

Adebiaye, Franck.  
Velvetyne, 2010,  
URL : <https://velvetyne.fr>, page consultée le 19 septembre 2021.



## Une fonderie atypique

Velvetyne (VTF) est une fonderie numérique libre créée en 2010 par Frank Adebiaye, ancien comptable passionné par l'informatique depuis toujours. Il s'est initié lui-même aux arts graphiques et à la typographie, puis a commencé à mêler ces différentes pratiques à son activité dans la comptabilité. Il a créé un blog sur la typographie, dont VTF est la suite logique. Même s'il a commencé seul, il a rapidement invité Jérémy Landes-Nones et Sébastien Hayez à se joindre à lui dans ce projet qui aujourd'hui, propose également des workshops. Ainsi, le collectif Velvetyne peut être séparé en 2 groupes complémentaires ; d'une part, l'équipe dirigeante, qui discute de la direction dans laquelle se dirige la fonderie et des prochains workshops. D'autre part, on retrouve les concepteurs publiés dans le catalogue du site, mais qui ne prennent pas part aux décisions de la fonderie.

Velvetyne propose une grande diversité de polices de caractères, qui partagent toutes un esprit punk, parfois expérimental, où chaque nouvelle typographie diffère de ce qui est déjà proposé au sein du catalogue. Ce choix est une vraie volonté de Velvetyne d'offrir un espace aux typographies atypiques, peu lisibles, étranges, souvent inclassables dans les systèmes habituels tels que Thibaudeau ou Vox Atypi. On y retrouve également la volonté d'accroître la diversité typographique, qui peut être proposée par des typographes autant que par des graphistes, artistes, ou autres.

En plus de l'originalité de la police, le concept ou l'histoire sous-jacente à sa fabrication sont une partie importante de la sélection des typographies. On retrouve particulièrement cette notion au sein de la Cantique, une typographie de Sébastien Hayez, professeur et graphiste qui raconte au travers de plusieurs articles les différents moments de sa vie qui ont marqué l'évolution de la typographie, initiée dans un projet de glyphs modulaires.



Une autre particularité de cette fonderie est qu'elle propose uniquement des typographies sous licence libre SIL (OFL). Cela vient de leur pensée que les polices propriétaires ne sont pas les meilleurs outils pour certaines pratiques créatives, originales, mais qui méritent néanmoins des polices de caractères de haute qualité.

Cette qualité vient par ailleurs du fait que chaque typographie proposée est souvent retravaillée en collaboration avec des membres de la fonderie, ce qui mène à des sets de glyphs qui peuvent devenir très complets.

# Liens au libre

La Mourier en est un bon exemple, c'est une typographie géométrique basée sur un carré de  $7 \times 7$  unités, avec des lignes ne devant pas être clôturées. Conçue par Eric Mourier en 1971, elle a été numérisée par Sébastien Hayez en 2002, puis a été ajouté en 2020 l'alphabet Cyrillic par Alex Ash, suivi peu après de l'ajout des bas de casse et divers autres symboles par Ariel Martín Perez.

De fait, les polices Velvetyne peuvent être utilisées sur n'importe quel support imaginable, mais nécessitent néanmoins de créditer le nom du créateur et de la fonderie à chaque utilisation.



VTF met en avant l'éthique libre, et notamment le système de production du logiciel libre, avec pour idée qu'une pratique de la typographie plus ouverte et transparente, augmenterait le dynamisme de la création contemporaine, à la manière du développement des logiciels libres. Dans cette même approche, les utilisateurs peuvent continuer à développer et adapter les typographies en fonction de leur propre point de vue et de leurs besoins particuliers.

Par ailleurs, les choix de publications se font par l'équipe dirigeante, qui prend des décisions collégiales sur les prochaines polices à travers un processus de vote anonyme. Le système de fonctionnement se fait alors horizontalement, contrairement à une hiérarchie verticale telle que l'on peut généralement la retrouver ailleurs.

En plus de leur activité principale de fonderie, l'équipe VTF propose des workshops et des interventions. Cette pratique s'inscrit pleinement dans l'esprit de partage des connaissances et des processus de création intrinsèques au logiciel libre.

De fait, la licence SIL (OFL) est une base nécessaire, comme le souligne l'équipe VTF, pour enseigner la création et la distribution des typographies, mais aussi pour la populariser, la rendre plus accessible au grand public, afin que tout le monde puisse comprendre son fonctionnement.

Par ailleurs, le collectif Velvetyne n'a aucune intention de générer de gros profits, et fonctionne sur un système de donations, une valeur attachée à leurs convictions. Avec l'idée que toutes les créations graphiques ou typographiques ne devraient pas faire partie d'une économie de marché, mais devraient être librement accessibles à tous. Si ce système est actuellement applicable à VTF par le fait que ses membres ne sont pas dépendants du projet pour leur rémunération, il semble néanmoins difficilement applicable à plus grande échelle.

# Collectif OSP. Visual Culture, 2014, URL : <http://osp.kitchen/tools/visualculture>, 19 septembre 2021.



## INSIDE THIS REPOSITORY

OSP TOOLS VISUALCULTURE /

- └ ICEBERG
- └ VC\_PYPOPLER
- └ VISUAL
- └ VISUALCULTURE
- └ GITIGNORE APPLICATION/OCTET-STREAM
- └ README-OSP MD APPLICATION/OCTET-STREAM
- └ README TXT TEXT/PLAIN
- └ REQUIREMENTS TXT TEXT/PLAIN

## Versionnage du visuel

Visual Culture est un projet libre, une adaptation de Git (un logiciel de versionnage décentralisé et développé par Linus Torvald (créateur de Linux), une sorte de système d'archivage des versions du code source d'un projet pour le visuel, lancé par le collectif OSP (Open Source Publishing) au début de l'année 2014 sous la forme d'un financement participatif qui n'a pas abouti. OSP, à l'initiative de ce projet, est alors un collectif de graphistes qui ont la particularité de ne travailler exclusivement qu'avec des outils libres et open source. Par ailleurs, ce projet qui a été développé en interne au sein du collectif existait déjà avant le lancement du financement, et existe toujours aujourd'hui, puisqu'il fait partie intégrante de leur site web. On ne retrouve cependant pas toujours certaines fonctionnalités qui auraient dû être développées grâce au financement participatif.

D'autres systèmes tels que Git existaient déjà auparavant, avec des fonctionnements plus ou moins proches de ce dernier, mais les plus performants étaient souvent très chers et proposaient un système fermé et centralisé. Git a alors su se démarquer par un fonctionnement décentralisé et en pair à pair, permettant d'échanger directement les informations entre utilisateurs, sans avoir nécessairement à utiliser un serveur commun qui centralise les données. C'est une démarche qui s'inscrit pleinement dans l'esprit du Web, d'autant plus que Git est un projet libre et donc complètement ouvert.

Mais comme son histoire l'indique, ce projet a été initialement créé pour et par des développeurs. De fait, il rend uniquement compte des changements textuels au sein d'un fichier. Même aujourd'hui, des interfaces visuelles très développées de Git telles que GitLab ou Github ne permettent pas d'afficher de manière efficace les différentes versions d'une production visuelle.

Visual culture repose donc sur cette base qu'est Git, et a pour but d'offrir un logiciel pour visualiser le processus de création, archiver et échanger des données, des changements dans des projets donnant une grande part au visuel tel que le graphisme.

Dans l'idéal, on pourrait alors visualiser les différentes versions d'un fichier, ce qui permet de comprendre et d'explorer l'histoire de ces objets numériques, comme une sorte de système d'archivage. Une fois le fichier modifié, ces modifications peuvent être partagées avec les collaborateurs, mais aussi avec le public, dans une optique d'ouverture qui est propre au libre.

Aujourd'hui, même si le financement participatif n'a pas abouti, ce projet continue de vivre au travers du site web d'OSP.

# Liens au libre

On peut retrouver pour chacun de leurs projets un historique des changements, accessible visuellement sous la forme d'une longue frise chronologique, où chaque changement est visible sous la forme de petites notes, avec un espace plus ou moins important entre chaque en fonction du temps qui s'est écoulé entre deux modifications.

Même si l'on retrouve à l'intérieur des projets une visualisation des fichiers images, il ne semble pas y avoir un historique de ces versions de fichiers, et les notes présentes sur la frise chronologique n'affichent pas d'images.

DENIS VERBALISED	— add README	THURSDAY, 7TH JULY 2016 - 15:15
DENIS UNWRAPPED	— import from zip	THURSDAY, 7TH JULY 2016 - 15:16
COLM CLAIMED	— changes to the project base template to be up to date with gitlab.constant	TUESDAY, 17TH JANUARY 2017 - 12:15
STÉPHANIE VILAYPHIOU TOLD	— move requirements at project root (easier to install)	THURSDAY, 2ND FEBRUARY 2017 - 16:03

Visual Culture est ainsi un outil qui me semble complémentaire à une pratique du graphisme qui se veut plus ouverte, puisque c'est un projet qui permet une collaboration et une étude approfondie des fichiers réalisés par les créateurs, en présentant l'historique des images, des typographies, ainsi que les choix qui ont mené à appliquer ces diverses modifications.

Ce projet offre une autre approche du graphisme, dans laquelle la notion de partage de la connaissance est au centre de la création puisqu'il s'enracine dans la dynamique des logiciels libres. Il a pour but de partager des méthodes de collaboration très utilisées dans le milieu du développement, de les rendre accessibles à un public de professionnels de l'image.

Cette méthode de travail, qui est aujourd'hui très répandue dans le monde de la programmation, me paraît représenter une voie intéressante pour la pratique du graphisme, puisque plus qu'un système de versionnage, Visual Culture s'inscrit dans une démarche ouverte, qui permet de transmettre la pratique du graphisme, son fonctionnement, d'ouvrir les connaissances à tous. Mais qui permet également, me semble-t-il, d'avoir un outil qui pourrait mieux présenter les différentes versions et évolutions d'un fichier ou d'un projet avec un client, mieux réadapter un design en ayant accès à ses anciennes versions, et tout cela de manière simplifiée, avec un système d'archivage puissant.

Anderson, Rasmus.  
Inter, 2016,  
URL : <https://rsms.me/inter/>, 19  
septembre 2021.



Localisation → <https://rsms.me/inter/>

Source(s) → Idem, <https://rsms.me/inter/lab/?antialias=default>, <https://github.com/rsms/inter>

# Libre & international

Inter est une typographie libre et open source conçue par Rasmus Andersson, graphiste et typographe Suisse vivant à San Francisco, qui travaille principalement avec ce qui se rapproche de près ou de loin à des programmes. Cette typographie est une linéale simple et lisible, basée sur une grille et une taille fixe, adaptées pour la lecture sur écran. Elle est très complète et supporte de nombreuses langues, corps, avec une grande quantité de glyphes et de signes, dont des principes qui simplifient l'écriture.

Inter est donc une linéale contemporaine, qui s'inscrit dans la lignée du style international (d'où son nom, qui évoque également le principe d'ouverture, de dépassement des frontières), notamment par sa structure très géométrique, qui rappelle la structure des linéales du milieu du XXe siècle tel que l'Helvetica, ainsi que l'idée de rationaliser, simplifier, favoriser la compréhension rapide et efficace des messages, qu'ils soient textuels ou visuels. Certains pleins et déliés sont plus marqués que sur une Helvetica, permettant ainsi une meilleure lisibilité sur écran avec des lettres bien définies et différenciables, puisque sa fonction principale est d'être intégrée dans des interfaces, pour du titrage autant que du texte courant.

Dans cet objectif d'intégration dans des interfaces, ou d'utilisation courante, Inter intègre de nombreuses subtilités qui permettent une écriture au clavier simplifiée pour obtenir certains signes nécessitant normalement de trouver le code ASCII, des interlettrages modifiés, par exemple pour des formules mathématiques, ou des chasses fixes pour les nombres si nécessaire...

## Features

Inter comes with many OpenType features that can be used to tailor functionality and aesthetics to your specific needs. Some of these features can be combined to form a great number of alternative variations.

Contextual alternates		calt
This feature is usually enabled by default and causes certain characters to adjust themselves or be replaced depending on the surrounding context.		
Disabled	Enabled	
3x9	3x9	
12:34, FE→X	12:34, FE→X	
4.2	4.2	
(SEMI)PERMANENT	(SEMI)PERMANENT	
SFO → STO	SFO → STO	
IIA → OGG	IIA → OGG	
ARN ↔ OGG	ARN ↔ OGG	
M@N m@n	M@N m@n	
Smile :-)	Smile :-)	

Tabular numbers		tnum
Fixed-width numbers are useful for tabular data, where comparing columns across rows is desired.		
Disabled	Enabled	
1234567890	1234567890	
1131711	1131711	
0040900	0040900	
11:31,711	11:31,711	
00:40,900	00:40,900	
0,45, 0,91, +0,08	0,45, 0,91, +0,08	
1,00, 0,44, -0,13	1,00, 0,44, -0,13	
0,00, 1,13, ~7,12	0,00, 1,13, ~7,12	

Fractions		frac
This feature is contextually sensitive and will convert "words" of numbers separated by forward slash into proper fractions. This feature is dynamic and allows for any fractions. Note that the digits used for fractions are custom-made for their small size, and are even made separately from the slightly larger Superscript and Subscript numbers.		
Disabled	Enabled	
1/3 3/4 1/5	½ ¾ ¼	
18/29 16/5	1⁷/₂₉ ₁⁶/₅	
1337/591038	1³³⁷/₅₉¹⁰₃₈	

Case alternates		case
Switches out some glyphs to work better with capital letters and numbers.		
Disabled	Enabled	
(Hello) [World] (9000)	(Hello) [World] (9000)	
SCHOOL @ RUN	SCHOOL @ RUN	
3 * 9 ≈ 12 * 1	3 * 9 = 12 * 1	
*++××#×#×-><#%	*++××#×#×-><#%	
→ ← → ← - - - :	→ ← → ← - - - :	

Compositions		ccmp
Inter provides several custom made glyphs for compositions like A + enclosed-combining-circle.		
Disabled	Enabled	
Figure ⓧ	Figure ⓘ	

Discretionary ligatures		dlig
Alternate style for a few characters. This feature is usually disabled by default.		
Disabled	Enabled	
↳What?	↳What?	

## Font metrics

This font was originally designed to work at a specific size: 11px. Thus, the Units per EM (UPM) is defined in such a way that a power-of-two multiple of one EM unit ends up at an integer value compared to a pixel. Most fonts are designed with a UPM of either 1000 or 2048. Because of this we picked a value that is as high as possible but also as close as possible to one of those common values (since it's reasonable to assume that some layout engines and rasterizers are optimized for those value magnitudes.) We ended up picking a UPM of 2816 which equates to exactly 256 units per pixel when rasterized for size 11pt at 1x scale. This also means that when rasterized at power-of-two scales (like 2x and 4x) the number of EM units corresponding to a pixel is an integer (128 units for 2x, 64 for 4x, and so on.)

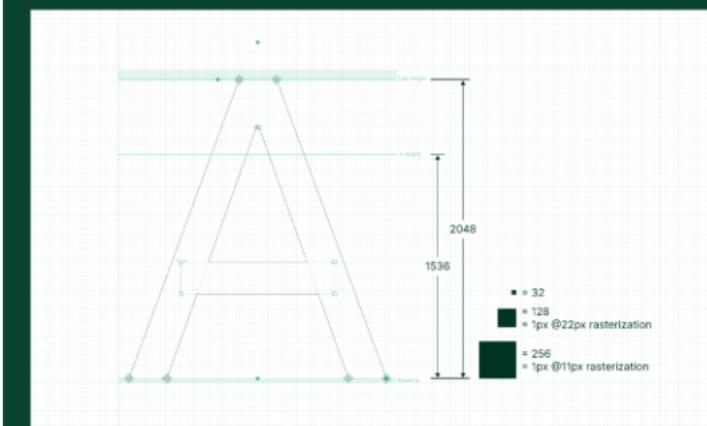
However, as the project progressed and the typeface was put into use, it quickly became clear that for anything longer than a short word, it was actually hard to read the almost monotonically-spaced letters.

A second major revision was created where the previously-strict rule of geometry being even multiples of 256 was relaxed and now the rule is "try to stick with 128x, if you can't, stick with 64x and if you can't do that either, never go below 16x." This means that Inter is now much more variable in pace than it used to be, making it work better at higher resolutions and work much better in longer text, but losing some contrast and sharpness at small sizes.



The glyphs are designed based on this "plan": most stems and lines will be positioned at EM units that are even multiples of 128, and in a few cases they are at even multiples of 64 or as low as 16.

A UPM of 2816 is great for Inter since that means its cap height is exactly 2048 units (64x 32-unit squares) and its x-height is 1536 (48x 32-unit squares) which both makes the design easier (can deal with only integers, never any fractions, plus use a perfect grid) and it makes the target "small size" of 11dp a pixel-perfect match — at 11px rasterization 1 pixel is exactly 256 units in the design! At 11dp with a 2x scaling factor 1 pixel is 128 units, 64 units at a 3x scaling factor and so on. This makes it feasible to really tune Inter for detailed rasterization.



### Metrics:

- UPM: 2816
- Ascender: 2728
- Cap height: 2048
- x-height: 1536
- Descender: -680

Translating between EM units and pixels:

Le site web présentant la fonte est très complet, et permet d'expérimenter avec la typographie, de visualiser tous ses glyphs, d'en apprendre un peu plus sur sa création. En plus d'un site qui explique la typographie et les choix effectués, on peut retrouver cette dernière sur une page Github extrêmement complète qui permet à tous d'y avoir accès, et de participer à son développement.

Il y a de fait une volonté d'ouverture de la typographie, avec de nombreux documents qui permettent à chacun de savoir comment participer à son amélioration. De nombreux formats de fichiers sont proposés, permettant d'utiliser n'importe quel programme de création typographique. Par ailleurs, des outils complémentaires ont été créés pour tester les modifications effectuées (notamment un outil web, le « Lab », que l'on retrouve sur le site de la typographie), et automatiser certaines tâches telles que l'export en de nombreux formats.

# Liens au libre

Si la pratique du logiciel libre peut mener par sa liberté à tester et expérimenter, elle peut aussi mener, comme dans le cas de cette typographie, à créer un outil purement utilitaire, aujourd'hui utilisé par de nombreuses grandes entreprises telles que Figma, Github, Mozilla... Cette typographie est intéressante car elle met en avant une vraie ouverture et une volonté de partager mais aussi de rendre accessible la pratique typographique, et cela au sein d'une typographie qui s'inscrit dans une conception du style international très stricte.

D'autres linéales sous licence libre créées spécifiquement pour des interfaces telles que la Roboto de Google sont aujourd'hui déjà très utilisées, mais outre les caractéristiques qui font que ces deux typographies diffèrent, le principe de développement n'est pas le même. Roboto est développée par une grande entreprise (Google) pour elle-même. Malgré sa licence libre, les designers internes à l'entreprise sont ceux qui vont diriger la conception. De l'autre côté, Inter a été développée par un graphiste typographe, avec un principe clé qui est l'ouverture et le partage, où tout est documenté et simplifié, pour que n'importe quelle personne le souhaitant puisse participer à l'amélioration de la typographie. La liberté ne se rapporte alors pas uniquement à la licence, mais aussi à la manière de créer et partager, ce qui n'est pas nécessairement le cas d'autres typographies ayant plutôt une pratique open source, utilisant plus l'ouverture du logiciel pour une approche économique.

Abbink, Mike.  
IBM Plex, 2018,  
URL : <https://www.ibm.com/plex/>,  
19 septembre 2021.



## Typographie open-source

IBM Plex est une famille typographique libre et open source, très complète, créée par Mike Abbink, typographe et designer d'identités visuelles de marques. Elle a été initiée par IBM pour renouveler leur typographie interne en 2018.

C'est initialement une linéale avec de petites fioritures inspirées des caractéristiques du logo. De fait, la linéale peut se rapprocher des mécanes, avec des empattements de la même épaisseur que les fûts, carrés, et placés en angle droit. Cela met en avant une certaine clarté et une grande robustesse dans la typographie. On y retrouve par conséquent le lien homme-machine qui est au cœur de l'entreprise depuis sa création, avec des linéales mélangeant des empattements mécaniques, et des fûts sans empattements rappelant la Franklin Gothic, aux tracés pouvant par endroits rappeler l'écriture à la plume. Mais plus qu'une linéale, c'est une famille typographique complète composée de glyphes avec ou sans empattements, où les glyphes à empattements reprennent le lien homme-machine visible dans les linéales, avec de nombreuses courbes, notamment au niveau de l'oreille du « r » qui rappelle la Clarendon. On retrouve également un set de caractères condensés, ainsi qu'un set de glyphes monochromes conçu et optimisé en grande partie pour le développement informatique ; une activité qui fait aujourd'hui encore partie intégrante de l'entreprise.

La fonction principale de cette famille est de créer une nouvelle identité typographique pour la marque. C'est une typographie qui par ses caractères diversifiés va permettre d'être diffusée sur de nombreux supports de l'entreprise. Le fait qu'elle soit sous licence libre lui permet aussi de gagner en visibilité puisqu'elle est facile d'usage et accessible gratuitement.

Le site web présentant la typographie est par ailleurs extrêmement complet, et la présente de manière globale, tout en rentrant par instant dans des petits détails, définissant de la sorte chaque choix esthétique et son impact sur ce que véhicule la typographie. Le site est découpé en diverses sections très bien ordonnées, qui permettent réellement de comprendre la volonté du typographe derrière la création de cette famille typographique, ainsi que tous les choix qui l'ont guidé.

Man

A graphic comparison of the letter 'n' from the 'Man' and 'Machine' styles of the IBM Plex font. The 'Man' style is on the left, featuring a dark teal 'n' with a small teal circle at its top left corner. The 'Machine' style is on the right, featuring a light teal 'n' with a large, prominent black 'n' nested inside it, representing the monochromatic design mentioned in the text.

Machine

# Liens au libre

Là où certaines typographies telles que Inter, ou même les diverses typographies que l'on peut retrouver sur Velvetyne, sont diffusées dans un objectif de partage, la volonté d'IBM semble toute autre. En effet, lorsque l'on se rend sur leur GitHub qui contient leur typographie, on ne retrouve que le minimum, avec une page de base présentant avant tout l'entreprise et la manière dont utiliser et installer la typographie.

Dans une autre optique, Inter par exemple, présente sur sa page GitHub des informations complémentaires du site et plus techniques ; mais la page met aussi et surtout en avant les différents moyens de participer au développement de la typographie, avec de fait les règles de conduite qui en découlent, et l'on peut même y retrouver des outils qui ont été créés à l'occasion.

De fait, la démarche de rendre libre sa typographie semble pour IBM être plus une décision qui vise à diffuser la typographie à la plus grande quantité de personnes possible dans un objectif de pure communication, plus qu'une raison éthique qui donnerait la possibilité à chacun d'accéder facilement aux sources de la création typographique.

Ce projet fait ainsi partie d'une sorte de détournement de la pensée du logiciel libre, se rapprochant en réalité plus de l'open source. Malgré tout, le fait de rendre accessible les sources de ses créations graphiques est une démarche intéressante pour une entreprise de cette envergure, sachant que leur site web est particulièrement bien organisé, et exprime clairement les choix esthétiques. Il s'arrête néanmoins en surface dès que l'on touche à des questions plus techniques qui ne sont que très peu abordées, faisant écho à une diffusion qui vise plus le marketing que le partage de l'information.



Meet the IBM Plex® typeface, our new corporate typeface family. It's global, it's versatile and it's distinctly IBM.

We designed the IBM Plex typeface carefully to both meet our needs as a global tech company and express who we are as IBMers. It took two years and a lot of work to get here, but today we have a signature typeface we're proud and excited to share with the world. Discover more about our development of the [IBM Plex typeface](#).

The IBM Plex typeface is an open-source project and available for download and use following the Open Font License (OFL). The IBM Plex family comes in Sans, Serif, Mono and Sans Condensed, all with roman and true italics. The fonts have been designed to work well in user interface (UI) environments, as well as other mediums. This project provides all source files and file formats to support most typographical situations. Currently, IBM Plex Sans supports Extended Latin, Arabic, Cyrillic, Devanagari, Greek, Hebrew, Japanese, Korean and Thai. Chinese will follow in 2022.

Thanks for trying the IBM Plex typeface! We hope you like it.

Add the IBM Plex typeface to your device

Please download the latest zip files from our [releases page](#) for installation.

Web usage

# Bastide, Raphaël. *Each Page A Function*, Paris : éd. LeMégot, 2019.



## Zine et programme

Each page a function est à la fois une publication imprimée de 24 pages format A5 sortie en 2019, et un programme qui lui est associé. Tous deux ont été réalisés par Raphaël Bastide. On retrouve le zine publié sous la maison d'édition Le Mégot, qui édite des fanzines, des livres, des affiches et des objets depuis 2010 à Paris et à Metz.

Raphaël Bastide est un enseignant en art, design et culture numérique à Montreuil. Il a une pratique qui se situe entre l'artiste et le graphiste ; il crée des objets, des programmes, des performances, des instruments et divers outils. Par ailleurs, il publie la plupart de ses travaux sous des licences libres, ce qui est un point majeur de sa pratique et des projets qu'il a lancés ou auxquels il participe ; PrePostPrint, Use & Modify, Fragile Forge, Velvetyne Type Foundry, qui ont tous en commun un ancrage profond dans la culture libre.

Ce zine est donc un petit format, et repose sur le principe d'utiliser des micro-programmes, sous la forme d'une application web (HTML/CSS/Javascript) conçue pour expérimenter l'acte de dessiner. Cette application propose donc plusieurs outils (programmes) qui perturbent le processus de dessin en ajoutant des contraintes ou des augmentations, mais aussi des comportements temporels, spatiaux, auditifs ou formels. Cette micro-édition présente alors une couverture rouge, typographique, dont le dessin des lettres a été réalisé à partir d'un des programmes créés.

L'intérieur est composé de 19 dessins numériques réalisés avec 19 outils développés pour le projet, et l'on retrouve sur ces pages uniquement les visuels créés, centrés et en gros plan, avec pour chacun la fonction utilisée en bas de page. A la fin, une sorte de glossaire présente le temps utilisé pour la création des micros programmes, et le temps utilisé pour réaliser le dessin. Le zine en lui-même se présente presque comme un catalogue d'œuvres puisqu'on y retrouve chaque fonction qui est mise en relation avec des créations permettant de l'illustrer.

En plus de l'édition qui a été proposée, le site web en lui-même est disponible de manière libre, sur le web, et permet à chacun de créer sa propre édition en utilisant les mêmes contraintes artificielles que Bastide. L'édition finale peut être téléchargée au format PDF, donc imprimée. De plus, toutes les sources ayant permis la création du zine sont disponibles en accès libre sur GitLab.

On retrouve dans cette création une grande part d'expérimentation, qui permet de repenser l'acte de dessiner, mais aussi et surtout de créer de nouveaux outils avec des fonctions très spécifiques. Des outils créés expressément pour un projet, une tâche précise, ce qui va à l'opposé des logiciels de l'industrie graphique tels que la suite Adobe, et son écosystème qui veut être capable de pouvoir tout faire, avec des outils qui se veulent universels, pouvant mener à un formatage de la manière de concevoir.

# Liens au libre

La pratique du dessin n'est par ailleurs plus uniquement visuelle, puisqu'elle intègre ici l'accès au microphone de l'ordinateur pour utiliser l'outil audioTrembling() amenant de l'aléatoire dans la création, une part expérimentale. En outre, l'objet en lui-même, sous forme de zine, reste ancré dans une pratique expérimentale, et semble plus être fait pour présenter le but du projet que pour créer un grand volume d'éditions destinées à la vente. En effet, on ne retrouve sur le site de l'éditeur que 30 tirages, chacun vendu au prix de 5€.

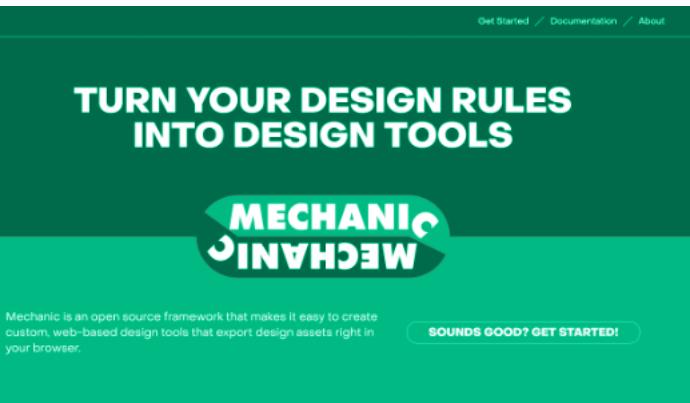


Utiliser le web comme outil de création est une notion intéressante, notamment dans le milieu de la publication, car elle permet de lier encore plus fortement deux types de supports d'édition : le physique et le numérique. Par ailleurs, les langages du web et le web en lui-même étant dès l'origine des plateformes de publication de contenu, notamment écrit, il me semble qu'il y a un vrai intérêt à lier ces deux champs, d'autant plus sur une plateforme aussi ouverte.

Une des fonctions intéressantes de ce programme est celle nommée frameOfVersions(). Elle permet de prendre automatiquement des « captures » de chaque étape de création de l'illustration, qui sont définies par un tracé. Ces captures des étapes sont alors disposées tout autour de l'illustration finale, permettant facilement de visualiser les actions qui ont été effectuées sur la composition globale. On a alors une sorte de micro logiciel de versionnage, à la manière d'un git du visuel, avec des archives de chaque étape de la création de l'illustration.

La notion d'aléatoire et d'expérimentation qui est ressortie de ce projet est quelque chose que l'on retrouve souvent dans la création libre, grâce au concept de « fork », ou le fait de se réapproprier un projet et d'en créer un nouveau. Ainsi, les projets libres permettent à nos connaissances, outils et systèmes d'être ouverts et partagés, d'évoluer et d'être façonnés en fonction des différents usages que les individus ou les studios peuvent imaginer. D'ailleurs, les micro-programmes de ce projet sont eux-mêmes basés sur un autre projet libre nommé Paper.js, et sûrement d'autres pour l'export en PDF.

# Design Systems International. *Mechanic*, URL : <https://mechanic.design/>, page consultée le 2 novembre 2021.



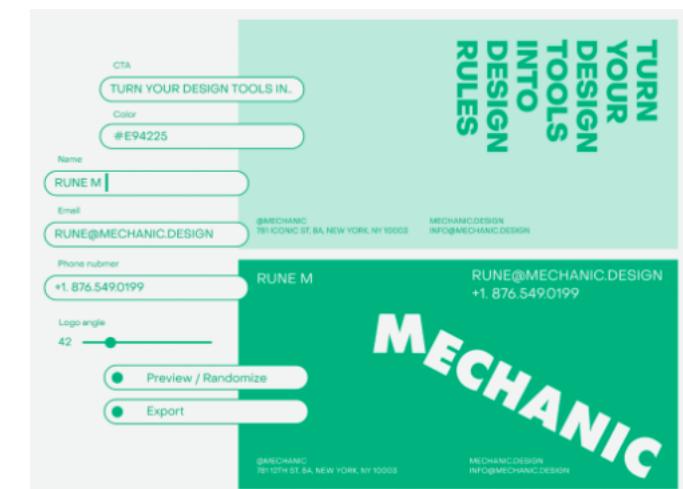
## Logiciel personnalisé

Ce projet nommé Mechanic est un framework libre (un ensemble de composants/structures logiciels, pouvant permettre de développer un logiciel) qui facilite la création d'outils de design personnalisés, et basés sur les technologies Web. Il permet donc de créer soi-même son propre outil destiné au design graphique. Ce framework est ainsi développé par Design Systems International, un récent studio de design lancé en 2017, et fondé par Rune Madsen et Martin Bravo, avec des studios à New York, Copenhague et Santiago de Chili. Ce studio explore le génératif et les moyens de créer des systèmes graphiques, notamment dans le graphisme et les médias numériques, principalement pour le développement d'identités visuelles.

Le studio utilisait déjà le code dans le cadre de leur processus créatif, en concevant des prototypes fonctionnels, permettant d'expérimenter diverses compositions plutôt que de trop se fonder sur des lignes directrices. Ils accordent ainsi une part très importante au génératif dans leurs créations. Par ailleurs, ces derniers ont récemment refait toute l'identité visuelle de la Fondation Processing (créatrice notamment de Processing, un environnement de développement dédié à la programmation de visuels) à partir d'un outil développé par leur soin, ayant notamment permis de créer les nouveaux logos des divers produits de la fondation.

C'est ainsi à la suite du développement de nombreux outils en interne que le studio va décider de développer Mechanic, et de le proposer de manière libre.

Fondé sur des technologies web, Mechanic a alors pour but de permettre de créer des systèmes visuels facilement répliquables, en s'éloignant de fait des outils de conception actuels, qui sont calqués sur des processus manuels, et qui malgré les progrès technologiques, peuvent exiger un travail répétitif de la part des designers. Mais plus qu'un simple framework, Mechanic permet aux designers de créer leurs propre interface en fonction du projet, puisqu'au lieu d'avoir un programme qui nécessite de toujours modifier le code pour appliquer des modifications, des composants permettent ici de créer facilement des champs à remplir, des boutons, des sliders, etc. Finalement, de créer une interface entière, et donc un outil spécifique pour le projet.



# Liens au libre

À noter cependant que toutes les interfaces d'outils créées sont présentées sur une seule colonne, ce qui limite l'ergonomie, et pousse à ne créer des outils que pour les éléments variables qui vont être modifiés.

En outre, Mechanic intègre divers outils web très répandus, et notamment p5.js, la version Web de Processing. De fait, cela étend encore plus l'intérêt de ce framework, dans lequel on va pouvoir créer non seulement des visuels à l'aide du code (et même créer ses propres éléments d'interface permettant de modifier le code), mais aussi y intégrer de la typographie, une mise en page réfléchie, et de fait, permettre un export vers divers supports, notamment imprimés.

Par ailleurs, le site web présentant le projet est très complet, et propose notamment un guide pour savoir comment fonctionne l'outil, comment l'installer, démarrer un projet... Ceci est très caractéristique de ce que l'on peut retrouver dans les différents frameworks et travaux consacrés au Web.

En plus de cela, on peut accéder à une sorte de projet « test » sur la page du site, permettant d'avoir un aperçu de ce dont est capable de produire cet outil, et même d'exporter notre production finale.

L'intérêt de cet outil réside dans la grande modularité qu'il permet d'avoir, tout en essayant de combler les problèmes qui lui sont intrinsèques.

En effet, l'outil propose une plus grande flexibilité et puissance dans la création par le développement, puisque ce dernier permet en théorie de tout faire. Néanmoins, cela demande des connaissances techniques qui peuvent se révéler être très importantes.

Mais ici, Mechanic tente de résoudre une partie du problème en proposant au sein du développement la possibilité d'intégrer facilement ses propres éléments d'interfaces, qui faciliteront les modifications de placement, rotation, alignement, couleur, etc.

De fait, ce framework propose de mettre en place son propre logiciel de design graphique. Si cette notion peut sembler impressionnante aux premiers abords, elle est facilitée dans un premier temps par le fait que tout est très bien documenté, mais aussi par l'usage de standards souvent connus des Designers tels que les langages du Web, mais aussi d'autres outils puissants tels que Processing, permettant donc d'élargir encore plus les possibilités proposées par Mechanic.

Malgré tout, cette solution nécessite une connaissance du développement informatique, ce qui n'est pas nécessairement le cas de tous les graphistes aujourd'hui, et en empêche donc une partie de pouvoir y avoir accès.

## DOCUMENTATION

- Get started
- Design functions
- Templates
- Examples
- Add a function
- Add an input
- Text
- Number
- Color
- Boolean
- Image
- Slider
- Select
- Add a preset
- Handler arguments
- Select an engine
- Canvas
- RWS

## GET STARTED

Mechanic\* is a powerful open source framework that helps forward-thinking designers move away from a manual design workflow by converting their design rules into code.

To start working with Mechanic, you need to create a project. Run the following command in the terminal:

```
npm init mechanic@latest my-project
```

Copy

Make sure you have Node.js (version 12.20 or greater) and npm installed.

This will install the Mechanic command-line interface and prompt you with instructions on how to customize your new Mechanic project. You can choose between creating a new design function based on a template or a more sophisticated example. As a result, you will end up with a folder with your chosen project name and the following files in it:

- `README.md` contains some pointers on how to start using your Mechanic project.
- `package.json` contains all the dependencies needed to use Mechanic.
- `mechanic.config.js` contains the Mechanic configuration.
- `mechanic.js` is a folder that will contain your design functions.

## DESIGN FUNCTIONS

Design functions are JavaScript functions that generate a design asset. They are the main building block of Mechanic, and a project can have one or more design functions. Based on the content in the design functions, Mechanic creates a design tool you will be able to use in your web browser.

The generated asset can be either static or animated, simple or complex. A design function creates assets using a specific web technology and a corresponding Mechanic engine that enables it. Currently, the available options for engines are SVG, Canvas, React.js and p5.js.

Each design function definition should live in a separate folder in the `designs` folder. Each function folder will contain an `index.js` file with the following elements:

**Main file**  
This is where you define the code that processes your design and this code will be passed to the chosen engine as a `function()` function. Always receives a single argument, and your own variable to handle the value based on the value in the object. Learn more about [functions](#).

**Inputs**  
`inputs` is an object defining the variables that will be passed to your design function. Each input has a type and a default value. Use `inputs` to change these inputs and run the function with updated values. Learn more about [inputs](#).

**Precise**  
Provides you the opportunity to set precise values or values for the inputs that can't be set with a single click. The design tool can reflect with a single click. Learn more about [precise](#).

**Settings**  
Allows you to set general configuration for your function. This is where you specify the engine that your design function will be using among the design function provides a value or an array with `mechanic.set`.

## TEMPLATES

Templates are simple design functions created to show you how to use Mechanic with specific web technologies for either animated or static assets. You can use one to get to know Mechanic, or use one as a base to start your design function.

Use the `npm init mechanic@latest` command to create a new Mechanic project with a single design function based on a template. Once you have created your Mechanic project, you can use the `mechanic new` command to add more design functions.

The currently available templates you can find are:

- `Canvas Image`: Produce a static asset using the Canvas engine.
- `Canvas Video`: Produce an animated asset using the Canvas engine.
- `p5.js Image`: Produce a static asset using the p5.js engine.
- `p5.js Video`: Produce an animated asset using the p5.js engine.
- `React Image`: Produce a static asset using the React engine.
- `React Video`: Produce an animated asset using the React engine.
- `SVG Image`: Produce a static asset using the SVG engine.

Hyde, Adam.  
Paged.js, 2017,  
URL : <https://www.pagedjs.org>,  
19 septembre 2021.

## Logiciel d'édition augmenté

Paged.js est une bibliothèque Javascript libre et open source (un ensemble de fonctions déjà créées permettant d'étendre les possibilités d'un langage de programmation, et de simplifier le développement), qui permet de réaliser des mises en pages de documents imprimables, en PDF, directement depuis un navigateur.

Ce projet s'adresse principalement au domaine de la production, et a été initié par Adam Hyde en 2017, qui est aussi à l'initiative de nombreux autres projets mettant en avant l'open source dans le milieu de la publication, au travers de nouvelles manières de la repenser. Il dirige des projets techniques, construit des plateformes, des méthodologies au sein de Coko (Collaborative Knowledge Foundation), afin de soutenir la production et la publication de documents et connaissances en open source. Aujourd'hui, Julie Blanc est l'une des contributrices les plus importantes de ce projet, en apportant notamment son savoir en CSS. Elle prépare actuellement une thèse en ergonomie et design graphique, questionnant notamment la mise en page et l'automatisation dans le milieu du graphisme.

Paged.js est ainsi une bibliothèque Javascript qui va permettre d'étendre les possibilités de langage de base du Web, telles que le CSS, afin de l'adapter à la production de mises en page. Cette extension de langage s'appelle alors un Polyfill, ou prothèse d'émulation, puisqu'elle permet de faire supporter de nouvelles normes aux navigateurs qui ne les supportent pas encore.

Le projet s'inscrit dans une démarche ancrée dans l'esprit du Web, puisque les nouvelles fonctions ajoutées au CSS suivent les normes de médias paginés publiées par le W3C, l'organisme qui gère les normes mises en place au sein du web.

Ainsi, grâce à cette bibliothèque et à l'apprentissage de certaines de ses nouvelles fonctions, il est possible de créer des documents dédiés à la publication au sein même de son navigateur, en utilisant les langages standards du web tels que le HTML et CSS. Avec un peu d'entraînement, n'importe quel graphiste ayant des connaissances sur la création de sites web peut créer des objets de publication allant du fanzine à un livre plus étoffé (pour lequel Paged.js a été imaginé), au sein d'un navigateur. Il y a ainsi cette volonté de créer un système complètement ouvert, aussi accessible grâce à un GitLab, mais surtout un site web très bien documenté, que ce soit au niveau de la mise en place globale d'un projet ou des divers petits détails de la bibliothèque.



by Cabbage Tree Labs

OPEN SOURCE · STANDARDS COMPLIANT · COMMUNITY DRIVEN

# Liens au libre

« L'auteur et le designer travaillent ensemble sur le livre, chacun a ses fichiers. Au vu de la mise en page, chacun peut décider de changer des choses, l'auteur peut couper du texte par exemple, le designer peut déplacer des éléments, ajuster localement ses règles de mise en page, mais en évitant d'intervenir dans le fichier de l'auteur pour cela. »

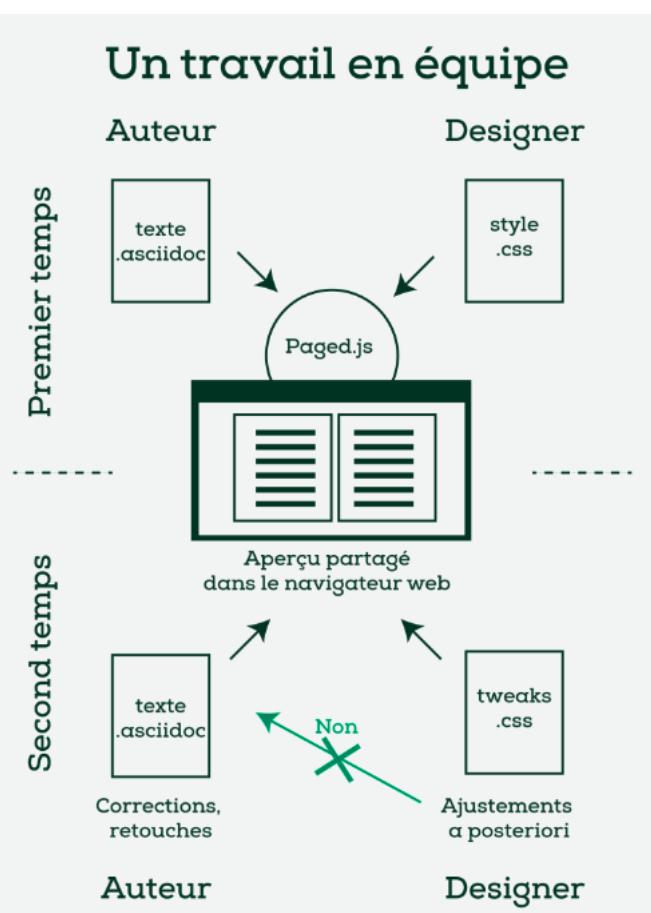
Taffins, Nicolas. « Dans les recoins de la double page (Paged.js à la maison, saison 2) », dans le blog Polylogue, 1 mars 2021, URL : <https://polylogue.org/apres-la-page-la-double-page/>, 27 octobre 2021.

Ce qui est intéressant ici, c'est que Paged.js permet de complètement repenser la manière de faire de la mise en page, en utilisant des standards et formats complètement ouverts. De plus, le fait que la bibliothèque se base sur les langages du web, donc des langages de programmation, permet également de mettre en place diverses expérimentations en utilisant des éléments génératifs autant que des animations. Paged.js a de fait été repris par PrePostPrint, qui est à la fois un label et un site web dont l'objectif est de promouvoir des objets et des ressources graphiques basées sur des créations expérimentales, libres, et bien documentées. Il a pour but de rassembler divers logiciels et travaux utilisant des techniques d'édition expérimentales et d'aider à rendre leurs projets et outils plus accessibles. Cette bibliothèque permet également de repenser la publication de manière plus modulaire et plus ouverte, comme le souligne Nicolas Taffin, qui travaille aux éditions C&F ; « L'auteur et le designer travaillent ensemble sur le livre, chacun a ses fichiers. Au vu de la mise en page, chacun peut décider de changer des choses, l'auteur peut couper du texte par exemple, le designer peut déplacer des éléments, ajuster localement ses règles de mise en page, mais en évitant d'intervenir dans le fichier de l'auteur pour cela. ».

En effet, en travaillant directement chacun de son côté sur des fichiers qui vont ensuite être réunis dans la mise en page finale, il est possible de créer un livre où les textes de l'auteur ainsi que les modifications du graphistes peuvent être appliquées en temps réel grâce à un serveur hébergeant les fichiers, comme le fait actuellement Nicolas Taffin au sein de la collection Interventions (en utilisant ici NextCloud, une alternative libre au drive de Google). Ainsi, avec quelques idées et extensions, Paged.js peut devenir une plateforme d'édition en temps réel, ce qui est notamment permis par son ancrage dans une logique de logiciel libre.

En outre, Paged.js met aussi en avant la question de la pérennité des fichiers. En effet, là où des logiciels tels que InDesign de Adobe utilisent des formats de fichiers propriétaire et complètement opaques sur leurs fonctionnement, Paged.js a la force de proposer des formats de fichiers supportés par quasiment n'importe quel logiciel d'édition de texte brut, puisque ce dernier repose sur des langages de programmation ouverts. Si ici on a un cas extrême avec des langages web, il semble néanmoins que les outils libres ont cette force de reposer sur des standards, et à minima sur des formats de fichiers ouverts, permettant de fait d'avoir des créations qui ne dépendent pas d'un seul outil, donc des créations qui vont avoir la capacité d'être réutilisées dans le temps.

## Un travail en équipe



# Creuzet, Quentin. « Croisades graphiques – le minimalisme à l'ère du néolibéralisme », dans revu Pli, n° 6, septembre 2020, p.117-124.

« En ressort un web où tout un chacun est contrôlé, scruté, testé, mesuré avec des technologies de plus en plus évoluées qui viennent remettre en cause le rôle éminemment subversif du designer, pour n'en faire qu'un robot à la solde du marketing digital. »

p.122

« L'utilisateur, lui, se retrouve cantonné à évoluer dans un environnement à l'apparence simpliste, où tout semble facile à l'utilisation et n'a ainsi plus à questionner ses usages. »

p.122

## Libre et minimalisme

« Croisades graphiques – le minimalisme à l'ère du néolibéralisme », est un article, un essai rédigé par Quentin Creuzet, designer et développeur Web, cofondateur du studio de design F451 avec Domitille Debret.

Cet article est présent dans le 6ème numéro sorti en septembre 2020 de la revue Pli, fondée en 2015 par 3 jeunes sortant d'école d'architecture et de design, dont Christopher Dessus, Marion Claret et Adrien Rapin.

Au sein de l'article, Quentin Creuzet fait part d'une vision personnelle du design et de son évolution dans notre monde actuel, empreint de cette nécessité absolue de tout rapporter à un besoin de revenus et de bénéfices.

Ainsi, son article se découpe en 3 étapes, qui vont nous faire explorer le design autant que l'architecture, afin d'emmener sa réflexion vers l'état et la forme du minimalisme actuel dans le web design.

Pour cela, il commence par faire un état des connaissances actuelles que l'on a sur le Design, sur son histoire. Il va notamment s'appuyer sur un exemple phare ; Massimo Vignelli. Ce grand porte-parole du style international en Amérique est notamment reconnu pour sa carte du métro New-Yorkais, caractéristique du style international qui prône le minimalisme. Finalement, ce dernier n'est aujourd'hui qu'un style visuel, ayant mis de côté la vision politique qui en est issue, faisant part de la plus grande neutralité, en utilisant Helvetica pour toute chose.

Dans une seconde partie, Creuzet veut nous rappeler que le minimalisme n'est à l'origine pas seulement formel. Plus qu'un courant esthétique, c'est une pratique, un processus, un mode opératoire, et il le démontre en provoquant plusieurs exemples.

Il parle ainsi de « croisade » et de « règne du minimalisme » (p.117-118), deux mots qui font part d'une position extrême dans les choix du minimalisme au début du XXe siècle, alors portés par Jan Tschichold, et qui vont lui valoir d'être en partie repris par le fascisme allemand. Il cite ensuite Müller-Brockmann, et la transparence qui est mise en point d'honneur dans la pratique du designer, puis l'architecture avec le brutalisme, reposant sur cette même idée de laisser transparaître la structure et le matériau utilisé.

Finalement, dans une dernière partie, il nous fait part de cette transposition du mouvement brutaliste dans le web, en reprenant les termes et règles qui définissent l'architecture brutaliste et en les adaptant à la page web.

Il fait part de la volonté de minimalisme et de transparence qui est à l'issue de ce mouvement, qui arrive en désaccord avec les pratiques actuelles du design sur le web.

# Liens au libre

En effet, sur nombre de sites contemporains, le design vise le plus souvent à se servir de l'utilisateur comme source de revenus, aux moyens d'un pseudo minimalisme de surface qui ne pousse qu'à lui faire oublier de questionner sa pratique.

Le designer n'est alors plus « qu'un robot à la solde du marketing digital » (p.122).

Le brutalisme sur le web se veut à contre-courant en défendant les valeurs premières de la plateforme. Mais aujourd'hui, le brutalisme a été repris comme effet de mode par de nombreuses marques, et se retrouve n'être plus qu'une coquille esthétique, vidée de tout sens. Ainsi, il se questionne sur la place du designer, ses convictions liées à des idées formelles étant absorbées par un marketing qui recrache uniquement de l'esthétique et qui broie l'idéologie d'origine.

Cet écrit est intéressant car il permet de faire un autre parallèle ancré dans le courant du logiciel libre initié par Richard Stallman. Le libre tire son origine du logiciel et notamment d'un système d'exploitation nommé GNU (GNU is Not Unix), diffusé sous un nouveau type de licence qui donne une liberté totale à l'utilisateur sur le programme (la licence GNU GPL).

Il manquait alors un noyau libre pour avoir un système complet, et Linus Torvald en a développé un, qu'il a par la suite raccroché au projet de Richard Stallman : GNU/Linux est né. Derrière ce système libre, l'architecture complète est faite pour être la plus transparente et compréhensible possible par un humain (une pratique en partie héritée de la philosophie Unix). Dans la mesure du faisable, Linux rend ses composants disponibles via des fichiers ou des objets qui ressemblent à des fichiers. Les processus, les périphériques, les sockets réseau sont tous représentés par des objets semblables à des fichiers.

Mais cette liberté totale d'utilisation que l'on retrouve au travers de nombreux logiciels a aujourd'hui été en grande partie détournée avec l'open source, qui même s'il base ses règles sur le libre, ne soutient pas nécessairement l'éthique qui y est liée.

Ainsi, l'open source est aujourd'hui repris par des géant de la tech tel que Alphabet (Google), Meta (Facebook), et même Microsoft, sous diverses formes en y ajoutant des surcouches propriétaires ou des licences multiples qui ne sont pas complètement libres.

Ces logiciels au semblant de libre ne sont alors plus qu'une couche superficielle, omettant souvent toute la philosophie derrière le projet.

Une autre partie intéressante du texte est le fait décrit page 122 : « L'utilisateur, lui, se retrouve cantonné à évoluer dans un environnement à l'apparence simpliste, où tout semble facile à l'utilisation et n'a ainsi plus à questionner ses usages. ». Si Creuzet parle ici de l'utilisateur, le designer peut se retrouver dans un même environnement, au travers d'Adobe et de produits voulant effacer toute difficulté au profit de l'efficacité et (selon eux) de la créativité, le menant de fait à ne plus questionner ses outils.

Maudet, Nolwenn.  
*Designing design tools*, 2017,  
URL : <https://designing-design-tools.nolwennmaudet.com/>, 3 novembre 2021.

« *The stagnation of the design tool landscape led to the progressive invisibility of design software in designers practice.* ».

« *Experienced designers know that off-the-shelf, general software tools obscure the potential of software as a medium for expression and communication. Writing custom, unique tools with software opens new potentials for creative authorship* » (Reas, C. & McWilliams, C. (2012).  
Progammer avec Erik van Blokland, Catalogtree, Amanda Cox, Nicholas Felton, FIELD, LUST, Boris Müller, onformative, Jonathan Puckey, Sosolimited & Trafik, Graphisme en France, code>>outils>>design. Centre National d'Art Plastique.)

## Un outil plus ouvert

Cette thèse de doctorat en Interaction Homme-Ordinateur a été réalisée en 2017 par Nolwenn Maudet, designer d'interaction et chercheuse en design, travaillant notamment au sein du collectif BAM qui prône l'autonomie. De fait, cet écrit prend un ton académique, presque scientifique puisqu'il va avoir une démarche fortement basée sur des pratiques que l'on retrouve au sein de la culture scientifique, avec différents tests proposés directement à des designers. Nolwenn Maudet explore alors le fond tant que la forme des logiciels de production graphique, et remet en question le fonctionnement de ces logiciels, en les opposant à la manière de créer et d'utiliser les outils des designers aujourd'hui. Elle met en avant d'autres manières d'imaginer le logiciel du graphiste, notamment au travers du développement et d'outils qui proposent plus de liberté. En effet, les logiciels actuels, et notamment ceux de la suite Adobe Creative Cloud, n'ont que très peu évolué depuis les 3 dernières décennies, et sont devenus tellement omniprésents aujourd'hui que l'on a tendance à omettre de les questionner : ils en deviennent alors invisibles.

Cette thèse porte donc sur un décalage entre la pratique quotidienne des graphistes et les outils actuels du design graphique qui sont proposés.

Ainsi, les outils de conception traditionnels tels que la suite Adobe dissocient la créativité de l'utilisation de l'outil, en priorisant plutôt des valeurs comme l'efficacité et la facilité d'utilisation, qui permettraient d'avoir plus de temps pour être créatif.

La thèse se décompose en deux grandes parties, présentant en tout 13 chapitres.

Dans la première partie de sa thèse, Nolwenn Maudet étudie les pratiques des designers, au travers d'outils numériques qui permettent une étude de leur comportement au quotidien ou directement dans les logiciels qu'ils utilisent. Ces études sont basées sur 4 principes utilisés par les graphistes dans leur travail, tels que la couleur, l'alignement, la mise en page, et la collaboration.

Dans une seconde partie, une fois ces pratiques définies par une étude approfondie des comportements des designers graphiques et de leur nécessité, l'auteur s'intéresse à la création d'outils de conception qui soutiennent ces principes majeurs et leur exploration par des designers.

Nolwenn Maudet propose ainsi de repenser les logiciels de conception, de les transformer pour en outre rapprocher la pratique du développement, en la liant à l'interface graphique. Le développement informatique est en effet une pratique vers laquelle se tournent de nombreux designers afin de retrouver une plus grande liberté dans leur conception.

# Liens au libre

« Why couldn't we accept that tools influence us and that we could choose them depending on their impact ? Shouldn't we ask ourselves which tool is appropriate before mechanically resorting to our usual software ? » (Donnot, K. (2011). Outil Numérique et Design Graphique. Thèse, École des Beaux-Arts de Rennes. )

« That is why, for example, a command approach to perform alignment does not necessarily ease designers work. In fact, this approach limits exploration. »

« Therefore I argue that we need to preserve the power of graphical user interface tools while enhancing them with more computational power. »

Dans l'optique de repenser son rapport à sa pratique, et donc à l'outil, cette notion d'invisibilité de l'outil me semble très intéressante. En effet, la pratique du designer revient sans cesse à questionner les outils qui peuvent lui servir pour une tâche autant que pour un résultat donné, comme on questionnerait le choix d'utiliser une imprimante laser plutôt qu'une risographie. L'outil, notamment pour le designer, n'est pas quelque chose qui est censé aller de soi ; il engendre des questions esthétiques, techniques, voire même parfois éthiques, qui font partie du processus de création et ne devraient pas être prises comme acquis.

Par ailleurs, au sein de sa thèse, Nolwenn Maudet présente la création de ces outils basés directement sur la pratique des graphistes, et la manière avec laquelle ces derniers se réapproprient ces outils en les adaptant à une pratique plus personnelle. Cette réappropriation passe par une possibilité accrue d'adapter ses outils. Ils peuvent de fait évoluer par leur flexibilité, permettant de créer des outils puissants. Cette manière d'imaginer l'outil du graphiste comme malléable s'inscrit complètement dans l'éthique hacker, mais aussi dans la démarche libre, et se rapproche de fait des outils libre actuels qui ont tous cette capacité d'adaptation par la programmation sous de nombreuses formes (que ce soit directement en ligne de codes ou par des systèmes d'imbrication, de nodes...).

Autre fait intéressant qui n'est mentionné qu'à la fin du site, cette thèse a été rédigée uniquement sous logiciels libres en utilisant le format web avec du HTML, et la thèse imprimée a été exportée en utilisant HTML2PRINT, un logiciel permettant de convertir une page HTML en PDF imprimable développé par OSP. Par ailleurs, dans la même lignée qu'Anthony Masure, cette thèse a été publiée sous la licence CC-BY-NC-SA, intégrant d'autant plus ce document dans une pratique du libre par l'utilisation de logiciels alternatifs spécifiques qui permettent de passer du format Web à l'imprimé.

# Masure, Anthony. Design et humanités numériques, Paris : éd. B42, coll. Esthétique des données, 2017



## Le designer et ses outils

*Design et humanités numériques* est un essai rédigé de mars 2015 à mai 2017 par Anthony Masure, chercheur en design. Le livre a été publié en 2017 en tant que premier livre de la collection Esthétique des données aux éditions B42, mais aussi comme premier texte sous licence libre de la maison d'édition. Cette dernière est réputée pour ses ouvrages consacrés au domaine du design et de l'art contemporain. Ce texte est ainsi un essai qui questionne et met en relation le design et les humanités numériques, pour une collection qui portait à l'origine le nom de 404. Le travail sur cet essai est issu de la proposition de Nicolas Thély, chercheur et professeur en art, esthétique et humanités numériques dans le jury de thèse d'Anthony Masure, de rédiger un prolongement de sa thèse.

Ce livre prend deux formes : une forme imprimée qui reprend une partie de sa thèse, et en extrapole une autre au travers de 7 chapitres qui peuvent être lus indépendamment, et un site web qui contient des contenus complémentaires. Cette seconde partie en ligne permet d'avoir accès librement à des contenus complémentaires tels que des images, des descriptions et analyses d'objets, qui ont été retranchés de la version imprimée pour des questions de place et d'encombrement liées à une volonté d'accessibilité.

Accessibilité notamment économique mais aussi due à son format réduit.

Le site permet dès lors de trouver des images qui complètent ces contenus, et le site devait également proposer un an plus tard une version de l'ouvrage sous licence libre dans son intégralité.

A travers cet essai, Anthony Masure tente de nous faire prendre du recul sur les liens qui peuvent exister entre le design et les enjeux de ce que l'on appelle « recherche et développement ». Il lie alors le design aux humanités numériques, qui pourraient être définies comme l'association de pratiques savantes en littérature, en sciences humaines, avec des matériaux numériques de tout ordre, qu'ils soient logiciels ou matériels. Ainsi, ce qui rapproche selon lui ces deux domaines est que les humanités numériques, à l'instar du design, sont aujourd'hui confrontées à des processus numériques depuis de nombreuses années. Il est de fait logique que ces deux champs se rencontrent, autant dans la production que la transmission des savoirs, au travers des matériaux numériques.

Au sein de l'essai, il développe le fait que le domaine de la recherche et du développement est aujourd'hui soumis à des exigences de rentabilité, et qu'il semble que le design connaisse fréquemment, en tous cas dans le langage commun, l'objectif de faire vendre.

# Liens au libre

Il souhaite de fait remettre en question ce point de vue, avec une idée du design qui pourrait au contraire déjouer ses attendus de rentabilité qui peuvent venir normer nos environnements (on peut le voir notamment au travers de systèmes d'exploitation tels que Android et iOS, qui sont de plus en plus semblables à chaque itération, ou même au travers de sites web de vente, utilisant toujours les mêmes structures). Le design devrait venir en opposition à cette évolution actuelle de notre paysage, et devrait entamer des réflexions visant à réinventer nos cadres de vie, à questionner de nouveau ce qui existe avec l'apport du numérique, plutôt que de seulement l'adapter. Ainsi, dans le cadre de la recherche d'Anthony Masure, le design n'a pas seulement pour vocation de mettre en forme des contenus produits, mais il peut aussi re-questionner la façon même de produire les savoirs, que ce soit par leur valorisation, leurs formes graphiques, ou même les différents médias sur lesquels ceux-ci s'inscrivent.

Anthony Masure défend ainsi l'idée que le design a plus qu'une simple valeur d'exposition, de besoin ergonomique, qui peuvent mener à de plus en plus de normes. Il a une capacité de transmettre le savoir au moyen de démarches qui peuvent être plus expérimentales, et notamment au sein des Humanités Numériques. De fait, comme il le dit dès la fin de son introduction, il y a de la place pour une autre forme du design, qui ne suit pas la seule logique industrielle.

Pour la suite, je me suis alors plutôt intéressé au troisième chapitre, nommé "L'injonction à la créativité et à l'innovation", de la page 57 à 73. Ici, Masure s'intéresse à la place de la productivité et au terme d'innovation dans le champ du design (et notamment du logiciel), à ce qui permettrait de déplacer des habitudes, des attendus, que l'on impose au design ou que le design s'impose parfois à lui-même. Ce qu'il redoute est un design à sens unique, tourné vers une logique d'efficacité, mais qui risque de ce fait d'être de plus en plus normé. Au contraire, il aimerait aller vers un design qui active une pluralité de possibles, qui ne soit pas à sens unique. Il cherche des questions plus ouvertes que le seul objectif d'être une machine à produire, ce qui est notamment mis en avant par la « Suite Creative Cloud », fonctionnant comme une injonction à la créativité, à la nécessité d'être créatif.

Dans une première partie introductory, Masure pose de nombreuses questions, et notamment celles-ci : « Où situer alors le travail du designer ? Travaille-t-il pour l'utilisateur ou pour le profit des usages que les firmes privées souhaitent faire de cet espace ? » (p.57). Ces dernières me semblent intéressantes, car elles posent une question fondamentale sur le design qui me semble bien lié à la pratique du libre. Le libre, par son fonctionnement, donne un accès complet à la connaissance de son outil, en autorisant à faire ce qu'il veut du programme, tout cela dans un objectif de donner une liberté totale à l'utilisateur.

Cette volonté est apparue en opposition aux logiciels « privateurs », qui peuvent avoir pour objectif de se servir de l'utilisateur comme objectif premier, et non pas le servir.

La deuxième partie continue de poser cette question de service, notamment à travers cette citation : « Celui qui est "au service de" n'est pas libre, il cherche, dit Montaigne, "à s'enrichir". » (p.60). Il revient sur l'utilisation d'interface simplifiées, et de fait vantées par les sociétés, mais qui en réalité sont une expérience réfléchie de bout en bout afin d'absorber le temps de l'utilisateur. Est-ce vraiment là le travail du designer ? De fait, comme pour le logiciel, il me semble que le designer doit avant tout créer en ayant pour objectif de servir l'utilisateur, et non pas de l'utiliser dans des objectifs de rentabilité à la manière d'Instagram et de son design créés pour capter l'attention. L'objectif est d'avoir une pratique la plus transparente possible.

Dans un troisième temps, Masure présente les différences fondamentales entre le design innovant et le design inventif. Le terme innovation n'est pas historiquement issu du champ du design. Il vient plutôt du milieu de la gestion, de l'économie, du brevet, de la mise au secret des procédés techniques. De fait, si le design va trop loin dans ce champ, il risque de se refermer, et de ne plus être intéressant pour nous en tant que citoyens.

Il faudrait alors plutôt créer pour l'humain, avoir un travail créatif « pour la vie » (p.67), et de fait « les logiciels dits "de création" jouent un rôle essentiel, puisque leur appellation même indique un nœud entre les notions de "programmes" (supposant une modélisation) et de "créativité" (qui ne peut avoir lieu sans écarts). » (p.67).

La quatrième partie développe alors cette notion de logiciel de création, et notamment le quasi monopole de la suite Adobe, qui a créé une chaîne de production qui dirige toutes les pratiques des designers. « La chaîne graphique se réduit ainsi progressivement à un seul acteur qui concentre tout le pouvoir dans un schéma pyramidal. » (p.70). En effet, Adobe est devenu une sorte d'usine du créatif, dans laquelle chacun utilise les mêmes outils avec une logique de productivité dans le domaine de la création, mais aussi dans la recherche de clients au travers de leur plateforme Behance. Une dépendance est dès lors créée, et notamment dès l'enseignement, au sein de laquelle la suite Adobe est privilégiée. Ceci est d'autant plus néfaste que l'accès aux logiciels nécessite un abonnement, et les fichiers créés n'utilisant pas des formats ouverts, seuls les outils Adobe peuvent les ouvrir, ce qui conduit à une nécessité pour tous de se soumettre à la « suite créative ».

Finalement, si le design va trop loin dans le champ de l'innovation autant que dans celui de la créativité, il risque de se refermer, et ne plus être intéressant pour nous en tant que citoyens. Au contraire, en tant qu'être humain et designer, il vaudrait mieux concevoir des objets, des systèmes techniques plus ouverts, plus libres, mieux partagés. Et ceci notamment dans des logiques d'open source et de libre accès aux techniques ainsi qu'aux savoirs.

Autre fait intéressant concernant cette fois-ci la vente du livre en lui-même : il n'y a eu aucun problème de vente, malgré une licence libre qui rendait de fait le texte accessible librement. De la même manière, même si sa thèse était sous licence libre, un éditeur est venu vers lui afin de la faire publier, et les stocks ont été écoulés en moins d'un an, provoquant de fait un nouveau tirage. Il semble donc qu'en plus d'être bénéfiques à la pratique, les licences libres n'empêchent pas la vente de produits.

« Les icônes remplacent les "lignes de commande" qui se voient "refoulées" un peu plus en profondeur à chaque itération du "système d'exploitation", limitant d'un côté nos possibilités d'accéder au centre technique des machines, et permettant, d'un autre côté, à de plus en plus de personnes d'accéder au "monde du computationnel". »

p.25

« Le design apparaît comme la nouvelle fondation pour la conceptualisation et la production des connaissances. » (Anne Burdick, Johanna Drucker, Peter Lunenfeld, Todd Presner, Jeffrey Schnapp ; Digital\_Humanities, op. cit., p.117), p.32 du livre.

« Où situer alors le travail du designer ? Travail-t-il pour l'utilisateur ou pour le profit des usages que les firmes privées souhaitent faire de cet espace ? »

p.57

« Celui qui est "au service de" n'est pas libre, il cherche, dit Montaigne, "à s'enrichir". »

p.60

« les logiciels dits "de création" jouent un rôle essentiel, puisque leur appellation même indique un nœud entre les notions de "programmes" (supposant une modélisation) et de "créativité" (qui ne peut avoir lieu sans écarts). »

p.67

« La chaîne graphique se réduit ainsi progressivement à un seul acteur qui concentre tout le pouvoir dans un schéma pyramidal. »

p.70