

Designing, Coding & Delivering HTML Email

A beginners survival guide from



Petitioner has added romanette (i) to the first page and sequential pagination to pages 65-66. Otherwise, it leaves the original page numbering.

MICROSOFT CORP.
EXHIBIT 1023

(i)

Table of Contents

Introduction.....	3
About the Third Edition	3
But First: “What The Heck is MailChimp?”	5
How HTML Email Works	6
The Multipart/Alternative MIME Format.....	6
Image Files in HTML Email	6
Free Hosting Services	7
Delivering HTML Email.....	7
Designing and Coding.....	9
Tools of the Trade	9
Design Guidelines	10
Set your width to about 500-600 pixels.	11
Simple Layouts and Tables Work Best	12
Special Considerations for Webmail Services.....	13
CSS in HTML email	13
Flash, (and JavaScript, ActiveX, movies, and other stuff that won't work) in HTML email.....	15
Point emails to landing pages.....	15
Background Colors	15
Anatomy of a Good, Healthy HTML Email Newsletter	17
Plain-text Email	19
The Art of Plain-text email	19
Common Mistakes to Avoid	22
Common Mistakes to Avoid	23
Designing around spam filters.....	27
How anti-spam systems work	27
Bayesian Filtering	27
Blackhole Lists	28
Email Firewalls	28
Challenge/Response Filters.	29
Tactics for Avoiding Spam Filters.....	30
Testing & Troubleshooting Your Email Designs	32
Web Designers: emails aren't the same as web pages!	32

Testing in Different Email Applications	32
AOL	33
Testing Browser-based Email Services	36
Yahoo!Mail	36
Yahoo!Mail Beta.....	36
Hotmail.....	36
Gmail	36
Check Different ISPs, too	37
Send tests to friends and co-workers	37
Or, just use MailChimp's Inbox Inspector	37
Email Marketing Basics & Best Practices	41
The Definition of Spam.....	41
The CAN-SPAM Act of 2003	42
Double Opt-in.....	42
Pros & Cons	47
Measuring Performance	48
Open Rates	48
Click Rates	49
Unsubscribe Rate	49
Bouncebacks	50
MailChimp Helps Manage Your Bounces	50
Traffic to your website	51
Signups since last campaign.....	51
Wrapping Up.....	52
Wrapping Up.....	53
“My First Email Campaign” Checklist for Beginners.....	53
Appendix.....	57
Useful Resources	58
Email Marketing Sites	58
Email & Marketing Related Blogs.....	58
Products & Tools	59
Graphic design programs	59
Miscellaneous	59
Email Applications & Known Issues	61

Introduction

You don't have to be a professional web designer to use this guide, but a little HTML knowledge will definitely help. First, we'll cover all the **basics**, like how HTML emails work (and why it always seems to break when you try to send it yourself). Then, we'll get into **technical stuff**, like how to design and code them. Finally, we'll run through email marketing **best practices**, such as for list management, deliverability and measuring performance.

About the Third Edition

According to our server logs, this guide has been downloaded over 100,000 times since we first wrote it in 2001. Hopefully, it's helped a few people out there get their email marketing off to a good start.

We wrote the first edition because when we started MailChimp, you had to know a lot of technical stuff before you could get into email marketing. Back then, an email marketer was a jack of all trades, designing their emails, coding them into HTML, setting tracking links and images, managing bounces, and setting up databases on their websites.

Nowadays, email marketing services (like MailChimp) have launched tools that do all that mundane stuff for you. We manage your lists, give you signup forms, send welcome emails, handle double-opt-ins, set one-click unsubscribe links, and we even give you built-in HTML email templates (so you don't even have to know how to code HTML email anymore). All you have to do now is come up with good content, and click send.

But there are new challenges, which we'll cover in this 3rd edition:

- Most email programs now block all images in your HTML email by default. You've probably noticed how you have to right-click on an email, or push some button, to make images show display in the emails you receive. How should you design around that?
- Desktop email programs (like Outlook, Lotus, Eudora, and Apple Mail) have made great progress in standardizing the way they display HTML email. But now webmail services (like Yahoo, Gmail, and Hotmail) are getting more popular, and they have their own unique challenges. Two steps forward, one step back. What works and what doesn't in webmail clients?

- Spam filters are very sensitive to the content in your email. How should you write your content?
- There's so much spam out there, spam filters need their own spam filters now. They're called firewalls, and they block email before they even get to your recipients' spam filters, based on reputation. How do you protect your reputation?
- Spam laws require that every email marketer follow some very important rules now. If you break any of the rules, you can get your pants sued off. What do you need to include in every single email you send to your customers?
- Blacklists used to simply block email based on what server you sent from. Now, they scan the content of your messages, and they look for domain names that have been found in reported spam. Even if you've never sent an email campaign before, you can find yourself on a blacklist, if one of your reseller partners has been sending spam with your domain name in it.

But First: “What The Heck is MailChimp?”

[MailChimp](#) is a product of [The Rocket Science Group](#), a web development company in Atlanta, Georgia. Our one, driving motivation is to “build tools that are Useful, Simple, Efficient, and that people *love* to use.” **U.S.E.** is the formula and philosophy for everything we do (incidentally, it started out as s.u.e, but that sounded way too litigious).

Way back in 2001, we noticed some of our clients were trying to send out HTML email campaigns and monthly newsletters to their customer lists, but stuff kept breaking. They were rigging Microsoft Outlook “stationery”, or trying to send ginormous PDF attachments. Images were missing, tables were broken, formatting was off. Their customers were complaining. They were exceeding their ISP bandwidth quotas. Bounces were out of control. Sheesh! It was a *mess*.

They could handle *design* just fine. Even a little HTML coding. But it was the *delivery* part that they needed help with. See, to send HTML email *properly*, you have to set up a server to deliver in “Multipart-Alternative MIME format,” then track bounces with “VERPs” and other geeky stuff like that (we’ll go over that soon).

So we built a simple little tool where they could log in, copy paste their customer list, copy paste their email code, and hit “send.” And since *they* took care of the *creative* stuff, and we just handled the *boring* tasks that “only a helper chimp should do,” we called it “MailChimp.” We’ve since added some “email design” tools to MailChimp, to help people who don’t know one bit of HTML. Now, you don’t have to be a web designer (or even have access to your own server) to use MailChimp.

Thousands of customers from all over the world have discovered the simplicity of MailChimp. Maybe if you find this guide useful, you’ll give MailChimp a try, too (trust us, you’ll love it).

Learn more at: <http://www.mailchimp.com/>

How HTML Email Works

Before you can start designing, coding, and sending HTML emails, you should know how it works and what tools you'll need. Here's some background information every email designer and marketer should know...

The Multipart/Alternative MIME Format

The most important thing you need to know about HTML email is that you **can't just attach** an HTML file and a bunch of images to a message and hit "send." Most of the time, your recipients' email applications will break all the paths to your image files (because they'll move your images into temporary folders on your hard drive). And you can't just paste all your code into your email application, either. Most email apps send messages in "plain-text" format by default, so the HTML won't render. Your recipients would just see all that raw source code, instead of the pretty email it's supposed to render.

You need to send HTML email from your server in "**Multipart-Alternative MIME format**." Basically, that means your mail transfer agent bundles your HTML *code*, PLUS a plain-text version of the message, **together into one email**. That way, if a recipient can't view your beautiful HTML email, the good-old-fashioned plain-text version of your message is auto-magically displayed. It's kind of a nerdy gobbledy-geek thing, which is why a lot of people mess things up when they try to send HTML email themselves. You either need to program a script to send email in multipart/alternative MIME format, or just use an outside vendor (ahem, like [MailChimp](http://www.MailChimp.com)) to deliver things for you.

Image Files in HTML Email

Embedding images and photos into messages is the number one reason people want to send HTML email. The proper way to handle images in HTML email is to *host them on a web server*, then "*pull them in*" to your HTML email, using "absolute paths" in your code. Basically, you can't send the graphics along with your message. You host the graphics on a web server, and then the code in your HTML email downloads them whenever the message is opened.

Incidentally, this is how "open tracking" works. You place a tiny, invisible graphic into the email, and then track when it's downloaded. This is why open tracking only works in

HTML email, not plain-text, and why the new email applications that block images by default (to protect your privacy) can screw up your open rate stats.

When coding Image Tags in HTML email...

Do this:

```

```

Instead of this:

```

```

Note: If you just use MailChimp's built-in email designer, we host your graphics on our server for you (free). No need to FTP files anywhere, or code anything.

Free Hosting Services

When it comes to hosting the images for your HTML email, you really need your own server to do it. Don't try hosting images on a free "image hosting service," because those websites often put scripts in place to prevent you from linking to them in emails (they can't handle all the traffic). And since you really do get what you pay for, free image hosting services tend to be pretty unreliable under heavy traffic conditions. Also, spammers use free image hosting services all the time, to "cover their tracks." If you don't want to look like a spammer, use your own web server. If you use an email marketing service (like MailChimp) they usually come with a newsletter builder tool with image hosting capabilities built-in.

Delivering HTML Email

Many newbies make the mistake of setting up forwarding lists, or "CC'ing" copies of a message to all their customers. This causes all sorts of problems, like when a customer hits, "reply to all." Plus, there's no way to do any kind of individual tracking or personalization when they CC: a big group like that. Finally, it just looks so unprofessional and impersonal when recipients can see your entire list of other recipients like that. Blech.

That's why when an email marketing system (like MailChimp) sends your campaign, we take your message and send it one at a time to each recipient on your list (really, really fast). Unlike your work computer linked to your local ISP (which probably has a standard monthly bandwidth limit), email marketing vendors like us use dedicated mail servers that are capable of sending hundreds of thousands of emails (even millions, for larger vendors) per hour.

Other factors you should consider when delivering HTML email campaigns:

- If you send from your own server, your ISP may throttle your outgoing emails or shut your account down, if you send too much too fast (like more than 500 per hour). They may also shut you down if you exceed your monthly bandwidth limit. If you've ever sent an email campaign to a big list of recipients, and you attached a giant PDF, this has probably happened to you before.
- Email firewalls and ISPs who receive your emails don't like receiving tons of emails from one single IP address at once (unless they *normally* receive tons of emails from that IP). So if you only send occasional email campaigns from your IP, you may want to throttle your delivery, or spread them across multiple IPs, to avoid accidental blocking. Email marketing services usually split your campaign up into pieces and send it out over lots of different IP addresses.
- If you send emails from your desktop email program, chances are you're connecting through your local ISP. If you don't have a dedicated IP address setup with your ISP, you're probably sending emails from a dynamic IP address. ISPs and spam filters don't like receiving lots of emails from a dynamic IP address, because it looks like a hijacked home computer. You should always send from a dedicated IP address, or use an email marketing vendor (like MailChimp).

Designing and Coding

Now that you have a basic understanding of how HTML email campaigns work, and how you should deliver them, we can start talking about designing and coding them (the fun part!).

Tools of the Trade

If you want to get good at coding HTML email, here are the tools you'll need...

1. **A design application**, like Adobe Photoshop, Fireworks, Dreamweaver, or Illustrator. Use these tools to layout your template, and slice out your graphics (like your company logo, and product photos). We're really partial to Fireworks, but to each his own.
2. **A good, "pure" HTML/Text editor**, like BBEdit and TextWrangler for the Mac, or HomeSite, and NoteTab Pro for the PC. You *could* use "WYSIWYG" style tools to generate HTML, like Microsoft FrontPage, or maybe Adobe GoLive or Macromedia Dreamweaver. But they have their disadvantages. WYSIWYGs very often throw in gobs of junk into your code (FYI, Microsoft says the extra code is to help with "round tripping" which means you can export from Word to Powerpoint to HTML, then back again, without losing anything). All this extra code usually ends up just breaking things, or setting off spam filters. There are some WYSIWYGs out there that generate "clean code" but we've found those are a bit "*too perfect*," because they're designed for web pages, not HTML email (so they don't know all the things you have to "rig" in your code.). If you want to do HTML email *correctly*, you should learn to code HTML "from scratch" with a good text editor.

Free HTML Email Templates:

If all this talk of designing and coding HTML is too nerdy for you, consider using MailChimp's built-in HTML email templates. They're super flexible, and can be customized to match any brand. Plus, they've already been tested in all the major email programs. See some nice examples of what our customers have done:

http://www.mailchimp.com/screenshots/template_examples/

and watch a demo movie of our template design tool:

3. **Your own web server**, to host all your images and archives. You should create a folder on your website for email newsletters. This is where you'll store images, then point to them with absolute paths in your HTML code. You might also want a folder on your website to store archived newsletters. When we first started out, we created a folder on our website called, /monkeywrench/ and then kept campaigns under subfolders named by /date/ to store each campaign. Everybody has their own organization style, though. Just be sure to come up with a system and process for archiving, so that each newsletter isn't a huge chore to publish.

Nowadays, we don't even bother, because we use MailChimp's built-in archive functionality. It'll save a copy of every campaign you send from our system. You can build a page on your website called "Newsletter Archives" and paste a little code snippet from MailChimp. We'll keep your page updated with all your most recent campaigns. Saves so much time.

4. **An FTP program**, like Fetch for the Mac, or CuteFTP for PCs. When you design your email and slice out all your graphics, you'll use FTP to move those files from your computer to the server.
5. **A test machine (or two)**, loaded with as many different email applications and accounts as possible. You'll want to check your HTML email designs under lots of different scenarios. Unlike web design, where you only have to check your work in a small handful of browsers, there are *tons* of email applications out there, and they all handle HTML email differently. If you use MailChimp, you can use our Inbox Inspector tool and let us do all the testing for you:
<http://www.mailchimp.com/add-ons/inboxinspector/>

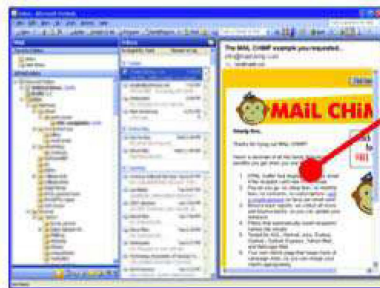
Design Guidelines

Here are some pointers for designing your email. Early warning: If you're a seasoned web designer, you're not going to be happy with all the compromises and hacks you'll have to make when designing for email applications.

Set your width to about 500-600 pixels.

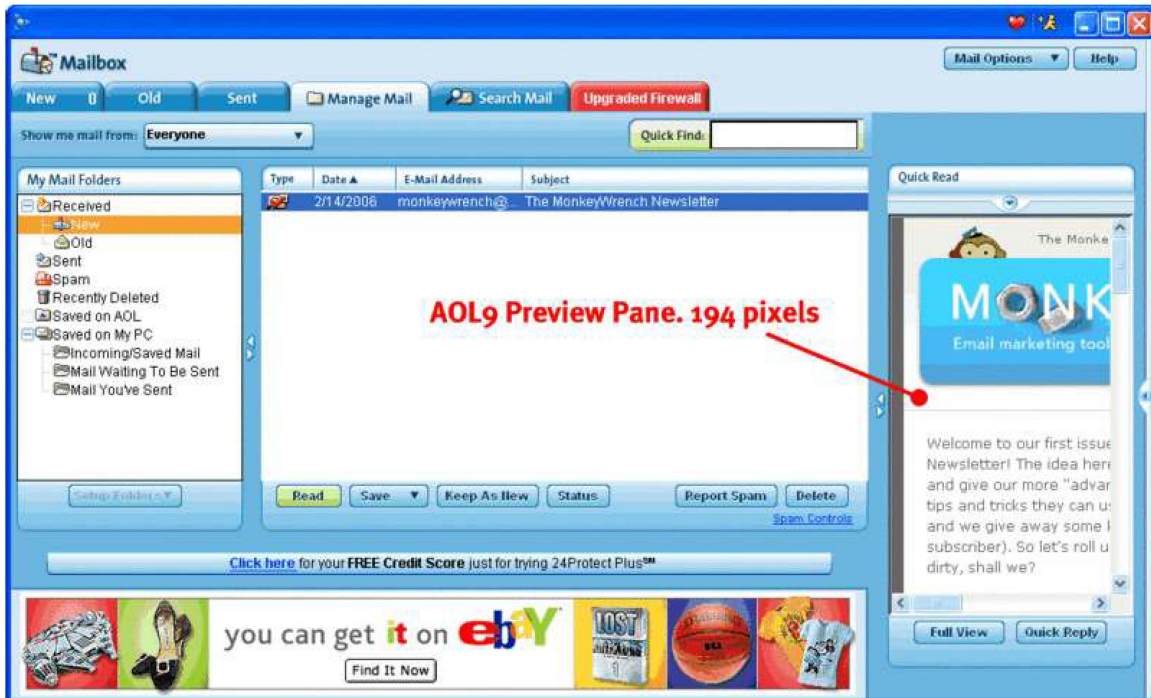
If you've ever worked on a website, you're probably used to designing pages to fit in 800x600 pixel resolution screens, or maybe even 1024x768. But that won't work when you design HTML email. Most recipients will be looking at your email through their "preview pane" which is usually a small portion of their available screen (see screenshot below).

Outlook 2003 on 1024x768 monitor



**Preview pane
is about 440
pixels wide on
this monitor.**

Above is a screenshot of an HTML email in Microsoft Outlook's preview pane. Outlook is maximized to full screen (1024x768 pixels), but as you can see, the preview pane is only about 440 pixels wide. In **AOL 9**, the "preview pane" is only about 194 pixels wide, then when you click "Full View," it typically opens up a window that's about **540 pixels** or so. This is why you shouldn't design emails to be much larger than **600 pixels** in width.



Simple Layouts and Tables Work Best

You wouldn't believe all the different ways email applications garble up HTML email. You'll need to keep your email designs nice and simple. Avoid complicated layouts, too many embedded tables, and tables with too many rows and columns. A simple two column table with a row across the top is about as complex as you can get. There are lots of email applications that don't cooperate when you code tables with colspans (table cells that "span" across multiple columns) in them. One application that's particularly finicky when it comes to tables (and just about everything *else*) is Lotus Notes. We can't stress it enough. You've got to keep your layouts very, very simple, if you want it to work across all the major email apps.

If you're working on an email layout and find the tables are getting complex, and you're having to code too many "COLSPANS," you might think about chopping it up into separate tables. For instance, we almost always use separate tables for the header, body, then footer of our emails. Better to go with more "simple" tables, than one big "complex" table.

If you're a CSS ninja, you're probably rolling your eyes in disdain with all this "TABLES" talk. But trust us --- you're not going to be able to use all that cool, timesaving CSS stuff

in your HTML emails. DIVs and CSS-positioning don't work. We'll cover that later in this guide.

Special Considerations for Webmail Services

A lot of your recipients will be checking their email in their **web browser**, such as with Yahoo!Mail, Hotmail, or Gmail. Since they're viewing email in browsers, there are certain things you need to know when you code your HTML emails:

- They'll strip out your <HTML>, <HEAD>, and <BODY> tags, in order to keep *your* code from interfering with *their* web pages. That means...
- Any background colors you specify in your <BODY> tag will be lost. So wrap your entire email **inside** a larger, 100% wide table, and set a background color to that table.
- Any CSS that you place inside the <HEAD> tags will be lost. So don't try to link to CSS files hosted on your server. Use embedded CSS instead, and make sure it's below the <BODY> tag. If you're a web designer, that'll definitely make you feel "kinda dirty" because that's not exactly the "proper" place to put embedded CSS. But no worries---it'll work. If you just can't stomach that, use inline CSS.
- Some webmail clients are starting to strip even the most basic CSS, even if it's embedded below the BODY tag (like we described above). We think it's because they don't want any of *your* CSS interfering with *their* CSS, or maybe because they're using so much JavaScript and AJAX technology, stuff just gets buggy. Keep that in mind, and do lots of testing
- **Tip:** you can use MailChimp's Inbox Inspector to get screenshots of your work in all the major email programs and webmail clients, to see if your CSS is breaking. Saves a lot of time and hassle.

<http://www.mailchimp.com/add-ons/inboxinspector/>

CSS in HTML email

CSS (Cascading Style Sheets) is a Godsend for websites, but don't put too much faith in it with HTML email. For instance, you shouldn't expect DIVs to work. And ***definitely*** stay away from CSS *positioning*. Won't work. If you're a CSS-standards freak, you are going to hate yourself after coding HTML email.

You'll have to rely on old-fashioned <TABLES> for your layouts in HTML email, and only use CSS for simple font formatting and colors. Always design your CSS to "fail"

gracefully.” That means if someone took away your CSS, your design and content would still display decently. Before you send your HTML email, delete the CSS and see what it looks like. Did you have tiny text that is now gigantic, and blowing out your tables?

CSS is *especially* a touchy situation if you’re using a WYSIWYG to design your HTML email, because WYSIWYGs insert all kinds of crazy CSS and DIVs by default. It’s one of the drawbacks of WYSIWYGs for HTML email. It may help you get things coded, but you still have to understand HTML enough to go back and *remove* some of the code, so things won’t break in email applications. You might as well code by hand.

Using Microsoft Word to generate HTML is even worse than using WYSIWYGs. Word adds so much code behind the scenes, it’s unbelievable. Don’t do it. Learn to code HTML, or use our built-in HTML email designer.

If your email design uses lots of CSS, be sure to check it in Google’s Gmail. At the time of this writing, they don’t fully support CSS, so you’ll see your fonts defaulting to Arial, and black in color. Here’s what our [MonkeyWrench Newsletter](#) looked like in Outlook 2003 (which supports CSS just fine) vs. Gmail...



Gmail stripped out virtually all of my CSS! Where did my font sizes, font colors, and line-height go? And since the fonts are so much larger in Gmail, it could have blown out my table cells (if I didn't know what I was doing). But notice the top line (next to the peeking chimp). It wrapped my text into 2 rows. Blech!

Flash, (and JavaScript, ActiveX, movies, and other stuff that won't work) in HTML email.

You can *deliver* HTML emails with Flash in them, but most recipients won't be able to *view* them. Not unless they're using an email program that uses a browser to render their HTML email. Microsoft Outlook was a program that used Internet Explorer to render HTML email, so Flash would sometimes work there. But Microsoft Outlook 2007 is no longer going to use IE to render email. They're using Microsoft Word instead, so Flash is not going to work there anymore. Anyways, most **anti-virus applications** block the code used to embed Flash movies, and they block JavaScript, ActiveX, and even background music files. All that stuff has been used in the past to spread viruses, so it's commonly blocked now.

Point emails to landing pages

If you've got a great animation or movie to show to your recipients, just send a simple, intriguing GIF or JPEG graphic in your HTML email, then link it to a "landing page" on your website with the animation in it. Same goes for JavaScript, ActiveX, and movie files. Anti-virus applications block them from running. So fancy rollover or pop-up navigation and streaming videos just aren't going to work either. The only email application on our test machines that will actually play fancy stuff (like Flash and movies) seems to be Outlook 2000 (which is kind of old, these days). Unless you know for a fact that every single one of your recipients uses Outlook 2000, and you know their anti-virus applications won't block your stuff from running, don't send Flash, JavaScript, ActiveX, or movies in HTML email. Just won't work reliably.

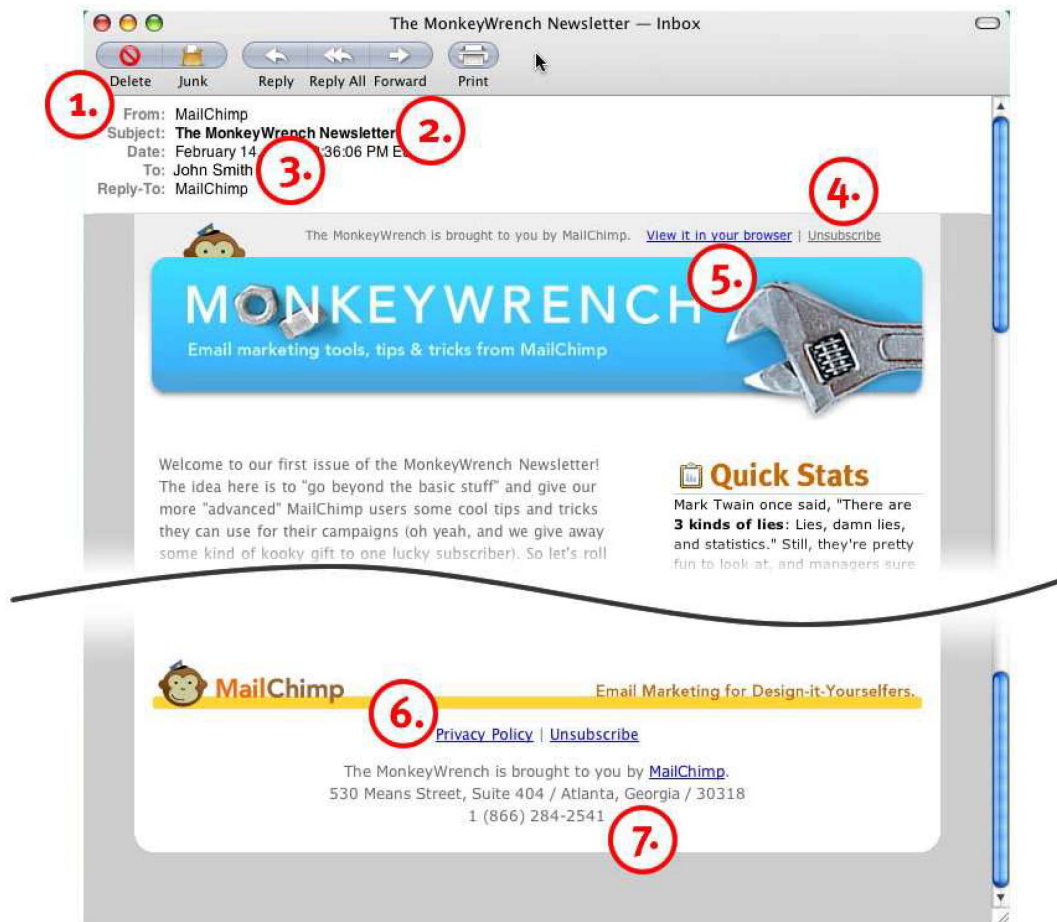
Background Colors

Remember when we told you (above) about how browser-based email services (like Yahoo!Mail and Hotmail) strip out your <HTML>, <HEAD>, and <BODY> tags? That's what you need to remember when coding your background colors and images. You

normally specify that stuff in your <BODY> tag, but you can't rely on that with HTML email. You'll need to create a big, 100% wide "table wrap" around your email. Set your background color in that table, and use some CELLPADDING as needed. If you're into CSS, you might be tempted to use a <DIV> instead. But we've seen DIVs break quite frequently in email applications, so we don't recommend them.

Anatomy of a Good, Healthy HTML Email Newsletter

Here's what a well-built HTML Email Newsletter looks like:



1. Your company name in the "From." Recipients should recognize who the email is from instantly. It can't be deceptive in any way (duh). If a recipient has to strain his brain to remember who you are, he'll click "this is spam" instead of opening.
2. A relevant subject line (don't be "spammy"), with your company or newsletter name in it. So they instantly know who the email is from, and what it's about (hence, "subject" line).
3. The "To:" field of your email should be personalized to the recipient's name, not their email address. MailChimp lets you "merge" recipient names into this field, if you have it in your database. You would insert *|FNAME|* *|LNAME|*, for

example. Search our help documents for “merge tags” if you want detailed instructions.

4. A one-click opt-out link that removes people from your list immediately. Consider placing it at the *top* of your email (as well as in the footer), so that people who want off your list can find it really easy (instead of clicking their “Junk” button). If you use MailChimp’s Managed List functionality, you would use our [*|UNSUB|*](#) tag to generate your unsubscribe link (it’s required in every email you send from our system). Any recipient who clicks it will be instantly removed from your list. If you use our built-in HTML email templates, we insert that link into the footer for you (no coding required).
5. In addition to your opt-out link, you might also include a link in your header for recipients to “view this email in your browser.” Point it to an archived version of your email on your server. This helps if the email got forwarded to friends, and then got mangled along the way. MailChimp automatically keeps an archived version of all your campaigns on our server. You can just use our [*|ARCHIVE|*](#) tag in your emails to link to yours. If you use our built-in HTML email templates, we automatically place this link at the top of all your emails.
6. A link to your company’s privacy policy should be in your email. If you don’t have a privacy policy, you can use the DMA’s privacy policy generator. Google the phrase, “DMA privacy policy generator”
7. Your valid, physical mailing address (P.O. Boxes aren’t good enough), and as much contact information as possible. The more contact information you provide, the more reputable your email will look.
8. Bonus: It’s a really good idea to also include some kind of reminder text, like “You are receiving this email because you signed up at our website.” People forget opting-in to lists, and they get a little trigger-happy with the “this is spam” button. That can get you reported to the major ISPs, so you want to prevent that as much as possible. Also, in the rare case that a recipient reports you to an anti-spam organization, having this reminder text can make the difference between a server admin blacklisting you forever, or contacting you for further explanation. If you use our built-in email templates, MailChimp inserts that reminder text into your email footer automatically.

Plain-text Email

Don't forget your plain-text email! You need it for people who can't (or won't) view HTML. Sometimes, the plain-text version of your email is the one that gets displayed if your recipient checks his email on his mobile device.

If you don't take the time to create the plain-text version of your HTML email, you'll just look like a lazy spammer (spam filters will penalize you for only sending HTML). When you send a campaign in MailChimp, you'll need to create both versions of your message (HTML & Plain-text). Then we "bundle" them together in "multipart format" and send them off. Your recipients' email applications then decide which version to display.

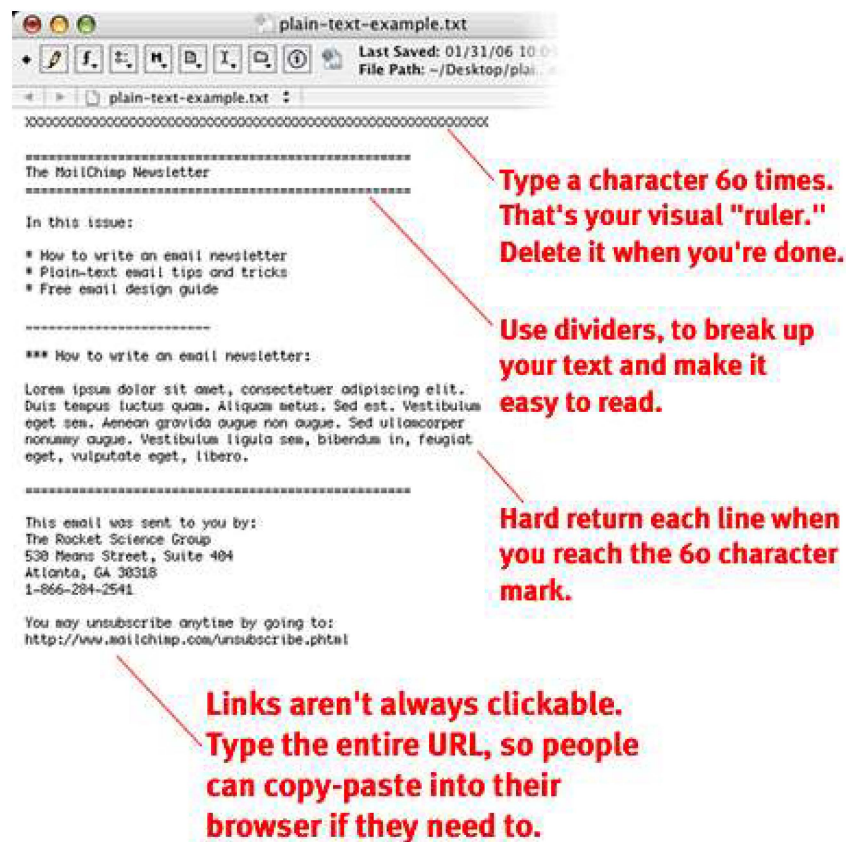
The Art of Plain-text email

There's definitely an art to constructing your plain-text email. You can't just use the same text from your HTML email. It has to be formatted *juuuust* right, so the best way to do it is to build it from scratch, or copy-paste the content from your HTML email, then reformat it heavily.

Here are some tips for your plain-text emails:

1. Use a plain text editor, like TextWrangler for the Mac, or Windows NotePad. Don't use something like Microsoft Word, because it won't give you the kind of "raw" text files you need. You need an application that gives no formatting whatsoever.
2. If you're using MailChimp, create a plain-text message in your text editor, then simply copy-paste it into the "plain text message" field in MailChimp. No uploading required. Or, you can just type straight into MailChimp's interface. One thing I like to do is pop-up preview my HTML email, highlight all my content, then paste it into the plain-text field in MailChimp. Then, I go and reformat everything.
3. By default, some email applications start to wrap your plain-text messages at about 60 characters or so. That's not really a big deal to most people. Just let your text wrap wherever their email programs want it to. But if you have content that you absolutely do not want to wrap unexpectedly (like in the middle of a hyperlink), one nice little trick is to type a letter 60 times across the top of your message, and use that as your visual "ruler." As your content approaches the end of that ruler, hit your return key and "hard wrap" the line. Again, this is only if

- you want very strict formatting. These days, I let my plain-text emails just wrap fluidly.
4. People don't read email. They *scan* email. Especially plain-text email. So make it really easy to find important information with bullet points and visual markers (see screenshot below)
 5. You can't code a clickable link with plain-text email. You just have to type out the entire URL. Some email applications will make it clickable automatically. Some won't.



One thing we've noticed is that a lot of people spend so much of their time on the HTML version of their message, they're too "drained" to work on their plain-text email (we're guilty of that ourselves). So they either ignore it, or they slap some junk in, or they "cheat" a little, and type in something like, "Evidently, you can't view HTML email, so visit this URL in your browser to see our newsletter in all its glory." We think this is a big mistake. So don't put *all* your energy into the HTML version of your email. Save some love for your plain-text message too. If you find yourself spending so much time coding

HTML that you're too pooped to make a plain-text email, you should consider setting up one re-usable template that you can just select, and "fill with content" every campaign. No need to reinvent the wheel for every campaign.

MailChimp Trick:

When you setup a managed list in MailChimp, you have the option to "let my users choose plain-text only" when they fill in your opt-in form. MailChimp will automatically deliver **only** the plain-text formatted email to them when you send your campaigns.



MonkeyBrains!

MailChimp's MonkeyBrains blog is chock full of HTML email design examples, tips, tricks, and hacks, plus email marketing news and best practices.

<http://mailchimp.blogs.com>

Common Mistakes to Avoid

Here are some of the most common mistakes people make when coding HTML emails:

1. Not coding “absolute paths” to their images. Remember, attaching graphics and using “relative paths” won’t work. You need to host the images on your server, then link to them in your code.
2. Using JavaScript, or ActiveX, or embedding movies. That stuff just doesn’t work in HTML email.
3. Getting over ambitious with designs. Designing HTML email isn’t the same as designing web pages. You can’t have all the CSS-positioning, DIVs, DHTML, and complex, embedded tables like you can in web pages.
4. Forgetting---or refusing---to include an opt-out link. It’s stupid and unprofessional not to allow recipients to unsubscribe from your list. Oh yeah, it’s illegal, too. Be sure to read and understand [The CAN-SPAM Act of 2003](#).
5. Linking to an external CSS file. You usually put CSS code in between the <HEAD></HEAD> tags of web pages, right? But browser based email applications (like Yahoo, Hotmail, and Gmail) strip <HEAD> and <BODY> tags from HTML email. Embed your CSS below the <BODY> tag.
6. Letting your permission “grow cold.” If you’ve been collecting opt-ins at your website like a good email marketer, but you haven’t sent an email in several months, your subscribers have forgotten all about you. So when they receive a full blown email newsletter from you “out of the blue” they’re going to click their “this is spam button” in their email application (related blog post: [Surviving the inbox whack-a-mole game](#)). Don’t let your permission grow cold. Send occasional emails. If you’ve been collecting emails for years, and you’re only *just now* sending your first email to everyone, you should send a little “re-invitation” to your list. Something like, “A really long time ago, you subscribed to my email list. Well, I’ve finally got some time to start sending my really cool newsletter, and I want to make sure you still want to receive it. Click here to sign up...” At the very least, send an email and place some kind of text at the top that reminds them of where and when they opted in. People usually forget about opting-in to something after about 6 months or so.
7. Sending to a list without permission. This is the worst offense. Lots of “innocent” marketers, who “mean well” commit this heinous crime. Here are some common ways legitimate marketers can inadvertently become known as “evil spammers:”



- Getting an email list from a tradeshow. “But I’m exhibiting there, and the tradeshow host said it would be okay, and when people purchased tickets, the fine print said that we could email them, and...” Nope. You’re spamming. Don’t do it. If they didn’t give you permission to email them, they didn’t opt-in to your list. If they didn’t opt-in, you’re spamming. Even if you can legally send them email marketing, those recipients are more likely to report you for spamming them. Then, you’ll get blacklisted. Plus, your company will look really slimy. Don’t do it. If a tradeshow host is collecting email addresses, then they should be doing the emailing. It’s all about permission, and setting expectations.
 - Getting a list of “fellow members” from some trade organization. Just because they joined a club, and the club posted contact information so everyone could keep in touch with each other, it doesn’t mean each member gives you permission to send them newsletters and offers. If you sent them a personal greeting from your own email account, they probably wouldn’t mind. That’s what the organization’s “members list” is for. But add them to a list and send them a huge email newsletter, and you’ll be reported for spamming. Don’t do it.
 - They go to events, and swap business cards. The business cards just get thrown into a “prospects” pile. Years later, they get an intern to finally type all the contact info from those cards into a database. Then, one day out of the blue, they send a big, fat email newsletter to everyone. As if they actually want to hear from you! Trust us, they don’t.
 - Fish bowl of business cards. Similar to the example above, but they hold an event, and collect business cards in a fishbowl to win a “door prize.” The people who dropped their business cards into the bowl wanted the coffee t-shirt you’re giving away (plus all the fame & glory). They didn’t opt-in to your list.
8. Using a WYSIWYG to “code” your HTML. WYSIWYGs are notorious for generating absolutely horrible HTML. They insert so much junk code, it’s unbelievable. Even the ones that generate “clean” code don’t know how to “rig” things to work in email applications (like sticking your embedded CSS *below* the <BODY> tag). To code HTML email properly, you need to learn a little HTML. It’s really not that hard. Plus, you can download free HTML email templates at our website: <http://www.mailchimp.com/resources/templates/>

9. Forgetting to test. Thoroughly. When you send HTML email, you've really got to test it in as many email applications you can. Then you have to test on different operating systems. Then different ISPs (we'll explain shortly). If you keep things simple, and build a rock solid, thoroughly tested **template** for each newsletter, you won't have to test so much. But you should *always* send at least a few campaigns to yourself before sending it out to your entire list.
10. Sending nothing but a big, gigantic graphic as their HTML email. Sigh, that's what spammers do. And since most email programs block images by default, what do you think your recipients see when (if) they open your email? Many spam filters will block your email if you don't have a healthy balance of images *and* text.



The MonkeyWrench Newsletter

**Subscribe to MailChimp's MonkeyWrench Email Newsletter.
We'll keep you up-to-date on more "advanced" email coding
tricks, obscure hacks, power tips from MailChimp
programmers, and oh yeah---free prizes.**

<http://www.mailchimp.com/monkeywrench/>

Designing around spam filters

A guide on email newsletter design wouldn't be complete without a section on spam filters. You have to design your emails a certain way, if you want your messages to make it past them. But you can't go overboard, either. Spam filters can smell fear. If you try too hard, they know it, and they attack. So just be cool, act normal, and don't make any obvious mistakes.

How anti-spam systems work

Before getting into the nitty-gritty design tips, you need to know how all the various anti-spam mechanisms work out there. Once you get an overall understanding of how they work, designing around them will be much easier.

Bayesian Filtering

This is one of the most important ones to learn about, since it's installed in so many email applications these days (like Outlook, Mozilla Thunderbird, and Apple Mail).

Bayesian filters work by watching users classify email as "junk" (such as when they click a "this is spam" button). It reads the junk mail, compares it to other emails you called "junk," and looks for common traits in the subject line, the content, the hyperlinks, the



sender, etc. Over time, Bayesian filters learn to scan for those "traits" in every email message you receive. Every time they find something that looks "spammy," they assign a "score" to it. For instance, using "Click here!" might get you 0.7 points. Using **bright red fonts** might get you 2 points. Including the word, "mortgage" might get you 1 point, but using the word, "Viagra" will get you 5 points. Once an email exceeds some threshold (set by the user), the email is classified as spam, and thrown into the junk folder. It's amazing how many different things Bayesian filters look for. To read some, [go to the MailChimp Blog and search for "Funny Spam Assassin Criteria"](#)

MailChimp also comes with a spam filter checker (look for the "Inbox Inspector" under our Account tab). It'll tell you exactly what your email's spam score is, and what you need to change if you want to improve your chances with spam filters.

Black Lists

A while back, some server admins got really, really, angry at all the junk mail they received. So they started to track the IP addresses of the servers that sent them spam, and put them on “blacklists.” Anytime they received email from a server on their blacklist, it was deleted immediately. It worked pretty well. So they started to share their lists with other server admins. And other admins started to add on to those blacklists. Pretty soon, the blacklists got very, very big. Large ISPs started to sync up with them. If you send email that’s very “spammy” you could end up on one of these blacklists (whether the recipients are opt-in or not). Once you make it on to a blacklist, good luck getting off. Geeks are notoriously difficult to negotiate with (we know, because we’re geeks). This is why you need to make sure you only send to recipients who gave you verifiable permission to email them, and you need to have proof that each one of them opted in to your list (such as through a double opt-in system). Want to know the easiest way to get blacklisted? Let your company’s sales team blast out an email newsletter to a list of “prospects” that they collected from conferences, and that they scraped off of websites. You’ll get blacklisted, guaranteed.

Email Firewalls

Managing email servers (and incoming spam, viruses, and phishing attacks) can be a lot of work. So large corporations usually install “email firewalls” (Google the terms, [“Barracuda Firewall”](#) or [“Postini”](#) for some examples) to handle their incoming email.

Think of them as “spam filters on top of spam filters.” They’re big, heavy-duty gatekeepers, and they’re not friendly at all.

They often use a combination of Bayesian-style/adaptive filters, community reporting, blackhole lists, and a little bit of proprietary “magic pixie dust” to keep spam out of the company. Most of the time, when your email’s not getting through to a larger company, it’s their firewall. You can think of these firewalls as kind of Xenophobic and paranoid. They’re all twitchy, and tend to ask questions like, *“Okay, is this sender new to me? Why is he sending copies of the same, exact email, to a bunch of people in our company? Spammers do that kinda stuff. Hmm, how long has their server been around? Can I really trust this sender?”* Spam firewalls are usually only a problem when you first start sending campaigns to a big client or something. You’ll experience some deliverability issues in the beginning, because you’re “new.” They’ll eventually “learn” to let you

through. To expedite things, you may have to ask the IT people in charge of the firewall to “white list” your IP Addresses (or the IP address of your email service provider).

If you’re interested in learning more about email firewalls, search for “email firewalls” at the MailChimp blog:

<http://mailchimp.blogs.com>

Challenge/Response Filters.

These are more common among “at-home” recipients (because they’re too intrusive to use at work). When you send email to someone with a challenge/response filter, here’s what happens. If you are not already in that person’s “buddy list” or “address book,” then you’re considered a stranger to him. And if you’re a stranger, you *could* be a spammer. So their challenge/response filter sends you an automatic reply with a question that you have to answer, or some link you have to click (this is to prove you’re a human, and not a spambot). The screenshot you see to the right is an example of a typical challenge/response reply, from Spamarrest, a very popular vendor.



The thing to remember here is that you have to be “white listed” if you want your emails to get through. So when people fill out your opt-in forms on your website, ask them to “please add our email address to your address book...” Use your opt-in process as a way to “[set expectations](#)” and get “whitelisted” up front. Whenever you send a newsletter, make sure the “reply-to” address that you use is valid, and that a human checks it after each campaign. You can expect to receive a couple auto-replies like this after every campaign you send. Whenever we send our MailChimp Monkeywrench newsletter, we get about half a dozen of these autoreplies. I have to manually click each one of them if I want the email to get delivered.

Tactics for Avoiding Spam Filters

So now you know how a lot of anti-spam systems work. You really have to “think like a spam filter” when you design your emails. Just like you “think like a search engine” when you design web pages. Spam filters read your emails, and look for similarities with known spam. You don’t want to do things that’ll get you accidentally thrown into the junk folder.

And the best way to learn what ***not*** to do is:

1. Open up your junk email folder.
2. Look through all that spam you got. *Really* read it. Look at how they design things, and look at how they write their copy. Notice all the similarities?
3. Now, don’t do what they do.

Okay, okay, to be more specific, here’s some stuff to avoid:

- Using **bright red fonts**, or going crazy with colors, styles, and formatting. Check out the article, “[Let’s dissect some spam!](#)” at our MonkeyBrains blog to see what I mean.
- Yelling with lots of exclamation points!!!!!!
- YELLING WITH ALL CAPS!!!!
- Using spammy words, like “mortgage, viagra, etc.”
- Saying, “free” or “click here!” or “click here now!” or “act now!” or “limited time!” (be especially careful of your unsubscribe link, where you might say something like, “click here to unsubscribe.”)
- If you use a WYSIWYG to code your HTML email, make sure the HTML code is “clean.” WYSIWYGs often throw in lots of useless code that’s invisible to the user, but spam filters look at it, and think you’re a sloppy spammer who doesn’t know HTML.
- Bad HTML in general can get you spam filtered. Missing table tags, content below the closing </HTML> tag, or empty <TITLE></TITLE> tags will get your message thrown into the junk folder. Don’t get sloppy with that code! Note: the empty <TITLE> information is very common if you use WYSIWYGs. They throw that in by default, and expect that you go back and fill it in. Don’t forget to go back and give the document a title!
- Don’t use “dummy text” in your message, even if you’re sending a test campaign. Spam filters get suspicious of all that “lorem ipsum” stuff. When sending test campaigns, use content that’s as real as possible. Don’t type in one

paragraph, then copy-paste it ten times. Spam filters can see all the duplicate content, and they think it's spam.

- Don't use the word "test" in your subject line.
- Don't get too creative with your spelling, because the spammers do that all the time (how many m0rtg4g3 ref1nanc3 spam messages are in your inbox right now?).
- Don't make an HTML email that's just a bunch of pretty graphics. You need some text in your message, too. Otherwise, the spam filters will have nothing to read, and will think your message is junk.
- Don't send the HTML email by itself. Always include that plain-text alternative message.

Learn More:

Read this article to learn more about how spam filters think:

http://www.mailchimp.com/resources/how_spam_filters_think.phtml

Testing & Troubleshooting Your Email Designs

After you've finally figured out how to design and code your first HTML email template (and, of course, your plain-text alternative version as well), don't just plop in your content and start sending campaigns to your list. You need to test your template first. Make sure it'll work in all the different email applications out there. Once you've found all the little bugs, and your template is "rock solid," you can start sending.

Web Designers: emails aren't the same as web pages!

If you're a web designer, you're probably used to testing web pages in a few different browsers, like Internet Explorer, Mozilla Firefox, and Mac's Safari. And you're probably familiar with a few annoying inconsistencies between all the browsers, and you have a couple hacks to make things look right.

Multiply all that by ten, for email design. There are tons of email applications out there that you need to test on, and they all render HTML email in their own annoying ways. Do you need to test every single newsletter across every single application, every single time you send? Of course not. Just setup a good, simple, rock solid template and test it as much as possible. Then all you do is swap out content. Here are some tips for testing your email designs...

Testing in Different Email Applications

In the past, the only way to test your campaigns was to setup a bunch of accounts at all the major ISPs, setup some test computers with different operating systems and email programs, and send test campaigns to them. It would take hours to do all this, and it really took all the fun out of email marketing.

That's why we created our Inbox Inspector tool. You build your campaign, and click one button. We'll generate screenshots of how your campaign will look in every major email program, and we'll tell you if any of the major spam filters and email firewalls will block your message, and why. We'll point out what content you need to fix in order to not get blocked. We'll even check for typos and HTML coding errors. Learn more about the Inbox Inspector at:

<http://www.mailchimp.com/add-ons/inboxinspector/>