Email and Frameworks in PHP

Email in PHP

- PHP uses Mail() function to send an email from your website.
- Parameters required
  - (Recipient’s email address, subject, body)
- Optional parameters
  - (Header, parameters)

mail (to, subject, message, headers, parameters)

Though, using mail() is quite simple, but it involves lot of considerations. You need to check the web server, SMTP, authentication etc. which makes writing mail program cumbersome.

The code for simple mail function is as follows:

```php
<?php
$mesg= "Welcome to e-pgpathshala";
$headers="From: bhumikagshah@gmail.com";
$str=mail("bumikagshah@gmail.com","Test Mail",$mesg,$headers);
if ($str==true) {
    echo "Message sent ";
} else {
    echo "Message not sent";
}
?>
```

The above program would immediately return you Message sent, but might always not result is success.
Hence to overcome the uncertainties we have used the `phpmailer()` which is available for download on github.

**PHPMailer()**

- It is a type of library which has a collection of functions that provide with building and sending email messages. It supports various ways of sending mails like `mail()`, `qmail`, direct SMTP mail etc.
- We would be using PHPmailer to send our E-Mail.

<table>
<thead>
<tr>
<th>Mail</th>
<th>PHPMailer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not so user friendly</td>
<td>User friendly</td>
</tr>
<tr>
<td>You have to integrate the mail server yourself (Using SMTP has its own considerations)</td>
<td>Does everything automatically</td>
</tr>
</tbody>
</table>

Program for Sending Mail

```php
<?php
require 'PHPMailer-Master/PHPMailerAutoload.php';

$mail = new phpmailer();
//spl_autoload()

$mail->isSMTP(); // Set mailer to use SMTP
$mail->Host = 'smtp.gmail.com'; // Specify main and backup SMTP servers
$mail->SMTPAuth = true; // Enable SMTP authentication
$mail->Username = 'bhumikagshah@gmail.com'; // SMTP username
$mail->Password = 'Your Password'; // SMTP password
$mail->SMTPSecure = 'tls'; // Enable TLS encryption, `ssl` also accepted
```
$mail->Port = 587; // TCP port to connect to

$mail->setFrom('bhumikagshah@gmail.com', 'epgpathshahla');
$mail->addReplyTo('bhumikagshah@gmail.com', 'epgpathshala');
$mail->addAddress('bhumikagshah@gmail.com'); // Add a recipient
$mail->addCC('cc@example.com');
$mail->addBCC('bcc@example.com');

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

$bodyContent = '<h1>How to Send Email using PHP in Localhost by Bhumika</h1>';
$bodyContent .= '<p>This is the HTML email sent from localhost using PHP script by <b>bhumika</b></p>;

$mail->Subject = 'Email from Localhost by Bhumika';
$mail->Body    = $bodyContent;

if(!$mail->send()) {
    echo 'Message could not be sent.';
    echo 'Mailer Error: ' . $mail->ErrorInfo;
} else {
    echo 'Message has been sent';
}
?>
You will also have to include Autoloader file of the part of library, which we have used on top of the program, The code of autoloader program is as follows:

* PHPMailer SPL autoloader.
* @param string $classname The name of the class to load

```php
function PHPMailerAutoload($classname)
{
    //Can't use __DIR__ as it's only in PHP 5.3+
    $filename = dirname(__FILE__).DIRECTORY_SEPARATOR.'class.'.strtolower($classname).'.php';
    if (is_readable($filename)) {
        require $filename;
    }
}
```

if (version_compare(PHP_VERSION, '5.1.2', '>=')) {
    //SPL autoloading was introduced in PHP 5.1.2
    if (version_compare(PHP_VERSION, '5.3.0', '>=')) {
        spl_autoload_register('PHPMailerAutoload', true, true);
    } else {
        spl_autoload_register('PHPMailerAutoload');
    }
} else {
    /**
     * Fall back to traditional autoload for old PHP versions
     * @param string $classname The name of the class to load
     */
    function __autoload($classname)
    {
        PHPMailerAutoload($classname);
    }
}
Mail.php

<?php
$mail=new PHPMailer();
$mail->SMTPDebug=0;
$mail->IsSMTP();
$mail->SMTPAuth=true;
$mail->SMTPSecure='ssl';
$mail->Host="smtp.gmail.com";
$mail->Port=587;
$mail->IsHTML(true);
$mail->Username="bhumikagshah@gmail.com";
$mail->Password="Your Password";
$mail->SetFrom("bhumikagshah@gmail.com");
$mail->Subject="This is My Email subject";
$mail->Body="This is Body of send email";
$mail->AddAddress("bhumikagshah@gmail.com");
if(!$mail->Send())
{
    echo "Mail Error:.$mail->ErrorInfo;
}
else
{
    echo "Message has been sent";
}
?>
Troubleshooting Mail issues

Gmail Security settings

You will have to edit Gmail security settings so as to allow mail from programs.

Visit: accounts.google.com
After allowing lesser secure apps, you will be able to get mail from your program.

**Troubleshooting Email program:**

- Check Gmail security
- Check SMTP
- Php.ini
- Sendmail.ini
Frameworks in PHP

- Provides **structured way**, with built-in modules: Frameworks give users a basic structure with built-in modules to develop applications faster and with much ease.

- Uses **MVC Architecture**: Frameworks in PHP use MVC architecture, wherein the business logic exists independently of data layer and presentation layer.

- Enforces **coding standards** and development guidelines: The framework enforces the development standards and development guidelines which helps in standardizing the development process.

- **Improved stability and Quality**: Dividing modules in parts increases the development speed and hence improves scalability and quality.

- Easy to upgrade and maintain developed application.

- **Good community support**

Types of Frameworks for PHP

- Laravel
- **Codeigniter**
- CakePHP
- Symfony
- Zend Framework

We will take an overview of each of the Framework and CodeIgniter Framework in detail.

**LARAVEL**

- Popularly known as Framework for Web Artisans, Laravel is open source framework which follows MVC architecture and consists of simple toolkit which enables developers to create complex web applications easily. Additionally, it also provides robust security features. It has a huge ecosystem with instant hosting and deployment.

**CakePHP**

- Already a decade old, CakePHP is designed to ease the common web development tasks and consists of all-in-one toolbox. It can exist isolated and also as a whole. The rich set of libraries make coding easy for developers.

**Symfony**

- Popularly known as PHP web application framework, it is a set of reusable PHP components/libraries. It helps in reducing repetitive tasks and has a low performance overhead. It has set of tools and a development methodology.
Zend Framework

- It is an object oriented framework which is basically a collection of professional PHP packages. It has a collection of 60+ packages. Due to its configuration options, it is not recommended for small projects, but works best for the complex projects.

**Comparative analysis**

<table>
<thead>
<tr>
<th>PHP Frameworks</th>
<th>Laravel</th>
<th>Symfony</th>
<th>CodeIgniter</th>
<th>CakePHP</th>
<th>Zend Framework</th>
</tr>
</thead>
<tbody>
<tr>
<td>Latest Version number</td>
<td>5.4</td>
<td>3.2</td>
<td>3.1.3</td>
<td>3.4</td>
<td>2.5</td>
</tr>
<tr>
<td>Features</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Database agnostic migration</td>
<td>• YAMLL • Event dispatcher • Dependency Injector • Templating engine • Application profiling • Template engine class • Query builder • Data encryption • Authentications facilities • Console tools • Controllers • Modeless forms • Plugins • End-to-end encryption • Package dependency manager • Continuous integration service</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prominent projects</td>
<td>• Deltanet Travel • World Walking • FusionInvoice • Asgard CMS</td>
<td>• Drupal • Laravel • phpBB • OROCRM</td>
<td>• Buffer • Small SEO Tools</td>
<td>• BMW • Express • Sainsburys Bank</td>
<td>• BBC • BNP PARIBAS • Cisco Webex</td>
</tr>
<tr>
<td>License</td>
<td>MIT</td>
<td>MIT</td>
<td>MIT</td>
<td>MIT</td>
<td>New BSD</td>
</tr>
</tbody>
</table>

**CodeIgniter**

- It is a light weight PHP framework. Code igniter is known for its flexibility and easy installation.
- CodeIgniter is popular among beginners and developers as it is very easy to learn and has a rich set of APIs.
- The crud (create/read/update/delete) functionality is well integrated hence database related tasks are implemented easily and quickly.

**CodeIgniter Framework**

![CodeIgniter Framework Image]

**CodeIgniter (Sample Project)**

![CodeIgniter Sample Project Image]
Database.php

```php
// The $query_builder variables lets you determine whether or not to load
// the query builder class.

$active_group = 'default';
$use_query_builder = TRUE;

$db['default'] = array(
    'dsn' => ..., 'hostname' => 'localhost',
    'username' => 'root',
    'password' => ..., 'database' => 'police',
    'dbdriver' => 'mysql',
    'dbprefix' => '',
    'pconnect' => FALSE,
    'db_debug' => (ENVIRONMENT !== 'production'),
    'cache_on' => FALSE,
    'cachedir' => '',
    'char_set' => 'utf8',
    'db维护' => 'utf8_general_ci',
    'swap_pre' => '',
    'encrypt' => FALSE,
    'compress' => FALSE,
    'stricton' => FALSE,
    'failover' => array(),
    'save_queries' => TRUE
);
```

Program: Leave application

```php
class LeaveModel extends Ct_Model {

    public function getNotifyAdminLeaveApplication() {
        $query = "SELECT la.id, employee_id, profilepicture, leave_category_id FROM leave_applications la, employees e, leave_categories lc WHERE la.employee_id = e.id AND la.leave_category_id = lc.id;"
        $leaveApplication = $this->db->query($query);
        return $leaveApplication->result();
    }

    public function getNotifyAdminLeaveApplicationsCount() {
        $query = "SELECT * FROM leave_applications WHERE notify_admin = 1;"
        $leaveApplication = $this->db->query($query);
        return $leaveApplication->num_rows();
    }

    public function getMyPendingLeaveCount() {
        $query = "SELECT id FROM leave_applications WHERE employee_id = ". $this->id . " AND notify_admin = 1;"
        $leaveApplication = $this->db->query($query);
        return $leaveApplication->num_rows();
    }
```