



Email Design & Coding Recommendations

General guidelines for designing HTML emails and coding HTML, mobile and text emails

In the ever-shifting landscape of digital marketing, the needs of consumers and the demands of the email channel evolve frequently—and so must design and coding best practices.

This guide contains our most up-to-date recommendations to help ensure your success in the design and coding of emails for all screens. Follow these steps to increase the chances that your emails will achieve high deliverability rates, render properly and drive results.

Contents

Standard email elements and basic HTML email design recommendations	1
Preheader text	1
Header & navigation bar	1
Email body	1
Recovery module	2
Footer	2
Social network sharing links	3
Social network community links	3
A note on preview pane optimization	3
Mobile-influenced design best practices	4
Mobile design recommendations	5
Getting started with mobile design	5
Knowing your subscribers' opening behavior	
Knowing your subscribers' devices	
Choosing your mobile email design strategy	
Mobile-optimized email	6
Ways to optimize for mobile	
Responsive email	6
Examples of how responsive email design can be utilized	
Mobile-optimized landing pages	8
A note on whether to include plain-text email	9
Coding recommendations	10
Significant considerations for coding HTML email	10
Establishing an effective process for email coding	10
Before coding	
Reviewing assets after approval	
Creating slices in Photoshop and exporting graphics	
Coding	

Render testing	
A deeper look at coding particulars and top mistakes to avoid	12
Unsupported HTML features	
Scripts, flash and embedded video	
Cascading style sheets (CSS)	
Unnecessary tags and code	
Basic layout: Working with tables	13
Important considerations	
Using outer tables to encapsulate content	
Spacing: Padding, margin and spacer.gifs	
Working with images	17
General notes on image formatting	
Background images	
Graphics: ALT attributes	
Issues in Gmail and Hotmail	
Graphics within <TD> Tags	
Image maps & embedded links within images	
Working with text & links	19
General tips	
Using web-safe fonts	
Using to achieve text breaks	
Using hex values for color	
Special characters	
Links	
Responsive email	21
External style sheet	
Code structure	
Testing	
Platform-specific fixes	23
Coding for Outlook 2007, 2010 and 2013	24
Major issues to be aware of	
Background images	
Animated GIFs	
Vertical expansion of <TD> tags	
Limited support for CSS	
Poor and sometimes unpredictable rendering behavior	
Outlook.com's larger default line-height	
Testing: The final step before launching	27
Code validation	
Render testing	

Breaking the rules

28

Standard email elements and basic HTML email design recommendations

This section describes the anatomy of the basic marketing email and provides general design optimization tips. Best practices for email design leave room for diverse approaches, but the following guidelines serve as an effective starting point.

Preheader text

We recommend including preheader text at the top of every email. The preheader is composed of one line of HTML text at the top of the email, above the header/logo (in some situations, it may require two lines of copy, but shorter is typically better). Keep in mind that this is the first text that will appear in a subscriber’s preview pane. Effective preheaders typically do the following:

- Summarize the email’s primary call-to-action and/or build on the subject line.
- Include a link to the primary landing page.
- Include a “view with images” link so subscribers can view a web-hosted version of the email.

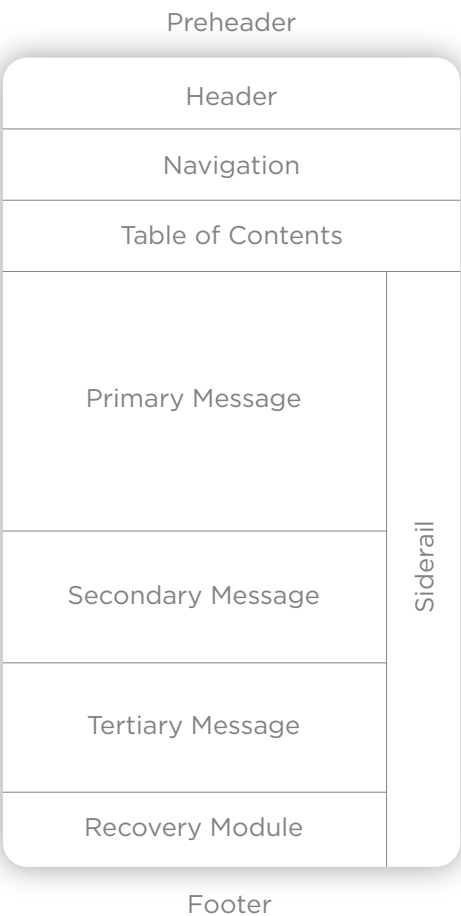
Header & navigation bar

Place your logo on the left side of your header. It’s important for branding and for recognition of your emails.

Use HTML text for your navigation bar links. Consider periodically changing the links included in your navigation bar to feature topical items corresponding with the target of a particular campaign (for example, the best links to include with a sale message may differ from the best links to include with an account management message). It may make sense for your brand to place a promotional link in the navigation bar, such as a graphical element “breaking” the header. This serves as an extra place to capture clicks from subscribers who may not scroll to read the full email.

Email body

The “email body” conveys your primary and secondary message(s). Most subscribers will spend only a few



seconds on your email, and the below moves can help you get the most from their brief moments of attention:

- Establish clear content groupings and a messaging hierarchy by leveraging color, size and other elements.
- Make your copy easy to skim by keeping it brief (four lines is a good rule-of-thumb maximum), and consider using bullets to convey services or product features quickly.
- Encourage scrolling with unique layout patterns and playful peeks.
- Use button treatment for primary calls-to-action (CTAs).
- Ensure text links use a web-safe font, stand out and look clickable.
- Leave space for expanding fonts; some email readers and platforms display fonts larger than others.
- Focus on readability when selecting text and background colors and when determining text size. Keep your body copy 13 point or larger and legal text 11 point or larger (like the text in the footer) so that you don't lose readers who might struggle to read small print.
- Use HTML text wherever possible.
- Design for a usable experience whether a subscriber has images on or off, since most inboxes don't load images by default.

Recovery module

Recovery modules are optional content blocks that appear toward the bottom of emails. They generally contain several links to product categories, clearance items, articles and other resources or points of interest for subscribers. Recovery modules sometimes capture clicks from subscribers who haven't found anything of interest in the body of the email or who are looking for specific information.

Footer

The footer is the HTML text that appears at the bottom of the email (also referred to as "legal text").

- Include disclaimers for promotions included in the email, if applicable.
- Include an unsubscribe link. Use a subhead, bold text or different colored text to make this link easy to find, preventing spam complaints. If you have included an unsubscribe link in your preheader or header, we recommend also including one in the footer because subscribers have been trained to seek it there.
- Include other administrative links such as those to update preferences, change email address and subscribe (for those who receive the email as a forward). These links should appear before the unsubscribe link to prevent those who would like to change their email address from unsubscribing and

then re-subscribing.

- Consider placing an “add to address book” link in the footer.
- Include your mailing address (street address or P.O. box).
- Include copyright information.

Social network sharing links

Share links allow subscribers to share your email—or portions of it—with their friends and family members on social networks such as Facebook, Twitter, Pinterest and Google+.

- Include links allowing subscribers to share your entire email as a permanent fixture in your header.
- Include links to share specific elements of your email when appropriate (for example, you might include “Pin It” functionality on a particularly engaging photo to encourage sharing via Pinterest).

Social network community links

We recommend including links toward the bottom of your email inviting subscribers to connect with your brand via social sites on which you have a presence.

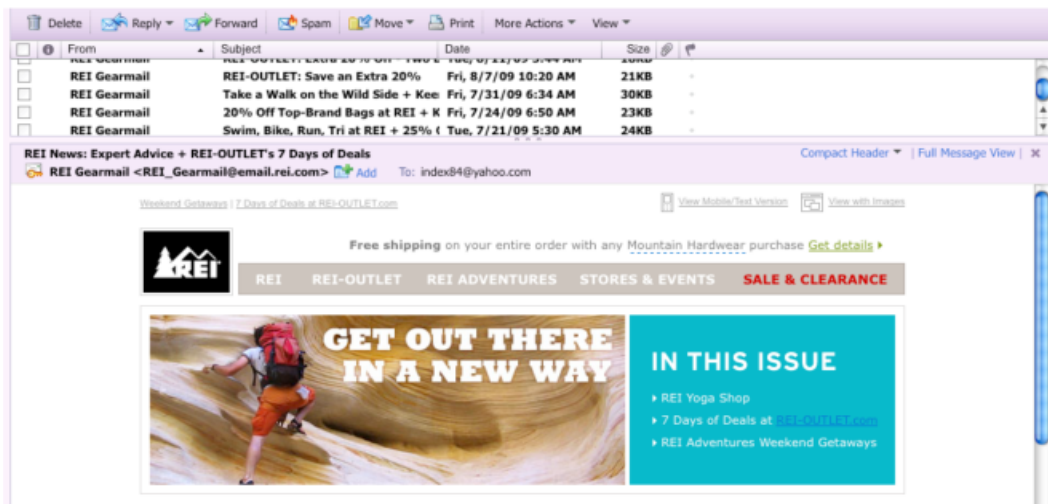


Consider grouping community links with mobile app, SMS subscription, blog and RSS subscription links. These links work well together because they all present ways for you and your customers to stay in touch.

A note on preview pane optimization

Most subscribers use preview panes to read their emails, limiting the visible area of the email. Although preview pane sizes vary, thinking of the visible area as a square with 320-pixel sides is a safe approach. The messaging in this space deserves a high level of consideration; often subscribers decide whether to engage more deeply with an email depending on whether the preview pane interests them.

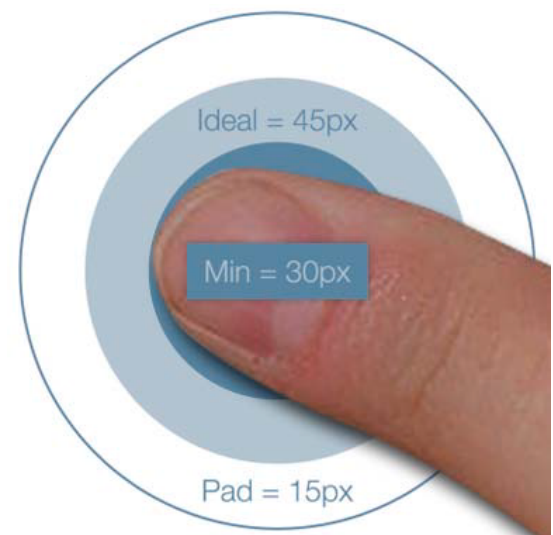
- Be mindful of the height of your preheader and header navigation bar since these elements limit the amount of the email body that is visible in the preview pane.
- Include your headline and primary call-to-action in the preview pane.



Mobile-influenced design best practices

Whether or not your company has made a conscious choice to target a mobile audience, we now live in a multi-screen world: consumers read email on all kinds of devices, so email needs to offer a positive experience on smartphones, tablets and desktop screens. Because of this reality, the following best practices have emerged for all email design.

- Avoid using long subject lines, which will push email content even farther down an already-small mobile screen.
- Ensure that your most valuable content is within the top left 320-pixel area of your email. Not only is this helpful for preview pane optimization as described above, but some mobile devices don't scale an email to fit the width of the screen. Because of this, some viewers will only see the top left corner of the email at 100% of its normal size.
- Make emails easy to scroll, encouraging rapid scanning while still conveying key information. Check that text is legible, even when scaled down on a smaller screen.
- Keep emails brief, allowing the landing page to tell a more detailed story.
- Ensure that CTAs are easy to click by using large buttons and links, adding padding between sections, and designing entire sections to be clickable. When pressed to a touch screen, the human finger requires more space to click accurately than a mouse does. Make sure that your CTAs are padded at least 10–15 pixels to avoid frustrating tap errors.
- Keep email file sizes small for quick load times at any connection speed.



Mobile design recommendations

With mobile email readers composing a significant (and increasing) percentage of many brands' subscribers, it's essential that design teams have the expertise to deliver seamless mobile experiences. Reports continue to demonstrate that mobile subscribers become less engaged when forced through an experience that isn't designed optimally for their device. Making choices about email design for mobile requires you to develop an understanding of your subscriber behavior and of the resources available for your brand to invest in mobile design.

Getting started with mobile design

Knowing your subscribers' opening behavior

Get a breakdown of your subscribers' behavior, including the percentage of your subscriber base reading on mobile devices, and which type of mobile devices and operating systems they're using (you can get this information from Return Path or Litmus). You may be surprised to find more opens coming from tablets than phones, or from iOS than Android. Getting a solid grasp of your subscribers' habits will allow you to invest in mobile optimization in a way that reaches the largest possible audience.

Knowing your subscribers' devices

While Android devices make up a larger share of the market, it currently looks as though iPhone and other Apple iOS devices contribute to a larger share of mobile opens. That said, it's important to keep in mind that open metrics may be skewed by the fact that many devices block images by default, while iOS does not.

Google's Android operating system is used on a variety of devices from companies including Samsung, HTC, LG and Motorola. Because Android is open source, manufacturers often customize it, which sometimes modifies the email app in ways that lead to inconsistent rendering across devices. Windows Phone and BlackBerry also present their own unique challenges.

It's likely that the level of support for email across apps and operating systems will continue to change; this is an area that calls for ongoing testing and readjusting.

Choosing your mobile email design strategy

There are two primary approaches to designing in a way that prioritizes your mobile audiences: mobile-optimized email and responsive email. Once you understand your subscribers' behavior and assess the time, energy and financial resources your brand can commit to mobile design, you can select the most appropriate approach, described below.

Mobile-optimized email

Mobile-optimized email is HTML email designed specifically for small mobile devices that render designs at 320–480 pixels. This type of email still renders on desktop screens, but it's designed to fit small screens best. When most of your subscribers open email from their smartphones and small tablets, it's time to put those customers first by delivering a mobile-targeted experience.

The drawback of mobile-optimized email is that it limits the experience you can provide to your desktop readers: email designed to render on a small screen looks small when opened from a desktop and doesn't utilize all of the available desktop screen real estate. Further, the narrowness of the design limits design layout options.

Even so, mobile-optimized email delivers a positive experience for most subscribers on mobile and desktop screens. It's a solid strategy for mobilizing your email program without significantly increasing the time or money invested in the program.

Ways to optimize for mobile

— Keep mobile-optimized emails simple.

Use clear messaging and imagery that help convey the primary message. Strip away non-essential elements. Mobile subscribers are often on-the-go or engaged with other activities (watching television, chatting with friends, etc.). Straightforward, uncluttered messaging is the most effective way to get these subscribers to act.

— Lighten your email file sizes.

This will increase the chance that they will render effectively on mobile devices. Some mobile email clients will require an additional button-press to download the rest of the email when file weight is too big. The optimal email weight for mobile is 20K or less, and while that weight may not be realistic for some marketers, the more you can slim down the file weight the better. Mobile Internet speeds, while sometimes quite fast, are inconsistent and often dependent on a subscriber's location. It's also worth noting that subscribers pay per megabyte when downloading content, so sending lighter files shows respect for your customers.

Responsive email

An email coded responsively and designed upon a flexible framework will automatically adapt its layout to better fit the customer's current screen size, no click required.

Responsive email is composed of one HTML file that uses CSS media queries to listen for a device's screen width. Copy and imagery can be wrapped, hidden, resized or optimized when the layout reaches a specified width. In order for this to happen, the mobile email app must recognize the responsive media queries, which the native email apps on iOS, Android and some newer BlackBerry devices currently do.

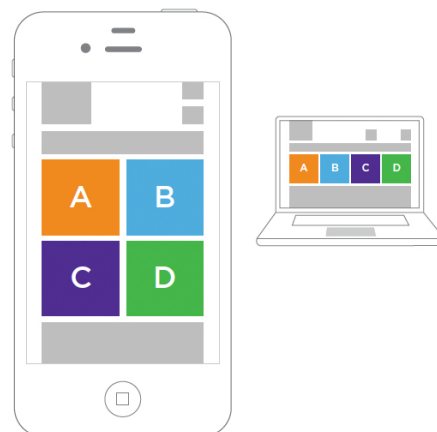
The majority of Windows Phones do not support responsive media queries, although future updates may include support. When a responsive email is opened on a device that does not recognize the media queries, the email simply opens as any other basic HTML email would on a mobile screen.

Responsive email is a bit more difficult to design and code than mobile-optimized email, requiring a higher initial investment. However, it results in a better user experience than other approaches and has been shown to facilitate higher click-through conversion when aligned with the right audience.

Examples of how responsive email design can be utilized

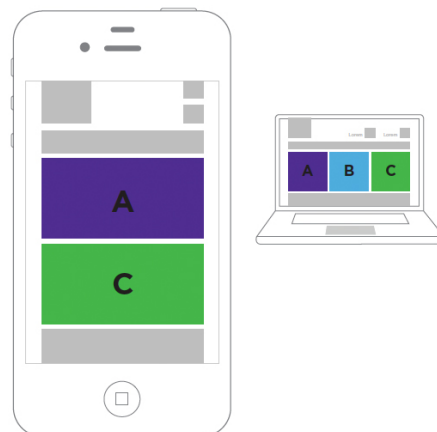
— Wrap elements.

Wrapping elements lets your email design reflow to fit within a mobile screen. When designing elements to wrap, think in terms of the grid structure for both the mobile and desktop versions: for example, the Apple iPhone has a screen width of 320 pixels, so if the desktop version of an email is designed to be 640 pixels wide and divisible into two columns, elements can wrap without the need to scale.



— Hide elements.

Sometimes, images take too long to load on mobile devices or make your email too long for mobile customers. Responsive design allows you to hide images and other elements, streamlining the experience for the mobile reader. The responsive email can hide an entire text block or text within a block.



— Adjust elements.

Many elements can change across versions to deliver an optimal mobile experience. Elements that can change include font size, color, family, weight,

decoration, style, variant, line-height, margin, padding and many more.

— Optimize images.

In addition to hiding them, images can be manipulated to look great on any device. If an email's hero image is too wide for a small screen, it's easy to swap the image out for one that is specifically cropped to fit a mobile screen. The email's images can also be resized, hidden or cropped by hiding slices. Further, if an image looks blurry, you can "sharpen" it by saving the image at double the size and shrinking it to fit the layout.

Below is an at-a-glance picture of what can be done within responsive design:

Change style <ul style="list-style-type: none">- Font size- Font color- Font family- Font weight- Font decoration- Font style- Font variant- Line height- Margin/padding	Change position <ul style="list-style-type: none">- Move text- Move image- Move container Change visibility <ul style="list-style-type: none">- Hide text- Hide image slice- Hide bg image- Hide container	Change content <ul style="list-style-type: none">- Swap url- Swap text element- Swap bg image Change size <ul style="list-style-type: none">- "Chop" image- Scale image- Scale container- Scale html element
---	--	--

Mobile-optimized landing pages

Without a web experience designed for their needs, customers might disengage before converting. After you've invested time and thought delivering email that renders effectively on mobile devices, complete the experience with a mobile-optimized landing.

- Make it easy for users to find what they originally clicked on in the email.
- Utilize alternatives to Flash, such as HTML5, CSS3 and JavaScript.
- Be clever with the use of space by embracing accordions, carousels, modals, drop downs, tabs, etc.
- Make the checkout process as quick and easy as possible. Minimize the number of form fields and investigate simple solutions.
- Optimize pages for speed and reduce files size—fast load times are critical to keeping customers from disengaging.
- Optimize your landing page widths to be more mobile-friendly.
- Keep landing page copy brief. Save the more expansive details for another page deeper on your site.
- Remember that for touch screen users, there is no hover-state for links; if they touch, they click.

A note on whether to include plain-text email

In most cases, we recommend that email marketers no longer include plain or rich text versions of email creative along with their HTML versions.

In the past, Internet Service Providers' (ISPs) formatting requirements demanded a text creative version. However, the growing sophistication of smartphones and the evolution of ISP infrastructure have eliminated this need.

While HTML content may not render properly for recipients who have images blocked (and also might not render well on some older devices), including a link to a web version of the creative in the email preheader is sufficient to address this concern. Mailing a plain text version along with HTML creative has a negligible effect on deliverability—only a very small percentage of users would view a text version.

Our exception to this recommendation applies to marketers targeting IT professionals, to whom we recommend sending both HTML and plain text versions. This particular audience has a higher propensity to request, read and engage with text-only emails.

Coding recommendations

The last thing any hardworking email marketing team wants is for a technical oversight to negatively impact the subscriber experience. This section details today's email coding best practices to help ensure that your HTML emails render properly for the widest array of email clients.

Despite the simplicity of the final product delivered to the inbox in email marketing—a single HTML file with no interactive components—coding for email is a challenging discipline that requires highly specialized skills and industry knowledge, which often differ from the coding skills and knowledge required for web development.

The primary challenge in coding email-friendly HTML involves the number of platforms developers must contend with. In regular web development, one must code for different web browsers and operating system platforms. Browsers and platforms must still be considered when coding for the email channel, but the more crucial consideration is how different webmail clients (such as Yahoo, Gmail, Hotmail and AOL) and desktop email clients (such as Outlook 2007 and 2010) interpret and render coded HTML.

Significant considerations for coding HTML email

- Suppression of images in the inbox environment is commonplace.
- Support for standard CSS positioning is inconsistent amongst email platforms.
- Using table-based code is essential for achieving consistent and accurate email rendering.
- Some web-based email platforms (notably Gmail) strip out `<STYLE>` tags and/or anything in the `<HEAD>` section of an HTML file.
- There is inconsistent support for standard interactive features of web development including:
 - [JavaScript](#)
 - [Flash](#)
 - [Embedded video](#)
 - [Forms](#)
- There are more platforms to be concerned with than in regular web development, and many of them are frequently updated without notice. Working with a multitude of platforms means more opportunities for mistakes. What looks fine in one platform may look terrible in another.

Establishing an effective process for email coding

Following a consistent end-to-end process from design hand-off to coding to testing will help your team ensure the best results and establish a framework for repeatable success.

Before coding

We recommend standardizing your design team on Photoshop, the tool of choice for web and email designers. Photoshop is the industry standard for working with graphics files, making it easy to share assets with both internal and external resources.

Establish a standard system for reviewing multiple-version emails. If design modules will change between versions, you'll make a choice about whether to have the modules embedded side by side in the PSD or to use layer comps to switch from one version to the next with a click. It's essential that all teams are aware of how versioning is being portrayed.

When going through creative reviews, it's best for email coders to get involved before final approval of the design, particularly when the team creates responsive email. When working on a responsive email, it's helpful to draw the table structure for the desktop version and check that it can be used to create the responsive version of the email. Keep in mind that if there is something in the responsive version, it **MUST** have some sort of code representation in the desktop version that can be manipulated to display the responsive version. If the desktop version can't support the responsive version, collaborate with the design team around responsive email limitations and see what can be done to come up with a design that supports both desktop and responsive elements.

Reviewing assets after approval

As you prepare to slice and export images from the Photoshop file, review the design to identify which areas will be clickable, and determine which items will be HTML text (system fonts such as Arial, Verdana, etc.) and which will be graphical.

Creating slices in Photoshop and exporting graphics

Make sure to use a consistent naming convention for image files (i.e., "title_main.gif" for the main title text). Although this can be cumbersome, it is helpful when making templates that will be reused. When naming images, try not to use words that may trigger browser-installed ad block programs. Some such keywords in particular include: ad, promotion, banner.

Coding

Focus on clean, organized code to avoid render errors and make it easier to share files with other teammates. Because DIV- and CSS-based layouts are not supported by many email clients, you will need to use tables in your HTML. The bulk of the rest of this document will provide tips and considerations for coding your emails.

Render testing

While there are great tools for previewing your emails across multiple platforms at once, nothing beats manually reviewing your campaigns on the major email platforms, on browsers and on your target audience's favorite devices.

A deeper look at coding particulars and top mistakes to avoid

The following pages offer a deeper look at coding support and specifics across the major email environments.

Unsupported HTML features

Several common HTML features used on the web are not supported in email environments.

Scripts, Flash and Embedded Video

Some email clients still disable interactive JavaScript and other scripting elements within HTML email, often categorizing emails containing scripts as spam. Because of this, JavaScript (and any other scripting language) should be used cautiously and tested with segmented lists.

Cascading style sheets (CSS)

Externally linked CSS files—and anything else in the `<HEAD>` section of an HTML document—are commonly stripped out by email client platforms. In addition, CSS class definitions in `<STYLE>` tags are not well supported by all email clients, most notably Gmail. For non-responsive emails, styling code should be included inline within HTML elements instead of in external files or `<STYLE>` tags. For responsive emails, external CSS files are recommended as webmail email clients don't recognize or render the code in the externally linked CSS.

If you are interested in experimenting with styling properties and attributes, Campaign Monitor has a list of styles and their support across different browsers and environments. It is updated with relative frequency and can be found here: <http://www.campaignmonitor.com/css/>.

Unnecessary tags and code

To keep the file size to a minimum and for more readable and maintainable code, always try to eliminate as many tags and attributes as possible. Here are some items that aren't necessary:

META	And other tags commonly included in the <code><HEAD></code> section will not be recognized in webmail clients
BORDER="0"	Not required in <code><TABLE></code> tags
VALIGN	Only required when overwriting the default

<TBODY> and <TH> Never required

NOWRAP Not required (and causes display problems in some email clients)

At the time of writing, Gmail and Yahoo both truncate emails over a certain character count, 102k characters and 100k characters respectively. They do provide a link that allows a user to view the full email, but the initial email may look broken in the inbox environment. This makes it all the more important to make sure your code is not bloated with unnecessary tags.

Basic layout: working with tables

Because we cannot use the most recent HTML and CSS standards for laying out the design of an email, we invariably rely on tables in combination with inline styling to position and style elements.

Important considerations

- Utilize nested tables to create the desired combination of rows and columns for the layout of the email.
- Specify widths accurately in `<TABLE>` and `<TD>` tags when positioning elements. Multiple columns should also add up correctly to the overall table width.
- Set `CELLPADDING` and `CELLSPACING` to “0” and specify a `WIDTH` value. Do not use any `BORDER` attribute other than “0”; Entourage and Gmail have spotty support for border values on tables.
- Always ensure the `WIDTH` of the table is equal to the sum of the `<TD>` tags contained within the table; Outlook 2007, Internet Explorer 6 and Safari are all unforgiving of discrepancies between `<TABLE>` and `<TD>` `WIDTH` sums.
- Use `COLSPAN` and `ROWSPAN` with care. While they are great assets in table layouts, if your rows and columns are not accurately represented within the `COLSPAN` and `ROWSPAN` your email layout will break. The Outlook platform is the most sensitive to inaccurate `COL-` and `ROWSPANS`.
- Ensure that `COLSPAN` and `ROWSPAN` values match exactly with the sum total of columns being merged.
 - For example, if there are only three `TDs` in a row, but a `COLSPAN` value of 4 is specified, Outlook 2007 will insert additional space into the table for the “fourth” column, which will throw off your table layout (the same holds true for `ROWSPAN` if you specify a value greater than the number of actual rows).

Using outer tables to encapsulate content

To make sure the design aligns correctly and to catch any unclosed table elements, an outer table should be used to enclose the entire HTML design.

(NOTE: One caveat around this is that Outlook 2007+ automatically inserts page breaks into HTML elements that are over 1800 pixels in length. The appearance of this error is hard to track because it shows itself in several forms; for example, as a half-loaded image, a large space between email sections or sometimes just an extra 10 pixels in your preheader. When in doubt, turn your email into smaller chunks for testing, or remove sections to make the email shorter.)

Example: Container table

The outer table in this example ensures the design is left aligned, and that everything within has a white background. It also encloses the rest of the design so that any broken/unclosed table elements won't spill over into the email browser window.

```
<BODY>

<TABLE CELLPADDING="0" CELLSPACING="0" WIDTH="600"
BGCOLOR="#FFFFFF">

<TR><TD ALIGN="LEFT">

<!-- design is coded here -->

<!-- end design -->

</TD></TR>

</TABLE>

</BODY>
```

Spacing: Padding, margin and spacer.gifs

Spacing is complicated to get right across email platforms due to inconsistent implementation of the padding and margin properties. In early 2013, Outlook.com (formerly Hotmail) began stripping margins from email code. As a result, to achieve the desired spacing in email, you should use the padding property to achieve horizontal spacing. Because `padding top` and `padding bottom` are poorly supported, we recommend using `
` tags for vertical spacing between text if possible. If that's not possible because spacing can't be achieved through line breaks, we recommend using empty tables for vertical spacing.

`<DIV>` tags are extremely helpful when creating space around text, as the `<P>` tag has inconsistent support across email environments. `<DIV>`s should be used

within a <TD> tag. A few things to take note of:

- Don't use any margin property as it is not supported in Outlook.com. Plan the structure of the code with this in mind.
- Use padding on the <DIV> tag, not the <TD>.
- Be careful to close <DIV> tags correctly, as incorrect closing will cause dramatic rendering issues in Gmail.

One thing that complicates matters further is that the Outlook platform doesn't support padding but does support margins. The recommended way to achieve your spacing in the Outlook platform is to specifically use MSO (Microsoft Office) styles to apply padding in Outlook. These styles are embedded in HTML comment tags and are not recognized by any environment other than Microsoft Office. If you go this route, make sure that the tool you use to send email does not remove your comments. Microsoft Office doesn't apply multiple classes to an element, so you cannot stack multiple classes (`class="className1 className2"`) on elements as you would in regular web development. The example below shows the use of classes to achieve spacing in Microsoft Outlook.

Example: Using MSO-specific styles to achieve horizontal spacing around text:

```
<body>

<!--[if gte mso 9]>

<style type="text/css">

    .ml15{ padding:0 !important; margin-left:15px !important;}

    .mr15{ padding:0 !important; margin-right:15px !important;}

    .mm1{ padding:0 !important; margin-left:15px !important;
margin-right:15px                !important;}

</style>

<![endif]-->

<div style="font-family:Arial, verdana, sans-serif;font-
size:13px;line-height:18px;color:#333333; padding-left:15px;
padding-right:15px;" class="mm1">Padding left and padding right
example.</div>

<div style="font-family:Arial, verdana, sans-serif;font-
```

```

size:13px;line-height:18px;color:#333333; padding-left:15px; "
class="ml15">Padding left example.</div>

<div style="font-family:Arial, verdana, sans-serif;font-
size:13px;line-height:18px;color:#333333; padding-right:15px;"
class="mr15">Padding right example.</div>

</body>

```

You can also achieve spacing using extra <TD>s to provide horizontal spacing, but the method presented above is better because it keeps character counts lower and keeps code more readable.

Example of vertical spacing between text

```

<div style= "font-family:Arial, verdana, sans-serif;font-
size:13px;line-height:18px;color:#333333; padding-left:15px;"
class="ml15">Padding left example.</div>

<table cellpadding="0" cellspacing="0"><tr><td height="10"></
td></tr></table>

<div style="font-family:Arial, verdana, sans-serif;font-
size:13px;line-height:18px;color:#333333; padding-right:15px;"
class="mr15">Padding right example.</div>

```

Outlook 2013 has introduced a new bug for email coders to consider. When creating a <TD> that is less than 19 pixels, the <TD> will expand to be 19 pixels tall. To avoid this, you can put a spacer gif in your <TD>. Also be sure to set both the height and line height on the <TD>.

Example: Creating a <TD> shorter than 19 pixels.

```

<tr><td height="1" bgcolor="#dc6a2b" style="line-
height:1px;"></td></tr>

```

Example: Creating space around text and an image. Note how the width of the <TD> containing the image is equal to the padding and the image width combined.

```

<td align="left">

<div style="line-height:18px; padding-left:10px;">

```

```
It&rsquo;s tricky to keep up with constantly changing airfares...

</div>

</td>

<td width="310"><div style="padding-left:10px;"><img src= "image/img.
gif""width="300" height="176" style="display:block;" /></div></td>
```

Due to the Outlook's poor support of padding and Outlook.com's removal of margins, we don't recommend using either the margin or padding properties on a `<TD>`.

Working with images

Background images, alt attributes and images in Gmail and Hotmail are areas of concern.

General notes on image formatting

- GIF format is best for simple line art and purely graphical elements.
- JPEG format is best for rich, multi-layered and photographic elements.
Images exported in JPEG format should normally be optimized at 60% quality, though in some rare cases you may need to increase the quality to as much as 75% to reduce aliasing artifacts.
- PNG format is best for high-quality transparency and transparency on a non-white background. Be careful, though, as it also has the highest footprint of the image family.

Background images

Because not all email platforms support the `BACKGROUND` property—most notably Outlook—care should be taken when using background images. Be sure to include a matching `BGColor` value whenever using a background image so that any text overlaid on the section will still display for users without background images and/or with images disabled. It is actually possible to mimic background images in Outlook by layering text elements over image elements using vector elements. The website <http://emailbg.net/> has a tool that generates code to insert into your email. It may take a little adjusting to get the text positioned just so, but is a very helpful resource.

Graphics: ALT attributes

Always use the ALT attribute on any tags that include text. The ALT attribute should match the contents of the image, but not be overly long. In the case of long sentences or paragraphs, ellipses can be used to indicate that there is more text (and encourage users to enable images).

Example: ALT text for graphical paragraphs.

In this example, the coder added an ellipsis at a logical breaking point. The ALT text should read: "Inspired by the glamour and sophistication of New York apartment style..."

Inspired by the glamour and sophistication of New York apartment style, our Penthouse furniture and accessories bring contemporary polish into any room.

Issues in Gmail and Hotmail

Images need to be displayed as block level items to ensure proper rendering in Gmail and Hotmail for users viewing with Firefox, Chrome and Safari. Simply add `style="display:block;"` to each image.

```

```

For images inline with text (for example, arrow images next to text links) you should include `style="display: inline;"` instead of `display:block;`.

```
<a href="http://example.com">Here is a link with an inline arrow at the end </a>
```

Graphics within <TD> tags

There are two things to be aware of when an tag is contained within a <TD> tag:

(1) If the image is shorter than 19 pixels, you need to specify the height and line-height <TD> tag as well as in the image tag.

```
<tr><td width="30"height="5" style="line-height:5px;"></td></tr>
```

(2) If an tag is the last or only element within a <TD> tag, the closing </TD> tag should immediately follow the tag with no spaces or breaks. This is because some browsers will interpret any space (space character, tab, enter, etc.) as a desired space, which results in a gap after the image.

```
<tr><td width="250"></td></tr>
```

Image maps & embedded links within images

Image maps are not supported by all email platforms. For links embedded within a larger image, the image should be sliced in Photoshop so that the linkable area is a separate image file that can be coded with an anchor link in the HTML file.

(1) Determine image before slicing. The text “wshome.com” should be a link, but the rest of the paragraph should not.

Visit our online catalog at
[wshome.com](#) today to shop
the complete collection.

(2) The image is sliced into multiple files that will be coded in a table. The image outlined in yellow will be coded with an anchor link enclosing the image.



Before going this route, it may be good to discuss with your employer or client the benefits of linking the entire image, which results in a larger clickable area.

Working with text & links

General tips

- Define font size in pixels, not points. Pixel is the Web standard because it is relative to screen resolution. Points are absolute length, and different browsers and platforms may display these values differently. When scaled up or down, pixels may look worse than EMs (another relative unit that depends on user browser settings). As images are also displayed in pixels, it makes more sense to keep the font size relative to the size of the images being used in the email design, and pixels are the preferred font unit for HTML email.
- Use the `line-height` style attribute to match line spacing to the design file.
- Use one or two different font types maximum, and provide a sufficient list of alternate fonts in your inline style declaration specification for a given piece of copy. The fonts should be listed in terms of: desired, acceptable or generic type (e.g. Arial, Verdana, Sans-serif).
- Don't use variants of the CSS shorthand “font” notation (i.e. “font: 12px arial;”). The font-family, size, color and line height should all be specified separately, similar to the example below. Gmail in particular has issues properly parsing shorthand font notation.

```
<div style="font-family:Arial, Verdana, sans-serif;font-size:12px; line-height:17px;color:#000000;"> ... </div>
```

Using web-safe fonts

For serif fonts use Georgia, Times or Times New Roman. For sans-serif fonts use Arial, Verdana or Tahoma.

Using
 to achieve text breaks

Be cautious when using the
 tag to force manual breaks in blocks of text. Some email clients, such as Outlook 2007 and the iPhone, render leading and font sizes larger than normal and may wrap text before the manual line breaks as in the example below. It is best to let text wrap based on the container it's placed in whenever possible.

Code:

Here is some text that has manual
 line breaks added to it. This may not look
 as nice on an iPhone.

Web-based email client

Here is some text that has manual line breaks added to it. This may not look as nice on an iPhone.

iPhone

Here is some text that has manual line breaks added to it. This may not look as nice on an iPhone.

Using hex values for color

Don't define colors using shortcut words (i.e., COLOR="RED"); always use the hexadecimal value as specified in the final creative file (i.e., COLOR="#bb0000").

Example: HTML text usage.

In the example below, a combination of standard font and style attributes are used for the best results across multiple email browsers.

```
<td align="left">

  <div style="font-family: Arial,Verdana; font-size: 11px; color:
  #333333; line-height: 15px;">

    Here is some example text that is being given the proper
    Style treatment.

  </div> </td>
```

Special characters

Special text characters should always use the proper HTML escape codes to ensure recipients on all platforms are able to view the text properly. For example, the character é should be coded using the HTML escape code `é`. For apostrophes, use the single quote escape code `’`. Other escape codes can be found by searching online for “HTML Escape Codes.”

Links

With hyperlinks, keep in mind that many web-based email providers and client email programs have their own parsing engines and have widely varying methods for displaying text hyperlinks. Such parsing engines will typically apply their own styling to text hyperlinks, which often clash with the creative design of the EDM. A combination of anchor tags with style attributes provides the best results across a wide variety of email services (see example below).

Other considerations with hyperlinks

- Do not include commencing and ending spaces.
- Do not include commas, periods and other punctuation within hyperlinks.
- Always include `http://` at the beginning of web addresses.
- Do not use dotted decimal addresses (e.g., `94.31.231.18`) in like URLs because many email clients may categorize the email as spam.

Example: Hyperlink formatting.

Inline style attributes are used for the best results across multiple email browsers.

```
For best results <a href="http://www.example.com" style="font-family:Verdana; font-size: 11px; color: #333333; text-decoration: underline">click here</a>.
```

Responsive email

When coding responsive emails, simpler is better. Changing the order of modules is hard, and it makes the email prone to breaking. An example of changing the order is going from having the structure hero module, secondary module A, secondary module B in the non-responsive design to having the structure secondary module A (as a banner), the hero, secondary module B. It's best to keep elements showing up in the responsive version in the same order as they appear in the non-responsive version, left to right and top to bottom. This may seem limiting, but there are different ways to structure code based on the responsive outcome, so it's best to examine both designs in detail before beginning code.

Code structure

When you're coding an email, keep in mind that the HTML for both the non-responsive and responsive version is in one file. When you make the non-responsive email, it needs to have the necessary elements and code structure for the responsive version. When starting out, it's often a good idea to draw out your table structure on the email and write out how it will transform from the non-responsive to the responsive version of the email.

Make your classes as reusable as possible. That is, avoid creating classes for specific elements, where `class="image1" .image1{ width:200px !important; height:150px !important; }`. Instead, create one class per style that you want to apply. This makes the classes easily reusable on another element. It can be helpful to break the CSS into sections—a height section, a width section, a fonts section, a margin section, etc.—and to alphabetize your classes within that section.

When your code is well-organized, classes will be easily found, ensuring that you don't have duplicate classes in your code.

```
@media screen and (max-width: 600px){  
  
  /* generals */  
  
  *[class].block{ display:block !important; }  
  
  *[class].hide{ display:none !important; }  
  
  /* height */  
  
  *[class].h5{ height:5px !important; }  
  
  *[class].10{ height:10px !important; }  
  
}
```

External style sheet

We recommend using external style sheets to contain your CSS. Some webmail clients, such as AOL and Outlook.com, have begun to render responsive styles. Advancement of email platforms is to be applauded, but it turns out that Internet Explorer 9 can't apply the `display:block;` style to a `<TD>`, which results in a broken rendering when the responsive styles trigger there. There are some fixes regarding this involving the use of the float property, but that tends

to introduce issues in combination with some other code. Using an external style sheet is a good solution because both Android and iOS native mail clients recognize the `<LINK />` tag and render your responsive styles.

Testing

Testing is very important for responsive emails—be sure to test on your audience’s most used devices. If you’re testing the iPhone, test the most recent version of the OS as well as the version just before that. When testing Android, make sure you test more than one device as some phone manufacturers alter the OS, and the functionality of responsive emails may be affected by those changes.

Platform-specific fixes

Earlier in this document we mentioned that `<STYLE>` tags are not supported by some of the major email platforms and that one shouldn’t include CSS definitions with a `<STYLE>` tag. There are, however, a few cases in which we do recommend including a few simple items within a `<STYLE>` tag to ensure proper rendering. The example below is used to fix issues on various mobile devices, which tend to resize text, resulting in brokenly rendered email.

Example: Fix for mobile device text resizing.

```
<style>

html { -webkit-text-size-adjust:none; -ms-text-size-adjust: none;}

</style>
```

The `–webkit...` and `–ms...` line forces mobile devices to render text at the pixel size specified in the HTML file instead of at the minimum font size specified in the subscribers’ operating system preferences. Not including this rule can lead to dramatically increased font size within emails. Include the code below just after the `<BODY>` tag to keep your font sizes as specified.

By default, Yahoo currently applies a blue color to text links as well as to certain keywords identified by the platform. The first Yahoo example demonstrates a method to use for intended links, while the second can be used if a generated link is causing a design or linking issue.

Example: Fix for links in Yahoo.

Include the code below with the `<A>` tag. The `` tag will prevent Yahoo mail from changing the color of text in your links for words the Yahoo platform automatically generates links for.

```
For best results <a href="http://www.example.com" style="font-family:
Verdana;
```

```
font-size: 11px; color: #333333; text-decoration: underline"><span
style="color:
```

```
#333333;">click here</span></a>.
```

Example: Fix for links generated by Yahoo.

If a word is being targeted by the Yahoo platform, the following fix can be applied after testing to maintain the intention of the text. Include the `<A>` tag, but leave out the `HREF`. We recommend applying this method only when necessary.

```
Buy our <a style="font-family: Verdana; font-size: 11px;
color: #333333; text-decoration: none;"><span style="color:
#333333;">lawn chairs</span></a> by clicking <a href="http://
www.example.com" style="font-family: Verdana; font-size: 11px;
color: #333333; text-decoration: none;"><span style="color:
#333333;">here</span></a>.
```

Coding for Outlook 2007, 2010 and 2013

Microsoft Outlook 2007, 2010 and 2013 warrant special attention due to their poor support for standard HTML and CSS features, coupled with Outlook's wide adoption rate. More than 40% of business email users use Outlook 2007+, and the number will only increase in coming years. Outlook 2013 has recently been released and has been confirmed to have the same flaws as its predecessor.

Major issues to be aware of:

- It offers no support for background images.
- It offers no support for animated GIFs.
- It offers limited support for CSS.
- It provides unpredictable rendering behavior.

Background images

Background images are not easily supported on the platform, creating a conflict between using HTML text—which usually leads to higher click-through and conversion—and achieving optimal design that will look the same across email platforms.

We recommend striving for a balance. In most cases, it's important to use HTML text when possible, but it's just as important to ensure the final coded files will

still look attractive when they reach the inbox in Outlook 2007+.

Example: Background image with BGCOLOR.

Try to limit the sections of HTML files that rely on background images. When you do use background images, always include a BGCOLOR value that closely matches the image so that the design will not have a glaring white block where the background image should be.

```
<td height="23" bgcolor="#cde9ff" background="images/111009_nbg.jpg">
```

As stated above, it is possible to mimic background images in the Outlook platform by using vector elements to layer text over images, and the website <http://emailbg.net/> has a tool that generates code to insert into your email. Please note, you cannot nest a table that uses this background image method within another table already using that same method.

Animated GIFs

Animated GIFs do not rotate in Outlook 2007+. In these platforms, only the first frame of the animation displays, in essence making the GIF a non-animated file. If you are using an animated GIF in your message, be sure that the first frame can stand on its own. For example, don't start an animated GIF with a fade-in from a solid color.

Vertical expansion of <TD> tags

Outlook 2007 and 2010 expand <TD> tags that are under 19 pixels in height to be 19 pixels tall. In the past, the way to avoid this behavior was to add a height to the <TD>. Outlook 2013 has complicated matters, and that fix no longer works. There are two fixes available that work, although one doesn't work on 1-pixel tall <TD>s.

This example works on <TD>s from 2 to 19 pixels in height and doesn't use as much code as the second fix. Unfortunately, it doesn't work on <TD>s that are 1 pixel in height in Outlook 2010.

```
<table cellpadding="0" cellspacing="0"><tr><td height="17" style="font-size:17px; line-height:17px;">&nbsp;</td></tr></table>
```

This works on all <TD>s under 19 pixels.

```
<table cellpadding="0" cellspacing="0"><tr><td height="12" style="height:12px; line-height:12px;"></td></tr></table>
```

Limited support for CSS

Caution should be used when using CSS properties in Outlook 2007. Positioning properties such as padding and margin can be particularly problematic. A few do's and don'ts to keep in mind:

- The `margin-top` property rarely works correctly, so avoid using it in your code.
- `padding-top` will always be applied to any `<TD>` tags within a single row, even when specified in only one of the `<TD>` tags. Avoid using padding on `<TD>` cells. Instead, using spacer tables if you need to achieve spacing.

Example: Applying top spacing in Outlook.

```
<tr>

<td width="200">

    <table cellpadding="0" cellspacing="0"><tr><td height="10"
style="font-size:10px; line-height:10px;">&nbsp;</td></tr></
table>
```

This text has 10 pixels of space before it starts</td>

```
</td>

<td width="200">This text doesn't need top padding</td>

</tr>
```

Poor and sometimes unpredictable rendering behavior

As described above, be cautious when using the `ROWSPAN` and `COLSPAN` attributes. For example if you specify a larger `ROWSPAN` value than there are rows in your table, Outlook 2007 will make your total table taller than it would have been if you had used the correct `ROWSPAN`.

Be careful of very long emails (over 1,800 pixels). Due to Outlook's dependency on the Word rendering engine and its page-break feature, it will create gaps in sections of the code around the 1,800-pixel mark. Depending on the coding context, the issue can sometimes be resolved, and sometimes it can't be resolved without completely recoding the HTML file from scratch.

Outlook.com's larger default line-height

For the past few years, Hotmail.com/Outlook.com have been applying a

line-height of over 100% to emails. This results in increased spacing around text in emails. To avoid it, we recommend using the following style, which targets the Outlook.com email code and resets the line-height to 100%.

```
<style type="text/css">

.ExternalClass *{line-height:100%; }

</style>
```

Testing: The final step before launching

While not the most time-intensive step, testing is arguably the most important. We've discussed above how the major email platforms all have different HTML rendering engines, creating more complex coding challenges for email than for regular web development. In addition, the major web-based email platforms are constantly rolling out modifications to their systems unannounced, and these changes often have an impact on how messages will be rendered. Because of this, even if you are using a template that does not change, it's important to test your messages on an ongoing basis.

Code validation

Make sure all your HTML code has been validated and ensure that no nesting errors are present. Improper tag closing will cause rendering errors in many email clients. Most HTML editors such as Dreamweaver include built-in HTML validation capabilities. We recommend validating against the XHTML 1.0 Transitional standards.

Render testing

While there are great tools for previewing your emails across multiple platforms, nothing beats manually reviewing your campaigns on the major email platforms. IntelliClick recommends sending test messages to the major web-based email clients (Yahoo, Gmail, Outlook.com and AOL) as well as reviewing how the message renders in Outlook. Make sure to check in Firefox, Chrome and Internet Explorer 8+. If you have the ability to check both PC and Mac platforms across a variety of browsers, that's even better. If your audience uses mobile devices to open your emails, try and find out what devices are the most common and test your emails on those devices as well.

Breaking the rules

The best practices described above are some of the safest and most reliable techniques for designing and coding your email messages, but you may find success when ignoring a few of them. In fact, constant experimentation is good—and often necessary—to tackling new problems and making sure any email you send can render in the inbox. When using a questionable or new practice, test it first to determine how your list responds. If the data, sales or responses you receive outweigh the potential functionality issues, proceed with caution and enjoy your success.