Email Development Best Practice

- **Tables are your friend. When in doubt, table.[Ref. Campaign Monitor]**

Tables are by far the most reliable way to achieve a consistent layout.

Tables allow us to take advantage of the align attribute. Using align="left", we can cause tables to stack on top of each other on smaller screens.

Spacing Techniques in HTML Email.[Ref.Email on Acid] Popular Methods with Pros and Cons:

Methods	Pros	Cons	Conclusion
Cellpadding The space, in pixels, between the cell wall and the cell content	Support. This is widely supported across all email clients	The lack of ability to override it. We don't have the ability to do that because it's an HTML attribute, not CSS.	Use it in places where there is no need to change the cellpadding value for different email clients or devices. Ex: Adding a guttering around an entire email container.
Empty Cells empty <tds> added to force spacing around content.	Used with height and/or width property adds proper spacing required	It doesn't always work. Empty table cells don't always retain their height.	
Breaks the line break, 	Adding spacing between text or forcing line returns in your content	Various email clients are known to read the linebreaks at different height, making it virtually impossible to create a pixel perfect.	Only use breaks to break copy (text) in the email content, do not use it force spacing in other places.
** ** A non-breaking space ( ) is a common filler for otherwise empty cells.	This creates a space character which is transparent.	When very small cells are used for layout purposes, the non-breaking space will be too big.	
Padding and Margin Part of the CSS box model, padding and margin are common methods used in web development for creating spacing			It's best to stick to using padding on table cells, rather than on tables.We can add padding anyw here that could change dynamically.

Recommendation[by EOA]: ***Use of cell height and/or width values with the empty cells [ex:<td height="40" style="font-size:1px;line-height:1px;"], or colspan, rowspan and padding on the table cells.***
As long as you're not using borders, this should help you achieve identical layouts in all browsers.

Building an effective HTML email template [ref. Campaign Monitor]

Tips	Use

Set Widths in Each Cell rather than on the Table	The combination of widths on the table, widths on the cells, HTML margins and padding, and CSS margins and padding can be chaotic.
	Email clients are unreliable when it comes to deducing the correct width of a cell, so it's safest to explicitly set one.
	`<table cellspacing="0" cellpadding="0" border="0">`
	`  <tr>`
	`  <td width="150"></td>`
	`  <td width="350"></td>`
	`  </tr>`
	`</table>`
	
	Pixel widths are the most reliable, as using percentages can give you some wacky results, especially when using nested tables.

Nest Tables for Consistent Spacing	Even when margins and padding are supported by most email clients, results will be inconsistent. If the spacing is critical to you, try nesting tables inside your main table instead. Old school! To set your cell padding, either set it once on the whole table with the *cellpadding* parameter, or use CSS to set padding in individual cells. Mixing the two is likely to cause problems, and is best avoided.

Know your framework[ref. EOA]

There are a two popular approaches to coding an email layout: **Responsive**
Hybrid fluid / Spongy

Responsive	**Hybrid fluid**
Start with a 100% width table (to which we can apply styles that will affect the whole email) and then floating (using align="center") a fixed-width table in the center of this wrapper. Using media queries, the width of this fixed-width can be adapted to various screen sizes	Uses container tables that are set to width="100%" and constrained by a max-width style. On screens wider than the max-width, the table will reach its max and become no wider. On smaller screen, like tablets and phones, the table will naturally fill the available space.
The most common uses of this technique are to control font sizes, image sizes, and to make some tables become 100% width so that they will fill a mobile screen. You can also use media queries to hide content that isn't necessary for mobile users. This method uses : Media queries, Fluid media(images), Fluid grids(%).	This method is responsive in Gmail, Gmail app, and in almost all web clients except in Outlook (various versions), which doesn't support max-width. To overcome this limitation, "ghost tables" can be used to constrain the email in Outlook. Ghost tables are fixed-width tables built around the fluid tables, but wrapped in MSO conditional tags.

Key features to follow while coding email templates

Features	
CSS	***Use inline styles:*** ***Avoid shorthand CSS when possible:*** "margin-top: 5px" may work where "margin: 5px 0 0 0;" does not. ***Use full six-digit hex codes:*** Especially avoid three-digit hex codes. Some clients don't react well to these. ***Get used to !important*****:** To override styles added or modified by the email client (especially web clients) and override a default style with a mobile-specific one.
Images	***Use absolute addresses for images*** ***Space around Image:*** Use display:block and it will usually remove this extra spacing.

Mobile-friendly layouts and design elements [Ref. Campaign Monitor]

- Single-column layouts that are no wider than 500 to 600 pixels work best on mobile devices. They're easier to read, and if they fall apart, they'll do so more gracefully.

- Links and buttons should have a minimum target area of 44 × 44 pixels, as per Apple guidelines. Nothing is more unusable than clouds of tiny links on touchscreen devices.
- The minimum font size displayed on iPhones is 13 pixels. Keep this in mind when styling text, because anything smaller will be upscaled and could break your layout.
- When possible, use display: none; to hide extraneous details in your mobile layout. Elements like social sharing buttons may be great in the desktop inbox but aren't always easy to use by the recipient on mobile devices.

Coding mobile emails

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

@media only screen and (max-device-width: 480px) {
/* mobile-specific CSS styles go here */
    table[class=contenttable] { width: 320px !important; }
img[class="headerimage"] { width: 325px !important;  }
p[class="date"] { display: none !important; }
}
/* regular CSS styles go here */
table.contenttable { width: 640px; }

</style>
```

Building responsive layouts

There is an elegant way to create responsive 2-column layouts, without resorting to mile-long stylesheets in media queries**.**
One of the golden rules of email design is 'where possible, use HTML attributes instead of CSS'. For example, attributes like align="left" and cellpadding="10 are far more reliable than their approximate CSS equivalents,float: left; and padding: 10px.

```
<table width="640" border="0" cellpadding="0" cellspacing="0">
  <tr>
    <td>
      <table width="320" border="0" cellspacing="0" cellpadding="20" align="left">
       <tr>
         <td><p>Column Left Content</p></td>
       </tr>
      </table>
      <table width="320" border="0" cellspacing="0" cellpadding="20" align="right">
       <tr>
        <td><p>Column Right Content</p></td>
       </tr>
      </table>
   </td>
</tr>
</table>

This can be responsively coded as


<style type="text/css">
  @media only screen and (max-device-width: 480px) {
    table[class=contenttable] {
       width:320px !important;
    }
  }
</style>

<table width="640" border="0" cellpadding="0" cellspacing="0" class="contenttable">
```