DESIGN WEB

Préface d'Élie **Sloïm** & Laurent **Denis**

Corinne **Schillinger**

Intégration web

Les bonnes pratiques

LE GUIDE DE SURVIE DE L'INTÉGRATEUR !

EYROLLES

Intégration web

« Avec le mouvement d'industrialisation des métiers du Web, chaque métier voit émerger un livre de référence. Pour l'intégrateur web, c'est chose faite grâce à ce livre, qui représente un jalon important dans l'histoire de cette discipline en évolution permanente ! »

Élie Sloïm,
Président de Temesis
et Directeur des projets
Opquast et Openweb

Intermédiaire entre le chef de projet, le designer et le développeur, **intégrateur web** (ou développeur *front-end*) est un métier à part entière. Maillon indispensable dans la création de sites, il rassemble tous les éléments pour monter les pages et livrer un site **performant, accessible et utilisable**. Son travail nécessite tout autant de compétences techniques en HTML, CSS et JavaScript qu'une bonne appréhension de la qualité web (ergonomie, accessibilité, performance…).

S'appuyant sur l'expérience de l'auteur, souvent appelée à la rescousse sur des projets mal engagés, ce livre offre aux intégrateurs débutants, comme aux plus expérimentés, un recueil précieux des meilleures pratiques, méthodes et outils pour améliorer et réussir leurs intégrations dans le respect des standards du Web.

Une référence pour tous les créateurs de sites web !

Intégratrice senior et Experte AccessiWeb en Évaluation, **Corinne Schillinger** s'est forgée une solide expérience dans le domaine du développement *front-end* avant de créer son entreprise, Inseo, en 2009. Ayant à cœur de promouvoir les standards et la qualité web, elle a intégré l'association Paris Web en 2011, rejoint l'équipe éditoriale du Train de 13h37 et partage son expérience en animant régulièrement des ateliers à l'occasion d'évènements dédiés à la conception web.

AU SOMMAIRE

DESIGN WEB

www.editions-eyrolles.com
Groupe Eyrolles | Diffusion Geodif

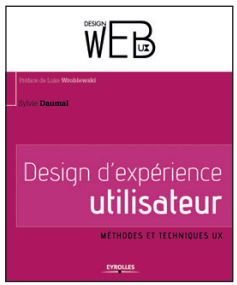
Code G13370
ISBN 978-2-212-13370-7

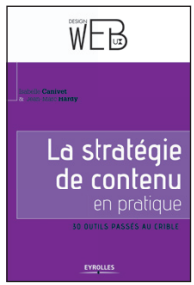
Conception Nord Compo

Intégration web : les bonnes pratiques

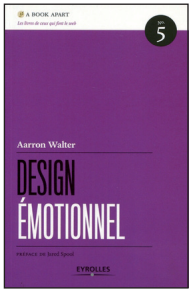
LE GUIDE DE SURVIE DE L'INTÉGRATEUR !

Dans la collection Design Web

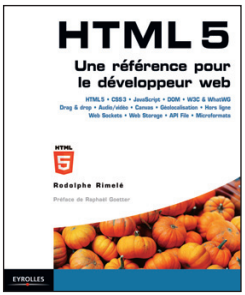


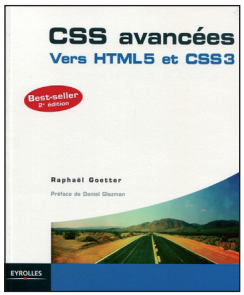


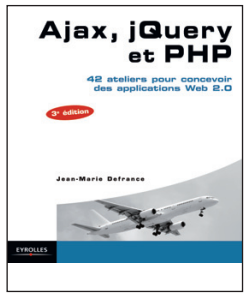
Dans la collection A Book Apart

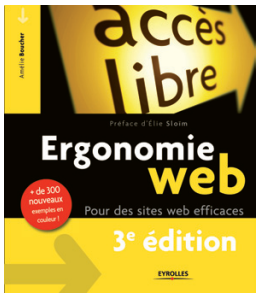


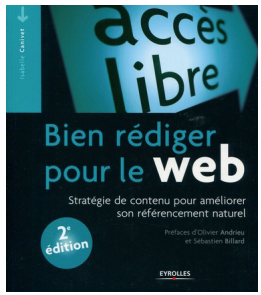
Chez le même éditeur











Corinne Schillinger

Intégration web : les bonnes pratiques

LE GUIDE DE SURVIE DE L'INTÉGRATEUR !

Préface d'Élie Sloïm & Laurent Denis

EYROLLES

ÉDITIONS EYROLLES
61, bld Saint-Germain
75240 Paris Cedex 05
www.editions-eyrolles.com

Remerciements à Virginie Caplet (http://cheekfille.com) pour ses illustrations d'ouverture de parties et à Renaud Scapin pour la réalisation des pictogrammes.

ISBN : 978-2-212-13370-7

PRÉFACE

Rappelez-vous. Il y a quelques années, la plupart des webmasters (car on les appelait encore comme cela) utilisaient des logiciels comme Dreamweaver, Frontpage ou Hotmetal pour créer leur sites Internet. Ces logiciels WYSIWYG *(What You See Is What You Get)* permettaient de faire du code plus ou moins propre pour un navigateur ultra-majoritaire, en l'occurrence, Internet Explorer.

À cette époque, l'intégrateur n'existait pas et le webmaster n'avait pratiquement besoin d'aucune compétence spécifique ni d'aucune formation : il était en pleine invention de son métier, de ses outils et de ses technologies.

C'est alors que sont arrivées successivement les feuilles de styles et la séparation du fond et de la forme. Le processus de création de site web s'est alors découplé en deux phases successives : le design puis l'intégration. Ces deux activités ont longtemps été assumées par des personnes aux profils similaires, avant que chacun des deux métiers ne se constitue progressivement.

Intégrateur front-end est aujourd'hui un vrai métier et Corinne Schillinger nous le prouve à travers l'exposé détaillé de son utilisation des technologies, de ses méthodes, approches, processus, standards et pratiques. Il s'y dessine finalement un véritable écosystème, un ensemble de relations entre métiers du Web, où l'intégrateur joue un rôle clé.

L'intégrateur a une fonction pivot entre plusieurs compétences. On pense immédiatement au découpage et à l'intégration HTML-CSS, ainsi qu'au développement JavaScript côté client. En sus, on lui demande de maîtriser d'autres compétences beaucoup plus variées : la gestion de la compatibilité multinavigateur, la maîtrise des questions de performances, d'expérience utilisateur, la prise en compte d'ARIA en matière d'accessibilité, le choix de recourir ou non à HTML5, etc.

L'intégrateur doit jongler avec les exigences du référencement, de la performance, de l'ergonomie, de l'accessibilité, de la typographie, etc. Il doit en outre gérer la diversification des médias et des modes d'interaction : desktop, mobile, tablettes. Bientôt, il devra certainement assumer la généralisation du tactile.

En tant que « qualiticiens », nous ne pouvons que confirmer cet état de fait : lorsque nous travaillons sur les bonnes pratiques de la qualité web, bon nombre d'entre elles se concentrent sur la phase d'intégration.

L'intégrateur front-end doit donc savoir maîtriser un grand nombre de standards, de bonnes pratiques, de compétences, de méthodes, aussi bien du point de vue technique que managérial. Il était plus que temps de rassembler tous ces éléments dans un livre.

Jusqu'à présent, les ouvrages existants se concentraient principalement sur des sujets techniques, comme CSS, HTML, JavaScript. Là est la force de l'auteur : en plus de poser les bases, de formaliser et de déterminer ce qui est de l'ordre du savoir minimal de l'intégrateur, elle a su dépasser la technique pour aborder ces sujets sous l'angle métier.

La publication de livres comme celui de Corinne Schillinger est l'un des nombreux signes que le processus d'industrialisation est en marche. Alors, oui, la connaissance technique est fondamentale, mais, non, elle ne suffit pas. Des experts techniques, il en existe beaucoup, mais des personnes qui comprennent la philosophie qui sous-tend les techniques et la façon de les déployer sont beaucoup plus rares. C'est pourquoi il faut aller beaucoup plus loin.

Aujourd'hui, nous entrevoyons le moment où l'intégrateur va enfin devoir aller au-delà de la technique. Il est désormais celui qui doit accorder des violons souvent discordants, entre les exigences du design, du référencement, de la DSI… Il commence à en acquérir les moyens avec l'industrialisation du Web. Il est appelé à devenir de plus en plus un manager de solutions, appuyé sur une maîtrise rigoureuse des techniques et surtout des clés de décision.

D'une certaine manière, au même titre que les autres professionnels du projet web, l'intégrateur devient un véritable gestionnaire de risques. Les risques auxquels il doit faire face et qu'il sera de plus en plus amené à gérer sont particulièrement critiques pour les clients et les utilisateurs finaux.

Avec cette évolution, l'intégrateur se servira différemment de ses références techniques actuelles. Il aura rarement à produire du code : le cœur de son métier sera de choisir des solutions. Il n'est déjà plus un simple exécutant opérationnel : il lui revient de plus en plus souvent d'arbitrer sur le choix des formats, sur celui d'un framework JavaScript, voire d'un framework CSS, ou même sur la décision quant au CMS à utiliser.

Au terme de cette maturation du métier, sa compétence proprement technique pourra alors être réservée aux seuls cas où une certaine forme d'expertise sera requise pour des solutions sur mesure.

Gageons que si le mouvement d'industrialisation des métiers du Web se poursuit, il faut s'attendre à l'émergence de livres de référence sur chacun des principaux métiers du Web. Pour ce qui est du métier d'intégrateur, l'ouvrage que vous avez entre les mains sera à l'évidence l'un de ces livres de référence et, au minimum, un jalon important dans l'histoire de ce métier en évolution permanente.

Élie Sloïm (Président de Temesis, Directeur des projets Opquast et Openweb)

et Laurent Denis (Expert accessibilité)

TABLE DES MATIÈRES

AVANT-PROPOS

Avant d'entrer dans le vif du sujet, permettez-moi de vous présenter cet ouvrage en quelques mots.

À qui s'adresse ce livre ?

Ce livre s'adresse à tous les intégrateurs, professionnels ou amateurs, qui ont à cœur de réaliser une intégration qui soit aussi propre, accessible et performante que possible. Il se veut clair et exhaustif, sans toutefois reprendre les bases fondamentales des langages HTML et CSS. De ce fait, il risque d'être assez difficile d'accès aux débutants souhaitant découvrir les joies du code, car il requiert une connaissance minimale de ces deux langages pour pouvoir être pleinement exploité.

Malgré tout, il cible un public relativement large puisqu'il est destiné à toutes les personnes qui ont envie de produire un code de qualité : les étudiants souhaitant approfondir leurs cours devraient y trouver leur compte tout autant que les professionnels aguerris désirant mettre leurs connaissances à jour.

Pourquoi ce livre ?

La qualité globale d'un site repose bien souvent sur des points de détail qui ont un impact positif ou négatif sur son utilisabilité, son accessibilité et ses performances d'affichage. Et bien que les outils et techniques aient considérablement évolué ces dernières années, l'intégration est une discipline qui reste encore très artisanale : son résultat fluctue en fonction des connaissances de l'intégrateur.

Or, bien souvent, il ne manque pas grand-chose pour que la qualité de l'intégration soit considérablement améliorée : spécifier un changement de langue, prévoir l'indisponibilité d'une ressource ou tirer parti de l'héritage en sont quelques exemples.

Forte de ce constat, j'ai mûri l'idée de créer une sorte de recueil de bonnes pratiques de l'intégration web. Mais ce n'est qu'une fois le travail rédactionnel commencé que je me suis rendue compte de l'ampleur réelle de la tâche. Car même si certains usages sont considérés comme étant de bonnes pratiques, beaucoup restent à la discrétion du développeur : l'intégration n'est pas une science exacte, elle dépend d'une multitude de paramètres qui en font une discipline aussi complexe qu'intéressante.

L'écriture de ce livre aura donc été l'occasion d'une remise en cause de mes méthodes de travail. Le plus difficile aura été d'évaluer la pertinence réelle de ma méthode, car même si elle me convenait parfaitement, rien ne pouvait me laisser penser qu'il s'agisse de la « meilleure » façon de faire. Je me suis donc posée de nombreuses questions, ai discuté avec d'autres professionnels, et ai souvent changé d'avis devant la justesse des arguments avancés. Vous tenez entre vos mains le fruit de ces dix-huit mois de recherches, de questionnements, d'introspection et de discussions passionnantes avec des intégrateurs de tous bords.

Durant ce laps de temps, la technique a évolué et des méthodes qui étaient autrefois conseillées sont aujourd'hui considérées comme de mauvaises pratiques : dix-huit mois dans un domaine où les techniques évoluent constamment, c'est beaucoup. Et s'il s'agit-là de la première version papier, sachez que certaines parties en sont au moins à leur troisième mise à jour : je n'ai eu de cesse de réactualiser le contenu pour assurer la véracité des informations qui y figurent.

C'est pourquoi j'espère sincèrement que ce livre répondra à vos attentes. S'il vous permet d'approfondir de nouvelles techniques, de vous mettre à niveau, de prendre connaissance de certains effets de bord indésirables ou de découvrir une nouvelle méthodologie, j'aurai gagné mon pari.

Comment le livre est-il organisé ?

Ce livre est organisé dans un ordre logique qui respecte les différentes étapes devant être effectuées pour mener à bien une intégration. Ces dernières sont réparties en quatre parties que sont la préparation en amont, la conception du contenu HTML, la mise en forme CSS et la préparation des livrables.

Vous aurez certainement remarqué qu'aucune partie n'est dédiée à JavaScript. Pourtant, il y aurait de quoi faire : un livre entier pourrait même être consacré à ses bonnes pratiques de développement. Mais je n'estime pas avoir les compétences suffisantes pour pouvoir l'aborder avec le degré d'expertise que je souhaiterais. C'est pourquoi je préfère laisser cette tâche à un développeur JavaScript qui saura le faire bien mieux que moi.

Pour autant, cela ne signifie pas que la question de son utilisation ne soit pas abordée : son emploi impose de nombreuses contraintes qui sont évoquées tout au long de ce livre. Car même si l'intégrateur ne développe pas lui-même les fonctions, il se doit de

maîtriser les bases du langage et de pouvoir concevoir certaines interactions à l'aide (ou non) d'un framework.

La méthode présentée ici n'a pas vocation à devenir universelle : je doute d'ailleurs qu'aucune méthode ne fasse jamais cette unanimité. Elle tend plutôt à être aussi exhaustive que possible et aborde toutes les étapes qui peuvent affecter la qualité finale de l'intégration. C'est pourquoi ce livre débute avec l'organisation de l'espace de travail et se termine avec la préparation des livrables. Tout au long de son développement, vous retrouverez ces notions d'accessibilité, de sémantique, de lisibilité, de maintenabilité et de performance qui composent ce tout qu'est la qualité web.

REMERCIEMENTS

Si je m'écoutais, je remercierais la terre entière pour la chance qui m'a été donnée de concrétiser cette opportunité. Je commencerais d'ailleurs par ma maman, sans qui rien de tout ça n'eut été possible. Mais s'agissant d'un livre professionnel, je crains malheureusement que cela ne soit guère adapté… Je vais donc me restreindre aux personnes qui ont donné de leur temps et n'ont pas hésité à partager leur expertise pour que ce livre puisse voir le jour. Si cet ouvrage vous plaît, c'est aussi à toutes ces personnes que vous le devez : sans leurs remarques constructives, leurs encouragements et leur disponibilité, vous n'auriez certainement pas ce livre entre les mains.

Occasionnels ou au long court, les relecteurs ont eu la lourde de tâche de vérifier l'exactitude de mes assertions. Ils ont relevé les erreurs, approximations et incohérences qui avaient échappé à ma vigilance, et m'ont conseillé de nombreux ajouts très pertinents. C'est pourquoi je tiens à remercier chaleureusement :

- Anthony Ricaud, Laurent Denis, Loïc Mathaud et Sébastien Delorme, pour avoir gentiment accepté de se prêter au jeu ;
- Vincent Valentin, qui a trouvé un peu de temps entre deux biberons pour me relire et répondre à mes interrogations diverses et variées ;
- Jérémie Patonnier, pour avoir passé de longues heures, tard le soir, à relire plusieurs chapitres et pour m'avoir fait de nombreux retours toujours très judicieux, bien qu'un peu « bruts de décoffrage » parfois ;
- Et enfin, Marie Alhomme qui a accepté de relire ce livre dans son intégralité. Je la remercie du fond du cœur pour tout le temps qu'elle y a consacré et les longs moments passés à discuter détails techniques, méthodologie et bonnes pratiques.

Dans un registre différent, je tiens également à remercier :

- Noëlie Amiot pour ses éclairages sur certains points techniques ;
- Élie Sloïm et Laurent Denis qui m'ont fait l'honneur de rédiger à quatre mains la préface de cet ouvrage ;
- Virginie Caplet, une graphiste de talent, qui a accepté — pour mon plus grand bonheur — de réaliser les magnifiques illustrations qui introduisent les parties principales de ce livre ;
- les éditions Eyrolles pour m'avoir fait confiance, et plus particulièrement Karine Joly qui a fait preuve d'une patience sans borne, acceptant sans mot dire mes retards de livraison et n'ayant de cesse de m'encourager jusqu'à la dernière minute ; Sophie Hincelin et

Géraldine Noiret qui ont repris le projet juste avant sa finalisation et se sont démenées pour le boucler dans les temps ;
- Et surtout Yannick — mon homme — qui m'a incitée à me lancer dans cette aventure, soutenue de la première ligne jusqu'à la dernière relecture, rassurée quand j'étais en pleine phase de doute, et supportée même lorsque des nuits bien trop courtes me rendaient absolument exécrable. Si j'ai réussi à mener ce projet à son terme, c'est surtout grâce à lui.


Recette du projet réussi
- 1 intégrateur consciencieux
- des dossiers bien ordonnés
- des technos maîtrisées
- du café décaffeiné
- de la patience
FARINE

Partie 1

BIEN PRÉPARER SON PROJET

On estime bien souvent que l'intégration d'un site web débute avec la conception du code HTML/CSS. Or si le code est très certainement la partie la plus visible du site, il est important de ne pas considérer la préparation du projet comme une simple formalité qu'il est possible d'expédier. Qu'il s'agisse de l'organisation des éléments (fichiers et répertoires), du choix des outils ou de la mise en place de l'environnement de test, toutes les étapes préparatoires doivent faire l'objet de la plus grande attention.

1

Organiser son espace de travail

Une arborescence logique et facilement compréhensible, une règle de nommage appliquée à l'ensemble des composants ou encore la présence d'une fiche de suivi à la racine du projet participent pleinement à la qualité globale du site et contribuent à simplifier son évolution future. Ainsi, même s'il n'existe aucune règle absolue à suivre, ni aucune organisation qui soit parfaite, il est important de réfléchir à la façon dont le projet va être organisé, avant de rédiger la moindre instruction.

Dans ce chapitre

- Arborescence : créer un dossier par projet
- Uniformité : adopter une convention de nommage
- Qualité : mettre en place une checklist détaillée

ARBORESCENCE : CRÉER UN DOSSIER PAR PROJET

Avoir tous les éléments à portée de main

La première chose à faire avant de démarrer une nouvelle intégration est de préparer l'espace de travail. L'idéal est de rassembler tous les éléments relatifs à un projet dans un même dossier ou répertoire. Il suffit simplement d'organiser à l'intérieur de celui-ci une arborescence claire dans laquelle viendront se placer les documents. En centralisant ainsi toutes les données, vous ne perdrez pas de temps à chercher des éléments manquants ou à vérifier que vous vous référez à la dernière version des spécifications. De même, un collègue arrivant sur le projet en cours de route aura une vue exhaustive du projet et de son avancée.

Bien sûr, chaque intégrateur a sa façon de faire et son propre système d'organisation. L'essentiel étant qu'il s'y retrouve et qu'une personne tierce ne soit pas perdue dans l'arborescence. Voici par exemple une classification en trois dossiers, qui permet de répartir simplement tous les éléments du projet.

- Le dossier `sources` accueille les documents utiles à l'intégration. Il peut s'agir du code à reprendre, des maquettes à intégrer ou encore des textes à publier. Suivant le nombre d'éléments (et surtout de maquettes), il peut être utile d'avoir recours à des sous-dossiers.
- Le dossier `spec` recueille l'ensemble des spécifications fonctionnelles du projet. Y prennent place le cahier des charges, le zoning, l'arborescence ou encore les demandes d'évolution.
- Enfin, le dossier `www` est réservé à l'intégration proprement dite. Il constitue la racine du site web et contient l'ensemble des éléments qui vont être livrés ou mis en ligne, au contraire des sources et des spécifications qui ne seront pas poussées en production.

Certains préfèrent, pour plus de clarté, rassembler l'ensemble des ressources en ligne. Indispensable pour les gros projets (longs de plusieurs mois, sur lesquels interviennent plusieurs personnes en parallèle), cette organisation présente le défaut de nécessiter une connexion Internet lorsque vous travaillez. Ce qui n'est pas toujours le cas si vous êtes en situation de mobilité ou si vous faites les frais d'une coupure temporaire.

GLOSSAIRE Les composants d'un projet

Chaque nouvelle intégration amène son lot d'éléments à analyser. Détaillant le projet dans son ensemble ou traitant de points particuliers, les spécifications sont généralement composées de plusieurs documents : présentations, textes, images, graphiques...

Le **cahier des charges** (ou *brief*) est sans conteste l'élément le plus important de toutes les spécifications : il sert de fil conducteur au projet et fait référence en toute situation. Il décrit entre autres les objectifs à atteindre, la forme du livrable, les critères d'évaluation, les contraintes à respecter, le délai de réalisation et, bien sûr, le résultat attendu.

L'**arborescence** (ou **architecture**) se concentre uniquement sur l'organisation du site. Généralement représentée sous la forme d'un arbre inversé, elle permet de visualiser rapidement l'agencement des pages et les interactions qui les lient les unes aux autres.

Le ***zoning*** schématise l'organisation générale d'une page en identifiant les différentes zones à l'aide de blocs (voir l'annexe A1). Il permet d'illustrer le fonctionnement global et de dégager les zones de contenu principales. Chaque zoning s'attachant à un seul type de page (page d'accueil, page de contenu...), il est fréquent d'en avoir plusieurs pour un même site.

Établi à partir du zoning, le ***wireframe*** (ou maquette fil de fer) met en avant le contenu. Les zones précédemment identifiées sont remplies et leur contenu est organisé de la manière la plus fonctionnelle et ergonomique possible (voir l'annexe A2).

Partant de l'agencement validé dans le wireframe, le graphiste imagine et réalise un univers propre à la marque qui véhicule ses valeurs et son identité. Il s'agit de la **maquette graphique**.

Organiser les répertoires en fonction du type de fichiers

Une fois que l'arborescence est en place et que les éléments sont répertoriés, il convient de préparer le dossier d'intégration (`www` dans notre exemple).

Il arrive que l'organisation générale soit fournie par le client. C'est souvent le cas lorsqu'il s'agit d'une refonte et qu'il n'est pas envisageable de modifier le système de dossiers utilisé. En l'absence de consignes particulières, il est important de regrouper les fichiers suivant leur type et leur fonction.

Bien que certains usages se soient mis en place, il n'existe aucune règle stricte concernant le nommage des éléments. Ainsi, il est fréquent de rencontrer plusieurs approches différentes : l'emploi de noms très courts (`i` pour les images, `s` pour les scripts, etc.), de noms complets (`images`, `scripts`, etc.) ou encore de noms abrégés (`img`, `js`, etc.). Au final, les noms utilisés importent relativement peu, du moment qu'ils sont courts et explicites. C'est pourquoi la tendance générale consiste à privilégier les appellations raccourcies et à utiliser l'extension de fichier comme nom de dossier : `css`, `js`, `swf` ou encore `flv`. Cette façon de faire est un bon compromis entre les performances et la lisibilité des noms abrégés.

Cas particulier : les images

Au cours d'une intégration, trois types d'images sont à distinguer : les images de contenu, les images décoratives et les images temporaires ou contribuables.

Les images de contenu

Porteuses de sens, elles font partie intégrante du code HTML : c'est le cas d'un logo ou d'une icône de menu. Généralement, elles sont enregistrées dans un dossier `img` à la racine du dossier d'intégration. Bien sûr, ce dernier peut contenir l'un ou l'autre sous-

dossier (comme `icone` ou `picto`), pour regrouper certaines images en fonction de leurs similitudes.

Les images décoratives

Utilisées pour la mise en forme du document, elles sont appelées depuis le code CSS. Certains les enregistrent dans un dossier `img` à la racine de `css`. Une autre manière de faire consiste à les placer dans un dossier `theme` directement dans `www`. Ainsi, il suffit d'ajouter un sous-dossier pour chaque nouveau thème ou déclinaison saisonnière (comme les fêtes de Noël ou la période des soldes).

Bonne pratique Utiliser les classes et identifiants comme noms d'images

Nommer les images avec les noms de classes et d'identifiants qu'elles habillent permet d'établir un lien entre le code et les images. La gestion des ressources en est ainsi facilitée, puisque chaque élément peut aisément être relié à la structure HTML et aux styles CSS.

```
#footer { background: url("../theme/footer.jpg") repeat-x 0 100%; }
```

Les images temporaires (ou contribuables)

Issues de la maquette, elles sont généralement employées par le graphiste pour illustrer l'une ou l'autre mise en situation. Il peut par exemple s'agir d'un avatar, d'une illustration d'article, ou encore de la capture d'une vidéo. Faciles à identifier, elles doivent être stockées dans un dossier distinct qui ne sera pas mis en production. Dans notre arborescence, le dossier `tmp` se trouve à la racine de `www`. Nous y plaçons l'ensemble des éléments temporaires (images, sons, vidéos...) utilisés au cours de l'intégration.

Conseil Employer des images factices

Dummy Image *(Dynamic Dummy Image Generator)* est un générateur d'images factices. Générées suivant les paramètres passés dans l'URL, les images peuvent être utilisées pour remplacer les fichiers temporaires. Ainsi, il n'est plus nécessaire d'exporter des éléments qui ne seront pas utilisés en production. Cela permet surtout d'attirer l'attention du client sur ses futures contributions (les dimensions du visuel étant incluses dans l'image).

http://dummyimage.com

Conseil Créer un « dossier projet » type

Une fois que vous avez opté pour une arborescence qui vous convient et avec laquelle vous vous sentez à l'aise, enregistrez-la soigneusement, car il est fort probable que vous la réutilisiez sur l'ensemble de vos intégrations ultérieures. Ne perdez donc pas de temps à repartir de zéro à chaque fois : dupliquez plutôt cette arborescence au démarrage de chaque nouveau projet.

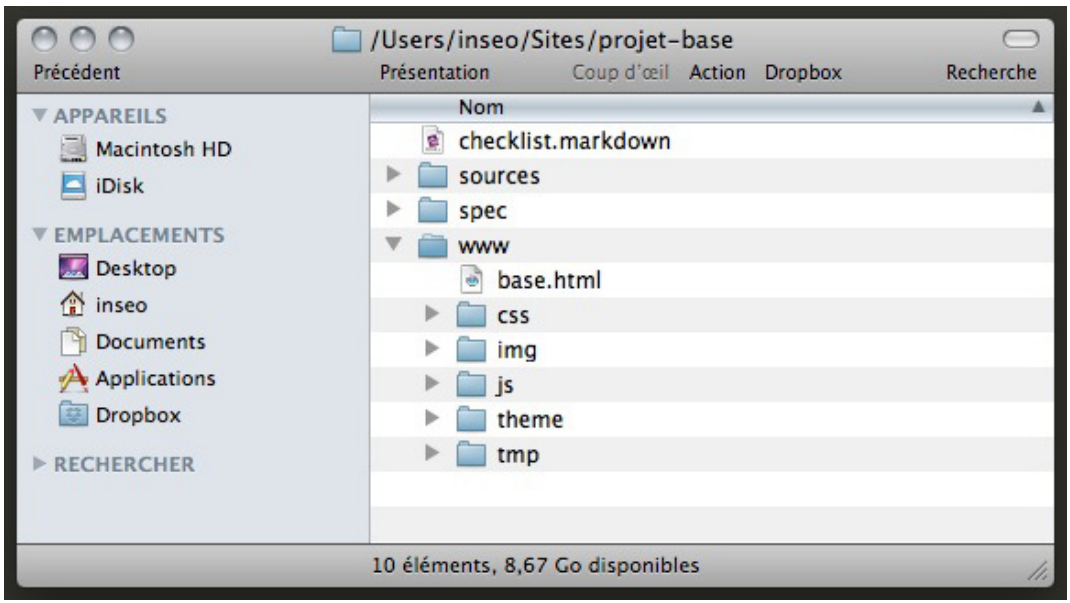


Fig. 1-1 : Exemple d'arborescence réutilisable à chaque nouveau projet

Adopter une convention de nommage

L'une des erreurs les plus fréquentes en intégration est sans nul doute l'erreur d'inattention. Qui n'a jamais passé de longues minutes à essayer de déboguer un élément au comportement étrange, avant de se rendre compte qu'une erreur s'était glissée dans l'appel du sélecteur ?

L'absence de convention de nommage standardisée force les intégrateurs à inventer la leur. Malheureusement, rares sont ceux qui se fixent un ensemble de règles à suivre et qui les appliquent d'un bout à l'autre du projet. Bien souvent, chacun établit ses règles de manière plus ou moins arbitraire sans réelle réflexion : les singuliers sont mélangés aux pluriels, l'anglais au français, les traits d'union côtoient les tirets bas et les majuscules sont utilisées sans raison apparente.

Un bon moyen pour diminuer le risque d'erreurs est de mettre en place une nomenclature stricte que l'on applique à tous les composants du projet : dossiers, fichiers et sélecteurs. Cela évite les allers-retours inutiles pour vérifier si le nom de fichier est en anglais ou en français, ou pour savoir exactement à quels endroits placer les tirets de séparation. Il est possible que certains y trouvent peu d'intérêt et considèrent qu'une bonne logique personnelle suffit amplement. Mais c'est sans compter les personnes qui reprendront le projet une fois l'intégration terminée : il y a fort à parier qu'un déve-

loppeur ait à dynamiser le tout ou qu'une tierce personne soit en charge de la maintenance. En les prenant en considération, vous leur rendrez un fier service !

BONNE PRATIQUE Demander conseil au développeur

Suivant le système de gestion de contenus utilisé, il est fort probable qu'un certain nombre de classes, d'identifiants et d'usages soient prédéfinis. Demandez au développeur de vous les donner et prenez-en connaissance avant de figer vos règles de nommage. Vous lui éviterez ainsi une surcharge de travail et il vous en sera reconnaissant.

Exemples de règles de nommage

Une nomenclature n'a pas besoin d'être très verbeuse. Il s'agit de faire simple et compréhensible. Une liste d'affirmations accompagnées d'exemples telle que celle présentée ci-après est donc largement suffisante.

- Employer des noms anglais pour tous les éléments, car la v2 sera multilingue : `.alert` au lieu de `.alerte`.
- Indiquer tous les noms au singulier : `.slide` au lieu de `.slides`.
- Utiliser uniquement le trait d'union - : `.open-popin` au lieu de `.open_popin`.
- Donner des noms pertinents aux éléments : `#event` au lieu de `#block-block-1`.
- Utiliser les classes et identifiants comme noms d'images : `body.jpg` au lieu de `fond.jpg`.
- Privilégier les intitulés courts : `#slideshow-control` au lieu de `#playpause_slideshow_slider_block-3`.
- Différencier le premier élément d'une liste plutôt que le dernier, car c'est plus facile à programmer que l'inverse.

À préférer

```
<ul>
  <li class="first">Premier item</li>
  <li>Deuxième item</li>
  <li>Troisième item</li>
</ul>
```

À éviter

```
<ul>
  <li>Premier item</li>
  <li>Deuxième item</li>
  <li class="last">Troisième item</li>
</ul>
```

- Classes et identifiants à employer :
  - `#breadcrumb` pour le fil d'Ariane ;
  - `#search` pour le formulaire de recherche générique ;
  - `.submit` pour les éléments de soumission de formulaire ;
  - `.visually-hidden` pour les éléments visibles aux seules technologies d'assistance.

Où placer la convention de nommage ?

Une fois la convention de nommage établie, placez-la à la racine du dossier projet afin que chaque intervenant puisse en prendre connaissance et se l'approprier. En la rendant ainsi accessible, vous augmenter vos chances d'avoir un projet uniforme de bout en bout, et ce, quel que ce soit le nombre de personnes prenant part au projet.

QUALITÉ : METTRE EN PLACE UNE CHECKLIST DÉTAILLÉE

La *checklist* permet de vérifier méthodiquement certains points récurrents à prendre en compte lors de l'intégration. Bien qu'elle ne puisse pas garantir la qualité d'un site à 100 %, elle y contribue fortement. Le rendu navigateur, l'accessibilité, la conformité et la performance y trouvent naturellement leur place, mais vous pouvez bien sûr y ajouter d'autres paramètres spécifiques au projet, comme le détail des fonctionnalités à implémenter.

La liste proposée ici est le résultat d'un recoupement de trois listes distinctes : les bonnes pratiques en intégration d'Opquast (voir la section « Les bonnes pratiques Opquast » dans le chapitre suivant), « la checklist ultime pour le lancement d'un site web » de Dan Zambonini (Box UK) et « la checklist d'accessibilité que je m'étais juré de ne pas rédiger » d'Aaron Cannon (North Temple).

Ressources Les checklists

Opquast Checklists : http://bp.inté.net/1-1[1]

Box UK (Dan Zambonini) : http://bp.inté.net/1-2[2]

North Temple (Aaron Cannon) : http://bp.inté.net/1-3[3]

Véritable référence, la checklist Opquast dédiée à l'intégration regroupe 57 bonnes pratiques âprement débattues et triées sur le volet. C'est donc sans surprise que vous en retrouverez la grande majorité dans la liste ci-après. Scindée en trois parties, elle répond à plusieurs objectifs : servir d'index aux gabarits déjà intégrés, permettre l'évaluation individuelle de chaque page et, enfin, permettre l'évaluation transversale d'un ensemble de points communs aux différents gabarits. Suivant le contexte, ce fichier peut avantageusement être remplacé par un outil en ligne (comme Opquast Reporting) qui se charge de vérifier automatiquement un ensemble de points qui ne nécessitent pas d'être validés manuellement.

1. https://checklists.opquast.com
2. http://www.boxuk.com/blog/the-ultimate-website-launch-checklist
3. http://northtemple.com/1608

Index des maquettes

Toutes les pages à intégrer sont listées et associées à un numéro, afin qu'elles soient identifiables rapidement. Les URL sont par la suite progressivement ajoutées, permettant au fichier de servir d'index aux maquettes.

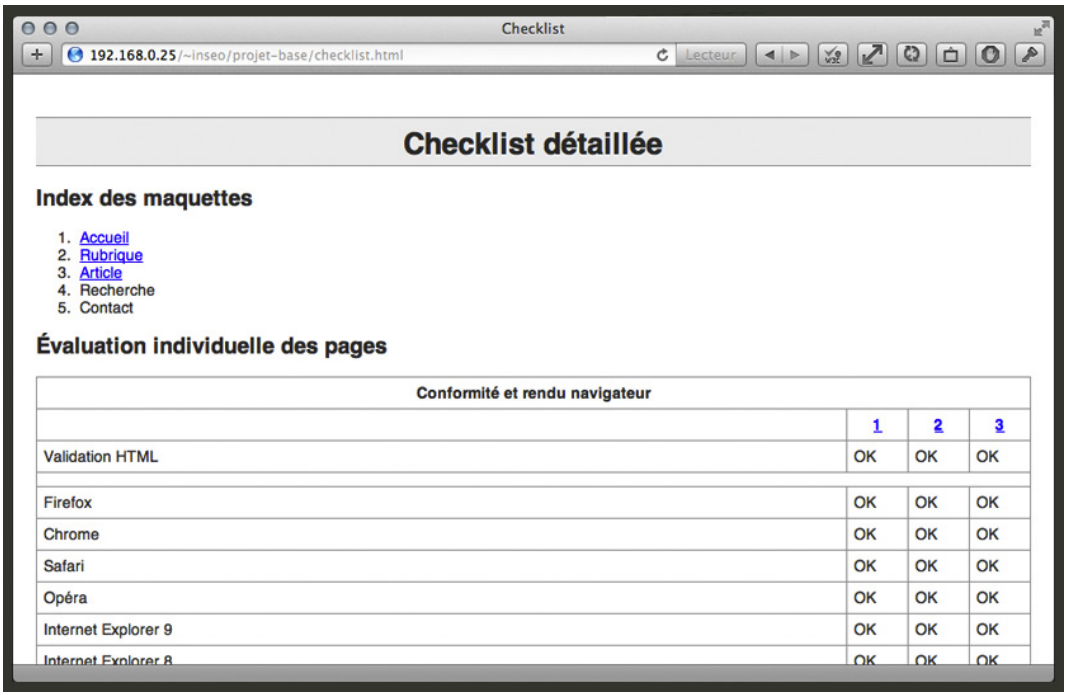


Fig. 1-2 : La checklist sert d'index aux maquettes et renseigne sur l'état d'avancement du projet.

Évaluation individuelle des pages

On regroupe ici l'ensemble des vérifications à faire de manière individuelle pour chaque page. Il s'agit sans conteste de la partie la plus importante de la checklist. En pratique, il s'agit d'indiquer pour chaque ligne si la vérification est un succès (OK), un échec (NOK) ou si elle n'est pas applicable dans le cadre du projet (N/A).

Les points à contrôler pour chaque page étant nombreux et variés, ils ont été regroupés par thématique. Cette partie de l'évaluation vous amènera donc à vérifier la conformité de vos pages et leur rendu sur les navigateurs visés, la qualité du code HTML, la structure et l'harmonisation du contenu, son accessibilité, l'optimisation pour le référencement et bien sûr le caractère fonctionnel du site.

Tableau 1-1 : Checklist d'évaluation individuelle des pages (version détaillée : https://github.com/inseo/bpi-checklist)

CONFORMITÉ ET RENDU NAVIGATEUR	1	2	3
Validation HTML			
Firefox			
Chrome			
Safari			
Opera			
Internet Explorer 9			
Internet Explorer 8			
Internet Explorer 7			
Internet Explorer 6			
iPhone			
iPad			
Android			
Blackberry			
Opera Mobile			

HTML	1	2	3
Le code source débute par un `doctype` dont la syntaxe est conforme aux recommandations du W3C.			
Une métadonnée définit le jeu de caractères de la page.			
Le jeu de caractères utilisé est UTF-8.			
Le code source ne contient ni éléments ni attributs de présentation.			
La capitalisation à des fins décorative est effectuée en CSS.			
Les mots ne comportent ni espaces ni balisage interne (`<span>L</span>ettrine`).			
Les styles en ligne sont utilisés de manière appropriée.			

Chaque lien est doté d'un intitulé dans le code source (entre les balises ouvrantes et fermantes de l'élément `<a>` ou, si nécessaire, via l'alternative textuelle d'un élément `<img>` ou `<object>`).			
Les icônes de navigation sont accompagnées d'une légende explicite (`alt` pour les éléments `<img>` et `title` pour les éléments `<a>`, `<button>` ou `<input>`).			

CONTENU			
	1	**2**	**3**
Aucun contenu de test ne subsiste (*lorem ipsum*, images temporaires).			
Le contenu est exempt de fautes d'orthographe et de grammaire.			
Il n'y a pas plusieurs variantes d'un même mot (*mail* ou *e-mail* ou *courriel*).			
Le ton employé est uniforme (*tu* ou *vous*).			
Les formules passe-partout (*en savoir plus*, *cliquez ici*) sont remplacées par des intitulés utiles.			
Les textes qui ne sont pas visibles par défaut (attributs `alt`, textes affichés via une fonction JavaScript, retranscriptions) sont pertinents.			
Les éléments de listes sont conclus de manières identiques (virgule, point ou point-virgule).			
Les entités HTML sont utilisées à bon escient (`&hellip;` pour …, `&laquo;` pour «, `&copy;` pour ©).			
Le contenu ne comporte pas de tableaux imbriqués.			
Les fenêtres modales sont dotées d'un bouton de fermeture explicite.			
Les images ne sont pas redimensionnées côté client.			
Chaque champ de formulaire est associé à une étiquette qui lui est propre (balise `<label>`).			
L'étiquette de chaque champ indique si la saisie est obligatoire (via un texte ou une image).			
Les listes d'options (balise `<select>`) sont présentées dans un ordre facilement identifiable.			
Les champs de formulaires traitant d'un sujet commun sont regroupés via `<fieldset>` sous un même intitulé (`<legend>`).			
Des liens d'aide ou des instructions en ligne facilitent la complétion des champs.			

Les captchas[4] sont remplacés par un test plus simple à effectuer.			
À la soumission, l'ensemble des erreurs de saisie sont indiquées à côté du champ concerné ou à un emplacement bien visible.			
L'utilisateur reçoit un feedback pour chaque action réalisée.			

ACCESSIBILITÉ			
	1	**2**	**3**
La langue principale du contenu est indiquée via l'attribut `lang` dans la balise `<html>`.			
Les passages en langage secondaire sont encadrés par une balise qui précise ce dernier.			
Le sens de lecture du contenu est précisé lorsqu'il diffère de celui par défaut (`dir`).			
Des liens d'évitement sont placés au début du code source.			
La navigation au clavier s'effectue dans un ordre prévisible (`tabindex`).			
Les en-têtes des tableaux de données sont balisés (`<th>`) et chaque cellule est associée à son en-tête.			
Les sons et vidéos sont déclenchés par l'utilisateur.			
Les animations ne bloquent pas la navigation ou l'accès aux contenus.			
Les animations, sons et clignotements peuvent être mis en pause.			
Aucun contenu ne clignote plus de trois fois par seconde.			
Une retranscription, un sous-titrage ou une traduction en langue des signes est disponible pour tous les sons et vidéos contenant un discours.			
Chaque vidéo transmettant une information essentielle sous forme visuelle est également disponible en audio description.			
Le contraste entre le contenu et le fond de la page est suffisant (se référer aux WCAG[5]).			
La couleur ne doit pas être le seul vecteur d'information (saisie obligatoire dans un formulaire, par exemple).			

4. Tests destinés à différencier un utilisateur lambda d'un robot, les captchas sont généralement utilisés dans les formulaires pour se prémunir d'une soumission massive effectuée par certains programmes malveillants.
5. Les WCAG *(Web Content Accessibility Guidelines)* sont formées de recommandations destinées à rendre les contenus web plus accessibles.