[](https://camo.githubusercontent.com/0d858d6dac4d3f6fab7d42de2c09d32ee2de9c5b/68747470733a2f2f7261772e6769746875622e636f6d2f5048504d61696c65722f5048504d61696c65722f6d61737465722f6578616d706c65732f696d616765732f7068706d61696c65722e706e67)

**PHPMailer - A full-featured email creation and transfer class for PHP**

**License**

This software is distributed under the [LGPL 2.1](http://www.gnu.org/licenses/lgpl-2.1.html) license. Please read LICENSE for information on the software availability and distribution.

**Installation & loading**

PHPMailer is available via [Composer/Packagist](https://packagist.org/packages/phpmailer/phpmailer) (using semantic versioning), so just add this line to your composer.json file:

"phpmailer/phpmailer": "~5.2"

or

composer require phpmailer/phpmailer

If you want to use the Gmail XOAUTH2 authentication class, you will also need to add a dependency on the league/oauth2-client package.

Alternatively, copy the contents of the PHPMailer folder into one of the include\_path directories specified in your PHP configuration. If you don't speak git or just want a tarball, click the 'zip' button at the top of the page in GitHub.

If you're not using composer's autoloader, PHPMailer provides an SPL-compatible autoloader, and that is the preferred way of loading the library - just require '/path/to/PHPMailerAutoload.php'; and everything should work. The autoloader does not throw errors if it can't find classes so it prepends itself to the SPL list, allowing your own (or your framework's) autoloader to catch errors. SPL autoloading was introduced in PHP 5.1.0, so if you are using a version older than that you will need to require/include each class manually.

PHPMailer does *not* declare a namespace because namespaces were only introduced in PHP 5.3.

If you want to use Google's XOAUTH2 authentication mechanism, you need to be running at least PHP 5.4, and load the dependencies listed in composer.json.

**Minimal installation**

While installing the entire package manually or with composer is simple, convenient and reliable, you may want to include only vital files in your project. At the very least you will need [class.phpmailer.php](https://github.com/PHPMailer/PHPMailer/tree/master/class.phpmailer.php). If you're using SMTP, you'll need [class.smtp.php](https://github.com/PHPMailer/PHPMailer/tree/master/class.smtp.php), and if you're using POP-before SMTP, you'll need [class.pop3.php](https://github.com/PHPMailer/PHPMailer/tree/master/class.pop3.php). For all of these, we recommend you use [the autoloader](https://github.com/PHPMailer/PHPMailer/tree/master/PHPMailerAutoload.php) too as otherwise you will either have to require all classes manually or use some other autoloader. You can skip the [language](https://github.com/PHPMailer/PHPMailer/tree/master/language/) folder if you're not showing errors to users and can make do with English-only errors. You may need the additional classes in the [extras](https://github.com/PHPMailer/PHPMailer/blob/master/extras) folder if you are using those features, including NTLM authentication and ics generation. If you're using Google XOAUTH2 you will need class.phpmaileroauth.php and class.oauth.php classes too, as well as the composer dependencies.

**A Simple Example**

<?php

require 'PHPMailerAutoload.php';

$mail = new PHPMailer;

//$mail->SMTPDebug = 3; // Enable verbose debug output

$mail->isSMTP(); // Set mailer to use SMTP

$mail->Host = 'smtp1.example.com;smtp2.example.com'; // Specify main and backup SMTP servers

$mail->SMTPAuth = true; // Enable SMTP authentication

$mail->Username = 'user@example.com'; // SMTP username

$mail->Password = 'secret'; // SMTP password

$mail->SMTPSecure = 'tls'; // Enable TLS encryption, `ssl` also accepted

$mail->Port = 587; // TCP port to connect to

$mail->setFrom('from@example.com', 'Mailer');

$mail->addAddress('joe@example.net', 'Joe User'); // Add a recipient

$mail->addAddress('ellen@example.com'); // Name is optional

$mail->addReplyTo('info@example.com', 'Information');

$mail->addCC('cc@example.com');

$mail->addBCC('bcc@example.com');

$mail->addAttachment('/var/tmp/file.tar.gz'); // Add attachments

$mail->addAttachment('/tmp/image.jpg', 'new.jpg'); // Optional name

$mail->isHTML(true); // Set email format to HTML

$mail->Subject = 'Here is the subject';

$mail->Body = 'This is the HTML message body <b>in bold!</b>';

$mail->AltBody = 'This is the body in plain text for non-HTML mail clients';

if(!$mail->send()) {

echo 'Message could not be sent.';

echo 'Mailer Error: ' . $mail->ErrorInfo;

} else {

echo 'Message has been sent';

}

You'll find plenty more to play with in the [examples](https://github.com/PHPMailer/PHPMailer/tree/master/examples) folder.

That's it. You should now be ready to use PHPMailer!

**Localization**

PHPMailer defaults to English, but in the [language](https://github.com/PHPMailer/PHPMailer/tree/master/language/) folder you'll find numerous (46 at the time of writing!) translations for PHPMailer error messages that you may encounter. Their filenames contain [ISO 639-1](http://en.wikipedia.org/wiki/ISO_639-1) language code for the translations, for example fr for French. To specify a language, you need to tell PHPMailer which one to use, like this:

// To load the French version

$mail->setLanguage('fr', '/optional/path/to/language/directory/');

We welcome corrections and new languages - if you're looking for corrections to do, run the [phpmailerLangTest.php](https://github.com/PHPMailer/PHPMailer/tree/master/test/phpmailerLangTest.php) script in the tests folder and it will show any missing translations.

**Documentation**

Examples of how to use PHPMailer for common scenarios can be found in the [examples](https://github.com/PHPMailer/PHPMailer/tree/master/examples) folder. If you're looking for a good starting point, we recommend you start with [the Gmail example](https://github.com/PHPMailer/PHPMailer/tree/master/examples/gmail.phps).

There are tips and a troubleshooting guide in the [GitHub wiki](https://github.com/PHPMailer/PHPMailer/wiki). If you're having trouble, this should be the first place you look as it's the most frequently updated.

Complete generated API documentation is [available online](http://phpmailer.github.io/PHPMailer/).

You'll find some basic user-level docs in the [docs](https://github.com/PHPMailer/PHPMailer/blob/master/docs) folder, and you can generate complete API-level documentation using the [generatedocs.sh](https://github.com/PHPMailer/PHPMailer/tree/master/docs/generatedocs.sh) shell script in the docs folder, though you'll need to install [PHPDocumentor](http://www.phpdoc.org/) first. You may find [the unit tests](https://github.com/PHPMailer/PHPMailer/tree/master/test/phpmailerTest.php) a good source of how to do various operations such as encryption.

If the documentation doesn't cover what you need, search the [many questions on Stack Overflow](http://stackoverflow.com/questions/tagged/phpmailer), and before you ask a question about "SMTP Error: Could not connect to SMTP host.", [read the troubleshooting guide](https://github.com/PHPMailer/PHPMailer/wiki/Troubleshooting).

**Tests**

There is a PHPUnit test script in the [test](https://github.com/PHPMailer/PHPMailer/tree/master/test/) folder.

Build status:

If this isn't passing, is there something you can do to help?

**Security**

Please disclose any vulnerabilities found responsibly - report any security problems found to the maintainers privately.

PHPMailer versions prior to 5.2.20 (released December 28th 2016) are vulnerable to [CVE-2016-10045](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-10045) a remote code execution vulnerability, responsibly reported by [Dawid Golunski](https://legalhackers.com/advisories/PHPMailer-Exploit-Remote-Code-Exec-CVE-2016-10045-Vuln-Patch-Bypass.html), and patched by Paul Buonopane (@Zenexer).

PHPMailer versions prior to 5.2.18 (released December 2016) are vulnerable to [CVE-2016-10033](https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-10033) a critical remote code execution vulnerability, responsibly reported by [Dawid Golunski](http://legalhackers.com/advisories/PHPMailer-Exploit-Remote-Code-Exec-CVE-2016-10033-Vuln.html).

See [SECURITY](https://github.com/PHPMailer/PHPMailer/tree/master/SECURITY.md) for more detail on security issues.

**Contributing**

Please submit bug reports, suggestions and pull requests to the [GitHub issue tracker](https://github.com/PHPMailer/PHPMailer/issues).

We're particularly interested in fixing edge-cases, expanding test coverage and updating translations.

With the move to the PHPMailer GitHub organisation, you'll need to update any remote URLs referencing the old GitHub location with a command like this from within your clone:

git remote set-url upstream https://github.com/PHPMailer/PHPMailer.git

Please *don't* use the SourceForge or Google Code projects any more.

**Sponsorship**

Development time and resources for PHPMailer are provided by [Smartmessages.net](https://info.smartmessages.net/), a powerful email marketing system.

Other contributions are gladly received, whether in beer 🍺, T-shirts 👕, Amazon wishlist raids, or cold, hard cash 💰.

**Changelog**

See [changelog](https://github.com/PHPMailer/PHPMailer/blob/master/changelog.md).

**History**

* PHPMailer was originally written in 2001 by Brent R. Matzelle as a [SourceForge project](http://sourceforge.net/projects/phpmailer/).
* Marcus Bointon (coolbru on SF) and Andy Prevost (codeworxtech) took over the project in 2004.
* Became an Apache incubator project on Google Code in 2010, managed by Jim Jagielski.
* Marcus created his fork on [GitHub](https://github.com/Synchro/PHPMailer).
* Jim and Marcus decide to join forces and use GitHub as the canonical and official repo for PHPMailer.
* PHPMailer moves to the [PHPMailer organisation](https://github.com/PHPMailer) on GitHub.

**What's changed since moving from SourceForge?**

* Official successor to the SourceForge and Google Code projects.
* Test suite.
* Continuous integration with Travis-CI.
* Composer support.
* Public development.
* Additional languages and language strings.
* CRAM-MD5 authentication support.
* Preserves full repo history of authors, commits and branches from the original SourceForge project.