How can we build professional content-rich emails achieving user agent compatibility?

Author:	Daniel Șerbănescu
Class:	Computer Science 115Q
Institution:	Zealand Institute of Business and Technology
Supervisor:	Vibeke Sandau
Research span:	25.04.2013 – 06.06.2013
Digital version:	data.serbanescu.dk/papers/email-HTML.zip

Table of contents

0. Abstract

Within this paper I am trying to find solutions to one of the biggest issues found in our standard-compliant digital world. Namely, the Email HTML rendering related to email client compatibility.

In order to find solutions, all of the technologies around the emails are explored. The focus is set on the specific aspect of those technologies that have an impact on how each email lands on users' screens. Among the technologies explored are HTML, MIME, CSS and VML.

Continuing the search for answers, the emails delivered by a few enterprises are passed through a series of acid tests[1]. The outcome of the tests is then analyzed and reviewed.

Based on the results of the first phase a new research is made. This time the focus is on the issues. The main aspects analyzed are the source of the issues while at the end potential solutions are revealed.

Summing the practices and guidelines would result in the artifacts of this paper. Within the artifacts would be presented a series of recipes that would help every newcomer to the field to achieve good email compatibility while respecting each email user agent.

At the end, a conclusion is drawn from the total work involved in the paper.

The conclusion summarizes that it is possible to create professional good-looking emails and achieve user agent compatibility at the same time.

1 Acid test – a test originally designed for testing a web page in different browsers in order to find compatibility problems; within the email context an email is tested in different email clients

1. Introduction

In December 2012 I was hired as a developer within the Media Department of the Seventh Day Adventist Church Denmark (referred to as SDA). The Media Department handles diverse media such as magazines, newspapers, online news, websites and broadcasting video shows over the Internet.

During my internship period, their primary goal was to move all of their websites' functionalities from their current CMS[1] system, Dynamicweb[2], to another CMS system, Joomla[3]. Since I had previous experience with Joomla and I knew its structure, they found me helpful in the integration and development area.

About the Company

SDA is an organization that focuses on managing churches across the globe. Furthermore, SDA has many other branches like scout troops, teen competitions, private schools, family counseling, humanitarian activities, media publishing, video broadcasting, and many many more.

I was an intern within the Media Department of the danish agency of SDA, where the audio-video materials, graphic content and web pages are handled. The Media Department maintains websites graphically, technically and administratively as well. They assure that the multimedia content is delivered to the consumers in the best possible way.

1.1. Problem Introduction

During the internship period I had been assigned to build an extension for an email management system, named Acyba Mailing. The extension needed to import text from a number of text files then interpret the contents and place relevant pieces of text into an email template. My assignment included building an email template as well. The generated emails needed to be appealing to the readers, so the template had to contain graphic elements and nicely marked-up text which would be rendered[4] by the readers' email clients.

As a result, every email could be generated automatically by the Acyba system because of the extension I have built. During the testing phase I noticed that the generated emails were not rendered properly in all email clients and therefore readers could sometimes see garbled content. In the next following days I worked hard to improve the template's code structure in order to achieve compatibility with all the email clients. In the end I managed to make the template compatible with a wider range of email clients, but in some circumstances there were still a small number of inconsistencies in the way the content rendered within different email agents.

I had briefly investigated some documents targeting the issues around content-rich emails

1 CMS – Content Management System – computer program that allows inserting, publishing, editing or modifying content from a central interface
2 Dynamicweb - is a All-in-One online proprietary CMS which can be tightly connected with one or more webshop, eCommerce, online marketing and integration solutions.
3 Joomla – is an Open Source CMS platform that provides free content management practices
4 Rendering – the act of interpreting/translating from the source code to the end-result visible by users

and I soon discovered that this is a very inexact science which makes one of the biggest issues among the content delivered on the Internet. The issues encountered intrigued me and got my interest to investigate the source of the problem at a deeper level.

1.1.1. What Led to The Problem?

SDA had a system that delivered daily emails. The system was able to send batch emails. The email contents were made and inserted manually one by one into the system. Then the system duplicated the emails in order to send a copy to each subscriber. During the time SDA noticed that this was a very time-consuming practice, and a lot of time was used for generating the emails from the raw content they had into nice formatted emails.

I was assigned to find out a better solution that could generate, manage and send the emails automatically. They also wanted to introduce graphic content into their emails. This would make their content more appealing and attractive to the readers.

During my research I found a system that could do just about all of the above mentioned tasks. This system is Acyba Mailing which is an important extension to Joomla (the engine that manages SDAs websites). The only missing link was a function that could read and interpret the raw content SDA had, and generate emails out of it.

I begun to build that specific function in order to integrate the Acyba system with the SDAs webpage.

By the time the extension was finished and some newsletter templates were built, I noticed during testing phase, that there were compatibility problems with the way some email clients interpreted the contents of the emails.

1.2. Problem Context

The context of this problem is mainly within email newsletter management systems. This kind of systems are often installed on the same server that also acts as email server. In the building process of the email design and content there are usually more persons involved. These persons are editors, graphicians, designers and web developers or programmers. When designing an email several factors should be kept in mind. Among these factors are quantity, size, type of content and flexibility of delivered emails, and some external factors such as the targeted reader groups or the load performance of the email servers.

1.3. Domain Introduction

The domain that covers this area is mainly **Email HTML**.

Email HTML is a term that describes a variation of the HTML[1] standard which is interpreted by email clients in order to render content-rich emails. It is improper to call it just HTML because its implementation is poorly developed across the email clients. That means that some of the email clients are capable of rendering the full range of HTML tags while others are limited to just a few tags.

1 HTML – HyperText Markup Language – the main markup language for creating web pages

As a term Email HTML is a misnomer. The HTML itself is not that troublesome as the CSS style properties used to define the fine markup of the HTML elements. Therefore when a web developer hears about Email HTML the first thoughts that comes to mind are the style problems and the difficulty of achieving cross-client compatibility.

1.4. Special Terms

Emailology – while this term is not present in a standard Oxford English Dictionary, it is used by the company "Email on Acid" to describe the science of building good-looking content-rich emails that render well in the majority of the email clients.

HTML – Hyper Text Markup Language – a language that describes the structure of an Internet page, the language is interpreted by web browsers and rendered accordingly

Email HTML – a variation of the HTML standard, used by email clients to render content-rich emails

CSS – Cascade Style Sheets is a standard defining advanced styles for HTML elements, the language is used to fine-tune and easily manage the design of multiple web pages

in-line CSS – CSS code used "in-line" within an HTML document, assigned to each element that requires styling

VML – Vector Modeling Language – programming language developed my Microsoft, aimed at building 2D and 3D geometrical shapes

Mail User Agent (MUA) – email client – software or service with capabilities of displaying an email

MIME – Multipurpose Internet Mail Extensions – an IETF[1] standard that makes creating complex email messages possible

1 IETF – Internet Engineering Task Force – organization that develops and maintains the Internet standards

2. Problem Formulation

The main problem discussed within this paper is:

How can we build professional content-rich[1] emails achieving user agent[2] compatibility?

Subproblems:

Which types of content is safely displayed across email agents?

How do enterprises handle this issue?

What practices would improve email compatibility between email clients?

Is it better to attach graphic components to emails instead of calling them remotely?

What are the main aspects one should focus on when developing email content?

1 Content-rich – the term refers at an assembley of well formatted text, pictures and other forms of design elements

2 Mail User Agent (MUA) – a program or a service that allows the user to send and receive emails

3. Methodology

The methodologies used in this paper will focus on the email HTML technologies rather than the tools used to build the resulting products of the research or the products themselves. Therefore the tools used will be just briefly described.

First I would present the environment were the issues occur. I will briefly present the technologies that cooperate in order to deliver the email content from content providers to customers.

Then, I will begin my study researching the basic structure of the email as it was developed over the years and I will mark the relevant theoretical knowledge gathered along the way.

I will continue my study with researching the source code of emails used by a wide range of other companies including online stores, R&D[1] companies, testing companies and social networks.

Afterwards I will conduct a new phase of research were I will focus on finding solutions to the problems that already exist within the enterprise environment.

In order to achieve a wider test analysis I will use machine-feedback generated by online email assessment tools.

I will base my decisions on my research on user agent problems, in order to find out about which would be the best approaches in building content-rich emails. In the decision making process I will have in mind the factors that lead to the most common problems that generate the biggest compatibility issues.

The main artifacts resulting from my work would be detailed information about the email code and code analyses resulting from empirical knowledge and machine feedback.

1 R&D – Research and Development

4. Email Theory

„Email (Electronic mail) is a method of exchanging digital messages from an author to one or more recipients." - Wikipedia.

4.1. Short History

The email emerged as a way to deliver **text content** from a sender to one or multiple receivers. This form of communication appeared in 1965 when users of a main-frame computer[1] could communicate electronically by leaving text messages in each other's user directory.

Ray Tomlinson is credited to invent the email in 1972 when users could easily send digital messages between different machines, long before the World Wide Web. The technology around the email was conceived and standardized to handle text emails. With the emergence of content-rich web sites, at the beginning of the 90's some of the email client developers thought to bring in similar functionalities to the emails as well.

Because the transition to email clients with capabilities of displaying HTML tags was very slow, during the following 10 years there was a strong opposition to HTML emails.
The main reasons for the opposition was the big size of the HTML emails which would result in higher bandwidth expenses. Another issue was the fact that email clients which were not ready for HTML were displaying the HTML code itself which made the emails unreadable.

4.2. Introduction

Structurally an email is defined as a two entity parts, the header and the body. The header contains information about the origin and destination of the email while the body contains the message itself.

Since the focus of this paper is on looks of the email, the body would be in the spotlight throughout the paper.

A simple email structure example:

```
From: foo@bar.net
To: bar@foo.net
Subject: Hello bar from foo

Hello bar! Did you notice this email?
```

In the above structure, the blue marked text represents the headers of an email while the green marked text represents the body. There should always be an empty line between the 2 entities.

1 Main-frame Computer – computer used for critical applications, within corporate and governmental organizations

4.3. Email Structure

The first email message standard was very simple, and was described in RFC822. According to the standard an email could contain only a header part and a body part separated by an empty line. But this standard did not fulfill the demandingly increasing requirements of the users, and therefore a new revised standard emerged.

The MIME appeared. MIME stands for Multipurpose Internet Mail Extensions which arrived to help users divide the email content in several more parts.

MIME made attachments and multiple versions of the same content possible. For example one could send the same message using both plain-text and formatted-text versions embedded in the same email message.

MIME brought a tree-like structure in the email world. Having a tree-like structure increased the possibilities for more complex, dynamic message content. This was a huge milestone in the email development.

A simple message body organized in a tree-like structure can be seen in the following figure. Where root means the container of the message body, and the blue elements represent multipart MIME entities while the red ones represent single part MIME entities.

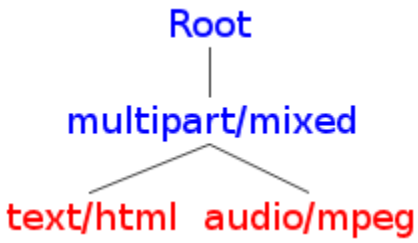


Figure representing a simple MIME structure, source: OpenPOP.NET project

Another more complex structure with more branches and leaves is represented in the following figure.

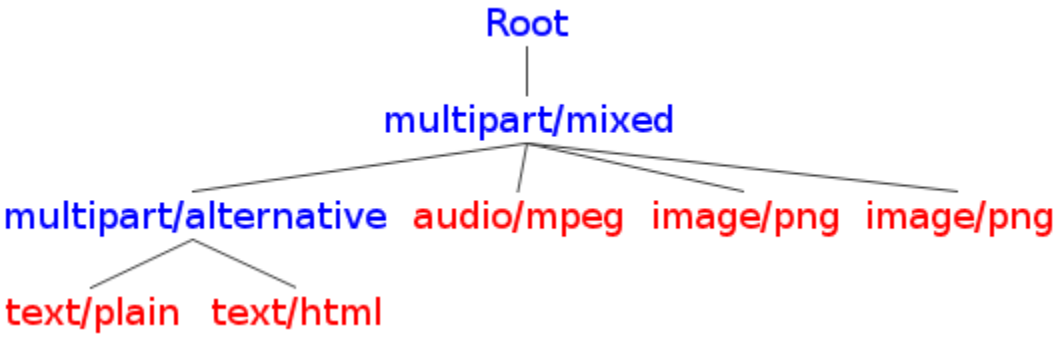


Figure representing a complex MIME structure, source: OpenPOP.NET project

Another shortage of RFC822 to overcome was the use of non-ASCII[1] characters which was now possible by declaring the character set used and the encoding in the subparts header.

1 ASCII - American Standard Code for Information Interchange – a standard used to define all the american text characters found on a type-writer; it was implemented in the early computer systems as well

Two popular encodings are base64 and Quoted-Printable. This development allowed mail users to send mails in virtually any language.

However only encoding systems that output ASCII characters can be used to transport text within a MIME container.

4.4. MIME Detailed

“Multipurpose Internet Mail Extensions (MIME) is an Internet standard that extends the format of email to support:

• Text in character sets other than ASCII

• Non-text attachments

• Message bodies with multiple parts

• Header information in non-ASCII character sets”[1]

4.4.1. MIME Headers

The MIME headers are used to define the content of each individual part of email body.

The headers contain information about the version, type of content, disposition of content and its encoding. The MIME headers are described in the following table:

Header Name	Description	Common Declarations
MIME-Version	marks the presence of a MIME-formatted email and informs the agent of its MIME version	MIME-Version: 1.0
Content-Type	indicates the Internet content type of the content present in the current email content part.	Content-Type: text/plain Content-Type: text/html Content-Type: multipart/mixed
Content-disposition	announces how the content would be presented; whether it needs to be presented in-line or as an attachment.	Content-disposition: attachment; filename=picture.jpg; modification-date:"Wed, 12 Feb 1997 16:29:51 -0500";
Content-transfer-encoding	defines how the ASCII representation of the binary data should be interpreted	Content-transfer-encoding: base64 Content-transfer-encoding: quoted-printable

1 Definition extracted from Wikipedia.org

Since all the email and MIME headers have to be represented as ASCII, the default encoding used within MIME headers is called **encoded-word** and can be used to write title and file names in foreign languages.

The encoded-word is interpreted in the following order:

"=?charset?encoding?encoded text?="

Multipart messages

The main reason to adopt the MIME was multipart messages. This is the most important characteristic of MIME.

There are different types of MIME multiparts:

mixed – used to combine different content types in an email

digest – used to send multiple text messages of type "message/rfc822"

message – contains an email message including headers; used in forwarded emails

alternative – describes that each part is an alternative version of the same content; often used to send alternate versions of HTML text and plain text

related – made for compound objects consisting of several related components where the proper display is achieved only with the use of all the parts involved, the main use of the related multiparts is to send a complete email (including the graphic elements)

report – contains data formatted for the mail server to read

signed – used to enclose digital signatures

encrypted – used to enclose encrypted content

form-data – used to express the submitted values through a form

mixed-replace – used to receive a “stream” of parts, each one replacing the previous one as soon as it is fully received

byteranges – used to receive different byteranges

The last 3 multipart types: form-data, mixed-replace and byteranges are not suited for email content.

4.5. text/html MIME Type

text/html is a MIME type that defines the hyper markup text content structured in HTML.

This MIME type was designed to declare markup text in HTML, which it did from the beginning.

The MIME developed itself out of the email world, and the HTML itself developed as well. To this day the HTML rendering capabilities of some email clients are very reduced compared to what could be rendered in a modern web browser. Therefore markup text should be constructed more carefully for email clients than for today's web browsers.

4.6. HTML

HTML – Hyper Text Markup Language is used to build rich content for the Web and email.

HTML is constructed of tags. There are two types of tags, opening tags and closing tags.

Opening tag: <html>

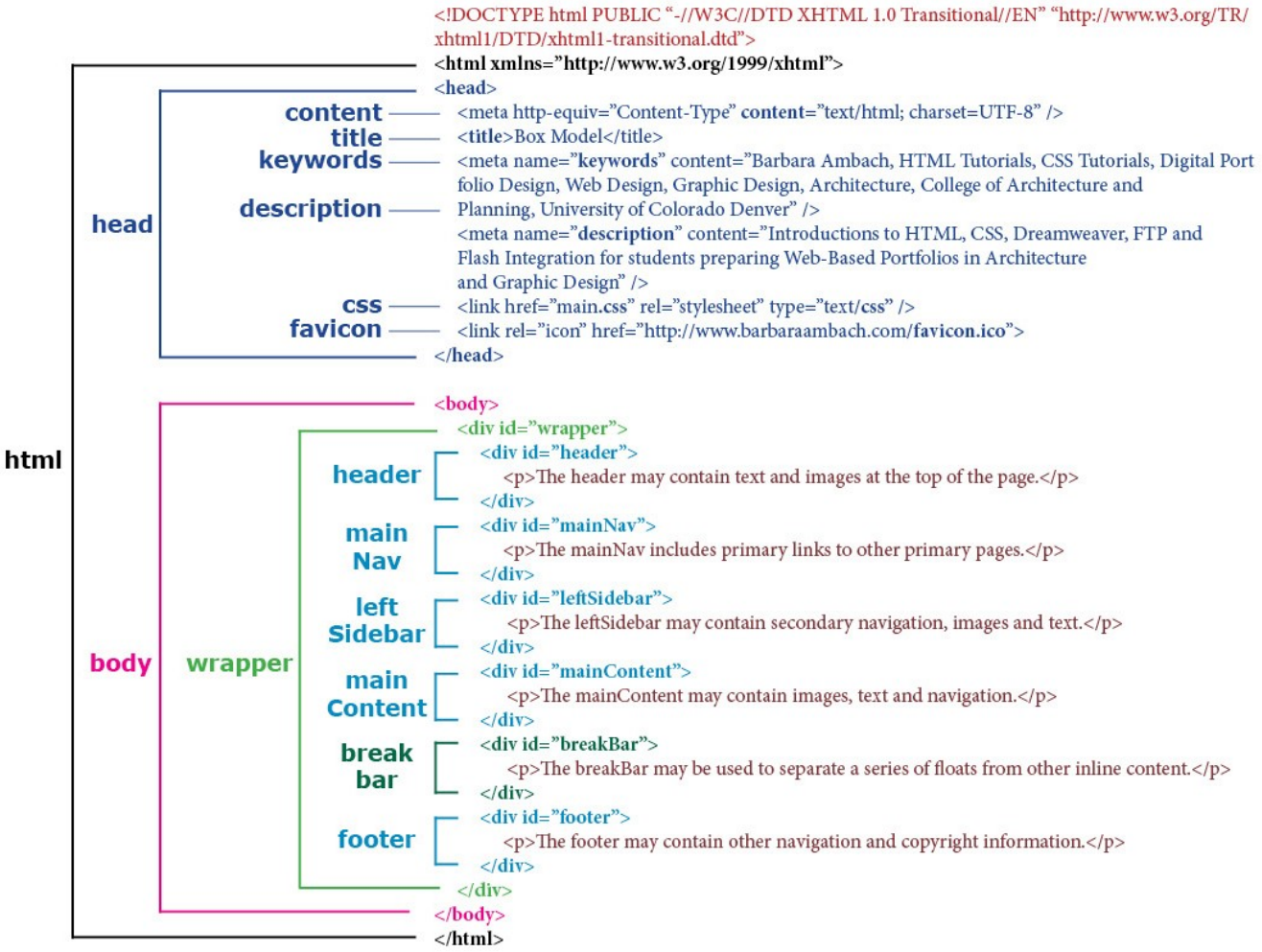
Closing tag: </html>

Some of the tags can have attributes which define more specific characteristics of an HTML element.

Example: <table width="300"> - describes the beginning of a table of 300 pixels of width.

HTML structure

Short overview of the HTML structure



Structure of HTML; source barbaraambach.com

As it can bee seen in the previous structure, the main components of HTML are <!DOCTYPE>, <html>,<head> and <body>.

<!DOCTYPE> - declares the type of the HTML document

<html> - announces the beginning of the HTML code

<head> - announces the beginning of the HTML headers

<body> - announces the beginning of the HTML body

These tags have to be present in every HTML formatted document.

4.6.1. HTML Within MUAs

Some email clients can render only basic HTML, therefore a lot of modern HTML elements are not used when developing email code. The most important highlights are:

- using tags is not common in email HTML therefore most of the emails contain old table-based structures

- styling parameters are only used in-line as properties of HTML entities

- some indentation and image-display codes requires to have one or multiple fall-backs in order to render properly in as many email agents as possible

- among the image formats widely supported are .jpg and .gif so any other picture format would likely generate problems

- the code need to be prepared with fall-backs for anything that could go wrong

- most of the email clients block pictures by default

- usually the maximum real estate of the emails is no bigger than 600px in terms of width

4.7. CSS

CSS – Cascade Style Sheets is a language designed to define fine style adjustments to the HTML components. CSS was added to HTML 4.0 to resolve the problem of HTML element styling and reduce the duplicate declaration within styling elements of the same kind across HTML documents.

CSS code is usually stored in separate CSS files referenced from multiple HTML files.

CSS code is organized in blocks which makes it very easy to manage.

```
a{
  font-family: verdana;
  font-size: 11px;
  color: #ffffff;
}

a:hover{
 color: #BDD1D2;

}
.headline {
  font-weight: bold;
  font-size: 12px;
  color: #ffffff;
}
```

Sample CSS code

CSS in Emails

There are known problems with CSS rendering in email clients.

According to a thorough study realized by Campaign Monitor there is a limited CSS support across the most popular email clients.

Style Element	Outlook 2007/10/13 +	Outlook 2000/03	Apple iPhone/iPad	Outlook.com	Apple Mail 4	Yahoo! Mail Beta	Google Gmail	Android 2.3 (Gmail) +
Tables								
border-collapse	✓	✓	✓	✓	✓	✓	✓	✓
border-spacing	×	×	✓	✓	✓	✓	✓	✓
caption-side	×	×	×	✓	×	✓	✓	✓
empty-cells	×	×	✓	✓	✓	✓	✓	✓
table-layout	×	✓	✓	✓	✓	✓	✓	✓

Short Overview of the supported styles in the most popular email clients.

There are a large number of inconsistencies between the email clients' CSS support. Developers should carefully choose their style tags and always think about fall-backs. If a certain style concept is realizable through HTML code, then the HTML alternative should be preferred over CSS.

4.8. Email Standards

To this day there are no official standards for the content of the emails. This is the reason why many email clients render their emails in very different ways.

However, there is a Email Standards Project hosted at email-standards.org, which is driven by enthusiasts and supported by many developers. They built carefully a standard email template and passed it through each and every popular email client to localize the inconsistencies. Then they made a report and published it online. They contacted every company that did not passed their acid test, and recommended them to improve the HTML rendering within their email client.

Some email clients developers, listened to their petition and improved their email HTML rendering. Among the responders are Yahoo, IBM and Google.

The Email Standards Project started in 2007 after the official lunch of Outlook 2007, with the focus of getting Outlook to change its mind about HTML rendering. The Email Standards organized also the fixoutlook.org website, which is made out of tweets of people complaining about Outlook. To the day, Outlook did not responded the petition.

As the developers are not willing to abide to a common sense standard, the source of the problem can not be fixed. Therefore we have to make our way around it.

4.9. Email Theory Conclusion

Within this chapter the theoretical knowledge about the technologies that cooperate in the process of rendering an email were shortly described.

This chapter reveals what MIME, HTML and CSS are and how they work within the email environment.

At the end the Email Standards project, which tries to make the developers agree on a common-sense email standard, is presented and a short explanation about why the project is not 100% successful is given.

5. Enterprise Email Analyses

In this chapter I will take a closer look at how other enterprises handle content-rich messages within their every-day customer targetted emails.

I will start an assessment of their email content, focusing on key concepts that, when combined, result in the quality of the email they are distributing to their customers.

During the assessment I would investigate emails received from 5 different enterprises with different business markets in the background. The target enterprises will be technology-related firms, social networks and online marketplaces.

In the end I will assess the solutions I implemented for SDA during my internship period and I will draw an overall conclusion based on the facts and the resulting empirical knowledge gathered along the way.

Analysis Introduction

Along this analysis I will take a look at the source code of the received emails and analyze their MIME structure. Afterwards I will extract and decode the HTML content from the email and pass it through the Litmus email testing service located at litmus.com. I will use the feedback provided by Litmus to conceive a conclusion over how the email behaves graphic-wise and code-wise in a wide range of popular email user agents.

Litmus is a company that channels their resources in email testing and email marketing analysis. I will use Litmus as my tool of choice for testing the email HTML code.

During my analysis I will focus on 3 key aspects:

- email structure

- physical appearance

- code sanity

A deeper explanation of the key aspects is presented in the following table:

Aspect	Inspection Focus
Email Structure	Everything related to the email structure that lies at the MIME level. The email structure will be compared with the content of the email in order to decide whether it complies with the purpose of the content or not.
Physical Appearance	Everything that catches the readers' eyes, from the design used to the readability issues. The results in physical appearance will be checked for cross-client rendering consistence and assessed for content readability.
Code Sanity	How well is the code maintained and how many problems it encounters within the popular email clients. A clean code reveals the hard work of a developer that has avoided, by any means possible, the inconsistencies between user agents.

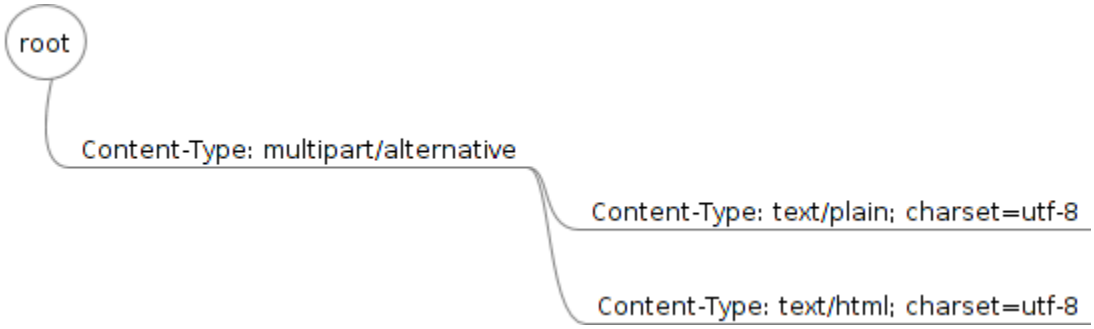
5.1. Freelancer.com

“Freelancer.com is the world's largest freelancing, outsourcing and crowdsourcing marketplace for small businesses.”[1]

I chose one of the announcement emails received from freelancer.com and started analyzing it with the key aspects in mind.

Email Structure

The structure of the email is as follows:



All the email parts uses the following encoding:

- Content-transfer-encoding: quoted-printable

The structure is simplistic, where plain text is a used as a fall-back when html can't be rendered. The charset encoding used, UTF-8[2], will not give any readability problems for users of today's email clients as it can safely encode any character in any language.

The email is an announcement letter containing a small number of remote graphic elements, therefore the simple MIME structure used fits very well with the purpose of this email, where the readable information is the main focus.

Physical Appearance

After parsing the email through the litmus.com service I noticed the following facts:

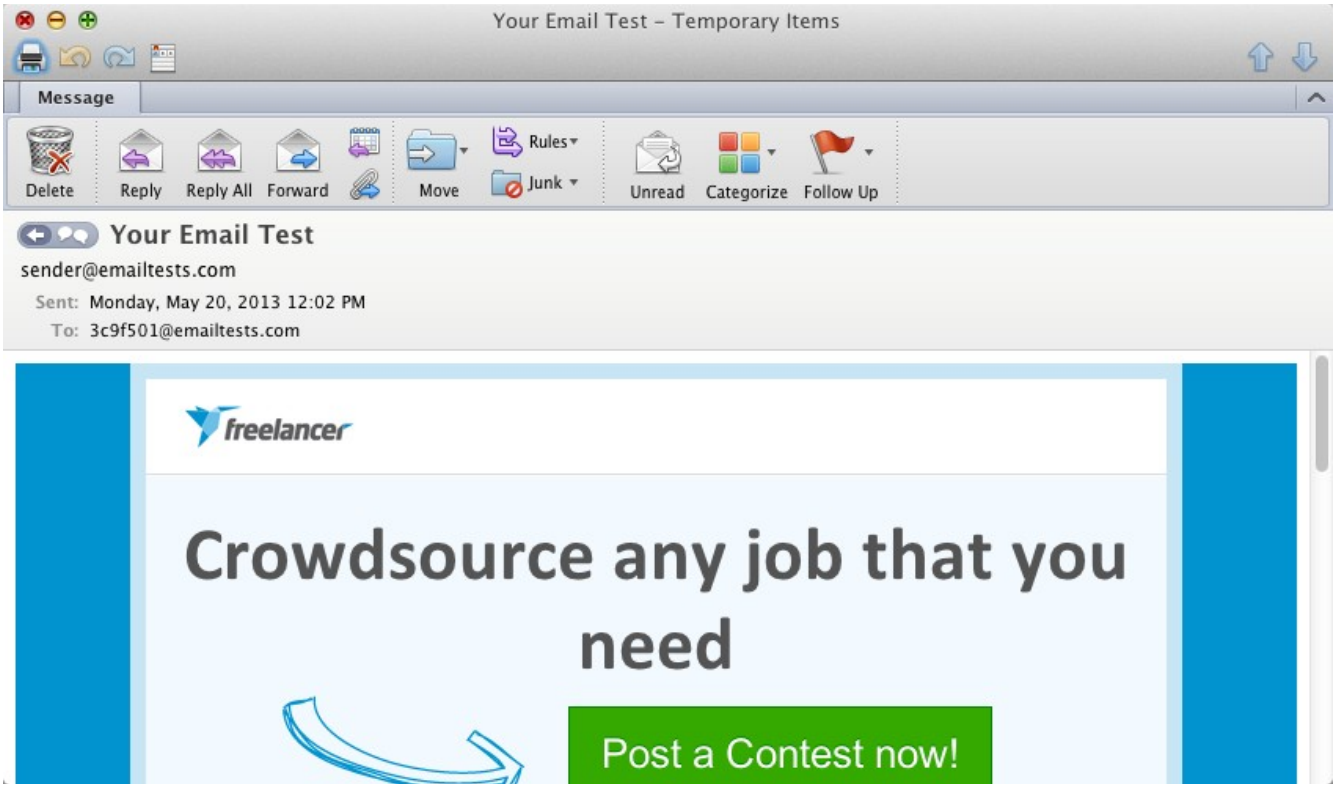
On the Desktop side the email was displayed well in: Apple Mail 5, 6 ; Lotus Notes 8, 8.5 ; Outlook 2000, 2002, 2003, 2007, 2010, 2011, 2013; Thunderbird 3.0 and latest.

The design used is very appealing and easy to read, when rendered correctly.

The email did not render correctly in older versions of Lotus Notes, the main problems can be found in the image rendering. The older versions of Lotus Notes are not able to render png pictures.

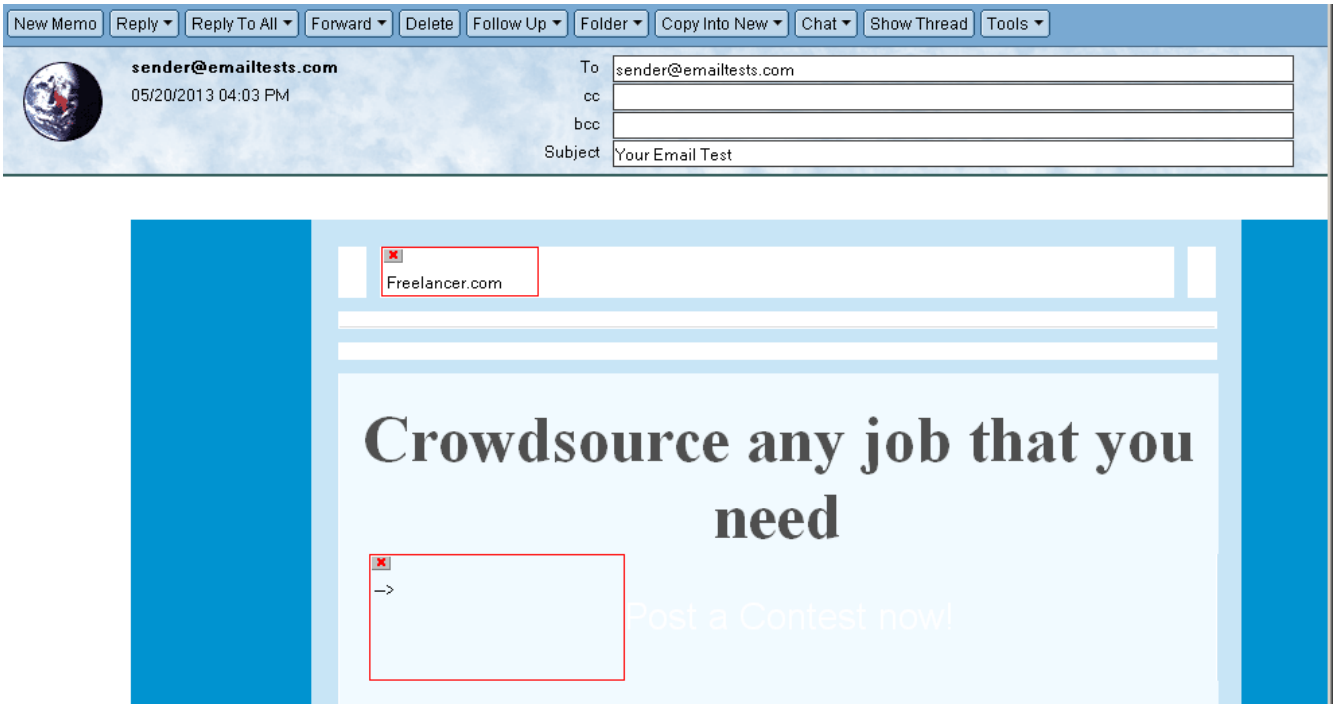
1 Freelancer's own company description
2 UTF-8 – Unicode Transformation Format - 8 bits – variable width encoding that can represent any character within the Unicode character set.

The following figure represents an email preview where there were no rendering problems.



The Email rendered in Microsoft Outlook 2011

As seen in the following figure, the pictures were not displayed within Lotus Notes 6.5, 7.



Freelancer's email faulty rendered in Lotus Notes 7

On the mobile side the email was displayed well in: Android 2.3, 4.0; Blackberry 5, Ipad, Iphone 4S, 5; Symbian. To be noted that the email displayed well on all mobile devices tested, although it was not built with the mobile users in mind. Therefore it is unpleasant to read the email on the mobile devices with low screen resolution.

Within the popular Webmail services the email was displayed well. Among the webmail user agents tested are: AOL Mail, Gmail, Outlook.com and Yahoo.

In conclusion, the email content displayed very well in a wide majority of user agents, however there is a small room for improvements, especially when addressing the pictures rendering. The design is well-structured and it is still readable when the images are blocked, which is a plus.

Code Sanity

According to Litmus' Code Analysis service the email code contained 304 potential problems from a set of a 862 compatibility rules tested.

Among those 304 potential problems the most important ones related to:

- no png pictures support in Lotus Notes

- background-color and background-image are not supported in various agents

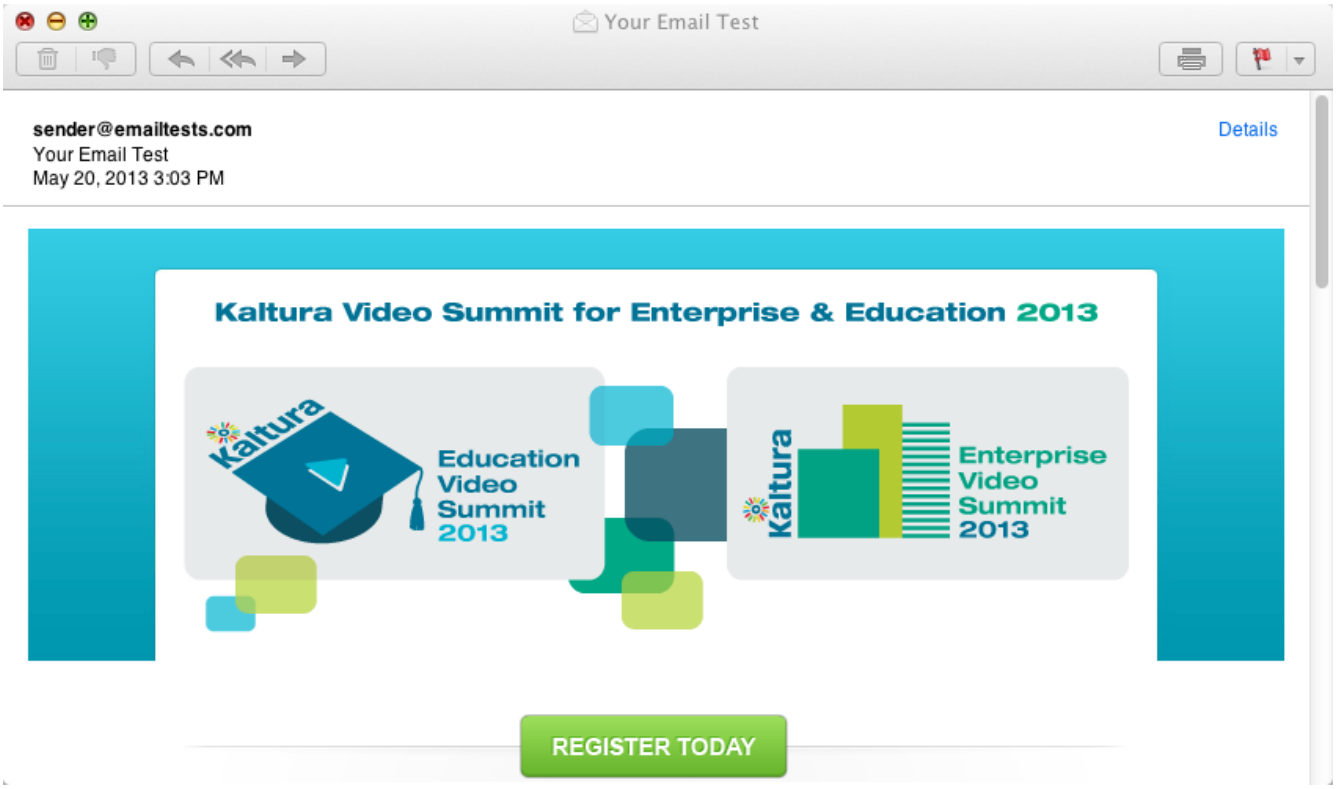
- margin and padding are not supported in various agents

Since the marked-up message displayed well in a majority of user agents, the code itself is good. Although the code is not clean and there is no sign of avoiding incompatibilities that could rise in other user agents.

5.2. Kaltura.com

Kaltura is the leading online video platform on the market. They provide the world's first Open Source Video Platform which rapidly found a place in the use-cases of enterprises, media companies, educational institutions, service providers and web publishers.

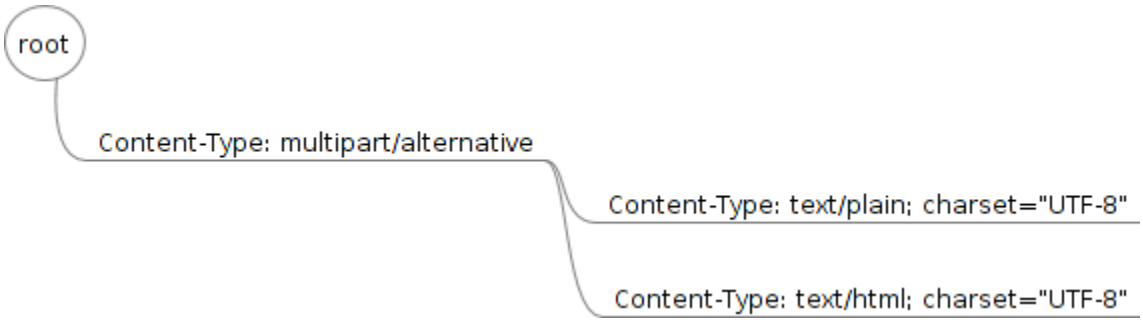
For this analysis I chose one of the announcements newsletter received from them. An email preview is shown in the following picture:



Kaltura's email code rendered in Apple Mail 5

Email Structure

The structure of the email is represented as follows:



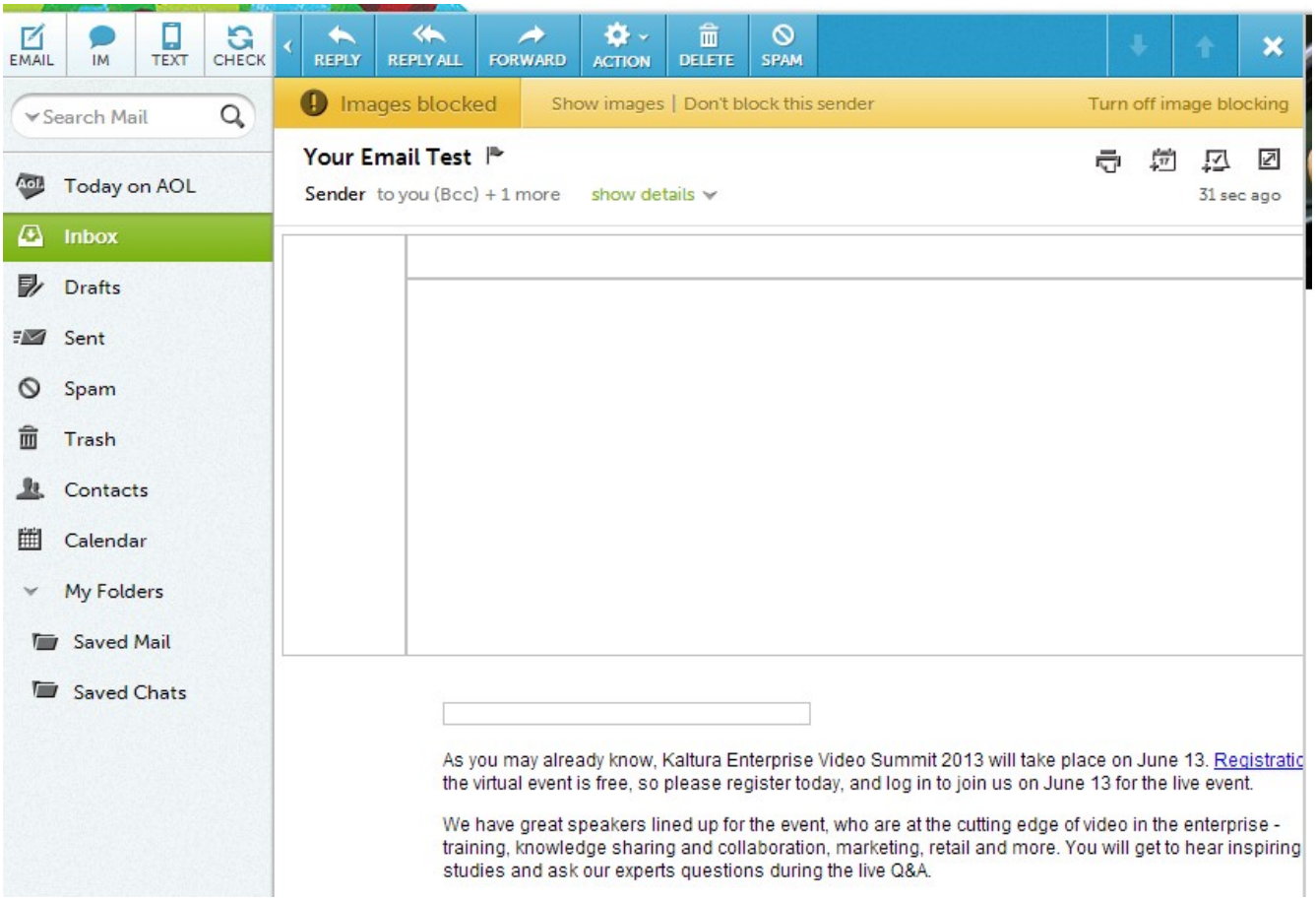
Besides the Content-Type, the following 2 MIME headers are declared as well:

```
Content-Transfer-Encoding: 7bit
Content-Disposition: inline
```

Arguably it can be said that structurally, the email could contain the top picture because it plays a significant role in the transmitted message. By not doing so the reader risks missing the important part of the message, if the user does not agree to display remote content.

For this reason the email structure used is not proper for this announcement email. Moreover the email begins with a banner for an upcoming event which would not be displayed if the user would not accept the use of remote graphic content.

When that happens, the banner, and all images in general, should at least have a fall-back text to be displayed. This email does not have such backup, and it would look like in the following picture, when a user who does not allow remote content receives it:



The email rendered in AOL email, as displayed by default, with remote content not allowed

Physical Appearance

On the Desktop side the email performed well in the following user agents: Apple Mail 5,6; Lotus Notes 7, 8, 8.5; Outlook 2000, 2002, 2003, 2007, 2010, 2011, 2013; Thunderbird 3 and latest. There was a small issue with Lotus Notes 6.5 and 7 which did not display a picture located at the bottom of the email because it was in png format.

Within the web services the email performed very well on all agents tested. The email was tested in AOL Mail, Gmail, Outlook.com and Yahoo! Mail. Each of these services were tested in all the popular web browsers: Internet Explorer, Firefox and Chrome.

On the mobile devices the email performed well in Android 2.3, 4.0, Outlook.com, Yahoo, iPad, iPhone 3GS, 4S, 5; Symbian and Windows Phone 7.5.

The content of this email is not mobile device friendly as it contains of wide pictures which are hard to follow because of the constant need for scrolling horizontally.

Test result previews shown in the Litmus Email Testing Service

As seen above the results are astonishing, this email is very close to 100% user agent reach. The only problem was in older versions of Lotus Notes where one of the pictures didn't displayed because of the format not being supported. Otherwise hats down!

As a conclusion I would like to point out that the content rich aspect of this email rendered successfully in the wide variety of email clients, and it delivered its message well.

Code Sanity

Litmus' code sanity check reports 571 potential problems out of 1009 compatibility rules.

Among the most potential problems encountered in the mail code are:

- indentation tags like margin, width, padding and border are not supported in some agents

- background-color is not supported in Outlook's clients and services

- spacing tags like line-height and white-space not supported in some user agents

The code in this email literally tries everything and takes a lot of fall-back measures which is a good practice, but at the point it is now it encounters a lot of errors in various user agents. Moreover it is very hard to maintain.

5.3. Google+

Google+ is one of the biggest social networks world wide. With over 500 million users there is no doubt that it has to deal with an enormous amount of emails.

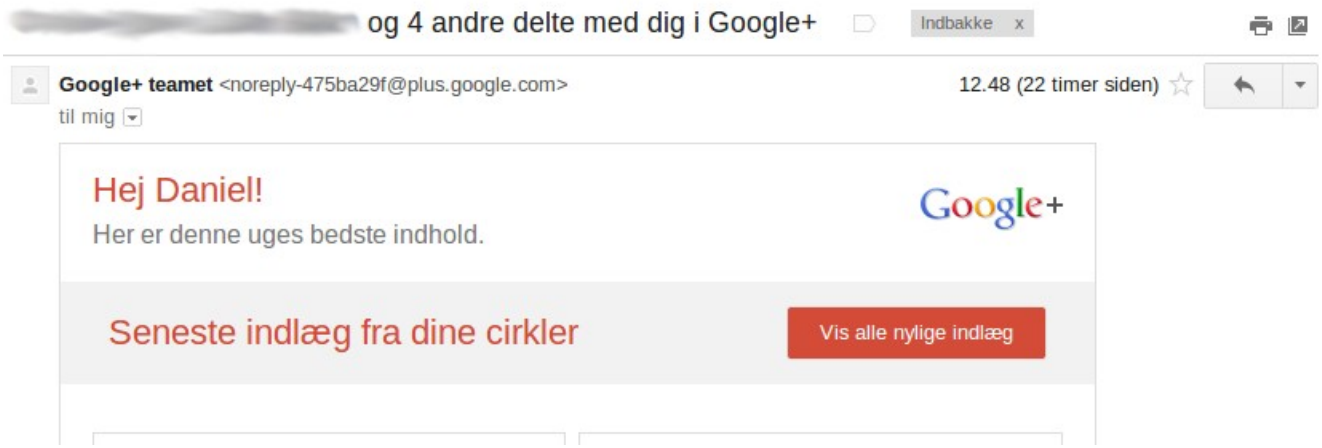
Therefore I will begin analyzing how good those emails really are.

Email Structure

Each MIME header has a Content-Transfer-Encoding defined as well.

The interesting part is that they are different, a base64 encoding is used for the plain text while a quoted-printable is used for the html part. With that being said, I don't see any problems with the email rendering.

The MIME structure used fits to the purpose and the content of the message. As this message is an announcement message.

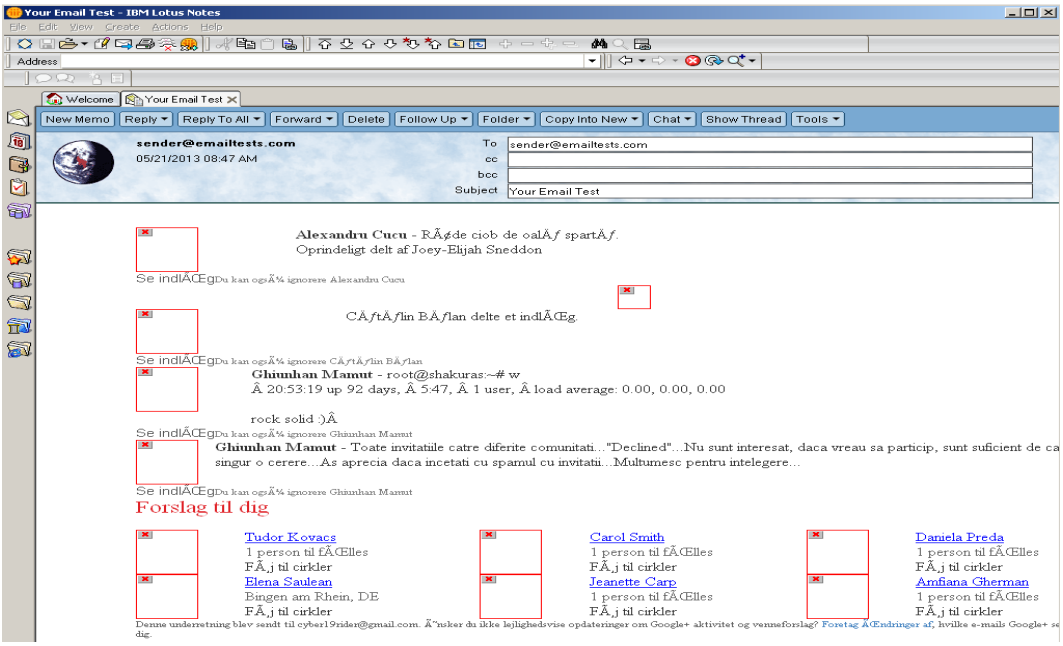


The email rendered in GMail

Physical Appearance

On the Desktop side the email performed well in the following user agents: Apple Mail 5,6; Lotus Notes 8, 8.5; Outlook 2000, 2002, 2003, 2007, 2010, 2011, 2013, Thunderbird 3 and latest. The email displayed great in almost all user agents. There are big problems with the pictures in Lotus Notes 6.5 and 7.

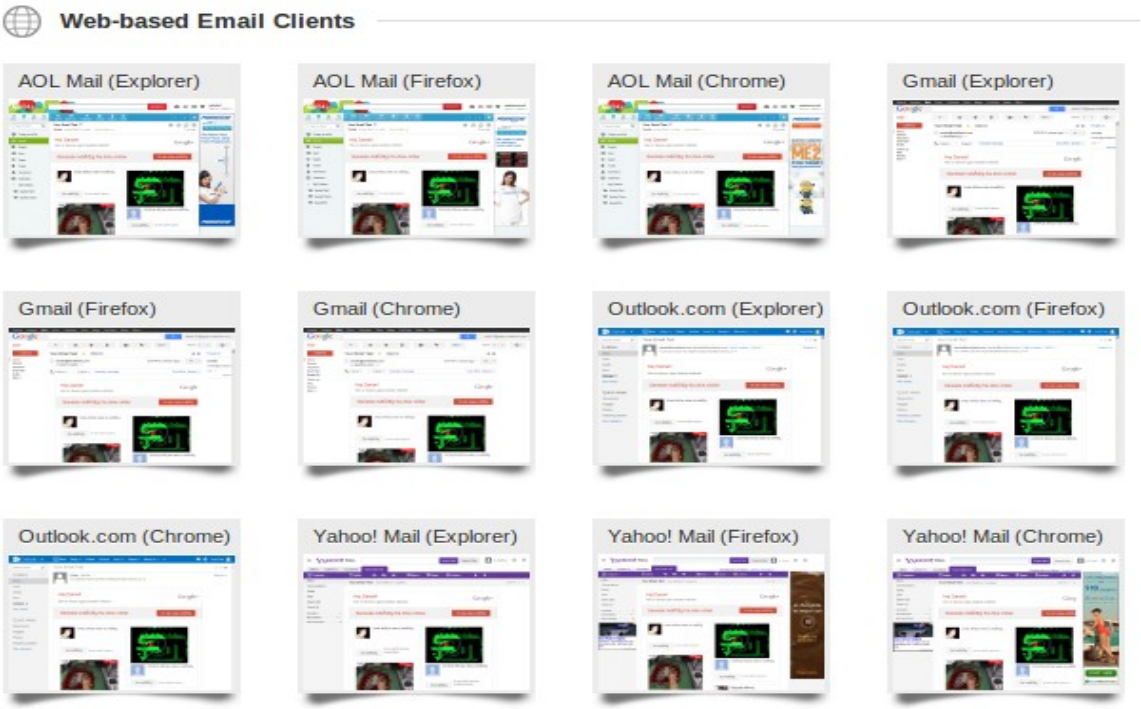
The problems encountered within the older version of Lotus Notes are shown below:



Lotus Notes 7 not rendering any of the pictures within the email

The pictures in this email are not rendered within Lotus Notes versions prior to 8, because they are of png type which is not supported in Lotus Notes.

Within the web services the email performed very well on all agents tested. The email was tested in AOL Mail, Gmail, Outlook.com and Yahoo! Mail. Each of these services were tested in all the popular web browsers: Internet Explorer, Firefox and Chrome.



An overview of the email displayed within the web services.

On the mobile side the email displayed well within all the MUAs tested. The tested devices were Android 2.3, Blackberry 5, Outlook.com, Yahoo, iPad, iPhone 3GS, 4S, 5; Symbian and Windows Phone 7.5

As a conclusion the email is not mobile device friendly and there are unsupported pictures in Lotus Notes versions prior to 8 and Android 4.0 devices.

Code Sanity

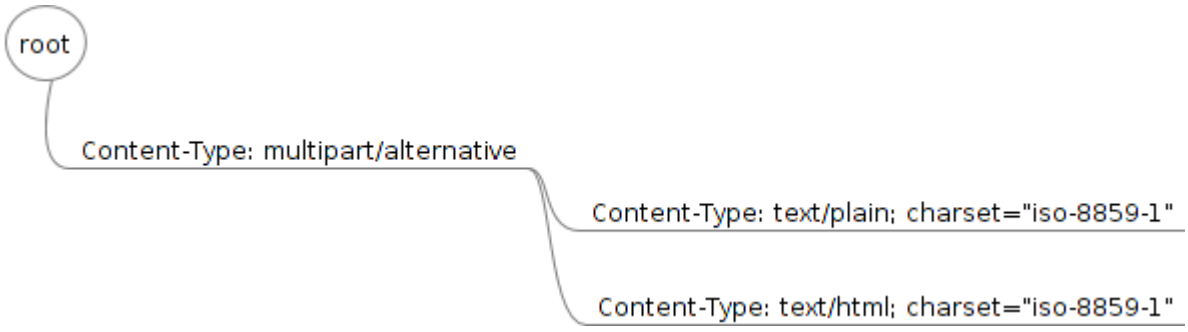
Litmus.com reported 5261 potential problems while applying 995 compatibility rules.

This is a sign of extremely bloated code. Many unsupported html tags were used in coding the email which comes to be unreliable and extremely hard to maintain.

5.4. sportsdirect.com

Sportsdirect is an online clothing shop. I will analyse their offer newsletter which is sent daily to their customers. The newsletter contains many graphic elements.

Email Structure

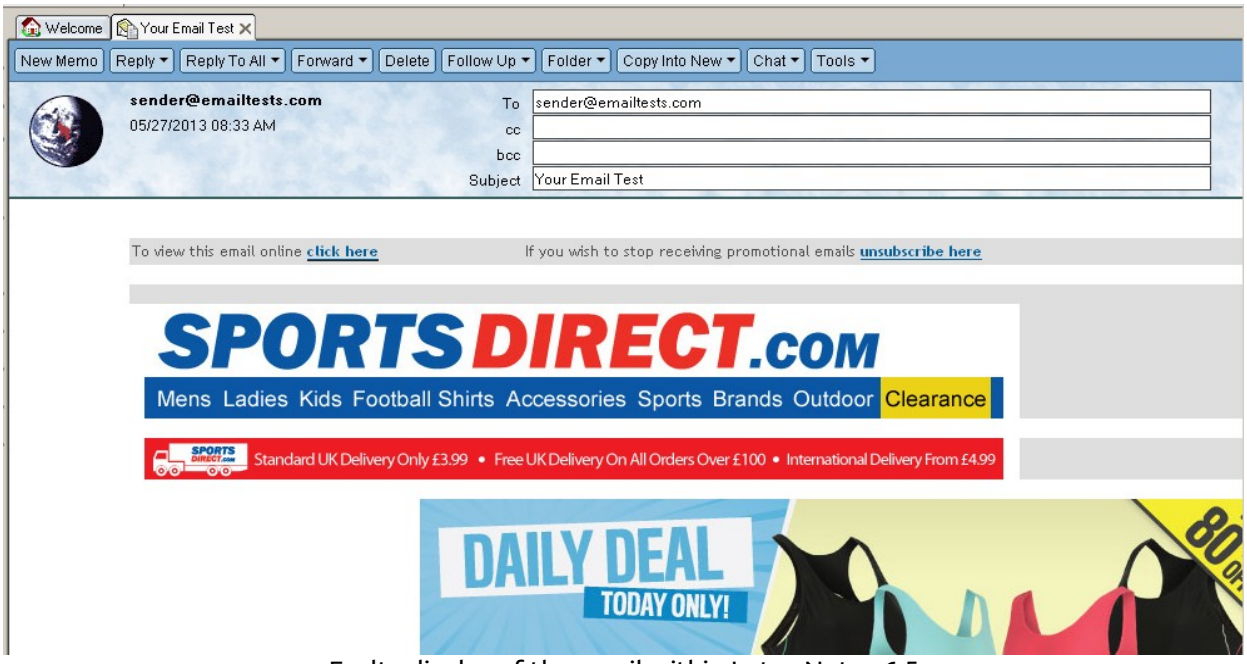


The encryption used for the plain text part is 8-bit while the encryption used for the html part is quoted-printable.

As a conclusion, the structure depicts a simple and good email structure.

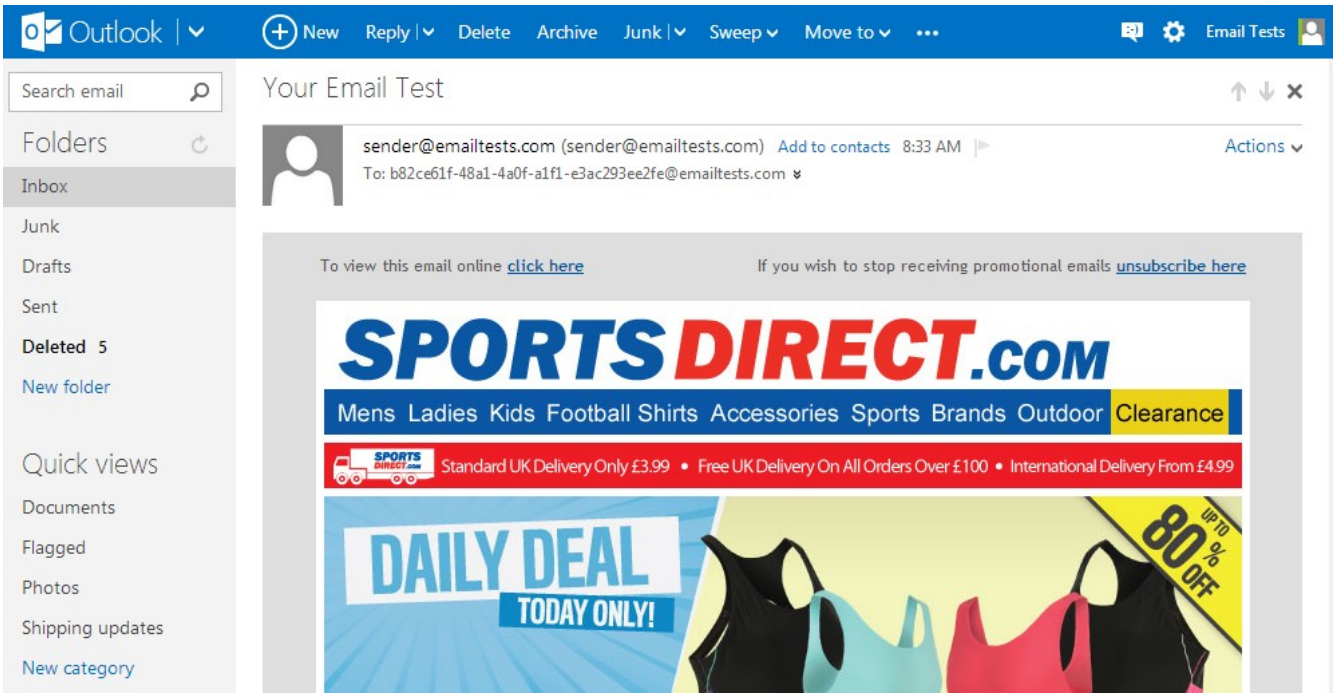
Physical Appearance

On the Desktop side the email performed well in the following user agents: Apple Mail 5,6; Lotus Notes 7, 8, 8.5; Outlook 2000, 2002, 2003, 2007, 2010, 2011, 2013; Thunderbird 3 and latest. The email displayed great in almost all user agents. There are indentation problems with Lotus Notes 6.5 where the top of the email gets shifted to the left.



Faulty display of the email within Lotus Notes 6.5

Within the web services the email performed very well on all agents tested. The email was tested in AOL Mail, Gmail, Outlook.com and Yahoo! Mail. Each of these services were tested in all the popular web browsers: Internet Explorer, Firefox and Chrome.

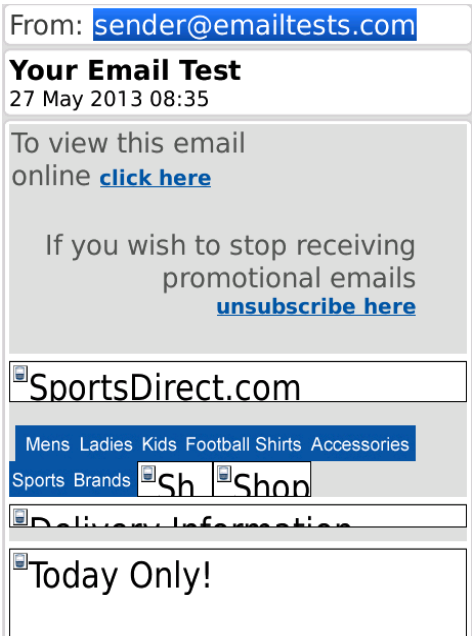


The email displayed on Outlook.com email service.

On the mobile devices the email performed well in Android 2.3, 4.0, Outlook.com, Yahoo, iPad, iPhone 3GS, 4S, 5; Symbian and Windows Phone 7.5.

There is a noticeable problem with Blackberry devices were the images were not displayed at all. (Notice the picture in the right side) Since the content of the email is mostly pictures this is a huge downside for its message.

Furthermore the content of this email is not mobile device friendly as it contains wide pictures which are hard to follow because of the constant need for scrolling horizontally.



Faulty display of the email in Blackberry 5

Code Sanity

litmus.com service reported 1079 potential problems out of 1009 rules checked.

Among the potential problems pointed are:

- indentation tags like margin, width, padding and border are not supported in some agents

- spacing tags like white-space are not supported in some user agents

- 3-character hex color code is not supported in any of the email clients

- background-color is not supported in Outlook's clients and services

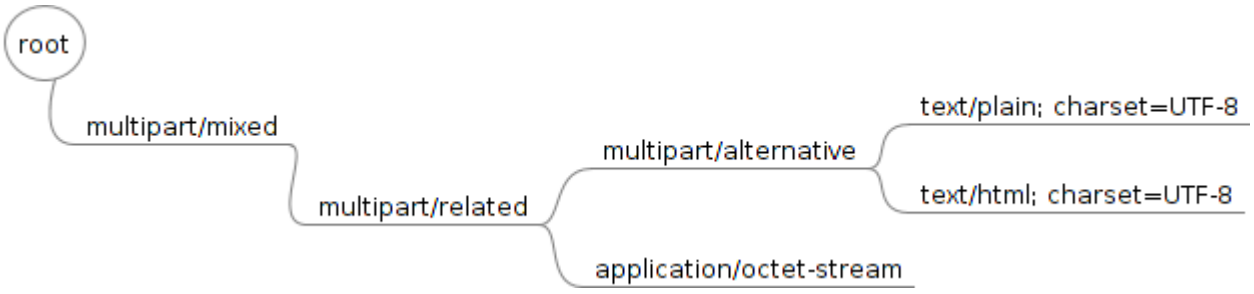
- image map and cursor not supported in some email clients

5.5. linkedin.com

Linkedin is a social network focused on the professional life and business market. It has millions of subscribers and therefore millions of emails to handle.

Email Structure

The graphical representation of the Content-type elements is as follows



From the figure we observe that Linkedin uses an advanced structure for its emails.

With the second node “multipart/related” MIME tells the user agent that the proper displayal of the email is only achieved when the client displays all its subparts. Namely the text part + the attached picture (marked as “application/octet-stream”).

With this practice Linkedin assures that the only picture present in the email is displayed always no matter whether the user accepts remote content or not.

Physical Appearance

On the Desktop side email displayed without issues in the following agents: Apple Mail 5,6; Lotus Notes 7, 8, 8.5; Outlook 2000, 2002, 2003, 2007, 2010, 2011, 2013, Thunderbird 3 and latest.

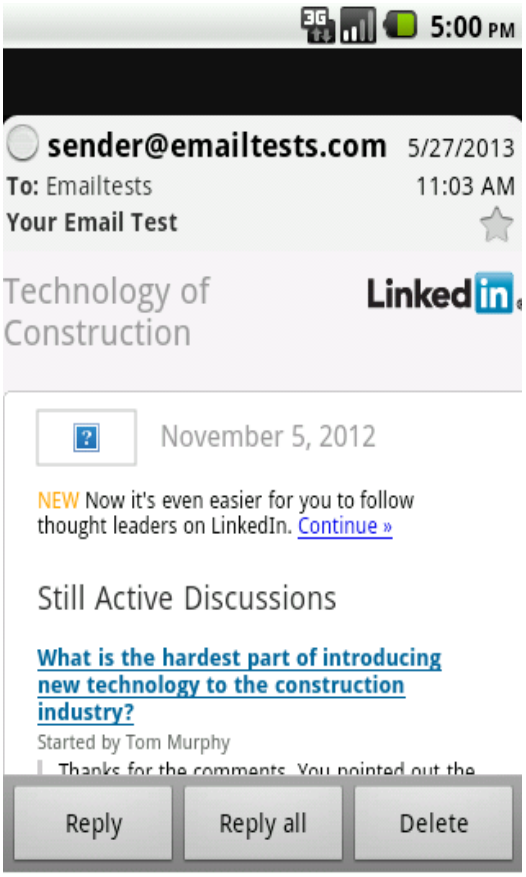
There were indenting issues in Lotus Notes 6.5.

On the mobile side the email displayed nicely in Android 2.3, 4.0, Outlook.com, Yahoo, Blackberry 5, iPad, iPhone 3GS, 4S, 5; Symbian and Windows Phone 7.5.

There is a plus for mobile clients since this email is also handling different screen resolutions, making the reading experience on a mobile device a thrill.

On the web side the email performed greatly within the following email services: AOL Mail, Gmail, Outlook.com and Yahoo! Mail. To be noted that each of these services were tested in all the popular web browsers: Internet Explorer, Firefox and Chrome.

There weren't any visible issues in rendering Linkedin's email within the web services tested.



Email displayed in an Android 2.3 email client

Code Sanity

There were 3172 potential problems when I checked the code against litmus.com's email testing service. This is a sign of code bloating.

Among the potential problems pointed are:

- indentation tags like margin, width, padding and border are not supported in some agents

- png images are not supported in some user agents

- spacing tags like line-height and white-space not supported in some user agents

- 3-character hex color code is not supported in any of the email clients

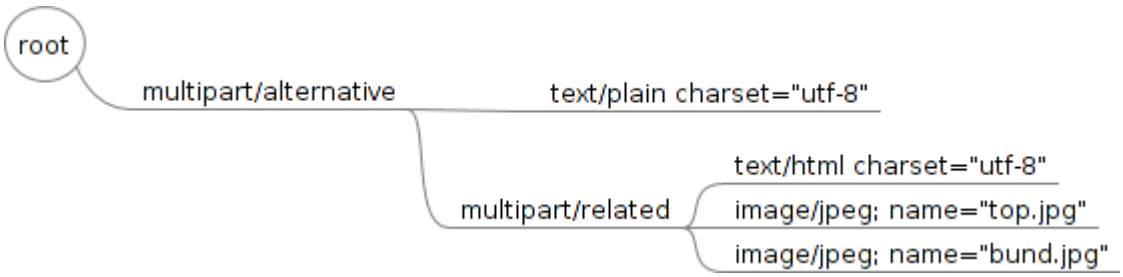
- background-color is not supported in Outlook's clients and services

5.6. adventist.dk

Now it is time to look at the newsletter I developed during my internship period. The newsletter is sent on a daily basis from the website adventist.dk to around 100 subscribers.

Email Structure

The graphical representation of the MIME Content-type structure is shown below:



To be noted that for the text parts the encoding used is 8-bit, while the jpg images are encoded in base64 and disposed in-line.

The email is structured in 2 main alternative parts, the first one being the plain text version while the other one an assembly of marked up text and in-line pictures. This structure achieves the intended goal of reaching as many readers with a full graphic-rich email without requiring any remote content acceptance from the user's side.

Physical Appearance

On the Desktop side there were problems with displaying the pictures in Lotus Notes versions 6.5 and 7. In all the other email clients the email rendered well. This email was tested in Apple Mail 5, 6; Lotus Notes 6.5, 7, 8, 8.5; Outlook 2000, 2002, 2003, 2007, 2010, 2011, 2013 and Thunderbird 3.0 and latest.

The picture on the right side shows how the email should be displayed when there are no rendering issues, while the next picture, located below, shows how the email would look in a client that has rendering problems. The client with rendering problems is Lotus Notes 7 which does not display the pictures referred in the email body because it does not support pictures set as a background.


SYVENDE DAGS ADVENTIST-KIRKEN

NETANDAGT

Den gyldne regel

Derfor: Alt, hvad I vil, at mennesker skal gøre mod jer, det skal I også gøre mod dem. Sådan er loven og profeterne.
Matt 7, 12

Er der i det hele taget nåde for sådan nogle dømmesyge mennesker som du og jeg? Halleluja, det er der! Og du kan finde hemmeligheden i ethvert tømrerforklæde. Det er et målebånd eller en tommestok, som enhver tømrer behøver for at have en ufravigelig, ensartet standard. Brug tommestokken og du får identiske dimensioner og de samme resultater hver gang. Men som enhver tømrer ved, må du overholde reglerne for tommestokken. Hvis du afviger fra dem, ødelægger du dit arbejde.

Dette var naturligvis den guddommelige tømrers pointe med at give os sin vejledning, en ensartet standard, som vi stadig kalder den gyldne regel. Vi har lært den udenad fra barnsben: *"Gør mod andre, hvad du ønsker, de skal gøre mod dig."* Enkelt og ligetil, behandl andre, som du ønsker at blive behandlet. Det er ikke raketvidenskab. Det er den gyldne regel for anstændig karakterdannelse.

Som min ven og kollega, Skip MacCarty, bemærker, har det været den gældende, gyldne regel for Faderen, Sønnen og Helligånden i al evighed: *"Den treenige kærlighedspagt åbenbarer sig, hver gang andre bliver behandlet sådan, som Han ville ønske at blive behandlet, hvis rollerne var byttet om ... Den gyldne regel (Matt 7, 12) måler dybden af Guds binding inderst inde i Treenighedens og udadtil i hele skabningen"* (In Granite or Ingrained? Pp.4, 5). Den guddommelige tømrer har altid levet efter sin egen gyldne regel. Det er også derfor Jesus, som havde enhver ret til at fordømme sin forræder Judas, stilfærdigt kaldte ham *"ven"* Matt. 26, 50. Jesus behandlede Judas præcis sådan, som han selv ønskede at blive behandlet, som en ven. Læg mærke til, hvordan han behandlede sine spottende bødler. Dommeren over alle på jorden (Joh 5, 22), som alene med rette kunne overdænge hele den sørgelige flok med skældsord - nægtede at fordømme dem. Da de strakte den guddommelige tømrers hænder ud for at nagle dem fast til træet, det materiale han selv havde arbejdet med engang, hentede han i stedet med sine sidste kræfter reglen frem fra sit knuste hjerte, sin gyldne regel. Og han behandlede sine fjender sådan, som han ville have ønsket, at de skulle behandle ham, hvis deres roller havde været byttet om. Han bad om nåde for at frelse dem.

Tror du, Jesu eksempel kan være den enkleste modgift mod nutidens kritiske ånd? At gå i forbøn for hende i stedet for at kritisere hende. At bede for ham i stedet for at dømme ham. Kan du forestille dig, hvad der ville ske, hvis vi levede efter Golgata tømrerens gyldne regel?

Dagens andagt (c) www.adventist.dk

Netandagterne for 2012 er fra bogen "The Chosen", skrevet af Dwight K. Nelson.
Bogen er udgivet af Review and Herald Publishing Association, 2011.
Du modtager denne mail, fordi du har sagt ja til en daglig andagt fra Adventistkirken. Svar venligst ikke denne mail.

Ønsker du at framelde dig denne service, send venligst en mail til: andagt@adventist.dk

A figure about how the email displays in Outlook 2013

The email faulty displayed in Lotus Notes 7

On the webmail services the email displayed well in all the major email agents like AOL Mail, Gmail, Outlook.com and Yahoo! Mail. The email was tested on all these 4 services inside of the top 3 most popular email browsers: Chrome, Firefox and Internet Explorer.

On the mobile side the email displayed well on Blackberry 5, Iphone 3GS, 4S, 5; Symbian and Windows Phone 7. There were problems with the Android's default email client which can not display in-line graphic elements when they are included in the message body as in-line elements.

Also there is a decreased readability of the email because the email was not built with the mobile devices in mind.

The following pictures shows how the email looks on Symbian and Iphone 3GS mobile devices. As it can be seen on the Symbian device is hard to read the email because the user has to scroll a lot both horizontally and vertically. On the other hand on the Iphone 3GS device it is hard to read without zooming and then scrolling horizontally and vertically.

The email displayed on a Symbian S60 mobile device

The email displayed on Iphone 3GS

Code Sanity

173 potential problems out of 1009 compatibility rules checked.

This reveals that the email was thoroughly checked and it does not have many unsupported elements in the code.

Among the problems raised there were:

- positioning tags like, width, padding and position are not supported in many MUAs

- the use of white-space is not supported in a small number of MUAs

5.7. Email Analysis Conclusion

The result of my email analysis concludes with the fact that there isn't a “one size fits all” model in the email rendering world. While some developers are trying to achieve the impossible they often end with an ugly code which becomes a nightmare to maintain.

One point that most of the developers are not yet concerned with is the growing number of mobile devices which have a smaller screen. Most of the emails' content do not fit the width of a mobile device screen. This makes reading email messages on a mobile device a painful experience.

Another point that some email developers miss is having as many fall-backs as possible because some user agents can render a certain aspect of the code but can not render another. And when the email rendering breaks, their email template should be prepared to break in the nicest possible way.

As a conclusion to the email analysis I will remark the main highlights below:

Issue	**Emails Affected**
The email content is not email friendly which gives the user a hard time reading it	5 out of 6 Freelancer, Kaltura, Google+, Sportsdirect, Adventist.dk
Pictures were not displayed within Lotus Notes versions 6.5 and 7 because the type of pictures used were in png format	4 out of 6 Freelancer, Kaltura, Google+, Linkedin
Bloated code structure	3 out of 6 Google+, Sportsdirect, Linkedin
Indentation problems within Lotus Notes 6.5 or 7	2 out of 6 Linkedin, Adventist.dk
Images does not load on mobile devices such as Android because the client is not able to render inline pictures or inline pictures get stripped upon mail downloading	2 out of 6 Linkedin, Adventist.dk
No pictures displayed in Blackberry	1 out of 6 Sportsdirect

As the Highlights are pointing out, the 2 most common problems with the email code are the mobile unfriendliness and picture rendering issues because of unsupported picture type.

6. Email Research

6.1. Introduction

Now that I see where the problems usually lie I'm going to try and find a middle way to get all the email clients to work with different types of content.

During this research I will take the issues discovered during the email analysis phase and try to find the best possible solutions.

The first aspect I will have in focus is the pictures, because they play a huge role in email aesthetics. Afterwards I will take a look at why it is increasingly harder to get the content to render properly in Outlook 2007 and above.

Then I will research how can I make the emails more readable on mobile devices.

As a last resort I will check how animations and videos can be presented within the content of the emails.

6.2. Images

Analyzing the results of the previous analysis, the big issues with pictures were for the reason that the picture-format used was not supported by some of the user agents. To be noted Lotus Notes 6.5 and 7.

I tested different tricks and read multiple articles about Lotus Notes not displaying background pictures, and I concluded that there is no way to achieve that.

Among the practices I tested are:

- in-line CSS styles

- styles declared within the head tang and wrapped by HTML comments as suggested in the article: “HTML Email and Lotus Notes” by sitepoint.com

The background images are not supported in Outlook 2007 and later as well as in Lotus Notes 6.5 and 7, but since Outlook can render VML[1], background images can be achieved with code target specifically for Outlook.

Even though the pictures were in .jpg format Lotus Notes 6.5 and 7 cannot display background images under any circumstances. So the developers either should not make use of background pictures at all or drop the support for Lotus Notes 6.5 and 7.

Looking at the email client itself I discovered on IBM's website that Lotus Notes 6.5 was released in September 2003, while Lotus Notes 7 was released in August 2005.

The support for Lotus Notes 6.5 was until 30 April 2010 while the support for Lotus Notes 7 was until 30 April 2011.

Lotus Notes' main target are the enterprises, and more than 8 year passed since the release

1 Vector Modeling Language – VML – proprietary language used to draw shapes and graphic elements

of Lotus Notes 7, and more than 2 years since they stopped the support for Notes 7. I think it is safe to assume that the number of companies that still uses these 2 old versions of the software is low, and decreasing with time.

However if the email's main targets are enterprises it is always better to be safe and NOT use background pictures, because some enterprises tend to keep their IT systems for longer periods of time.

For the reason that every email client tested can render .jpg pictures, it is safe to use it in the emails whenever there is a need for using pictures.

6.3. Outlook 2007+

Microsoft changed the rendering engine for Microsoft Outlook 2007 and later, from the Internet Explorer's rendering engine to Office Word's rendering engine. This change means that one of the most used desktop email client has significantly reduced the rendering of HTML elements within emails.

The most noted problem is that the background pictures are no longer supported. The only way a developer can achieve that is by using Microsoft's VML language code to specifically create a picture object in order to keep the format of the email intact.

There are a few ways of achieving this, and from reading and empirical experience the best and cleanest way to do it is explained in a Campaign Monitor blog post dating from 2010.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org
/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns:v="urn:schemas-microsoft-com:vml">
<head>
```

Code published on Campaign Monitor's blog

In the first part we notice the addition of the „**xmlns:v=”urn:schemas-microsoft-com:vml”**” in the HTML tag declaration. This tells the Microsoft's rendering engine that the email contains VML code. Other user agents are not affected by this declaration because they do not know how to interpret VML so they ignore the declaration.

Now in order to display a background picture within the email's body we have to have a code similar to the following:

```
<body style="margin:0; padding:0; width:100% !important;">

<!--[if gte mso 9]>
<v:background fill="t">
   <v:fill type="tile" src="http://www.example.com/background_image.jpg" />
</v:background>
<![endif]-->

<table width="100%" cellpadding="0" cellspacing="0" border="0">
  <tr>
    <td align="center" background="http://www.example.com/background_image.jpg">
    <table width="600" cellpadding="0" cellspacing="0" border="0">
        <tr>
          <td>
             <p> If you can see this over the image, background images are successful. </
p>
           </td>
        </tr>
      </table>
    </td>
  </tr>
</table>

</body>
```

Code published on Campaign Monitor's blog

Note the VML code that fills the background with a remote picture. The code is enclosed within HTML comments, so it would not affect the email rendering performance in other email clients.

Now in order to have multiple background pictures within the email, we would use background picture for table elements. The most recommended table element to use background pictures on is the table data „<td>” element which is also known as a table cell.

This can be achieved with a code similar to the following approach.

```
<table width="600" cellpadding="0" cellspacing="0" border="0">
  <tr>
    <!-- Backup background color is #DDDDDD -->
    <td bgcolor="#DDDDDD" style="background-image: url('http://www.example.com
/background_image.jpg');" background="http://www.example.com
/background_image.jpg" width="600" height="120" valign="top">
    <!--[if gte mso 9]>
       <v:rect style="width:600px;height:120px;" strokecolor="none">
          <v:fill type="tile" color="#DDDDDD" src="http://www.example.com
/background_image.jpg" /></v:fill>
       </v:rect>
       <v:shape id="theText" style="position:absolute;width:600px;height:120px;">
       <![endif]-->
    <p>If you can see this over the image, background images are successful.</p>
    <!--[if gte mso 9]>
       </v:shape>
    <![endif]--></td>
  </tr>
</table>
```

Code published on Campaign Monitor's blog

Within the code above a VML shape is created. Its dimensions needs to be defined, and its picture source as well as the dimensions of the overlapping text area. After that is achieved the overlapping content (which would usually be marked-up text) is inserted and then the shape is enclosed just before closing the table data element (represented by „</td>”).

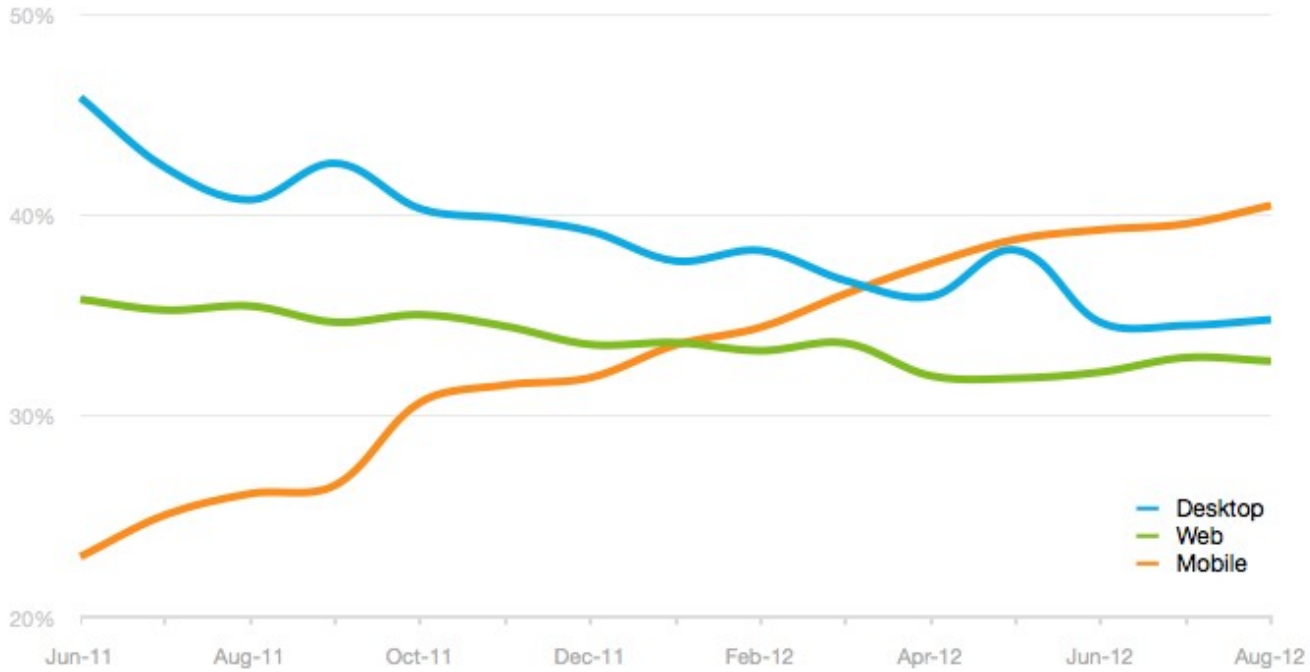
Again, all the VML code is carefully wrapped around HTML comments so that the other email user agents do not interpret it in any way.

From a developer's point of view, the previous procedure would not be nice when applied to every email. But this approach provides a work-around the Outlook's inability to display background pictures. So whenever a background picture is a must, this procedure can be safely used.

6.4. Mobile Rendering

The biggest issue in the email world right now is mobile friendliness, and recent studies show that the mobile consumers are increasing at a very fast pace.

Despite the growth of mobile email usage, most of the email developers are still behind with their designs and did not create mobile friendly emails.



Email Growth in 2011-2012, graphic realized by Campaign Monitor

The above diagram represents the growth of mobile email usage between the summer of 2011 and the end of the summer of 2012. And to give more recent data, I checked an email client popularity statistic for May 2013. The results are stunning. At the end of the month from the top 10 email clients 39% of the emails were read just on mobile devices. The full statistic is shown in the Appendices at section 10.1..

Since the mobile email is increasingly growing I'm starting to test and improve the SDA's email to become mobile friendly.

As stated in many papers and articles like “Optimizing your email for mobile devices with the @media query” published by Campaign Monitor on June 16, 2010; in order to achieve good mobile rendering we need to use CSS media queries.

What are CSS media queries?

A media query is a new standard, accepted by W3C[1] in June 2012. The queries are used specifically to target different screen resolutions. Since we have many small devices, the common use of media queries is to design marked up HTML for hand-held devices.

The mobile design is usually made in one column, and stacking links on top of each other is avoided, since it is hard to touch a specific link on a smart phone.

The media queries consist of a media type and an logical expression which validates to either true or false. For example:

```
@media screen and (min-width:500px) { ... }
```

The query above applies for screen resolutions with a width higher than 500px. The entire CSS code triggered for those devices would be inside the curly brackets.

1 W3C – World Wide Web Consortium – organization that assures the well being of the web standards

How to Use Them in Emails?

The use of media queries is relatively easy. They are used as any other CSS code, however there are a few exceptions in the HTML email.

Within the mails they are declared in the head tag like any other CSS style. This is the only way to make use of CSS media queries in an email message. To be noted that most of the head tags are stripped out in the web email services and desktop clients, but this does not apply in the mobile clients.

Best Approaches For Email Mobile Friendliness with Media Queries

From the articles read and my experience with the queries I concluded a few highlights.

The most important screen resolutions to target are and 320x480px and 480x320px which are used by most email devices, usually smart-phones.

A few other higher resolutions could be targetted, but at higher resolutions your standard email size would most probably fit the screen.

A full list of media queries for standard devices was published by css-tricks.com in the article „Media Queries for Standard Devices".

6.5. Animations

During my research there was a topic that caught my eyes. Namely, the use of animations within emails. I found a service that was claiming to do just that. So for the purpose of this research I will take an email template and insert some gif animations inside it then test it against a few mail user agents.

Results of a mail containing an animated runner logo

From the first test it looks like there is absolutely no problem with the gif rendering. All email agents tested rendered the animation. However since within the Litmus service I can see only static result pictures, I also checked another study. The results of the study shows that not all of the email clients animate the picture. Outlook versions 2007 and above plus Windows Mobile 7 displayed just the first frame of the animation.

This behavior was mentioned in a Campaign Monitor study with the title "Animated GIFs in email: A new approach to an old format" published on 12th April, 2012.

The second test reveals the same results while using a bigger animation.

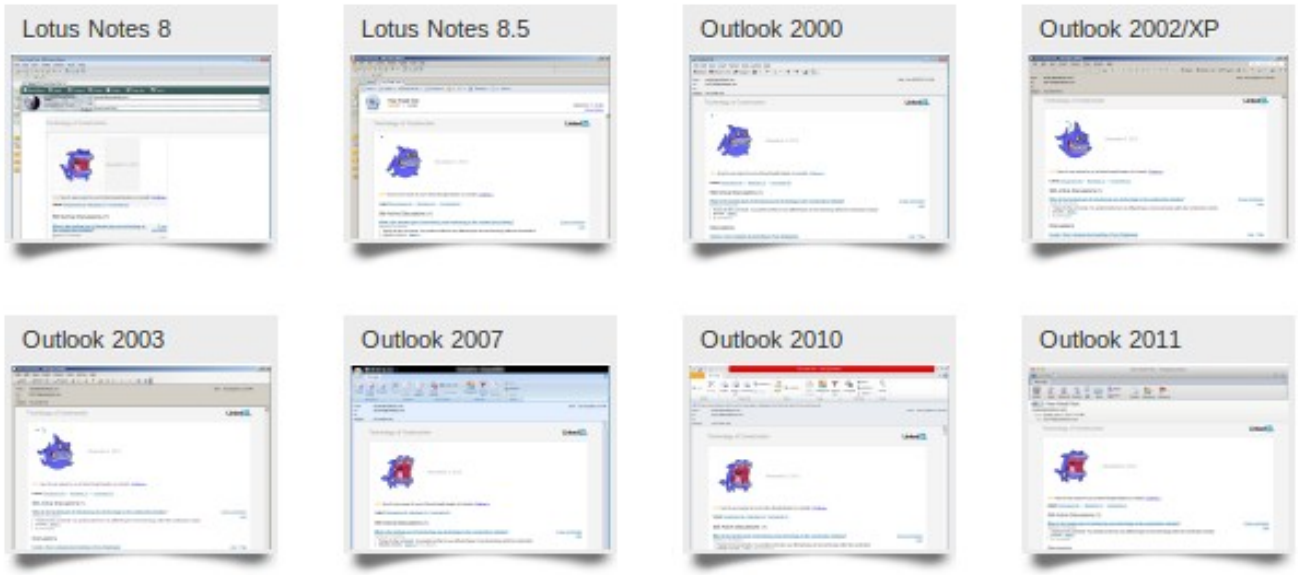


Image representing the first animation test with the running figure.

6.6. Video

Another intriguing subject is the embedding video inside the emails. While a very few number of email clients can render video, all of them can render pictures.

So the main approach to this is to use HTML5 in order to render video but have a picture fall-back that would display in those clients that can not render video. The fall-back picture would work like a link to the actual video.

A HTML5 sample code for embedding videos looks like:

```
<video width="640" height="360"
poster="http://www.yoursite.com/video-image.png" controls="controls">
        <source src="http://www.yoursite.com/yourvideo.mp4"
type="video/mp4" />
        <a href="http://www.yoursite.com/"><img src="
http://www.yoursite.com/video-image.png"
         width="640" height="360" /></a>
</video>
```

Code released on pinpointe.com

According to the study realized by pinpointe.com on 20th March 2013 only Apple email clients are able to display video directly in the email program. The clients that render video properly inside the email are: Apple Mail, Entourage 2008, Outlook Mac 2011, MobileMe and iPhone.

All the other clients would display the fallback picture that links to the actual video. Such an example can be seen in the next picture.

A video preview picture used inside an email; source pinpointe.com

6.7. Email Research Conclusion

In this chapter deeper explanations about the problems encountered in enterprise emails can be found In form of background research and step-by-step guides that would help solving the problems.

This chapter sums up solutions for displaying images, animations and video inside the email content as well as solutions for improving mobile readability of the emails sent these days.

Code samples and statistics results are also provided, as they give a deeper understanding of the reasoning taken for solving some of the problems.

7. Artifacts

Introduction

During the previous email analyses and problem research I also described the causes and the solutions to the problems. Therefore in this section I will sum up a guideline meant to help every developer to make their email design pass unaffected through the inconsistencies found across MUAs.

Guidelines for a good content-rich email

Before beginning to design a content-rich email, one should think about the purpose of the message first, then build a simple email structure to encapsulate the content.

When building a structure for the email, the first thought to have in mind should be the MIME structure (refer to section 4.4.).

Whenever there is a need to embed images, animations, video or any other non-text content, the best approach is to include the content in the email message as in-line content. However a remote fall-back link should be remembered. Refer to sections 6.2., 6.5., 6.6. and 5.6..

When using in-line pictures, one should do a research about the targeted user group then first choose the best approach to embed the pictures in the message body. Refer to sections 6.2. and 6.3..

There is also possible to embed animations or video content. In order to decide which one suits the message best check sections 6.5. and 6.6.. Animations are usually wide supported but videos can only be displayed within Apple targeted email clients.

Every mail developer should think about making their email design mobile-friendly as more and more email readers use mobile devices to read their emails. Refer to section 6.4.

A lot of testing is required during the development. As a developer one should assure that there is always fall-back content to be displayed if something goes wrong during the rendering process.

With a simple code structure, testing and research about the targetted user group there is possible to build good looking cross-client compatible emails even though the email clients developers do not follow a specific standard.

HTML email development is harder than HTML web development, as there are more popular MUAs in use than web browsers, and the technologies used in the HTML mail development are often old, deprecated and limited.

8. Conclusion

As a conclusion to this paper, I would point the key aspects that makes this paper the proper answer to the HTML rendering problem.

During the paper writing, I learned a lot about the email developing, both from theory and practice. I first made sure I know all the relevant theoretical aspects about the technologies in use, then I tried to see how other enterprises approach this problem. I learned both from their strengths and weaknesses.

I tried to find answers not only to the problems I was facing, but also to the problems other enterprises are facing.

As a conclusion I will present a concise answer to my initial question and subquestions:

How can we build professional content-rich emails achieving user agent compatibility?

In order to build emails at enterprise level, one needs to know specifically what the email would contain and who the targeted user group is. Once this is known a series of practices that will assure our email renderability and readability across the different user agents, need to be applied to the email. The practices are summed up in the section 7.

Which types of content is safely displayed across email agents?

It is safe to present styled text, pictures with the exception of background pictures, animations and background colors. Other elements could be displayed as well but they are not rendered similarly in all the email agents. More details can be found In chapter 5 and 6.

How do enterprises handle this issue?

Most of the enterprises handles just a part of the issue, they have most likely done a study of the targeted user group and targeted their emails towards that group. However they should consider to improve their emails with mobile friendliness in mind.

What practices would improve email compatibility between email clients?

There are a series of practices like simplistic structure, simple and safe formatting styles, avoiding background pictures when dealing with enterprise customers, and others. More practices can be found in the section 7.

Is it better to attach graphic components to emails instead of calling them remotely?

Yes it is definitely better but we should not forget to have a remote call as a fail safe, especially because some mobile email clients do not retrieve anything besides text from the email server without the user's request.

What are the main aspects, one should focus on when developing email content?

The MIME Structure, targetted user group and the message itself. Then a proper email HTML design can be created.

9. Bibliography

Every article or resource described here has it's own reference link to the source.
I recommend the use of the digital version of the article or to google the article title.

I do not assume responsibility if one or more of the links die with the time. However the articles might survive and be at just a Google search away.

History of email – article written on nethistory.info

emailology.org – information about what is supported in the popular email clients and guides on how to build html for emails

emailonacid.com – email testing platform and email developer community

litmus.com – email testing platform

email-standards.org – project aiming to achieve a comprehensive, integral standard that email client developers should follow.

HTML email - Wikipedia – information about the HTML email flavor

Email - Wikipedia – information about the email characteristics

RFC2822 – Internet Message Format standard

RFC5322 – The reassessed version of Internet Message Format

HTML Email: Whenever Possible, Turn It Off!

OpenPop.NET – Email Introduction

The History of Notes and Domino

IBM Lotus Notes – Wikipedia

Media Queries for Standard Devices

How to embed video in email – Pinpointe.com

HTML Email: Coding Like It's 1999

Microsoft defends Outlook HTML decision

Animated GIFs in email: A new approach to an old format

Emailology: A CSS Template for Developing HTML Emails in Mobile Devices

Optimizing your email for mobile devices with the @media query

Media queries – Wikipedia

HTML Email and Lotus Notes

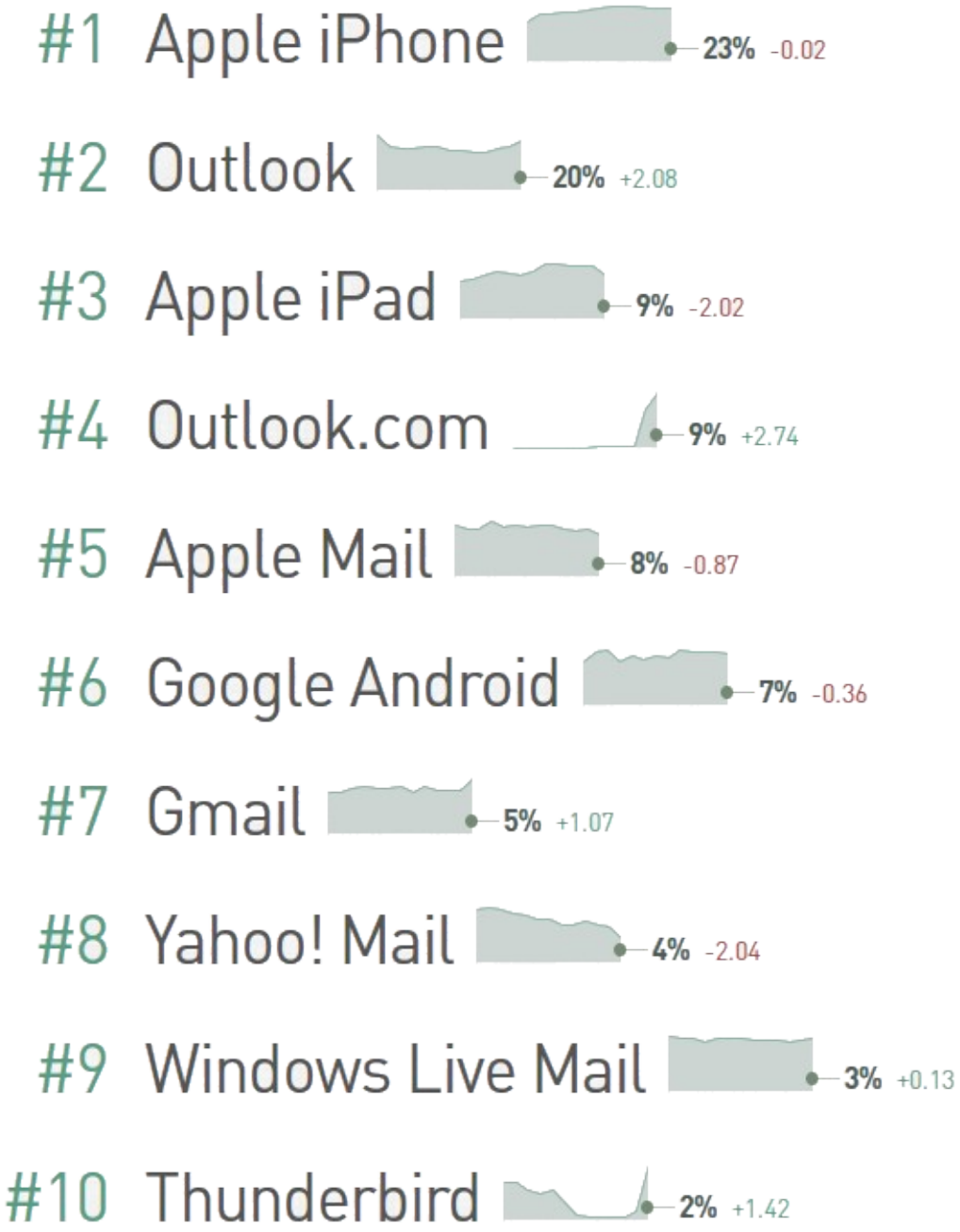
Video in Email – Campaign Monitor

10 Tips for Mobile Friendly Emails

10. Appendices

10.1. Email Client Marked Share – May 2013

The study was made by Litmus LTD and publicized on the emailclientmarketshare.com website. Here is a preview of the results:



10.2. Source Code for Email Research

Because of the length of the email code used, as well as the impracticalities of pasting code on paper, I decided to attach the code used within my researches in digital format only.

Therefore in order to check and inspect the code, maybe even retest it within Litmus or other tools I recommended you to visit the following link and get the digital version of the report as well as the source code used.

The digital version of the report with the referenced code is present at:

data.serbanescu.dk/papers/email-HTML.zip

A few advantages of the digital version of the report:

- presence of intern clickable links
- presence of external clickable links
- presence of the full code used during the research
- hover overview of the foot notes and explanations
- improved readability
- the PDF[1] file is an hybrid PDF and it can be easily edited with free software like LibreOffice

1 PDF – Portable Document Format – a document format that focuses on achieving the same renderability experience across multiple platforms and environments.