

# Solução de problemas

SKmedix editou esta página 24 days ago · 55 revisões

## Solução de problemas de PHPMailer

Seja qual for o problema que você está tendo, primeiro **certifique-se de que está usando o PHPMailer mais recente**. Se você baseou seu código em um exemplo, você encontrou algum lugar além do aqui no GitHub, é muito provavelmente desatualizado - baseie seu código nos exemplos na [pasta de exemplos](#). Cerca de 90% das [perguntas sobre o estouro de pilha](#) fazem esse erro.

## Carregando aulas

### Usando compositor

O [Composer](#) economiza uma grande quantidade de trabalho - manipulando dependências de pacotes, atualizações e download, e gera um autocarregador agradável para que você não tenha que fazer isso você mesmo. **Carregando através do compositor é o método preferido de usar o PHPMailer em seu projeto**. Tudo o que você precisa fazer é exigir o autoloader do compositor:

```
require './vendor/autoload.php';
```

É particularmente importante se você estiver usando a autenticação XOAUTH2, pois exige classes dependentes satisfeitas pelo compositor. As dependências não estão incluídas por padrão, porque elas não são necessárias por todos e não funcionam nas versões anteriores do PHP que o PHPMailer oferece, então você as encontrará na seção "sugerir" do `composer.json` arquivo do PHPMailer. Você deve copiar essas dependências para o seu próprio `composer.json`'s `require` seção, em seguida, `composer update` carregá-los e adicioná-los ao seu carregador automático.

Se você não fizer isso, é provável que você veja erros como este:

```
Erro fatal : Class 'League \ OAuth2 \ Client \ Provider \ Google ' não encontrado no
```

Para corrigir isso, configure o compositor como descrito ou baixe essa classe e todas as suas dependências e carregue-as manualmente.

### Usando o autocarregador do PHPMailer

Não há muito tempo, o PHPMailer mudou a forma como carregou as classes para que fosse mais compatível com o compositor, muitos frameworks e o [padrão de autoloading PHP PSR-0](#). Observe que, porque apoiamos o PHP de volta à versão 5.0, não podemos suportar o [padrão de PSR-4](#) mais recente, nem podemos usar espaços para nome. Anteriormente, o PHPMailer carregava a classe SMTP explicitamente, e isso causa problemas se você deseja fornecer sua própria implementação. Você pode ter visto scripts antigos fazendo isso:

```
require 'class.phpmailer.php';
```

Se você fizer isso apenas, o **envio de SMTP falhará** com um `Class 'SMTP' not found` erro. Você precisa incluir explicitamente o `class.smtp.php` arquivo (leia o README para obter informações sobre quais arquivos você precisa), ou use as abordagens recomendadas para usar o compositor ou o carregador automático fornecido, como este:

```
require 'PHPMailerAutoload.php';
```

▼ Páginas 10

[Casa](#)

[Sobre as vulnerabilidades CVE 2016 10033 e CVE 2016 10045](#)

[Gmail XOAUTH2 Usando o Google API Client](#)

[Substituindo a classe SMTP](#)

[Enviando listas](#)

[Depuração de SMTP](#)

[Solução de problemas](#)

[Resolução de problemas disponíveis](#)


[Tutorial](#)

[Usando o Gmail com XOAUTH2](#)

Clone este wiki localmente

<https://github.com/PHPMailer/PHPMailer/wiki>



 Clone no Desktop

## Habilitando a saída de depuração

Se você estiver usando SMTP (ou seja, você está ligando `isSMTP()`), você pode obter uma transcrição detalhada da conversa SMTP usando a `SMTPDebug` propriedade. As configurações são as seguintes:

- 1: mostrar cliente -> mensagens do servidor apenas. Não use isso - é muito pouco provável que lhe diga algo útil.
- 2: mostrar cliente -> servidor e servidor -> mensagens do cliente - esta geralmente é a configuração desejada
- 3: como 2, mas também mostra detalhes sobre a conexão inicial; use apenas isso se você tiver problemas para se conectar (por exemplo, a conexão com o tempo limite)
- 4: Como 3, mas também mostra tráfego detalhado detalhado. Só é realmente útil para analisar bugs de nível de protocolo, muito detalhados, provavelmente não é o que você precisa.

Defina esta opção, incluindo uma linha como essa no seu script:

```
$mail->SMTPDebug = 2;
```

O formato de saída se adaptará à saída de linha de comando ou HTML, embora você possa substituir isso usando a `Debugoutput` propriedade.

## "Erro SMTP: Não foi possível conectar-se ao host SMTP".

Isso também pode aparecer como **SMTP connect() failed** ou **Called Mail() without being connected** em saída de depuração. Isso geralmente é relatado como um problema de PHPMailer, mas é quase sempre uma falha de DNS local, bloqueio de firewall (por exemplo, como o GoDaddy faz) ou outro problema em sua rede local. Isso significa que o PHPMailer não consegue entrar em contato com o servidor SMTP que você especificou na `Host` propriedade, mas não diz exatamente por quê. Também pode ser causado por não ter a `openssl` extensão carregada (veja as notas de criptografia abaixo).

Algumas técnicas para diagnosticar a origem desse erro são discutidas abaixo.

### Vai Papai

O popular fornecedor de hospedagem dos EUA, GoDaddy, impõe restrições muito rígidas (ao ponto de se tornar quase inútil) ao enviar um e-mail. Eles bloqueiam SMTP de saída para portas 25, 465 e 587 para todos os servidores, exceto os seus. Este problema é o assunto de muitas [perguntas](#) frustrantes [no estouro de pilha](#). Se você encontrar seu script funcionar em sua máquina local, mas não quando você carregá-lo para GoDaddy, isso será o que está acontecendo com você. A solução é extremamente mal documentada pelo GoDaddy: você **deve** enviar através de seus servidores e também desativar todos os recursos de segurança, nome de usuário e senha (ótimo, hein?!), fornecendo esta configuração para PHPMailer:

```
$ mail -> isSMTP ();  
$ mail -> Host = ' relay-hosting.secureserver.net ' ;  
$ mail -> Port = 25 ;  
$ mail -> SMTPAuth = false ;  
$ mail -> SMTPSecure = false ;
```

GoDaddy também se recusa a enviar com um `From` endereço pertencente a qualquer domínio aol, gmail, yahoo, **hotmail**, live, objetivo ou msn (veja [seus documentos](#)). Isso ocorre porque todos esses domínios implementam medidas SPF e DKIM anti-falsificação, e falsificar seu endereço é falsificação.

Você pode achar mais fácil mudar para um provedor de hospedagem mais esclarecido.

## Leia a transcrição SMTP

Se você definir `SMTPDebug = 2` ou mais alto, você verá o que o servidor SMTP remoto diz. Muitas vezes, isso lhe dirá exatamente o que está errado - coisas como "Senha incorreta", ou às vezes um URL de uma página para ajudá-lo a diagnosticar o problema. **Leia o que diz** . O Google faz muito isso - veja abaixo informações sobre a configuração "Permitir aplicativos menos seguros".

## Falhas de DNS

---

Estes são frequentemente vistos como tempos limite de conexão, ou "Falha temporária na resolução de nomes", "não foi possível resolver o host", "falha no getaddrinfo" ou erros similares. Verifique se o seu DNS está funcionando usando a `dig` ferramenta (do `dnsutils` pacote no Debian / Ubuntu):

```
dig + short smtp.gmail.com
```

Você receberá algo assim se seu DNS estiver funcionando:

```
gmail-smtp-msa.1.google.com.  
173.194.67.108  
173.194.67.109
```

Se isso falhar, o PHPMailer não poderá enviar e-mails porque não poderá obter o endereço IP correto para se conectar. Se, talvez, você não tenha um nome no DNS, você pode usar um endereço IP diretamente como o nome do host. Para corrigir isso, você precisa descobrir por que seu DNS não está funcionando - talvez você não tenha configurado seus resolvers?

## Verifique que esteja lá

---

Mesmo um servidor com todos os serviços desativados geralmente responderá a pings simples, então se você souber que seu DNS está bem, verifique se o servidor está realmente lá:

```
ping smtp.gmail.com
```

Você deve ver algo assim (pressione ctrl-C para pará-lo):

```
PING gmail-smtp-msa.1.google.com (74.125.133.108): 56 data bytes  
64 bytes from 74.125.133.108: icmp_seq=0 ttl=43 time=72.636 ms  
64 bytes from 74.125.133.108: icmp_seq=1 ttl=43 time=68.841 ms  
64 bytes from 74.125.133.108: icmp_seq=2 ttl=43 time=68.500 ms
```

## Verifique se é um servidor de correio

---

Pode ser que algum outro serviço esteja funcionando na porta SMTP para a qual você está tentando se conectar. Você pode verificar isso usando a `telnet` ferramenta, assim (conectando-se ao gmail na porta do serviço de envio):

```
telnet smtp.gmail.com 587
```

Isso deve dar-lhe algo como isto:

```
Trying 173.194.67.109...  
Connected to gmail-smtp-msa.1.google.com.  
Escape character is '^['.  
220 mx.google.com ESMTP ex2sm16805587wj.30 - gsmtip
```

(Digite `quit` para sair disso). Se a porta 587 não funcionar, você pode tentar a porta 465 ou a porta 25 e usar o que funciona - embora tenha em mente que a porta 25 geralmente não suporta criptografia (ver notas de criptografia).

Se não produzir saída ou algo que não comece `220` , então o seu servidor está desligado ou você obteve o servidor errado.

## Redirecionamento de firewall

---

Outra coisa a procurar aqui é que o nome com o qual o servidor de correio responde deve estar relacionado ao servidor que você solicitou, como você pode ver no exemplo acima - nós pedimos `smtp.gmail.com` e obtivemos `gmail-smtp-mla.1.google.com`, o que parece ser algo com o google - Se, em vez disso, você ver algo como o nome do seu ISP, isso poderia significar que o firewall do seu ISP está redirecionando você de forma transparente para seus próprios servidores de e-mail, e você provavelmente verá falhas de autenticação porque você está logando no servidor errado. É provável que isso aconteça na porta 25, mas é menos provável que aconteça nas portas 465 e 587, então é mais uma razão para usar criptografia!

## Bloqueio SELinux

---

Se você SMTP -> ERROR: Failed to connect to server: Permission denied (13) vir um erro como , você pode estar executando o SELinux impedindo que o PHP ou o servidor web enviem e-mails. Isto é particularmente provável no RedHat / Fedora / Centos. Usando o `getsebool` comando, podemos verificar se o daemon httpd está autorizado a fazer uma conexão pela rede e enviar um email:

```
getsebool httpd_can_sendmail
getsebool httpd_can_network_connect
```

Este comando retornará um booleano ativado ou desativado. Se estiver desligado, podemos ativá-lo:

```
sudo setsebool -P httpd_can_sendmail 1
sudo setsebool -P httpd_can_network_connect 1
```

Se você estiver executando o PHP-FPM via fastcgi, talvez seja necessário aplicar isso ao daemon fpm em vez de httpd.

## Bloqueio IPv6

---

Alguns provedores de serviços (incluindo o Digital Ocean) fornecem conectividade IPv6 para servidores, mas bloqueiam SMTP de saída em IPv6 enquanto o permitem no IPv4. Isso pode ser trabalhado ao configurar a `Host` propriedade diretamente em um endereço IPv4 (a `gethostbyname` função somente faz pesquisas IPv4):

```
$mail->Host = gethostbyname('smtp.gmail.com');
```

O único problema com esta abordagem é que você acaba pedindo para se conectar a um endereço IPv4 explícito, o que normalmente irá fazer com que você falhe as verificações do nome do certificado. Você pode desativar isso (veja `SMTPOptions` em outro lugar neste documento), mas isso deve ser considerado uma solução ruim - a solução certa é consertar sua rede.

Nota: Ao usar o serviço Digital Ocean, verifique se a sua porta SMTP está realmente desbloqueada, pois é uma empresa com base nos EUA que contém uma série de diretrizes para não cair no espaço, então você deve pedir o desbloqueio e seguir as etapas para confirmar com o Digital Ocean the Purpose de enviar seus e-mails com o PhpMailer.

## Falhas de autenticação

---

Se sua autenticação está falhando, existem várias causas prováveis:

- Você tem o nome de usuário ou a senha errados
- Sua conexão está sendo desviada para um servidor diferente (como acima)
- You have specified authentication without encryption

Generally, you do not want to send a username or password over an unencrypted link. Some SMTP authentication schemes do add a minimal level of security (sending short hashes rather than clear text), but these provide only minimal protection, and so most servers do not allow authentication without encryption. Fix this by setting `SMTPSecure = 'tls'` and `Port = 587` as well as setting the `Username` and `Password` properties.

## Gmail, OAuth2 and "Allow less secure apps"

From December 2014, Google started imposing an authentication mechanism called [XOAUTH2](#) based on [OAuth2](#) for access to their apps, including Gmail. This change can break both SMTP and IMAP access to Gmail, and you may receive authentication failures (often "5.7.14 Please log in via your web browser") from many email clients, including PHPMailer, Apple Mail, Outlook, Thunderbird and others. The error output may include a link to <https://support.google.com/mail/bin/answer.py?answer=78754>, which gives a list of possible remedies. There are two main solutions to this in PHPMailer:

- Enabling "[Allow less secure apps](#)" will usually solve the problem for PHPMailer, and it does not really make your app significantly less secure. Reportedly, changing this setting may take an hour or more to take effect, so don't expect an immediate fix.
- PHPMailer added support for XOAUTH2 in version 5.2.11, though **you must be running PHP 5.5 or later** in order to use it. Documentation on how to set it up can be found on [this wiki page](#).

## Using encryption

There's no doubt that you should use encryption at every opportunity, otherwise you're inviting all kinds of unpleasant possibilities for phishing, identity theft etc.

To use any kind of encryption you need the [openssl PHP extension](#) enabled. If you don't have it installed, or it's misconfigured, you're likely to have trouble at the `STARTTLS` phase of connections. Check this by looking at the output of `phpinfo()` or `php -i` (look for an 'openssl' section), or `openssl` listed in the output of `php -m`, or run this line of code:

```
<?php echo (extension_loaded('openssl'))?'SSL loaded':'SSL not loaded')."\n"; ?>
```

As for what kind to use, the answer is generally simple: Don't use SSL on port 465, it's been [deprecated since 1998](#) and is only used by Microsoft products that didn't get the memo; use TLS on port 587 instead:

```
$mail->SMTPSecure = 'tls';
$mail->Host = 'smtp.gmail.com';
$mail->Port = 587;
```

or more succinctly:

```
$mail->Host = 'tls://smtp.gmail.com:587';
```

Don't mix up these modes either; valid combinations are `tls` on port 587 (or possibly 25) and `ssl` on port 465. `ssl` on port 587 or `tls` on port 465 *will not work*.

## Opportunistic TLS

PHPMailer 5.2.10 introduced opportunistic TLS - if it sees that the server is advertising TLS encryption (after you have connected to the server), it enables encryption automatically, even if you have not set `SMTPSecure`. This *might* cause issues if the server is advertising TLS with an invalid certificate, but you can turn it off with `$mail->SMTPAutoTLS = false;`.

## PHP 5.6 certificate verification failure

In a change from earlier versions, PHP 5.6 verifies certificates on SSL connections. If the SSL config of the server you are connecting to is not correct, you will get an error like this:

```
Warning: stream_socket_enable_crypto(): SSL operation failed with code 1.
OpenSSL Error messages: error:14090086:SSL
routines:SSL3_GET_SERVER_CERTIFICATE:certificate verify failed
```

The correct fix for this is to replace the invalid, misconfigured or self-signed certificate with a good one. Failing that, you can allow **insecure** connections via the `SMTPOptions` property introduced in PHPMailer 5.2.10 (it's possible to do this by [subclassing the SMTP class](#) in earlier versions), though this is not recommended:

```
$mail->SMTPOptions = array(
    'ssl' => array(
        'verify_peer' => false,
        'verify_peer_name' => false,
        'allow_self_signed' => true
    )
);
```

You can also change these settings globally in your `php.ini`, but that's a **really** bad idea; PHP 5.6 made this change for very good reasons.

Sometimes this behavior is not quite so apparent; sometimes encryption failures may appear as the client issuing a `QUIT` immediately after trying to do a `STARTTLS`. If you see that happen, you should check the state of your certificates or verification settings.

## cURL error 60

You may see the error `cURL error 60: SSL certificate problem: unable to get local issuer certificate`. This may be because your CA file is out of date or missing. You can [download the latest CA cert file from curl](#), install it somewhere accessible and point at it from your `php.ini` file with the `openssl.cafile` and `curl.cainfo` properties.

This error can also be caused if your PHP is using a libcurl compiled with `libressl` (a common option on homebrew) which has [a bug relating to this](#) instead of the default `openssl` or OS X's built-in Secure Transport - running `curl -v` will tell you what yours is compiled with, like this:

```
curl 7.48.0 (x86_64-apple-darwin15.4.0) libcurl/7.48.0 OpenSSL/1.0.2g zlib/1.2.5
libssh2/1.7.0 nghttp2/1.9.2
```

A standard OS X installation will use Secure Transport:

```
curl 7.43.0 (x86_64-apple-darwin15.0) libcurl/7.43.0 SecureTransport zlib/1.2.5
```

## Testing SSL outside PHP

In order to eliminate PHP config or your code from encryption issues, you can use your local `openssl` installation to test the config directly using its built-in SMTP client, for example:

```
openssl s_client -starttls smtp -crlf -connect smtp.gmail.com:587
```

You should expect a response like this:

```
CONNECTED(00000003)
depth=2 /C=US/O=GeoTrust Inc./CN=GeoTrust Global CA
verify error:num=20:unable to get local issuer certificate
verify return:0
---
Certificate chain
 0 s:/C=US/ST=California/L=Mountain View/O=Google Inc/CN=smtp.gmail.com
  i:/C=US/O=Google Inc/CN=Google Internet Authority G2
 1 s:/C=US/O=Google Inc/CN=Google Internet Authority G2
  i:/C=US/O=GeoTrust Inc./CN=GeoTrust Global CA
 2 s:/C=US/O=GeoTrust Inc./CN=GeoTrust Global CA
  i:/C=US/O=Equifax/OU=Equifax Secure Certificate Authority
```

```

---
Server certificate
-----BEGIN CERTIFICATE-----
MIIEgDCCA2igAwIBAgIIQKPDG0sroxQwDQYJKoZIhvcNAQELBQAwSTELMAkGA1UE
BhMCVVMwEzARBgNVBAoTCkdvb2dsZSBJbmMxJTAjBgNVBAMTHEdvd2dsZSBJbnRl
cm5ldCBBDXRob3JpdHkgRzIwHhcNMTYwNDA3MDkwMzU5WhcNMTYwNjMwMDgyMDAw
WjBoMQswCQYDVQQGEWJVUzETMBEGA1UECAwKQ2FsaWZvcn5pYTEwMBQGA1UEBwwN
TW91bnRhaw4gVmlldzETMBEGA1UECgwKR29vZ2x1IEluYzEXMBUGA1UEAwOc210
cC5nbWpCbC5jb20wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQNsdHDL
zDdAFIunNFHuvBgE3ri8CinYqrPwh8LPhxNo7gnIxSOIASz1Ba1xm4uBpInsWJLK
RxcqjQfGRRki558+ed5L2TrX3uoznEGAsoptatSPDuDaSttHjKX6Z0jbBEAxHp4r
ozp1RFucGma7WkF1XR7htjdoFwFCVN/u++Bhp1vJw09RY2iwywjIVGqY4V9hYGHo
00PKJ3SHTkIPHKZS1hSum5f2P197cKrQVrFYx2dDMowJ1Cq8eEf1sp+38UXQEfqjo
BYNAj29ihimwVC/bN+6gZcn+vsR2w77p8tkLLzqY/vZ67Una6Qa+eV4B15KmwWk
D1C1MDdoHTI8HD19AgMBAAGjggFLMIIBRzAdBgNVHSUEFjAUBgggBgEFBQcDAQYI
KwYBBQUHAWIwGQYDVDR0RBBiWEII0c210cC5nbWpCbC5jb20waAYIKwYBBQUHAQEE
XDBaMCsGCCsGAQUFBzAChh9odHRwOi8vcGtpLmdvb2dsZS5jb20vR0RBRzIuY3J0
MCsGCCsGAQUFBzABHh9odHRwOi8vY2xpZW50czEuZ29vZ2x1LmNvbS9vY3NwMB0G
A1UdDgQWBBSEsJM0ANFUgqu5Qc3/VU+g11U0UjAMBGNVHRMBAf8EAJAAMB8GA1Ud
IwQYMBaAFerdBhYbvpZotXb1gba7Yhq6WoEvMCEGA1UdIAQaMBGwDAYKKwYBBAHW
eQIFATAIBgZngQwBAGIwMAYDVR0fBCKwJzAloCOgIYYfaHR0cDovL3BraS5nb29n
bGUuY29tL0dJQucylmNybdANBgkqhkiG9w0BAQsFAA0CAQEAhttyyIAjATMjXG03
klGoKwHAZQ4ViSe2pt/DEMDUJNXBfJ+v6SI9wBE3QRH26P/m5LkwoBe0rSiaNsiw
CrSZiBGFAj6/0BUUciHIPc/dKMYRFZ61wParXD0VFJBtCV7cBSVvU3aw0YMPoufR
8Utlj0aTnm7pLqViGRy65EUwztznVe7eIi91X3pKPjg+TkoJmsbRes1ySmeQ06LV
1cWgd2HMOap0HK+cyQP2Uuo4ZAo5Hgiy9nnDRMmvShT2dKbIv19JyrfXPGuz/E7I
6z/Z/Fi7i1Srrpx/Frd8XwRCNQJPWfd2cV6NqGLwNR2qSCA0gJaWdIvJYqITw01L
cAh6QQ==
-----END CERTIFICATE-----
subject=/C=US/ST=California/L=Mountain View/O=Google Inc/CN=smtp.gmail.com
issuer=/C=US/O=Google Inc/CN=Google Internet Authority G2
---
No client certificate CA names sent
---
SSL handshake has read 3494 bytes and written 491 bytes
---
New, TLSv1/SSLv3, Cipher is AES128-SHA
Server public key is 2048 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol : TLSv1
    Cipher : AES128-SHA
    Session-ID: 936F1A0663F5CE73943C00650C2FB2B9612E1F9819D38A7CD853DB9130D0E5EE
    Session-ID-ctx:
    Master-Key:
C092C10C71219E0BE8358CD432120D94CA39B01EDDA8A7007B08D7E86A74B6A16B14345610255063E1B0A2DB55D0E

    Key-Arg : None
    Start Time: 1460541074
    Timeout : 300 (sec)
    Verify return code: 0 (ok)
---
250 SMTPUTF8

```

(just type "QUIT" to get out of that). Notice that the verify return code is 0, which indicates successful verification. The `verify error:num=20:unable to get local issuer certificate` is not a problem. You can make the same kind of connection to your own server, or using different ports, though if you connect to port 465 you should skip the `-starttls smtp` option.

## "Could not instantiate mail function"

This means that your PHP installation is not configured to call the `mail()` function correctly (e.g. `sendmail_path` is not set correctly in your `php.ini`), or you have no local mail server installed and configured. To fix this you need to do one or more of these things:

- Install a local mail server (e.g. postfix).
- Ensure that your `sendmail_path` points at the sendmail binary (usually `/usr/sbin/sendmail`) in your `php.ini`. Note that on Ubuntu/Debian you may have multiple `.ini` files in `/etc/php5/mods-available` and possibly other locations.

- Use `isSendmail()` and set the path to the sendmail binary in PHPMailer ( `$mail->Sendmail = '/usr/sbin/sendmail';` ).
- Use `isSMTP()` and send directly using SMTP.

## Addressing

It's important that you use valid email addresses. Every place that PHPMailer accepts an email address property, it expects an RFC821-format address, **not** an RFC822 one, for example `user@example.com` , **not** `Joe User <user@example.com>` . All the functions that accept an email address, like `addAddress` will return a boolean `true` if the address was accepted. Domain names containing non-ascii chars like `café.com` will use IDN 'punycode' format, which can't be evaluated properly until you ask PHPMailer to `send()` , so errors relating to them will appear later than for regular addresses.

## It's still not working!

If any of the above checks fail, PHPMailer will not work either, and usually there's nothing that PHPMailer can do about it. So go fix your network, then try again. If you are not in control of your own firewall or DNS, you probably need to raise a support ticket with your ISP to fix this (it's very common for them to block or divert port 25 outbound). If they won't fix it, you need to replace your ISP. *PS: BlueHost doesn't support smtp.gmail.com, they want you to use their SMTP server. The work around would be to use email associated with BlueHost and their host address Or send using mail() function in this case.*

## Where else to get help?

Several resources are worth checking:

- [The code examples](#) provided with PHPMailer. Base your code on these, not some ancient example from 2003.
- [The API docs](#).
- [The code itself](#) - it's very well commented.
- [The issue tracker](#) - it's very likely a problem similar to yours has happened before, so **search in there before** opening a ticket. If you do create an issue, be sure to include your code, preferably the minimum necessary to reproduce or define the problem so that we have a chance to see what you're seeing - saying "It doesn't work" is not a bug report!
- [StackOverflow](#) - há uma tonelada de perguntas do PHPMailer, a grande maioria dos quais pode ser corrigido lendo esta página! **Procure as perguntas** para a mensagem de erro que está a ver **antes de** postar uma pergunta. Se você postar uma pergunta em SO, certifique-se de marcar isso `PHPMailer` para que possamos vê-lo e **também não** abra um problema aqui. O rastreador de problemas aqui é destinado a erros reais no PHPMailer, não problemas com seu servidor.

