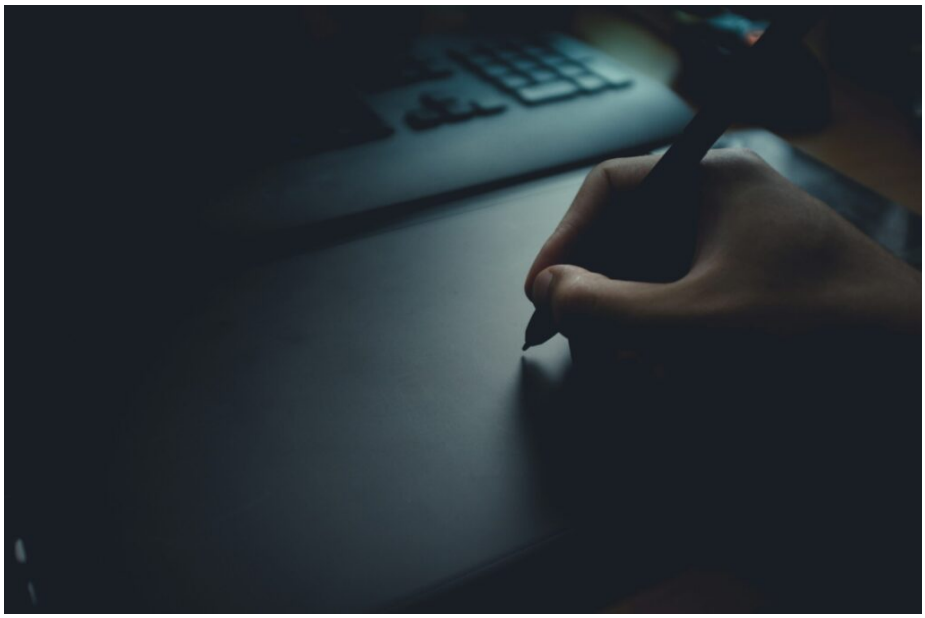
Issue 068: Design

Adrian Kosmaczewski

May 6th, 2024



Welcome to the sixty-eighth issue of *De Programmatica Ipsum*, about *Design.*

In this edition:

- We analyze the impossible dialogue[1] between graphic designers and developers.
- In the Library section[2], we review "Design for Hackers" by David Kadavy[3].
- In our Vidéothèque section[4], we watch "Helvetica" by Gary Hustwit[5].

We would like to thank our patrons who generously contribute every month (or have contributed in the past) to our work and help us run this magazine. Thank you so much! In alphabetical order: Adam Guest, Adrian Tineo Cabello, Benjamin Sheldon, Christopher Nascone, Colin Powell, Franz Lucien Moersdorf, Guillermo Ramos Álvarez, Jean-Paul de Vooght, Dr. Juande Santander-Vela, Patryk Matuszewski, Paul Hudson, Quico Moya, Roger Turner, and Szymon Licau.

Enjoy this issue! Please subscribe to our free newsletter[6] to stay updated about new releases, share the articles on social media, or contribute[7] if you would like to support our work with a donation via Liberapay[8].

[1] https://deprogrammaticaipsum.com/the-impossible-dialogue-revisited/
[2] https://deprogrammaticaipsum.com/category/library/
[3] https://deprogrammaticaipsum.com/david-kadavy/
[4] https://deprogrammaticaipsum.com/category/videotheque/
[5] https://deprogrammaticaipsum.com/gary-hustwit/
[6] https://deprogrammaticaipsum.com/newsletter/
[7] https://deprogrammaticaipsum.com/contribute/
[8] https://liberapay.com/

Cover photo by Glenn Carstens-Peters[9] on Unsplash[10].

[9]https://unsplash.com/@glenncarstenspeters?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

[10]https://unsplash.com/photos/person-using-track-pad-P1qyEf1g0HU?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

The Impossible Dialogue, Revisited

Adrian Kosmaczewski

May 6th, 2024

During the preparation of this edition of the magazine, I realized that we have talked so often about design, that we might as well advertise ourselves as a design magazine. The reason is that design is a broad concept, encompassing various meanings, definitions, roles, and ideas. Or maybe that is because, well, software developers *are* designers.

In the pages of previous editions we have directly covered subjects such as web design[1], desktop app design[2], computer game design[3], design by contract[4], algorithm design[5], design patterns[6], test-driven design[7], object-oriented design[8], programming language design[9], platform design[10], organizational design[11], skeuomorphism[12], hardware design[13], cross-platform app design[14], design processes[15], and even fashion design[16]. So why another

[1] https://deprogrammaticaipsum.com/andy-clarke/
[2] https://deprogrammaticaipsum.com/jenifer-tidwell/
[3] https://deprogrammaticaipsum.com/books-about-game-design-and-development/
[4] https://deprogrammaticaipsum.com/issue-38-design-by-contract/
[5] https://deprogrammaticaipsum.com/john-maccormick/
[6] https://deprogrammaticaipsum.com/the-gang-of-four/
[7] https://deprogrammaticaipsum.com/kent-beck/
[8] https://deprogrammaticaipsum.com/james-coplien/
[9] https://deprogrammaticaipsum.com/alan-perlis-and-the-evolution-of-programming-languages/
[10] https://deprogrammaticaipsum.com/geoffrey-g-parker-marshall-w-van-alstyne-sangeet-paul-choudary/
[11] https://deprogrammaticaipsum.com/divide-et-impera/
[12] https://deprogrammaticaipsum.com/issue-40-skeuomorphism/
[13] https://deprogrammaticaipsum.com/breaking-the-3-ghz-barrier/
[14] https://deprogrammaticaipsum.com/dr-dobbs-and-the-deathly-cross-platform-app/
[15] https://deprogrammaticaipsum.com/on-the-aversion-to-writing/
[16] https://deprogrammaticaipsum.com/tenue-correcte-exigee/

issue about "design" itself, and even more importantly, what do we mean with the word "design", anyway?

In a now classic interview[17] published in Wired Magazine in February 1996, Steve Jobs[18], still CEO of an ailing computer company called NeXT (just months away from a historic return to Apple) said a now famous phrase:

> Design is a funny word. Some people think design means how it looks. But of course, if you dig deeper, it's really how it works. The design of the Mac wasn't what it looked like, although that was part of it. Primarily, it was how it worked. To design something really well, you have to get it. You have to really grok what it's all about. It takes a passionate commitment to really thoroughly understand something, chew it up, not just quickly swallow it. Most people don't take the time to do that.

This quote, put in context by the events that would follow his return to Apple merely 10 months later, have been repeated once and again by a whole generation of graphic and industrial designers. Mostly because before the 2000s and the rise to prominence of Jony Ive[19], the role of designers in the world of software was, at best, derided. The iMac G3[20] released in August 1998 is today considered a major historic milestone[21] in the story of personal computing; but at the time, it was widely ridiculed, starting with its "bondi blue" color[22].

Even today, among software developers, the best we can say is that designers are tolerated, but not much more.

It is difficult to know how many software developers are working today on user-facing software, involving usability and visual constraints. Many of them are working "behind the scenes", providing APIs and databases and other "low-level constructs", not meant to be seen directly by the eyes of the profane. Those developers are those more prone to dismiss visual design considerations as frivolous or useless.

The truth is, those same software developers have a different meaning of the word "design", akin to that found in expressions such as "Test-Driven Design" or "Object-Oriented Design". This kind of design is close in spirit to that explained in "The US Army / Marine Corps Counterinsurgency Field Manual"[23] by U.S. Army General David Petraeus.[24] (I bet you never thought we would mention this person or this book in this magazine, now, did you.)

In section 4-2 of this otherwise surprisingly interesting book, "design" is defined in contraposition to "planning"; while these two words are closely related, they are very different upon closer inspection:

> It is important to understand the distinction between design and planning. (...) While both activities seek to formulate ways to bring about preferable futures, they are cognitively different. Planning applies established procedures to solve a largely understood problem within an accepted framework. Design inquires into the nature of a problem to conceive a framework for solving that problem.

[17] https://www.wired.com/1996/02/jobs-2/
[18] https://deprogrammaticaipsum.com/steve-jobs/
[19] https://en.wikipedia.org/wiki/Jony_Ive
[20] https://en.wikipedia.org/wiki/IMac_G3
[21] https://www.theregister.com/2010/11/29/bondi_blue_imac/
[22] https://encycolorpedia.com/0095b6
[23] https://www.myselfdefensetraining.com/wp-content/uploads/2013/08/US-Army-Marine-Counter-Insurgency-Manual-Authored-by-Lt-General-David-H.-Petraeus-US-Army-Lt-James-Amos-US-Marines.pdf
[24] https://en.wikipedia.org/wiki/David_Petraeus

> In general, planning is problem solving, while design is problem setting. Where planning focuses on generating a plan—a series of executable actions—design focuses on learning about the nature of an unfamiliar problem.

In short, while planning deals with the known, design deals with the unknown. Design is the process by which the unknown becomes known.

And guess what: visual designers and software developers (both at the backend and frontend) are, essentially, designers. The design of algorithms and the design of user interfaces are not much different from one another, a fact that might enrage hordes of snob software developers, swimming into the waters of their own hubris and scoffing at such claims. So be it.

Different software developers have to deal with different constraints. For example, DevOps engineers busy stuffing code inside a Docker container might not realize the amount of design they have to deal with at any step of the way: should they base their work on `alpine:latest` or `scratch`? Can they? Should they include all libraries, or can they strip a few to slash some megabytes? What programming languages would be better suited to make this container run faster and smaller?

Mobile app developers might have to deal not only with similar considerations of code size and efficiency (particularly when those apps are downloaded through questionably slow mobile networks), but also with the whims of Cassowary linear constraints[25] that define how their user interfaces spread over the screens of devices with various sizes.

Web app developers (or, as they are known nowadays, "full-stack developers") would in turn tweak their CSS stylesheet endlessly, until their responsive designs adapt properly to most mobile and desktop platforms alike.

Machine-learning developers, arguably one of the most sought-after professions at the time of this writing, will try to build their large language models in such a way that they can make sense of the highest possible amounts of data minimizing GPU time (and as a consequence, dollar spending in cloud resources).

There is, however, another common issue between software developers (particularly those involved in the creation of user interfaces, such as mobile, desktop, or web apps) and graphic / industrial designers. This problem is not dissimilar to that existing between software developers and their managers, one that we described in detail in our June 2021 article titled "The Impossible Dialogue"[26].

In this particular case, it is not uncommon for software developers to not only ignore, but even dismiss the required understanding of design language and primitives. At the same time, designers often lack the knowledge required to understand the various constraints imposed by software and hardware platforms.

Hence, the impossible dialogue appears once again on our radar. Just like between software developers and their managers. (The fact that so many impossible dialogues surround software developers should raise some concerns among my readers, if you ask me.)

I have witnessed (and documented[27]) many such clashes in the past, where UI and UX designers sometimes miss the mark in their visual creations, either forcing software developers

[25] https://en.wikipedia.org/wiki/Cassowary_(software)
[26] https://deprogrammaticaipsum.com/the-impossible-dialogue/
[27] https://akos.ma/blog/10-things-every-iphone-app-designer-should-know/

to create inefficient software solutions, lengthening the development process, and enforcing a misguided perception of the craft of visual design down the road.

Such conflicts usually end up resolved through the mediation of some higher placed member of management: either by privileging one or the other camps, but seldom by bringing them together in a healthy interaction that benefits everyone as a team, and also the end product (and the riches that might come with it).

The rise of the iMac, the iPod, the iPhone, the iPad, and the iWhatever, raised the perception and status of visual designers in the software industry to a status never before seen. In less than 15 years, we went from a world of 16-color Windows 3.1 apps to Web 2.0 shiny buttons. "Design" became a force to reckon with. Unfortunately, the rise in importance was also followed by a rise of hubris, and that clearly does not help anyone.

Thankfully, however, some designers denounce[28] this state of things:

> Design is expected to produce "new" or "beautiful" or "special" things. When we look around with this mentality, things outside of "design" are seen as "normal" or "ugly" in contrast. (...)
>
> Meanwhile design, once an almost unknown profession, has become an important source of pollution. (...)
>
> Design is vulgarly interpreted as temporary entertainment, which however loses all value as soon as the show ends, when the initial emotion fades. And when everyone wants to be unique, everyone ends up being the same.

Mike Monteiro is more worried[29] about the social role of designers, and the need to regulate their profession. We agree, and this magazine has argued[30] for similar regulation[31] in the past.

> This roomful of designers, however, was quite taken aback by the idea that our industry, an industry which now regularly designs devices that go inside human bodies, or control our medication, or is writing logic for putting driverless tractor trailers on the street, should need professional licensing. (...)
>
> There's no longer room in Silicon Valley to ask why. Designers are tasked with moving fast and breaking things. *How* has become more important than why. *How* fast can we make this? *How* can we grab the most market share? *How* can we beat our competitors to market?

We do not need to return to a time of crappy Windows 3.1 software, nor we need the current state of things, with beautiful yet broken software in eternal Beta, busy deconstructing society and democracy left and right. To revert the current state of things means that we must find a healthy equilibrium between the needs and capabilities of both visual designers and developers. But I hate giving good people bad news: this equilibrium requires making this impossible dialogue possible, and it requires a middle point, to which both parties must travel[32], humbly and consistently.

Besides stating the obvious, that is, that beauty sells, many a software company would be wise to integrate the more visual-oriented brains of a graphical or industrial designer in their

[28] https://normadesign.it/en/log/nothing-special/
[29] https://monteiro.medium.com/designs-lost-generation-ac7289549017
[30] https://deprogrammaticaipsum.com/primum-non-nocere/
[31] https://deprogrammaticaipsum.com/on-the-need-of-regulation-in-the-iot-industry/
[32] https://deprogrammaticaipsum.com/david-kadavy/

teams, to compensate for the algorithmic mindset, characteristic of most software developers out there. Even worse, we could fall into the trap of misapplication of the concept of lateralization of brain functions[33], a classic analogy that has been misused by sociology and management; but if such an exercise works to reduce the gap between software engineers and visual designers, it might serve its purpose after all.

Bringing some much-needed neurodiversity to our teams is something that has been proven time and again to be a beneficial and healthy attitude, yielding better software; not only looking better, but also with better-working features.

Cover photo by Med Badr Chemmaoui[34] on Unsplash[35].

[33] https://en.wikipedia.org/wiki/Lateralization_of_brain_function

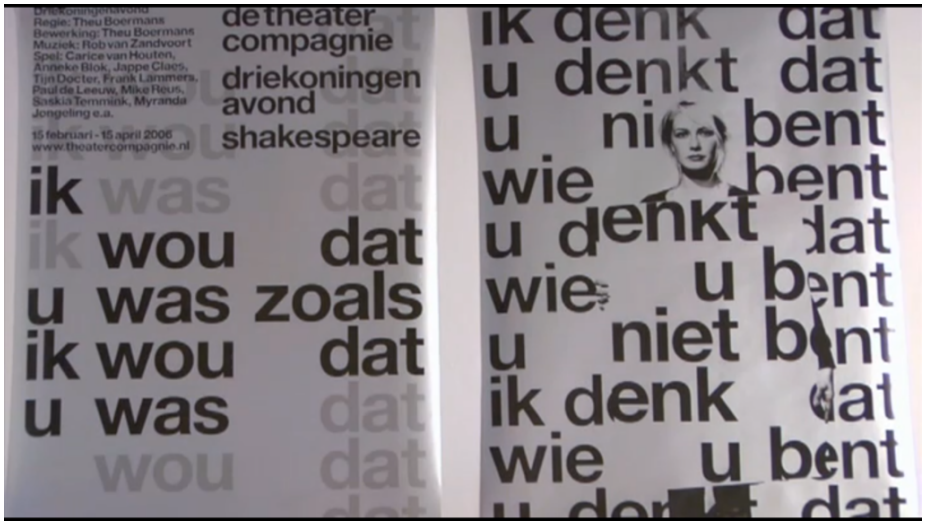
[34] https://unsplash.com/@medbadrc?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

[35] https://unsplash.com/photos/blue-ballpoint-pen-on-white-notebook-ZSPBhokqDMc?utm_content=creditCopyText&utm_medium=referral&utm_source=unsplash

Gary Hustwit

Adrian Kosmaczewski

May 6th, 2024



According to the Oxford Dictionary of English (second edition, revised 2005), the word "Grotesque" comes from the mid 16th century French word "crotesque". This is confirmed by Le Robert Dictionnaire Historique de la Langue Française, by Alain Rey et al. The word derives from the Italian adjective *grottesca*, related to *grotto,* or cave. It was first used to designate a certain type of art found in ancient Roman basements that apparently hurt some sensibilities. The Oxford Dictionary gives the following two meanings to the word: first, "comically or repulsively ugly or distorted". Second, "incongruous or inappropriate to a shocking degree".

Nothing very positive.

Around 1835, an English type founder named William Thorowgood[1] christened "grotesque" the first sans-serif typeface ever, called "7-line Pica Grotesque". For those who might not know what "sans-serif" means, open LibreOffice Write or Microsoft Word and compare Arial and Times New Roman on the same document: the former is sans-serif, while the latter is serif. The word "serif" refers to the little squiggles (yes, that is the scientific word) that appear at the end of the letters, and which are not visible in, well, "sans" serif typefaces ("sans" meaning "without" in French). Ah, well, just compare both and you will see what I mean.

(See how I was able to use your pre-existing knowledge of modern, visual computing environments to my advantage? Explaining such a concept in 1957 would have been considerably harder. We live in admirable times, mostly thanks to the combined efforts of Microsoft and Apple in the early 1990s, who begat TrueType[2], thus making everyone an expert in typography during the past 35 years.)

[1] https://en.wikipedia.org/wiki/William_Thorowgood
[2] https://en.wikipedia.org/wiki/TrueType

Giving the "grotesque" moniker (or, sounding even stronger, the German equivalent "Grotesk") to a whole family of typefaces clearly transpires the sentiment of the era towards these "lesser" forms. At the end of the 19th century, however, they started to become ubiquitous, and the Berthold Type Foundry of Berlin came up with one of the most popular ones of its age: "Akzidenz Grotesk"[3]. (So, not only it was grotesque, it was also accidental. Talk about bad karma.)

In 1957, the Haas Foundry in Münchenstein, Switzerland decided that Akzidenz Grotesk deserved an upgrade, and they decided to name it "Neue Haas Grotesk". But you know, Haas belonged to a German company named Stempel[4], which itself belonged to an American company named Linotype[5]. The marketing team of Linotype was dismayed by the name (I am being polite here), and decided to rename it to, well, "Helvetica", since it was created in Switzerland.

Imagine pulling down the "Font" menu in Microsoft Word and being greeted with "Neue Haas Grotesk". Ugh. I would immediately choose Comic Sans instead.

(At the risk of alienating my Swiss readership, I find that in general the Swiss have many qualities, but naming things is not one of them. Think about what Phil Karlton said[6] about computer science, but applied to pretty much every field of life.)

So all of this diatribe is just to tell you the story of how the typeface "Helvetica" came to be. And the movie of this month's Vidéothèque, "Helvetica" by Gary Hustwit[7], tells precisely this story but in much more glorious detail (and without alienating any part of his viewership in the process).

This 80 minute documentary is surprisingly interesting, given that, well, it is a documentary about a typeface after all. It turns out that Helvetica (the typeface) is everywhere, and what this film does to you is to show it to you, so you can realize the sheer number of times you have looked at it without seeing it. Case in point: do you know what the logos of MetLife, BMW, Greyhound, Sears, Jeep, Toyota, Kawasaki, Target, Tupperware, Nestlé, Verizon, Parmalat, Lufthansa, JCPenney, Staples, AGFA, National, Panasonic, American Apparel, USPS, and the NASA Space Shuttle have in common? Exactly.

I will not spoil any more of the movie for you, but there are three quotes that will definitely appeal to the Swiss readership of this magazine (yes, precisely the one I just lost a few paragraphs ago).

Wim Crouwel[8] at minute 11:33:

> You can't do better design with a computer, but you can speed up your work enormously.

Erik Spiekermann[9], who despises Helvetica ("it hasn't got any rhythm!") but who completely understands the country where it comes from, at minute 37:37:

> The whole Swiss ideology, the guy who designed it, tried to make all the letters look the same. Hello? You know, that's called an army, that's not people.

[3] https://en.wikipedia.org/wiki/Akzidenz-Grotesk
[4] https://en.wikipedia.org/wiki/Stempel_Type_Foundry
[5] https://www.linotype.com/
[6] https://martinfowler.com/bliki/TwoHardThings.html
[7] https://www.hustwit.com/helvetica/
[8] https://en.wikipedia.org/wiki/Wim_Crouwel
[9] https://en.wikipedia.org/wiki/Erik_Spiekermann

(By the way, check out this history of typography[10] by Erik Spiekermann, you will thank me later.)

Lars Müller, author of "Helvetica: Homage to a Typeface"[11], at minute 43:05:

> The image of Helvetica as the corporate typeface made it so-called the typeface of capitalism, which I would actually reject, and say it's the typeface of socialism, because it is available all over and it's inviting dilettantes and amateurs and everybody to do typography, to create their own type design, and I think that's a good thing.

After the release of this documentary in 2007, Gary Hustwit finished his "designer's trilogy" of films with "Objectified"[12] (2009), and "Urbanized"[13] (2011). He later published a few more acclaimed documentaries, including "Rams"[14] in 2018. This is not, I am afraid, a documentary about random access memory, but about Dieter Rams[15], the legendary designer of Braun in the 1960s and 1970s, author of "Ten Principles of Good Design" that, in the opinion of the author of these lines, should be applied to software design as well.

But there is more: another documentary from Gary Hustwit is opening as this article hits your browser window: "Eno"[16], a movie about English musician Brian Eno[17], built with generative AI... and literally changing every time you watch it. (One wonders how much it must cost in GPU dollars to achieve such a feat?) Definitely on my watch list.

Watch this month's Vidéothèque movie, "Helvetica", by Gary Hustwit[18], on Vimeo VOD[19] and other platforms. I promise it is not a grotesque film, at all.

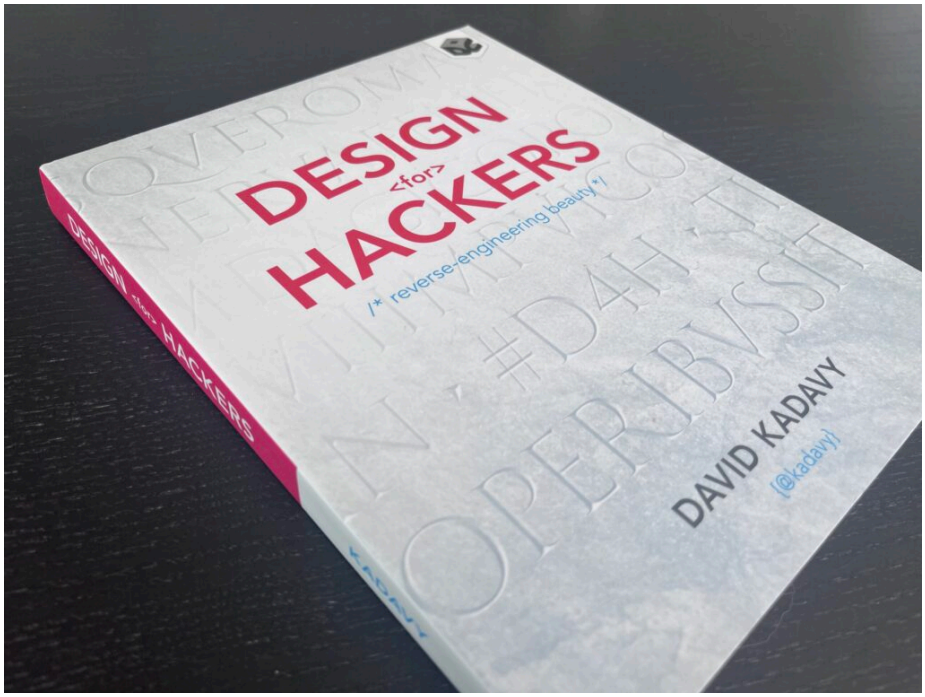
Cover snapshot chosen by the author.

[10] https://www.spiekileaks.com/typography-history-and-technology
[11] https://www.lars-mueller-publishers.com/helvetica
[12] https://www.hustwit.com/objectified
[13] https://www.hustwit.com/urbanized
[14] https://www.hustwit.com/rams
[15] https://en.wikipedia.org/wiki/Dieter_Rams
[16] https://www.hustwit.com/eno
[17] https://en.wikipedia.org/wiki/Brian_Eno
[18] https://www.imdb.com/title/tt0847817/
[19] https://vimeo.com/ondemand/helvetica3

David Kadavy

Adrian Kosmaczewski

May 6th, 2024



Few ecosystems react as viscerally and as brutally to "bad" visual design than whatever Apple has brought into the world. From the first Macintosh to the latest Vision Pro, the whole idea of making apps for Apple platforms involves, hopefully sooner than later, a severe and serious evaluation of style along with functionality.

Let us look at some examples of when said ecosystem reacted viscerally and brutally. On October 3rd, 2009, Brandon Pittman published an article titled "The Worst Twitter Client... Ever" on the Smoking Apples website, nowadays available[1] thanks to the commendable work of the Internet Archive. Here are some quotes from this gem of an article:

> If there was ever a reason Steve Jobs didn't want people making apps for iPhone, ChillTwit was it. (...) I honestly believe the SA editors assigned me this review to see if I'd kill myself. (...)
>
> I can't find one redeeming quality about this app. (...) the fact that he's trying to get $0.99 out of people pisses me to no end. (...)
>
> (...) if you buy this, we're not friends anymore.

You get the idea.

[1] https://web.archive.org/web/20091006005807/http://smokingapples.com/iphone/app-store-iphone/the-worst-twitter-client-ever

In the same vein, one year earlier, John Gruber, author of the influential Daring Fireball blog, published a picture on his Flickr account[2] (we cannot get any more retro than this!). The caption reads:

> A screenshot from Stevens Creek Software's upcoming iPhone app, TripLog/1040. I'm not even sure where to start.

There is a direct connection between Pittman's and Gruber's criticisms (and the discussion[3] it generated in the latter case) with MKBHD's recent review of the Humane AI Pin[4] and the controversy[5] surrounding it. But as usual, I digress. The important point here is that, for some platforms like iOS, looks matter more than software developers would like to admit, and reviews can make or break entire product lines (and even businesses) in no time.

Of course, not all ecosystems display such levels of vitriol against amateurish looks. In no particular order, Windows, Android, and Linux in general, are environments where functionality, price, and stability will always be primed over good looks. This is fine and good, but where it becomes problematic is where users and developers on these platforms deride or make fun of a supposedly "lesser" world such as Apple's.

Again, hubris and disdain does not help anyone. Some users prefer beautiful things, some others prefer working software, and some others prefer both at the same time.

Faced with the need to publish applications that work well and look decently good, the question software developers ask themselves is, then, how can you learn good visual design when all you have is a computer science degree? Turns out that a designer from Silicon Valley not only answered this question, but also created a fundamental piece of work in the process.

Published in 2011, "Design for Hackers" by David Kadavy dives into the common elements of design: typography[6], composition, color, and starts a much-needed discussion about why design is important, and how technology both influences and is influenced by it.

It is profusely illustrated, and the tagline "reverse-engineering beauty" gives a good hint of the contents and the intended audience.

"Design for Hackers" can work as a study and reference material, including some appendixes at the end with useful typography tips and tricks, for example about how to pair fonts, or how to use them properly in various contexts.

This work belongs to my favorite kind of reading: a treatise that bridges the gap between two worlds that display impossible dialogues[7] and have a tendency to clash for the pettiest of reasons. Designers and engineers often coexist in software development teams in a false dichotomy, angry at each other, blaming each other, instead of trying to understand each other-which, spoiler alert, is the most important thing you can do with fellow humans, at all times.

Proportion, whitespace, hierarchy, rhythm, are all words that have different meanings in the minds of software developers, and to be able to have a conversation with the designers in your team, you would be wise to learn them.

---

[2] https://flickr.com/photos/gruber/2635257578
[3] https://daringfireball.net/linked/2008/07/09/triplog-redux
[4] https://www.youtube.com/watch?v=TitZV6k8zfA
[5] https://www.youtube.com/watch?v=QztFpzKsdeA
[6] https://deprogrammaticaipsum.com/gary-hustwit/
[7] https://deprogrammaticaipsum.com/the-impossible-dialogue-revisited/

Inversely, I can only recommend visual designers to learn more about the constraints and limits of software and hardware, and to get acquainted with the design guidelines of your chosen platform. I cannot tell how many times I have had to explain to designers those limits, trying to make them understand the tradeoffs in cost and speed of development their teams could reach by following them. Once again, the biggest issue blocking teams from delivering good software is communication.

For those software developers interested in the art and science of visual design, I could recommend some other important works. First, the quintessential "The Design of Everyday Things" by Don Norman[8]. We have talked in the pages of this magazine about "Designing Interfaces" by Jenifer Tidwell[9] (about desktop software design) and "Transcending CSS" by Andy Clarke[10] (about web app design), and we repeat the recommendation now. Finally, take a look at "Graphic Design: The New Basics" by Ellen Lupton and Jennifer Cole Phillips[11] for a complete overview of basic concepts.

If you look to deepen your knowledge of visual design with additional information, head over to books.design[12], an incredible resource created[13] by Norma, a "non-graphic design studio in Turin". Finally, for a social perspective of the impact of design in our modern world, "Design is a Job"[14] by Mike Monteiro (of who we have already talked about[15] in this magazine) is a mandatory reading. Also noteworthy, the newsletter "Not a Designer"[16].

Making better software does not only mean making applications with good modularity, plenty of unit tests, and many useful features, but also with better looks, decent levels of usability, and with enough attention paid to accessibility. These are indeed useful qualities, and will help you to, at least, avoid scathing reviews of your next app.

Cover photo by the author.

[8] https://mitpress.mit.edu/9780262525671/the-design-of-everyday-things/
[9] https://deprogrammaticaipsum.com/jenifer-tidwell/
[10] https://deprogrammaticaipsum.com/andy-clarke/
[11] https://ellenlupton.com/Graphic-Design-The-New-Basics
[12] https://books.design/
[13] https://normadesign.it/en/projects/books-by-designers/
[14] https://www.designisajob.com/
[15] https://deprogrammaticaipsum.com/mike-monteiro/
[16] https://notadesigner.io/