LIVRE BLANC HTML EMAIL

Table des matières

Présentation

Badsender est né en 2010 sous l'impulsion de Jonathan Loriaux et était alors un blog dédié à l'emailing[1]. Au fil du temps, l'audience du blog a pris de l'ampleur et Badsender est devenu l'activité principale de Jonathan, comme rédacteur et comme consultant indépendant. En 2014, Badsender s'est peu à peu transformé en activité d'**agence fullservice spécialisée dans l'emailing[2] et plus largement dans la mise en place de campagnes eCRM**. Actuellement, nous travaillons pour des annonceurs tels que Lagardère, La Fédération Française de Tennis, Viadeo …

Notre mission est d'apporter une expertise pointue dans notre domaine et d'être considérés comme une extension des équipes internes.

Qui parle ?

Thomas Defossez[3], Email designer et spécialiste de l'intégration HTML pour l'email : Thomas a démarré sa carrière en tant qu'intégrateur emailing chez Experian avant de créer sa propre agence web. Depuis 2013, Thomas collabore à divers projets de Badsender et depuis 2017 est devenu salarié de la structure. **Aujourd'hui, devenu le principal ambassadeur de l'email design et de l'intégration email chez Badsender.**

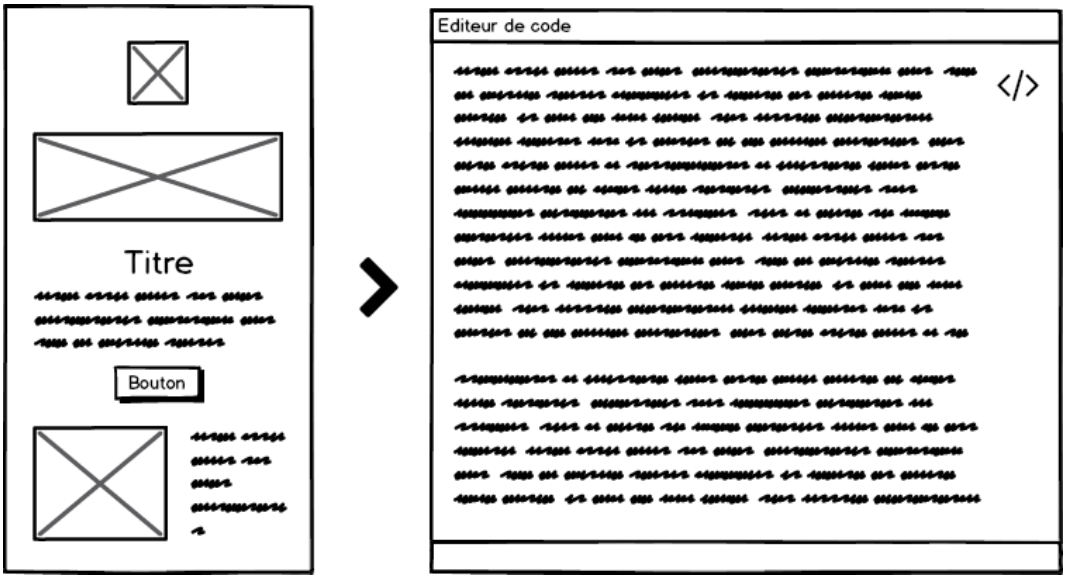
[1] *http://www.badsender.com/*
[2] *http://agence.badsender.com*
[3] *https://twitter.com/tde_badsender*

Partie I

Intégration HTML email

Introduction

L'intégration HTML d'un email[4] est la méthode par laquelle un intégrateur va **transformer une création graphique** *(au format PSD pour le fichier source, accompagné d'un fichier JPEG pour confirmation)* **en email au format HTML**.



Créé depuis Balsamiq V.3

A la question : *« Est-il plus complexe de coder un email ou une page web ? »*[5] Nous répondrons ceci : *« Cela dépend du point de vue. »* **Cela dépend aussi du développeur.** L'intégration des emails n'est finalement pas si complexe, en ce sens où elle consiste en du « bête et méchant ». Elle ne se limite qu'à la conception d'un seul fichier HTML, conçu via des tableaux imbriqués, avec des styles « en ligne », et un dossier d'images rattachées, quand le développement d'une page web fera appel à plusieurs fichiers, plusieurs types de langages, de multiples balises HTML et propriétés CSS, etc.

[4] *https://www.definitions-marketing.com/definition/integration-email-html/*
[5] *https://litmus.com/blog/the-tyranny-of-tables-why-web-and-email-design-are-so-different*

Créé depuis Balsamiq V.3

Les techniques sont radicalement différentes. Et c'est là où la construction en tableaux imbriqués rend les choses légèrement plus complexes pour le développeur lambda habitué à faire appel à des éléments sémantiques : , <section>, <p>, <hn>, <footer>, <header>, <nav>, etc… La conception sous forme d'HTML tabulaire *(tableaux imbriqués)* permet de garantir, en principe, un résultat identique sur l'ensemble des clients mails.

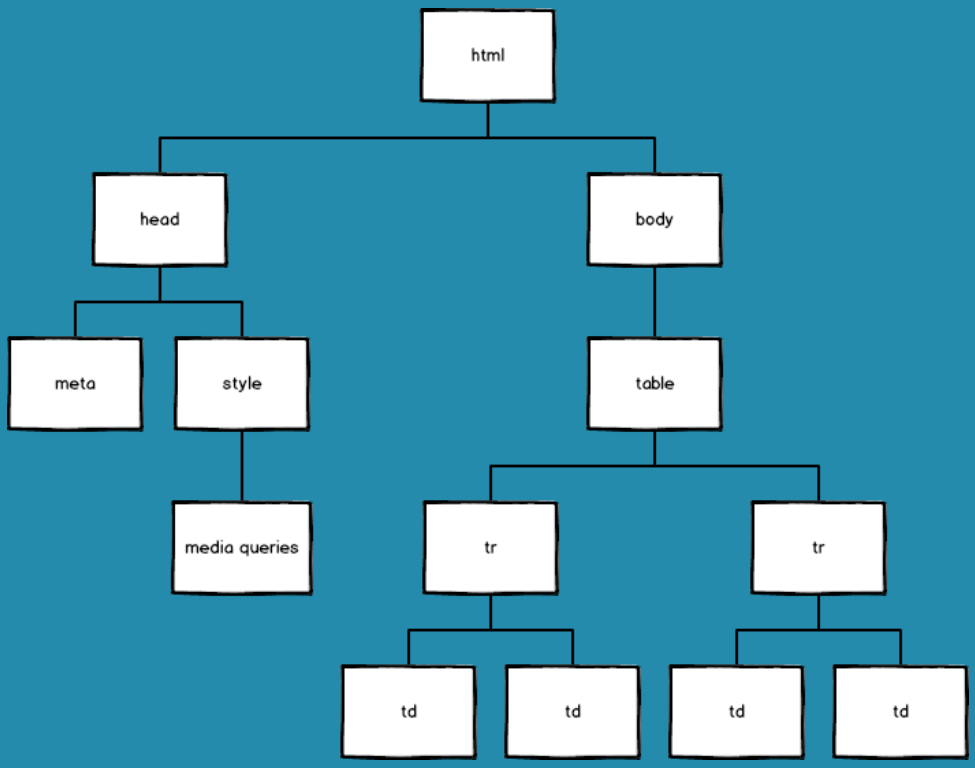
De plus, là où la conception d'une page web sera validée sur quelques navigateurs, un email sera, lui, testé sur bien plus de clients mails, possédant leur propre interprétation, leurs multiples versions, sur des webmails soumis, eux-aussi, aux aléas des navigateurs de consultation… Quand une page web est « testée » sur 4 ou 5 navigateurs *(Chrome, Firefox, Internet Explorer, Opera, Safari)* et leurs différentes résolutions, un email est visualisé sur Thunderbird, Outlook *(et ses multiples versions)*, Gmail, Yahoo Mail !, Free, Orange.fr, Office 365, Google App, Apple Mail, Lotus Notes, AOL, Free.fr, Outlook.com, mais aussi sur les « déclinaisons » de versions, de navigateurs, de résolutions… **Elle se différencie donc de l'intégration HTML des pages web car elle doit prendre en compte les difficultés d'interprétation du code par les interfaces de messagerie[6].** Car il faut savoir qu'il n'existe pas de conventions techniques standardisées au niveau international pour l'intégration HTML d'email[7] *(quand il existe le W3C, World Wide Web Consortium[8], pour les pages web)*. Mais qu'est-ce exactement que le HTML ?

[6] *https://www.campaignmonitor.com/css/*
[7] *https://www.lafabriquedunet.fr/email-marketing/guide/coder-email-html-css/*
[8] *https://fr.wikipedia.org/wiki/World_Wide_Web_Consortium*

HTML

INTRODUCTION

HTML[9] *(pour HyperText Markup Language)* est un langage de balisage qui permet d'encadrer des éléments présents dans une page *(images, titres, paragraphes)* par des balises pour leur permettre d'être mis en forme dans un second temps *(via le langage CSS)*. Il ne s'agit donc pas à proprement parler d'un langage de programmation. La visualisation d'un fichier HTML peut se faire sur un navigateur. **Les documents HTML ont une structure en arbre, avec un élément racine contenant tous les autres éléments.**



Créé depuis Balsamiq V.3

BALISES

Les balises HTML[10] permettent d'insérer ou de créer, au sein d'un document HTML, des éléments : textes, visuels, data … La liste est longue[11]. **Une balise pour créer un élément HTML[12] commence avec le sigle < et se termine par le symbole >.** Comme pour la création d'un élément, il sera aussi nécessaire de clôturer la création de cet élément. Pour ce faire, la balise de clôture doit commencer par le sigle </ et se terminer par le sigle >.

```
<td>Contenu</td>
```

[9] *http://www.comprendre-internet.com/Qu-est-ce-que-le-HTML.html*
[10] *https://www.w3schools.com/html/html_basic.asp*
[11] *https://www.w3schools.com/tags/ref_byfunc.asp*
[12] *https://www.w3schools.com/html/html_elements.asp*

Un élément HTML consiste donc la plupart du temps en une balise de début, et une balise de fin, avec le contenu inséré entre les deux. Quelques balises HTML sont autofermantes[13]. Dans ce cas, la syntaxe est la suivante :

```
<img src="http://www.chemin.fr/images/nomdelimage.jpg" />
```

Les balises HTML ne sont pas sensibles à la casse. Cependant, il est recommandé de les écrire en minuscules. Les éléments HTML peuvent être imbriqués *(des éléments contiennent d'autres éléments)*. **Tout document HTML consiste en une imbrication d'éléments HTML.**

```
<!DOCTYPE html>
<html>
<body>

<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td>Contenu</td>
  </tr>
</table>

</body>
</html>
```

Il est à noter que quelques balises ont des comportements spécifiques[14] : soit elles se comporteront comme **des éléments de type block**, ou comme des éléments **de type inline** :

- Un élément de type block commencera toujours sur une nouvelle ligne et prendra toute la largeur disponible dans la page. De plus, il pourra contenir d'autres éléments de type block, ainsi que des éléments de type inline.

- Au contraire des éléments de type block, les éléments de type inline ne vont pas commencer sur une nouvelle ligne, mais s'insérer dans la ligne actuelle. Ils prennent uniquement la largeur qui leur est nécessaire *(c'est-à-dire la largeur de leur contenu)*.

La connaissance du type d'un élément HTML et des différences de comportement entre les éléments de type inline ou block va être essentielle pour créer un email fonctionnel : **Il ne sera pas possible d'agir de la même manière en CSS pour manipuler les éléments de type block et de type inline.** Ces nuances de comportements[15] selon les balises vont être abordées ci-dessous.

[13] *http://xahlee.info/js/html5_non-closing_tag.html*
[14] *https://www.pierre-giraud.com/html-css/cours-complet/block-inline-html-css.php*
[15] *https://www.w3schools.com/html/html_blocks.asp*

<doctype>[16]

Doit être sur la première ligne du code HTML, et **doit commencer toute intégration HTML d'emailing.** Il ne s'agit pas à proprement parler d'une balise mais d'une déclaration, d'une instruction dédiée au navigateur sur la version HTML dans laquelle le code est écrit. Il existe plusieurs déclarations possibles[17] :

```
HTML 5 : <!DOCTYPE html>
HTML 4.01 Strict : <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
HTML 4.01 Transitional : <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
HTML 4.01 Frameset : <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">
XHTML 1.0 Strict : <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
XHTML 1.0 Transitional : <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
XHTML 1.0 Frameset : <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
XHTML 1.1 : <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

Selon la déclaration, des différences d'interprétation de code peuvent apparaître sur certains webmails[18] [19]. En vérité, dans le monde de l'email marketing, il faut distinguer plusieurs types de clients mails[20] : Ceux qui respectent le doctype indiqué, ceux qui utilisent un doctype HTML5, ceux qui n'ont pas de doctype, et les clients qui utilisent un autre doctype.

<html>[21]

Précise au navigateur qu'il s'agit d'un document HTML. Elle représente la racine de tout document HTML. **La balise <html> est le conteneur de tous les autres éléments HTML** *(exceptée la balise <!doctype>)*.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
</html>
```

[16] *https://www.w3schools.com/tags/tag_doctype.asp*
[17] *https://www.w3.org/QA/2002/04/valid-dtd-list.html*
[18] *https://www.campaignmonitor.com/blog/email-marketing/2010/11/correct-doctype-to-use-in-html-email/*
[19] *https://www.emailonacid.com/blog/article/email-development/doctype_-_the_black_sheep_of_html_email_design/*
[20] *https://emails.hteumeuleu.fr/2016/10/quel-doctype-faut-il-utiliser-dans-un-email/*
[21] *https://www.w3schools.com/tags/tag_html.asp*

<head>[22]

Peut contenir le titre du document, des scripts *(bannis dans l'emailing)*, des styles, des informations meta… Ces données pourront être spécifiées directement dans la balise <head> du document : <title>, <style>, <meta>, <link>. Il s'agit plutôt d'un élément structurel *(au même titre que les tags <html> ou <body>)*.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>

</head>

<body>
</body>
</html>
```

<meta>[23]

Permet de fournir des metadata[24] sur le document HTML. **Les metadata ne sont pas visibles sur la page et peuvent être utilisées par des navigateurs pour savoir, par exemple, comment afficher le contenu d'une page.** Ils permettent, entre autres, de prendre le contrôle sur le viewport (la partie visible d'une page), de spécifier le type de codage des caractères spéciaux, de permettre un comportement responsive sur certains supports nomades… Les <meta> sont toujours renseignés au sein de l'élément <head>. Voici un aperçu des <meta> que nous mettons généralement en place, chez Badsender, au sein de nos développements :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<!--[if !mso]><!-->
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<!--<![endif]-->
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="format-detection" content="telephone=no" />
<meta name="format-detection" content="date=no" />
<meta name="format-detection" content="address=no" />
<meta name="format-detection" content="email=no" />
</head>

<body>
</body>
</html>
```

[22] *https://www.w3schools.com/tags/tag_head.asp*
[23] *https://www.w3schools.com/tags/tag_meta.asp*
[24] *https://www.emailonacid.com/blog/article/email-development/demystifying-meta-tags-in-email/*

<style>[25]

Utilisée pour **définir des mises en forme « graphiques » au sein d'un document HTML.** A l'intérieur de cette balise, il est possible de spécifier la façon dont les éléments HTML s'afficheront. Un document HTML peut contenir plusieurs balises <style>. Elle est aussi primordiale pour la mise en place des media queries et du responsive.

Le style d'un élément HTML peut, en théorie, s'appliquer via trois méthodes :

- En incluant des règles CSS dans la balise HTML <style> de la section <head> du document HTML *(ce qui nous concerne dans ce chapitre)*.

```
<style>
  .cellule_titre {background-color:#FF0000; font-size:24px; font-
style:italic;}
  </style>
```

- En pointant vers un fichier CSS externe, contenant toutes les règles CSS.

```
<link rel="stylesheet" type="text/css" href="styleemail.css" />
```

- Ou en utilisant du style CSS inline.

```
<tr>
  <td style="font-family:Arial, Helvetica, sans-serif; font-size:14px;
color:#333333; text-align:left;">Contenu textuel.</td>
</tr>
```

[25] *https://www.w3schools.com/tags/tag_style.asp*

Dans le monde du code d'emailing, **la seconde méthode est à proscrire pour le moment.** La structure du fichier HTML ressemble donc à ceci :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<!--[if !mso]><!-->
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<!--<![endif]-->
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="format-detection" content="telephone=no" />
<meta name="format-detection" content="date=no" />
<meta name="format-detection" content="address=no" />
<meta name="format-detection" content="email=no" />
<style type="text/css">
body {
    margin:0px auto !important;
    padding:0px;
}
</style>
</head>

<body>
</body>
</html>
```

Créé depuis Balsamiq V.3

<body>[26]

Définit le corps du document HTML. La partie « visible » d'un document HTML est contenu entre les balises <body> et </body>. L'élément contient tous les contenus d'un document HTML : textes, visuels, liens, tableaux, lignes, etc… **C'est dans cet élément que la construction en tableaux imbriqués peut commencer.**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<!--[if !mso]><!-->
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<!--<![endif]-->
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="format-detection" content="telephone=no" />
<meta name="format-detection" content="date=no" />
<meta name="format-detection" content="address=no" />
<meta name="format-detection" content="email=no" />
<style type="text/css">
body {
    margin:0px auto !important;
    padding:0px;
}
</style>
</head>

<body>
        <table cellpadding="0" cellspacing="0" border="0" width="100%"
style="width:100%; margin:0px auto;" align="center">
            <tr><td></td></tr>
        </table>
</body>
</html>
```

<table>[27]

Une balise de référence pour l'intégrateur emailing, c'est une de ses meilleures amies. Un tableau HTML se construit à l'aide de cette balise. Lorsqu'on parle de tableaux imbriqués, on pense automatiquement à cette balise. Un tableau est constitué de ligne(s) *(<tr>)*, composée(s) elle(s)-même(s) de cellule(s) data *(<td>)* ou de cellule(s) titre(s) *(<th>)*. Concrètement, un tableau HTML

[26] *https://www.w3schools.com/tags/tag_body.asp*
[27] *https://www.w3schools.com/tags/tag_table.asp*

pour emailing est structuré ainsi :

```
<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td> </td>
  </tr>
</table>
```

Un tableau HTML pourrait aussi contenir des éléments HTML comme <caption>, <col>, <colgroup>, <thead>, <tfoot>, <tbody>. Mais **un tableau HTML pour un email restera le plus simple possible**, pour garantir une interprétation et un rendu communs sur l'ensemble des clients mail.

<tr>[28]

<tr> *(pour Table Row)* définit simplement **une ligne dans un tableau**. Peut contenir un ou plusieurs éléments <td> ou <th>. C'est certainement l'élément sur lequel le moins d'attributs HTML et de styles CSS seront spécifiés. Le plus souvent, dans une intégration HTML d'emailing, les balises <tr> restent vierges.

<td>[29] *et <th>*[30]

Après la balise <table>, la seconde meilleure amie de l'intégrateur emailing : **définit une cellule dans un tableau HTML.** Elle peut se présenter sous la forme d'une cellule standard *(<td> pour Table Data)*, ou sous la forme d'une cellule « titre » *(<th> pour Table Header)*.

Si en SEO ces deux balises ont une distinction sémantique, elles ont surtout dans l'emailing une variation technique. En effet, la propriété CSS display :block attribuée à une <td> n'est pas correctement supportée sur mobile Android depuis Android 4.4 KitKat[31]. C'est dans ce cas bien précis qu'un intégrateur emailing utilisera un élément <th> à la place d'une <td>.

Tips 'N Tricks : il est à savoir qu'une <th> a une mise en forme par défaut sur certains clients mails et navigateurs. Il est donc nécessaire de lui appliquer un « reset » CSS en lui spécifiant un style= « font-weight :normal ; padding :0px ; » : ainsi, les textes ne seront pas automatiquement en gras, et aucune marge interne ne sera présente.

Une cellule peut contenir aussi bien des données textuelles (texte brut) que des données visuelles (images, visuels, gifs animés). Finalement, **ce sont les cellules qui contiennent la plupart du temps l'ensemble du contenu d'un email.** Il est très rare de renseigner des données de contenu en dehors des cellules. Ainsi, une cellule ressemblera dans sa version finale pour des données textuelles, à ceci :

```
<tr>
    <td style="font-family:Arial, Helvetica, sans-serif; font-size:12px;
color:#000000; text-align:left; line-height:18px; padding:15px;"
align="left">Contenu textuel.</td>
  </tr>
```

[28] *https://www.w3schools.com/tags/tag_tr.asp*
[29] *https://www.w3schools.com/tags/tag_td.asp*
[30] *https://www.w3schools.com/tags/tag_th.asp*
[31] *https://litmus.com/community/discussions/261-td-display-block-not-working-on-android*

Les balises <table>, <tr>, <td> doivent donc s'ouvrir, puis être refermées : il ne s'agit donc pas de balises autofermantes[32]. Tout oubli de fermeture de balise pourrait entraîner un problème de rendu *(même si certains clients mails ou navigateurs peuvent parfois corriger ces oublis)*.

<img>[33]

<img> *(pour image)* permet d'insérer un visuel au sein d'un document HTML. Cette balise requiert principalement deux attributs obligatoires : src et alt. Il est à noter que les visuels ne sont pas, techniquement, insérés dans le document. **Ils sont plutôt appelés via cette balise.** <img> crée donc un espace réservé à l'image appelée. L'appel à l'image en question se fait via l'attribut src.

Plusieurs formats d'image sont envisageables dans un email : jpeg, gif, ou png.
Chacun de ces formats a ses qualités… Et ses défauts[34] :

	JPEG	GIF	PNG
AVANTAGES	Parfait pour les dégradés, il permet aussi de **régler la qualité de la compression de l'image.** Ce format peut afficher plusieurs millions de couleurs. Il est idéal pour toute photo ou visuel de produit.	Permet la transparence, et permet aussi de choisir la palette de couleurs. Mieux encore, ce type de format permet de générer des gifs dits « animés ».	Permet la transparence, **tout en gardant une bonne qualité d'image.** Le PNG-8 est très similaire au format gif, puisqu'il supporte 256 couleurs et permet la transparence, et peut même s'avérer plus léger qu'un gif. Le PNG-24 est similaire au jpeg, puisqu'il peut comprendre jusqu'à 16 millions de couleurs.
INCONVENIENTS	Ne permet pas la transparence et peut s'avérer **relativement lourd** sans compression.	Ce format n'est pas très bon dans les dégradés. De plus, il ne supporte qu'un maximum de 256 couleurs. Il est en général utilisé pour les textes, pictogrammes, ou logos.	Il fut un temps mal supporté par quelques clients mails. Son poids peut parfois s'avérer un peu trop lourd.

[32] *http://xhtml.le-developpeur-web.com/balise_autofermante-xhtml.php*
[33] *https://www.w3schools.com/tags/tag_img.asp*
[34] *https://www.whoishostingthis.com/blog/2014/12/06/jpeg-gif-png/*

<a>[35]

Indispensable au sein d'une intégration HTML d'email, la balise <a> *(pour anchor)* **crée un hyperlien** qui sera utilisé entre autres pour lier un élément HTML à une page Web. L'attribut indispensable à la balise <a> se nomme href. Cet attribut permet d'indiquer la destination ou la fonctionnalité du lien. Sans attribut href, la balise <a> n'est plus un hyperlien, et pourrait servir d'ancre au sein d'un document HTML *(fonctionnalité à proscrire, nous verrons pourquoi par la suite)*.

**[36]

 (pour « couvrir ») **permet de regrouper plusieurs éléments** inline, des parties et morceaux de textes, **afin d'y attribuer un code CSS particulier.** Si au sein d'un paragraphe un groupe de mots doit s'afficher en gras, le code sera le suivant :

```
<td>Phrase dans laquelle <span style="font-weight:bold;">un groupe de mots
s'affichera en gras</span> sans impact sur le reste du texte.</td>
```

*
*[37]

 (pour line break) permet de créer un retour à la ligne. Très utile dans l'intégration HTML d'email, où l'utilisation des balises <p> est à éviter.

```
<td>Portion de texte 01.<br />
    Portion de texte 02 après retour à la ligne.</td>
```

* *[38]

Dans un document HTML, une *(pour division)* **définit une division ou une section.** L'élément est souvent utilisé comme un conteneur pour d'autres éléments HTML. Dans l'emailing, son utilisation est très périlleuse. Cette balise a une mise en forme *(donc, des propriétés CSS)* par défaut sur certains clients mails et navigateurs et ses propriétés de positionnement et de mise en forme *(float, margin)* ne sont pas toujours bien supportées. Elle ne doit être utilisée, à notre sens, que dans des cas bien particuliers :

- Pour la mise en place de boutons entièrement cliquables *(via l'outil buttons.cm*[39]*)*
- Pour la mise en place d'images de fond *(via l'outil backgrounds.cm*[40]*)*
- Pour la construction d'emails responsive sans media queries *(Emails Fluid*[41]*)*

[35] *https://www.w3schools.com/tags/tag_a.asp*
[36] *https://www.w3schools.com/tags/tag_span.asp*
[37] *https://www.w3schools.com/tags/tag_br.asp*
[38] *https://www.w3schools.com/tags/tag_div.asp*
[39] *https://buttons.cm/*
[40] *https://backgrounds.cm/*
[41] *https://webdesign.tutsplus.com/tutorials/creating-a-future-proof-responsive-email-without-media-queries--cms-23919*

<p>[42]

<p> *(pour paragraph)* peut être utilisée dans un email, mais il faut alors **prendre en compte sa mise en forme graphique automatique** par certains clients mails *(et donc appliquer un reset CSS à cet élément)*. Cette balise a un intérêt sémantique, mais il est préférable de renseigner les textes simplement en HTML brut, au sein des cellules directement.

<hn>[43]

Comme la balise <p>, <hn> *(pour heading, valeurs de <h1> à <h6>)* peut être utilisée dans un email, mais il faut alors **prendre en compte sa mise en forme graphique automatique** par certains clients mails *(et donc appliquer un reset CSS à ces éléments)*. Ces balises ont un intérêt sémantique, mais il est préférable de renseigner les textes simplement en HTML brut, au sein des cellules directement.

Commentaires[44]

Fonctionnalité vitale pour tout bon intégrateur d'email, **elle permet d'insérer des commentaires au sein d'un code HTML.** Cela autorise surtout le développeur à démarquer rapidement les différents modules de son intégration, afin de s'orienter plus rapidement et aisément en cas de mise à jour ou de débogage, mais aussi à expliquer ou compléter son code, ce qui peut aider considérablement à la compréhension du code lors d'une relecture. Les commentaires HTMLs commencent par < !-- et se terminent par -->. Tout code compris entre ces deux éléments ne sera pas affiché à l'écran.

```
<!-- DEBUT MODULE ARTICLE PRINCIPAL -->

<table cellpadding="0" cellspacing="0" border="0" width="600"
style="width:600px; margin:0px auto;" align="center" class="width300px">…
</table>

<!-- FIN MODULE ARTICLE PRINCIPAL -->
```

[42] *https://www.w3schools.com/tags/tag_p.asp*
[43] *https://www.w3schools.com/tags/tag_hn.asp*
[44] *https://www.w3schools.com/tags/tag_comment.asp*

ATTRIBUTS

Chaque élément HTML **peut avoir un ou plusieurs attributs.** Ces attributs sont des valeurs supplémentaires qui permettent de configurer les éléments ou d'adapter leur comportement. La liste suivante n'est pas exhaustive mais reprend les attributs HTML les plus couramment utilisés dans le codage d'email.

Alt[45]

Certains clients mails bloquent par défaut les images[46]. Parfois, les destinataires peuvent changer leurs paramètres de messagerie pour bloquer l'affichage des images. **Lorsque les images ne sont pas affichées, le contenu de l'attribut alt** *(pour texte alternatif)* **apparaît.**

Capture d'écran issue du blog Litmus

Ce texte offre une alternative textuelle au contenu non textuel d'un email[47]. Si les images ne s'affichent pas, convenons que **cela peut réduire considérablement la compréhension de l'email**[48] par les destinataires. Il est donc très important de bien renseigner l'attribut alt, et d'optimiser sa mise en forme graphique[49], pour garantir un rendu optimal de l'email lorsque les images ne sont pas affichées. Gardons cependant en tête que le style ne sera pas appliqué dans tous les clients mails. Cela sera possible en indiquant quelques propriétés CSS via l'attribut style de l'image.

```
<img src="images/nomdelimage.jpg" alt="Texte alternatif" style="display:block;
font-family:Arial, Helvetica, sans-serif; color:#333333; font-size:16px;
margin:0px auto;" />
```

Capture d'écran issue du blog Litmus

[45] *https://www.w3schools.com/tags/att_img_alt.asp*
[46] *https://emails.hteumeuleu.com/how-webmails-block-images-4cc2ba4d2d0e*
[47] *http://www.pompage.net/traduction/Bien-utiliser-le-texte-alternatif*
[48] *https://www.campaignmonitor.com/resources/guides/alt-text-in-email/*
[49] *https://litmus.com/blog/the-ultimate-guide-to-styled-alt-text-in-email*

Le texte alternatif a aussi une fonction d'accessibilité : il est annoncé par les logiciels de revue d'écran, rendant le contenu ou la fonction de l'image accessibles aux utilisateurs ayant des troubles visuels ou cognitifs. Sans texte alternatif, les logiciels de revue d'écran indiquent parfois les autres informations disponibles de l'image *(nom du fichier, dimensions)*.

Il faut donc retenir, pour la rédaction du texte alternatif, que **son contenu devrait être précis et équivalent au contenu et aux fonctions de l'image.** Être court : en quelques mots, aussi peu que possible. Ne pas être redondant avec d'autres textes alternatifs ou d'autres informations textuelles de l'email. Ne pas contenir « image de… » ou « graphisme de… ». Si un lien est présent sur l'image, le texte alternatif peut résumer la fonction présente sur le lien, en rapport avec l'image.

Lorsqu'il s'agit d'une image décorative, un attribut alt vide peut suffire. Lorsqu'il s'agit d'une grande image découpée, il est préférable de ne renseigner qu'une seule fois le texte alternatif dans l'image la plus grande, et de ne pas le répéter pour chaque morceau d'image. Les images d'arrière-plan *(background-image)* ne peuvent avoir de texte alternatif. Si le contenu d'une image est important, l'image ne doit pas être appliquée en arrière-plan. Finalement, il est à retenir que le texte alternatif se choisit en fonction du contexte.

Src[50]

Src *(pour source)* est **obligatoirement requis pour spécifier le chemin de l'image appelée.** Il va donc de pair avec la balise <img>. La syntaxe de cet attribut est la suivante : src= « urldelimage ». Dans une intégration HTML locale, les chemins seront relatifs : ils pointeront vers un dossier où se trouvent les images.

```
<img src="images/produit_01_visuel.jpg" />
```

Une fois les images hébergées sur un serveur, les chemins des visuels seront dits absolus *(ou urls absolues)*.

```
<img src="http://www.badsender.com/images/produit_01_visuel.jpg" />
```

Href[51]

Href *(pour Hypertext REFerence)* spécifie la destination du lien. Généralement, il s'agit d'une url *(de type http:///)*, mais cette url peut aussi contenir des informations de tracking et prendre un aspect plus complexe. Enfin, il peut aussi s'agir de chemins plus « spécifiques » :

- Un lien vers un élément de la page ou de l'email avec une id précise. Cependant, cette fonctionnalité est à proscrire dans l'intégration HTML d'emailing *(nous verrons pourquoi par la suite)*.

- **Un autre protocole :** mailto *(pour la rédaction d'un email)* ou tel *(pour générer un appel vers un numéro précis)*. Dans le cas d'un mailto, l'attribut href prendra la syntaxe suivante :

[50] *https://www.w3schools.com/tags/att_img_src.asp*
[51] *https://www.w3schools.com/tags/att_a_href.asp*

```
<a
href="mailto:someone@example.com?cc=someoneelse@example.com&bcc=andsome
oneelse@example.com
&subject=Summer%20Party&body=You%20are%20invited%20to%20a%20big%20summe
r%20party!">Send mail!</a>
```

Dans ce cas précis, plusieurs champs ont été spécifiés dans le protocole mailto : les adresses mail en copie *(via les variables cc)*, le sujet du mail *(via la variable subject)* et enfin, le contenu du mail *(via la variable body)*.

Dans le cas du protocole tel, la syntaxe sera la suivante :

```
<a href="tel:+33344547301">Appelez-nous !</a>
```

Title[52]

Permet d'ajouter de l'information supplémentaire sur un élément. Cet attribut peut être, dans un emailing, mis en place sur les visuels par exemple, pour proposer aux destinataires, sur les navigateurs et webmails qui le permettent, **l'apparition d'une infobulle au survol de l'image.** Cependant, il peut être utilisé sur n'importe quel élément HTML.

Width[53]

Spécifie la largeur d'un élément, en pixels. L'attribut width est très important : il sera, la plupart du temps, attribué à des tableaux, à des cellules, ou à des visuels. Cet attribut est souvent renseigné en pixels *(dans ce cas, on ne précise pas l'unité)*. Il pourra aussi être spécifié en pourcentage *(dans ce cas, on précise l'unité pourcentage)*.

```
<table width="400" border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td> </td>
    </tr>
  </table>
```

Il a donc pour vocation de **préciser la largeur d'un élément HTML.** Il est possible de ne pas toujours le renseigner sur une cellule ou sur un tableau, pour laisser l'élément prendre la largeur de son contenu. Avec la croissance de l'utilisation des méthodes fluides et hybrides, les tableaux seront régulièrement renseignés en pourcentage, pour s'adapter à la résolution de la fenêtre de visualisation.

Height[54]

Spécifie la hauteur d'un élément, en pixels *(dans ce cas, on ne précise pas l'unité)*. L'attribut height sera, pour sa part, principalement utilisé sur les images et visuels de l'intégration HTML. Il est important de renseigner correctement cet attribut pour éviter tout désagrément de rendu sur quelques clients mails lorsque les images ne seront pas affichées *(excepté lorsque les visuels ont une largeur indiquée en pourcentage)*.

---

[52] *https://www.w3schools.com/tags/att_global_title.asp*
[53] *https://www.w3schools.com/tags/att_width.asp*
[54] *https://www.w3schools.com/tags/att_height.asp*

Si l'attribut width concerne aussi les cellules et tableaux, nous déconseillons la mise en place de hauteur fixe sur ces éléments : couplé à certaines propriétés CSS *(margin, padding)*, il peut être interprété différemment selon les clients mails. De plus, l'intégration d'un email doit être « flexible » et étirable, la structure doit pouvoir s'adapter à son contenu. **L'intégration doit respirer, garder un peu d'élasticité, pour s'adapter à son contenu.**

Cellpadding[55]

Définit l'espace *(en pixels)* entre le bord de la cellule et le contenu de la cellule. Par précaution, nous recommandons de toujours spécifier cet attribut à chaque <table>, et de renseigner sa valeur à 0. Les marges internes des cellules seront générées via la propriété CSS padding, car même si l'attribut cellpadding est correctement supporté par l'ensemble des clients mails, **il sera bien plus complexe d'en modifier la valeur en responsive**, contrairement à la propriété CSS padding.

Cellspacing[56]

Définit l'espace entre les cellules. Comme pour le cellpadding, nous recommandons d'indiquer cet attribut pour chaque <table> créé, et d'en renseigner la valeur à 0.

Border[57]

A ne pas confondre avec la propriété CSS border, cet attribut précise la largeur en pixels d'une bordure. Pour éviter tout problème de rendu, une bonne pratique consiste **à renseigner cet attribut sur toute image, et à spécifier sa valeur à 0**.

Bgcolor

Régulièrement utilisé dans l'intégration HTML d'un email, il permet de spécifier une couleur de fond *(bgcolor pour background-color)*. Sa valeur sera, la plupart du temps, **un code hexadécimal**. Cet attribut sera spécifié principalement sur des cellules ou tableaux, pour s'assurer de sa bonne interprétation par l'ensemble des clients mail[58].

Il existe plusieurs manières de préciser la couleur : celle-ci peut être en toutes lettres[59] *(red, yellow)*, en code rgb *(pour Red Green Blue, avec des valeurs allant de 0 à 255)*, en code rgba *(pour Red Green Blue et Alpha, pour la transparence)*, hsl *(pour Hue Saturation Lightness, Teinte Saturation Luminosité)*, hsla ou en code hexadécimal *(HEX value)*.

Le code hexadécimal, le plus utilisé sans doute car le mieux supporté, est composé de six chiffres. Chaque paire de caractères dans le code Hex représentant l'intensité de rouge, vert et bleu dans la couleur, respectivement. La valeur d'une paire de caractères varie de 00 *(la plus faible intensité)* à FF *(qui représente l'intensité la plus élevée)*. La couleur blanche par exemple est faite par chacune des

---

[55] *https://www.w3schools.com/tags/att_table_cellpadding.asp*
[56] *https://www.w3schools.com/tags/att_table_cellspacing.asp*
[57] *https://www.w3schools.com/tags/att_img_border.asp*
[58] *https://litmus.com/blog/background-colors-html-email*
[59] *https://htmlcolorcodes.com/fr/noms-de-couleur/*

trois couleurs primaires, à leur pleine intensité, ayant pour résultat le code couleur Hex #FFFFFF. Le code Hex n'est pas sensible à la casse.

```
<table width="400" style="width:400px; margin:0px auto;" align="center"
border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td bgcolor="#F2F2F2" style="padding:20px; font-family:Arial, Helvetica,
sans-serif; color:#000000; font-size:14px; line-height:1.4; text-align:left;"
align="left">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras
consequat egestas velit sit amet posuere. In hac habitasse platea
dictumst.</td>
    </tr>
  </table>
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras consequat egestas velit sit amet posuere. In hac habitasse platea dictumst.

Valign[60]

Permet de renseigner l'alignement vertical au sein d'une cellule *(<td>)*. Il peut prendre plusieurs valeurs : top, middle, bottom, baseline. Son utilisation sera particulièrement remarquée dans le cas ou deux cellules *(ou plus)* d'une même ligne contiennent des contenus d'hauteurs différentes *(ou variables)* et qu'il faudra décider du positionnement vertical de ces données.

```
<table width="400" style="width:400px; margin:0px auto;" align="center"
border="0" cellspacing="0" cellpadding="15">
    <tr>
      <td valign="top" bgcolor="#F2F2F2" style="font-family:Arial, Helvetica,
sans-serif; color:#000000; font-size:14px; line-height:1.4; text-align:left;"
width="50%" align="left">Lorem ipsum dolor sit amet, consectetur adipiscing
elit. Cras consequat egestas velit sit amet posuere.</td>
      <td valign="top" bgcolor="#f9f9f9" style="font-family:Arial, Helvetica,
sans-serif; color:#000000; font-size:14px; line-height:1.4; text-align:left;"
width="50%" align="left">In hac habitasse platea dictumst.</td>
    </tr>
  </table>
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras consequat egestas velit sit amet posuere.	In hac habitasse platea dictumst.

[60] *https://www.w3schools.com/tags/att_td_valign.asp*

Align[61]

Permet d'aligner horizontalement des contenus *(à la base, au sein d'un élément)* ou des éléments de type block. Peut prendre plusieurs valeurs : left, right, center, justify... Selon sa valeur, le contenu de la cellule sera ferré à gauche, à droite, centré, ou justifié…

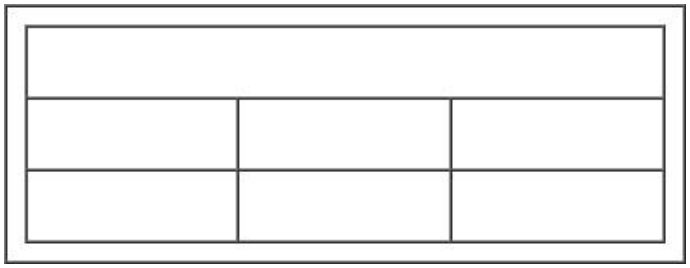
```
<tr>
  <td align="center">Ici, un contenu texte centré</td>
</tr>
```

Il est préférable, lorsqu'il s'agit d'un texte dans une cellule, de doubler cet attribut par un style= «text-align :[valeur] ; ». Si l'attribut est renseigné sur un tableau directement, il faudra doubler l'attribut par un style= « margin :0px auto ; » dans le cas où le tableau devrait être centré.

Colspan[62]

Colspan *(pour « column span »)* définit **le nombre de colonnes qu'une cellule devrait contenir**. Aucune unité ne doit être renseignée pour sa valeur. Cet attribut est nécessaire lorsqu'au sein d'un même tableau, une ligne ne contient qu'une seule cellule, alors que la ligne du dessous en contient trois par exemple. Dans ce cas de figure, la code se présentera alors ainsi :

```
<table width="100%" border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td colspan="3"> </td>
    </tr>
    <tr>
      <td> </td>
      <td> </td>
      <td> </td>
    </tr>
  </table>
```



L'utilisation de cet attribut est, si possible, à proscrire dans l'intégration HTML d'un email : il complexifie la structure et la compréhension du code, mais aussi son update ou les modifications qu'il faudrait apporter sur la structure de l'email. Il vaut mieux créer, dans ce cas de figure, de nouveaux tableaux :

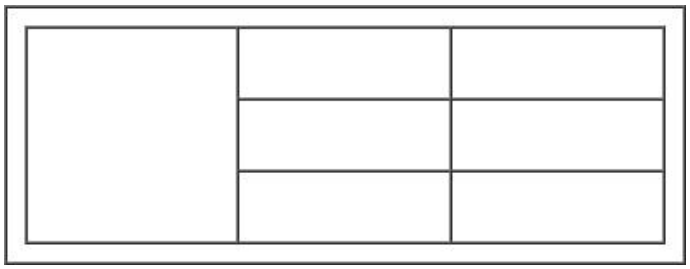
[61] *https://www.w3.org/TR/html401/present/graphics.html#h-15.1.2*
[62] *https://www.w3schools.com/tags/att_td_colspan.asp*

```
<table width="100%" border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td> </td>
    </tr>
    <tr>
      <td><table width="100%" border="0" cellspacing="0" cellpadding="0">
          <tr>
            <td> </td>
            <td> </td>
            <td> </td>
          </tr>
        </table></td>
    </tr>
  </table>
```

Rowspan[63]

Rowspan *(pour « row span »)* spécifie **le nombre de lignes qu'une cellule devrait contenir**. Aucune unité ne doit être renseignée pour sa valeur. Cet attribut est nécessaire lorsqu'au sein d'un même tableau, une colonne ne contient qu'une seule cellule, alors que les colonnes sœurs en contiennent trois par exemple. Dans ce cas de figure, la code se présentera alors ainsi :

```
<table width="100%" border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td rowspan="3"> </td>
      <td> </td>
      <td> </td>
    </tr>
    <tr>
      <td> </td>
      <td> </td>
    </tr>
    <tr>
      <td> </td>
      <td> </td>
    </tr>
  </table>
```



Comme pour l'attribut colspan, l'utilisation de cet attribut est, si possible, à proscrire dans l'intégration HTML d'un email : il complexifie la structure et la compréhension du code, mais aussi

[63] *https://www.w3schools.com/tags/att_td_rowspan.asp*

son update ou les modifications qu'il faudrait apporter sur la structure de l'email. Il vaut mieux créer, dans ce cas de figure, de nouveaux tableaux :

```
<table width="100%" border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td> </td>
      <td><table width="100%" border="0" cellspacing="0" cellpadding="0">
          <tr>
            <td> </td>
          </tr>
          <tr>
            <td> </td>
          </tr>
          <tr>
            <td> </td>
          </tr>
        </table></td>
      <td><table width="100%" border="0" cellspacing="0" cellpadding="0">
          <tr>
            <td> </td>
          </tr>
          <tr>
            <td> </td>
          </tr>
          <tr>
            <td> </td>
          </tr>
        </table></td>
    </tr>
  </table>
```

Target[64]

Précise « où » le lien contenu dans une balise <a> doit être ouvert. En effet, il existe plusieurs possibilités : le lien peut être ouvert dans une nouvelle fenêtre ou un nouvel onglet *(la valeur sera alors égale à « _blank »)*, dans le même onglet ou la même fenêtre *(comportement par défaut, la valeur sera alors égale à « _self »)*. Il s'agit ici des valeurs généralement utilisées. **Dans une intégration HTML d'email, pour optimiser l'ergonomie de l'email et ne pas forcer le destinataire à quitter son contenu, la valeur « _blank » est généralement renseignée.**

[64] *https://www.w3schools.com/tags/att_a_target.asp*

Style[65]

Spécifie un style « inline » pour un élément HTML. Cet attribut outrepassera tous les autres styles, spécifiés par exemple dans la balise <style> ou dans un fichier CSS externe.

```
<tr>
  <td style="font-family:Arial, Helvetica, sans-serif; font-size:14px;
color:#333333; text-align:left;">Mon contenu</td>
</tr>
```

Id[66]

Correspond à un identifiant unique pour un élément HTML. Sa valeur doit donc, elle-aussi, être unique au sein d'un document HTML. Il ne peut y avoir deux ids identiques *(sur leur valeur)* dans un même document HTML. Cet attribut est principalement utilisé pour spécifier un style bien précis à un élément, et en particulier lors de l'utilisation des media queries : cela permet de cibler, dans les media queries, l'élément unique de l'intégration HTML concerné par les modifications CSS à effectuer.

```
<table width="100%" border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td style="font-family:Arial, Helvetica, sans-serif; color:#000000;
font-size:12px; padding-top:20px;" id="titre">Mon contenu</td>
    </tr>
  </table>
```

Class[67]

A, finalement, la même fonction dans l'intégration HTML d'emailing que l'attribut id, à savoir : pointer un élément pour lui attribuer des styles CSS bien précis. A la différence près que **l'attribut class n'a pas besoin d'être unique**. Ainsi, Il sera tout à fait possible de renseigner plusieurs class à un seul et même élément HTML. Par exemple :

```
<table width="100%" border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td style="font-family:Arial, Helvetica, sans-serif; color:#000000;
font-size:12px; padding-top:20px;" class="fontsize14px paddingtop10px">Mon
contenu</td>
    </tr>
  </table>
```

Via l'attribut class, nous pouvons ainsi spécifier plusieurs propriétés CSS que nous pourrons par la suite définir dans nos media queries. Il est aussi possible d'appliquer la même class sur de multiples éléments HTML au sein d'un même document.

[65] *https://www.w3schools.com/tags/att_global_style.asp*
[66] *https://www.w3schools.com/tags/att_global_id.asp*
[67] *https://www.w3schools.com/tags/att_global_class.asp*

Dir[68]

Spécifie la direction du texte ou du contenu d'un élément HTML. Cet attribut peut être utilisé sur n'importe quel élément HTML. Ses valeurs peuvent être les suivantes : ltr *(pour left to right)*, rtl *(pour right to left)* et auto. **Cet attribut sera particulièrement utile en responsive, pour modifier « l'ordre » de modules** *(<table> ou <td>)*.

[68] *https://www.w3schools.com/tags/att_global_dir.asp*

CSS

INTRODUCTION

CSS[69] pour Cascading Style Sheets *(feuilles de style en cascade[70])*. Le CSS est un langage informatique qui décrit **comment les éléments HTML doivent être affichés** à l'écran, sur papier ou sur d'autres supports. Les standards définissant le CSS sont publiés par le World Wide Web Consortium *(W3C)*. Contrairement aux logiciels, les spécifications CSS ne sont pas développées par versions successives, qui permettraient à un navigateur de se référer à une version en particulier. CSS est développé par « niveaux », ce qui contraint chaque nouveau niveau à intégrer le précédent, et chaque implémentation à être compatible avec la précédente : CSS1 est donc développé pour être un sous-ensemble de CSS2, qui est lui-même développé pour être un sous-ensemble de CSS3. Ceci explique en partie la lenteur de l'avancement normatif de CSS.

Comme nous l'avons vu dans les chapitres précédents, le CSS peut être ajouté aux éléments HTML de 3 manières :

- **Inline** - en utilisant l'attribut style dans les éléments HTML.
- **Interne** - en utilisant un élément <style> dans la section <head>.
- **Externe** - en utilisant un fichier CSS externe.

Nous utiliserons dans le codage email les styles en ligne et interne.
Les caractéristiques applicables aux CSS sont exprimées sous forme de couple « propriété: valeur ». Ces couples sont séparés par des « ; ». Quand une propriété CSS peut prendre plusieurs valeurs, ces valeurs sont séparées par des « , ». Les valeurs peuvent être selon les cas exprimées à l'aide d'unités normalisées, ou de mots-clés propres à CSS. Dans un style interne, les propriétés sont regroupées par blocs de règles, délimités par les accolades {}.

CSS EN LIGNE

Faut-il arrêter d'écrire ses styles en ligne dans des e-mails[71] ? La question s'est souvent posée aux intégrateurs emailing : Pouvons-nous cesser d'écrire nos styles en ligne lorsque Litmus semble se l'autoriser[72] ? *(Car il faut tout de même reconnaître que cette méthode est un peu fastidieuse)*. Jusqu'en 2016, Gmail ne supportait presque que des styles en ligne. Les webmails de Yahoo, AOL, Outlook.com et même les versions d'Outlook de 2007 à 2016 sur Windows supportent les balises <style> depuis bien longtemps. La mise à jour de Gmail en 2016 semblait avoir tout changé en ajoutant officiellement le support de balises <style> et d'attributs class et id.

Et pourtant : Nombre de clients internationaux ne supportent pas la balises <style>. Des clients populaires comme Libero *(Italie)*, Mail.ru, Yandex *(Russie)*, Nate, Naver *(Corée)*, T-online *(Allemagne)*, Telstra *(Australie)* ou Terra *(Brésil)*. Mais il y a aussi certaines versions de certains clients mail, comme l'application iOS de SFR ou l'application Android d'Orange. Et enfin, il y a toujours Gmail sur Android avec un compte autre que Gmail *(surnommé GANGA)*, Yahoo dans leur application Android ou leur webmail mobile, et Outlook.com sur mobile dans l'ancienne version de leur webmail.

De plus, les clients mail ont leurs propres styles par défaut. Le HTML de votre email pourrait donc, potentiellement, hériter de styles trop génériques des webmails. L'avantage d'écrire ses styles en ligne est que cela surpasse automatiquement les styles par défaut des clients mails.

---

[69] https://www.w3schools.com/html/html_css.asp
[70] https://fr.wikipedia.org/wiki/Feuilles_de_style_en_cascade
[71] https://emails.hteumeuleu.fr/2017/02/faut-il-arreter-d-ecrire-ses-styles-en-ligne-dans-des-e-mails/
[72] https://litmus.com/community/discussions/6116-here-s-why-litmus-didn-t-inline-css-for-its-first-newsletter-of-2017

Le principal avantage d'écrire ses styles en ligne réside dans le fait que ceux-ci vont alors supplanter les styles par défaut des clients mail *(car les clients mails ont leurs propres styles par défaut, comme Outlook.com qui ajoute des styles comme background-color ou border-bottom sur chaque balise <style>)*.
Le principal inconvénient : le poids de l'email est plus lourd *(parfois jusqu'à 25% de différence avec un email sans styles en ligne)*, puisqu'il faut renseigner le style de chaque cellule, chaque texte, et cela même si les styles sont parfois identiques. Un code HTML plus léger signifie qu'il sera plus rapide à envoyer, plus rapide à charger, et surtout, évite de tronquer l'email sur certains clients mails comme Gmail qui coupe n'importe quel contenu après les 102ers kilo-octets d'un email. A l'inverse, ne pas écrire les styles en ligne pour des éléments qui n'apparaissent qu'une seule fois dans l'email peut surcharger inutilement le code *(un attribut class + une balise <style> pèseront plus lourd qu'un simple attribut style en ligne)*.

Pour simplifier l'écriture des styles en ligne, il existe aujourd'hui des inliners, snippets ou outils :

- Snippet pour Dreamweaver[73]
- Emmet[74]
- Liste d'inliner[75]

PROPRIETES CSS

Les propriétés CSS[76] permettent de moduler, travailler, **modifier un ensemble de caractéristiques techniques et graphiques d'un élément HTML**. Police, couleur, taille, graisse, alignement des textes, largeur, hauteur, position, couleur et image de fond et comportement des conteneurs, bordures, marges internes… De multiples possibilités sont offertes sur les mises en forme graphiques des éléments HTML via les propriétés CSS. Il n'existe aucune limite à la déclaration de ces propriétés CSS au sein d'un attribut style. Toute propriété CSS pourra être modifiée directement dans les media queries pour la version responsive d'un email. La liste suivante n'est pas exhaustive mais reprend les propriétés CSS les plus couramment utilisées dans le codage d'email.

Font-size[77]

Spécifie la taille de typographie d'un texte. La valeur est la plupart du temps renseignée en pixels *(unité fixe)*, mais il est tout à fait possible de choisir parmi plusieurs valeurs : xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger… Il est aussi possible de renseigner la valeur du font-size en pourcentage *(valeur évolutive)*.

Font-family[78]

Spécifie la famille de police d'un texte. La propriété font-family peut contenir plusieurs noms de familles comme typographies de secours *(alors séparées par des virgules)*. Si le navigateur ou le client mail ne supporte pas la première police spécifiée, il passera à la police suivante *(et ainsi de suite)*.

```
font-family : Arial, Helvetica, Sans-Serif ;
```

[73] *https://helpx.adobe.com/dreamweaver/using/reuse-code-with-snippets.html*
[74] *https://emmet.io/*
[75] *https://mosaico.io/email-client-tricks/email-inliners/*
[76] *https://www.w3schools.com/cssref/*
[77] *https://www.w3schools.com/cssref/pr_font_font-size.asp*
[78] *https://www.w3schools.com/cssref/pr_font_font-family.asp*

Il existe deux types de familles de typographie : family-name *(Times, Courier, Arial)* et generic-family *(serif, sans-serif, cursive…)* Dans l'idéal, il vaut mieux débuter avec une typo bien précise *(family-name)* et terminer la déclaration de la propriété avec une famille générique, pour laisser la possibilité à la machine, au client mail ou au navigateur de choisir une typographie dans une famille générique s'il ne trouve aucune des typographies renseignées précédemment.

Font-weight[79]

Permet de **définir la graisse du texte concerné**. Ses valeurs peuvent aussi bien être définies en toutes lettres *(normal, bold, bolder, lighter)* qu'en une suite de chiffres *(de 100 à 900, par pallier de 100, 400 correspondant à la valeur normal, et 700 à la valeur bold)*.

Color

Permet de spécifier la couleur d'un texte. Il existe plusieurs manières de définir la valeur d'une couleur *(cf. attribut bgcolor).*

Padding[80]

Permet, en CSS, de **spécifier des marges internes** à un élément HTML *(l'espace entre un contenu et sa bordure, entre le contenu et son conteneur)*. Elle peut être déclinée selon les côtés concernés : padding-top, padding-right, padding-bottom, padding-left.

Détaillons les multiples cas de figure de ses valeurs :

- 4 valeurs *(padding :10px 5px 15px 20px)* :
  - Padding-top à 10px
  - Padding-right à 5px
  - Padding-bottom à 15px
  - Padding-left à 20px

- 3 valeurs *(padding :10px 5px 15px)* :
  - Padding-top à 10px
  - Padding-right et padding-left à 5px
  - Padding-bottom à 15px

- 2 valeurs *(padding :10px 5px)* :
  - Padding-top et padding-bottom à 10px
  - Padding-right et padding-left à 5px

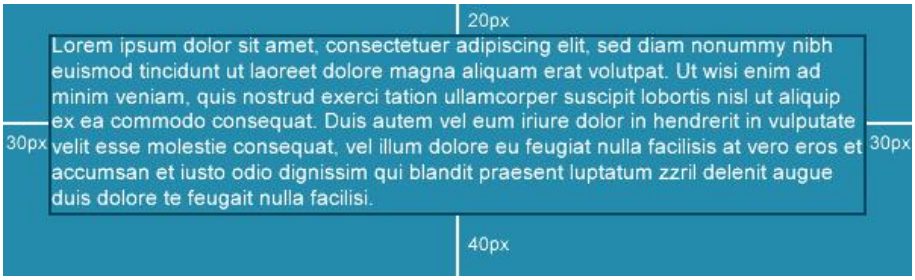
- 1 valeur *(padding :10px)* :
  - Les quatre paddings à 10px

Les valeurs négatives ne sont pas autorisées dans la propriété padding. Attention : Les padding sont parfois interprétés différemment suivant les clients emails : Quand une largeur ou hauteur est

[79] *https://www.w3schools.com/cssref/pr_font_weight.asp*
[80] *https://www.w3schools.com/cssref/pr_padding.asp*

précisée, certains clients emails ajoutent le padding à la largeur/hauteur, quand d'autres considèrent que le padding est à inclure dans la largeur/hauteur, ce qui peut provoquer des différences de rendu.

```
<table width="400" border="1" cellspacing="0" cellpadding="0">
    <tr>
      <td bgcolor="#f5a316" style="padding:20px 30px 40px;"><table
width="100%" border="1" cellspacing="0" cellpadding="0">
          <tr>
            <td style="font-family:Arial, Helvetica, sans-serif; font-
size:14px; color:#FFFFFF; line-height:1.4; text-align:left;"
align="left">Nullam hendrerit condimentum orci vel aliquam. Donec vitae
aliquam leo. Ut pulvinar mi quis metus venenatis convallis.</td>
          </tr>
        </table></td>
    </tr>
  </table>
```



Margin[81]

Cette propriété permet, en CSS, de **spécifier des marges externes** à un élément HTML. Comme pour la propriété padding, elle peut contenir en quatre valeurs les 4 côtés de l'élément concerné. Cependant, la propriété margin est encore mal interprétée par quelques clients mails, en particulier Outlook. Il est donc recommandé de ne pas faire appel pour le moment à cette propriété CSS.

Seul cas de figure où la propriété CSS margin est requise : l'alignement horizontal au centre d'un élément HTML : dans ce cas bien précis, il sera nécessaire de préciser un margin :0px auto ; à l'élément concerné, pour s'assurer que l'élément de type block en question *(tableau par exemple)* soit bien centré.

Line-height[82]

Permet de préciser l'interlignage d'un texte. Sa valeur peut être exprimée en pixels, en points *(pour des unités fixes)*, en pourcentage, ou sans unité. Dans ce dernier cas, la valeur concernée sera multipliée par la font-size attribuée pour obtenir l'interlignage final. Les valeurs négatives ne sont pas autorisées.

[81] *https://www.w3schools.com/cssref/pr_margin.asp*
[82] *https://www.w3schools.com/cssref/pr_dim_line-height.asp*

Text-align[83]

Permet de préciser l'alignement horizontal d'un texte au sein d'un élément HTML. Cette propriété peut prendre les valeurs suivantes : left, right, center, justify. La valeur justify n'est pas correctement interprétée par tous les clients mails. Justifier un texte permet de faire prendre au texte la largeur de son conteneur *(comme dans les journaux et magazines, et le phénomène est surtout propre au Print d'ailleurs)*. Il est toujours préférable de compléter cette propriété CSS text-align par l'attribut align.

Vertical-align[84]

Permet de préciser l'alignement vertical d'un contenu au sein d'un élément HTML. Cette propriété, qui a finalement la même fonction que l'attribut HTML valign, mérite d'être complété par l'attribut pour s'assurer une interprétation correcte sur l'ensemble des clients mails.

Float[85]

La propriété float est utilisée pour positionner des éléments dans une page web. Il est cependant fortement déconseillé d'utiliser cette propriété dans l'intégration HTML d'emailing. Elle souffre en effet d'un mauvais support de plusieurs clients mails. Préférez, à la place, modifier le comportement des éléments via la propriété display.

Display[86]

Permet de **jouer sur le comportement d'un élément HTML**. Rappelons que des éléments HTML *(comme <img>)* se comporte en élément inline. Tandis que d'autres *(comme <table>)* se comportent en élément block. Un élément block commence toujours sur une nouvelle ligne et prend toute la largeur de son conteneur. Il est possible de modifier ces comportements en précisant à la propriété display les valeurs inline, inline-block, none ou block *(entre autres)*.

En responsive, lorsqu'un élément inline *(<th> par exemple)* doit prendre toute la largeur, et que l'élément frère doit passer en-dessous, il sera possible de modifier les styles de ces éléments via les media queries en modifiant donc leur comportement en display :block.

Position[87]

Définit le type de **méthode de positionnement** utilisé pour un élément HTML. La valeur de cette propriété peut prendre les valeurs suivantes : *static, relative, fixed, absolute, sticky*. Si cette propriété est plutôt bien supportée sur les navigateurs et dans l'utilisation sur des pages web, comme pour de nombreuses autres propriétés CSS, la propriété position n'est en revanche pas correctement interprétée sur la majorité des clients mails. Mail il est tout à fait possible de mettre en place quelques hacks HTML et CSS[88] [89] pour que cette propriété puisse fonctionner correctement *(du moins, sa simulation)*.

---

[83] *https://www.w3schools.com/cssref/pr_text_text-align.asp*
[84] *https://www.w3schools.com/cssref/pr_pos_vertical-align.asp*
[85] *https://www.w3schools.com/css/css_float.asp*
[86] *https://www.w3schools.com/cssref/pr_class_display.asp*
[87] *https://www.w3schools.com/css/css_positioning.asp*
[88] *https://blog.gorebel.com/absolute-positioning-in-email/*
[89] *http://www.badsender.com/2017/11/09/email-css-position-absolute/*

COMMENTAIRES[90]

En CSS comme en HTML, il est tout à fait possible d'ajouter des commentaires, ou de passer certains morceaux du code en commentaires. Cela permet non seulement de **commenter son code pour tout update futur**, ou toute reprise du code par un autre intégrateur, mais aussi de ne pas supprimer certaines propriétés qui pourraient s'avérer utiles dans d'autres intégrations. La syntaxe est cependant différente de l'HTML : en CSS, tout texte compris entre /* *(début de commentaire)* et */ *(fin de commentaire)* sera bien interprété en tant que commentaire.

```
<style type="text/css">
    /* Début class pour cacher des éléments sur Outlook 2007 à 2013 */
    .visibility_mobile {
        display:none;
        mso-hide:all;
        visibility:hidden;
        max-height:0;
        font-size:0;
        line-height:0;
        padding:0;
    }
    /* Fin class pour cacher des éléments sur Outlook 2007 à 2013 */
    </style>
```

Attention : Les commentaires CSS ne peuvent être insérés que dans la balise <style>. En dehors de la balise <style>, il faudra utiliser des commentaires HTML.

[90] *https://css-tricks.com/snippets/css/comments-in-css/*

Encodage

Les pages Web peuvent être rédigées dans toutes sortes de langues et de très nombreux caractères peuvent être utilisés, ce qui requiert soit un jeu de caractères par type d'écriture, soit un jeu de caractères universel. Lors de l'apparition de HTML, le jeu de caractères universel Unicode n'était pas encore inventé, et de nombreux jeux de caractères se côtoyaient, notamment ISO-8859-1 pour l'alphabet latin et ouest-européen, Shift-JIS pour le japonais, KOI8-R pour le cyrillique. **Aujourd'hui, le codage UTF-8 de Unicode est le plus répandu.**

Le protocole de communication HTTP transmet le nom du jeu de caractères. L'en-tête HTML peut comporter le rappel de ce jeu de caractères, qui devrait être identique, sauf erreur de réglage. Enfin, à la suite d'un mauvais réglage, le jeu de caractères réellement utilisé peut encore différer du jeu annoncé. Ces mauvais réglages causent généralement des erreurs d'affichage du texte, notamment pour les caractères non couverts par la norme ASCII.

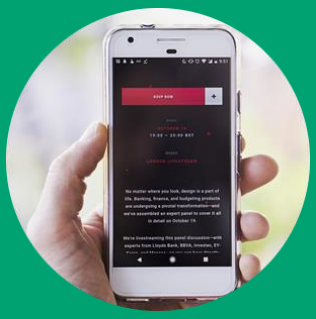
Car, à l'origine, les fichiers HTML sont faits pour être encodé[91] en ASCII, **c'est à dire sans caractères spéciaux**. En ASCII, les caractères sont codés sur 7 bits, soit un code composé de sept chiffres qui sont tous égaux à 0 ou à 1. Il permet donc de représenter 2^7 soit 128 caractères différents. 128 caractères sont suffisants pour mémoriser notre alphabet, les chiffres, des éléments de ponctuation, mais pas suffisant pour stocker nos caractères spéciaux *(accents, cédilles...)*.

Avant la généralisation d'Unicode, des entités ont été définies pour représenter certains caractères non ASCII : chaque caractère spécial est donc traduit par un code alpha numérique. Plusieurs listes de ces caractères sont disponibles sur le web[92]. Aujourd'hui encore, utiliser une table de conversion permet de se prémunir contre les incompatibilités. Un outil développé par Badsender[93] permet d'encoder la plupart des caractères spéciaux présents dans la langue française : é,è,à,ô,ü, « , » et tant d'autres ne poseront désormais plus aucun souci d'incompatibilité ou d'accent remplacé par un code incompréhensible.

Indentation

En informatique, l'indentation[94] consiste en l'ajout de tabulations ou d'espaces dans un fichier, pour une meilleure lecture et compréhension du code. L'indentation d'un fichier HTML est importante : elle facilite la lecture du code, la résolution de bugs, la dissociation des différents éléments et de leur correcte ouverture et fermeture. Un code HTML pour email optimal se doit absolument d'être correctement indenté.

[91] *http://formation.upyupy.fr/html-xhtml/caracteres-speciaux/*
[92] *https://www.jchr.be/html/caracteres.htm*
[93] *http://test.badsender.com/krktr/*
[94] *https://fr.wikipedia.org/wiki/Indentation*



Partie II

Email Mobile

Introduction

L'email mobile n'est plus facultatif. **L'email mobile est impératif**[95]. Une campagne emailing se doit, aujourd'hui, de proposer une version adaptée à ses multiples supports de consultation. Pour s'en convaincre, il suffit de prendre en compte ces chiffres, reportés par des acteurs majeurs de l'email marketing[96] :

- Le mobile représente 54% de toutes les ouvertures d'email[97].
- Apple iPhone est le 1er client email actuellement selon Litmus.
- L'ouverture des emails sur mobile a grimpé de 180% en 3 ans[98].
- La population mondiale de mobinautes est estimée[99], pour la fin de l'année 2018, à 2,2 milliards.
- Gmail est le 2nd client email selon Litmus, mais 75% des utilisateurs Gmail accèdent à leur compte depuis leur mobile.
- Apple iPad est le 3ème client email le plus utilisé selon Litmus.

Comment ne pas alors concevoir un email pour ce nouveau genre d'interface ? L'email évolue constamment, mais l'email marketing mobile n'est, en ce sens, pas vraiment une nouveauté. De nombreuses techniques ont été créées, testées, implémentées et continuent d'évoluer, pour garantir au destinataire mobinaute une autre expérience de consultation.

Car c'est bien là que se situe l'intérêt de la chose : si l'email en consultation Bureau est en soit une particularité technique dans le monde du développement web, l'email mobile requiert et impose un certain nombre de contraintes pour **optimiser la compréhension du message sur un écran désormais bien plus étroit et petit** que celui d'un ordinateur : Il n'est plus possible de se contenter de tailles de typographies trop petites, de textes illisibles, de visuels difficilement identifiables, ou de Call to Action miniatures.

Là où l'intégrateur emailing s'attachait, il y a peu, à obtenir un rendu parfait sur des logiciels de messagerie complexes sur l'interprétation de code *(Outlook)*, la priorité repose aujourd'hui sur un rendu mobile de qualité, ergonomique, accessible, pratique.
La technique désormais la plus courante repose dans le Responsive Web Design.

[95] *https://www.campaignmonitor.com/dev-resources/guides/mobile/*
[96] *https://www.emailmonday.com/mobile-email-usage-statistics/*
[97] *Litmus, «State of Email», Mars 2017*
[98] *Campaign Monitor, «Email Interaction across mobile and desktop»*
[99] *The Radicati Group, «Email Statistics Reports, 2014-2018»*

Email Responsive Design

Le responsive web design englobe les méthodes de conception qui proposent, grâce à différents principes et techniques, des contenus et une ergonomie auto-adaptables en fonction des interfaces de consultation utilisées par le visiteur, **et donc une consultation confortable, quel que soit le support utilisé**. Le responsive dans l'email marketing induit donc de faciliter la lecture ou consultation d'un email depuis un support mobile. Le destinataire peut consulter le même email *« à travers une large gamme d'appareils avec le même confort visuel et sans avoir recours au défilement horizontal ou au zoom avant/arrière sur les appareils tactiles notamment, manipulations qui peuvent parfois dégrader l'expérience utilisateur, tant en lecture qu'en navigation »*[100].

Autrement dit, à écran réduit, proposer un contenu tout aussi lisible, compréhensible. Mais pas seulement : par extension, à connexion réduite, proposer un contenu qui s'affichera tout aussi rapidement. A interaction impossible, proposer des solutions de secours. En vérité, l'utilisation de client de messagerie mobile éclipse peu à peu celle des clients Webmail et bureau, ce qui signifie que **fournir une mauvaise expérience de lecture sur petit écran peut désormais déranger la majorité des destinataires**. Cela peut conduire à une diminution des taux de réponse, ou comme Return Path le résume dans un rapport récent :

« …those that aren't tracking which device their subscribers are reading their emails on, or optimizing their emails or websites for mobile devices stand to lose out. A poor user experience could mean no response, no action, or plainly put, no ROI. »[101]

Le responsive dans l'email marketing est donc une collection de pratiques graphiques, de techniques *(requêtes de médias, grilles fluides, images fluides)* qui visent à fournir **l'expérience visuelle optimale sur diverses plateformes**.

CONTRAINTES LIEES AUX SUPPORTS NOMADES

Les supports mobiles impliquent, par certaines de leurs caractéristiques, un certain nombre de contraintes qu'il est judicieux de réviser ici :

- L'écran d'un smartphone ou d'une tablette est plus petit que celui d'un ordinateur : **L'espace de navigation est donc restreint**.
- Le réseau n'est pas stable en permanence : La connexion internet peut être fluctuante depuis les mobiles, voir peu présente.
- Les fonctionnalités de survol ne peuvent être détectées sur des écrans tactiles.

De ces contraintes découlent certaines exigences sur le produit :

- Le superflu visuel dû à la petite taille n'est pas autorisé.
- L'organisation des éléments, l'ergonomie, la gestuelle sont autant d'éléments auxquels il faudra porter grande attention.
- Les fonctionnalités ne peuvent pas se loger en nombre important sans noyer l'utilisateur.
- Il ne faut garder que les informations importantes pour les clients et éviter de transposer la version desktop sur mobile.
- Le design doit être simple et léger pour supporter les instabilités réseaux.
- Étant donné la connexion instable ou fluctuante, il est important de concevoir les mails en fonction, notamment en réduisant au maximum la quantité de données à recevoir. Il est très désagréable, pour les utilisateurs, de devoir patienter pendant le chargement des images.

[100] *https://fr.wikipedia.org/wiki/Site_web_adaptatif*
[101] *Email in Motion: How Mobile is Leading the Email Revolution, Return Path, 2011*

BONNES PRATIQUES / US ET COUTUMES

Si la consultation d'email sur mobile supplante dorénavant la consultation sur bureau, **cela induit des impacts éditoriaux, graphiques, ergonomiques, mais aussi techniques**. Si l'on prend cette donnée en compte, il devient logique d'appliquer quelques pratiques sensées pour améliorer l'expérience utilisateur d'une consultation d'email sur support mobile.

La meilleure façon d'appréhender, de comprendre, deviner et saisir ces pratiques reste l'expérience utilisateur, et donc, de tester. Keep in mind : *« Ce qui me gêne, gênera mes destinataires… Assurément. »*

Changer la navigation

Eviter, voir **bannir tout scroll horizontal** et privilégier le scroll vertical.

Plus que jamais, garder le message concis et placer tous les éléments de conception importants dans la partie supérieure de l'email. Le Call to Action principal est-il immédiatement visible lorsque l'email est ouvert sur mobile ? Le destinataire doit-il scroller pour l'apercevoir ? Cela passe aussi par la possibilité de cacher *(ou montrer)* des contenus spécifiques sur mobile, d'ajouter ou supprimer des marges autour des contenus, et de changer l'ordre des modules et la hiérarchie des informations…

Modulation des Call to Action

Faciliter la zone et agrandir la marge de clic sur un Call to Action, tout simplement. Un Call to Action aux dimensions de 140 pixels de large sur 40 pixels de hauteur semblera, sur un écran de Bureau, sans doute bien assez visible et aisément cliquable.

En sera-t-il de même sur mobile ou tablette ? La précision d'un doigt n'est pas la même que celle d'un curseur de souris. Qui n'a jamais eu la mauvaise surprise de cliquer sur un Call to Action contenu dans un email consulté sur mobile, sans que rien ne se passe ? Ou de se retrouver en mode « sélection texte » après avoir tapoté sur un lien ? La réaction est parfois immédiate et lourde de conséquence : le destinataire, frustré, peut quitter sa consultation.

La taille des Call to Action est un facteur clé dans les emails mobiles. Agrandir la zone de clic, ou la taille même du bouton est une tâche impérative lorsque l'on passe en responsive. Les liens et boutons doivent avoir, dans l'idéal *(et selon d'anciennes recommandations et instructions d'Apple)* une zone cible minimale de 44 pixels de hauteur sur 44 pixels de largeur.

Taille des textes

Augmenter la taille des textes HTML si besoin pour **garantir leur lisibilité**, compréhension et accessibilité. Un texte de 10px sera-t-il toujours lisible sur un écran d'Iphone 5S ? Faudra-t-il zoomer pour en saisir la teneur ? Pour le savoir, le test en rendu réel reste toujours la meilleure option…

Redimensionner / changer les visuels

Un visuel pour la version Bureau d'un email ne sera pas forcément adapté à une version mobile : considérez un visuel de 600 pixels de large, dans lequel un texte de 14 pixels est inséré. Si l'on divise rapidement la largeur de ce visuel pour arriver à la résolution la plus basse d'une consultation sur

mobile *(soit 300 pixels)*, on en déduit tout aussi rapidement la taille du texte sur la version mobile : 7 pixels. Il devient alors impossible de lire ou de comprendre ce texte.

Capture d'écran, source Rue du Commerce

Le responsive peut impliquer de prévoir des visuels propres à la version mobile, ou de redimensionner un visuel Desktop pour la version Responsive.

Ordre des modules

Posons-nous les bonnes questions : lorsqu'une configuration de contenu se présente ainsi : visuel à gauche, texte et Call to Action à droite, quel sera le rendu par défaut en mobile ? Il y a de fortes chances pour que le visuel soit en première partie, et le texte ainsi que le Call to Action en second. Si le visuel est cliquable, ou qu'il comprend l'offre principale, ce n'est peut-être pas un souci. Mais si ce n'est pas le cas ? S'il s'agit uniquement d'un visuel d'illustration ?

L'attribut HTML dir permet d'alterner l'ordre des contenus pour proposer le plus rapidement possible au destinataire les contenus que l'expéditeur jugera judicieux.

Temps de chargement

Sans doute l'objectif le plus important à atteindre dans l'email responsive : **diminuer au maximum le temps de chargement de l'email sur consultation mobile**. Un design d'emailing peut vite s'avérer relativement lourd en code HTML et en images rattachées. Même si les préconisations recommandent de limiter au maximum le poids d'un email en version Desktop, il devient capital de le réduire drastiquement en version responsive afin d'en améliorer la vitesse de chargement et d'affichage.

Cela peut passer par plusieurs techniques : n'afficher que les images indispensables à la bonne compréhension de l'email et ne pas faire apparaître les visuels superflus. Cacher des modules «

secondaires », dont la présence n'impacte en rien la compréhension de l'intégralité de l'email *(menu, bandeau de réassurance)*.

Poids des données

Un email sans image est sans doute moins attractif qu'un email avec images, c'est un fait. Même si une des tendances de l'email marketing en 2018 réside dans l'augmentation d'envoi d'email en « full text »[102], il reste que l'image a une part importante dans le comportement du destinataire sur l'email. Cependant, les images représentent, de manière certaine, le plus gros volume de données à télécharger lors de l'ouverture d'un email.

Dans ce cas bien précis, comment optimiser le temps de chargement de ces images ? Comment pallier les complexités des débits de réception de chacun, et afficher rapidement des visuels malgré tout de qualité ?
C'est justement par l'intermédiaire des requêtes *(Media Queries)* que l'expéditeur pourra décider de ne pas afficher certains visuels, certains modules, à l'aide de propriétés CSS comme display:none. Les éléments concernés ne seront pas affichés, cela pourra donc réduire considérablement le poids de l'email et de ses données.

De plus, il est primordial **d'optimiser le poids des images elles-mêmes**, non seulement lors de leur exportation et découpes, mais dans un second temps, via des plateformes spécialisées dans la compression d'images[103] comme JPEGmini[104], tinyjpg[105], tinypng[106], ou ezgif[107] par exemple. La qualité sera, la plupart du temps, conservée, tandis que le poids sera considérablement diminué.

L'extension des images est, là aussi, plus qu'important. Si l'on pouvait encore en faire l'impasse sur des consultations exclusivement Desktop, il n'est plus question de passer à côté de ce sujet avec les contraintes de connexion et données : une image au format .gif ne fera, sans aucun doute, pas le même poids que la même image au format jpeg ou png.

Bien que la qualité d'un visuel soit importante dans une campagne, il est judicieux de se poser les bonnes questions : Faut-il privilégier la qualité maximale d'une image, quitte à en impacter son temps d'affichage ? Nous sommes, en particulier avec le web et le mobile, dans un monde d'immédiateté. **Il devient laborieux de patienter pour l'affichage d'un contenu**. L'information doit s'afficher dans l'instantané ! Infliger au destinataire un temps d'attente pour l'affichage des informations ne peut qu'impacter négativement le taux de clics sur une campagne.

[102] *https://litmus.com/blog/top-email-design-trends*
[103] *https://compressjpeg.com/fr/*
[104] *https://www.jpegmini.com/*
[105] *https://tinyjpg.com/*
[106] *https://tinypng.com/*
[107] *https://ezgif.com/*

Media queries

La spécification CSS3 Media Queries définit les techniques pour l'application de feuilles de styles en fonction des périphériques de consultation utilisés pour du HTML.

HISTORIQUE

Avec CSS2 et HTML4, il était déjà possible de spécifier un média de destination pour l'application d'une ou plusieurs feuilles de style. C'est ainsi que l'on a pu associer des règles CSS complémentaires pour l'impression, modifiant la mise en page, favorisant tel élément ou faisant disparaître un autre inutile à la sortie sur papier, par exemple un menu de navigation. La balise <link> est alors dupliquée pour autant de feuilles de style que nécessaire, et **comporte un attribut media précisant le contexte dans lequel les styles doivent être appliqués** :

```
<!doctype html>
<head>
<meta charset="utf-8">
<title>Media Queries !</title>
<link rel="stylesheet" media="screen" href="screen.css" type="text/css" />
<link rel="stylesheet" media="print" href="print.css" type="text/css" />
</head>
<body>
...
</body>
```

L'attribut media peut prendre *(depuis CSS2)* les valeurs suivantes : screen *(Écrans)* | handheld *(Périphériques mobiles ou de petite taille)* | print *(Impression)* | aural *(CSS 2.0)* / speech *(CSS 2.1) (Synthèses vocales)* | braille *(Plages braille)* | embossed *(Imprimantes braille)* | projection *(Projecteurs ou présentations avec slides)* | tty *(Terminal/police à pas fixe)* | tv *(Téléviseur)* | all *(Tous les précédents)*.

La philosophie des Media Queries *(ou requêtes de media)* est d'offrir un panel de critères plus vaste et plus précis, à l'aide de propriétés et de valeurs numériques, ainsi que de combinaisons multiples de ces mêmes critères. Le but est de **cibler plus finement** les périphériques de destination en fonction de leurs capacités intrinsèques.

Dans la majorité des cas, on utilise les media queries pour produire des améliorations spécifiques à l'affichage sur les mobiles, qui sont directement concernés par des critères sur les dimensions de l'écran *(en termes de résolution et d'espace disponible)* et sur l'utilisation tactile.

Ainsi on retrouvera le plus fréquemment des règles pour :

- Agrandir la taille du texte.
- Agrandir la taille des contrôles et zones cliquables *(pour une utilisation au doigt)*.
- Faire passer le contenu sur une seule colonne.
- Masquer ou afficher des éléments spécifiques.
- Ajuster les dimensions et marges.

L'email responsive utilise une requête multimédia *(@media)* avec un ensemble spécial de styles CSS qui agissent comme des instructions conditionnelles ou des règles dynamiques. Soigneusement planifiées, elles peuvent aider à rendre les e-mails plus lisibles sur différentes tailles d'écran. Les requêtes multimédias détectent la taille de l'écran d'un périphérique, puis activent différents ensembles de règles en fonction de la taille de l'écran. Celles-ci peuvent être très simples à mettre en œuvre ou assez complexes, en fonction de ce que vous souhaitez accomplir. Elles nécessitent plus de

planification et de test que les emails standards et ne fonctionnent pas dans tous les clients de messagerie.

FONCTIONNEMENT

Dans le contexte du courrier électronique, les types de média définissent les styles CSS à utiliser en fonction de la taille de l'écran. Ce type de média indique *"Si l'e-mail est affiché sur une taille d'écran de 480 pixels ou moins, utilisez le code CSS suivant"*.

« La spécification CSS3 Media Queries définit les techniques pour l'application de feuilles de styles en fonction des périphériques de consultation utilisés pour du HTML »[108] : **Cette affirmation est à adapter à l'emailing** : les media queries, un code CSS3 particulier, permet d'appliquer des mises en forme particulières en fonction des périphériques de consultation. Les media queries sont donc, par leur définition, la suite logique du responsive web design. Il s'agit là d'une partie intégrante du responsive, puisque nous adaptons dynamiquement du design à l'aide de CSS. Concrètement, les media queries indique ceci :

« Si le media de consultation a une largeur inférieure/supérieure à [valeur]pixels, alors : »

Ces règles complémentaires permettent donc de modifier la mise en page, favorisant tel élément, ou faisant disparaître tel autre, devenu inutile ou trop encombrant. Les directives media queries sont intégrées directement dans la balise <style> d'une intégration HTML. Il sera alors nécessaire d'utiliser la règle @media suivie directement du type de média *(en général, screen pour l'email marketing)*.

L'écriture de ces requêtes est relativement explicite[109] : **une media query est une expression dont la valeur est toujours vraie ou fausse**. Il suffit d'associer les différentes déclarations possibles avec un opérateur logique pour définir l'ensemble des conditions à réunir pour l'application des styles compris dans le bloc adjacent.

Les opérateurs logiques peuvent être :

- and "et",
- only "uniquement"
- not "non"

Pour obtenir l'équivalent du "ou", il suffit d'énumérer différentes media queries à la suite, séparées par des virgules : si l'une d'entre elles est valable, alors l'ensemble de la règle sera appliqué. En général, on combine ensemble un type de média *(screen, all...)* et une expression grâce à « and », bien qu'une expression seule puisse être utilisée. L'expression est toujours écrite entre parenthèses.

Les deux exemples suivants ciblent les écrans de largeur inférieure à 640 pixels grâce à la règle max-width associée à la valeur 640px.

```
@media screen and (max-width: 640px) {
  .bloc {
    display:block;
    clear:both;
  }
}
```

[108] *https://www.alsacreations.com/article/lire/930-css3-media-queries.html*
[109] *https://litmus.com/blog/understanding-media-queries-in-html-email*

Les combinaisons peuvent être multiples. Ici on s'adresse à un écran dont la résolution en largeur est comprise entre 200 et 640 pixels :

```
@media screen and (min-width: 200px) and (max-width: 640px) {
  .bloc {
    display:block;
    clear:both;
  }
}
```

La plupart des critères *(ou fonctionnalités)* peuvent être préfixés par min- et max- lorsqu'elles acceptent des valeurs numériques pour définir des valeurs minimales ou maximales à respecter : device-aspect-ratio *(ratio du périphérique de sortie, par exemple 16/9)*, aspect-ratio *(ratio de la zone d'affichage)*, device-height *(dimension en hauteur du périphérique)*, device-width *(dimension en largeur du périphérique)*, width *(dimension en largeur de la zone d'affichage) …*
Les dimensions pourront être évaluées avec des unités *(px, em)*.

La fonctionnalité *(ou critère)* la plus couramment utilisée dans l'email marketing reste la largeur, puisque c'est la contrainte principale qui nous amène à utiliser des media queries et proposer une version plus lisible. On parlera alors de « points de rupture » : Un point de rupture, c'est un jalon, une taille particulière à partir de laquelle on bascule sur un autre style d'affichage, d'autres propriétés CSS.

Directement dans le code HTML de l'emailing on attribue ainsi des class et id à plusieurs éléments contenus qui s'afficheront, ou non, en fonction de la résolution du terminal destinataire, qui prendront en compte telle ou telle modification de taille de typographie, de couleurs, de marges, de positionnement, etc...

SUPPORT[110] [111]

La première chose dont il faut avoir conscience, c'est qu'il n'est pas possible d'avoir des Media Queries en style CSS en ligne. Les media queries doivent obligatoirement être renseignées dans la balise <style>, présente dans le <head> de l'email. Un email responsive sur un client qui ne supporte pas le CSS dans <head> est donc d'ores et déjà exclu.

Pour cibler class et id, la syntaxe sera la suivante :

```
<style type="text/css">
    @media only screen and (max-width:600px) {
    .width300px {
    width:300px !important;
   }
   }
   </style>
```

Le support des media queries est tout de même plutôt bon sur l'ensemble des clients mails, comme en témoigne ce tableau récapitulatif :

[110] *https://litmus.com/help/email-clients/media-query-support/*
[111] *https://www.campaignmonitor.com/css/media-queries/media/*

DESKTOP	MOBILE	WEBMAILS
✓ AOL Desktop	✓ Android 4.2.2 Mail	✗ AOL Mail
✓ Apple Mail	✓ Android 4.4.4 Mail	✓ G Suite
✗ IBM Notes 9	✗ AOL Alto Android app	✓ Gmail
✓ Outlook 2000-03	✗ AOL Alto iOS app	✓ Google Inbox
✓ Outlook 2007-16	✓ BlackBerry	✗ Outlook.com
✓ Outlook Express	✓ Gmail Android app	✓ Yahoo! Mail
✓ Outlook for Mac	✗ Gmail Android app IMAP	
✓ Thunderbird	✓ Gmail iOS app	
✗ Windows 10 Mail	✗ Gmail mobile webmail	
✓ Windows Live Mail	✓ Google Inbox Android app	
	✓ Google Inbox iOS app	
	✓ iOS 10 Mail	
	✓ iOS 11 Mail	
	✓ Outlook Android app	
	✓ Outlook iOS app	
	✓ Windows Phone 8 Mail	
	✓ Yahoo! Mail Android app	
	✓ Yahoo! Mail iOS app	

Fluid/Hybrid

L'approche du codage Hybrid[112] *(parfois appelé « Spongy Coding »)* est **une réaction directe aux clients mails tels que Gmail *(sur certaines versions)* qui ignorent les media queries**. Créée par Fabio Carneiro, popularisée par Mike Regan et Nicole Merlin, cette technique utilise toujours des tableaux et des images fluides qui, contrairement aux emails responsive avec media queries, sont ici « fluides par défaut ». Au lieu d'utiliser des media queries pour déclencher ces comportements fluides, le codage hybrid favorise les commentaires conditionnels de Microsoft pour contenir la largeur des tableaux sur des écrans plus grands. Les emails hybrid fonctionnent donc selon les principes suivants :

- Tableaux et éléments fluides par défaut.
- La propriété CSS max-width pour contraindre la largeur des éléments sur desktop.
- Les commentaires conditionnels Outlook pour limiter la largeur sur Outlook.

Regardons un morceau de code d'un email hybrid d'un peu plus près :

[112] *https://litmus.com/blog/understanding-responsive-and-hybrid-email-design*

```
<table border="0" cellpadding="0" cellspacing="0" width="100%">
  <tr>
      <td bgcolor="#00a9f7" align="center">
          <!--[if (gte mso 9)|(IE)]>
          <table align="center" border="0" cellspacing="0" cellpadding="0"
width="600">
          <tr>
          <td align="center" valign="top" width="600">
          <![endif]-->
          <table border="0" cellpadding="0" cellspacing="0" width="100%"
style="max-width: 600px;" >
              <tr>
                  <td align="center" valign="top" style="padding: 40px 0px
40px 0px;">
                      <!-- IMAGE -->
                      <img alt="Example" src="http://placehold.it/600x300"
width="600" style="display: block; width: 100%; max-width: 100%;" border="0">
                  </td>
              </tr>
              <tr>
                  <td align="center" valign="top" style="padding: 0px 10px
20px 10px;">
                      <!-- HEADLINE -->
                      <p style="color: #ffffff; font-family: sans-serif; font-
size: 24px; font-weight: bold; line-height: 28px; margin: 0;">Announcing Some
News</p>
                  </td>
              </tr>
              <tr>
                  <td align="center" valign="top" style="padding: 0px 10px
60px 10px;">
                      <!-- COPY -->
                      <p style="color: #b5e2f7; font-family: sans-serif; font-
size: 16px; font-weight: normal; line-height: 24px; margin: 0;">Lorem ipsum
dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget
dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient
montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque
eu, pretium quis, sem. Nulla consequat massa quis enim.</p>
                  </td>
              </tr>
          </table>
          <!--[if (gte mso 9)|(IE)]>
          </td>
          </tr>
          </table>
          <![endif]-->
      </td>
  </tr>
</table>
```

Ici, nous utilisons donc des valeurs en pourcentage pour les largeurs de tableaux ou d'images. **Cela permet au contenu de l'email de « remplir » toutes les tailles d'écrans possibles**[113]. Pour éviter que le destinataire ne voie un email très large sur son bureau, nous devons ajouter la propriété CSS max-width à la table et à l'image. Cependant, Microsoft Outlook ne comprend pas actuellement cette propriété. Il sera donc nécessaire d'utiliser des commentaires conditionnels pour créer des tables invisibles à largeur fixe, qui ne seront présentes que sur Outlook. Tous les éléments étant fluides par défaut, notre email fonctionne donc très bien sur toutes les résolutions d'écrans. Et comme il ne s'appuie sur aucune media query pour déclencher ce comportement responsive, il fonctionne également sur les clients mails qui n'interpréteraient pas les media queries.

L'avantage de la méthode hydbrid réside donc dans le fait qu'elle prend en charge tous les clients de messagerie. Etant donné que tout se passe dans le corps *(<body>)* et qu'aucune media query ni style dans le <head> n'est nécessaire, cette méthode fonctionne partout. De plus, il sera toujours possible d'ajouter malgré tout des media queries pour améliorer plus encore le rendu sur les clients mails supportant ces requêtes. En revanche la méthode hybrid demande un travail particulier sur le design de l'email : puisque les media queries ne sont pas envisageables, il vous faudra adopter un système de colonnes pour obtenir un rendu graphique correct sur mobile. Ces mêmes colonnes *(sur un système de deux colonnes côte à côte)* ne pourront pas dépasser une largeur maximale de 300 ou 320 pixels de large, puisqu'il s'agit de la résolution la plus basse sur mobile. Ce qui vous oblige à vous contraindre à une largeur maximale de 600 à 640 pixels de large pour votre email en version desktop.

Avant de mettre en place une telle technique et méthode de codage, ou même avant de commencer le design d'un email, il est donc important de prendre en compte ces quelques questions :

- Dois-je prendre en compte les clients mails ne supportant pas les media queries *(à analyser en rapport avec les statistiques et retours de campagnes précédentes)*
- Quelle est la complexité de ma mise en page ?
- Dans quelle mesure suis-je à l'aise avec une telle complexité de code ?

Il faut beaucoup de pratique et d'expérimentation pour développer des emails hybrides. Si vous n'avez pas les connaissances ou le temps d'investir dans l'apprentissage du code hybrid, les media queries et le responsive traditionnel conviendront. Analysez aussi tout design avant de décider de la manière de coder.

[113] *https://blog.chamaileon.io/what-do-scalable-fluid-responsive-and-hybrid-email-design-mean/*

Partie III

Contenus à utiliser avec précaution

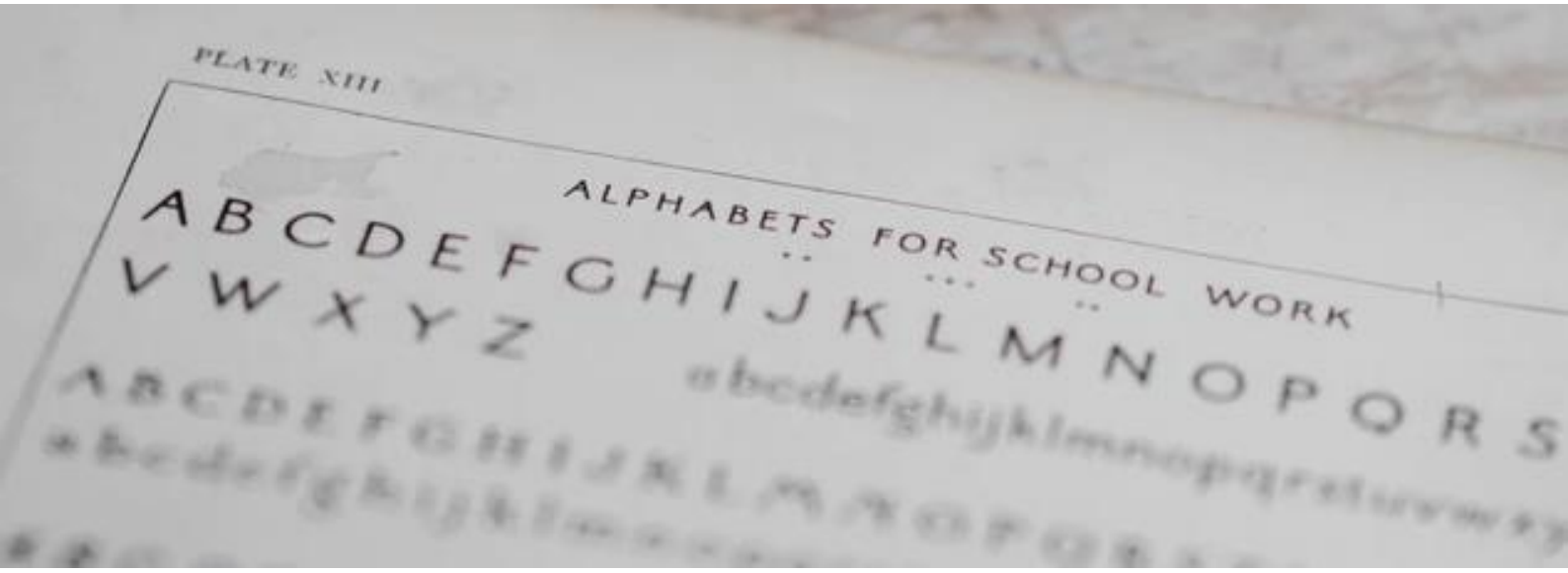
Introduction

L'intégration HTML pour email est un sujet complexe en partie dû à la manière de coder. Mais, comme nous avons pu l'entrevoir jusqu'à présent, sa difficulté réside aussi dans le support des balises HTMLs, des attributs, et des propriétés CSS selon les clients mails. Pour autant, tout Designer d'emailing souhaiterait ne pas voir sa créativité bridée par un ensemble de trop nombreuses contraintes. Cependant, avant tout début de création, il faut impérativement connaître les contenus à utiliser avec précaution, et ceux qui seront tout simplement bannis. Commençons.

Images d'arrière-plan

Il est parfois nécessaire d'utiliser des images de fond. Même si tout designer d'emailing est assez fébrile et réticent à la mise en place d'image de fond, elles sont parfois indispensables pour amener un peu de rythme à l'ensemble d'une créa. Si le sujet des images de fond est sensible, c'est surtout parce qu'**il est complexe en soit sur le plan technique**. Des solutions existent, parfois lourdes et ardues, mais elles existent[114]. Campaign Monitor a même mis au point un générateur il y a maintenant quelques temps pour nous simplifier la vie[115].

Les images de fond sont donc difficiles à mettre en place au sein d'une intégration HTML d'email car le support est très complexe. Cela demande un temps considérable en tests d'email rendering. Malgré l'ensemble des techniques existantes, aucun ne permet un support total. Dans la mesure du possible, il est nécessaire de séparer le contenu textuel du Design. Cela évite ainsi d'exporter des informations textuelles en image, ou de passer par la technique des images de fond.



Typographie

TYPOS STANDARDS OU WEBSAFE FONT

Les typos dites standards ou websafe, **sont des polices installées par défaut sur la machine cliente**[116]. Il existe plusieurs configurations différentes *(Windows, Mac OS, Linux…)*, et chaque

[114] *https://litmus.com/blog/the-ultimate-guide-to-background-images-in-email*
[115] *https://backgrounds.cm/*
[116] *https://www.anthony-brard.com/les-fonts-web-safe*

configuration possède son propre jeu de police de base *(Arial est fournie sur Windows mais pas sur Mac OS, et inversement pour Helvetica)*. En CSS, nous allons donc, au sein d'un email, chercher à définir une liste de polices *(ou de famille de polices)* plus ou moins similaires et compatibles pour assurer un rendu homogène pour tout le monde.

La liste des typos Websafe est assez restreinte[117] :
Arial, Helvetica, Times New Roman, Times, Courier New, Courier, Verdana, Georgia, Palatino, Garamond, Bookman, Comic Sans MS, Trebuchet MS, Arial Black, Impact.
Il existe plusieurs catégories de polices : Sans-Serif *(fonts sans empattement, assimilables à des typographies « bâton », sans fioriture)*, Serif *(fonts avec empattement)*, Monospace *(ou police à chasse fixe : chaque caractère prend la même largeur/hauteur)*, Fantasy, Script.
Chacune de ces catégories comprend donc plusieurs typographies, plus ou moins bien supportées sur tel ou tel système d'exploitation[118].

Le principal avantage de ces typos réside dans le fait qu'elle ne nécessite aucun chargement additionnel puisqu'elles sont déjà présentes sur le poste client *(et limite donc au maximum le temps de chargement pour l'affichage d'un email, puisqu'aucun appel à une typo extérieure)*.
Les inconvénients : en fonction de l'OS, vous ne les aurez pas forcément toutes. Se limiter à ces typographies contraint aussi la créativité.

Il existe malheureusement des spécificités sur certains polices, rendant le choix des typos websafe parfois complexes. Ainsi, les polices Symbol ou Webdings fonctionnent seulement sur IE, et seront substituées par d'autres polices sur les autres navigateurs[119].

WEBFONT[120], TYPOS SPECIFIQUES, TYPOS EXOTIQUES

Quel client mail prend en charge les polices web ?
Apple Mail, Outlook 2011 pour Mac, iOS Mail, Apple Mail, ipad et iphone (autre que compte GANGA), Android Mail (autre que compte GANGA), Thunderbird.
Quel client mails ne prend pas en charge les polices web ?
Lotus Notes 8, Lotus Notes 8.5, Gmail, Google Apps, Office 365, Yahoo!, Outlook.com, Outlook 2003/2007/2010/2013/2016, AOL, Free.

Gardez donc à l'esprit que l'appel de web font est un exercice assez périlleux, à manipuler avec précaution. Son support est encore relativement faible. Cependant, si vous voulez vraiment y faire appel, une méthode est à privilégier : En 2010, Google a lancé sa propre bibliothèque gratuite de polices de caractère : Google Fonts. Créez une balise <link>, en tête de votre intégration HTML, dans le <head>. Au sein de cet élément <link>, faites appel à l'url de la police.

```
<link href="https://fonts.googleapis.com/css?family=Open+Sans"
rel="stylesheet">
```

Ensuite, appelez la police en question en premier lieu, dans votre propriété font-family, suivie des typos ou familles de secours :

```
font-family: 'Open Sans', sans-serif;
```

[117] *https://websitesetup.org/web-safe-fonts-html-css/*
[118] *https://www.cssfontstack.com/*
[119] *http://www.ampsoft.net/webdesign-l/WindowsMacFonts.html*
[120] *https://www.campaignmonitor.com/blog/email-marketing/2016/07/10-things-need-know-web-fonts-email-right-now/*

Il existe d'autres méthodes[121] d'import de typos spécifiques : La première consiste à spécifier un @import dans le code CSS :

```
<style type="text/css">
    @import url('http://fonts.googleapis.com/css?family=Open+Sans');
</style>
```

La seconde méthode fait appel à la requête @font-face :

```
<style type="text/css">
    @font-face{
        font-family:'Open Sans';
        font-style:normal;
        font-weight:400;
        src:local('Open Sans'),
            local('OpenSans'),

url('http://fonts.gstatic.com/s/opensans/v10/cJZKeOuBrn4kERxqtaUH3bO3LdcAZYWl9
Si6vvxL-qU.woff') format('woff');
    }
</style>
```

Dans l'idéal, et si l'intégration HTML vous le permet, privilégiez toujours la méthode <link>.

RETINA

Introduction

En dehors des tailles d'écran, la révolution de l'email mobile introduit un autre défi pour les intégrateurs email : les écrans à haute résolution. Ces écrans, souvent appelés écrans Retina, ont le pouvoir d'améliorer une campagne email en offrant aux abonnés une meilleure expérience. En 2010, Apple a sorti l'iPhone 4, qui comportait alors ce qu'Apple appelait un écran Retina. Ce type d'écran était l'un des premiers écrans à haute résolution *(haute résolution de points par pouce, densité de pixels sur l'écran)*. Plus le DPI *(Dots per inch)* est élevé, plus les images et le texte sur cet écran sont détaillés et clairs. Les autres fabricants d'écrans n'ont pas tardé à monter eux-aussi au créneau, et les écrans Retina sont maintenant présents sur les tablettes, ordinateurs portables ou ordinateurs de bureau.

En raison de la manière dont les affichages haute résolution fonctionnent, les images non optimisées pour le Retina finissent par apparaitre, dans un email[122], floues et pixelisées sur ces mêmes écrans.

[121] *https://litmus.com/blog/the-ultimate-guide-to-web-fonts*
[122] *https://litmus.com/blog/understanding-retina-images-in-html-email*

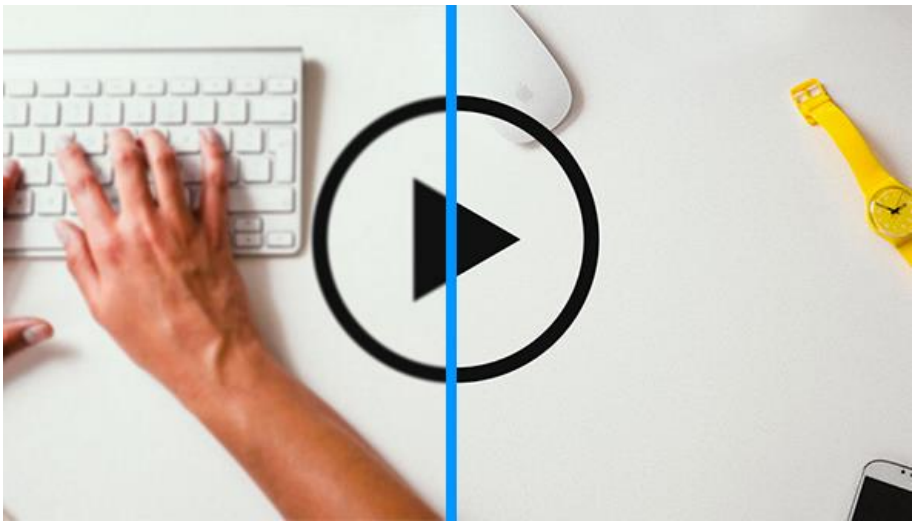


Image non-Retina VS image Retina, source Litmus

A mesure que les utilisateurs s'habituent à l'utilisation des écrans Retina, ils s'attendent aussi à ce que les textes et images apparaissent clairs et nets. Si vous n'optimisez pas ces éléments, les destinataires voient une conception dégradée, ce qui peut entrainer une perte de confiance envers la marque.

Technique

Les écrans à haute résolution ont 2 fois plus de pixels par pouce que les autres écrans traditionnels. Par conséquent, **les images d'un email doivent être deux fois plus grandes** pour que les écrans Retina puissent afficher plus de pixels[123]. Il est donc nécessaire, tout d'abord, de concevoir les images de votre email avec des dimensions deux fois plus importantes. Ensuite, dans le code HTML, nous appelons la taille d'affichage originale prévu dans les attributs width et height, pour mettre correctement à l'échelle cette image bien plus grande. Ainsi, pour une image devant, à l'origine, avoir une largeur de 120 pixels et une hauteur de 50 pixels, nous la travaillerons sur une largeur de 240 pixels pour une hauteur de 100 pixels. Cependant, dans le code HTML, la définition de ses attributs width et height restera la suivante :

```
<img alt="Texte alternatif" src="nomdelimage.jpg" width="120" height="50"
style="display:block;" border="0">
```

Il est aussi possible de faire appel à des images Retina pour des images de fond, même si la technique est un peu plus complexe[124]. Gardez à l'esprit que des images deux fois plus grandes sont aussi… souvent plus lourdes. De plus, contraindre une image à des largeurs et hauteurs plus petites pour aussi amener à des problèmes de clarté lors de l'affichage de l'image. Avant d'introduire le Retina dans vos campagnes, assurez-vous que son utilisation est indispensable.

[123] *https://www.emailonacid.com/blog/article/email-development/mobile-optimization-retina-images-in-email/*
[124] *https://litmus.com/community/discussions/4931-using-retina-images-as-background-images*

Partie IV

Contenus à ne pas utiliser

JavaScript

JavaScript[125] est un langage de programmation qui élabore du contenu web interactif et permet, sur les sites web, de proposer des effets de transition, d'animation, comme des effets d'auto-scrolling, des slides et carrousels, mais aussi de vérifier, valider et envoyer des formulaires *(entre autres utilités)*. Cependant, dans le monde de l'email marketing, **ajouter du javascript dans un code d'email vous assurera d'envoyer votre campagne directement dans la boîte Indésirables** de vos destinataires.

Pourquoi ? Sur une page web, c'est l'internaute qui décide des pages web visitées. Il décidera de ne pas visiter les pages qui, selon lui, peuvent contenir des éléments susceptibles de nuire à son ordinateur ou à sa sécurité. Avec les emails, c'est l'expéditeur qui contrôle le mieux les emails envoyés, et le destinataire a bien moins de contrôle. C'est là qu'interviennent les robots anti-spam et les techniques de filtrage : les paramètres de sécurité des programmes de messagerie empêchent le fonctionnement de Javascript dans les emails par mesure de sécurité pour empêcher, entre autres, les virus, ou parce que les scripts présents dans le mail pourraient cacher du contenu malveillant.

Elément iframe

Il s'agit ici d'une balise HTML qui permet d'intégrer une page HTML au sein d'un autre document HTML. Iframe pour Inline Frame. La balise iframe permet notamment d'insérer dans une page web des éléments qui proviennent d'un autre serveur sans que le visiteur en ait conscience. Et c'est bien là la problématique de l'iframe dans l'emailing : Comme les iframes peuvent potentiellement créer un lien vers un contenu malveillant *(comme le JavaScript)*, de nombreux clients mails les désactivent. Le support de cette balise est donc très faible.

[125] *http://email-a-table.fr/le-javascript-dans-les-emails,495*

CLIENT	AFFICHAGE DE L'IFRAME
AOL Webmail	✗ No
Gmail	✗ No
Windows Live Hotmail	✗ No
Yahoo! Mail	✗ No
AOL 9	✗ No
Apple Mail ¾	✓ Yes
Entourage 2008	✗ No
Lotus Notes 6/7	✗ No
Lotus Notes 8	✓ Yes
Outlook 2000	✓ Yes
Outlook XP/2003/2007/2010	✗ No
Thunderbird	✓ Yes
Windows Mail	✓ Yes
Blackberry	✗ No
iPhone/iPad	✓ Yes
Windows Mobile 5	✗ No
Windows Mobile 6	✓ Yes
Android (client par défaut)	✓ Yes
Android (client Gmail)	✗ No

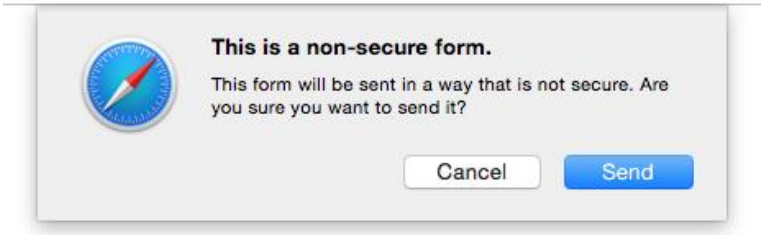
Flash[126]

Le Flash[127] ne doit désormais plus être intégré au sein d'un email : son support est tout d'abord absolument déplorable. Seul Mac Mail affichera le contenu. La plupart des clients mails n'afficheront pas le contenu Flash, ni son contenu alternatif. Mais, de plus, des avertissements de sécurité seront aussi énoncés sur certains clients mails *(Outlook)*. La technique Flash doit donc définitivement être proscrite.

Formulaires HTML

Les formulaires sont importants car ils permettent de partager des données provenant d'abonnés ou de clients. Le problème est qu'un formulaire dans un courrier électronique n'est pas sécurisé et que, même s'il existait un moyen de le faire, les clients de messagerie que les abonnés utilisent peuvent voir le formulaire comme un risque de sécurité et envoyer une alerte à l'abonné qui le découragerait de le compléter.

[126] *https://www.definitions-marketing.com/definition/flash/*
[127] *https://www.campaignmonitor.com/blog/email-marketing/2006/01/the-truth-about-flash/*



Alerte de sécurité, source Litmus

De plus, les champs de texte, les entrées de texte, les boutons radio et les cases à cocher sont des éléments de formulaire HTML simples, mais le bouton d'envoi nécessite souvent du JavaScript. Certains clients de messagerie afficheront les formulaires, mais les abonnés ne seront malheureusement pas en mesure de les envoyer. Évitez les formulaires HTML et envisagez une alternative.

ALTERNATIVES

Questions à choix multiples

Il est tout à fait possible, pour obtenir les mêmes retours qu'un formulaire, de proposer au destinataire des réponses aux questions posées au sein d'un email, et de lui laisser le choix de ses réponses. Selon la réponse cliquée, le tracking sera différent, ou la page de redirection différente. **La méthode de QCM peut être une alternative** si vous souhaitez obtenir des réponses à des questions directement dans vos campagnes emailing.

Choix orientés, source Prisma Media

AMP

Introduction

AMP[128] [129] [130] [131] [132] [133] [134] [135] **pour Accelerated Mobile Pages** : initiative de Google en 2015 visant à accélérer le temps de chargement des pages mobiles en éliminant les codes et scripts inutiles tels que le Javascript. En raison de sa popularité et de son efficacité, AMP s'est développé. Maintenant, il ne sert pas qu'à charger plus rapidement des pages ou articles : Il peut créer des carrousels sur mobile, accepter les commentaires d'utilisateurs via des formulaires, créer des interactions personnalisées, etc… Logiquement, **l'évolution de l'AMP vise aussi à s'étendre sur l'un des canaux de communication les plus populaires au monde : le courrier électronique**.

AMP4EMAIL

L'objectif est d'améliorer et de moderniser l'expérience de messagerie grâce à la prise en charge du contenu dynamique et de l'interactivité tout en assurant la sécurité des utilisateurs.

En introduisant l'AMP dans le courrier électronique, les destinataires pourraient alors effectuer des tâches généralement réservées aux sites Web. AMP pourrait changer la donne de l'impact de l'email et promet des expériences de messagerie plus engageantes, interactives et exploitables. Google présente ainsi les possibilités suivantes :

- Concevoir des composants interactifs en utilisant une vaste bibliothèque de composants AMP pris en charge, tels que le carrousel, les formulaires, les listes…
- Aider les concepteurs d'email à mettre à jour leur contenu automatiquement et proposer de l'interactivité aux utilisateurs.

L'avantage le plus important dans l'AMP réside sans doute dans le fait que les utilisateurs pourraient agir rapidement via l'email, sans avoir à visiter une page de destination distincte : cela limite ainsi le nombre d'étapes et d'interactions, **ce qui n'est que positif dans une campagne emails**, le taux de conversion pouvant être largement optimisé : des tâches telles que la saisie d'un formulaire, la planification d'une rendez-vous, parcourir de catalogues interactifs, des listes, les sauvegarder… L'AMP utiliserait aussi un contenu dynamique pour proposer des emails mis à jour automatiquement, sans durée de vie.

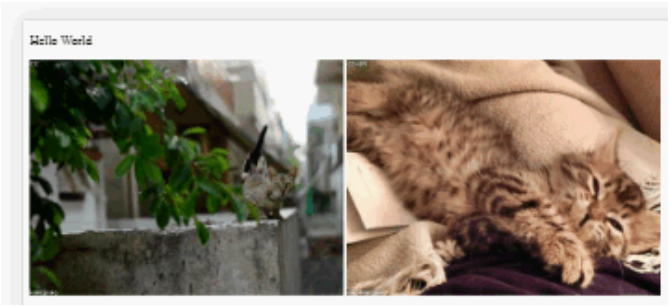
Technique

Jusqu'à présent, un email peut être conçu en deux formats : version en texte brut, ou version HTML. La version AMP est donc désormais à prendre en considération. Cela signifie que lorsque vous envoyer un courrier électronique, la version AMP apparaître sur les clients de messagerie prenant en charge AMP, et utilisera la version HTML ou texte brut pour tous les clients ne prenant pas en charge AMP.

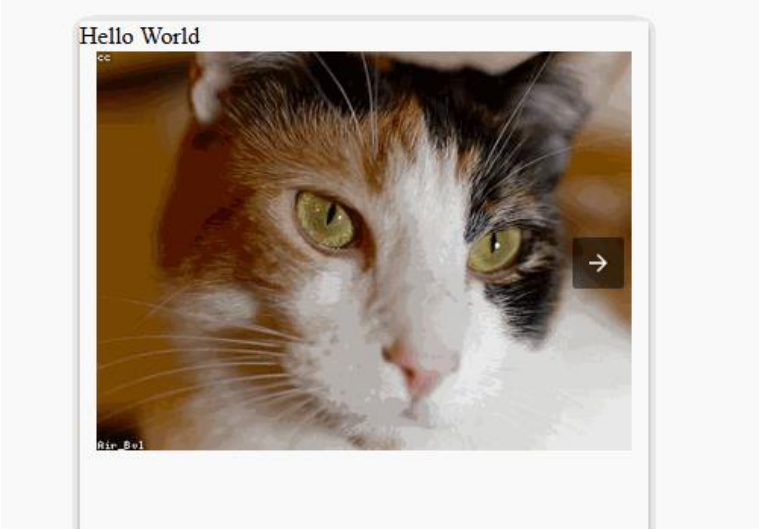
---

[128] *https://ignitevisibility.com/amp-for-email/*
[129] *https://www.emailmonday.com/google-amp-for-email-everything-you-wanted-to-know/*
[130] *https://www.lafabriquedunet.fr/email-marketing/articles/google-amp-emails-interactifs/*
[131] *https://www.blog.google/products/g-suite/bringing-power-amp-gmail/*
[132] *https://blog.internet-formation.fr/2018/02/google-introduit-amp-for-email-pour-gmail/*
[133] *https://www.presse-citron.net/amp-google-veut-rendre-e-mails-plus-interactifs/*
[134] *http://blog.agence-bmobile.com/2018/02/amp-email-ce-que-le-gmail-interactif-va-peut-etre-changer-pour-vous/*
[135] *https://www.youtube.com/watch?v=sj2KANF9o3k*

Version texte brut



Version HTML



Version AMP

La version AMP n'affectera pas les versions HTML ou en texte brut de l'email. L'AMP dans l'email marketing prendrait en charge la plupart des composants AMP déjà utilisés pour les pages AMP, notamment :

- **Contenu dynamique** *(amp-from, amp-selector, amp-bind, amp-state, amp-list, amp-mustache)*
- **Présentation** *(amp-acordion, amp-carousel, amp-sidebar, amp-image-lightbox, amp-lightbox, amp-fit-text, amp-timeage)*
- **Media** *(amp-img, amp-anim)*

AMP for Email est un sous-ensemble de AMP qui, comme AMP, offre des fonctionnalités de type Javascript pour les emails. Comme tous les clients de messagerie bloquent Javascript par défaut, AMP offre une alternative limitée et sécurisée à Javascript sans donner aux expéditeurs la possibilité d'exécuter du code dans leur courrier électronique. La spécification « AMP HTML pour Email spec » consiste à ajouter une nouvelle partie MIME « text-x-amphtml » à votre adresse e-mail contenant le balisage HTML AMP. Le raisonnement est que les clients de messagerie compatibles AMP rendraient alors cette partie au lieu de la partie HTML et que d'autres clients non-AMP continueraient à rendre la partie HTML.

Controverses[136]

Si l'interactivité semble être un avantage dans un email, il peut devenir un inconvénient pour la consultation des sites Web. AMP brouillerait la frontière entre les courriels et les sites Web. Ses détracteurs voient aussi dans cette technique un autre moyen pour Google de contrôles et de s'insérer dans l'expérience web de ses utilisateurs : les motivations seraient donc moins centrées « expérience utilisateur » qu'il n'y semblerait.

A ce jour, **AMP pour email est uniquement disponible pour les développeurs qui en demandent l'accès**, mais Google prévoit de le déployer sur Gmail plus tard dans l'année. Google espère aussi que les autres clients de messagerie adopteront également AMP. C'est là-aussi que le bât blesse : non seulement, en développant cet outil, Google demande aux intégrateurs emails de réaliser un nouveau travail en plus de celui qu'ils devaient accomplir jusqu'à présent. Mais si Google se l'autorise, Microsoft ou un autre système pourrait tout aussi bien créer son propre langage. La complexité d'une campagne email n'en sera que plus encore affectée.

Conclusion

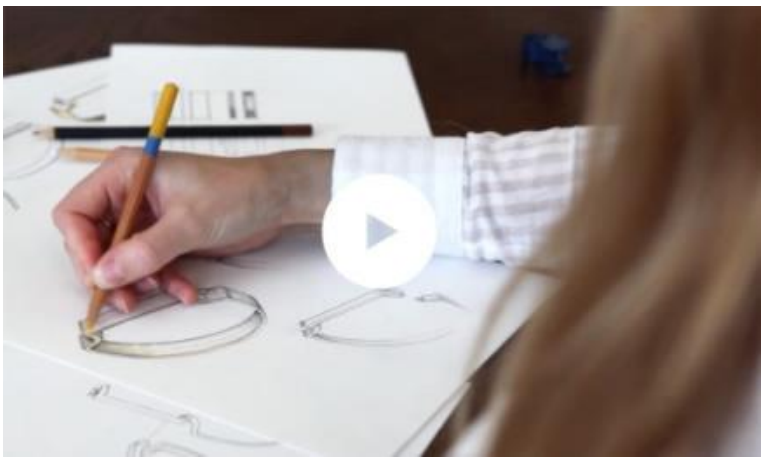
Il reste encore beaucoup de zones d'ombre quant à l'AMP : toutes les actions effectuées depuis un mail codé avec AMP seront-elles traçables ? Les expéditeurs auront-ils accès à des analyses leur indiquant qui a interagi avec leurs emails ? Au-delà de cela, il est difficile de savoir s'il y aura une option pour les utilisateurs de désactiver AMP en faveur des emails HTML normaux. De même, la date de sortie de ce langage et son support sont pour le moment encore flous. Enfin, les « emailgeeks » insistent sur le fait qu'il serait intéressant de **se concentrer sur les normes HTML et CSS existantes** pour autoriser les formulaires, les animations CSS et l'interactivité au lieu d'étendre une spécification controversée et contrôlée par Google à l'environnement de messagerie.

[136] *http://www.emailmarketingtipps.de/2018/02/19/amp-for-email-controversies/*

Médias intégrés

La vidéo est engageante et divertissante. L'inclusion de la vidéo dans un email[137] [138] peut entraîner **une augmentation du taux d'ouverture de 19% et le taux de clics augmenterait de plus de 50%**. Cependant, ces médias ne s'afficheront pas dans la boîte de réception, à moins que le client de messagerie de votre abonné prenne en charge le format et les balises HTML5. Puisqu'un seul client de messagerie répandu, Apple Mail, prend en charge ces balises, il est préférable d'éviter les médias intégrés et d'envisager une alternative. Pensez aux gifs animés ou à la technique « Faux vidéo » de Kristian Robinson[139] [140]. Pour envoyer des fichiers audios, partagez un lien vers un fichier audio.

L'utilisation d'un bouton de lecture au-dessus d'une image statique est sans doute le moyen le plus simple de créer un lien vers un contenu vidéo hébergé sur site.



Exemple de player sur visuel

Un autre moyen très facile de donner l'illusion d'une vidéo dans un email consiste **à utiliser un gif animé**. Le format gif est très largement pris en charge, gardez cependant à l'esprit qu'il ne fonctionne pas partout : Dans Outlook 2007, 2010, 2013 et Windows Phone 7, seule la première image de votre fichier GIF sera affichée. Assurez-vous alors que toutes les informations clés se trouvent dans cette première image. Soyez aussi attentifs au poids de votre gif animé.

[137] *https://litmus.com/blog/the-pros-and-cons-of-video-in-email-video*
[138] *https://www.campaignmonitor.com/resources/guides/video-in-email/*
[139] *http://freshinbox.com/blog/faux-video-video-in-email-using-sprites/*
[140] *https://codepen.io/kristianrobinson/pen/EwERve*

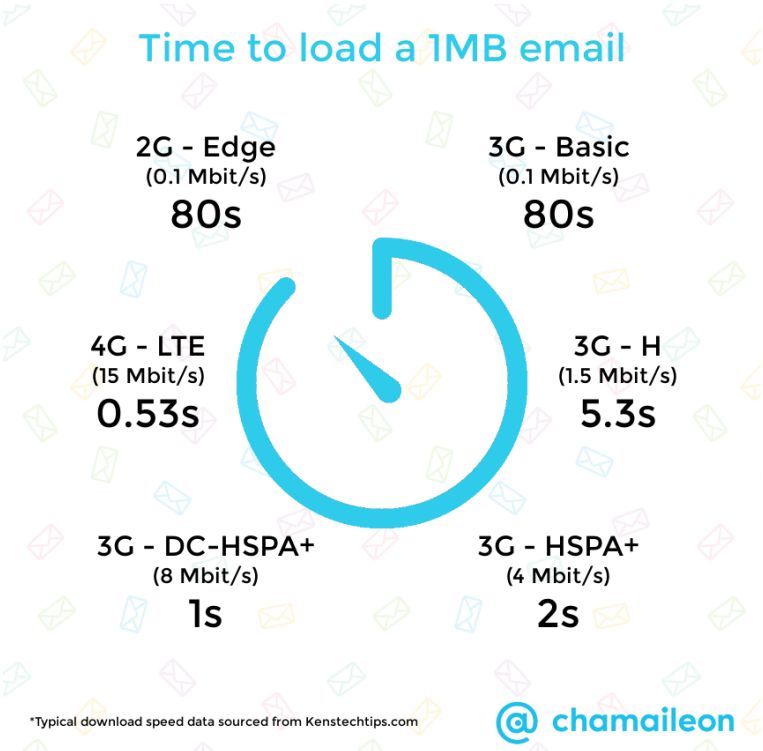
Partie V

Sujets à discorde

Poids

INTRODUCTION

Le poids d'un fichier HTML comme des images rattachées est un sujet particulièrement important dans l'emailing[141]. Cela impacte plusieurs points non négligeables d'une bonne campagne : sa délivrabilité, son temps de chargement. Pour se faire une idée du temps de chargement nécessaire à l'affichage d'un email, voici une petite illustration des temps de chargement d'un email de 1 Mo sur les réseaux mobiles. En analysant ce document, **nous vous recommandons de conserver un poids d'email** *(images + fichier HTML)* **inférieure à 1 Mo** *(de préférence autour de 500 ko)* pour offrir une expérience de chargement appropriée, même pour ceux qui lisent vos emails avec une vitesse de téléchargement limitée.



Statistiques de temps de chargement d'un email – Source : Chamaileon

DES IMAGES

Les images sont une ressource considérable dans le temps de chargement d'un email. Si un email peut sembler long à télécharger et à s'afficher, cela peut être dû au poids des images rattachées. Les débits de connexion sont variables, et avec les supports mobiles dorénavant omniprésents, il est indispensable d'optimiser au maximum le nombre et le poids des images pour en améliorer la vitesse de chargement. Pour ce faire :

- Privilégiez les bons formats d'image : GIF, JPEG, PNG. Sachez choisir l'extension selon la qualité, le contenu, et le rendu souhaité de l'image.
- Utilisez des « optimiseurs » d'image pour réduire le poids, tout en gardant la meilleure qualité possible.
- Assurez-vous que le serveur sur lequel sont hébergées vos images seront en mesure de les afficher rapidement à grande échelle.
- Utilisez le moins d'images possible : chaque image que vous incluez dans votre email sera téléchargée de votre serveur. Si vous avez 20 images, cela signifie 20 requêtes au serveur.

[141] *https://blog.chamaileon.io/limitations-of-html-email-design-email-width-and-size/*

DU FICHIER HTML

Faut-il se restreindre à un certain poids concernant le fichier HTML ? Gardez à l'esprit : less is more ! Si le poids d'un fichier HTML est réduit à son minimum, son temps de chargement ne s'en trouvera qu'optimisé ! De plus, il faut savoir que Gmail tronque tout contenu après les 102ers kilooctets d'un email.

Pour réduire le poids d'un fichier HTML, il faut tout d'abord y penser en amont, avant tout début d'intégration : la méthode d'intégration *(codage responsive, hybrid)* peut impacter le poids total d'un fichier HTML. Il faudrait presque penser au poids du fichier bien plus en amont encore, lors de la création graphique de l'email : un email très « haut », avec une version responsive complexe, nécessitant de nombreuses media queries, devra attirer votre attention !

Pour réduire le poids du fichier HTML, supprimez toutes les media queries, styles, attributs, propriétés, commentaires qui ne seraient pas absolument nécessaires. Vous pouvez aussi décider de « minifier » votre fichier HTML, en supprimant l'indentation, mais assurez-vous que les propriétés CSS et le code HTML resteront inchangés.

Hauteur

Outlook 2007 et 2010 utilisent le moteur de rendu de Microsoft Word pour afficher un email. Microsoft Word propose différents modes d'affichage. Outlook utilise la vue « Web ». **Au sein de ce mode existe des limites de texte[142]**. Si la limite est dépassée, un retour à la ligne est automatiquement inséré. La dernière valeur remontée était de 23,7 pouces, soit environ 1790 pixels. Pour ne pas rencontrer ce problème, coupez votre intégration en tableaux séparés.

Sémantique

INTRODUCTION

Le HTML sémantique est l'utilisation du balisage HTML visant à renforcer le sémantisme *(la signification)* des informations contenues, c'est-à-dire leur sens, plutôt que de se borner à définir leur présentation *(ou apparence)*. Le HTML sémantique est traité par les navigateurs courants, mais aussi par de nombreux autres agents utilisateurs.

Par exemple, les spécifications HTML récentes déconseillent l'utilisation de la balise <i>, qui indique un style de police italique, au profit de balises sémantiquement plus précises, telles que , qui indique une mise en valeur. C'est la feuille de style CSS qui spécifie ensuite si cette mise en valeur est représentée par un style italique, un style gras, un soulignement, une prononciation plus lente ou plus forte, etc. Le fait de baliser les informations de manière différente permet aux agents Web *(ou user-Agent)* tels que les moteurs de recherche et autres logiciels **d'évaluer précisément l'importance du texte**.

Cependant, la sémantique dans l'email pourrait ne pas avoir d'intérêt, en ce sens où aucun référencement n'est nécessaire. Si la sémantique commence à voir le jour dans l'email, c'est principalement pour des notions d'accessibilité.

[142] *https://templates.mailchimp.com/development/css/outlook-conditional-css/*

ACCESSIBILITE

L'accessibilité[143] inclut des notions que nous ne pouvons plus ignorer désormais. L'accessibilité d'un email[144] ne réside pas seulement dans son support sur les clients mails, son design lisible, la faculté qu'il offre à comprendre l'information, ou la présence d'une version full text ou d'un lien vers une page miroir. **L'accessibilité, c'est aussi la possibilité de travailler avec les outils d'assistance[145]**. Et c'est là que la sémantique prend aussi tout son sens. Les outils d'assistance sont les lecteurs d'écrans, les loupes, les logiciels pour accentuer le contraste ou les teintes, la possibilité de naviguer à travers l'email sans l'usage de la souris…

Les nouvelles bonnes pratiques recommandent alors d'utiliser les bons outils pour une meilleure compréhension : utiliser des <p> pour des paragraphes, des <ul> et <li> pour des listes à puces, des <table> pour des tableaux… Finalement, le balisage sémantique dans un email aura une fonction d'accessibilité principalement pour les lecteurs d'écran qui, en scannant le contenu de l'email, seront alors en mesure de définir correctement les titres, les listes, les textes sur lesquels l'accent est mis, les textes spécifiques, les citations, etc…

[143] https://fr.slideshare.net/M_J_Robbins/accessibility-in-email-eoainsights
[144] https://fr.slideshare.net/paulairy/a-type-of-accessibility-65004974?next_slideshow=1
[145] https://litmus.com/blog/ultimate-guide-accessible-emails

Partie VI

Clients mails et OS

Introduction

Les développeurs Web prennent en compte une poignée de navigateurs lors de la conception et de la création d'un site Web. La conception de sites Web basés sur des normes a amené la plupart des navigateurs à une parité relative. Mais dans le domaine du courrier électronique, il existe des dizaines de clients de messagerie[146], dont certains existent depuis des décennies. Les grandes entreprises, dont la mise à jour est lente, ont toujours des employés qui s'appuient sur des clients obsolètes tels que Lotus Notes ou Outlook 2000.

Outre les logiciels de messagerie, les webmails présentent deux fois plus de problèmes : vous êtes forcés de prendre en compte la façon dont le client de messagerie lui-même affiche et interprète les langages HTML et CSS, mais aussi comment le navigateur dans lequel le client est affiché interprète ces mêmes langages. Un email affiché dans Gmail sur Firefox peut sembler très différent d'un e-mail dans Gmail sur Internet Explorer 8.

Avec la croissance exponentielle du mobile, une nouvelle scène pour les clients mail s'est matérialisée. Voici une liste non exhaustive des clients de messagerie les plus utilisés[147].

Clients desktop

Les clients Desktop ou logiciels de messagerie sont la manière traditionnelle de lire le courrier électronique, car ils sont fortement liés aux environnements de bureau. **L'utilisation réelle est dominée par Outlook**. Mais à mesure que le nombre de lecteurs mobiles augmente, la consultation sur ordinateurs de bureau continue de connaître une baisse significative.

MICROSOFT WINDOWS

Le logiciel Outlook de Microsoft est de loin le plus populaire au monde. Pendant des années, Outlook a été intégré au système d'exploitation Windows, ce qui a assuré son utilisation à grande échelle dans les foyers et les entreprises.

Outlook 2007/2010/2013

En 2007, Microsoft a choisi de remplacer le moteur de rendu HTML d'Outlook jusqu'alors basé sur Internet Explorer 6 par un moteur de rendu basé sur Word. En conséquence, Outlook 2007 a support CSS catastrophique. Les versions 2010 et 2013 souffrent du même problème.

Tips 'N Tricks

Commentaires conditionnels

Les commentaires conditionnels[148] permettent en réalité de fournir des styles ou des éléments HTML spécifiques à Outlook. Le fonctionnement est identique à celui du ciblage d'Internet Explorer lors de la conception d'un site Internet, mais ici nous ciblons Microsoft Office, ou « mso ». Précéder le commentaire de « gt » *(pour greater than)* ciblera les versions plus récentes à la version ciblée. Précéder le commentaire de « lt » *(pour lesser than)* ciblera les versions plus anciennes à la version ciblée. Ajouter un « e » à « gt » ou « lt » permettra de cibler les versions supérieures, inférieures ou égales, respectivement.

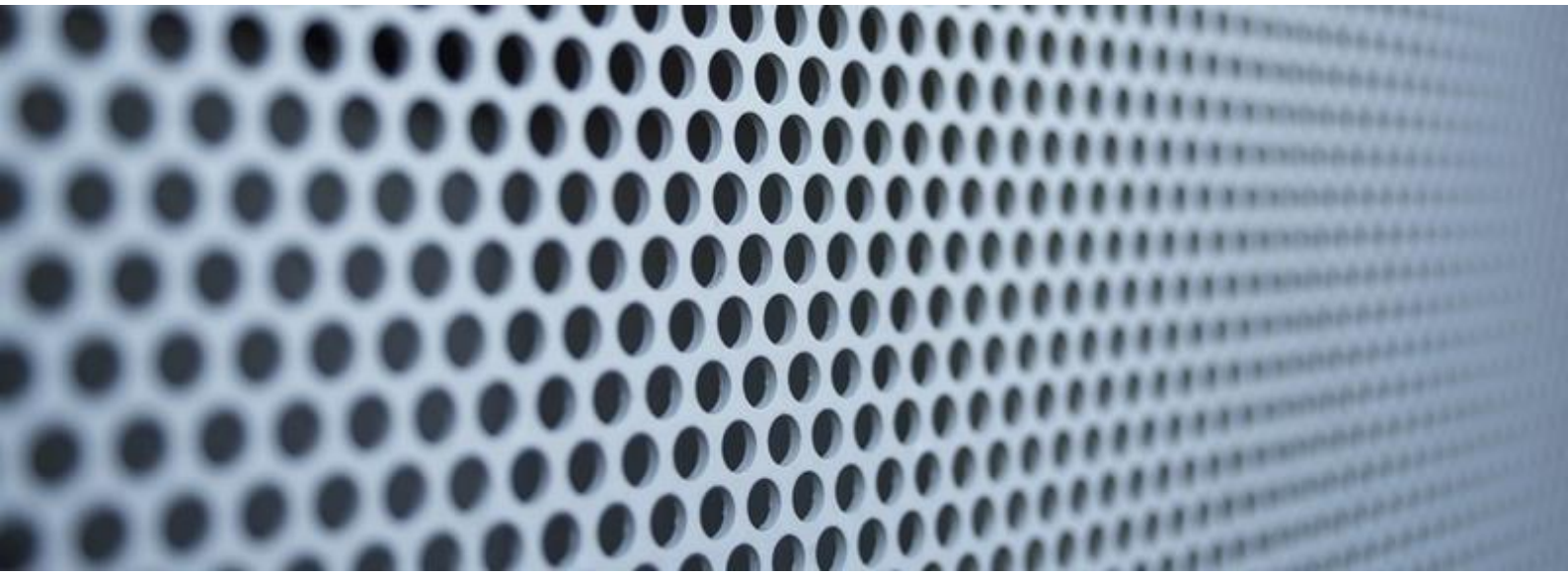
[146] *https://templates.mailchimp.com/concepts/email-clients/*
[147] *https://fr.wikipedia.org/wiki/Syst%C3%A8me_d%27exploitation*
[148] *https://templates.mailchimp.com/development/css/outlook-conditional-css/*

```
<!--[if gte mso 9]>
    <style type="text/css">
    /* Your Outlook-specific CSS goes here. */
    </style>
<![endif]-->
```

Dans l'exemple ci-dessus par exemple, « gte » est ajouté pour que le commentaire s'applique uniquement aux versions de Microsoft Office supérieures ou égales à la 9ème version. La Version 9 d'Outlook correspond à Office 2000. Voici d'ailleurs la liste des différentes versions et de leurs correspondances :

- Outlook 2000 : Version 9
- Outlook 2002 : Version 10
- Outlook 2003 : Version 11
- Outlook 2007 : Version 12
- Outlook 2010 : Version 14
- Outlook 2013 : Version 15

Les commentaires conditionnels sont remarquables pour éliminer les bugs et réduire considérablement les problèmes de codage liés à Outlook.



DPI

DPI pour « dots per inch ». En français : points par pouce.
Les écrans d'ordinateurs sont habituellement configurés en 72 dpi[149] *(du moins c'est ce qui est fixé dans les années 1990)*, soit 72 pixels sur 72 pixels, soit 5184 pixels dans un carré de 2,54 cm de large sur 2,54 cm de haut. Cela évolue en 2011, où la résolution fictive de 96 dpi prend le dessus[150]. **Mais une résolution en 120 dpi devient le paramètre par défaut sur la plupart des nouveaux ordinateurs portables haute résolution** *(HiDPI devices) (ou peut-être que la machine est ainsi configurée pour des raisons d'accessibilité)*. Cependant, lorsqu'un DPI supérieur est utilisé, tous les éléments textuels et graphiques sont mis à l'échelle, comme lorsque vous appliquez un zoom sur les textes depuis Mozilla Firefox par exemple.

[149] *http://sebsauvage.net/comprendre/dpi/index.html*
[150] *https://www.lesintegristes.net/2011/05/06/web-resolution-72dpi/*

Or, comme Outlook utilise Microsoft Word pour le rendu d'un email, le rendu sur Outlook en 120 DPI sera donc affecté par ce changement. Le code sera modifié, mais seulement sur certaines parties, tout en laissant d'autres parties inchangées. Concrètement, les valeurs en pixels des attributs width et height des <table> et <td> resteront inchangées *(donc, en pixels)* quand les autres valeurs en pixels de l'email seront transformées en points *(la propriété font-size par exemple)*. Cette combinaison de valeurs non évolutives *(pixels)* et de valeurs évolutives *(point)* crée des problèmes de rendu que nous apercevons sur Outlook en 120 DPI.

Une solution existe pour pallier ces problèmes de rendu :

1. **Ajouter un XML Namespace au tag <html>**. Ce point est essentiel pour qu'Outlook puisse interpréter la suite des modifications à effectuer.

```
<html lang="en" xmlns="http://www.w3.org/1999/xhtml"
xmlns:o="urn:schemas-microsoft-com:office:office">
```

2. **Corriger le DPI pour les images**. Nous allons ajouter un morceau de code spécifique à Outlook juste avant la fermeture de la balise <head>. Ce morceau de code nous permet de forcer le paramètre DPI souhaité, soit 96.

```
<!--[if gte mso 9]>
<xml>
    <o:OfficeDocumentSettings>
    <o:AllowPNG/>
    <o:PixelsPerInch>96</o:PixelsPerInch>
    </o:OfficeDocumentSettings>
</xml>
<![endif]-->
```

3. **Utiliser les largeurs et hauteurs en propriété CSS** en plus / ou à la place des attributs. Cela alourdit un peu plus la tâche et le code HTML, mais c'est indispensable *(et finalement, il ne s'agit que d'une bonne habitude à prendre)*. Cela est seulement nécessaire lorsque les largeurs et hauteurs sont définies en pixels bien sûr, puisque le pourcentage est déjà en soit une valeur évolutive. Ainsi :

```
<table border="0" cellspacing="0" cellpadding="0" width="600">
```

Deviendra :

```
<table border="0" cellspacing="0" cellpadding="0" style="width:
600px;">
ou
<table border="0" cellspacing="0" cellpadding="0" style="width:
600px;" width="600">
```

Et il en va de même pour les cellules[151] ! Gardez ce correctif en mémoire pour tout ce qui pourrait contenir des attributs avec l'unité « pixel » finalement : cellpadding, cellspacing, codes spécifiques Outlook pour les emails hybrides…

[151] *http://www.courtneyfantinato.com/correcting-outlook-dpi-scaling-issues/*

Image de fond

La mise en place d'images de fond sur Outlook est un exercice particulièrement périlleux. Une technique particulière fait appel au langage VML[152] *(Vector Markup Language)* de Microsoft pour garantir l'affichage des images d'arrière-plan dans la suite de clients de messagerie Outlook de Microsoft. Campaign Monitor a mis au point un générateur d'image de fond « à l'épreuve des balles »[153] s'appuyant sur cette technique.

Image de fond et DPI

La largeur en pixels et la hauteur renseignées dans la cellule comprenant l'image de fond en question ne sont pas correctement retranscrites sur 120 DPI, et ce, même si elles sont bien renseignées en propriétés CSS. De ce fait, la cellule reste à la largeur et hauteur défini, et le contenu est complètement sectionné.

Pour cela, une astuce existe. Remplacez le tag <html> en tête de code par le morceau de code suivant :

```
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en"
xmlns:v="urn:schemas-microsoft-com:vml" xmlns:o="urn:schemas-microsoft-
com:office:office">
```

Et ajoutez ensuite, juste avant la fermeture du tag <head>, le morceau de code suivant :

```
<!--[if mso]>
<xml xmlns:w="urn:schemas-microsoft-
com:office:word"><w:WordDocument><w:AutoHyphenation/></w:WordDocument></xml>
<![endif]-->
```

Call to Action

Les bonnes pratiques d'email marketing veulent que tout « Call to action »[154] soit généré en HTML et CSS : cela permet tout d'abord un affichage immédiat de ces contenus primordiaux sans téléchargement ou affichage des images nécessaires. De plus, cela permet aussi de moduler la taille de l'élément plus facilement à l'aide de media queries sur la version responsive de l'email, mais d'améliorer aussi l'accessibilité de l'email, étant donné que les données de l'élément seront, peut-être, plus facilement lisibles et mieux interprétées par les lecteurs vocaux ou autres outils d'accessibilité.

Cependant, un CTA implique parfois une méthodologie d'usage et reconnaissance visuelle quelque peu « rentrée dans les normes » et dans les habitudes du destinataire *(bordure, bords arrondis, bouton entièrement cliquable)*. Autant de propriétés graphiques et d'interactions difficiles à reproduire dans un client mail complexe comme Outlook. En effet, la propriété « display :block », qui permettrait de rendre la balise <a> entièrement cliquable en lui attribuant une largeur, hauteur et line-height fixe, est laborieuse à mettre en place dans un email. De plus, les propriétés CSS permettant de reproduire l'effet de bord arrondi seront, là-aussi, difficilement interprétées sur la plupart des clients mails.

---

[152] *https://fr.wikipedia.org/wiki/Vector_Markup_Language*
[153] *https://backgrounds.cm/*
[154] *https://litmus.com/blog/a-guide-to-bulletproof-buttons-in-email-design*

Campaign Monitor a là-aussi grandement simplifié la vie des intégrateurs emailing en proposant une plateforme[155] générant un code, selon quelques critères renseignés, code qui donnera un rendu plus que correct sur la majorité des clients mails attendus. Sur Outlook, les effets et rendu souhaités seront obtenus à l'aide de commentaires conditionnels et d'un code VML. Le principal inconvénient de cette solution est la duplication (selon les critères renseignés) du bouton, soit la génération de deux liens *(identiques)* à modifier lors d'un update. De plus, la plateforme ne renseigne pas d'attribut target à cet élément. Enfin, la difficulté et les limites viennent du fait que la largeur du bouton est fixe et ne s'adapte pas à son contenu, et qu'un CTA avec plus d'une ligne de texte ne peut être envisagée.

Sachez aussi qu'il ne vous sera pas possible *(pour le moment)* de cumuler l'utilisation des « Bulletproof buttons » et des « Bulletproof background images », car Outlook ne prend pas en charge les éléments VML imbriqués.

Cellule vide

Sur Outlook 2013, les cellules de tableau vides[156] [157], ou les cellules contenant un   ; ont une hauteur minimale d'environ 15 pixels, quel que soit l'attribut height que vous avez défini.

Cela peut être gênant lorsque vous utilisez des cellules vides pour créer des marges entre des paragraphes, des marges internes de cellules, ou, pourquoi pas, des lignes horizontales. Mais il existe une solution. Si vous avez une cellule vide la sorte :

```
<td height="5" bgcolor="#cccccc"></td>
```

Outlook donnera donc à cette cellule une taille de 15 pixels de haut minimum, sauf si vous ajoutez un style="line-height:5px; font-size:5px;" où 5px serait la hauteur de la cellule que vous recherchez. Ce qui nous donne donc :

```
<td height="5" bgcolor="#cccccc" style="font-size: 5px; line-height:
5px;"> </td>
```

Margin et Float

Outlook.com supprime systématiquement les propriétés CSS float et margin[158] *(et ses dérivées)*. Nous utilisons très peu ces deux propriétés chez Badsender, mais pour autant nous devons aborder le sujet. En fait, Outlook.com ne supprime pas les propriétés Float et Margin. Ni floaT et margiN. Ni FlOaT ou MaRgIn. Ou FLOAT et MARGIN. Bref : **en changeant simplement la casse de la propriété, on passe au travers du filtrage d'Outlook.com**[159]. La syntaxe des propriétés CSS n'étant pas sensible à la casse, cela ne pose pas de problème de compatibilité ailleurs.

Google Font[160] [161]

Sur Outlook 2010 et 2013, les typos non web safe sont remplacées par la typographie Times New Roman, et ce quelles que soient les typos de secours mises en place dans le code. Une 1ère solution[162]

---

[155] *https://buttons.cm/*
[156] *https://www.campaignmonitor.com/blog/email-marketing/2012/08/outlook-2013-says-no-to-empty-table-cells/*
[157] *https://litmus.com/community/discussions/960-outlook-2013-breaks-td-height*
[158] *https://www.campaignmonitor.com/blog/email-marketing/2013/01/outlook-com-drops-margin-and-float-support-entirely/*
[159] *https://emails.hteumeuleu.fr/2014/08/outlook-com-supporte-bien-css-float-margin/*
[160] *https://litmus.com/community/discussions/36-outlook-and-fallback-fonts*
[161] *https://www.campaignmonitor.com/resources/guides/web-fonts-in-email/*
[162] *https://litmus.com/blog/the-ultimate-guide-to-web-fonts*

serait de ne cibler qu'Outlook avec des commentaires conditionnels et d'attribuer à toutes les cellules comprenant des textes en typo spécifique *(via une class par exemple)* ce code :

```
<!--[if mso]>
<style>
.myfont{
font-family: Arial, sans-serif !important;
}
</style>
<![endif]-->
```

Une autre solution existe, qui permet d'afficher correctement les typos de secours sur Outlook. Pour cela, ajoutez le style suivant dans votre code :

```
<html>
<head>
<style>
[style*="Open Sans"] {
    font-family: 'Open Sans', Arial, sans-serif !important
}
</style>
</head>
```

Puis, déplacez simplement l'appel à la typo spécifique à la fin de votre propriété font-family :

```
<td style="font-family: Arial, sans-serif, 'Open Sans'"></td>
```

Outlook 2000/2002/2003

Contrairement à ses homologues plus modernes, les premières versions d'Outlook ont tendance à mieux afficher les e-mails HTML simplement en raison de la différence de moteur de rendu. Avant 2007, les clients de messagerie Outlook étaient alimentés par le moteur Internet Explorer. Bien qu'ils soient meilleurs au rendu HTML que les clients Outlook modernes, **ces versions reposaient encore sur la technologie de navigateur Internet Explorer et partageaient donc les mêmes problèmes**.

Windows Live Mail

Live Mail est un client de bureau gratuit pour Windows, intégré à la suite logicielle Windows Essentials. Compte tenu des résultats de Microsoft, ce client de messagerie est étonnamment bon. Malheureusement, bien qu'il prenne en charge le format CSS, sa part de marché totale est relativement faible.

Lotus/IBM Notes

IBM Notes - anciennement Lotus Notes - est un client de bureau Windows doté de fonctionnalités similaires à Microsoft Outlook. **Son support CSS est tout aussi médiocre** *(voire pire)* que les dernières versions d'Outlook. Ce client mail est à exclure si l'on cherche un rendu correct d'une intégration HTML d'email. Son utilisation est d'ailleurs de plus en plus restreinte, et il représente aujourd'hui une part infime sur le marché des clients mails.

APPLE OSX

Les clients de messagerie du système d'exploitation OSX d'Apple utilisent le moteur de rendu WebKit[163] pour afficher les e-mails HTML, ce qui signifie que les e-mails rencontrent généralement peu de problèmes dans les clients de messagerie de la plateforme.

[163] *https://fr.wikipedia.org/wiki/WebKit*

Mail

Apple Mail est sans faille en tant que client de messagerie. Il est basé sur le moteur de rendu WebKit, qui alimente les navigateurs tels que Apple Safari et Google Chrome jusqu'au navigateur de la PlayStation 3. **Son support CSS est à la pointe de la technologie**, et vous ne rencontrerez que très rarement des problèmes de rendu, tant son support CSS est avancé.

Outlook

Microsoft a conçu également un client Outlook pour Mac, mais contrairement à ses cousins Windows, il est optimisé par WebKit. Cela signifie qu'il fonctionne aussi bien qu'Apple Mail avec un excellent support HTML et CSS.

Entourage

Entourage est un autre client de messagerie de bureau Microsoft, et est le prédécesseur d'Outlook pour OSX. Entourage 2008 a un moteur de rendu HTML basé sur WebKit, et avec cela, un support CSS assez solide. Microsoft a cessé de développer Entourage en faveur d'Outlook pour Mac.

MULTIPLATEFORMES

Thunderbird

Thunderbird, un client de messagerie de Mozilla, le fabricant de Firefox, dispose d'une très faible part de marché. Il s'agit néanmoins d'un client mail solide qui prend plutôt bien en charge le support HTML et CSS.

WEBMAILS

Un webmail ou solution de messagerie en ligne est le troisième moyen le plus populaire de recevoir et de lire des messages électroniques, après le mobile et le bureau. L'augmentation du nombre de lecteurs mobiles a nui aux webmails, dont l'utilisation continue de baisser modérément.

Outlook.com/Hotmail

Outlook.com, anciennement Hotmail, est l'offre de messagerie Web de Microsoft. C'est également le client de messagerie Web le plus populaire, ce qui signifie qu'il influe grandement sur la conception du courrier électronique en général. Malheureusement, il souffre de quelques bizarreries qui peuvent être gênantes pour la mise en page de votre email.

Yahoo! Mail

Le deuxième webmail en termes de popularité après Outlook.com/Hotmail, ce qui en fait un client important à prendre en compte lors de la conception et de la création de votre courrier électronique. Le support CSS de Yahoo est parmi les meilleurs.

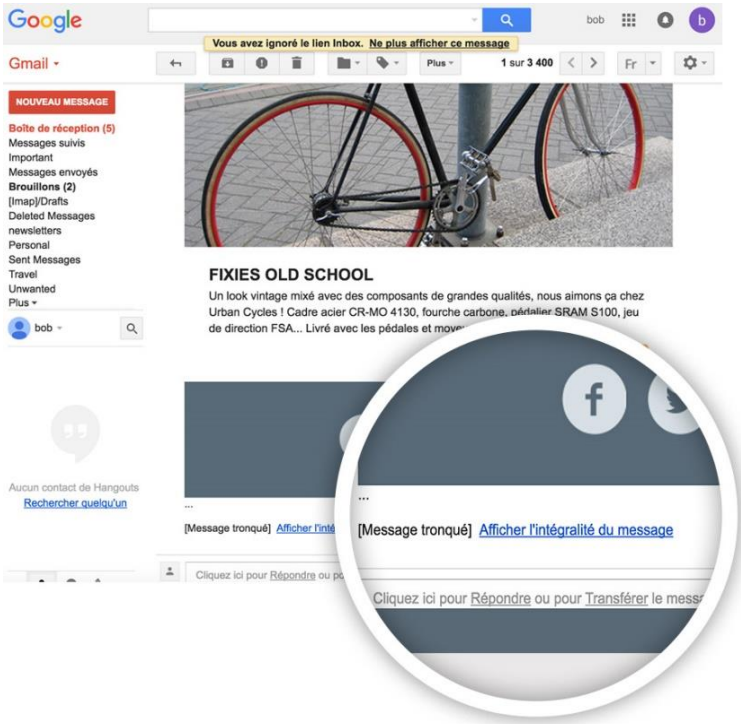
Gmail

Tips /N Tricks

Limite des 102ko

Si un mail excède un poids de 102ko, **Gmail affichera les 102 premiers kilooctets du message**[164], puis génèrera un texte informant que le message a été tronqué et vous invitera à cliquer sur le message « *Afficher l'intégralité du message* ».

[164] *https://audreytips.com/gmail-emailing-tronque/*



Capture d'écran du message de Gmail – Source : email-designer.net

Cela demande déjà une première action au destinataire, et il faut essayer de limiter au maximum les interactions, et aller au plus vite au message en lui-même. Avec ce système, le destinataire n'a pas accès directement au contenu, vous risquez de perdre son attention et donc, de le perdre tout court. **Cela affectera le taux de clics de vos campagnes.**

De plus, le lien de désabonnement généralement situé en bas de mail n'est du fait plus visible, ce qui signifie que, techniquement, l'email n'est plus compatible avec les obligations légales du CAN-SPAM[165] et du RGPD … Enfin, le code de suivi que vous placez généralement en bas de mail aussi est absent : vous êtes dans l'impossibilité de connaître le taux d'ouverture précis de votre campagne. Sur mobile, il arrivera régulièrement qu'il faille « recharger » l'email, ou scroller vers le bas, et parfois, aucune indication n'apparaît.

Est-ce possible de concevoir des emails avec un poids inférieur à 102ko ? Il faut déjà savoir que **Google ne prend en compte que le code HTML, et non les images**. Mais 102ko sont vite atteints, surtout lorsqu'il s'agit de templates avec de nombreux modules, de versions responsives avec des media queries conséquentes, etc.

Quelques conseils pour rester dans les clous :

- **Evitez de surcharger inutilement votre code** : Supprimez toute propriété, cellule, tableau, ligne, media querie qui pourraient être superflue *(padding à la place des td vides par exemple pour des marges internes)*.
- **Gardez le message de votre campagne simple**, engageant, sans trop d'informations. Assurez-vous de ne diffuser que les informations vraiment nécessaires.
- **N'envoyez pas plusieurs campagnes avec le même objet**, car Gmail combine ces emails dans une même « discussion », ce qui peut rendre le message plus lourd que 102ko.

[165] *http://www.badsender.com/2009/08/03/can-spam-act-spam-loi-etats-unis-optin-optout/*

Si vous ne parvenez pas à réduire le message à moins de 102ko, alors prenez les bonnes décisions : proposez à tout prix un lien vers une page miroir dans les premiers 102ko, insérez votre pixel de suivi dans l'en-tête de l'email *(même si ce n'est pas très esthétique)*, et placez le lien de désabonnement dans les 102ers ko aussi !

AOL

AOL est sans doute le meilleur client de messagerie Web : Il a le meilleur support CSS, y compris celui pour les images d'arrière-plan, le positionnement et l'affichage, et même certaines propriétés CSS3. Malheureusement, AOL dispose d'une base d'utilisateurs restreinte.

Clients mobile

APPLE IOS

La plate-forme iOS d'Apple n'est pas limitée à l'iPhone. Il comprend également l'iPad et l'iPod Touch, tous deux pouvant recevoir des e-mails. L'application de messagerie iOS est basée sur WebKit, comme son grand frère OSX. Sa prise en charge de CSS est incomparable, ce qui vous laisse une grande liberté dans la conception de vos emails. Le courrier iOS a cependant quelques petites particularités dont vous devez être conscient :

- Il nécessite la propriété CSS min-width: 100% pour englober l'ensemble de l'email.
- Il a besoin de la propriété CSS webkit-text-size-adjust.
- Les styles graphiques des adresses, numéros de téléphone, dates ou personnalités du monde musical sont remplacés par des liens automatiques bleus avec soulignement.

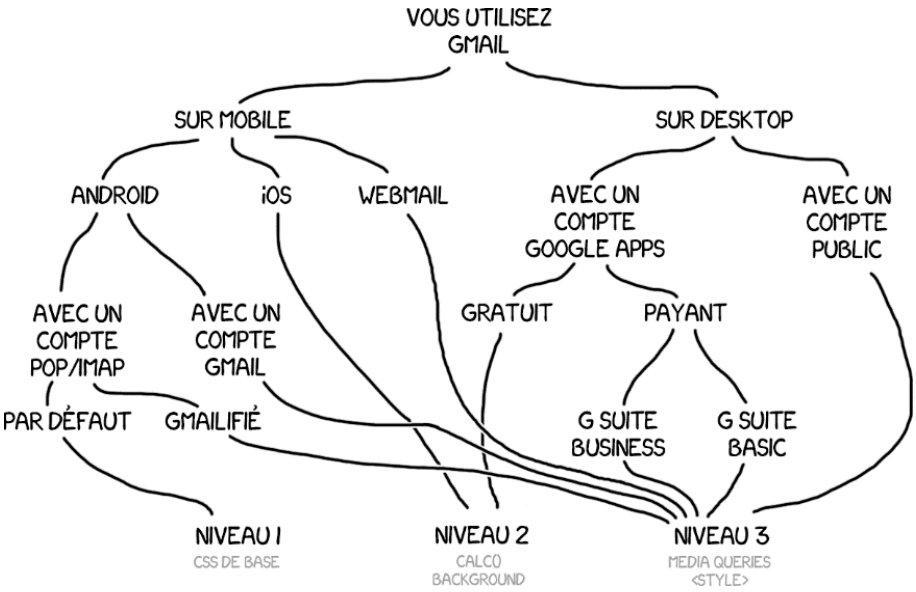
GMAIL APP

A la question : *« Est-ce que ça fonctionne dans Gmail ? »*, nous répondrons : *« Ça dépend… Sur quelle version ? »*.

Il faut prendre conscience que le support CSS dépend en fait de multiples facteurs. Google a plusieurs clients mails officiels pour Gmail *(G Suite Basic, G Suite Business, G Suite for Education, G Suite for Nonprofits, etc…)*. Mais l'application Gmail sur Android permet aussi d'utiliser n'importe quelle adresse de n'importe quel fournisseur de messagerie depuis novembre 2014. Et en février 2016, Google a aussi introduit une fonctionnalité pour « Gmailifier »[166] une adresse et activer ainsi certaines fonctionnalités de Gmail, même sur des adresses tierces.

Depuis son update en 2016, Gmail est donc censé garantir un meilleur support des propriétés CSS [167]. Or, selon le type de client, le support CSS n'est pas le même. Donc, selon l'utilisation de Gmail, vous trouverez différents types de support CSS, que l'on pourrait alors résumer ainsi :

[166] *https://blog.google/products/gmail/gmailify-best-of-gmail-without-gmail/*
[167] *https://developers.google.com/gmail/design/reference/supported_css*



Support CSS par Gmail après la mise à jour de 2016 – Source : emails.hteumeuleu.com[168]

Les différents niveaux ici évoqués sont des niveaux établis par Rémi Parmentier sur le support du CSS.

- **Le niveau 1 correspondrait ainsi à du CSS « de base »**. Ce niveau ne concerne désormais plus que l'application Gmail Android configurée avec un compte tiers *(surnommé GANGA pour Gmail Android with a Non Gmail Account)*.
- Le niveau 2 correspond, quant à lui, au **niveau 1 + le support de la propriété calc() et des background images**.
- Quant au niveau 3, il s'agit donc du **niveau 2 + le support de la balise <style>**, donc le support des media queries, des attributs « id » et « class », des sélecteurs CSS... C'est donc aujourd'hui la majeure partie des versions de Gmail qui sont à ce niveau de support, et c'est une très bonne chose en soit !

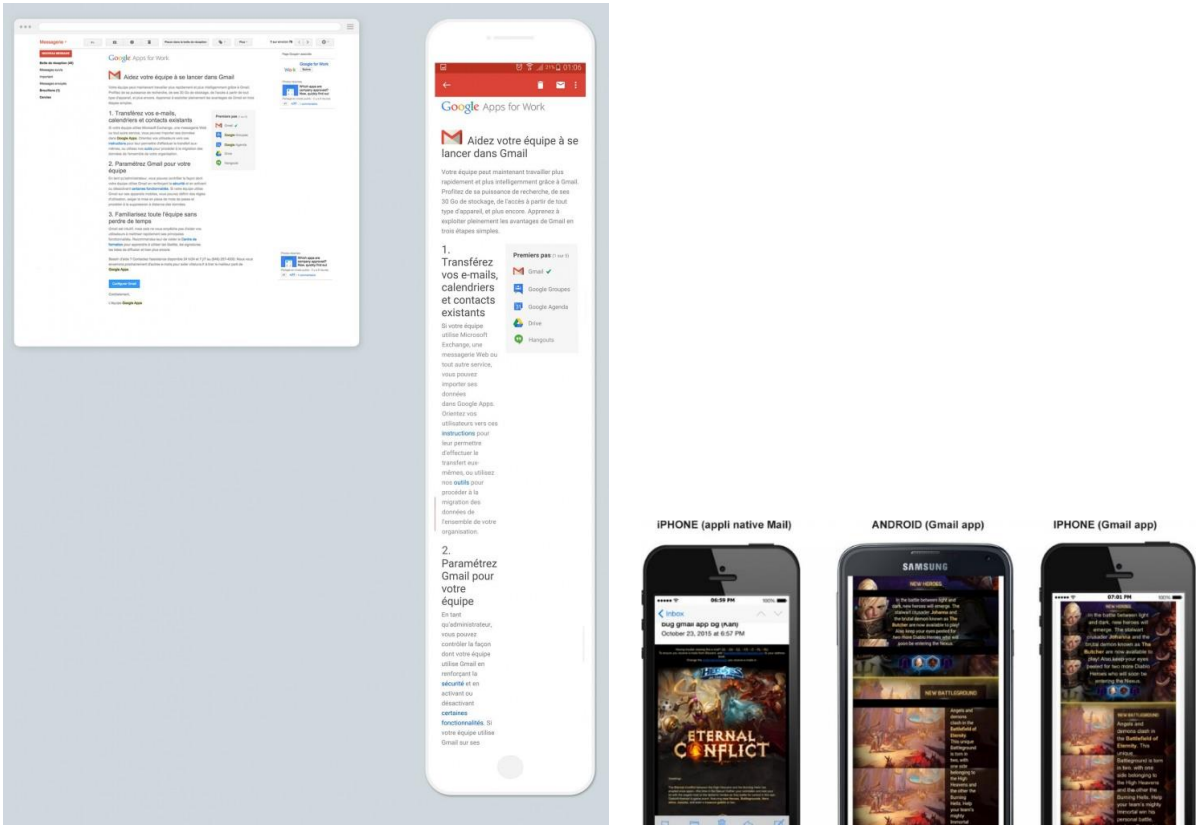
Mais quelle sera la part de consultation de vos campagnes depuis un mobile Android avec un compte POP/IMAP par défaut ? D'où notre volonté farouche chez Badsender de défendre une solution Fluid pour proposer, malgré tout, une version mobile adaptée sur support mobile même lorsque les media queries ne sont pas supportées.

Tips 'N Tricks

Forcer rendu desktop sur Gmail app

Il ne reste donc plus que très peu de types d'utilisation de Gmail qui n'interprètent pas les <style> et les media queries. Mais cela peut tout de même représenter encore une grosse part de consultation sur vos campagnes. Et si vous ne codez pas vos emails en Fluid, vous constaterez que Gmail app redimensionne automatiquement la mise en page de votre emailing à la taille du smartphone *(ce qui peut provoquer de sérieux problèmes de rendu : les colonnes sont bien plus étroites si une mise en forme en colonne est utilisée bien évidemment, les images de fond vont se répéter, des cassures apparaissent…)*.

[168] *https://emails.hteumeuleu.com/trying-to-make-sense-of-gmail-css-support-after-the-2016-update-53c15151063a*



Captures d'écran – Source : email-designer.net

Si vous souhaitez forcer Gmail à afficher la version Desktop *(sur les supports qui ne supportent pas les media)*, il existe deux fixs : l'un propre à Android, le second pour iphone. Cette solution n'est pas forcément idéale, puisqu'il faut parfois zoomer pour lire certains contenus, mais au moins **la mise en forme reste cohérente** puisque la version Desktop sera affichée à l'échelle du smartphone.

Pour Android :

- Créer un spacer *(gif transparent d'un pixel)*
- Créer une class .hide {display :none ;} dans l'entête du document HTML pour masquer le spacer sur les autres clients de messagerie.
- Insérer le morceau de code HTML suivant dans le tableau principal de votre email.

```
<tr>
  <td height="1" class="hide" style="min-width:600px; font-size:0px; line-height:0px;"><img height="1" src="spacer.gif" style="min-width: 600px; text-decoration: none; border: none; -ms-interpolation-mode: bicubic;" /></td>
</tr>
```

Pour iphone :

- Créer une class .gmailfix {display :none ; display :none !important} dans l'entête du document HTML.
- Insérer le morceau de code HTML suivant en bas de votre code HTML.

```
<div class= « gmailfix » style=" white-space:nowrap; font:15px
courier; line-height:0;">           
                 
                 
           </div>
```

Width= « 100% »[169] [170] [171] [172] [173]

Tout récemment nous avons constaté sur l'application Gmail sur Android que nos tableaux, même avec un attribut width= « 100% », se limitaient en largeur à leur contenu *(une sorte de rétrécissement automatique)*. Nous ne sommes pas parvenus à reproduire le problème. Nous avions donc juste ajouté un style= « width :100% !important ; margin :0px auto ; » qui, semble-t-il, résolvait le problème sur les tests réels.

Mark Robbins postait il y a deux cela deux mois sur le forum de la communauté Litmus qu'il avait rencontré le même souci sur plusieurs campagnes, mais qu'il n'arrivait désormais plus à le reproduire. Lui aussi remarquait que Gmail app refusait de respecter la largeur 100% attribuée à certains tableaux et adaptait la largeur au contenu. Mark Robbins ajoutait même que, même lorsqu'un style= « width :100% » était mis en place, cela ne résolvait pas le souci. En revanche, il avançait le fait que l'ajout d'un style= « min-width :100% ; » attribué au tableau en question résolvait le problème.

En fait, c'est chez Mosaico que l'explication a été trouvée : L'application Gmail pour Android *(et pas sur iOS)* supprimerait parfois, sous certaines circonstances, les propriétés width et min-width des tableaux, cellules et images en ajoutant une class .munged sur les éléments concernés. Cette class .munged est définie de la sorte par Gmail App sur Android :

```
table.munged {
    width: auto !important;
    table-layout: auto !important;
}
td.munged {
    width: auto !important;
    white-space: normal !important;
}
```

Pour rentrer plus dans la technique via le code source généré par Gmail App sur Android : un script effectue une certaine logique pour ajouter cette class selon certaines variables, dont une constante WEBVIEW_WIDTH *(relative au support de consultation même)*.

```
documentWidth = document.body.offsetWidth;
goalWidth = WEBVIEW_WIDTH;
```

Le script ajoute la class, et observerait alors la largeur de l'email résultant de ce changement. Si le résultat est satisfaisant *(une constante TRANSFORM_MINIMUM_EFFECTIVE_RATIO est utilisée pour*

---

[169] *https://litmus.com/community/discussions/4624-fixed-gmail-app-not-respecting-100-width*
[170] *https://mosaico.io/email-client-tricks/gmail-android-tries-to-shrink-your-email/*
[171] *https://litmus.com/community/discussions/4410-gmail-app-stacked-column-width-woes-no-media-queries*
[172] *https://emails.hteumeuleu.fr/2016/01/gmail-app-on-android-tries-to-shrink-your-email-with-munged-classes/*
[173] *https://github.com/bago/android-gmail-emulator*

déterminer le taux de réduction réussi), la class sera conservée, sinon, le script inversera toutes les opérations.

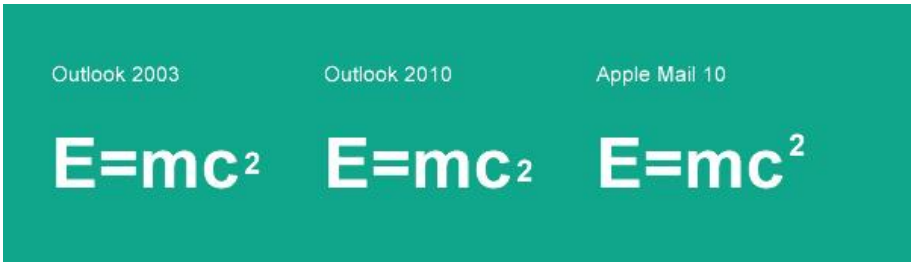
```
// If the transformations shrank the width significantly enough, leave them in
place.
// We figure that in those cases, the benefits outweight the risk of rendering
artifacts.
if (!done && (elWidth - newWidth) / (elWidth - docWidth) >
TRANSFORM_MINIMUM_EFFECTIVE_RATIO) {
    console.log("transform(s) deemed effective enough");
    done = true;
    }
```

Les class .munged étant appliquées et conservées sur certains tableaux, cellules, ou images seulement, cela peut expliquer le côté « aléatoire » de ce bug. Enfin, il faut retenir que ce problème ne concerne que Gmail App sur Android, par sur Ios. Mozaico en déduisait que cela pouvait être dû aux différences de taille d'écran et de résolutions, ou au fait que le même script n'était pas forcément utilisé sur ios.

Tips ‘N Tricks

<SUP>

Chaque client mail a une interprétation particulière *(du moins, c’est l’impression que l’on peut avoir lorsque l’on lance un test)* de cette petite balise HTML. Si la différence de rendu graphique semble souvent infime, elle peut être bien plus importante sur des textes dans une taille de typographie beaucoup plus conséquente. De même, lorsque vous souhaitez renseigner une taille de typo particulière sur la balise <sup>, le rendu peut vraiment devenir problématique.



Il suffit, pour s’assurer d’un rendu identique de cette balise sur la majorité des clients mails, de mettre en place le code HTML/CSS suivant[174] :

```
<sup style="font-size:[valeur]px; line-height:0; vertical-
align:[valeur]px">&reg;</sup>
```

<STRONG>, <EM>, <I>, <U>, <MARK>

Nombre d’intégrations HTML font utilisation intempestive des balises , , <b>, <i> ou encore <u> pour mettre en forme graphiquement *(pas pour une utilisation sémantique)* une partie de texte. Et c’est une mauvaise habitude. Effectivement, tout texte au sein d’une balise s’affichera bien en gras. Tout texte dans une balise s’affichera bien en italique, et ainsi de suite…

Mais ces balises ne sont en aucun cas vouées à une mise en forme graphique d’un texte HTML. C’est une mise en forme par défaut qui leur est pour le moment attribuée, ou l’ensemble des clients mails et navigateurs semblent unanimes sur l’affichage. Cependant, lorsque l’on consulte les définitions et usages de chacune de ces balises sur le site de w3schools[175], on constate bien que ces balises ont plutôt une fonction « sémantique ».

- <i> : Cette balise définit une partie de texte dans un autre ton ou une autre « humeur ».
- <u> : Cette balise représente un texte qui doit être « stylistiquement » différent du texte normal , tel que des mots mal orthographiés.
- : Cette balise définit un texte important.
- : Cette balise définit un texte souligné, dans le sens « l’accent est mis sur… »
- <mark> : Cette balise permet de mettre en évidence des parties d’un texte.

La balise <mark> peut déjà définitivement être supprimée de la liste : son support « graphique » n’étant pas supporté sur tous les clients mails. Quant aux autres balises gardez donc à l’esprit que leur vocation est sémantique. La sémantique au sein d’un emailing peut avoir une certaine fonction utile *(surtout en termes d’accessibilité[176] [177])*. Mais s’il ne s’agit que d’une mise en forme graphique,

[174] *http://blog.emailwizardry.co/2012/12/how-to-fix-superscript-characters/*
[175] *https://www.w3schools.com/tags/*
[176] *https://litmus.com/community/discussions/6509-html-semantic-elements-in-email*
[177] *https://litmus.com/community/discussions/1524-do-you-care-about-semantics-in-html-emails*

préférez l'utilisation d'un avec la propriété CSS correspondante : font-weight, font-style, text-decoration, background-color.

Ce point ne pose pas réellement de souci de rendu sur les clients mails pour le moment, mais il est nécessaire de l'aborder, car, à l'avenir, **peut-être que la mise en forme graphique des balises , , <i>, <u> ou <mark> pourrait changer**.

SOULIGNEMENT ET LIENS AUTOMATIQUES

Le soulignement automatique des adresses, numéros de téléphone, dates ou suite de chiffres ou de mots sur iOS, avec ce bleu électrique si caractéristique, vient parfois interférer sur votre créa[178] [179] ! Mais si la mise en forme laisse à désirer, il est tout de même intéressant de laisser le lien présent automatiquement sur ces morceaux de texte. Pourquoi ? **Parce que c'est une question d'ergonomie, de fonctionnalité et d'accessibilité !** Si un numéro de téléphone peut devenir automatiquement cliquable pour générer un appel, ou une adresse pour amener sur Google map, ou encore une date pour créer un évènement dans un calendrier, ça serait dommage de supprimer ces fonctionnalités.

Cependant, admettons que la mise en forme du texte bleu peut aussi poser des problèmes de lisibilité *(si le fond de la cellule est dans une teinte de bleu ou de violet par exemple)*. Cela peut impacter sur la lisibilité de l'email et de son contenu.

Si l'on regarde de plus près sur le code généré sur iOS, on se rend compte que, sur les appareils iOS, **l'attribut « x-apple-data-detectors » est ajouté** aux liens en question.

```
<a href="#" x-apple-data-detectors="true">
```

On peut alors appliquer en conséquence un « tip » : dans la déclaration <style>, spécifier à tous les liens avec cet attribut d'appliquer des « color », « font-size », « font-family », « font-weight », et « line-height » en inherit *(héritées de leur cellule)*.

[178] *https://github.com/FunWithEmail/HTML-Email-Hacks*
[179] *http://removebluelinks.com/*

```
a[x-apple-data-detectors] {
    color: inherit !important;
    text-decoration: none !important;
    font-size: inherit !important;
    font-family: inherit !important;
    font-weight: inherit !important;
    line-height: inherit !important;
}
```

Ainsi, ces liens automatiques hériteront automatiquement des couleurs, font-weight, line-height définies dans la <td>, tout en conservant leur fonctionnalité ! Bien sûr, n'oubliez pas le « !important » pour écraser la mise en forme automatique *(et spécifiez, si besoin, un « text-decoration :none » si vous ne souhaitez pas voir ces textes soulignés)*.

Si, malgré tout, des morceaux de texte ou chiffres devaient comporter des liens alors qu'il n'y a aucune utilité *(suite de nombre sur un code SIRET par exemple)*, ou que vous êtes sur Gmail qui depuis Septembre 2017 a appliqué les mêmes fonctionnalités qu'iOS sur les adresses et les numéros de téléphone, il existe encore un autre petit tip qui devrait vous permettre de résoudre le problème : mettez en place un autour du texte concerné :

```
<span class="appleLinks01">RCS 195 645 362</span>
```

avec une class nommée, par exemple, « appleLinks01 ». Puis, dans le <style> de votre email, spécifiez que cette class doit être dans une certaine couleur, sans text-decoration :

```
.appleLinks01 * { text-decoration:none !important; color:#000000 !important; }
```

Sur le webmail Zimbra *(utilisé par Free et par laposte.net)* : Zimbra transforme les dates en lien de couleur bleue pour connecter la date avec l'agenda intégré. Là-aussi, un petit morceau de code permet de résoudre le problème :

```
span.Object {
    color: inherit !important;
    }
    span.Object-hover {
    color: inherit !important;
    text-decoration:none !important;
    }
```

<A> INVALIDES ET OFFICE 365[180] [181]

Il y a maintenant plus de trois ans, on apprenait **qu'Office 365 supprimait tout tag <a> avec un attribut href vide** *(ou sans attribut href)*. En vérité, cela est un peu plus complexe encore : Lorsque vous insérez, au sein de votre attribut href, autre chose qu'une URL, Office 365 va faire apparaître ce contenu au sein de l'email à l'intérieur de crochets. Ainsi, le code suivant :

```
<a href="Hello">world</a>
```

Deviendra :

[180] *https://litmus.com/community/discussions/4594-outlook-365-displays-tbd-on-empty-href-tags*
[181] *https://litmus.com/community/discussions/1475-office-365-links*

Alors que Justin Khoo *(de chez Freshinbox)* et qu'Elliot Ross *(Action Rocket / Taxi for Email)* précisaient qu'ajouter un « # » *(pour une ancre)* ou un http:// *(pour une url valide)* juste avant le contenu résoudrait le problème, nous avons tout de même l'impression que cela n'est pas suffisant aujourd'hui *(du moins pour le #)*. Car il nous arrive souvent de mettre en place des #toreplace au sein d'attributs href, qui vont tout de même s'afficher en dur dans nos emails sur Office 365… Cela peut devenir particulièrement problématique lorsque vous essayez d'insérer des ancres par exemple au sein de votre email *(commençant donc par un #)*.

La solution est donc bien sûr de renseigner des urls valides au sein des attributs href de vos balises <a> *(et d'abandonner, du moins pour le moment, l'usage des ancres qui ne sont pas bien supportées sur l'ensemble des clients mail[182])*. Rémi Parmentier détaillait même que les styles appliqués à une balise <a> « problématique » étaient appliqués sur le premier élément avec un attribut style dans le code suivant[183]. Ainsi, sur Outlook.com, le code suivant :

```
<a style="color: red; background-color: yellow;"></a>
<p>A paragraph</p>
<p>Second paragraph with no style</p>
<p style="">Third paragraph with style</p>
<p style="">Fourth paragraph with style</p>
```

[182] *https://www.campaignmonitor.com/blog/email-marketing/2010/11/the-trouble-with-anchor-links-in-email-newsletters/*
[183] *https://github.com/hteumeuleu/email-bugs/issues/10*

Devenait :

```
<p>A paragraph</p>
<p>Second paragraph with no style</p>
<p style="color: red; background-color: yellow;">Third paragraph with
style</p>
<p style="">Fourth paragraph with style</p>
```

Et pour aller plus loin : **tout style d'une balise <a> invalide sera « poussé » vers une sorte de tableau de styles, tableau ensuite inséré sur le premier élément avec un attribut style**. Donc :

```
<a style="color: cyan; background-color: cyan;">cyan</a>
<a style="color: purple; background-color: purple;">purple</a>
<a style="color: black; background-color: black;">black</a>

<p style="">Empty style 1</p>
<p style="">Empty style 2</p>
<p style="">Empty style 3</p>
```

Deviendra :

```
<p style="color: cyan; background-color: cyan; color: purple; background-
color: purple; color: black; background-color: black;">Empty style 1</p>
<p style="">Empty style 2</p>
<p style="">Empty style 3</p>
```

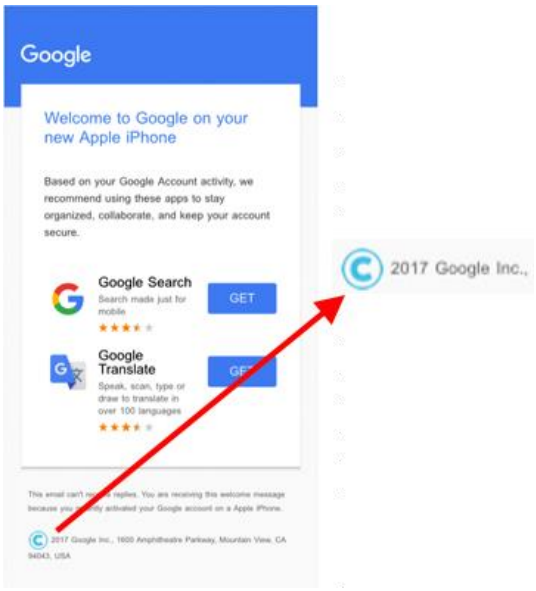
Efforts croissants des clients mails

INTRODUCTION

On perçoit désormais, et ce sans doute grâce à la communauté grandissante de l'email marketing, que les problèmes de rendu remontés sont généralement vite solutionnés, non seulement par des hacks & tips développés par des intégrateurs, mais aussi par le client mail lui-même *(et c'est important de le noter, car cela montre la volonté de certains clients mails, pas tous mettons-nous bien d'accord, d'évoluer)*.

GOOMOJI

Ainsi, le 27 novembre 2017, EmailonAcid remontait sur son blog le fait que les symboles ™*(trademark)*, ®*(registered)* ou ©*(copyright)* étaient remplacés, sur Gmail iOS, par des Emoji bleus[184] *(plus précisément des goomoji)* de taille supérieure à celle du texte substitué *(24 pixels sur 24 pixels)*.



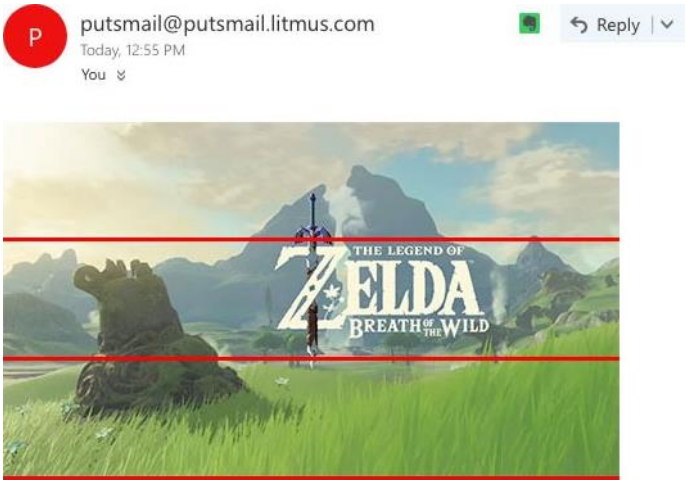
Captures d'écran – Source : email-designer.net

Plusieurs fixs ont été développés et testés, plus ou moins réussis.
Deux jours plus tard, le problème était résolu par Gmail.

[184] *https://www.emailonacid.com/blog/article/email-development/gmail-ios-replacing-copyright-symbols-with-blue-emoji-and-how-to-fix-it*

ESPACES SOUS LES IMAGES

Nous avons contribué chez Badsender, à notre manière, à la correction d'un bug sur Outlook.com. Il y a un peu plus d'un an, nous constations sur nos tests d'email preview **qu'un espace venait s'insérer sous nos images sur Office 365 et Outlook.com**.



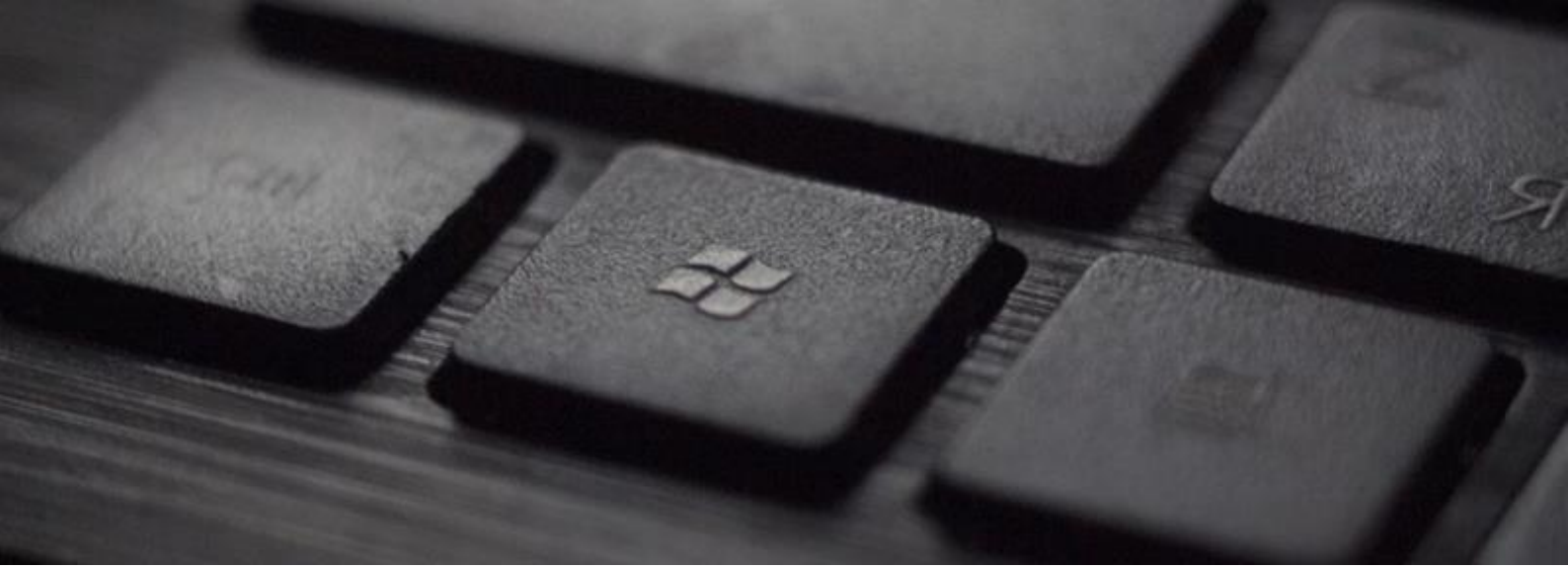
Captures d'écran – Source : hteumeuleu.com

Nous avions une solution : ajouter un style= « font-size :0px » sur les cellules contenant ces images. Mais nous n'avions pas pour autant compris le problème. Nous avons posté un message sur le forum de la communauté Litmus[185] pour comprendre l'origine du problème.

S'en suivirent alors un ensemble de discussions *(Outlook venait en fait transformer une simple balise <img> en <button><img>)*, un ensemble de hacks *(ajout d'id commençant par OWATemporaryImageDivContainer sur les images)*. Mais Rémi Parmentier s'empressa de clôturer le sujet[186] : **Le staff d'Outlook.com avait mis en place une mise à jour qui corrigeait ce problème**.

A nouveau, les équipes des clients mails concernés cherchaient à résoudre, par eux-mêmes, les problèmes soulevés, et contribuaient ainsi à l'amélioration de l'intégration HTML des emails.

[185] *https://litmus.com/community*
[186] *https://emails.hteumeuleu.com/outlook-coms-latest-bug-and-how-to-fix-gaps-under-images-ee1816671461*



Conclusion

Apprenez à connaître les clients et systèmes d'exploitation pour lesquels vous allez concevoir graphiquement et techniquement vos emails. C'est tout aussi important que de savoir concevoir *(graphiquement et techniquement)* des courriers électroniques. Cette tâche se fait pas à pas en analysant régulièrement, via les statistiques des campagnes, les clients mails sur lesquels vos campagnes sont consultées.

Les webmails et logiciels de messagerie BtoB ne sont clairement pas les mêmes que ceux du BtoC. Ce n'est pas le seul élément à prendre en compte bien sûr, il ne s'agit que d'un exemple, mais les contraintes techniques qui peuvent en découler sont bien différentes. Analysez la moyenne d'âge de votre cible, son type de support de consultation *(résolution, OS)*, son pays de résidence… Connaissez votre cible pour ajuster vos emails, vos méthodes de codage *(comme de design)* en conséquence.

Si vous n'avez aucun retour sur vos campagnes, ou que vous débutez votre travail sur l'emailing, inspirez-vous alors d'une vue d'ensemble de l'utilisation et des parts de marchés des clients de messagerie, comme celle proposée par Litmus[187]. Ces statistiques sont automatiquement mises à jour chaque mois. **Vous pourrez utiliser le top 10 des principaux clients comme base**.

[187] *http://emailclientmarketshare.com/*

Partie VII

Mantra de l’intégrateur d’emails

Qu'est-ce qui est le mieux pour l'utilisateur ?

La maquette d'un email utilise des polices tierces ?[188] Je peux coder cet email tout en images. Ou je peux intégrer cette police en CSS et laisser le texte en HTML, sachant cependant que ça ne fonctionnera pas partout.
Tous les clients mails ne supportent pas la balise <style> ? Je peux écrire tous mes styles en ligne. Ou je peux m'assurer que l'email se dégradera gracieusement et s'affichera aussi bien que possible sur les autres clients mails.
La question n'a pas de réponse absolue, car chaque projet est spécifique. **Ce qui fonctionnera pour un projet ne fonctionnera peut-être pas demain pour un autre**.

Tout dépend de la cible

Chaque projet est spécifique, en ce sens où la liste des destinataires est spécifique à chaque projet. Selon la cible de la campagne, et selon son comportement *(le support utilisé pour la consultation de l'email, ou le client mail sollicité)*, certaines techniques de codage, certaines méthodes pourront être privilégiées à d'autres *(si la consultation sur des webmails internationaux est absente, et l'ouverture depuis un GANGA très faible, alors il est peut-être intéressant de ne pas coder le CSS en ligne par exemple)*.

Pour résumer[189] :

UTILISATION SECURITAIRE	UTILISER AVEC PRECAUTION	NE PAS UTILISER
Mises en page statique avec tableaux Tableaux HTML et tableaux imbriqués Largeur du modèle <= 600px CSS simple, inline Polices standards	Images d'arrière-plan Polices web spécifiques/exotiques Le CSS dans la balise <style>	JavaScript <iframe> Flash Audio intégré Vidéo intégrée Formulaires

Utilisation sécuritaire : facilement pris en charge par la plupart des clients de messagerie.
Utiliser avec précaution : prise en charge limitée pour la plupart des clients mails.
Ne pas utiliser : n'est pas pris en charge par la plupart des clients de messagerie.

[188] *https://www.hteumeuleu.fr/le-mantra-de-l-integrateur/*
[189] *https://mailchimp.com/fr/help/limitations-of-html-email/*

Anticiper l'intégration HTML dès la phase de Design

Le travail sur le design de l'email doit être mené en connaissance des contraintes techniques à prévoir lors de la phase de codage. C'est pour cela qu'un Designer Print ne convient pas au design des emails. Ni un Designer Web d'ailleurs. Car un Designer Web pensera l'email comme une page *(s'il n'en connaît et n'en maîtrise pas les complexités)* et ne prendra donc pas en compte, par exemple : le fait de ne pas concevoir des emails de largeur supérieure à 600 pixels, de n'utiliser que très peu de typos spécifiques, d'utiliser avec précaution les images de fond, de ne pas mettre en place de vidéo, etc.

Partie VIII

Outils

Introduction

L'intégration HTML d'un emailing est faite de multiples étapes, nécessitant chacune des outils bien spécifiques : conception et développement du code HTML, auto-complétion, indentation, tests d'email preview / email rendering, livraison au client… Voyons donc en détail les outils et plateformes propres à ces différentes phases.

Conception

DREAMWEAVER

Dreamweaver[190] est **un éditeur de site web WYSIWYG** pour Microsoft Windows, et Mac OS X créé en 1997, commercialisé par Macromedia puis Adobe Systems. Dreamweaver fut l'un des premiers éditeurs HTML de type « tel affichage, tel résultat », mais également l'un des premiers à intégrer un gestionnaire de site. Ces innovations l'imposèrent rapidement comme l'un des principaux éditeurs de site web, aussi bien utilisable par le néophyte que par le professionnel.

Dreamweaver offre deux modes de conception par son menu affichage.
L'utilisateur peut choisir entre un mode création permettant d'effectuer la mise en page directement à l'aide d'outils simples, comparables à un logiciel de traitement de texte *(insertion de tableau, d'image, etc.)*. Il est également possible d'afficher et de modifier directement le code *(HTML ou autre)* qui compose la page. On peut passer très facilement d'un mode d'affichage à l'autre, ou opter pour un affichage mixte. Cette dernière option est particulièrement intéressante pour les débutants qui, à terme, souhaitent se familiariser avec le langage HTML.

BRACKETS

Brackets[191] est un éditeur open source pour le web design et le développement sur des technologies Web telles que HTML, CSS et JavaScript. Le projet a été créé et est maintenu par Adobe, et est publié sous une licence MIT.

NOTEPAD++ ET SUBLIME TEXT

Il est tout à fait possible de coder un fichier HTML à partir d'un éditeur de texte. Notepad++ pour Windows et Sublime Text pour Apple sont tous 2 des éditeurs de texte ayant des fonctionnalités aidant au développement HTML *(Couleurs des balises, etc.)*. Ils sont également gratuits.

[190] *https://fr.wikipedia.org/wiki/Adobe_Dreamweaver*
[191] *https://fr.wikipedia.org/wiki/Brackets*

Tests

INTRODUCTION

La partie “Tests” est sans nul doute une des étapes la plus importante de l’intégration HTML. Toute campagne HTML doit être testée avant envoi au client, en particulier pour s’assurer de son bon rendu. Cette étape ne devrait, en aucun cas, être négligée par l’intégrateur. Pour confirmer un rendu correct et validé par le client, plusieurs solutions sont envisageables :

SUPPORT PHYSIQUE

La méthode à privilégier reste, bien évidemment, le test en réel, sur supports physiques. Peu de structures ont les moyens de mettre cela en place. Le panel de supports à tester est large, et cela demande des investissements conséquents. Gardons cependant à l’esprit que n’importe quel support physique à disposition immédiate ne doit être négligé. **Au moins 1 test sur un appareil physique devrait être indispensable** : les plateformes d’email preview ne pourront sans doute jamais couvrir l’intégralité de la multiplicité des versions logiciels, des résolutions, des paramètres particuliers de connexion… Elles ne peuvent que difficilement vous donner une estimation du temps passé à télécharger et afficher les images, des coupures ou bugs qu’engendreront parfois le poids ou la hauteur de l’email testé…

SOLUTIONS D’EMAIL PREVIEW

Introduction

« Email rendering » ou « email preview » sont en corrélation : Ils désignent « la prévisualisation du rendu d’un email ». L’email preview rend possible le test de l’email rendering d’une campagne. L’un ne va pas sans l’autre. L’email preview ne fait que précéder et confirmer *(ou corriger si besoin)* l’email rendering. Nous souhaitons, en tant que designer, chef de projet/campagne, intégrateur, nous assurer du bon rendu de l’intégration HTML, et la prévisualisation d’email est là pour ça.

Comme le définit donc le site Définitions marketing.com[192], *« l’email rendering désigne la problématique de rendu visuel des messages email au sein des différents outils et services de messagerie coté destinataires d’une campagne. Une des caractéristiques de l’email marketing est le fait qu’un même message puisse donner des rendus visuels différents d’un environnement de lecture à l’autre en fonction de l’outil utilisé et de l’email rendering engine, mais également en fonction des différentes configurations d’une même solution. Les tests d’email rendering précédant une campagne permettent de voir comment le code est interprété par les différents environnements, de voir ce que le message donne selon que l’option d’affichage des images est activée ou non et de voir éventuellement les aperçus en fenêtre de prévisualisation. »*

[192] *https://www.definitions-marketing.com/definition/email-rendering/*

Les plateformes d'email preview sont devenues des alliés non négligeables *(voir indispensables)* pour s'assurer un rendu cohérent et correct sur la majorité des clients mail et supports[193]. Certains de ces outils sont plus ou moins « avancés », et développent et proposent des fonctionnalités intéressantes, optimisant sans relâche les campagnes de leurs clients : optimisation de code, inspiration graphique, reporting, test spam et délivrabilité, optimisation du temps de chargement et du poids des images, rendering instantané, images « mozifiées »… Tout est une question de …moyens ! **La bataille se joue aussi sur le nombre de rendus délivrés**. Passons en revue les plateformes d'email preview les plus connues et majeures du marché :

Litmus

LA plateforme d'email preview par excellence. Litmus[194] se démarque par l'ensemble des fonctionnalités qu'elle propose : Builder et tests en temps réel, tests des liens, email analytics, spam testing, ressources, communauté et forums, innovations… Sans aucun doute, Litmus a dépassé le seuil de la simple plateforme d'email preview pour devenir un acteur incontournable de l'email marketing. Véritable référence, la marque organise désormais des « Litmus Live » pour partager sur l'email marketing : Design, ergonomie, code, nouveautés, optimisation de campagnes… Tout *(ou presque)* y est abordé.

EmailOnAcid[195]

Hébergement

Lors de la phase d'email preview, certaines plateformes proposeront un hébergement automatique des images rattachées à l'intégration HTML. D'autres en revanche nécessiteront que les images soient déjà hébergées sur un serveur et que les chemins de ces images au sein de l'intégration HTML soient bien absolus.

Le logiciel Filezilla[196], logiciel FTP *(pour File Transfert Protocol)* reste sans doute le plus simple d'utilisation pour pouvoir héberger correctement les images. Ce logiciel est disponible pour Windows, Mac OS X, et GNU/Linux.

Livraison

Couramment, la méthode de livraison de l'intégration HTML réside dans l'envoi d'un fichier ZIP contenant le fichier HTML, ainsi que le dossier d'images rattaché. Pour cela, la solution la plus efficace reste le « Dossier compressé » proposé par Windows, ou encore la possibilité de compresser directement ces éléments via les outils Winrar[197] ou Winzip[198], utilitaires de compression de données Windows, qui se focalisent sur les formats de compression de données RAR et ZIP.

---

[193] *https://litmus.com/blog/why-is-email-rendering-so-complex*
[194] *https://litmus.com/*
[195] *https://www.emailonacid.com/*
[196] *https://filezilla-project.org/*
[197] *https://www.win-rar.com/start.html?L=10*
[198] *https://www.winzip.com/win/fr/downwz.html*

Partie IX

Conclusion

En HTML, l'intégration emailing est un art délicat, en partie dû au nombre de clients mail sur le marché qui ont des comportements bien différents. Cette disparité de l'interprétation des emails au format HTML provient de plusieurs facteurs :

- Les clients e-mails lourds *(Outlook, Thunderbird, Lotus Notes...)* ont un fonctionnement radicalement différent des webmails *(Gmail, Yahoo Mail, Hotmail, Outlook.com...)*.

- Ces mêmes webmails imposent des règles très strictes sur le code HTML et CSS contenu dans les emails car les messages sont eux-mêmes affichés dans une structure de page web qui a ses propres styles.

- Certains moteurs sont très exotiques, par exemple Outlook 2013 utilise toujours le moteur d'interprétation HTML de Word, ce qui signifie virtuellement que la prise en charge de CSS dans Outlook n'a pas changé entre les versions 2007, 2010 et 2013, là où les navigateurs ont fait des bonds de géants.

Encore plus que le web, la réalisation d'emails graphiques en HTML au pixel près ou presque sur toutes les plateformes est une utopie[199]. D'ailleurs, **l'intérêt n'est plus vraiment dans cet objectif, mais dans les versions mobiles de ces emails, et leur accessibilité**.

[199] *https://emails.hteumeuleu.fr/2016/05/est-ce-qu-un-e-mail-doit-avoir-le-meme-rendu-partout/*

Partie X

Ressources bibliographiques

De nombreux textes de ce livre blanc sont issus des multiples blogs/sites Internet traitant du sujet de l'email marketing. Lorsque les sources sont de qualité et vérifiées et que les textes sont parfaitement bien rédigés, pourquoi chercher à les retranscrire autrement ? *« Rendez à César ce qui appartient à César, et à Dieu ce qui appartient à Dieu. »* Nous ne les remercierons jamais assez pour leur contribution et espérons ne commettre aucun oubli en répertoriant ci-dessous les contributeurs concernés.

Badsender

http://www.badsender.com/2009/08/03/can-spam-act-spam-loi-etats-unis-optin-optout/
http://test.badsender.com/krktr/
http://www.badsender.com/2017/11/09/email-css-position-absolute/
http://www.badsender.com/
http://agence.badsender.com

Campaign Monitor

https://www.campaignmonitor.com/blog/email-marketing/2010/11/the-trouble-with-anchor-links-in-email-newsletters/
https://www.campaignmonitor.com/resources/guides/web-fonts-in-email/
https://www.campaignmonitor.com/blog/email-marketing/2013/01/outlook-com-drops-margin-and-float-support-entirely/
https://buttons.cm/
https://www.campaignmonitor.com/blog/email-marketing/2012/08/outlook-2013-says-no-to-empty-table-cells/
https://backgrounds.cm/
https://www.campaignmonitor.com/resources/guides/video-in-email/
https://www.campaignmonitor.com/blog/email-marketing/2006/01/the-truth-about-flash/
https://www.campaignmonitor.com/blog/email-marketing/2016/07/10-things-need-know-web-fonts-email-right-now/
https://www.campaignmonitor.com/css/media-queries/media/
https://www.campaignmonitor.com/blog/email-marketing/2011/04/media-query-issues-in-yahoo-mail-mobile-email/
https://www.campaignmonitor.com/dev-resources/guides/mobile/
https://www.campaignmonitor.com/resources/guides/alt-text-in-email/
https://www.campaignmonitor.com/blog/email-marketing/2010/11/correct-doctype-to-use-in-html-email/
https://www.campaignmonitor.com/css/
Campaign Monitor, «Email Interaction across mobile and desktop»

Chamaileon

https://blog.chamaileon.io/limitations-of-html-email-design-email-width-and-size/
https://blog.chamaileon.io/what-do-scalable-fluid-responsive-and-hybrid-email-design-mean/

EmailonAcid

https://www.emailonacid.com/
https://www.emailonacid.com/blog/article/email-development/gmail-ios-replacing-copyright-symbols-with-blue-emoji-and-how-to-fix-it
https://www.emailonacid.com/blog/article/email-development/mobile-optimization-retina-images-in-email/
https://www.emailonacid.com/blog/article/email-development/demystifying-meta-tags-in-email/
https://www.emailonacid.com/blog/article/email-development/doctype_-_the_black_sheep_of_html_email_design/

Email Designer
http://email-designer.net/supprimer-modifier-couleur-liens-webmail-zimbra/

EmailMonday
https://www.emailmonday.com/google-amp-for-email-everything-you-wanted-to-know/

Google
https://blog.google/products/gmail/gmailify-best-of-gmail-without-gmail/
https://developers.google.com/gmail/design/reference/supported_css
https://www.blog.google/products/g-suite/bringing-power-amp-gmail/

Hteumeuleu
https://emails.hteumeuleu.fr/2016/05/est-ce-qu-un-e-mail-doit-avoir-le-meme-rendu-partout/
https://emails.hteumeuleu.com/outlook-coms-latest-bug-and-how-to-fix-gaps-under-images-ee1816671461
https://emails.hteumeuleu.fr/2016/10/quel-doctype-faut-il-utiliser-dans-un-email/
https://github.com/hteumeuleu/email-bugs/issues/10
https://emails.hteumeuleu.fr/2016/01/gmail-app-on-android-tries-to-shrink-your-email-with-munged-classes/
https://emails.hteumeuleu.com/trying-to-make-sense-of-gmail-css-support-after-the-2016-update-53c15151063a
https://emails.hteumeuleu.fr/2014/08/outlook-com-supporte-bien-css-float-margin/
https://emails.hteumeuleu.fr/2017/02/faut-il-arreter-d-ecrire-ses-styles-en-ligne-dans-des-e-mails/
https://emails.hteumeuleu.com/how-webmails-block-images-4cc2ba4d2d0e
https://www.hteumeuleu.fr/le-mantra-de-l-integrateur/

Litmus
https://litmus.com/
https://litmus.com/blog/the-pros-and-cons-of-video-in-email-video
https://litmus.com/blog/why-is-email-rendering-so-complex
https://litmus.com/community
http://emailclientmarketshare.com/
https://litmus.com/community/discussions/4594-outlook-365-displays-tbd-on-empty-href-tags
https://litmus.com/community/discussions/1475-office-365-links
https://litmus.com/community/discussions/6509-html-semantic-elements-in-email
https://litmus.com/community/discussions/1524-do-you-care-about-semantics-in-html-emails
https://litmus.com/community/discussions/4624-fixed-gmail-app-not-respecting-100-width
https://litmus.com/community/discussions/4410-gmail-app-stacked-column-width-woes-no-media-queries
https://litmus.com/blog/the-tyranny-of-tables-why-web-and-email-design-are-so-different
https://litmus.com/blog/the-ultimate-guide-to-styled-alt-text-in-email
https://litmus.com/community/discussions/36-outlook-and-fallback-fonts
https://litmus.com/blog/the-ultimate-guide-to-web-fonts
https://litmus.com/blog/a-guide-to-bulletproof-buttons-in-email-design
https://litmus.com/community/discussions/960-outlook-2013-breaks-td-height
https://litmus.com/blog/ultimate-guide-accessible-emails
https://litmus.com/community/discussions/4931-using-retina-images-as-background-images
https://litmus.com/blog/understanding-retina-images-in-html-email
https://litmus.com/blog/the-ultimate-guide-to-web-fonts
https://litmus.com/blog/the-ultimate-guide-to-background-images-in-email
https://litmus.com/blog/understanding-responsive-and-hybrid-email-design
https://litmus.com/help/email-clients/media-query-support/

https://litmus.com/blog/understanding-media-queries-in-html-email
https://litmus.com/blog/top-email-design-trends
https://litmus.com/community/discussions/6116-here-s-why-litmus-didn-t-inline-css-for-its-first-newsletter-of-2017
https://litmus.com/community/discussions/261-td-display-block-not-working-on-android
https://litmus.com/blog/background-colors-html-email
Litmus, «State of Email», Mars 2017

Mailchimp

https://mailchimp.com/fr/help/limitations-of-html-email/
https://templates.mailchimp.com/development/css/outlook-conditional-css/
https://templates.mailchimp.com/concepts/email-clients/

Mosaico

https://mosaico.io/email-client-tricks/gmail-android-tries-to-shrink-your-email/
https://mosaico.io/email-client-tricks/email-inliners/

Slideshare

https://fr.slideshare.net/M_J_Robbins/accessibility-in-email-eoainsights
https://fr.slideshare.net/paulairy/a-type-of-accessibility-65004974?next_slideshow=1

W3Schools

https://www.w3schools.com/tags/
https://www.w3schools.com/cssref/pr_pos_vertical-align.asp
https://www.w3schools.com/css/css_float.asp
https://www.w3schools.com/cssref/pr_class_display.asp
https://www.w3schools.com/css/css_positioning.asp
https://www.w3schools.com/cssref/pr_margin.asp
https://www.w3schools.com/tags/tag_html.asp
https://www.w3schools.com/cssref/pr_dim_line-height.asp
https://www.w3schools.com/tags/tag_doctype.asp
https://www.w3.org/QA/2002/04/valid-dtd-list.html
https://www.w3schools.com/cssref/pr_text_text-align.asp
https://www.w3schools.com/cssref/pr_font_weight.asp
https://www.w3schools.com/cssref/pr_padding.asp
https://www.w3schools.com/cssref/
https://www.w3schools.com/tags/tag_head.asp
https://www.w3schools.com/tags/tag_meta.asp
https://www.w3schools.com/cssref/pr_font_font-size.asp
https://www.w3schools.com/cssref/pr_font_font-family.asp
https://www.w3schools.com/html/html_css.asp
https://www.w3schools.com/tags/att_global_dir.asp
https://www.w3schools.com/tags/att_global_style.asp
https://www.w3schools.com/tags/att_global_id.asp
https://www.w3schools.com/tags/att_global_class.asp
https://www.w3schools.com/tags/att_a_target.asp
https://www.w3schools.com/tags/att_td_rowspan.asp
https://www.w3schools.com/tags/att_td_colspan.asp
https://www.w3schools.com/tags/att_td_valign.asp
https://www.w3schools.com/tags/att_table_cellpadding.asp
https://www.w3schools.com/tags/att_table_cellspacing.asp
https://www.w3schools.com/tags/att_img_border.asp

https://www.w3schools.com/tags/att_global_title.asp
https://www.w3schools.com/tags/att_width.asp
https://www.w3schools.com/tags/att_height.asp
https://www.w3schools.com/tags/att_img_src.asp
https://www.w3schools.com/tags/att_a_href.asp
https://www.w3schools.com/tags/att_img_alt.asp
https://www.w3schools.com/tags/tag_p.asp
https://www.w3schools.com/tags/tag_hn.asp
https://www.w3schools.com/tags/tag_comment.asp
https://www.w3schools.com/tags/tag_a.asp
https://www.w3schools.com/tags/tag_span.asp
https://www.w3schools.com/tags/tag_br.asp
https://www.w3schools.com/tags/tag_div.asp
https://www.w3schools.com/tags/tag_img.asp
https://www.w3schools.com/tags/tag_tr.asp
https://www.w3schools.com/tags/tag_td.asp
https://www.w3schools.com/tags/tag_th.asp
https://www.w3schools.com/tags/tag_body.asp
https://www.w3schools.com/tags/tag_table.asp
https://www.w3schools.com/tags/tag_style.asp
https://www.w3schools.com/html/html_blocks.asp
https://www.w3schools.com/html/html_basic.asp
https://www.w3schools.com/tags/ref_byfunc.asp
https://www.w3schools.com/html/html_elements.asp

Wikipedia

https://fr.wikipedia.org/wiki/Adobe_Dreamweaver
https://fr.wikipedia.org/wiki/Brackets
https://fr.wikipedia.org/wiki/WebKit
https://fr.wikipedia.org/wiki/Vector_Markup_Language
https://fr.wikipedia.org/wiki/Syst%C3%A8me_d%27exploitation
https://fr.wikipedia.org/wiki/Site_web_adaptatif
https://fr.wikipedia.org/wiki/Indentation
https://fr.wikipedia.org/wiki/Feuilles_de_style_en_cascade
https://fr.wikipedia.org/wiki/World_Wide_Web_Consortium

Autres

https://www.win-rar.com/start.html?L=10
https://www.winzip.com/win/fr/downwz.html
https://blog.gorebel.com/absolute-positioning-in-email/
https://www.lafabriquedunet.fr/email-marketing/guide/coder-email-html-css/
http://xhtml.le-developpeur-web.com/balise_autofermante-xhtml.php
https://www.whoishostingthis.com/blog/2014/12/06/jpeg-gif-png/
https://css-tricks.com/snippets/css/comments-in-css/
https://www.w3.org/TR/html401/present/graphics.html#h-15.1.2
https://filezilla-project.org/
https://www.definitions-marketing.com/definition/integration-email-html/
https://webdesign.tutsplus.com/tutorials/creating-a-future-proof-responsive-email-without-media-queries--cms-23919
https://www.definitions-marketing.com/definition/email-rendering/
https://github.com/FunWithEmail/HTML-Email-Hacks
https://helpx.adobe.com/dreamweaver/using/reuse-code-with-snippets.html

https://emmet.io/
http://removebluelinks.com/
http://www.comprendre-internet.com/Qu-est-ce-que-le-HTML.html
http://www.pompage.net/traduction/Bien-utiliser-le-texte-alternatif
http://blog.emailwizardry.co/2012/12/how-to-fix-superscript-characters/
https://github.com/bago/android-gmail-emulator
http://xahlee.info/js/html5_non-closing_tag.html
https://www.pierre-giraud.com/html-css/cours-complet/block-inline-html-css.php
https://audreytips.com/gmail-emailing-tronque/
http://www.courtneyfantinato.com/correcting-outlook-dpi-scaling-issues/
http://sebsauvage.net/comprendre/dpi/index.html
https://www.lesintegristes.net/2011/05/06/web-resolution-72dpi/
http://freshinbox.com/blog/faux-video-video-in-email-using-sprites/
https://codepen.io/kristianrobinson/pen/EwERve
http://www.emailmarketingtipps.de/2018/02/19/amp-for-email-controversies/
https://www.lafabriquedunet.fr/email-marketing/articles/google-amp-emails-interactifs/
https://blog.internet-formation.fr/2018/02/google-introduit-amp-for-email-pour-gmail/
https://www.presse-citron.net/amp-google-veut-rendre-e-mails-plus-interactifs/
http://blog.agence-bmobile.com/2018/02/amp-email-ce-que-le-gmail-interactif-va-peut-etre-changer-pour-vous/
https://www.youtube.com/watch?v=sj2KANF9o3k
https://ignitevisibility.com/amp-for-email/
https://www.definitions-marketing.com/definition/flash/
http://email-a-table.fr/le-javascript-dans-les-emails,495
http://www.ampsoft.net/webdesign-l/WindowsMacFonts.html
https://www.anthony-brard.com/les-fonts-web-safe
https://websitesetup.org/web-safe-fonts-html-css/
https://www.cssfontstack.com/
https://www.alsacreations.com/article/lire/930-css3-media-queries.html
https://compressjpeg.com/fr/
https://www.jpegmini.com/
https://tinyjpg.com/
https://tinypng.com/
https://ezgif.com/
https://www.emailmonday.com/mobile-email-usage-statistics/
http://formation.upyupy.fr/html-xhtml/caracteres-speciaux/
https://www.jchr.be/html/caracteres.htm
https://htmlcolorcodes.com/fr/noms-de-couleur/
https://unsplash.com
The Radicati Group, «Email Statistics Reports, 2014-2018»
Email in Motion: How Mobile is Leading the Email Revolution, Return Path, 2011

Pour nous contacter

Email : yesreply@badsender.com
Téléphone en France : **01 76 38 00 26**
Téléphone en Belgique : **02 808 18 87**

Badsender France : 3 Place Bellevue – 11170 Carlipa – France
Badsender HQ : Square de Meeûs, 35 – 3e étage – 1000 Bruxelles – Belgique