

iranphp articles

تابع mail در PHP
عبدالحمید جوکار
joukar@iranphp.net
.....

عنوان مقاله :
نگارنده :
آدرس پست الکترونیک :
تاریخ نگارش :

تابع mail در PHP:

در این مقاله ما خواهیم دید که چگونه زبان PHP را برای ارسال ایمیل تنظیم کنیم و همچنین نحوه فرستادن ایمیل‌های HTML و ایمیل‌های همراه با فایل ضمیمه (Attachment) را بررسی خواهیم کرد. قبل از اینکه به کمک PHP بتوانیم ایمیل بفرستیم باید PHP را برای این کار تنظیم کنیم. دقیقاً مانند اینکه بخواهیم برنامه ارسال و دریافت ایمیل (مانند outlook) را تنظیم کنیم.

برای این کار هم باید سرآغ فایل `php.ini` رفته و آن را با `editor` دلخواه خودتان باز کنید. اگر میخواهید کدهای خودتان را بر روی سروری غیر از سیستم خودتان اجرا کنید از این مرحله صرف نظر کنید و فرض را بر این بگذارید که سرور شما برای انجام این کار تنظیم شده است و در نتیجه به مرحله بعد بروید.

در فایل `php.ini` در قسمتی که با `[mail function]` عنوان‌گذاری شده است گزینه ای دارید به نام SMTP که باید مقدار آن را SMTP ایمیلتان بگذارید مثلاً `mail.softhome.net`

البته در فایل `php.ini` تنظیمات برای سرورهای ویندوز و لینوکس را جدا در نظر گرفته و شما باید بر اساس سیستمی که استفاده میکنید چیزی شبیه زیر را داشته باشید:

برای سیستمهای ویندوز

```
[mail function]
Setup for Windows systems
SMTP = smtp.my.isp.net
sendmail_from = me@myserver.com
```

و برای سیستمهای لینوکس:

```
[mail function]
Setup for Linux systems
sendmail_path = /usr/sbin/sendmail -t
sendmail_from = me@myserver.com
```

وقتی تنظیمات را انجام دادید وب سرور خود را `restart` کنید و اکنون همه چیز برای ارسال ایمیل آماده است!

ارسال ایمیل ساده (Plain Email):

حقیقتاً از روشی که PHP برای ارسال ایمیل در نظر گرفته ساده تر نمی توان تصور کرد! در حقیقت شما می توانید ارسال ایمیل را با تنها نوشتن یک خط انجام دهید! مانند زیر:

```
mail('recipient@some.net', 'Subject', 'Your message here.');
```

خط بالا یک ایمیل را به آدرس `'recipient@some.net'` با موضوع `'Subject'` و `'Your message here.'` به عنوان متن نامه ارسال می کند. همانطور که مشاهده کردید PHP ارسال ایمیل را بسیار ساده کرده است. ولی چندی‌ن راه حل پیشرفته وجود دارد که به ما این امکان را می دهد که ایمیل‌های HTML و ایمیل‌های همراه با فایل ضمیمه بفرستیم.

قبل از هر چیز این نکته را متذکر شوم که اگر `mail system` ی که شما در `php.ini` تعریف کرده اید ایمیل‌رسانی را برگشت (reject) دهد {برای مثال اگر در قسمت To آدرس یک ایمیل درست را ننوشته باشیم} این تابع یک پیغام خطا در مرورگر کاربر نمایش خواهد داد، دقیقاً مانند اتفاقی که در مورد سایر تابعهای PHP می افتد.

اما همانطور که می دانید ما می توانیم با نوشتن علامت @ قبل از تابع از نوشتن پیغام خطا در مرورگر کاربر جلوگیری کنیم.

اگر این نکته را با چیزی که تابع `mail` بر می گرداند (`true` یا `false` بسته به اینکه ایمیل ارسال شده باشد یا خیر) ترکیب کنیم کد زیر را خواهیم داشت:

```
if (@mail($to, $subject, $message)) {
```

```
echo('<p>Mail sent successfully.</p>');
} else {
echo('<p>Mail could not be sent.</p>');
}
```

به یاد داشته باشید که ارسال ایمیل نمی تواند تضمینی بر دریافت آن در مقصد باشد.

برای مثال اگر یک ایمیل به آدرس nonexistent.user@hotmail.com بفرستیم و فرض بر این باشد که این آدرس اصلا وجود ندارد، این آدرس برای تابع mail قابل قبول است و true را بر می گرداند ولی مطمئنا این ایمیل از بین می رود چون کسی صاحب آن نیست، پس در این مورد کاری از دست PHP بر نمی آید.

وقتی که می خواهیم یک ایمیل را به چندین آدرس بفرستیم کافیست که در پارامتر اول تمام آدرس ها را پشت سر هم نوشته و آنها را با علامت کاما "," از هم جدا کنیم. برای مثال:

```
mail('recipient1@some.net, recipient2@some.net',
    'An email to two people', 'Message goes here.');
```

خب، تا حالا اصول فرستادن یک ایمیل را بررسی کردیم، اما پردازیم به اصل مطلب و mail header ها و اینکه چه کارهایی میتوانیم با آنها انجام دهیم! **ایمیل های HTML و header ها:**

اکنون شما میتوانید از اسکریپت های PHP خود ایمیل بفرستید، چقدر جالب! من مطمئنم وقتی یاد بگیرید که چگونه ایمیل های HTML بفرستید احساس قدرت بیشتری خواهید کرد! پس ادامه میدهیم:

برای اینکه ایمیل های HTML را درک کنید ابتدا باید header های یک ایمیل را بشناسید. هر ایمیل دریافتی از دو قسمت تشکیل شده است: header ها و متن نامه (message body). در زیر نمونه یک ایمیل ساده که برنامه ایمیل شما دریافت کرده است را می بینیم:

```
Return-Path: <sender@elsewhere.com>
Delivered-To: you@some.net
Received: ...several lines like this...
From: Sender <sender@elsewhere.com>
To: You <you@some.net>
Subject: A Simple Message
Date: Mon, 11 Feb 2002 16:08:19 -0500
Organization: Sender's Company
X-Mailer: Microsoft Outlook, Build 10.0.2616

Hi there! <tap> <tap> Is this thing on?
```

تمام خطوط بالای خط سفید header ها هستند. در واقع یک ایمیل می تواند بیشتر از اینها هم header داشته باشد ولی برای اختصار در این مثال چند مورد اصلی را ذکر کرده ام.

همانطور که می بینید هر خط از header ها با نام آن header شروع می شود (From:, To:, Subject:, Date:, etc) و در ادامه آنها هم چند مقدار (value) قرار گرفته است. بیشتر header ها استاندارد شده هستند و یک مفهوم خاص برای mail program یا mail server می باشد. مسئول رساندن ایمیل به ما هستند، دارند. اما header های غیر استاندارد هم وجود دارند و مشخصه آنها این است که با X- شروع می شوند (مانند: X-)

Mailer که اغلب برای نشان دادن برنامه ای که برای ارسال ایمیل استفاده شده است به کار می رود (value) یک header نیاز به بیش از یک خط داشته باشد، خطوط اضافه باید با یک فاصله از سر خط شروع شوند. یک مثال در این زمینه را در قسمت بعد خواهیم دید.

وقتی که برنامه ایمیل شما به خط سفید (blank line) رسید می فهمد که header های نامه تمام شده و از این به بعد محتویات متن نامه است که باید نشان داده شود. در مثال ما، متن نامه همان خط آخر است.

تابع mail در PHP به شما اجازه می دهد که headerهای مورد نظر خودتان را به نامه اضافه کنید و PHP آنها را به headerهایی که خود به صورت اتوماتیک تولید می کند اضافه میکند. برای نمونه در مثال پایین یک header با عنوان X-Mailer: به نامه اضافه کرده ایم که PHP 4.x را به عنوان برنامه فرستنده ایمیل معرفی می کند.

```
mail('recipient@some.net', 'Subject', 'Your message here.',  
     'X-Mailer: PHP 4.x');
```

پارامتر چهارم که یک پارامتر اختیاری است اغلب برای نشان دادن From ایمیل استفاده می شود (علاوه بر Fromی که به صورت پیش فرض در php.ini تعریف کرده ایم). پس اجازه بدهید که یک header از نوع From به نامه اضافه کنیم تا این کار را برای ما انجام دهد:

```
mail('recipient@some.net', 'Subject', 'Your message here.',  
     "From: sender@some.net\nX-Mailer: PHP 4.x");
```

با توجه به اینکه headerها هر کدام در یک خط باید قرار داشته باشند پس ما باید هر دو خط را با \n از هم جدا کنیم (که این نیز خود نشان دهنده این است که ما باید پارامتر چهارم را درون " قرار دهیم برای اینکه PHP به کاراکترهای خاص نظیر \n اگر درون " قرار داشته باشند توجه نمی کند). Headerهای دیگری هم هستند که نام فرستنده و گیرنده نامه را قبل از آدرس ایمیل آنها مینویسد؛ به این صورت: name <email>. مثال:

```
mail('recipient@some.net', 'Subject', 'Your message here.',  
     "To: The Receiver <recipient@some.net>\n"  
     "From: The Sender <sender@some.net>\n"  
     "X-Mailer: PHP 4.x");
```

توجه داشته باشید که برای اضافه کردن نام به قسمت To، نمی توانیم نام را در پارامتر اول جا دهیم و تنها راه ممکن این است که یک header با عنوان To: به headerها اضافه کنیم.

Headerهای CC و Bcc: هم وجود دارند که مورد استفاده آنها را حتما خودتان می دانید:

```
mail('recipient@some.net, someone@some.net, metoo@some.net',  
     'Subject', 'Your message here.',  
     "To: The Receiver <recipient@some.net>\n"  
     "From: The Sender <sender@some.net>\n"  
     "cc: Interested <someone@some.net>\n"  
     "Bcc: Me Too <metoo@some.net>\n"  
     "X-Mailer: PHP 4.x");
```

فقط توجه داشته باشید که آدرس ایمیل تمام گیرنده ها به ترتیب To و CC و Bcc در پارامتر اول نوشته شده است، این نکته در جایی ذکر نشده است ولی من با تکیه بر تجربیات شخصی خودم به این نکته پی برده ام که اگر می خواهید ایمیل به تمام گیرنده ها برسد باید این کار را بکنید (مخصوصا در سرورهای ویندوز که زیادی حساس هستند!)

اخطار باگ:

دو باگ برای تابع mail در PHP وجود دارد که من اخیرا در PHP نسخه ۴٫۱٫۰ دیده ام؛ اول اینکه هدر CC: باید اینگونه تایپ شود: CC: یا CC: یعنی هر دو حروف بزرگ یا هر دو کوچک ... ترکیبی از حروف کوچک و بزرگ قاعداً باید کار کند ولی اینطور نیست! دوم اینکه در سرورهای ویندوز هدر Bcc: درست کار نمی کند. همانطور که می دانید هنگام ارسال نامه، هدر Bcc: باید از بین headerها حذف شود، ولی اینگونه نیست و گیرنده ایمیل می تواند هدر Bcc: را در بین هدرها ببیند!

خب حتما سوال می کنید که این همه چه ربطی به فرستادن ایمیل های HTML داشت؟! جواب: چند header خاص هستند که باعث می شوند برنامه دریافت کننده ایمیل آن را به عنوان ایمیل HTML بشناسد.

```
mail('recipient@some.net', 'Subject',  
'<html><body><p>Your <i>message</i> here.</p></body></html>',  
"To: The Receiver <recipient@some.net>\n" .  
"From: The Sender <sender@some.net>\n" .  
"MIME-Version: 1.0\n" .  
"Content-type: text/html; charset=iso-8859-1");
```

به متن نامه که با فرمت HTML نوشته شده و همچنین هدر های Mime-Version: و Content-type: توجه داشته باشید.
هدر Mime-Version: نشان دهنده این است که standard extended mail format مورد استفاده است. (Mime مخفف Multipurpose Internet Mail Extensions می باشد) که به ایمیل اجازه می دهد علاوه بر متن ساده دارای محتویات Content-type هم باشد. و هدر Content-type: مشخص می کند که متن از نوع HTML می باشد.

ترکیب Text و HTML در یک ایمیل (Mixed Messages):

یک ایمیل می تواند شامل ترکیبی از text ساده و html باشد که این باعث می شود ایمیل در بیشتر برنامه های ایمیل قابل دیدن باشد و دیگر شما قدرت html را به خاطر کاربرانی که از برنامه های قدیمی استفاده می کنند قربانی نمی کنید.
توجه داشته باشید که ایمیل های ترکیبی (mixed messages) ضعف های خاص خود را نیز دارا هستند. مثلاً به خاطر اینکه شما دو نسخه از ایمیل را در یک ایمیل ارسال می کنید پس قاعدتاً حجم ایمیل ارسالی از حجمی که باید به طور معمول داشته باشد بیشتر است و این نکته را هم به خاطر داشته باشید که برنامه های ایمیل قدیمی که mixed message را تشخیص نمی دهند ممکن است هر دو نسخه از ایمیل را به صورت فایل های ضمیمه نشان دهند (یکی text و دیگری html).

اجازه دهید نگاهی به یک ایمیل ترکیبی (mixed message) ساده بیندازیم و سپس کد PHP برای ارسال آن را بنویسیم:

```
Date: Mon, 11 Feb 2002 16:08:19 -0500  
To: The Receiver <recipient@some.net>  
From: The Sender <sender@some.net>  
Subject: A simple mixed message  
MIME-Version: 1.0  
Content-Type: multipart/alternative;  
boundary="==Multipart_Boundary_xc75j85x"
```

This is a multi-part message in MIME format.

```
--==Multipart_Boundary_xc75j85x  
Content-Type: text/plain; charset="iso-8859-1"  
Content-Transfer-Encoding: 7bit
```

This is the text portion of the mixed message.

```
--==Multipart_Boundary_xc75j85x  
Content-Type: text/html; charset="iso-8859-1"  
Content-Transfer-Encoding: 7bit
```

```
<html>  
<body>  
<p>This is the <b>HTML portion</b> of the mixed message.</p>  
</body>  
</html>
```

```
--==Multipart_Boundary_xc75j85x--
```

بعد از headerهای اصلی در بالای message، هدر MIME-Version: 1.0 را داریم. Headerی که ما را قادر می سازد جنبه های پیشرفته تر ایمیل را بسازیم. هدر Content-type: جایی است که اصل ماجرا شروع می شود:

```
Content-Type: multipart/alternative;
boundary=="Multipart_Boundary_xc75885"
```

Header: Content-type: را برابر با multipart/alternative قرار داده ایم که یک نمونه خاص است و به ما این قدرت را می دهد که در ایمیل دو یا چند فرمت مختلف را داشته باشیم (تا برنامه ایمیل کاربر مناسب ترین آن را انتخاب کند و نمایش دهد).

به علاوه اینکه ما از Content-Type برای set یک رسته boundary استفاده کرده ایم. برای اینکه خطوط header ها کوتاه باشد این قسمت از header: Content-type: را در خط دوم قرار داده ایم و همانطور که قبلا اشاره کردم باید با مقداری فاصله از سر خط آن را بنویسیم تا مشخص شود که ادامه header قبلی است. در این مثال من از "=="Multipart_Boundary_xc75j85x" استفاده کرده ام.

این رسته معنا و مفهوم خاصی ندارد (boundary در لغت به معنای مرز و سرحد می باشد و در تابع mail مرز هر قسمت از ایمیل را مشخص می کند مثلا اینکه از کجا تا کجا مربوط به قسمت text ایمیل است و از کجا تا کجا مربوط به قسمت html) من از کاراکترهایی مانند علامت مساوی و underline و یک رسته تصادفی متشکل از اعداد و حروف برای درست کردن رسته boundary استفاده کرده ام و در آخر ما از این رسته برای تقسیم کردن message به چند قسمت استفاده می کنیم.

جمله "This is a multi-part message in MIME format." هم برای کاربران برنامه های ایمیل قدیمی آورده شده است تا اگر احیانا ایمیل در برنامه آنها درست نمایش داده نمی شود دلیل آن را بدانند! پس از آن رسته boundary آمده است که شروع قسمت اول ایمیل را اعلام می کند:

```
--==Multipart_Boundary_xc75j85x
Content-Type: text/plain; charset="iso-8859-1"
Content-Transfer-Encoding: 7bit
```

```
This is the text portion of the mixed message.
```

حتما توجه کنید که وقتی می خواهیم رسته boundary را در کد بگذاریم باید با دو علامت dash پشت سر هم (--) شروع آن را اعلام کنیم. پس از boundary اول، قسمت text ایمیل را می نویسیم. هر قسمت از message با یک زوج header همراه است که Content-Type و Encoding آن را مشخص می کنند. در قسمت text مثال ما Content-Type برابر با text/plain (و با charset استاندارد iso-8859-1) و Encoding این قسمت 7bit می باشد (plain ASCII text) خط سفید نشان دهنده پایان header ها می باشد و در ادامه متن نامه آمده است. پس از همه اینها قسمت html شروع می شود:

```
--==Multipart_Boundary_xc75j85x
Content-Type: text/html; charset="iso-8859-1"
Content-Transfer-Encoding: 7bit
```

```
<html>
<body>
<p>This is the <b>HTML portion</b> of the mixed message.</p>
</body>
</html>
```

header ها در این قسمت نیز مانند قسمت text ایمیل هستند به جز در مورد Content-Type که text/html ذکر شده است. پس از متن (body) نامه که به زبان html نوشته شده است نوبت به بستن boundary می رسد:

```
--==Multipart_Boundary_xc75j85x--
```

(بستن boundary هم مانند شروع آن با دو علامت dash پشت سر هم مشخص می شود)

همانطور که می بینید ایمیل های ترکیبی (ترکیب text و html) پیچیده به نظر می رسند ولی در حقیقت این طور نیست و با یک نگاه عمیق تر به سادگی آنها پی می بریم. تنها در قسمت تولید رسته boundary است که باید کمی مهارت به خرج دهیم، من از این روش برای تولید رسته boundary استفاده می کنم:

```
$semi_rand = md5(time());
```

```
$mime_boundary = "==Multipart_Boundary_x{$semi_rand}x";
```

در حقیقت `unix timestamp` کنونی سیستم را با الگوریتم MD5 تبدیل به یک رشته شبه تصادفی کرده ایم. از این رشته هم برای استفاده در رشته `boundary` استفاده می کنیم.

با در نظر گرفتن تمام اینها شما باید بتوانید یک `mixed message` را با استفاده از PHP ارسال کنید.

به طریقه دو نیم کردن `message` با استفاده از رشته `boundary` خوب دقت کنید که در قسمت ضمیمه کردن فایل هم به آن احتیاج خواهیم داشت.

ضمیمه کردن فایل (File Attachments)

ارسال فایل ضمیمه همراه ایمیل هم تقریباً شبیه ارسال `mixed message` می باشد به جز در مورد `Content-Type` که برای کل `message` در نظر گرفته شده است (`multipart/mixed` به جای `multipart/alternative`).

یادآوری: منظور از "`Content-Type`" که برای کل `message` در نظر گرفته شده است همان `Content-Type` است که در پارامتر چهارم تابع `mail` قرار داده ایم و همانطور که دانستید یک ایمیل می تواند چندین `Content-Type` مختلف داشته باشد (قسمت های `text` و `html` و `attachment` هر کدام `Content-Type` خاص خود را دارند و تابع `mail` هم `Content-Type` کلی برای همه آنها در نظر می گیرد که در اینجا `multipart/mixed` است)

یک `header` جدید به نام `Content-Disposition` هم داریم که به برنامه ایمیل کاربر می گوید چطور با آن قسمت از ایمیل برخورد کند، مثلاً برای قسمت `attachment` این را داریم:

```
Content-Disposition: attachment
```

اجازه دهید که یک `form` را طراحی کنیم که پس از `submit` ایمیل را همراه با فایل ضمیمه (در صورت وجود) به مقصد ارسال کند. من خط به خط این اسکریپت را توضیح خواهم داد، در نتیجه در پایان شما علاوه بر داشتن یک تکه کد به درد بخور، طریقه کارکرد آن را هم به خوبی فرا خواهید گرفت:

کد را می توانید از آدرس زیر دریافت نمایید:

<http://www.webmasterbase.com/examples/phpemail/phpemail.zip>

ابتدا مقادیر `submit` شده را در متغیرهایی قرار میدهیم (فرض بر این است که `register_globals=off` می باشد)

```
// Read POST request params into global vars
$to      = $_POST['to'];
$from    = $_POST['from'];
$subject = $_POST['subject'];
$message = $_POST['message'];
```

فایلهایی را که `upload` می کنیم در یک آرایه خاص به نام `$_FILES` قرار می گیرند و در نتیجه ما به راحتی می توانیم مقادیر مورد نیاز را از آن بگیریم.

```
// Obtain file upload vars
$fileatt = $_FILES['fileatt']['tmp_name'];
$fileatt_type = $_FILES['fileatt']['type'];
$fileatt_name = $_FILES['fileatt']['name'];
```

برای مختصر شدن مقاله فرض میکنیم که پارامترهای `$to` و `$from` دارای مقادیری `valid` هستند (آدرس ایمیل) و ما آنها را چک نمی کنیم (به طور عادی آنها باید با `regular expressions` چک شوند).

در مرحله بعد `header` ایمیل را کامل می کنیم، برای شروع مقدار `from` را در آن قرار می دهیم:

```
$headers = "From: $from";
```

(این \$header در پارامتر چهارم تابع mail قرار می گیرد)

سپس متغیر \$fileatt را چک می کنیم که آیا path و نام فایل upload شده را در خود دارد یا خیر. برای این کار دستوری داریم به نام is_uploaded_file و از آن استفاده می کنیم:

```
if (is_uploaded_file($fileatt)) {
// Read the file to be attached ('rb' = read binary)
$file = fopen($fileatt, 'rb');
$data = fread($file, filesize($fileatt));
fclose($file);
```

در این قسمت محتویات و مشخصات فایل را در متغیر \$data قرار داده ایم.

الان باید header های ایمیل را جوری تنظیم کنیم تا بتواند نامه های multipart/mixed بفرستد:

```
// Generate a boundary string
$semi_rand = md5(time());
$mime_boundary = "==Multipart_Boundary_x{$semi_rand}x";

// Add the headers for a file attachment
$headers .= "nMIME-Version: 1.0\n" .
"Content-Type: multipart/mixed;\n" .
" boundary={$mime_boundary}";
```

و حالا می رسیم به متن نامه:

این دقیقاً همان قسمت text در مبحث قبلی می باشد:

```
// Add a multipart boundary above the plain message
$message = "This is a multi-part message in MIME format.\n\n" .
"--{$mime_boundary}\n" .
"Content-Type: text/plain; charset=\"iso-8859-1\"\n" .
"Content-Transfer-Encoding: 7bit\n\n" .
$message . "\n\n";
```

و اکنون با استفاده از روش Base64 فایل ضمیمه را به باینری تبدیل می کنیم (مناسب برای ارسال با ایمیل). تمامی برنامه های معروف ایمیل از روش Base64 encoding پشتیبانی می کنند، پس ما هم از این روش استفاده می کنیم. خوشبختانه PHP هم یک تابع برای encoding Base64 در نظر گرفته است:

```
// Base64 encode the file data
$data = chunk_split(base64_encode($data));
```

ما الان تمام چیزهایی که برای ضمیمه کردن فایل لازم بود را داریم. این هم از کد:

```
// Add file attachment to the message
$message .= "--{$mime_boundary}\n" .
"Content-Type: {$fileatt_type};\n" .
" name={$fileatt_name}\n" .
"Content-Disposition: attachment;\n" .
" filename={$fileatt_name}\n" .
"Content-Transfer-Encoding: base64\n\n" .
$data . "\n\n" .
"--{$mime_boundary}--\n";
```

این قسمت از کد تمام تغییرات و توضیحاتی را که لازم بود تا فایل را به عنوان attachment در نامه جا دهیم اعمال می کند.

اکنون باید با استفاده از تابع mail نامه را بفرستیم:

```
// Send the message
$ok = @mail($to, $subject, $message, $headers);
if ($ok) {
```

```
echo "<p>Mail sent! Yay PHP!</p>";  
} else {  
echo "<p>Mail could not be sent. Sorry!</p>";  
}
```

و این بود تمام چیزی که برای ارسال ایمیل باید می دانستیم!

خلاصه :

در این مقاله به احتمال زیاد چیزهای بی را درباره ایمیل آموختید که تا کنون شاید نمی دانستید. دانستیم که چگونه با تابع قدرتمند mail کارهای پیشرفته انجام دهیم.

آموختیم که :

چگونه header های ایمیل را تنظیم کنیم :

ایمیل های html بفرستیم :

هر دو فرمت html و plain text را در یک ایمیل قرار دهیم.

و سرانجام

با اضافه کردن چند تکنیک جدید به همه اینها توانستیم یک فایل را ضمیمه ایمیل کنیم.

اگر آدم پر حوصله ای هستید تلاش کنید تا این کد را برای ضمیمه کردن چند فایل در یک نامه گسترش دهید و یا متن های html را پشتیبانی کنید. و یا اگر یک برنامه نویسی گرا هستید تلاش کنید تا یک کلاس بسازید که تمام تابع هایی که ما به کار بردیم را encapsulate کند.

اگر واقعا علاقه مند شده اید RFC for MIME extensions را چک کنید تا از تمام قابلیت هایی که ایمیل دارد مطلع شوید. (RFC 2045)