AGENT SUMMIT

ePromo Guidelines

HTML

- Maximum width 700px (length = N/A)
- Image resolution should be 72dpi
- Maximum total file size, including all images = 200KB
- Only use inline CSS, no stylesheets
- Use tables, rather than layout
- Use more TEXT instead of graphics when possible
- The message should make sense without the graphics since recipients can BLOCK images
- Include at least 2 links within the message
- Most important info should appear at the top of the creative (emails are scanned in preview panes to determine relevance)
- Avoid thick borders, spam-like words, and excessive large fonts

DUE DATES

ePromo creative is due three business days before your scheduled send date. The AE web department will send a test message prior to the official send date.

NOT ALLOWED

- Flash
- Animated GIFs
- Javascript
- Form Fields
- Defining anything outside of the <body> tag, including <style>
- Background Images

PLAIN TEXT VERSION

If your ePromo is a single image, or a series of images only, please provide a plain-text version with your ePromo design. When all ePromos are sent, they go out with an HTML version and a plain-text version. The plain-text versions will go to those subscribers who specifically request to get plain-text emails. The number of plain-text subscribers is low, but this will maximize the number of subscribers that receive your message.

Email Campaign Guidelines and Best Practices

There are numerous factors that go into creating a solid email marketing campaign, but one of the most challenging aspects is developing an email that renders well across all the popular email clients.

Below is a list of tips that will help you get the best results.

Use Tables For Layout

Because clients like Gmail and Outlook 2007 have poor support for float, margin and padding, you'll need to use tables as the framework of your email. While nested tables are widely supported, consistent treatment of width, margin and padding within table cells is not. For the best results, when coding your table structure set the width in each cell, not the table.

When you combine table widths, td widths, td padding and CSS padding into an email, the final result is different in almost every email client. The most reliable way to set the width of your table is to set a width for each cell, not for the table itself. Such a structure would look something like this:

```
<table cellspacing="0" cellpadding="5" border="0">
<tr>
   <td width="140"></td>
   <td width="360"></td>
</tr>
</table>
```

Never assume that if you don't specify a cell width the email client will figure it out. It won't. Also avoid using percentage based widths. Clients like Outlook 2007 don't respect them, especially for nested tables. Stick to pixels. If you want to add padding to each cell, use either the cellpadding attribute of the table or CSS padding for each cell, but never combine the two.

Use Table Nesting When Appropriate

Table nesting is far more reliable than setting left and right margins or padding for table cells. If you can achieve the same effect by table nesting, that will always give you the best result across the buggier email clients.

Use A Container Table For Body Background Colors

Many email clients ignore background colors specified in your CSS or the <body> tag. To work around this, wrap your entire email with a 100% width table and give that a background color. That code would look something like this:

```
<table cellspacing="0" cellpadding="0" border="0" width="100%">
<tr>
   <td bgcolor="#000000">
   Your email code goes here.
   </td>
</tr>
</table>
```

You can use the same approach for background images too. Just remember that some email clients don't support them, so always provide a fallback color.

Avoid Unnecessary Whitespace in Table Cells

Where possible, avoid whitespace between your <td> tags. Some email clients, like Yahoo! and Hotmail, can add additional padding above or below the cell contents in some scenarios, breaking your design for no apparent reason.

Always Move Your CSS Inline

Gmail is the culprit for this one. By stripping the CSS from the <head> and <body> of any email, we're left with no choice but to move all CSS inline. The good news is this is something you can almost completely automate. Free services like Premailer will move all CSS inline with the click of a button. If you leave this step to the end of your build process so you can utilize all the benefits of CSS.

Avoid Shorthand For Fonts and Hex Notation

A number of email clients reject CSS shorthand for the font property. For example, never set your font styles like this.

```
p {
   font:bold 1em/1.2em georgia,times,serif;
}
```

Instead, declare the properties individually like this.

```
p {
   font-weight: bold;
   font-size: 1em;
   line-height: 1.2em;
   font-family: georgia,times,serif;
}
```

When declaring the color property in your CSS, some email clients don't support shorthand hexadecimal colors like color:#f60;. Instead declare your color like this: color:#ff6600;. Stick to the longhand approach for the best results.

Paragraphs

Just like table cell spacing, paragraph spacing can be tricky to get a consistent result across the board. Some designers revert to using double
 or DIVs with inline CSS margins to work around these shortfalls, but recent testing showed that paragraph support is now reliable enough to use in most cases (there was a time when Yahoo! didn't support the paragraph tag at all).

The best approach is to set the margin inline via CSS for every paragraph in your email, like this:

```
p {
   margin: 0 0 1.6em 0;
}
```

Again, do this via CSS in the head when building your email, then use Premailer to bring it inline for each paragraph later.

If part of your design is height-sensitive and calls for pixel perfection, avoiding paragraphs altogether and set the text formatting inline in the table cell. You might need to use table nesting or cellpadding / CSS to get the desired result. Here's an example:

```
<td width="200" style="font-weight:bold; font-size:1em; line-
height:1.2em; font-family:georgia,'times',serif;">your height sensitive
text</td>
```

Links

Some email clients will overwrite your link colors with their defaults, and you can avoid this by taking two steps. First, set a default color for each link inline like so:

```
<a href="http://domain.com/" style="color:#ff00ff">this is a link</a>
```

Next, add a redundant span inside the a tag.

```
<a href="http://domain.com/" style="color:#ff00ff"><span style="col-
or:#ff00ff">this is a link</span></a>
```

To some this may be overkill, but if link color is important to your design then a superfluous span is the best way to achieve consistency.

Images in HTML Emails

The most important thing to remember about images in email is that they won't be visible by default for many subscribers. If you start your design with that assumption, it forces you to keep things simple and ensure no important content is suppressed by image blocking.

With this in mind, here are the essentials to remember when using images in HTML email:

1. Avoid spacer images

While the combination of spacer images and nested tables was popular on the web ten years ago, image blocking in many email clients has ruled it out as a reliable technique today. Most clients replace images with an empty placeholder in the same dimensions, others strip the image altogether. Given image blocking is on by default in most email clients, this can lead to a poor first impression for many of your subscribers. Stick to fixed cell widths to keep your formatting in place with or without images.

2. Always include the dimensions of your image

If you forget to set the dimensions for each image, a number of clients will invent their own sizes when images are blocked and break your layout. Also, ensure that any images are correctly sized before adding them to your email. Some email clients will ignore the dimensions specified in code and rely on the true dimensions of your image.

3. Avoid PNGs

Lotus Notes 6 and 7 don't support 8-bit or 24-bit PNG images, so stick with the GIF or JPG formats for all images, even if it means some additional file size.

4. Provide fallback colors for background images

Outlook 2007 has no support for background images (aside from the hack mentioned above to get full page background images working). If you want to use a background image in your design, always provide a background color the email client can fall back on. This solves both the image blocking and Outlook 2007 problem simultaneously.

5. Only use images that are 72dpi.

Using print ready images at 300dpi can cause issues.

6. Use the display hack for Hotmail

For some inexplicable reason, Windows Live Hotmail adds a few pixels of additional padding below images. A workaround is to set the display property like this:

```
img {display:block;}
```

This removes the padding in Hotmail and still gives you the predictable result in other email clients.

7. Don't use floats

Both Outlook 2007 and earlier versions of Notes offer no support for the float property. Instead, use the align attribute of the img tag to float images in your email.

```
<img src="image.jpg" align="right">
```

If you're seeing strange image behavior in Yahoo! Mail, adding `align="top"` to your images can often solve this problem.

What About Mobile Email?

Responsive web design is the practice of crafting websites in a way that they are usable regardless of which device is used to access them. The responsive web is largely reliant on media queries to drive that adaptation. More recently, this approach has been brought to the world of HTML email.

A media query follows a pretty simple structure. For the purposes of email, the media query's styles are nested within the emails <style> tag:

```
<style type="text/css">
    .standardCSS{
        color:#505050;
        font-size:15px;
    }

    @media only screen and (max-width:480px){
        .mediaQueryCSS{
            color:#CCCCCC;
            font-size:20px;
        }
    }
</style>
```

Basically a stylesheet within a stylesheet, media queries are built from a few parts:

```
@media only screen and (max-width:480px){
}
```

First, they're always opened with the @media at-rule. Next comes a keyword. In this case, 'only', which restricts the display of the media query styles to the specified media type. After that, the 'media type' is set. The most commonly used media types are screen and print, providing different style rules for displays and printers, respectively. Another keyword, 'and', follows, and, finally, the 'media feature', which is the meat of the media query.

There are many different media features available, but we're really only concerned with using two of them:

```
max-width
max-device-width
```

The difference between them is how they calculate the maximum width that will trigger the media query. The feature max-width is measured against the available space of a browser, while max-device-width looks at the size of the device's screen. Essentially, max-width covers everything and max-device-width covers small displays.

We want to cover everything. In this case, the media feature is used to target an area that measures 480 pixels in width or smaller. 480px is the standard width of a mobile phone's screen in landscape orientation, and a good standard breakpoint to use in your code.

Media Queries and Inline Styles

It is necessary to inline CSS styles, either by hand or automatically before an email is sent. Since media query styles work on a trigger and are not default styles, it doesn't make sense to inline any of it. So, the email's normal CSS needs to be inlined and the media query CSS needs to override those styles once its triggered.

Because inline styles have the highest specificity in the cascade, it's necessary for every media query style rule you write needs to be marked with a !important declaration:

```
<style type="text/css">
    .standardCSS{
        color:#505050;
        font-size:15px;
    }

    @media only screen and (max-width:480px){
        .mediaQueryCSS{
            color:#CCCCCC !important;
            font-size:20px !important;
        }
    }
</style>
```

You can leave the media query styles in the <head> of your email, as clients that support media queries don't strip out the <head> or <style> areas.