life.[3]

At the age of thirteen, I got tired of manifesting these 'informational changes' onto my body so literally. I needed less obvious methods of relating to my subconsciousness. As I was searching for a new way to relate to the uncovered online information and my addiction to excavate it, I stumbled upon content-heavy Anorexia forums—a stream of information that soon became my favorite place for digital information.

Learning that deliberate hunger is a way of saying *no* to whatever the disordered is trying to negate felt just right. An act of reduction (of the body, of the self) seemed like a way to get rid of the information that, at that time, defined me so clearly.[4] I longed to be something other than I was. De-creation, *of* and *for* me, was about to be cultivated as a new form of relationship with the information entering my life as a teenager.

2 'The consumer must never feel completely at home in his present, or he will stop striving toward a more fully satisfied future' — Douglas Rushkoff

3 'All abstractions are abstractions of nature. To abstract is to express the virtuality of nature, to make known some instance of its manifold possibilities, to actualise a relation out of infinite relationality. Abstractions release the potential of physical matter. And yet abstraction relies on something that has an independent existence to physical matter—information. Information is no less real than physical matter, and is dependent on it for its existence.' — McKenzie Wark, *A Hacker's Manifesto*

4 'Anything a female person says or does is open to "interpretation." If the female anorexia isn't consciously manipulative, then she's tragic: shedding pounds in a futile effort to erase her female body, which is the only part of her that's irreducible and defining.' — Chris Kraus, *Aliens and Anorexia*

Through these anorexia forums I learned about the many subjects one can negate, often being the image of oneself, and parental control. Many teenagers who wanted to gain control over their young lives found comfort in practicing eating disorders. They could eat and puke as much as they wished, in order to reach the weight that best defined the parameters of the space they wanted to occupy. The size of their bodies, according to them, manifested their relation to their domestic context. The healthier the body, the more representative of its healthy environment it is. In contrast, sick bodies are independent of the environment in which they are rooted and where they mostly operate. Sick bodies bite the hand that feeds them. Hard.

The best way to share a personal 'progress' of shrinking with other users was to create an Ana story, a YouTube video made up out of images representing an authentic experience of the disorder.[5] Most videos were accompanied by sad music playing in the background.[6] Each story was told similarly:

> (intro) What did the subject look like as a healthy and normal person?
>
> (rising action) Until they discovered Ana online,
>
> (climax) Then scaled down to their lowest weight (average was 112 lbs)
>
> (falling action) Ended up in a hospital,
>
> (conclusion) And finally went back to being healthy and normal.

5 Ana is Anorexia. Similarly Bulimia is Mia, and Eating Disorder is, if not otherwise specified, Ednos.

6 The most popular song being 'She's Falling Apart' by Lisa Loeb *https://www.youtube.com/watch?v=FS48eB3NAKI.*

A lot of these stories were not fully developed before they were shared. If an Ana video story ended with the disorder's climax—the lowest weight—it meant that the situation will end in one of two ways: recovery, or death. There was something alluring about a video ending with an established definition of the climax point. Elaborate were my many speculations about how their stories ended. I nourished the need to discover how the story ended, and kept coming back to some of the YouTube accounts.

Most of the sufferers were my age and came from America, a land I learned to be the land of excess. From the pictures I could get an insight into their rooms, families, pets, domestic vibes. Most of them took pictures of their progress from their closet, a safe place to hide in case one of their parents walked in. The outdoor scenery around their house always seemed vast, but also rather empty. There was not much going on, except more lawns and houses.

Through these images, I could feel the comfort of the lights projected by their screens. But I also felt their loneliness; how there was no one's reaction to the video that could validate it.

The place where I myself was growing up did not occupy that much space on an urban scale, nor did it feel like my body was a manifestation of its size and the various eating rituals it hosted. Thus it was hard to point out the exact subject of my negation to eat: sometimes I would blame it on my parents, sometimes on missing my home country. In hindsight, I think it might have just been solidarity all along.

Even though I never shared my progress in these various threads, reading about the users' obsessive practices of

calorie counting, body weighing, controlled eating and puking gave me a feeling that if I participated, I could find a sense of belonging. To belong, I had to learn how to occupy the least amount of space in the room.

As my body was shrinking, the feeling of being part of the online community grew stronger. Once again I longed to display my learned information onto my body and into my surroundings. Weighing 45 kg at the lowest, I wanted to occupy the least amount of space in the many rooms I encountered.[7]

It was not boredom that urged me to put an end to this de-creation of my own body—for this process was very tricky to beat. It was the malnutrition, which stopped me from practicing other passions I found more important in life. Next to this, it was the moment I read the *unable-to-recover* story of a 30-plus mother lamenting about the mental pitfalls of eating disorders, hoping she wouldn't pass it on to her daughter. It was her story that got lost in the thread, because she didn't specify how much she weighed, nor how many calories she ate per day.

What I learned back then, is that information is not knowledge. A lot of information showing the reader which metrics and/or numbers one must reach to 'prove' that one does it right is, in the end, just a race to the finish line.[8] An anorexic person becomes successful when she shrinks to the point of being hospitalized. In

7 For a 177 cm tall person, this was bad, it amounts to a BMI of 14,3. A BMI below 17.5 in adults is one of the common physical characteristics used to diagnose anorexia. There are also different tiers of anorexia based on BMI ranging from mild (<17.5), moderate (16-16.99), and severe (15-15.99), to extreme (<15). A BMI below 13.5 can lead to organ failure, while a BMI below 12 can be life-threatening.

8 'He or she in some sense is comfortable because this world is so neatly organized and predictable, and value is placed on all the things the anorexic places value on (namely, food weight, and symptomology). For acute anorexics, everything becomes means for measurement.' — Kelsey Osgood, *How to Disappear Completely*

the hospital, she meets other anorexics who are counting and obsessing over their digits. Yet within this hospital context, the rules of the game change: instead of the lowest weight, death is the final destination of the race. The winner gets to die.

After seven years of practicing meditative hunger, the underlying irony of the situation helped me pull myself out of the race.[9] Recovering from one eating disorder, and then another, challenged me to question how my identity could be based on finding meaning in nothingness, within the shrinking of my self. Recovery meant letting go of information that materialized nothingness, an emptiness that, as a user, made me feel so strangely fulfilled.

9 'Adolescent sexuality, which employs ignorance as aphrodisiac, can become a girl's deepest experience of Marxian alienation from the fruits of her labor: as soon as a girl understands what she is doing, she is no longer fit to do it.' — Elizabeth Wurtzel, *Bitch*

Simply Smart

What is nothingness when digitalized? Metrics.

Let's go back to the bootcamp coding school class. I couldn't help but wonder: 'Really? Are we still here as users? Still only asked to share and organize our metrics?' I guess my experience of information breaking me down made me wonder what needs to be built in its place. Let us first break down the role of a web developer according to the given study material.

A web developer is smart.[1] He writes semantic code and knows all the HTML tags.[2] He works a lot. Forgets to take breaks. When his code is broken, he likes to punish himself ever so slightly by not sleeping. One week before the project's deadline, he will work until four in the morning every night. It has a strange effect on his skin, a condition he likes to justify as a reaction to the seasons changing. In his sleep, he often dreams of unresolved work-related issues. Sometimes he finds solutions in them. He remembers these. He starts his morning by applying solutions to the code, just to see if he was right in his dreams. He agrees to be slightly addicted to being right. He needs this validation, not only from his computer, but from people as well. He wants them to say that he works *a lot*. In this way, he doesn't have to say it himself, and it

1 'For the mainstream thrust of anti-intellectualism, as it stands today, characterizes thinking itself as an elitist activity. And even if one were to get excited about leaving the contortions of mental effort behind, today's anti-intellectualism makes no corollary call for us to return our fingers to blood and dirt, to discover orgiastic block, to become more autonomous in our ability to fulfill our basic, most primal needs, or to become one with the awe-inspiring forces of the cosmos.' — Maggie Nelson, *The Art of Cruelty*

2 Writing clean code which describes the computer what each text component represents; i.e. titles are put in <h1></h1> tags.

gets easier for him to focus if they know that he's currently not available. That's one of the reasons why he works *a lot*. Average web developers dream of working remotely from Thailand; true web developers dream of building their own software one day. In terms of location, they like to stay in close proximity to their money.

If you're on a path to become a web developer, this is what the bootcamp coding school will encourage you to become. If you're in a tricky bureaucratic situation, just like I was, fears will guide you into studying the material more closely, just as I did. But I was also lucky one day, when the most skilled coder in the class told me that web development is all about copy-pasting toward the client's needs. She did not study the material seriously, nor had any bureaucratic reasons to do so. She was a self-taught web developer who made me wonder what I'd choose to learn if I were to build my coding practice independently from the educational model thrown at me as a form of introduction to the field.

Getting insight into the tech industry taught me certain things I've mentioned before: how the 'true' web developers featured in my courses would, when hired by corporations or start-ups, mainly do the work of infrastructure maintenance, and not the expected expression, innovation, or experiment.

Let's question what *smart* means in this case by looking at another example: a young web developer.

In December 2020, I decided my Instagram feed needed more coding-related content from aspiring web developers. After typing 'web developer' in my search bar, the first account that popped up was that of a boy who, according to his bio, *hacks sh*t together*. At that precise moment, he became my new protagonist.

According to the metrics, he had 37,7K followers. Tapping on the latest image welcomed me to a very long scrolling session. As I was scrolling, I studied the bits surrounding each post: dates of posting, the written descriptions, hashtags, the camera's angle, filters, and even some of the comments. It seemed like a very eventful account.

I started investigating with my third eye—my editor's eye. A new picture was posted every three to six days. While indulging them, my first impulse as a user was to subconsciously categorize them as minimalistic. The saturated filter applied to these images added a spooky, cinematic atmosphere to my new protagonist's room. Nighttime shots gave me the sense that there's some *serious hacking* going on. And that it's pretty quiet. Most of the pictures showed my protagonist coding, checking their phone, looking out of the window, stretching, standing, and relaxing in front of their working station: a desk containing a laptop, a big screen, a keyboard, a mouse, a plant, and only sometimes a robot toy whose brand I couldn't recognize. From time to time, an acoustic guitar or electric bike seemingly broke the monotony of the repeated still life. The room seemed to be the only context in which my protagonist operated. In all of these images, the coder is never *not* present. In all of these images, the coder is never *not* working.

Each image was accompanied by a sort of written motivational letter. Some of these texts wished to reveal tips on how to become a successful web developer in just three steps, while others described the protagonist's learning trajectory, listing the tools and technologies he was currently teaching himself. He was keeping up with the modern frameworks and rapid changes of the tech industry.

In one of his posts, posted just before the end of 2020,

the protagonist shared with his followers that his end goal of the year was to make his web app completely bug-free. And as for 2021? He wanted to reach 100K followers. Unsure of whether this was the content I was looking for, I followed the account anyway. This might just be the push I never knew I needed, I thought to myself.

My protagonist's posts appearing on my feed made me curious about his audience. I started reading the comments from his followers and fans: besides fire emojis and multi-colored hearts, there were a lot of desperate questions, urgently asking for more tips on how to become a successful web developer. Tapping the names of his followers and supporters revealed a niche of very similar accounts. On these, my potentially new protagonists were posing next to their work stations, writing comparable motivational quotes, and settling for similar commodity comforts: a motorbike, a guitar, an ergonomic chair, an overpriced yoga ball. All of these profiles had enormous amounts of followers.

Unwilling to let my reflections end there, I decided to reach out to my chosen protagonist. After introducing myself as a (senior) researcher on *Working Trends*, I told him I was interested in hearing more about the methods through which the production of work and the production of its online representation merge together within the culture of influencers. A young web developer seemed to be an ideal figure to base my study on. *On a further note*, I told him, *I found you whilst looking for motivation to wrap up my own coding projects in 2020*. Before asking actual work-related questions, I wanted to know his age.

At the time of our conversation, he was almost eighteen. He told me that his interest in the tech & programming

field dated back to when he was twelve. Passion led him to join a bootcamp school where senior developers taught kids the basics of programming both hardware and software. In his further coding endeavors, he focused on deepening his skills in HTML, CSS, and Javascript.[3] Learning faster than he had anticipated made him lose interest in the ever-present, collective learning through coding workshops, so he decided to build stuff from home. Pure DIY energy of a self-taught coder, it was. Working solo from home gave him the time and headspace to build his own web product, which, according to him, worked the best for his specific needs.[4] It sells in 135 different countries and, at the time we were chatting, reached almost 40,000 users. User dependency made him upgrade the product to a Premium version, which also made it possible for users to support him in maintaining the service.[5] Working simultaneously at a company and a charity, his long-term goal was to make his products profitable enough to live off them. He had been earning money since he was sixteen, and continued to do so.

But how does he manage to maintain his IG account and a K amount of followers next to such a huge amount of actual work? He told me that he created his account back in the summer of 2017. Throughout 2018, the number of followers grew exponentially, even without any IG tricks (purchasing bots, hidden hashtags, follow for

3 Basic tools for building a website.

4 'I created one web app in particular, called YouTubeDLD, which (I like to claim) is world's best YouTube downloader on the market without ads, spam and limitations.' — My protagonist

5 'I recently introduced the Premium version, which allows users to support and help me to maintain the service by giving them the ability to download higher quality using my own encoding algorithm. In theory this means you can download 4K videos in 4K quality (which YouTube itself even doesn't allow its users to do).' — My protagonist

a follow back and subsequently unfollowing, and so on). Although he tried, he never really worked with sponsors. Instead, he spent chunks of time connecting with the programming community, all the while posting consistently. The digital maintenance of his IG account soon became a natural habit.

When I complimented that his bio 'I hack sh*t together/ I'm a web developer & designer ' sounded quite convincing to newbies like me, he confessed that he actually invented the trend of presenting himself as a young web developer at the age of fifteen. After writing '15-year-old Web Developer' in his bio, it didn't take long before other 15-year-olds picked up on this age claiming technique. However, he did not continue editing the number while growing up: he didn't find it that impressive anymore to be a 17-year-old web developer. Not because he had aged slightly, but because the tech & programming community grew insanely fast in just a few years, meaning that even younger (plus way older) generations joined continuously.

My next questions pushed him away: *Do you use any filters or physical tools in the actual space to make the images appear to be in the same light?* And if so, how much time do you spend on the post-production of an image for an IG post? Is the repetition of style important to you? And if so, could you please elaborate on why? Is posting every three to six days your natural flow, next to your other work? From one of your posts, I learned that you're still in school and that it feels like a joke to you compared to your other work. How do you organize the time to do both? Do you work in the evenings, and do you have time to do homework next to it? Being 'successful' in both fields (web-dev and IG) makes me curious about which one you invest more time in, your IG account or your work? Do

you have a registered company? Feel free to skip the following questions. Where do you find your clients? Do you have accounts on various freelance platforms? What is your social life like? Do you ever get tired? Would you consider hiring an assistant? Is programming and managing your IG account something you see yourself doing for a long time still? If you could give any advice to your generation, what would it be?

As if the algorithm could feel the death of our conversation, it started populating my feed with fashionable web-development content such as reels that hypothesized how programming languages would look and behave if they were a (female) human. At some point, I started getting content from 40-plus men sitting in front of their work stations, giving the camera a thumbs-up, writing motivational descriptions and doing exactly what the younger ones do. It was then that I realized why my protagonist opted out of his own trend. When I clicked on one such profile to check the number of followers, a similar K amount appeared. It works either way, I thought to myself, and decided to train my algorithm differently.

Both examples of a web developer make it obvious that the ultimate form of any work is fashion.[6] Just *claim* to be a front-end web developer. Claim to be a full-stack web

6 'We live in a world with multiple timescales, all moving simultaneously but at different speeds. Brand calls it the order of civilisation. Nature, or geological time, moves the slowest—like the skater in the middle of the pinwheel. This is the rate at which glaciers carve out canyons or species evolve gills and wings—over eons. On the next level is culture, such as that of the Chinese or the Jews—which lasts millennia. On the next concentric ring comes governance—the rather long-lasting systems of monarchies and republics. The next level is infrastructure—the roads and utilities those governments build and rebuild. Faster yet is commerce that occurs through that infrastructure. And finally, the outermost ring is that of fashion—the ever-changing styles and whims that keep the wheels of commerce fed.' — *Present Shock*, Douglas Rushkoff referring to Stewart Brand's book *The Clock of the Long Now*

developer. Claim to be a creative developer. Claim to be a UX designer. Claim to be an expert. Claim to be a female web developer. Claim to be a creative coder. Claim to be a web architect. Claim to be a web artist. Claim to be a critical web developer. Claim to be a code writer. Claim to be Ania Kubów. Whichever one you choose,'you will be organized, you will be an organism, you will articulate your body—otherwise you're just depraved.'— Sadie Plant, *Zeros and Ones*.

Is a web developer considered smart because of his dream to build a software, his eagerness to learn new languages, his industry-compliant nature, or his portfolio? All of these points depict the workload of maintenance. Can a web developer be considered smart, simply because he is bored of building the interfaces he taught the users to demand?[7] In my opinion, a smart web developer knows how to take a (short *and* long) break from coding, without a sense of guilt.

7 'Personal content: I feel like they are considered 'smart' when they use a new partly unsupported experimental property that makes my 2015 computer burn through hell, but then they place a pop-up telling me I don't deserve to get access to the content because I don't want to throw away my decently functioning computer for a new one.' — Unnamed Web Developer

Scott Gives Me an Assignment

Where the capitalist class sees education as a means to an end, the vectoralist[1] class sees it as an end in itself. It sees opportunities to make education a profitable industry in its own right, based on the securing of intellectual property as a form of private property. To the vectoralists, education, like culture, is just 'content' for commodification.

— McKenzie Wark, A Hacker's Manifesto

One day last year, I was sitting on a comfortable office chair in Scott's studio, looking out the window after twenty minutes of staring at a screen. The clouds over the tower cranes looked creamy; their image absorbed my slippery thoughts. I was supposed to regain my focus, prepare my ears for the assignment Scott was about to give me. Shifting my gaze towards the tower cranes helped—their image reminded me that, at this point in civilization, I will not live to see the day that these clouds *aren't* accompanied by cranes.

Some sort of 'private' social media platform was being built for the students of a renowned academy overseas. The purpose of the platform was to connect these students by allowing them to set up personal goals on their accounts, follow classes, events, or other students, and to share relevant media objects and references in their shared 'baskets'. To connect

1 The ascendant power over both labor and capital is the vectoralist class. It does not control land or industry anymore, just information.

them even more, an option for the students to set up a specific time zone was built. All data and information bits extracted from these interactions were to be used solely for educational purposes.

As most of these functions were already built and working, the project had reached a crucial phase in its development, which focused on user participation and connection—a phase I call the 'psychological touch'.

Scott was busy coding the invisible functions for online *what-if* situations. The code's logic behind most of the what-if situations is to make something happen (clicking) in order to extract bits. The question behind a what-if situation can be answered with at least three solutions. For example:

What if the user doesn't have friends?
1. Remind them.
2. Propose to connect their friends' list from another platform.
3. Suggest a friend.

What if the user doesn't go to an event?
1. Perhaps they could be interested in updates about said event?
2. You don't have to go but you can follow the event, which is part of a series, so maybe you can go next time.
3. Are you sure?

What if the user doesn't react to the picture of a cat?
1. Here is a cuter cat.
2. How about a reduced amount of cat content?
3. But all your friends like it!

What if the user doesn't move the cursor?
1. Wait for seven seconds, then show a video of a cursor on steroids. The longer the hovering, the bigger the cursor will grow.
2. Wait for seven seconds, then make the menu bounce.
3. Just let the ad appear with a big X button.

By answering these questions, the web developer's forever-aim is to make the user *do* something and populate the website with information that represents activity and traffic. Because most platforms are slightly alienating at first, these situations usually occur at the beginning of a platform's life—before the platform gets populated with the users' digital traces.

When the website gets busy due to the compliance of its users, the responsibility of the what-if codes passes onto the users, who then become the new psychological trigger for continuous user traffic. When users see some of their friends liking a post, they will probably like it too. In this case, the what-if code doesn't need to remind them that they have the digital superpowers to like a post.

These are general examples from the worldwide web. Some streams are relevant and some are absolutely irrelevant to the user who, by now, is embedded on the platform enough to be able to sense the information's relevance, and who feels alarmed when information lacks the promise of relevance's continuity. In the case of an educational platform, things get a bit trickier. Why would students *not* want to participate in online activities their educator provides? I see this as an argument Scott can easily win, for students want to score in the eyes of the institution that supplies their online educational activities.

Winning this argument gives Scott eligibility to control

how connected he wants the users to be. 'Good' web developers are expected to reduce the duration of user alienation by making the users generate bits as soon as they land on a platform. These class-A developers are deeply aware of the detours the users can potentially follow on a platform, and can guide these users into walking a 'straight line' by coding their behavior in advance.

Before giving me the assignment, Scott expressed how he wanted me to code the actual functions for the platform, but also added, with a smile, how I'm *just not there yet*. Until I gain eligibility (during my free time after work, that is) let's just work with what we have, shall we? Let's give this website a tone, for the English language is the one understood by all.

Unsure whether I sensed his pride, or simply guilt for giving me work he would rather not do himself, I directly followed Scott's instructions. My task was to write sentences that point out the user's inactivity and ~~trigger the user to generate data~~ motivate the user to share references, go to events, connect, and contribute to the overall knowledge production.

As a reaction to my excitement, Scott preached how user motivation could only be achieved through a language that *avoids* literally describing what the user isn't doing, but rather describes the consequences of their inactivity on the platform.

I *roger'ed* that. I used it as a mantra throughout the whole working day, which I spent writing a presentable version of the following list:

There are no upcoming courses you're a follower of.
(FACT)

You are not following any upcoming courses. (FACT)

Courses that you follow will be listed below. (SUGGESTIVE)

To see the courses below, you must first follow them. (TOO SUGGESTIVE)

There are no events scheduled this week.

There are no events scheduled for next week.

There are no events scheduled for the selected week.

You are not attending any events this week.

You are not attending any events next week.

You are not attending any events in that particular week.

Events which you are attending will appear below. (SUGGESTIVE)

Students, faculty, staff from the same time zone and of similar interests you follow will appear below/here. (SUGGESTIVE)

Activity baskets related to the courses you're following will appear below/here. (SUGGESTIVE)

There are no people from the courses you followed.

There are no baskets you're part of related to this course.

This person doesn't follow any courses.

You don't follow mutual courses with this person.

This person has no mutual courses with other people yet.

This person has not contributed to baskets with other people yet.

This person doesn't own any baskets yet.

This person doesn't contribute to any baskets yet.

You are not following any course related to this basket yet.

You are not a part of any basket related to any courses.

You are not.

It took a rather long time because I hated it, for similar reasons as to why I hate Wordle.[2] While re-reading my written suggestions what bothered me most was the impersonal tone behind them. Somehow, I couldn't make them sound as if they were written by Scott's guts, or crafted by his brain. Polite is hard—I screamed long and hard inside my assigned, linguistic trap, and I felt like a loser for not having a *natural* sense for it.

But let's get one thing clear: within the web development scene, it doesn't matter if politeness is a manner exercised in a web developer's daily life. It's a globally recognized strategy for building a platform. If he doesn't make his product speak politely to the user in order to get the so desired bits, the product is at risk of sounding either too feminine or too authoritative—the two tones that, by 2022's default, carry the risk of the user's flight mode. Let's deconstruct these web trends.

2 Its software design feels fun, at first. Guessing the word of the day right every single day of one's life can make one excited about waking up. But it can also overwhelm one's experience of daily life as it consumes good chunks of one's morning time or, depending what kind of a player one is, the whole day. Don't play Wordle. It's an attention sucking platform that makes you forget about the things you'd rather do throughout the day and prioritize attaining the joy of beating your pals who scored just as good as you did. The worst part of it is that cheating feels terribly wrong. And that I still sometimes play it.

Feminine-sounding software doesn't shy away from using their technical weak points as a digital marketing tool. Here's an example of how they intentionally sound vulnerable:

Whoops 😅. Please keep on patiently clicking the button until it works!

At the other end of the spectrum, we see an authoritative tone in some software that triggers fantasies of digital suicides, but never their commitment:

It's someone's birthday. Help them have a blast!

or

Are you aware that [user name]'s post was liked over 10 times already?

One user, confronted with his inability to opt-out, texted me: *Fuck Insta, but I'm still on it.* Fuck smartphones, but I still own one.

The neutral, polite tone of software protects it from sounding different—sounding like too much of something. The factual tone of politeness targets devotion of a user (who, if you recall, I've introduced before) and who needs to be promised that, as a respected client on the platform, their needs are constantly taken into account. Even if they're in fact not always met.

I need a small detour for a moment. In her book *All About Love*, bell hooks writes about respect (through a dysfunctional, yet loving family) in a way that resonates with my own reflections about how respect is gained within any kind of relationship. As we know from life's lessons, there are two possible intentions behind giving respect: the first one is established through the genuine

love and/or care for the other, and the second one is constructed towards the demand for a specific amount of reciprocal respect.

The first one, as hooks sees it, is given in the context of an intimate, ongoing relationship between two parties—a parent, a friend, a friend with benefits, a lover, spouse, collaborator. Their roles or titles are established over time. In the case of the latter—respect in return for respect—two parties don't necessarily *need* to get deeply involved or spend any time with each other for them to *want* the desired benefit from one another. Their roles or titles are set by default. They stay together because of a mutual need for the bits, thus proving the professionalization of their role/title: a web developer, a user, seller, buyer. Quantification makes them stronger.

If I were to apply these same lessons regarding respect to web development, I'd find it difficult to conclude that the web developer's intention behind welcoming the user on their platform comes out of pure *generosity* towards the user whose image, in the developer's eyes, is shaped by the statistics generated from their compliance.

Please read:

There are no baskets you're part of related to this course.

The sentence suggests that the reason why I'm here (following the courses I want, in order to be a good student) is inherently linked to these baskets I know nothing about, but about which I apparently *should* find out, such as why I need to carry them everywhere I go. Wait: let me ask one of my few friends on this platform real quick. Until I find out, let me just speculate. When I think of real-life baskets, I think about what's in them… It might be that the digital ones related to my courses are

still empty? It might be that they're filled with a surprise, by default? Let me check. But, actually, if the surprise is really in there, I might not even want it. I'd rather have my friend drop one in there, but they haven't replied yet. Maybe they're following a lecture I wasn't updated about. Where are the events? Let me see…

This is how a platform gets populated and becomes a common user reality.[3] Quickly. Seamlessly. Almost invisibly. Assuming that a listening ear means being able to register everything that the user opinionates about, and that this user needs to see the information bits in order to feel heard, we see how 'polite' software entitles itself to collect the user data and activity it needs in order to increase user dependency. And because users build up important archives while generating traffic, they soon become dependent on platforms that store and organize their data for them. Naughty baskets!

This is not really news, so let's not dwell on these findings for too long. What I'm more interested in, is looking at the impact of this user dependency on polite software and the fashion of our conversations in real life. (And in stating 'our', I am in fact stating that we're *all* users, including web developers.)

3 'The scaling of the economy of gender features most prominently across discussions surrounding "big data.' For example, every forty-eight hours online we as a global community generate as much information as was generated in written history from the beginning of civilization until 2003.1) This data we generate triggers monumental questions about mass surveillance and how the information tied to our digital selves can be used to track our every movement. Our Internet search histories, social media habits, and modes of online communication—what sociologist David Lyon calls 'actual fragments'—expose our innermost thoughts, anxieties, plans, desires, and goals. 2) Gender binary is a part of this engine: a body read online as male/female, masculine/feminine fulfills a target demographic for advertising and marketing. Google Ads explains gleefully to its users how 'with demographic targeting in Google Ads, you can reach a specific set of potential customers who are likely to be within a particular age range, gender, parental status, or household income. For instance, if you run a fitness studio exclusively for women, demographic targeting could help you avoid showing your ads to men.' — Legacy Russell, *Glitch Feminism*

There are no baskets. There is one basket. There are two baskets. There are three baskets. There are four baskets. There are five baskets. There are six baskets. There are seven baskets.

(I think we get the point.)

While rating a product or confessing about romantic endeavors in a chatroom here or there or everywhere, polite software awards the user with facts that inform the user about what they just did:

You just left the group,

Thank you for your feedback,

OK,

Thanks,

OK, OK, thanks, thanks. OK, thanks, thanks.

Rate?

It's exactly 21:00 PM.

It's your bed time.

It's a fact that you just left the group.

It's true that you just left the group.

The user is continuously fed facts, and will pass these on to others (be it software or a friend). Facts are known to compile together the system of truth.[4] When polite software collects the extracts of truth—all the things the user talks about, rates, and reacts to wherever the user is told to react—the logic behind its code becomes stronger

4 'The way to flourish in a media space biased toward nonfiction is to tell the truth. This means having a truth to tell.' — Douglas Rushkoff, *Program or be Programmed*

and kills any potential to behave against it. You cannot just *not* rate a product, but you can give it less stars. Are you sure you want to close the tab without sending that e-mail? This is what I call an extremely organized user experience, a linear story. Like waking up, going to work, and going to sleep. Everything works perfectly.

Understood as such, polite software becomes a product of not just controlled language, but of the imagination that stretches this vocabulary. A language that should be in continuous transformation, molded by those who speak it, moving towards understanding each other, is a language that is now a standard: one that says what should be understood between the speakers, and how deep a discussion can really go.[5]

Moving along with such a user experience can be compared to our behaviours within urban landscapes. Isn't the point of going outside to get from point A to point B? In Amsterdam more than in Mexico City, or Sarajevo, for sure. Although I'm totally fine with not having a goal while walking, I've seen a lot of my peers struggling to say yes to the idea of an aimless stroll. Just like them, I feel users suffer from feeling lost on platforms that *don't* extract bits from their interactions. They simply don't how to move about without being part of both the urban and software organization.

As for Scott and myself: I do not know what Scott wanted me to learn that day. What I *did* learn is that

5 'Information, like land or capital, becomes a form of property monopolised by a class of vec- toralists, so named because they control the vectors along which information is abstracted, just as capitalists control the material means with which goods are produced, and pastoralists the land with which food is produced. Information circulated within working class culture as a social property belonging to all. But when information in turn becomes a form of private property, workers are dispossessed of it, and must buy their own culture back from its owners, the vectoralist class. The whole of time, time itself, becomes a commodified experience.' — McKenzie Wark, *A Hacker's Manifesto*

politeness is not only a design choice, but the rule by which a web developer chooses to live. When isolated from the product and placed next to the hands that crafted it, politeness becomes a shield that protects the acquired years-old knowledge from intertwining with the emotional impulses (passion, anger, desire...) that have the potential to cause its shattering. Because most of the time is invested in maintaining a stable structure of knowledge, these impulses are rushed, postponed, or even completely neglected. This is the core behavior that builds up the stoic man: the logic of the world robbing him of his boner.[6]

———————————————————————————

'The object of information, never a subject in communication'

— Sadie Plant, Zeros and Ones

———————————————————————————

6 Ariana Reines, *A Sand Book*, poem: 'Hegeling before the glass'

Wordpress Cringe

The reasons for a so-called 'WordPress cringe' that exists for both its users and the web developers working with it are understandable: the plug-ins break with each update, and the back-end interface is, well, pretty *ugly*. Web developers cringing at WordPress software made me wonder why a different software with the same listing system couldn't be a solution (such as, for example, Kirby).

WordPress is a web builder software that is, at the time of writing this, nineteen years old. When it was released for free in 2003, it soon became the market leader of the blogging tools industry, allowing both users and developers to extend its default functionalities by writing their own plug-ins and sharing these with the rest of the online community.

In 2005, the world discovers WordPress version 1.5, an incredibly flexible theme system that adapts to the user rather than expecting the users to adapt to *it*. This already hints at the fact that there were some software limitations, but what this specifically meant was that headers, footers and sidebars could be placed on specific pages and not necessarily on all of them, such as before. I guess some users didn't like feeling constrained by component repetition across pages.

By 2010, WordPress version 3.0 enabled the user to customize the styling of the default *Twenty Ten*-theme. The theme became an annual mirror to the user's taste in grids, and continues to update itself every year.

The year is Twenty Twenty-Two: The Most Flexible Default WordPress Theme of all times. Built to reflect

the modern user needs, it is inspired by birds, offers six color palettes, and encourages as little use of CSS as possible—because *Global styles* are just a few clicks away. Apart from style, WordPress stays focused on improving a full website editing experience, for its future direction solely depends on the needs of millions of web publishers around the world, a.k.a. its users.

Years of WordPress' changing according to the user's and web developer's needs made WordPress survive as the leading publishing software on the internet, with more than 43 percent of all websites built under its wings. So, why oh why, do users *cringe* so hard at the software that updates itself recursively alongside their digital wishes? To answer this question, I'd like to explore the technical, stylistic, and personal perspectives of Wordpress' cringe.

1. What is Technical Cringe?

> Openness is an unruly concept. While free tends toward ambiguity (free as in speech, or free as in beer?), open tends toward obfuscation. Everyone claims to be open; everyone has something to share, everyone agrees that being open is the obvious thing to do—after all, openness is the other half of "open source"—but for all of its obviousness, being "open" is perhaps the most complex component of Free Software. It is never quite clear whether being open is a means or an end. Worse, the opposite of open in this case (specifically, "open systems") is not closed, but "proprietary"—signaling the complicated imbrication of the technical, the legal, and the commercial.
>
> — Christopher M. Kelty, *Two Bits: The Cultural Significance of Free Software*

When a software cracks open and enables web developers to access and expand its set of functionalities through modifying the default code and contributing to it with plugins, it makes itself technically vulnerable. It leads to standardization, civilization, exponential growth:

> Standardization was at the heart of the contest, but by whom and by what means was never resolved… We don't live in a world of "The Computer" but in a world of computers: myriad, incompatible, specific machines.
>
> — Christopher M. Kelty, *Two Bits: The Cultural Significance of Free Software*

Under the current conditions of the tech market, all of this/the software becomes a playground for competition; a potential space for exploitation. Not just a few, but *many* of the plugins that perform the same task, all coded by amazing web developers who don't really communicate with each other, end up competing for what the study material of these developers taught them: user attention. Each plugin, instead of finding itself one too many to choose from, finds itself wanting to be one in a million.

Openness is an unruly concept. The price of software's freedom is the user's unfriendliness: the technical slowdown that comes with every new software update.

Here, we end up confronted again with the simple fact that there is no *universal* user, although most of our software expects there to be. Some users are more attuned to the tools and technologies they're using, while others prefer to stay busy with non-technical things. Some users are curious enough to learn more about the back-end and maintenance of their digital presentation, while others prefer their websites to be independent.

The ecosystem of WordPress puts most web developers in a position of not questioning or even understanding the extent to which their clients and/or users want to express themselves on their respective digital interfaces. How do they want to use the back-end? How often? What is the function of the website?

Only when true user needs are understood can technology become a cultural meeting ground upon which we can look away from the image of a universal user and learn about their socio-political situation, class, and culture through their relationship with technology. This takes time, for this often constitutes the unpaid analysis of the process of web development. And because this analysis is not per se encouraged by the web developer during the whole development process, he opts for a standard setup of the back-end, one that leaves the users in fear of clicking that crucial update button.

2. What is Stylistic Cringe?

Because the WordPress's community shares all issue-related matters, the users can easily find out what exactly is wrong with the issues they're having. Sometimes the most curious of users even go further in researching possible solutions than the web developers do. Because of this transparency, developers refrain from identifying with the software and would rather opt to settle for a less user accessible techne-pack, or their own proprietary version of WordPress (such as BlockSmith).

This rarely discussed reason behind the web developer's pride has a cleaned-up version: instead of confessing to their opinions, a web developer finds themselves persuading their clients and/or users that the interface design of WordPress is *just horrible*. Because of the personal sensitivity that style installs in every user, the

user is easily persuaded and charges extra for the time it took their (presumably male) web developer to build their own CMS.

Irony alert: everyone cringes at the underlying fact that WordPress is accessible to all in the blink of an eye and can make anyone feel like they are able to code and build websites. I, personally, don't see why anyone *shouldn't* just build their own websites.

3. What is Personal Cringe?

Besides being the very best version of a Post-WordPress web developer I can be today, just because it felt cute, my reasons for *cringing* will most likely bring me to delete this identification later. Throughout my cringe studies, I was hoping that someone would complain about the listing and/or archival logic of the WordPress and WordPress inspired softwares, and how its modularity promises a false freedom that one eventually ends up feeling a victim of, or at the very least dubious of.

My cringe derives from confusion: I'm still puzzled by the technical differences between proprietary CMSs and WordPress; I still wonder why (and this is probably a question of governance) the users cringe when openness is technically allowed according to their wishes. Let's talk about this.

Post-Wordpress Web-dev, FUCK IT.

Psst, it's me. Your Post-WordPress web developer. Yeah, I'll tell you a bit more about my story.

(intro) What did the subject look like as a healthy and normal person?

It all started the day I created my first social media account. Having an account brought along with it the possibility to fill up my gallery with personal pictures, select and list top friends on one's profile, and fill in the lists asking about one's favorite movies, foods, and miscellaneous interests.

Nothing was blank on my Hi5 (Balkan Myspace) profile. I was crafting and re-crafting it daily. My favorite part was customizing the wallpaper through copying codes from other platforms in the platform's hypertext boxes. I went through many different kinds of wallpapers: from clouds to black and white skate parks during sunset. Once I entered my 'pitch-black phase', I focused more on the quality of my lists and made sure they showed some good taste.

Styling one's profile was the most enjoyable part of my user's experience. Profiles were exchanged amongst users as exemplary references to look at while building one's own.

Today, I can't tell if enabling the users to apply aesthetic changes on social media platforms (which were ultimately designed for user connection) was a deliberate aspect of the platform's design, but I do believe this was the reason why the switch to the currently most dominant one (Facebook), a platform where aesthetic customization is *disabled*, was so easy.

Between this moment of stylizing my wallpapers in 2009 and my bureaucratic situation in 2019 lies a huge gap in my coding practice. Attending the bootcamp school bridged this gap: I started building websites from scratch, not just inserting codes on someone else's platform.

The first few small-scale websites I built were handmade—meaning they were built with HTML, CSS, and Javascript; basic tools for building any kind of website. Learning how to build a button, for example, already made me realize that the design of this button can rather easily manipulate the user's mood, akin to how a doorknob of a random door would. Where does the door lead to? Is it locked? What's lurking on the other side? Should I *open* the door? Multiple, diverse types of buttons leading to a manifold of pages made me wonder about the amount of space a website occupies within various digital infrastructures.

(rising action) Which software affected the subject's condition?

At the time when I started building websites for clients, WordPress was the go-to software to use in order to get a sense of the server-client traffic behind a website. In the beginning, I charged very little and said yes to any project or deadline. The projects were simple; the deadlines extremely tight.

Because I felt rushed, I used theme-building helpers (such as Divi) to meet the client's expectations in time. The design would highly depend on the limitations and freedoms of the software's given functionalities: using theme builders that had an easy drag-and-drop functionality to create the looks of a digital space constrained my creative output. The limited possibilities these builders provided was not satisfying enough to feel creatively fulfilling to someone who wanted to break out of the default grids and blocks, a.k.a: me. Not *seeing* the code that built my decisions felt weird.

After I built two websites with extra builder products, I felt like a scam to the world of web development. I felt like *any*body could do it and that *every*body was doing it. As I was claiming the title of a web developer, I felt like I had to live up to it. But also, I just really liked coding. So I started picking up on PHP language and coding custom themes, after which I finally started feeling like a real web developer (whatever that means).

Identifying myself with this expert software in the company of other web developers made me feel quite shy at first, but upon discovering that the majority of web infrastructure is built with this same software at some point started making me feel average.

As I was building websites, it seemed to me that the software was updating itself according to the user's needs really fast, and that whatever users lacked in their editing freedom soon became an option in a plug-in library: a true catering to their needs. It is true that WordPress software updates not only grow alongside the needs of its user, but also follow the web developer's imagination, which, when turned into a plug-in, can be uploaded to the WordPress Plug-in library. That's what open-source means; contributors add themes and

plug-ins to it, for software improvement. 'All technology reflects the society that produces it, including its power structures and prejudices.' — Douglas Rushkoff, *Program or be Programmed*.

I don't know if being content with theory justifying my software choice was naive, but I was having fun. I was entertained by the underlying irony of the previously explained WordPress cringe.

(climax) How did the subject liberate oneself from the software's constraints?

That's when I became a Post-WordPress web developer. As a Post-WordPress web developer, I knew what I *didn't* want from my coding endeavors: plug-ins (unless custom-coded), hamburger menu (unless absolutely necessary), WP default styling (a.k.a. classification and names according to the semantics), a deadline within the month, and having to say yes to the reducing amount of pages as well as meeting the client once a week. I wanted to make rules that would bring me closer to building *light* websites at a *slow* pace—what my friend Clara calls 'handmade coding'.

With handmade coding, I could build a website like I'd imagined building a space back in the days of studying architecture: hand in hand with the inhabitant or citizen, the user, and their individual taste. Our collaboration would mean meeting and building each other's differences, on whatever interface necessary.[1]

The *post* in my title refers to the fact that even though I am aware of the WordPress cringe, it's the tool I choose to operate with despite its reputation. That in

1 *https://motherfuckingwebsite.com/*

my productivity I have found liberation, not software enslavement; that WordPress will succumb to my needs and accommodate the skills necessary for me to have whilst collaborating with my client.

Here comes an example of my liberation. The following code was written for the client's portfolio website and in agreement with the client:

```
<details>

      <summary style="margin-left: 50%">editor</
      summary>

      <?php $loop = new WP_Query( array( 'post_
      type' => 'editor', 'posts_per_page' => 10
      ) );

      if ( $loop->have_posts() ):

            while ( $loop->have_posts() ) :
            $loop->the_post();
            ?>
            <details class="secondary-details">
            <summary class="project" id="editor-
            project" onclick="editorProjectTop()
           "><?php the_title('ↄ '); ?></summary>
                  <div class="entry-content">
               <?php the_content(); ?>
                  </div>
                  <div class="mode-buttons">
               <button class="feel-btn" onclic
               k="editorProjectTop()">see, hear,
               sense</button>
               <button class="read-btn" onclick="
               editorProjectTop()">read</button>
                  </div>
            </details>
```

```
      <?php endwhile;

      else: ?>

            <p><?php echo '<div class="void"></
            div>';?></p>
      <?php endif;

      wp_reset_query();

      ?>

</details>
```

The code indicates that there is a button called 'editor' on the website, which is one of the professional roles that define the client's professional history. Clicking on this button opens a list of project titles ('secondary-details') in which this role is played out. Each project ('editor-project') is also clickable and when clicked opens the content ('entry-content') the client added to it in the admin panel of the website.

Content is divided between two reading modes: visual and textual. By default, textual content opens on the first click. Once you reach the end of the content, there is a button that switches between two reading modes ('feel-btn' and 'read-btn', *btn* being button). If there are no projects assigned to this role, one can choose to show void ('void'). A void, in my collaboration with this respective client, was represented by a dark blue circle piercing the soft aesthetics of the website. In a nutshell: handmade coding of a fairly light website.

The code explained here is my favorite code written thus far. It speculates on the client's future, doesn't limit it to one professional role, nor demands the existing roles to be filled with more professional

activities. It was written with the purpose to emphasise the uncertainty any professional role might have. These professional roles could fade out and turn into the designed void (dark blue circle), or they could be emphasized more and be accompanied by other projects in the future.

After passing two interviews for the 'creative developer' role at WeTransfer in 2021, I was invited to work on a technical assignment. The challenge was to code a 'whack-a-mole' game through which the company could get a sense of my creativity level. There was no framework, language, tool or imagination requirements, but there was one big ask: not to spend too much time on the task. In the span of the single week I was given to develop it, I spent a total of nine hours on it.

Adding to this, I must say my creativity depends on other people. I found and copy-pasted the simplest HTML/CSS/Javascript code of a whack-a-mole (by Ania Kubów) and decided to change the code according to my friends' answers to the question: *If you could whack one thing for the rest of your life, what would it be?*

One of the answers was Facebook:

```
<body>

      <header>

            <div class="display-flex">
                  <p>Facebook killed:</p>
                  <p id="score">0</p><p> times.</p>
            </div>
            <div class="display-flex">
                  <p>Time Left:</p>
                  <p id="time-left">666</p>
```

```
            </div>
        </header>

        <div class="grid">

            <div id="1" class="square"></div>
            <div id="2" class="square"></div>
            <div id="3" class="square
            facebook"></div>
            <div id="4" class="square"></div>
            <div id="5" class="square"></div>
            <div id="6" class="square"></div>
            <div id="7" class="square"></div>
            <div id="8" class="square"></div>
            <div id="9" class="square"></div>
            <div id="10" class="square"></div>
            <div id="11" class="square"></div>
            <div id="12" class="square"></div>
            <div id="13" class="square"></div>
            <div id="14" class="square"></div>
            <div id="15" class="square"></div>
        </div>

        <footer>

            <p>
                [person] undoubtedly wants to
                whack Facebook all the time,
                and for a long time. <br>
                Is your name not [person] <a
                href="/index.html">My name is
                not [person].</a></p>
        </footer>

</body>
```

This is the body of a web page that was built to show the score (how many times Facebook was whacked); above it one can see the countdown time. Underneath this, a grid of fifteen squares floats in which the logo of Facebook appears randomly, and which the user is expected to click or shoot. The footer of the page asks if the user's name is [person]'s name, a.k.a. if they wish to be someone else while whacking this evil, Zuckerberg-spawn thing.

The code was adjusted to the [person]'s specific desires in whacking Facebook, which was done in the following ways: even though the timer's countdown started counting down from 666 seconds, there was, if desired, additional time given for completion of the mission (1,000 seconds). Next to this, the Facebook logo would hang around in one of the grid's squares, long enough to shoot it. It was rather impossible to miss. Unfortunately, the message at the end of the countdown reminds the user that, despite their astonishing scores, Facebook still exists.

```
function countDown() {

      currentTime--

      timeLeft.textContent = currentTime

      if (currentTime == 0) {

            clearInterval(countDownTimerId)
            clearInterval(timerId)
            alert('There is no time left to
            whack Facebook. Despite your score
            of: ' + result + ' points, Facebook
            still exists.')
      }

}
```

```
let countDownTimerId = setInterval(countDown,
1,000)
```

I furthermore collected eight answers from my peers to the question *What would you whack?* The results were: a lack of confidence, eczema, architecture (i.e. one school building per day), sea creatures, Gerald Darmanin, applications and/or funding opportunities, and 'I would whack people who think that rationality is better than emotions and intuitions; I would whack everyone who expects and forces me to be "clear" according to *their* understanding of clarity; I would whack the idea that if you make work you have to be able to write about it; I would whack people who tell me what to do and how to do things without me asking for it; I would whack people who say that they would never drink their own pee if they were in the desert almost dying. *Liars.*'

I loved turning the project into research about what my peers would whack and discovering if my coded reflections on their wishes could get me hired for my creativity level. During the interview with a front-end web developer at WeTransfer, I was told that I was *sooo very* creative.

You can find out the answer to the question whether I got the job or not by reading the following 'list' of websites I dream of building, yet never have the time to:

```
<a href="https://whack-what.vercel.
app/">This was a technical assignment for
WeTransfer, but I didn't get the job.</
a><br>

<a href="/instagram-poetry/ig_bore.
html">IG bore</a><br><br>
```

```
<a href="#">Click here to stop scrolling</
a><br>

<a href="#">Blue Heart Agency</a><br>

<a href="#">I Just Wanna</a><br>

<a href="#">Button Poetry</a><br>

<a href="#">The Writer is Present</a><br>

<a href="#">We only got 10 mins to save
the content</a><br>

<a href="#">Mute everything</a><br>

<a href="#">Forever load, but not like
that</a><br>

<a href="#">Rotating structures, floating
points, and similar</a><br>

<a href="#">All known blanks</a><br>

<a href="#">Working station of profile
production</a><br>
```

It's not a list in code. Were I to write semantic code, it'd be wrapped in a tag, followed by a <li> tag around each <a> tag. Then I'd get rid of the line break tags (
). The way this list looks like right now doesn't bother me in the slightest. Any web developer inspecting it might find these codes very simple, because they *are*. And they bring me a lot of joy.

Female Freelancer

(falling action) Don't doubt the subject's feminism

If I do not become a corporation I will never beat the assholes to the moon

If someone is to go to the moon let me sing you a love song

Let it be me

— Ariana Reines, *A Sand Book*, 'The Long Love that in my Thought Doth Harbor'

In 2021 I was very much flirting with the idea of building my own software. When I shared my goal with Scott, he told me *Good luck*, then eventually asked: If writing semantic code is not your cup of tea, then why don't you just stick to your writing practice instead?

My first answer to that question would be to point out the irony behind it. After my fadeout from his studio, I seriously started answering his question with an occasional pat on the shoulder whenever I reminded myself of my software plan.

The only way out is through, I thought.[1] Compared to the lack of choices with BlockSmith software, my own software would allow more choice to the user—for that's

1 'When an uncontainable artist's influence won't go away, art history compromises by constructing hagiographies. At least that way the vision is contained. But you have to keep reminding yourself of the great dead artist's situation. That he also had contemporaries. That thoughts are never thought alone.' —Chris Kraus, *Aliens and Anorexia*

always the developer's goal. Maybe in collaboration with someone else's brain, I'd figure out what kind of software that would be.

Out of sheer excitement, I called Clara to see if she'd be interested in sharing this dream and becoming a powerful female duo. Slow, poetic, more creative, and, most of all: very capable.

This power woman image led me to believe that we could change the classic reasons as to why everybody wants to hire a female web developer right now, in tech corporations, companies, start-ups, educational institutions, etc. (Slow, poetic, more creative, capable, and, most of all: cheaper.)[2]

With a software by our side, Clara and I would improve the image of female employability, expanding it into safer spaces where the female developer can be her true self.

For example: art academies. Due to the well-known ego politics and patriarchal power dynamics that must be urgently dealt with when encountered, most academies don't have time to find out what exactly they want from the coding tutor they're looking for. They just know that, if they still want to be relevant and contemporary, they urgently need one. Because of a general lack of discourse regarding web development, it's hard for these academies to recognize various educational needs of the (art) practice. Their uncertainty puts them in a position of not being able to take any risks, and often leads them to play it safe, hiring the technically most skilled candidate who can fill a gap, solve a problem, quick, fail-proof.

2 Hiring a female web developer gives a good image/face to the ecosystem where she's been given a chance. She's made to believe that she is the architect of her temple (exquisite), capable of being in control over her money, time, decisions, selfhood, and soul. And all of that shall tame her enough to not resist the aesthetic. But when she does, the aesthetic will either kick her out of its mechanism, or it will be adjusted by professionals to suit her.

For example: in tech corporations and start-ups, where the teams consist of mostly men, a female web developer enters the team as an added 'light energy'. Where the team's goal is focused on the technical progress, a female web developer injects the milestones with her decorative, design touch. Most of the time, she does so in the tasks related to the project's documentation, presentation, and revision—the process, not the finalization. This is the world we live in.

But, with software by our side, we (Clara and I) could put an end to this kind of thinking. This, however, sounded too dreamy.

This power woman image threw me into the depths of identity politics more than the basics of software development—which, truth be told, is what got me excited about the plan in the first place. The burden of our software, no matter how technologically crafted, would be its representation, cornering us in the thread that continues around the equal positioning of women in the history of technology. It would also corner us into celebrating the fact that women *invented* the classification system; that men got recognition for it *before* women started doubting whether it's the best system; that even though women were the ones entirely writing the programs that enabled machines to work, they were kept on the sidelines (Do you remember the term 'kilo-girl'—1,000 hours of computing labor—in the 1940s? No? Look it up.)

I don't want my main role in web development to be the constant retelling of the history of women's (dis) placement within technology, as this is not *the only* way to build history. Yet for the sake of commemorating those women who contributed to computing throughout the history (1700s, a gap, then 1900s till present day), here

is a full list of these women worth remembering from a publication called *Computers at Work*: Nicole Reine Lepaure (built a clock for astronomical observations); Ada Lovelace (first computer programmer); Anna Winlock (mapped the universe for 2 cents/hour, which was half a man's pay back then); Antonia Maury (improved the system of classification by redesigning it); Annie Jump Cannon (developed a stellar classification system, namely the Harvard Classification Scheme);[3] Henrietta Swan Leavitt (discovered galaxies beyond the Milky Way); Dorothy Vaughan (NACA's first black manager, self-taught in FORTRAN, prepared her staff for the transition from human to machine computers); Mary Jackson (NASA's first black female engineer); Katherine Johnson; ENIAC (Kathleen McNulty Antonelli, Jean Jennings Bartik, Betty Snyder Holberton, Marlyn Wescoff Meltzer, Ruth Lichterman Teitelbaum, and Frances Bilas Spence); Kathleen Booth; Grace Murray Hopper; Margaret Hamilton; Mary Kenneth Keller; Katie Bouman, and Donna Strickland.

Extending this list with Clara would end up proving that we're competent enough for the industry. Doing 'average' development would disqualify us as developers; what's more, we would have to be excellent just to get recognized. We would have to become the stoic men.

The hardest thing about building software is deciding *what* to say, not the act of saying it. More choice is never the answer to the question of online freedom. As Douglas Rushkoff said: 'The more choices we make (or are forced to make) the more we believe our expectations will be met. But in actual experience, our pursuit of choice has the effect of making us less engaged, more obsessive, less

3 Stars were organized into classes using seven letters: O, B, A, F, G, K, and M (abbreviation of Oh! Be a Fine Girl—Kiss Me!).

free, and more controlled.' More beta-versions of choices lead to final choices offered by a new software. We're never *not* perceived as full of attention.[4] In the potential failure of my software, new systems would emerge in its wake.

In all this confusion, what seemed to be an immense desire to build a software became an immense desire to just be *materialised thought*. Can I be Sadie Plant's brain, or that of Guangyi Li, Mindy Seu, Nancy Wu? Can I be this website,[5] Carlo Rovelli's whispers[6] in *Real Review#11*, Ben Grosser's Minus, the digital minister Audrey Tang, or Maggie Nelson's latest book, *On Freedom*? But then I also know: I still want to continue writing code for others.

The web development's history, full of evidence on top of evidence, of rights and wrongs about the way one can and should be allowed to code, makes it hard to see coding as a craft for establishing a ground where realistic ambitions between developers and users can be met. Only when a developer is able to explain to their clients the technologies used, the reasons behind using them, their effect, and how to engage with them, can be

4 In *Bitch*, Elizabeth Wurtzel describes the men Amy Fisher was surrounded by: 'It seems that whenever people deal with the world in binary oppositions, choosing one thing only because it negates another, they are inevitably startled by the discovery that the quest for something completely different has only given them more of the same.'

5 *https://www.queeringthemap.com/*

6 In the text, Rovelli structures the consistency of modern reality from a technical point of view, but also from a personal, poetic interiority. Through his model of reality, he claims that objects are temporary visual projections in the current construct of time. He distinguishes between, what I understand, a chunk of concrete information to nod to, and agreement through feeling. Rovelli states that modernity is long gone, exactly because of this complexity of understanding basic notions within the time that has become rich with content. There are multiple times: of the physicist, of a lover waiting for his love to arrive, of a young man dreaming about his future, of an industrialist planning economical strategies, and so on. These examples, in particular, touched me and made me think of attunement as an everyday exercise.

seen to what extent realistic ambitions could be met. The developer, then, is building in collaboration; not just with the aim to solve a problem. Not all collaborations are problematic: this is what Silvio Lorusso encourages us to do in his article 'Code to Learn'.[7] New bootcamp schools are popping up all over the place, new ones arriving as soon as the old ones go bankrupt, while start-up companies and corporations are increasing the demand for more developers. Yet no matter how highly paid or desired these developers are made to feel, there will always be a satisfying amount of workers or developers *hating* their jobs, wishing for a better standard.

> **(the point) Is the point of the subject to empower users to also take control of the tools they're using?**

Were I to wrap up this book with a list of issues at stake, and another one listing the coping strategies for said issues, I'd definitely start from the main issue being the consequence of the tech industry's promises of global governance: user's inability to not participate in online activities from which data is extracted and accumulated. Imagine how a user's conscious choice to move through the online world, with limited choice-making via the clicking of buttons, would reflect on their online existence? Furthermore, on their online history of the self? I do not know, but I'd like to start imagining. Plant knows what I mean when she writes how 'zero was always something very different from the sign which has emerged from the West's inability to deal with anything which, like zero, is neither something in particular nor nothing at all.' — Sadie Plant, *Zeros and Ones*.

Reflecting on the current state of the internet, the production of digital histories, the hands behind it that

7 *https://silviolorusso.com/publication/learn-to-code-vs-code-to-learn/*

enact its unseen labor, as well as the enabling of the users' 'superpowers' while they're on a hunt for that cultural spark to connect with, has led me to realize that the internet could be a space for feeling *offline*. Maggie Nelson, in her book *The Art of Cruelty*, asked an all-round relevant question: 'Perhaps more controversially still, given our inarguable complicity in all kinds of systemic forms of global injustice: is there any space left for not watching, not focusing, not keeping abreast of all the events and atrocities unfolding in the world, as an ethically viable option?'

> **(rising action again) Hmm, but what does the subject want????**

I want to be free (from feedback loops), and I want others to be free, too.[8] Demanding freedom4all seems too naive here. The constraints from which any urgency is born requires addressing its demands through its relations to the neighboring factors (such as alienation, agency, self-governance, survival, consumption, power, class) and which, I believe, are only to be addressed by the body that feels trapped in its own constraints. The disordered. Some grow wings in boxes; they adapt.

I can't stop referring to Maggie Nelson. In *On Freedom*, she asks: 'What if we don't presume, however, that there is any bottom to our desire, that it doesn't lie in a black box at the bottom of the sea? What if learning to notice our shifting drives, identities, curiosities, disinterests,

8 In the second episode ('The Use and Abuse of Vegetational Concepts') of *All Watched Over by the Machine's of Loving Grace*, Adam Curtis explains cybernetics: 'Cybernetics said that everything, from human brains to cities and even entire societies, could be seen as systems regulated and governed by feedback… it seemed to offer a new insight into how order is maintained in the world… Cybernetics saw human beings not as individuals in charge of their own destiny, but as components in systems… They were just nodes in networks, acting and resting to flows of information.'

or aversions, be it over the course of an encounter or a lifetime, is our truer calling?'

Were I to speak from a defined and more developed role (of a writer, web developer, artist) would I dare to materialize these ambiguities I have found within my own dubious relationships with these practices? In the end, it really doesn't matter if you're a stoic man, a female web developer or if you've developed your own software, what extravagant move you make to inject yourself into tech history, if you're into coding, or part of a no-coding movement. What matters is how one chooses to relate to information. Meaning is not *in* the 'document', but in one's relation *to* it.

In his book *In the Flow*, Boris Groys provokes a linguistic agency for the human or user when he writes that 'today we practice our dialogue with the world primarily via the Internet. If we want to ask questions to the world, we act as Internet users. And if we want to answer the questions that the world asks us, we act as content providers.' Thank you for this, Groys. So, then. Let's *think*.

What's the point of making a website when you can start an Instagram account instead? What's the point of writing this book, in fact? Although revealed late, the starting challenge was to try and answer Scott's question: *If writing semantic code is not your cup of tea, then why don't you just stick to your writing practice?* But to not necessarily answer it. Sorry not sorry, Scott.

If there's one personal condition that this book tries to encover within my own usership, it is that of personal liberation from the circles I found harsh while dwelling on the internet. And that the amount of knowledge I've gained from writing this book summarizes not only the level of my commitment to the subject(s), but also the

context I am writing from: my social status, the gain and/or lack of my freelance gigs, my female ambiguity, my questionable convincing skills, plus the fact that sometimes, when the world around me gets too loud, starving is still my default defense mechanism.

fadeout

Acknowledgements

I am so happy to have reached this page where I can express how my contemplations were never done alone.

Writing 'Club-Wise: A Theory of Our Time' helped me to stay awake and less drunk throughout the sleepless nights of my life as a bartender, whereas writing 'Diary of a Stylist' made me feel human in a world where everything seemed to act just like an image. This publication, like most of my previous ones, started from an urge to break free from the structures I considered restraining to my nourishment of feeling free. And, just like the previous ones, it ends with the realization that everything is fine, no matter how many times Fuck it seemed like the only way out.

The peers mentioned below are the peers who made my internal inspections feel comfortable; who encouraged me to always, albeit critically, embrace boredom in life, a.k.a. to flee from the structures that bore me once I've fully stripped them of their essence.

Thank you, Clara Pasteau, for opening the web-dev scene for me and teaching me how to kiss my shoulder through the hardships. I still do it. This freelance path was not completely lone-wolf because of you. AND: I can't wait to start a company with you. Alina Lupu and Marlies van Hak: without your invitation to reflect on the precarious artistic practices throughout the pandemic, this subject wouldn't have felt like the gut feeling it did now, one I decided to follow in order to further reflect on my coding endeavors. Thank you for tickling the juiciest of my scars and passions. Talking about those who never cease to nourish me: this brings

me to Geert Lovink, forever. It's funny because it's so obvious; I smile because I don't know what to thank you for anymore. Thanks for mentally slouching with me. Oh and, stoic man, thank you for a short collaboration. It obviously inspired a big chunk of this book.

And then there is the production layer of gratitude: the INC team!!! Thank you for being so OK with me squatting the office, but also for the supportive fun times we have <3. It always feels like we're breaking the internet when we're together. Especially thanks to Laurence Scherz, Sepp Eckenhaussen, and Maria van der Togt for helping with the final editing and designing of the book. As for my feedback givers, let me drop some names: Alina Lupu, Françoué Giraremeunier, Aurélien Lepetit, Anesa Imamović—I absolutely loved every bittersweet word you threw at me after reading version 1.0. You made me feel proud of my public shyness. James P.A. Crossley! Thank you for the cover design and pleasant presence during the process of writing.

And then there is the final layer of gratitude. My dear Imamovićs (Mama and Tata), thank you for not understanding my theories at all. Eszter Kiss, for being the apple of my eye. Lacey Verhalen, for cheerleading (How is your book going?) while writing your thesis. Sophie Cloes, for adopting me in Brussels when and where I started writing the first few chapters of this book. Ksenia Perek, for convincing me to put my face all over the cover, then finally suggesting not to do it. Donauweg 8, for being the sexiest studio space in the middle of nowhere. Donauweg 8, for giving me the best studio mates (Roman Tkachenko, Lena Karson, Natalia Blahova). Stefan Pavlović ;) Thank you NXS World, for commissioning and hosting a part of the text (about the

young web developer). And of course, thanks to all my clients and friends for hiring me sufficiently so that I could write this book for free.

You make me all heart and no play.

References

BOOKS

Boris Groys, *In the Flow*, London: Verso, 2016.

bell hooks, *All About Love*, New York: HarperCollins, 2018.

Christopher M. Kelty, *Two Bits: The Cultural Significance of Free Software*, Durham: Duke University Press, 2008.

Chris Kraus, *Aliens and Anorexia*, Los Angeles: Semiotext(e), 2013.

Geert Lovink, *Stuck on the Platform*, Amsterdam: Valiz, 2022.

Maggie Nelson, *On Freedom*, Minneapolis: Graywolf Press, 2021.

Maggie Nelson, *The Art of Cruelty*, New York: W. W. Norton Company, 2011.

Kelsey Osgood, *How to Disappear Completely*, London: Gerald Duckworth & Co Ltd, 2014.

Sadie Plant, *Zeros and Ones*, London: Fourth Estate, 1998.

Ariana Reines, *A Sand Book*, United States: Tin House, 2019.

Douglas Rushkoff, *Present Shock*, London: Current, 2013.

Douglas Rushkoff, *Program or be Programmed: Ten Commands for a Digital Age*, New York: OR Books, 2010.

Legacy Russell, *Glitch Feminism*, New York: Verso, 2020.

Natasha Stagg, *Sleeveless*, Los Angeles: Semiotext(e), 2019.

Ocean Vuong, *On Earth We're Briefly Gorgeous*, New York: Penguin Press, 2019.

McKenzie Wark, *A Hacker's Manifesto*, The Anarchist Library, 2013.

Gerald Weinberg, *The Psychology of Computer Programming,* New York: Van Nostrand Reinhold Company, 1971.

Elizabeth Wurtzel, *Bitch*, London: Quartet Books Limited, 1999.

SONGS
Donna Ares, 'Prokleta je Amerika', 2012.
Lisa Loeb, 'She's Falling Apart', 2002.

ARTICLES
Silvio Lorusso, 'Learn to Code vs. Code to Learn: Creative Coding Beyond the Economic Imperative', *Graphic Design in the Post-Digital Age*, 2022, *https://silviolorusso.com/publication/learn-to-code-vs-code-to-learn/*.
Venkatesh Rao, 'Why We Slouch', *Ribbonfarm: Constructions in Magical Thinking,* December 20, 2018: *https://www.ribbonfarm.com/2018/12/20/why-we-slouch/*.

MOVING IMAGE
Adam Curtis, 'All Watched Over by the Machine', Episode #2: 'The Use and Abuse of Vegetational Concepts', BBC Productions, 2010.

WEBS
Audrey Tang, *https://audreyt.org/*.
A Website Is A Room, *https://a-website-is-a-room.net/*.
Cyberfeminism Index, *https://cyberfeminismindex.com/*.
How I Experience Web Today, *https://how-i-experience-web-today.com/*.
Minus, *https://bengrosser.com/projects/minus/*.
Motherfucking Website, *https://motherfuckingwebsite.com/*.
QUEERING THE MAP, *https://www.queeringthemap.com/*.

MAGAZINES
Real Review#11, 'What it Means to Live Today', London: REAL Foundation, Spring 2021.