Design
Arts
Médias

Si Jan Tschichold avait connu les feuilles de style en cascade : plaidoyer pour une mise en page comme programme

Julie Blanc

2022-04-22

Julie Blanc est doctorante en design graphique et ergonomie (EUR ArTeC / Université Paris 8 – EA349 / EnsadLab). Dans sa pratique professionnelle, elle contribue au développement de Paged.js et conçoit différents projets éditoriaux multi-supports avec l'utilisation des technologies du web, notamment le langage CSS. Site web: julie-blanc.fr

Résumé

Depuis une dizaine d'années, des designers graphiques utilisent les technologies du Web pour la composition de publications imprimées. Les feuilles de style en cascade CSS, pensées pour adapter la mise en forme des documents à une multitude de périphériques de sorties (écran comme imprimé), sont au cœur de ces pratiques. Nous tentons de montrer en quoi elles ont à voir avec des notions fondamentales de la mise en page – rôle traditionnel des typographes-maquettistes et designers graphiques – mais les redistribuent dans une série de concepts programmatiques.

Abstract

For a decade, graphic designers used web technologies for the typesetting and the layout of printed publications. Cascading style sheets (CSS), designed to adapt the layout of documents to a multitude of output devices, are the core of these practices. In this publication, we try to show how they are related to fundamental notions of page layout - the traditional role of typographers and graphic designers – and how they redistribute them in a series of programmatic concepts.

Introduction

Le 11 novembre 2021, dans le cadre de l'Open Publishing Fest, une session intitulée « Un voyage web2print » a réunit quatre designers graphiques et artistes – Kiara Jouhanneau, Raphaël Bastide, Julien Bidoret et Amélie Dumont – qui ont présenté des projets utilisant les technologies du web pour la conception d'objets imprimés. Sept jours plus tard, dans le même cadre, les designers Nicolas Taffin et Julien Taquet ont à leur tour discuté de la manière dont ces technologies aident les « *book developers* » à publier leur contenu dans une session nommée « Paged design around the corners ».

Ces sessions offrent un aperçu de ce qui anime aujourd'hui une petite communauté de designers graphiques qui s'emploient à utiliser les technologies du Web et la programmation pour la composition de publications imprimées. Les feuilles de style en cascade, appelées CSS (de l'anglais *Cascading Style Sheets*) sont au cœur de ces pratiques. Elles représentent un langage informatique descriptif permettant de coder la mise en forme de documents structurés sur le web. Pensées pour établir des constantes de mise en forme documentaire basées sur la structure sémantique, on trouve les feuilles de styles dans les traitements de texte et dans les logiciels de PAO (Publication Assistée par Ordinateur). La notion de cascade apparaît quand à elle avec le Web.

Dès son invention, CSS permet d'adapter la mise en forme des documents à une multitude de périphériques de sorties (écran comme imprimé), mais ses possibilités de mise en page pour les sorties imprimées ont longtemps été ignorées. Elles font aujourd'hui un retour en force auprès de communautés open-source de designers graphiques[1], auprès de l'industrie éditoriale[2] et dans des initiatives proposant de nouvelles plateformes de publications [3].

Aujourd'hui en France et notamment dans les écoles d'art et de design, l'utilisation de CSS pour l'impression trouve un écho dans des pratiques graphiques et artistiques pour ses possibilités d'expérimentationd graphiques multimédias. Les présentations de Kiara Jouhanneau et Julien Bidoret illustrent cette idée. Pour son projet de diplôme *Rêve Party*[4], Kiara Jouhanneau a ainsi utilisé les technologies du Web pour mélanger interactions à l'écran, *fonts variables* animées et

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

jeux d'impression. Pour un workshop consacré à Élisée Reclus[5], Julien Bidoret a proposé d'intervenir de manière interactive dans des fanzines ensuite imprimés. Un système de webcam connecté en direct permettait de prendre des photos disposées en fond de page. De manière plus générale, les améliorations récentes de CSS (grilles, boîtes flexibles, dégradés, rotations, transformations et animations, SVG, fontes variables, modes de fusion, etc.) rendent le langage de plus en plus attrayant pour les expérimentations graphiques. Les préoccupations éminemment politiques concernant les dimensions libres et open-source de ces technologies en renforcent l'engouement.

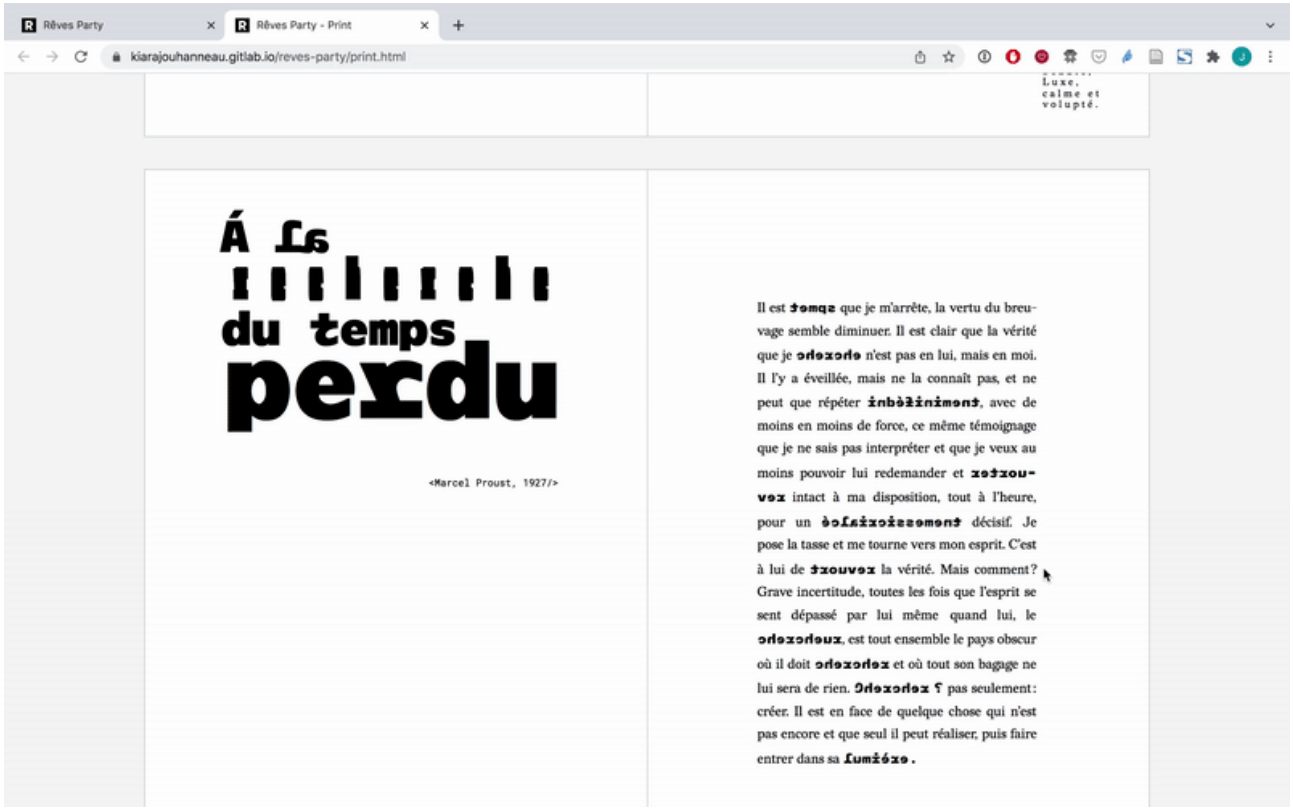


Figure 1. *Rêve Party*, projet de diplôme de Kiara Jouhanneau.

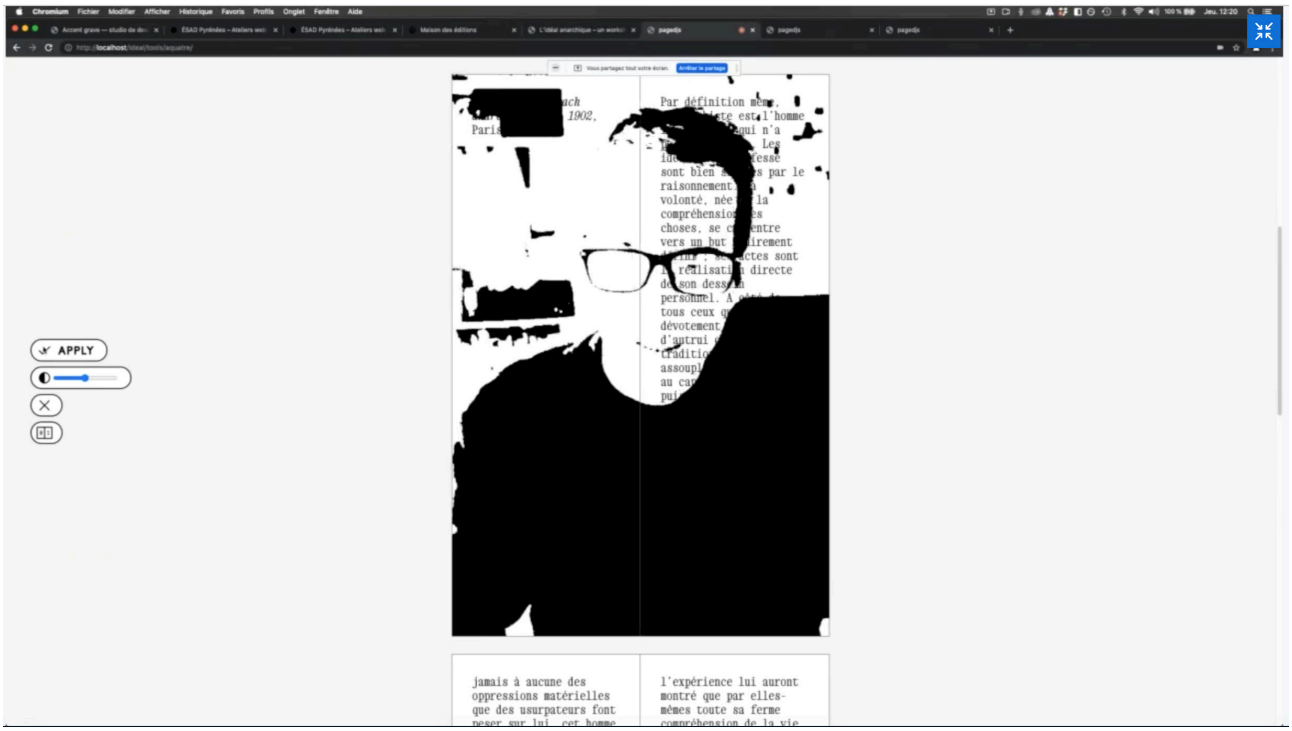


Figure 2. Capture d'écran de la présentation de Julien Bidoret à propos du worshop *L'idéal*

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

Anarchique.

Les présentations de Nicolas Taffin – designer graphique et éditeur – et Raphaël Bastide – artiste multi-facettes – montrent un autre aspect de l'utilisation des technologies du web pour l'impression, davantage tourné vers l'édition structurée[6] et la transformation des aspects collaboratifs de la chaîne éditoriale[7]. Ces technologies permettent en effet de concevoir des chaînes unifiées et délinéarisée grâce aux outils et aux méthodes issus de la programmation et auxquelles les technologies du web sont adaptées[8]. De plus, cette organisation technique permet de proposer des publications reposant sur le principe *single source publishing*[9], une méthode de gestion de contenu qui permet de produire plusieurs sorties (site web, livre numérique, livre imprimé, etc.) à partir d'un même contenu.

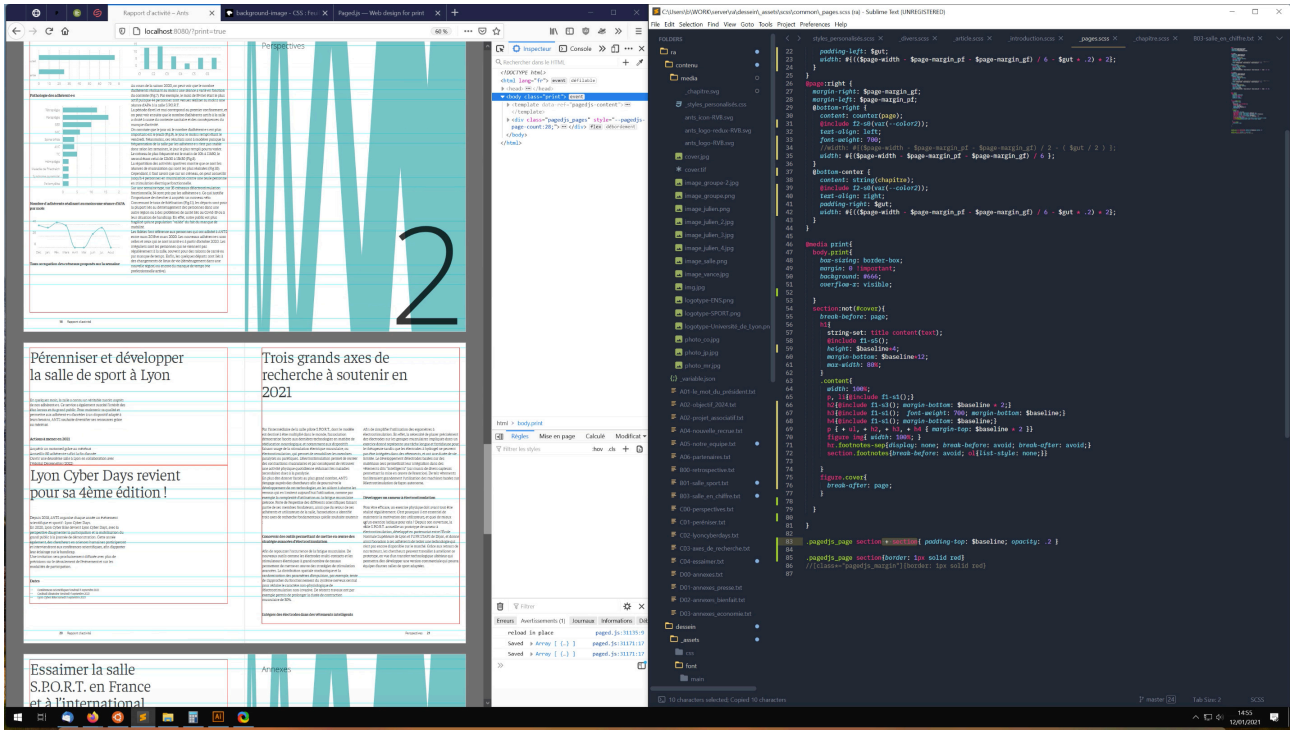


Figure 3. Mise en page affichée dans un navigateur web (à gauche) et son code source CSS (à droite). Projet en cours de Benjamin Gremillon.

Malgré ce que ces présentations donnent à voir, peu de designers graphiques utilisent réellement les technologies du Web pour la mise en page imprimée. Elles représentent un champ très spécialisé qui implique des connaissances complémentaires provenant de milieu souvent cloisonnés tant au niveau des formations qu'au niveau des métiers. Les compétences nécessaires concernent en effet autant l'édition structurée, le design graphique que le développement web. Et même au sein de ces champs, une bonne compréhension de d'HTML et de CSS est souvent rare.

De plus, ces pratiques souffrent d'une image négative de la programmation web et sa supposée incompatibilité avec une activité graphique sensible, précise et en maîtrise. Le vieil adage de la mise en relation du fond est de la forme est souvent présenté comme un des aspects primordiaux du design graphique (l'idée que chaque texte appelle une mise en page qui lui soit propre) et paraît illusoirement incompatible avec le principe technique de séparation du fond et de la forme que propose les technologies du Web.

Un des objectifs de cette publication est de rappeler que CSS est un langage tout à fait adapté à la conception graphique, puisqu'il a été créé pour décrire la présentation de document. Nous défendons même l'idée que son utilisation s'inscrit dans une suite logique de l'histoire de la composition et de la mise en page, si chère au célèbre typographe Jan Tschichold.

La notion de *feuille de style* à l'origine de l'appellation de CSS est définie comme « a set of rules

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

that associate stylistic properties and values with structural elements in a document, thereby expressing how to present the document[10] ». C'est une définition tout à fait compatible avec la production de livres imprimés. Elle nous rappelle qu'avant l'apparition de la PAO, le travail des designers et typographes consistait à fournir à l'imprimeur (composition au plomb) ou l'opérateur (photocomposition) un ensemble de règles stylistiques et de contraintes définissant le gabarit d'un livre et les caractéristiques des blocs typographiques. Coder en CSS, consiste à fournir ces mêmes informations au navigateur web.

En contextualisant les origines de la création du Web, nous expliquerons comment HTML et CSS ont été imaginés pour décrire, structurer, lier, rendre lisible et styliser les documents, servant ainsi tout à fait le rôle des designers graphiques, des maquettistes et des typographes. Puis, à travers trois exposés qui sont autant d'exemples, nous décrirons comment les technologies du Web traduisent des notions fondamentales de mise en page et de composition – structuration typographique, relations entre les éléments, grilles, gabarits, etc. En creux, nous montrerons comment certaines de ces notions sont rejouée par des logiques de programmation et s'ouvrent à de nouveaux possibles esthétiques.

1. Un Web de documents entre structure et présentation

La composition de livres imprimés est une activité vieille de plusieurs siècles avec un ensemble de règles et de traditions transmises par des générations de typographes, compositeurs et imprimeurs. De nouveaux possibles apparaissent avec chaque évolution technique, que cela soit dans les ateliers de composition mécanique (Monotype et Linotype) ou avec la lumière (photocomposition), puis avec la micro-informatique et ses logiciels de PAO. Puis, l'apparition du Web et de la demande accrue de publications multimédias font entrer dans la composition et la typographique des principes de flexibilité et fluidité des éléments pour les adapter à des pages sans dimensions fixes. Le développement web explose à la fin des années 2000 et devient peu à peu une spécialité cloisonnée de la composition des objets imprimée. Pour autant, si nous regardons attentivement l'histoire de l'apparition du Web, ce cloisonnement peut être remis en question.

En effet, si aujourd'hui, le Web semble plus à voir avec des sites marchands, des applications closes et des réseaux sociaux, nous avons tendance à oublier qu'il a principalement été créé pour la publication de documents. En 1991, lorsque Tim Berners-Lee, informaticien britannique au CERN, publie le premier site web présentant les caractéristiques de son invention, il déclare: « The WorldWideWeb (W3) is a wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents [11] ». Le scientifique tente ainsi de construire une toile de documents connectés, en particulier d'articles scientifiques, qu'il souhaite rendre accessibles aux scientifiques du monde entier gratuitement [12]. Ajoutons qu'en 1993, le Web a été placé dans le domaine public, signalant la volonté d'ouverture et d'universalisme du projet. [13]

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

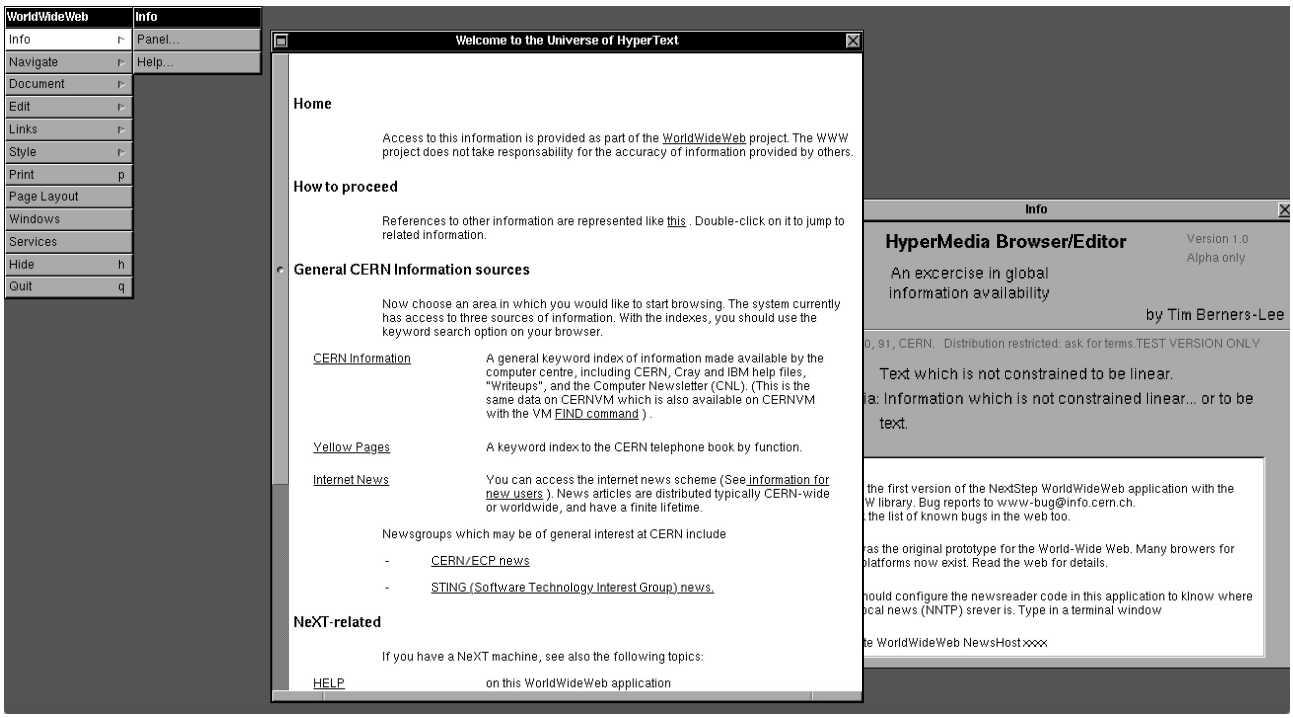


Figure 4. Premier site web publié le 6 août 1991, affiché dans le navigateur web original de NeXT (reconstruit en 2009 par le CERN: https://worldwideweb.cern.ch/browser/). Crédits: CERN.

Dans cette idée, le Web requiert un langage simple, lisible et accessible sur toute plateforme. C'est ainsi que Tim Berners-Lee, secondé par Robert Cailliau, ingénieur et informaticien belge, propose l'*HyperText Markup Language* (HTML), un langage de balisage permettant de représenter la structure d'un document web à l'aide de balises ajoutées entre des phrases ou des mots pour indiquer le rôle du texte. Afin de l'afficher sur n'importe quel terminal quelle que soit sa capacité graphique d'affichage[14], HTML est volontairement un langage très simple, et, surtout, sans aucune indication de mise en forme ou possibilité de contrôle de sa présentation, excluant toute modification des polices de caractères, des couleurs ou de la taille du texte[15].

Très vite, les auteurs des sites web, ainsi que les utilisateurs demanderont de plus en plus un moyen d'avoir la main sur ce rendu. Des efforts sont menés aussitôt pour proposer des concepts de feuilles de styles pouvant s'appliquer à cet HTML et répondre à des besoins de présentation. En 1994, Håkon Wium Lie, informaticien norvégien, rejoint par Bert Bos, informaticien néerlandais, formule une proposition de *Cascading HTML Style Sheets*, vite abrégé en CSS. La proposition repose sur l'ambition de séparer complètement la structure HTML de son rendu visuel grâce à des principes de sélecteurs des éléments HTML spécifiés en CSS. Le principe de cascade, c'est-à-dire la possibilité pour le style d'un document d'être hérité à partir de plusieurs « feuilles de style » séduit la communauté du web qui cherchaient un moyen d'arbitrer entre plusieurs sources de mise en forme des éléments (issues des préférences stylistiques des auteurs, des navigateurs et des utilisateurs). Il faudra cependant attendre le début des années 2000 pour que CSS commence à être correctement implémenté dans les navigateurs web. [16]

La création de CSS repose donc sur l'ambition de séparer le contenu et la structure sémantique d'un document de son rendu visuel. Ce principe de séparation du fond et de la forme est un principe informatique connu mais le Web est le seul environnement à avoir poussé cette idée aussi loin et à si grande échelle.[17] L'affichage doit être proposé pour une multitude de médias afin que n'importe qui puisse y avoir accès. [18] Dès sa première proposition, envoyée le 10 octobre 1994 sur la liste de diffusion www-talk@cern.ch, Håkon Wium Lie indique que CSS est, lui aussi, pensé pour tous les médias, y incluant le papier: « Current browsers consider the computer screen to be the primary presentation target, but [CSS] has the potential of supporting many output media, e.g. *paper*, speech and braille.[19]»

Les requêtes médias, proposée dès 2002 [20], consolideront cette idée et la rendront techniquement

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

possible. Elles consistent en des expressions CSS permettant de définir si les styles s'appliquent à tel ou tel type de média. Par exemple, tous les styles contenus dans la requête *@media print { ... }* ne s'appliqueront qu'à la sortie imprimée de la page web. (Tous les navigateurs possèdent aujourd'hui des fonctionnalités d'impression, souvent disponible dans le menu *Fichier* > *Imprimer*).

CSS est encore aujourd'hui le seul langage entièrement dédié à la présentation visuelle de documents et au rendu graphique des pages web. C'est une invention unique dans le domaine informatique pour répondre à un enjeu jusqu'alors jamais imaginée dans l'histoire des médias: pouvoir, techniquement, exprimer à quel type de périphérique et quelle taille d'écran une règle stylistique particulière doit s'appliquer. Il est de ce fait, un langage profondément graphique, parfaitement adapté à la mise en page de tous types documents, en y incluant la composition imprimée.

Pendant longtemps, les contraintes techniques lié aux technologies du Web ne permettaient pas de produire des mises en pages élaborées. Il n'était donc pas intéressant pour les designers graphiques de pratiquer ces technologies. Les choses se sont vites améliorées, les navigateurs web implémentant peu à peu des spécifications CSS permettant des rendus typographiques et graphiques de plus en plus poussés. Ces six dernières années ont notamment été marquées par des évolutions fulgurantes, depuis l'implémentation des grilles et des boîtes flexibles en CSS jusqu'au développement d'un nouveau format de polices de caractère permettant de rendre les fontes variables – le format OpenType variable fonts (OTVF).

Aujourd'hui, il est possible de concevoir des mises en page uniques (à l'écran comme pour l'imprimé) dans lesquelles la relation spatiale entre les éléments typographiques et les images est essentielle pour des raisons esthétiques et pour la compréhension du texte. Des fonctionnalités nécessaires à l'impression (sauts de pages, numéros de pages, titres courants, gabarits de pages, notes en bas de page, marges en miroir, etc.) peuvent aussi être utilisées directement en CSS[21] . En complément, l'utilisation de scripts permet de contourner les limitations des d'algorithmes de césures des navigateurs ou d'améliorer la prise en charge de certaines spécificités de mises en pages complexes (notes en marges, image pleine page, etc.). La composition pour l'imprimé avec les technologies du Web ne possède ainsi plus beaucoup de limites.

Nous allons à présent nous attarder sur la manière dont les technologies du Web traduisent des notions fondamentales de mise en page et de composition dans leur fonctionnement. Nous avons choisi d'explorer cette question à travers trois exposés liés à la structure du code, l'agencement des éléments par des grilles et des exemples d'hybridations entre logique du Web (flux, ancrage et flexibilité) et logique de l'imprimé.

2. Premier exposé: Structurer le livre, structurer le code

Si l'on considère un livre avec l'œil d'un auteur, d'un lecteur ou du designer graphique qui le compose, une édition est toujours structurée. Les éditions sont organisés en entités telles que les sections, les chapitres, les titres, les paragraphes, les notes, les citations, etc. Cette structure intellectuelle, logique, permet au lecteur de se repérer dans l'organisation du livre. Elle se reflète dans la structure visuelle de l'édition: les chapitres débutent sur une nouvelle page pour marquer une rupture forte, un titre sera mis en avant par une stylisation différente que le texte courant, les paragraphes sont indiqués par un retour à la ligne et un alinéa, etc.

Dans les logiciels actuellement les plus utilisés dans l'édition, InDesign pour la mise en page et Microsoft Word et Google Docs pour le traitement de texte, cette division entre structure logique et structure visuelle se retrouve la plupart du temps confondue. Les panneaux et galeries de « style » proposées permettent d'appliquer aux éléments des mises en forme prédéfinies par l'utilisateur qui peuvent ensuite être modifiées sur l'ensemble des éléments de même type. Un indice de ce manque d'approche de structuration logique est ainsi l'impossibilité de baliser des ensembles d'éléments afin de les regrouper en chapitres, en sections ou en encarts. Ces regroupements ne

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

peuvent s'exprimer que visuellement – par des sauts de pages, des encarts visuels, des espacements dans la page, etc.

Les logiciels de traitement de texte et de PAO, en privilégiant la composition visuelle de la mise en forme des textes, marquent donc pour les designers graphiques une perte technique dans la structuration logique des mises en page[22]. À l'inverse, l'utilisation des technologies du web impose ce type de structuration puisque HTML propose un balisage sémantique des documents. Cette structuration est de plus beaucoup plus précise grâce aux principes d'arborescence et d'attributs complémentaires.

La caractéristique la plus importante des documents HTML, se situe en effet dans leur fonctionnement en arborescence. HTML définit la structure de la page web afin que le navigateur puisse créer un DOM (*Document Object Model*), une représentation de la page web et de ses éléments sous forme d'arbre où chaque embranchement est appelé « nœud ». Ainsi, les éléments peuvent être imbriqués les uns dans les autres et font références les uns aux autres en tant que parents ou enfants, ou éléments frères – tout comme dans un arbre arborescent[23].

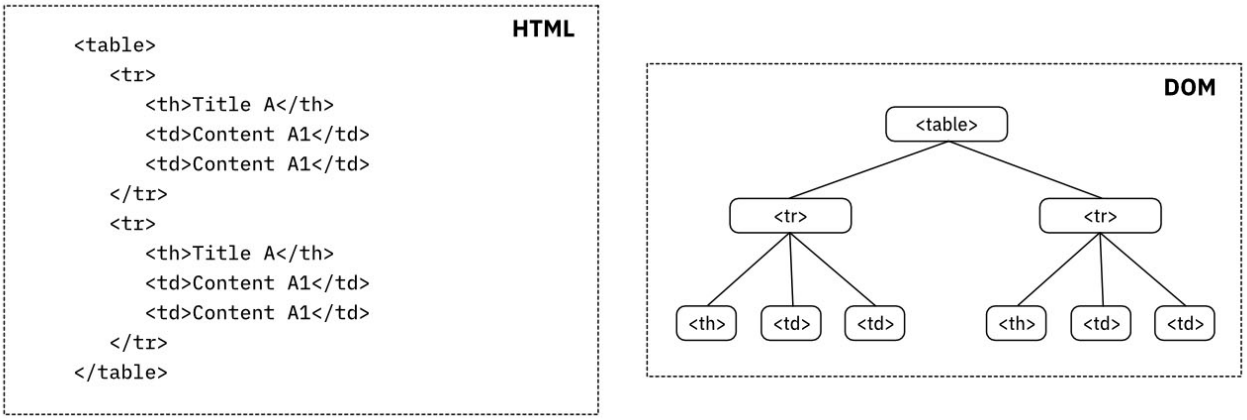


Figure 5. Code HTML d'un élément tableau et sa représentation sous forme de DOM.

Les balises actuellement disponibles en HTML sont suffisantes pour couvrir les principaux éléments utilisés en édition, comme le montre la structuration HTML de cet article-même. Les possibilités d'ajouts d'identifiants uniques ou de classes spécifiques pour définir des ensembles d'éléments permettent une structuration plus fine. Ainsi, le bout de code ci-dessous rend évidente la structuration d'un texte et attribue à chaque section des informations complémentaires:

```
<section id="introduction">
...
</section>
<section class="chapitre" id="chapitre-1">
...
</section>
<section class="chapitre" id="chapitre-2">
...
</section>
```

Les styles CSS sont associés aux éléments HTML grâce à des sélecteurs ce qui permet une approche très fine de l'organisation du document. Ces sélecteurs peuvent être simple – *.chapitre* sélectionne tous les éléments avec une classe « chapitre » – ou se combiner – *h1 + p* sélectionnera le premier paragraphe qui suit immédiatement un titre de premier niveau. Certaines combinaisons de sélecteurs CSS sont aussi très complexes afin de cibler des éléments de manière contextuelle sur la base de leurs ancêtre et leur place dans le DOM. La déclaration *.chapitre + aside > h2:nth-of-type(2)* permet par exemple de sélection le deuxième titre (*h2:nth-of-*

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

type(2) d'un élément d'aparté (aside) qui suit immédiatement un élément défini comme un chapitre (.chapitre).

Les mécanismes de propagation des valeurs de CSS – comme la cascade, l'héritage ou les valeurs initiales – permettent d'agir sur plusieurs éléments en même temps. Une même propriété peut se répercuter sur plusieurs éléments et inversement, un même élément peut hériter de plusieurs propriétés stylistiques. C'est l'organisation du code CSS et le choix des sélecteurs qui indique au navigateur quelles styles doivent s'appliquer, sans avoir besoin de spécifier explicitement les valeurs de toutes les propriétés pour chacun des éléments.

Le principal avantage est que les feuilles de styles peuvent être courtes et peu verbeuses. Il est facilement possible de monter en généralité par l'interaction des déclarations et des sélecteurs entre eux. De nombreuses informations stylistiques peuvent être déduites du contexte de l'arborescence et s'additionner entre elles. Un même type d'élément, par exemple un titre, pourra ainsi être stylisé de manière complètement différente en fonction de sa position dans la structure du livre (est-ce un titre de chapitre ou le titre de l'introduction?) tout en gardant certaines propriétés communes qui n'auront été déclarées qu'une seule fois pour tous les éléments de même type.

Ces mécanismes de feuilles de style ne s'appliquent pas seulement aux éléments textuels mais aussi aux éléments graphiques et iconographiques, et même à la structure globale du livre[24]. Cela astreint les designers graphique à penser la forme du livre comme un ensemble de règles contextuelles et hiérarchisées qui interagissent les unes des autres en s'appuyant sur la structure logique de l'édition. Ainsi, la construction d'une feuille de style engage l'idée d'une construction de relations significatives entre les éléments de la structure du texte et un aller-retour entre plusieurs granularités – règles s'appliquant au niveau du livre, au niveau d'un chapitres ou sur plusieurs éléments. Les exceptions restent toujours possibles grâce aux mécanismes d'identification unique qui permettent de les spécifier. Notons que cette structuration du livre sous forme de règles graphiques et typographiques est le travail du designer graphique de manière générale[25] mais cette idée est ici encodée dans le fonctionnement du langage CSS.

Cet exposé nous montre que le principe de séparation du fond et de la forme à travers HTML et CSS implique pour les designers graphiques une préoccupation importante de la hiérarchie des documents et leur compréhension sémantique. À travers les lignes de code, c'est une systématisation entre structure intellectuelle, structure techniques et structure visuelle qui est proposée. Les mécanismes des les feuilles de style CSS offre par ailleurs des logiques beaucoup plus puissantes que leurs équivalences dans des logiciels plus classiques de traitement de textes et de PAO.

3. Deuxième exposé: agencer les relations entre éléments

Une partie de l'activité de composition des designers graphiques consiste à agencer les éléments du livre dans l'espace de la page. Selon Jan Tschichold, une mise en page parfaite « repose sur une parfaite harmonie de toutes les parties » et cette harmonie « dépend du choix des bons rapports ou proportions » entre tous les éléments du livre (marges, bloc d'empagement, interlignages, interlettrages, etc.) [26]. Pour déterminer ces rapports et ces proportions, les designers graphiques ont conçu un ensemble de dispositifs structurants reposant sur différents repères visuels et un ensemble de règles destinés à guider et faciliter la création.

Parmi ces dispositifs à leur disposition, les deux plus connus sont la grille modulaire et la grille de ligne de base[27]. Elles permettent de faciliter l'alignement et la répartition du contenu et d'obtenir des espacements cohérents pour une meilleure cohésion de la composition globale malgré la diversité du contenu et des forces de corps typographiques différentes.

Ces grilles sont souvent conçues à l'aide de tracés géométriques et de calculs mathématiques calculés en amont puis traduits sous forme de valeurs fixes et de repères visuels dans les logiciels

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

de PAO. Des fonctionnalités de magnétisme permettent de faciliter le placement des éléments sur ces repères, mais globalement, un changement dans la grille implique à minima quelques corrections manuelles et parfois un long travail de remise en page dans le logiciel.

Les grilles peuvent être travaillées de façon très différentes avec les technologies du Web et tirer parti des avantages de la programmation. Pour illustrer cette idée, nous prendrons en exemple deux extraits de code produits pour un catalogue du Musée Saint-Raymond de Toulouse[28].

3.1. Aligner les éléments sur la ligne de base

Le premier extrait concerne l'alignement des éléments textuels selon l'idée d'une ligne de base imaginaire qui aurait une hauteur de 14 pixels:

```
body {
    --baseline: 14px;
    font-size: 12px;
}
h1 {
    font-size: 2.2em;
    line-height: calc(var(--baseline)*3);
    margin-bottom: calc(var(--baseline)*8);
}
p {
    font-size: 1em;
    line-height: var(--baseline);
}
figure {
    height: calc(var(--baseline)*12);
}
```

Ce code utilise les possibles des fonctions de calculs et de variables en CSS. Les variables CSS sont des propriétés personnalisées permettant de définir des valeurs utilisables et répétable à travers le document. Le premier bloc de l'extrait indique que la variable nommée *baseline* a pour valeur 14 pixels. Le reste du code utilise cette variable dans des fonctions *calc()*, qui permettent de réaliser des opérations mathématiques en CSS (addition, soustraction, division et multiplication) pour calculer la valeur d'une propriété de type numérique.

L'association de ces deux fonctions permet d'insérer une logique mathématique dans la définition des espaces de la mise en page afin de garder les alignements des éléments textuels. Ainsi, dans le deuxième bloc qui s'applique à tous les titre de premier niveau (*h1*), la deuxième ligne indique que l'interlignage (*line-height*) a pour valeur trois fois la hauteur de la ligne de base. Notons que ce système de calcul et de variable a aussi été utilisé pour définir la hauteur des figures du texte, afin qu'elles restent alignées sur la ligne de base comme nous le montre le dernier bloc.

3.2 Construire une grille modulaire

Le deuxième extrait (simplifié pour les besoins de la démonstration) montre comment a été encodé une grille modulaire utilisée pour les pages des cahiers d'images du catalogue :

```
.grid-img {
    --columns-nbr: 12;
    --rows-nbr: 10;
    --height-rows: var(--baseline)*5;
    width: 100%;
```

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

```
    height: calc(var(--height-rows)*var(--rows-nbr) + var(--
baseline)*(var(--rows-nbr)-1));
    display: grid;
    grid-template-columns: repeat(var(--columns-nbr), 1fr);
    grid-template-rows: repeat(var(--rows-nbr), 1fr);
    grid-gap: var(--baseline);
}
#grid-3 img {
    grid-column: 7/end;
    grid-row: 1/6;
}
```

Nous voyons à nouveau qu'un ensemble de variables et de calculs ont été utilisés pour la construction de la grille. Certaines de ces variables servent à stocker le nombre de colonnes et de rangées de la grille (respectivement, *--columns-nbr:12* et *--rows-nbr: 10*) ainsi que la hauteur d'une rangée de la grille qui correspond à un multiple de la ligne de base (*--height-rows: var(--baseline)*5*). Ces variables ont été réutilisées dans la suite du code. Ainsi, la hauteur de la grille (*height: calc(...*) additionne et multiplie différentes variables qui correspondent au nombre de rangées de la grille, à la hauteur de ces rangées (elle-même dépendante de la ligne de base) et à la hauteur des gouttières (correspondant à la hauteur d'une ligne de base). Le reste du code sont des propriétés spécifiques aux grilles CSS permettant de définir le nombre de colonnes (*grid-template-columns*), le nombre de rangées (*grid-template-rows*), la hauteur et la largeurs des gouttières (*grid-gap*) et le placement de l'image dans la grille (le dernière bloc de code).

Ce code fait ainsi correspondre les tracés de la ligne de base aux tracés de la grille modulaire pour que les images soient parfaitement alignées sur les deux types de grilles. Cela est visible sur la figure suivante où ont été affichés les tracés régulateurs des deux grilles à l'aide de l'outil d'inspection du navigateur.

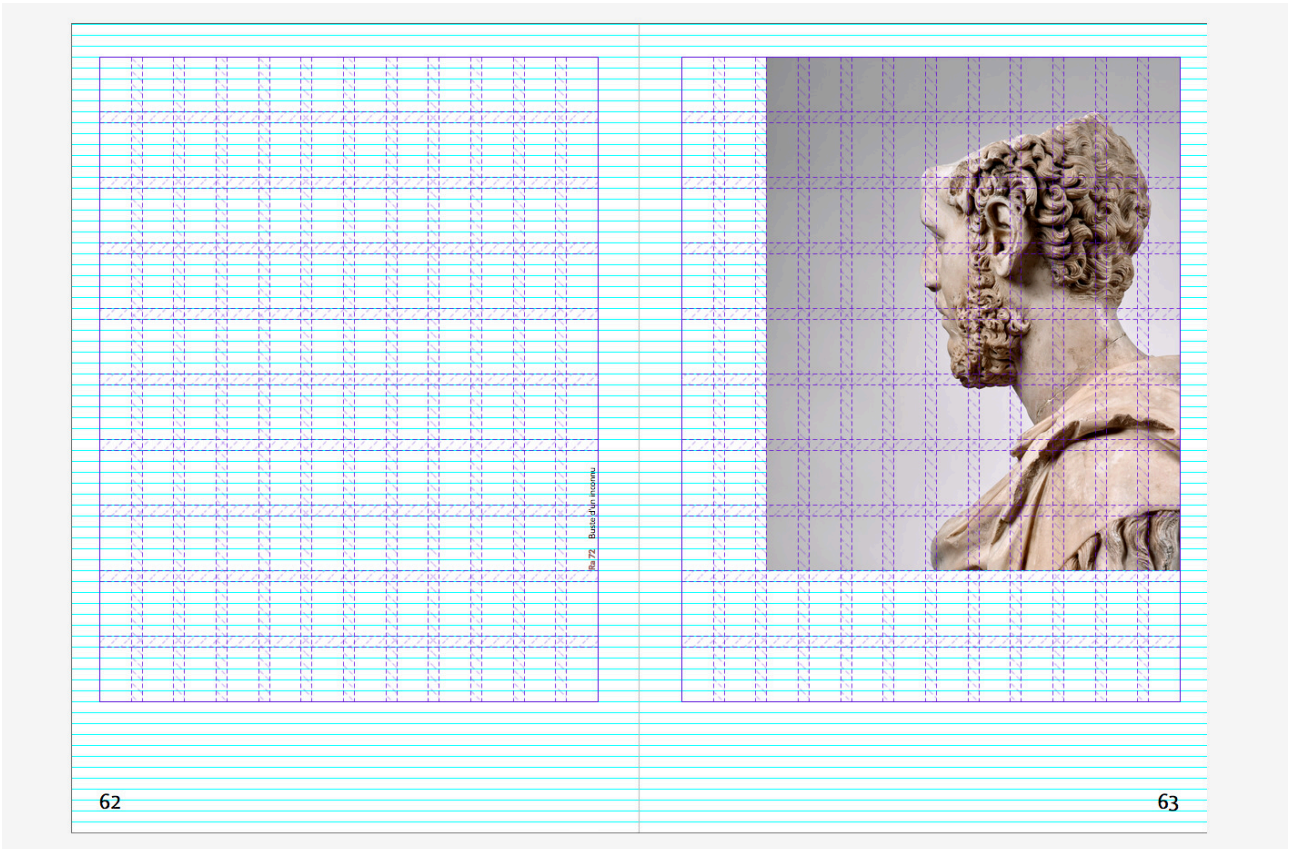


Figure 6. Tracés régulateurs de la ligne de base et de la ligne modulaire sur une double page du

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

catalogue, affichés à l'aide des outils d'inspection du navigateur web.

Dans ce code, les différents éléments de la mise en page – les interlignages, la taille et les marges des éléments textuels, la hauteur des figure ou encore les grilles – sont dépendants d'une seule valeur (ici, la hauteur de la ligne de base) et relatifs les uns aux autres. Ces relations sont encodées en CSS grâce au jeu des variables et des calculs (et grâce aux unités relatives pour la taille des polices de caractères). La force de ce système est qu'il est possible, à tout moment, de changer la ligne de base du document en changeant la variable *baseline*. Tous les éléments de la mise en page se seraient alors adaptés automatiquement en gardant les rapports de proportions encodés et en restant alignés sur les différentes grilles[29].

Cet exposé nous montre que les concepts programmatiques de variables et calculs s'allient au principe graphique des grilles (modulaire et de ligne de base) en systématisant certaines de leurs notions par un encodage technique fort. Ce code reste toutefois visible et lisible pour les designers graphiques, notamment parce que CSS est un langage descriptif.

4. Troisième exposé: hybrider logique du Web et logique de l'imprimé (flux, ancrage et flexibilité)

L'arrivée du Web a complètement transformé les manières de penser la mise en forme. La diversité des possibilités d'affichage des sites web – et particulièrement la variabilité des tailles d'écrans et des fenêtres de navigation – implique l'impossibilité de penser la mise en page selon un format fixe. Il en résulte que les documents web répondent à un certain nombre de caractéristiques – tels que le flux, l'ancrage ou l'adaptation dynamique – sur lesquels les designers graphiques s'appuient pour mettre en forme des sites web.

Les propriétés CSS offrent un panel de mécanismes permettant d'agir avec ces caractéristiques pour répondre aux besoins de mises en page variables (*responsive*): grilles, boîtes flexibles, flotteurs, positionnements relatifs et absolus, etc. Ces techniques, souvent réunies sous l'appellation de *css layout*, permettent de changer le positionnement des éléments les uns par rapport aux autres sans toucher à la structure HTML et en conservant leur place dans l'arborescence.

Bien que la conception d'un livre imprimé impose un retour à une logique de format fixe – celui des pages imprimées – l'utilisation des technologies du Web implique que le document HTML et les feuilles de styles CSS gardent leurs caractéristiques. Nous montrons dans ce troisième exposé, deux mises en page de livres imprimés qui mettent à profit les particularité du flux et les avantages de propriétés CSS censées répondre à des mises en pages flexibles de l'écran .

4.1. Flux et ancrage des éléments

Par défaut, le code HTML d'une page web est interprétée de manière séquentielle par les navigateurs qui affichent les différentes éléments balisés dans l'ordre où ils apparaissent dans la structure du code. Les éléments sont disposés par défaut les uns en dessous des autres et s'étirent horizontalement selon la place disponible. Cette affichage séquentiel se nomme le flux. Mettre en page un livre imprimé consiste à fragmenter ce flux qui sera distribué dans des pages au format fixe[30].

Toutefois, les éléments qui se suivent logiquement dans le HTML peuvent aussi avoir besoin d'être déplacés sur la page indépendamment de la logique du flux afin de rendre la mise en page plus équilibrée, plus lisible ou améliorer la compréhension du discours. Par exemple, des publications imprimées à la mise en page riche nécessitent un déplacement des éléments en haut ou en bas de la page, des images en pleine page, des éléments mis en exergues ou encore des notes de page.

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

Pour répondre à ces exigence de présentation, les propriétés de *css layout* conviennent tout à fait et certaines ont même étaient imaginées spécialement pour la mise en page imprimée[31]. Mais leur utilisation implique que les designers graphiques comprennent cette logique de flux et l'adapte à la page. Un avantage important est que, même si les éléments sont déplacés visuellement et « retirés » du flux grâce aux propriétés CSS, ils restent liés à leur emplacement d'origine dans le fichier HTML. Si on veut utiliser un vocabulaire plus proche de celui des designers graphiques habitués aux logiciels de PAO, nous pourrions dire que tout élément est « ancré » dans le flux – c'est-à-dire qu'un élément reste toujours lié aux éléments qui le précèdent et le suivent. L'ancrage au flux peut parfois être vu comme une contrainte mais il peut aussi être utilisé pour ses avantages.

Nous en trouvons un exemple dans la mise en page du livre *Controverses mode d'emploi*[32] conçu avec HTML et CSS par Sarah Garcin. Les notes du texte sont disposées dans l'espace disponible des marges extérieures du livre, tout en restant à hauteur de l'appel de note correspondant. Pour arriver à ce résultat, Sarah a utilisé des propriétés de positionnements CSS fixes et relatives en appliquant quelques lignes de code à l'ensemble des notes. Notons, qu'elle a dû coupler son CSS à un script calculant la hauteur des notes et les repositionnant, permettant d'éviter que les notes se chevauchent ou dépassent de la page[33]. Ce mécanisme de mise en page est beaucoup plus laborieux sur un logiciel de PAO classique, où les designers graphiques sont plus habitués à disposer ce type d'élément « à la main ». Pour le travail de Sarah Garcin, les notes étant liées à leur emplacement d'origine dans le document HTML, elles restaient correctement positionnées dans la page à hauteur de l'appel de note malgré tout changement dans la mise en page.



Figure 7. Double page du livre *Controverses mode d'emploi* présentant de notes en marge. Crédit: Sarah Garcin

4.2. Flexibilité, laisser de la marge à la machine

Afin de répondre aux caractéristiques du design de site web dans des navigateurs aux dimensions variables, une grande partie des propriétés CSS de mises en page permettent de définir le comportement d'éléments les uns par rapport aux autres dans un espace donné sans en définir leurs dimensions ou leur positionnement de manière fixe.

Pour cela, les propriétés de boîte flexible – « *flexbox* » – sont particulièrement intéressantes. Elles dictent la distribution d'éléments dans un espace donné selon des règles d'alignements et de

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

proportions. Cette méthode permet de rendre les éléments fluides car ils s'adaptent à l'espace disponible selon une direction choisie: ils s'étirent pour remplir de l'espace supplémentaire et se rétractent pour s'insérer dans des espaces plus petits[34]. Leur taille peut aussi être variable selon le contenu: par exemple, plus un titre a de mot, plus il prendra de l'espace. Particulièrement efficace pour le design de sites web s'adaptant à la taille de l'écran, ces méthodes de boîtes flexibles peuvent être tout aussi pertinentes pour le design de livres imprimés.

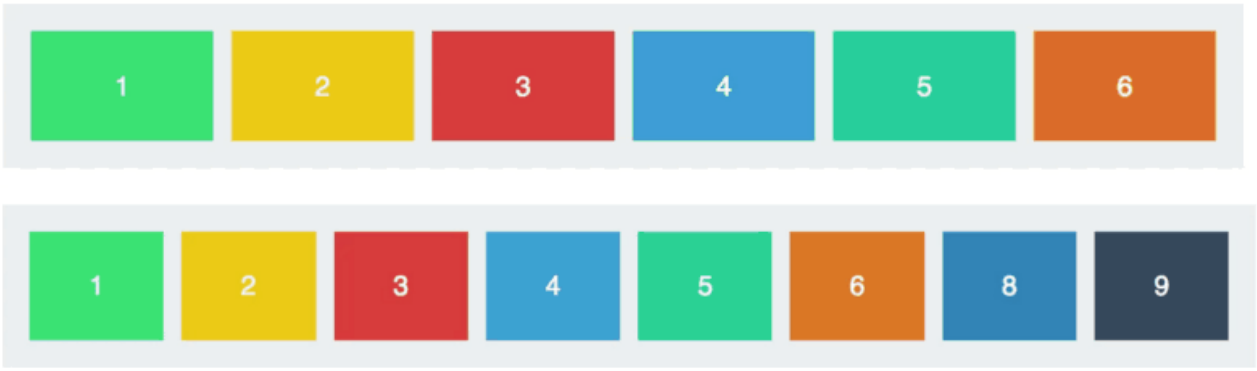


Figure 8. Exemples de comportement des éléments avec les propriétés *flexbox*. Crédit: freecodecamp.org

Nous en retrouvons un usage particulièrement créatif dans le livre *CityFab2.docs* mis en page par Amélie Dumont. Les pages de titres des tutoriels du livre sont constituées d'une série de cinq éléments constants: un titre, une image d'illustration, le logo du logiciel utilisé, un paragraphe introductif et une numérotation. Certaines propriétés des éléments sont définies de manière fixe (c'est par exemple le cas de la largeur des éléments du bas de la page) mais pour le reste, Amélie a utilisé les propriétés des boîtes flexibles pour « laisser le contenu prendre la place dont il avait besoin et donc obtenir des pages qui avaient une grille à peu près similaire mais en même temps qui n'étaient jamais tout à fait la même[35] ».

L'utilisation de filets autour des éléments souligne l'idée des boîtes et laisse trace dans la page de la manière dont la composition a été programmée. Les figures ci-dessus montre la mise en page finale de différentes pages de titres. Nous voyons bien que la composition est différente à chaque page – notamment la dimension des espaces – pourtant c'est la même syntaxe CSS qu'Amélie a appliquée à toutes les pages. En choisissant des propriétés flexibles, Amélie laisse « un petit peu de marge à la machine [...] et à la manière dont elle est programmée, pour finir la mise en page[36] ».

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

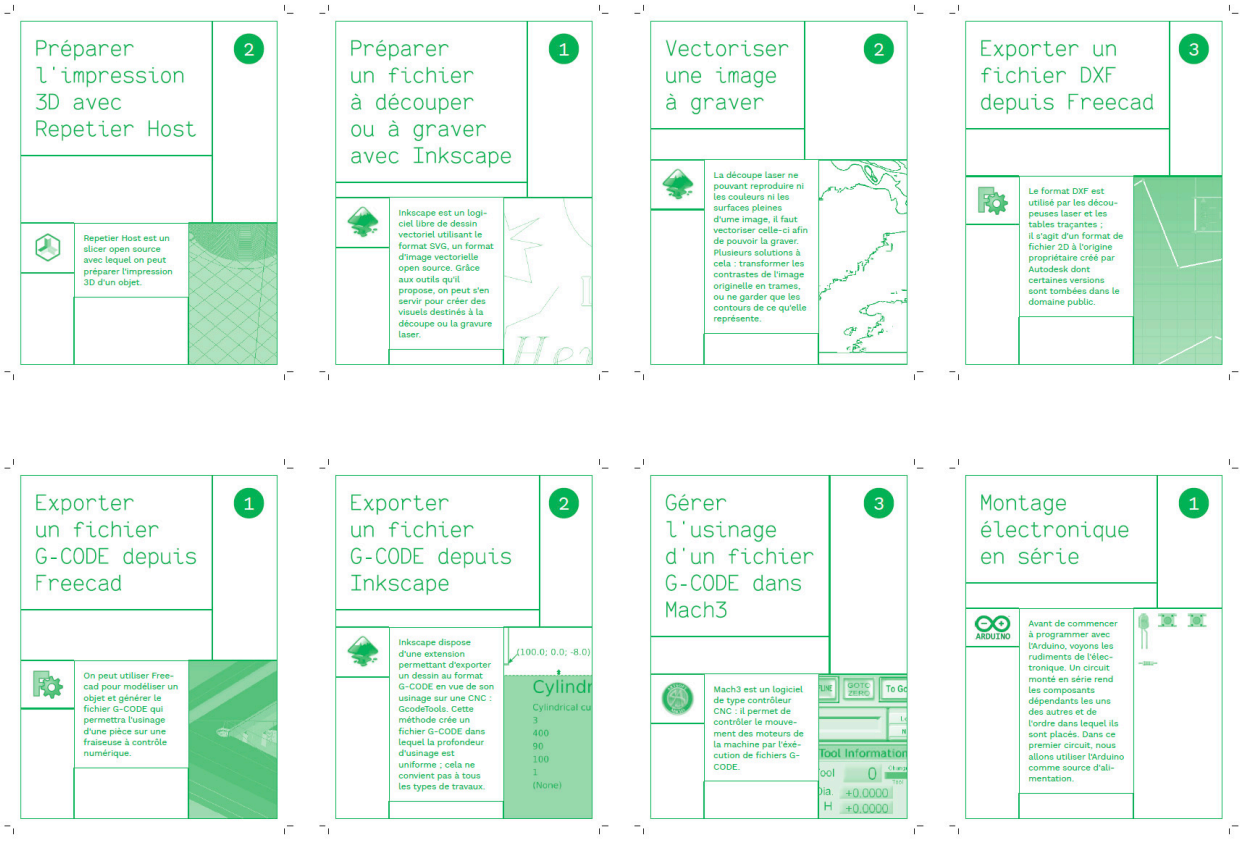


Figure 9. Pages de titres du livre *CityFab2.docs* mis en page par Amélie Dumont.



Figure 10. Scan d'une double page du livre *CityFab2.docs*. Crédit: Amélie Dumont.

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

Cet exposé et ses deux exemples nous montre donc un certain jeu entre des logiques de mise en page propres à l'affichage sur des écrans aux dimensions indéterminées et des logiques de mise en page pour un format fixe tel que celui des pages imprimées. Cette hybridation n'est possible que parce que CSS offre les mécanismes nécessaires aux deux médias dans une continuité technique assez unique.

5. Conclusion: la mise en page comme programme, penser avec le code

Dans son *Manuel de typographie et de mise en page*, François Richaudeau définit la typographique, entendue au sens large de la composition, comme des « règles de construction des systèmes [de pages] [37] ». En nous invitant à penser la mise en page de manière systémique à l'aide d'une série d'instructions et de règles dépendantes les unes aux autres, les mécanismes des feuilles de style CSS permettent de faciliter certaines décisions de mise en page rationalisables et paramétrables se prêtant particulièrement bien à la programmation (grille, hiérarchie typographique relative, ancrage des notes en marges, etc.).

Faisant cela, les technologies du Web rejouent techniquement des notions fondamentales de mise en page déjà présentes dans l'histoire du design graphique. Toutefois, elles redistribuent certaines de ces notions dans une série de concepts programmatiques basés sur des logiques d'arborescence, d'hérédité, de variables, de calculs mathématiques, de flux, d'ancrage. En cela elles montrent aussi certaines ruptures et autorisent un renouvellement des pratiques et des logiques créatives des designers graphique.

Notre troisième exposé met ainsi en évidence que les feuilles de style peuvent être utilisées de façon créative dans les mises en pages imprimées en montrant quelque chose de ce qui fait la caractéristique de CSS: un langage créé pour répondre à la variabilité des supports et des dimensions d'affichage d'un document. De la même manière, l'imbrication des éléments dans une arborescence contextuelle invite à penser des jeux graphiques pouvant potentiellement s'appuyer sur cette arborescence ou la refléter.

L'utilisation de scripts augmente encore les possibles; notamment parce que la mise en page pour l'imprimé et pour l'écran depuis un environnement unifié invite à des hybridations avec des processus et des technologies utilisées par ailleurs dans les mouvements artistiques de *creative coding* de manière plus générale (appel à des bases de données, utilisation d'API, dessins vectoriels génératifs, etc.)[38]

Ainsi, l'utilisation du code invite à concevoir des mises en page qui n'auraient pas pu être pensées si elle n'avait pas directement travaillées dans les navigateurs web et avec HTML et CSS, embarquant avec elles quelque chose qui touche aux caractéristiques de ces langages. Les technologies du Web invitent donc les designers graphiques à penser le code comme un appareil réflexif intégré au processus de création.

Il ne s'agit pas pour autant de réduire le design graphique aux seuls outils qui permettent de le produire. Ni de déléguer à un programme le savoir-faire des typographes-maquettistes et designers graphiques, en réduisant la mise en page à une approche objective et une méthode rationalisante.[39] Ainsi, une systématisation de certains principes de composition et de mise en page à travers le code ne signifie pas une optimisation des méthodes et principes de conception. Les procédures « automatisées » dans l'édition ne remplaceront jamais une bonne maquette, dédiée à la lecture, faite par un ou une designer graphique compétente et guidée par le jugement humain.

L'art de la composition et de la mise en page appartient aux designers graphiques et typographes, ils en connaissent les règles. Les technologies du Web leur permettent d'exprimer cet art en dialogue avec le code en les confrontant à certaines logiques de description, de systématisation et de programmation.

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

Toutefois, les décisions de mise en page restent de l'ordre d'une certaine « poétique[40] ». Jost Hochuli a montré la manière dont la conception d'un livre ne suit pas uniquement un processus logique et réfléchi et est aussi guidé par une part d'instinct, d'intuitions et de questionnements que seule l'expérience peut forger. La conception est aussi un dialogue avec un ensemble d'autres déterminants – comme le contexte de réception ou l'économie d'un projet. Les instruments techniques font tout autant partie de ces déterminants et leur influence sur les pratiques créatives doit s'éclairer à l'aune de chaque situation de conception singulière.

Ce travail a bénéficié d'une aide de l'EUR ArTeC financée par l'ANR au titre du PIA ANR-17-EURE-0008.

Bibliographie

André, Jacques, et Vincent Quint, « Structures et modèles de documents », dans *Le document électronique*, INRIA, 1990.

Baudin, Fernand, *L'effet Gutenberg*, Paris, Editions du Cercle de la Librairie, 1994.

Bos, Bert, « A brief history of CSS until 2016 », *w3.org*, publié le 17 décembre 2016, https://www.w3.org/Style/CSS20/history.html.

Caserta, John, *For/with/in: Graphic Design for, with, and in the Browser*, The Design Office, 2014, http://htmloutput.risd.gd/.

Conrad, Taylor, « Mais qu'est ce qu'ont bien pu nous apporter les systèmes WYSIWYG ? », *Cahiers GUTenberg*, no 27,1997, pp. 5-33.

Hochuli, Jost, *Un design de livre systématique*, Paris, Éditions B42, 2020.

Lie, Håkon Wium, *Cascading Style Sheets*, PhD thesis, University of Oslo, 2005, https://www.wiumlie.no/2006/phd/.

Müller-Brockmann, Josef, *Grid systems in graphic design: a visual communication manual for graphic designers, typographers and three dimensional designers*, Sulgen (Suisse), Verlag Niggli, 1981.

Rendle, Robin, « The Futures of Typography ». *Robin Rendle* (blog), publié le 06/01/2017, https://www.robinrendle.com/essays/the-futures-of-typography.

Richaudeau, François, et Olivier Binisti, *Manuel de typographie et de mise en page: du papier à l'écran*, Paris, Retz, 2005.

Tschichold, Jan, Nicole Casanova, et Muriel Paris, *Livre et typographie: essais choisis*, Paris, 2018.

Vilayphiou, Stéphanie, et Alexandre Leray, « Écrire le design : Vers une culture du code », *Back Cover*, no 4, 2011, pp. 37-44.

West, Octavio, « Automatic Layout and Typesetting. Will Machines Replace Designers in Publishing ? » *PUB800* (blog), publié le 28 novembre 2017, https://tkbr.publishing.sfu.ca/pub800/2017/11/automatic-layout-and-typesetting-will-machines-replace-designers-in-publishing/.

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

1. Une liste de collectifs et de designers indépendants francophones utilisant ces technologies ainsi que des exemples de publications sont réunis sous le label *PrePostPrint* et disponibles sur le site prepostprint.org.
2. Le groupe Hachette compose depuis quelques années certains de ces livres avec HTML et CSS (et le logiciel propriétaire Prince XML). Voir à ce propos: Dave Cramer, « Beyond XML: Making Books with HTML », *xml.com*, publié le 20/02/2017, https://www.xml.com/articles/2017/02/20/beyond-xml-making-books-html/
3. La Collaborative Knowledge Foundation (Coko), proposant des systèmes de publications collaboratifs et multisupports, est à l'origine du développement de Paged.js, un outil libre et open-source permettant de prendre en charge les fonctionnalités CSS nécessaire à l'impression et non disponibles encore dans les navigateurs web.
4. Kiara Jouhanneau, *Rêves Party*, projet présenté pour l'obtention du diplôme national des métiers d'art et du design à l'ensemble scolaire Saint-Étienne de Cahors, 2021, https://kiarajouhanneau.gitlab.io/reves-party/
5. *L'idéal anarchique*, workshop au Bel Ordinaire (Billère, France), du 14 au 18 novembre 2021, https://maisondeseditions.fr/ideal/
6. André, Jacques, et Vincent Quint, « Structures et modèles de documents » dans Bornes, Christian (dir.), *Le document électronique*, INRIA, 1990.
7. La structuration des documents offre des avantages en terme de collaboration entre différents métiers. La récupération de document déjà structurés depuis des fichiers partagés et versionnés couplé à un *reflow* « automatique » des documents générés à la volée permet d'éviter des interventions manuelles suite à l'ajout, la correction ou la suppression de texte. Particulièrement si ces interventions concernent des étapes d'édition tardives.
8. Fauchié, Antoine, et Thomas Parisot, « Repenser les chaînes de publication par l'intégration des pratiques du développement logiciel », *Sciences du Design*, n° 8, 12/2018, pp. 45-56. Fauchié, Antoine, « Une chaîne de publication inspirée du web », *quaternum.net*, publié le 13/03/2017, URL: \https://www.quaternum.net/2017/03/13/une-chaine-de-publication-inspiree-du-web/\.
9. Dunn, Max, « Single-Source Publishing with XML », *IT Professional 5*, n° 1, 01/2003, pp. 51-54.
10. « Un ensemble de règles qui associent des propriétés et des valeurs stylistiques aux éléments structurels d'un document, exprimant ainsi la manière de présenter le document ». Lie, Håkon Wium, « Cascading Style Sheets », PhD thesis, University of Oslo, 2005, p. 77.
11. Traduction: « Le WorldWideWeb est une initiative de recherche d'information hypermédia à grande échelle visant à donner un accès universel à un vaste ensemble de documents. » Berners-Lee, Tim, *World Wide Web*, publié le 6/8/1991, http://info.cern.ch/hypertext/WWW/TheProject.html.
12. « The [WWW] project is based on the philosophy that much academic information should be freely available to anyone.» Traduction: « Le projet est fondé sur la philosophie selon laquelle une grande partie de l'information académique devrait être librement accessible à tous. » Berners-Lee, Tim, *World Wide Web - Summary*, publié le 6/8/1991, http://info.cern.ch/hypertext/WWW/Summary.html.
13. Fondé en 1994, le World Wide Web Consortium (W3C) est l'organisme chargé d'imaginer et maintenir touts standards ouverts du web (HTML,XML, CSS, PNG, SVG, etc.) . Le consortium accueille toutes sortes d'organisation intéressées par le futur du web (entreprises tech, navigateurs web, laboratoires scientifiques, etc.) Tout y est développé de manière ouverte et aucune entreprise ne peut avoir le contrôle.

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

14. Notons qu'à cette époque, le Web devait aussi marcher sur des machines sans interfaces graphiques, qui étaient encore très courantes à l'époque.
15. Voir *XHTML2 Working Group*, 03/2007, https://www.w3.org/MarkUp/HTMLConstraints.html
16. En 1999, Internet Explorer 5 de Microsoft est le premier navigateur Web à supporter CSS de façon convenable.
17. Le site CSS Zen Garden, créé en 2003 par Dave Shea, se veut la démonstration de la possibilité de modifier librement le rendu affiché d'une même page web, uniquement grâce à CSS et sans aucune modification de son code HTML. Il présente, en décembre 2006, 986 designs différents de sa page d'accueil. http://www.csszengarden.com/
18. La devise du World Wide Web Consortium (W3C), organisme de standardisation à but non lucratif chargé de développer les technologies est ainsi « Web for All. Web on Everything. »
19. C'est nous qui soulignons. Traduction : « Les navigateurs actuels considèrent l'écran de l'ordinateur comme la cible principale de la présentation, mais [CSS] a le potentiel de prendre en charge de nombreux supports de sortie, par exemple le *papier*, les appareils de synthèse vocale et le braille. » https://www.w3.org/People/howcome/p/cascade.html.
20. *Media Queries, W3C Candidate Recommendation*, publié le 8/7/2002, https://www.w3.org/TR/2002/CR-css3-mediaqueries-20020708/.
21. Moyennant l'ajout d'un outil comme paged.js. Il est aussi possible d'utiliser des logiciels (propriétaires) permettant de générer directement de PDF en HTML et CSS: Prince XML, Antenna House, PDF Reactor, etc.
22. Concernant cette critique, voir: Conrad, Taylor; « Mais qu'est ce qu'ont bien pu nous apporter les systèmes WYSIWYG? », *Cahiers GUTenberg*, n° 27, 1997, pp. 5-33.
23. Par exemple, un élément de type tableau (\<*table*>) contient des lignes de tableaux (\<*tr*>) qui elles-même peuvent contenir des cellules (\<*td*>) et/ou des cellules d'en-tête (\<*th*>). On dit alors qu'une cellule est un élément enfant de l'élément ligne de tableau ou inversement, que le tableau est le parent des éléments lignes et cellules qu'il contient.
24. La construction de gabarits de page se fait en effet en fonction des éléments présents dans le texte. C'est parce qu'un élément est spécifié comme « chapitre » qu'un gabarit pourra être appliqué et les gabarits peuvent s'additionner entre eux en fonction des éléments de la page.
25. Un parfait exemple de relations et règles interdépendantes utilisées dans la mise en page se trouve dans la construction du bloc d'empagement (accueillant le texte dans l'espace de la page). Ses caractéristiques résultent d'un ensemble d'éléments qui interagissent les uns avec les autres: dimension des caractères, justification, longueur des lignes, taille des marges, espacement des lignes, etc.
26. Tschichold, Jan, Nicole Casanova, et Muriel Paris, *Livre et typographie: essais choisis*, Paris, 2018, p.10.
27. Müller-Brockmann, Josef, *Grid systems in graphic design: a visual communication manual for graphic designers, typographers and three dimensional designers*, Sulgen, Verlag Niggli, 1981.
28. Capus, Pascal, *Les sculptures de la villa romaine de Chiragan*, Toulouse, Musée Saint Raymond, 2020.
29. Notons que les calculs et les variables peuvent être utiles pour d'autres types d'usages. Par exemple, l'encodage numérique des couleurs permet d'imaginer le passage progressif d'une couleur à l'autre d'un élément à l'autre grâce à une série de calcul.
30. Un certain nombre de propriétés CSS permettent aux designers graphiques de contrôler la fragmentation de ce flux en imposant des sauts de page forcés, en évitant le découpage de certains éléments ou en paramétrant des sauts de pages contextuels: veuves, orphelines,

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192

titre trop bas sur la page, etc.

31. Par exemple, les flottements d'éléments liés à la page sont des spécifications dédiées à la mise en page imprimée. Voir Blanc, Julie, « Paged Media approaches : page floats », *pagedjs.org* (journal); publié le 01/02/2020, https://www.pagedjs.org/page-floats/.
32. Seurat, Clémence et Thomas Tari, *Controverses mode d'emploi*, Paris, Presses de Sciences Po, 2021.
33. La création de scripts JavaScript permet de prendre en charge des mécanismes de mises en page pour lesquelles des syntaxes CSS (en constante évolution) n'auraient pas encore été implémentées dans les navigateurs ou sont insuffisantes pour arriver au résultat souhaité.
34. Voir: « Even more about how Flexbox works — explained in big, colorful, animated gifs », *freecodecamp.org*, publié le 20/02/2017, https://www.freecodecamp.org/news/even-more-about-how-flexbox-works-explained-in-big-colorful-animated-gifs-a5a74812b053/
35. Entretien avec Amélie Dumont réalisé le 5 janvier 2021 par l'autrice de ce texte (non publié).
36. *Idem*.
37. Richaudeau, François, et Olivier Binisti, *Manuel de typographie et de mise en page: du papier à l'écran*; Paris, Retz, 2005, p. 25.
38. Pour des workshops menés dans des formations en art et design, il n'est pas rare que les étudiants et étudiantes mixe l'utilisation des technologies du Web pour l'imprimé avec des librairies comme Paper.js ou P5.js qui sont basé sur Javascript, davantage un langage de programmation que HTML et CSS (qui restent des langages descriptifs).
39. Il n'a pas été question dans ce texte des logiques algorithmiques de génération automatique de mise en page, notamment à partir de l'intelligence artificielle, qui sont pour nous de l'ordre d'un tout autre débat. Ce débat se situerait d'ailleurs bien plus du côté des logiciels de création, proposant de véritable « boîtes noires » à leurs utilisateurs. Voir à ce propos: Masure, Masure, « Copier/Varier. Standards, critiques, et contre-emplois des logiciels de création », *Multitudes*, n° 82, « Globalisations esthétiques », dir. Nathalie Blanc et David Christoffel, mai 2021.
40. Hochuli, Jost, *Un design de livre systématique?*, Paris, Éditions B42, 2020.

© La revue DAM
téléchargé le 2026-11-24 00:09:31, depuis le 216.73.216.192