Engine

Last known as LWEngine, redeveloped from the ground-up

Chapters

- Introduction
- Engine
  - Getting Started
    - Steps
  - Tips
- Config
- Templates
- Pages
- F.A.Q
  - Engine
    - I get the error message “Cannot load Engine libraries”
    - I get the error message “A database error occured”
    - I get the error message “Cannot initiate configuration”
    - I get the error message “Requires PHP version 5.3.0 and higher...”
    - I get the error message “Cannot initiate Template”
    - I get the error message “No template found”
    - I get the error message “Cannot run Engine”
    - How can I protect myself against MYSQL Injection attacks?
    - Can I get the current directory?
    - Can I get a URL for something on the website?
    - Can I generate a random ID/a random range of characters?
    - Can I start/clear a session?
    - Can I change my template?
    - Can I require everything within a directory?
    - What does “index.php” do in the root directory?
  - Config
    - I cannot get past the configuration setup screen
    - I cannot get past the database setup screen.
    - I cannot get past the template setup screen.
    - I cannot get past the finish setup screen.
    - I get the error message “Configuration file was set to automatically open, but it didn’t exist”
    - How can I get back to the setup process?
    - I’m in the middle of the setup process, can I restart the setup process?
    - Can I setup the configuration without using the setup process?
    - Can I create my own setup screen?
    - Can I create a configuration with default values?
  - Templates
    - Can I create a template?
    - Where are templates stored?

- What should the template hierarchy look like?
- Does my template have to contain pages?

Pages
- I get the error message “Page not found”
- I get the error message “Page script error”
- How do I create a page?
- How should a page be used?
- How can I customise the error page?
- How should I present my (web)page?

DBConnection
- How do I use DBConnection
- Do I need to create a new instance?

DBObject
- How do I use DBObject?
- Do I have to use DBObject?

Introduction

LWEngine was first created as a way to create a base structure to any type of website, whatever the type of website by using simple and easy to use methods of implementation so that the building blocks of the website is already there and just has to built to implement the functionality by building plugins and templates and using the prebuilt functionality as a backbone to the fundamental structure the engine provided.

Pages we originally constructed with a database where pages were based off their types with inheritance that could define the way the page was handled, and to remove the pressure of using a heavy load of MYSQL, loading the page into a simple model so that the page would be found by itself, and handled without effort.

Security was on the list as one of the top priorities, as a web developer, the most important thing when building a PHP application are the basic building blocks that create your web application, when the security holes from MYSQL Injection to private and unprotected files, it's useful to remove these first so that the real web application can be brought forward without the repetition of these steps. Your PHP code is also insecure when files aren't protected as they can give away some of your web applications special secrets and the only way around this is a tightly bound system that is highly coupled and very strict.

The plugins enabled a way of preventing people having to edit the way the system behaves by overriding what they don't like or want to change, sadly the truth was that afterwards, any simple website would override the stuff they didn't like so-much-so that the engine was literally hanging by threads whereby the original code was hidden in a mess of unused code. However, luckily the website was still fast.

Another way that LWEngine wanted to succeed was to make the setup as easy as possible for new hosts and clients so that it allowed anyone to be able to create a new website, LWEngine never came with a predefined template so when the setup was completed, users were expected to implement their own system which completely removed the ease-of-use factor quite quickly. As LWEngine was created in a fast pace business environment, it never had developed its own basic template to allow others without the business' template pre-installed and as a result, LWEngine was never released to the public as it lacked the time and patience of a new product branding. However, the configuration setup was the most important step that made the developer realise the potential for an simple small structural engine that could be used as a simple framework for a blogger, or website creator as a consumer based product and a heavily customisable and powerful framework that could be the stepping stone for a small company.

Here's to a better LWEngine and a brighter future

Engine

Most of the time when developing websites, I find that I often have to create, or at least, recreate the same functionality in each website or page. The engine is hopefully there to do some of it for you, however, you may find that the Engine does hardly any of the work for you as at most, it just handles some of the most fundamental and helpful functions to running itself for you. Some of these popular functions are explained below.

Getting Started

It should hopefully be relatively easy so we will go over the basics so that we can ensure that you get the process right. Making sure that before we start that we have Apache, PHP and MYSQL installed. PHP may not be configured to start with Apache on certain linux distros so you may have to set that up and you must use version 5.3.0 or higher. It's also handy to use PHPMyAdmin to view and maintain your database that you may create, if you decide to use one.

Steps

1. Put everything where you got LWEngine in your hosting directory
2. Go to the hosting IP address, or hostname (e.g. "127.0.0.1" or "localhost")
3. Go through the setup process
4. Create your web application

Tips

Use PHPMyAdmin to view and maintain your databases visually, that along with learning databases on an online course such as W3Schools will allow you to start learning how to create your own Web Application with a database rather easily and you will be able to see the outcome in a presentable manner rather than using MYSQL in a terminal.

If you're using Windows, it may just be easier to use an application such as Wamp Server or XAMPP to setup a small development environment on your computer rather than purchasing or using a host just yet as they can be quite expensive.

The use of a text editor such as Sublime Text, Atom or Notepad++ is always a good choice when you're developing a web application, as otherwise, applications such as Dreamweaver; insert too much code that could otherwise be self-taught and easier to debug in the long term.

Config

The configuration was part of LWEngine so that values that would otherwise be placed as constants or variables were put in a configuration so that they could be changed by a user if they ever wanted to do so. Of course some of this information is private as it could include things from database credentials to password hashing keys and password acceptance criteria.

However, with our strict rules set out by our htaccess file, we are able to hide most of the stuff that could be deemed unsafe and only unblock public files that will be used in a real world scenario such as Images, CSS and Javascript. However, it could easily be modified to include audio and video, however more appropriate platforms can handle this for us, especially in the copyright and encryption forte.

The config was once used to store a list of non-acceptable words for usernames and passwords, the default user account picture and more. As you can tell, it was very useful for the functionality it brought forward and it was easy to modify with a program or LWEngine itself.

Templates

Templates are quite new to this version of the engine but only in terms of their functionality. Previously, templates referred to what is presented to the page in terms of a header, middle and footer page to a website, however, it had a lot of problems whereby, there wasn't much automation. For example, templates would find a hard time finding their path without the help of the engine if the path of the engine was different from virtual hosts, customisation and more. Links we're also stuck in place and specified in different places allowing for multiple points of failure if the page changes their URL and it would be harder to trace. Instead, the automation helps allow for the information to defined once so that it is easier to debug and can be changed when required. Templates can also have behaviour implemented so that instead of just showing the name and traversing the pages, it could change the site depending on a user's login status, and so much more. Think of a template as more of a website rather than the view of your web page. Think of the template as the template of behaviour, so how your website behaves with the website, the browser and the user. Pages are then defined to create each URL behaviour and in the page, you can define each view that contributes to the template, or in other words, the **real** template that shows the website.

Pages

Pages are even more generic than before as last time pages we always defined by a simple database model and overridden by the page type using a value in a database. I would then use the term 'static pages' to refer to pages that would not be stored in a database and would have some functionality to it. Sadly it prevent real progress from being created as they had huge setbacks. I had implemented admin and user static pages that would dynamically changed, but they added more problems than they gave solutions. Instead, by making a generic abstract page class, the page tells the template what it is doing. For example, the getURL(): String function allows other classes to refer to itself when required, the isMatch(URL: String): Boolean function tells the template whether it matches the URL it's asking for. Instead of making the engine assume it has to be a kind of special URL to determine the functionality, the page defines the functionality for itself. For example, if I wanted it to match anything, I could just return true, if I wanted it to match a certain pattern, I could match it to a regular expression, or if I wanted it to be specific, I could specify it manually (i.e. return URL == "/home/"). This then allows the old functionality I wanted before with blog pages, user pages, normal pages and the old static pages.

Now these can be defined to do anything, easily with their own run and show function. Firstly, the run function will be run if the match is found so that all and/or most of the PHP is dealt with first (or technically the constructor could, but let's ignore that for now). The show function then is all about presentation to the user. How does the page show the information to the user? Does it use the same header and footer? Do you want to show different content, do you want to remove the header and footer? Does it use a different type of template altogether such as an authentication screen? These are questions that are up to you to answer and implement. It allows the functionality to all be there without the plugins and static pages to restrict and override the previous functionality.

At first, I wanted to define the pages and template as normal classes that just get instantiated, however, by making them abstract classes, you derive the class and implement your own functionality to your template so that you aren't limited on what you can create but by what you can imagine. Also, if the pages weren't abstract, you would have to define each run, show, match and title function which would be a very irritating experience.

F.A.Q

Engine

I get the error message "Cannot load Engine libraries"
The libraries required by the engine cannot be found so a fatal error has occurred because the application cannot proceed any further.

I get the error message "A database error occured"
The database cannot connect or there was trouble trying to protect the $_POST and $_GET variables from MYSQL Injection.

I get the error message "Cannot initiate configuration"
There was either an error in the configuration setup process or the configuration file cannot be found or used.

I get the error message "Requires PHP version 5.3.0 and higher..."
The engine requires PHP version 5.3.0 as it is the most recent secure and stable version you can find without getting any older, and without removing some core functionality.

I get the error message "Cannot initiate Template"
The template that it finds must specify a newly instantiated, derived template class in 'main.php' of your template directory.

I get the error message "No template found"
The template doesn't have an instantiated template object, make sure you get the main template filename by using the static getTemplate(name) function to return the full filename, after, you must use the function useTemplate() to load the template in the engine before you are able to run the engine.

I get the error message "Cannot run Engine"
This will happen when there is an error when the current template cannot be found or when an uncaught error has been thrown within the template instance.

How can I protect myself against MYSQL Injection attacks?
The engine must connect to a database to be able to protect you from MYSQL attacks so that the POST and GET values can be protected before you use them. When you need to use them, they will already be protected and ready to use. Just use $_POST and $_GET like before.

Can I get the current directory?
You can use __DIR__ to get the current directory of a class, however if you want the original directory of where the engine folder is located, you can use the static getLocalDir() function to get the file path.

Can I get a URL for something on the website?
A URL can be generated from the engine itself by using a static getRemoteDir(path) function. the getRemoteDir function allows you to get the relative URL path, however, to specify a URL typically found outside the website, you will require an absolute path, you can get it with a static getRemoteAbsolutePath(path) function.

Can I generate a random ID/a random range of characters?
The engine provide a static generateRandomString(length) function to generate a random ID or a random range of characters.

Can I start/clear a session?
The engine provides the facility to do so properly with the static functions startSession() and clearSession(), however, the engine already starts the session at the start of the engine's instantiation.

Can I change my template?
Just get the template filename with either the name using the static getTemplate(name) function, or get an array full of possible templates from the static getTemplates() function. Use the useTemplate(filename) function within the engine instance to change the engine's template.

Can I require everything within a directory?
The static function requireAllInDir(dir) function provides you the ability to import files from a directory, however, it may be easier, or better to just include the necessary files yourself.

What does "index.php" do in the root directory?
The index.php file includes a basic ErrorHandler class to help show errors in a nice HTML friendly manner before the engine is even found and instantiated so that any errors that can occur can be shown before the engine is loaded. The engine is then required by the index file and is run automatically. I normally would not change this functionality unless you want to change the folder that the engine is running within. However, I don't see many cases where this is normally an issue.

Config

I cannot get past the configuration setup screen
Make sure you have set the right permissions to the directory and to all the files and directory inside it. Under Linux, apache may run at www-data, nobody or root. Make sure you chmod all the files inside to 755 and to chown it to the user apache is using so that apache can read and write files. If you still cannot write the setup, you may be having problems writing the .htaccess file, for setup to complete, both ".htaccess" and "engine/config/config-data.json" have to written to the directory.

I cannot get past the database setup screen.
It's important that you get these credentials correct for the engine to be able to connect to MYSQL, this also helps when a template or your customised template wants to use a database connection. Are you sure that MYSQL is running?

I cannot get past the template setup screen.
The template screen loads the templates by folder name, if you cannot find the template, make sure that the template has a main.php file inside the templates folder name, and also, the template folder being within the templates directory. You must return a new instantiated and also, derived template class before the template setup screen recognises the template. You

may get shown an error before hand to correct these mistakes is the engine is unable to proceed or may pose risks in the future.

I cannot get past the finish setup screen.
The finish setup screen only sets up the config-data.json file after the setup is successfully complete. Please follow “I cannot get past the configuration setup screen” for information that must be causing the problem.

I get the error message “Configuration file was set to automatically open, but it didn’t exist”
By default, the configuration class is set to open the configuration file on instantiation. You should set this to false in your code if you’re having problems with your configuration file, please see “Can I create my own configuration file?” for more information on configuration customisation.

How can I get back to the setup process?
If for some reason, you want to reset the setup process, you can do so by deleting the config-data.json file found in the config folder inside the main engine directory (and optionally, and most favourably, the .htaccess file found in the root directory). You can use this method if you want to change your host information or if you want to give the engine to another as config-data will contain your private database information.

I’m in the middle of the setup process, can I restart the setup process?
You will have to delete the temp-config-data.json file inside the “config/setup/” folder inside the main engine directory. If you have .htaccess in your root directory, I recommend you delete it, as it was created by the server in the setup process.

Can I setup the configuration without using the setup process?
If for some reason, you may need to make the setup process easier/harder for whatever reason, you can setup the configuration manually. To do this, create your own config-data.json file in the config folder within the main engine directory (typically “/engine/”). In the file you will want to have these basic variables:

```
{
   "DB_HOST": "{the database host you’re connecting to}",
   "DB_USERNAME": "{the username used to connect to the database host}",
   "DB_PASSWORD": "{the password, if any}",
   "DB_NAME": "{the database name you are going to use in the database connection}",
   "TEMPLATE": "{the template name the engine must use}"
}
```

Just replace the curly braces and the information contained in the curly braces with the relevant information and you will be able to use the engine. However, please note that without without the generated htaccess file, you’re running the engine insecure.

Copy and paste the text below into a file named .htaccess in the root directory to make the engine safer. You can uncomment the https option to redirect everything to a https request.

```
Options -Indexes
RewriteEngine On
#RewriteBase /
#RewriteCond %{HTTPS} !on
#RewriteRule (.*) https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]

RewriteCond %{REQUEST_FILENAME} -f
RewriteRule \.(jp[e]?g|gif|png|css|js|ttf|woff|ico|bmp|pdf|doc[x]?)$ - [L]

#Redirect all files not match index.php
RewriteCond %{REQUEST_FILENAME} !(.*)/index\.php$
RewriteRule ^.*$ index.php?current_engine_page=$0 [L,NC,QSA]
```

Can I create my own setup screen?
Yes, this is indeed possible by placing a derived SetupChapter within the chapter folder and specifying the order within the main.php file within the “config/setup/” folder inside the main engine directory. You can then specify whether you want it inside the start and end of the setup and so on. See “Can I create a configuration with my own default values” for more configuration setup defaults.

Can I create a configuration with default values?
Sometimes, it’s better to give the user a default value, especially in a work or school environment as it helps remove the need to specify hosts and can help automate part of the process. You can specify the config defaults within the config class in the “/config/” folder in the main engine directory.

Templates

Can I create a template?
Yes, however, this in terms of the website behaviour rather than the design of the website, also known as a template. This kind of template defines the of the website's structure, including the pages, the people who created it and also the pages within the website. When making a template you have to create a new folder within the templates directory. The name of the folder is the name of the behaviour for the website, so for example, I could use MyWebsite, my brand name, the best names are whatever that defines the behaviour or the name of the website, however, it’s still up to you at the end of the day.

Next, you will want to create a main.php page and inside, you will define a new derived template. At the bottom of the page, you will have to return a new instance of class you just created to allow the engine to know the template you just added. Take a look at the code below to see a Template example.

```
<?php
class MyWebsite extends Template{
	public function getName(){
		return "My Website";
	}
	public function getPages(){
		return null;
	}
}
return new MyWebsite();
?>
```

Currently the template doesn't have any pages, but it's the starting block to creating the behaviour.

Where are templates stored?
Templates are stored in the '/templates/' directory within the main engine directory.

What should the template hierarchy look like?
With your template directory, you should aim to have a few folders to manage the structure of a typical website. This is currently the best hierarchy for the template:

- other - this is optional but it allows you to put additional classes in here if you want to free up space in your template directory
- public - this is where your client data (images, CSS, and Javascript)
    - css - separate the css from the others
    - js - separate the js from everything else
    - images - images can get messy without a proper structure
- views - this is where you put the different viewports within your website (e.g. header.php, footer.php, innerhome.php, etc)
- libs - put any PHP libraries in here that you may be using

Does my template have to contain pages?
Yes, for your template to be usable you have to include pages, otherwise an error will show saying that it failed to find an available page. To find out more about custom error pages, view "How can I customise the error page?".

Pages

I get the error message “Page not found”
This error occurs when the search was complete and nothing could be found as a suitable match to run as a page

I get the error message “Page script error”
When an implemented page throws an error, this error will be used to show that the page was the page developer's fault rather than an engine fault.

How do I create a page?
A page defines the behaviour of that particular page (however a page’s behaviour may create more than one page, virtually (e.g. regular expression to match for blog posts might look something like this: “\/(blog)/[0-9]+\/[0-9]{2}\/[0-9]{2}/[a-zA-Z0-9-]+[\/]?”, or otherwise in simple form; “/blog/year/month/date/title”). To create a derived class, you can follow and copy the code below as a simple starting point, I called the file Homepage.php

```
<?php
class Homepage extends Page{
        public function getName(){
                return "Home";
        }
        public function getURL(){
                return "/";
        }
        public function isMatch($URL){
                return $URL == $this->getURL();
        }
        public function run($template){
                //Do something here
        }
        public function show($template){
                //Show something here, for example...
                include(__DIR__ . '/views/header.php');
                include(__DIR__ . '/views/home.php');
                include(__DIR__ . '/views/footer.php');
        }
}
?>
```

You will have to change the template file to implement the changes, I will show you the new updated getPages() function

```
//Inside the MyWebsite template (assumed)
//The new getPages() function
include(__DIR__ . ‘/Homepage.php’);
public function getPages(){
	return array(new Homepage());
}
```

How should a page be used?
A page can be used anyway you want it to as long as it stays as a page within a template. The functionality within the page is up for you to decide. Most if the time the page shows what it would normally show on any other type of PHP implementation however instead of mixing PHP code with presentable data. We leave that to the show function. Other classes can then be created to mix in functionality such as a user class.

How can I customise the error page?
You can always create your own customised personal error 404 page by using a regular expression (or just returning true) in a new derived page object that matches everything in the isMatch() function and finally, make sure it is at the bottom of the array so it always finds it last and therefor, cannot find any of the pages above in the array.

However, for a customised error 500 page, the calls go directly to the ErrorHandler class (check out “What does index.php do in the root directory?”. To be able to change the error 500 page to your own, you will have to create the HTML for the error to be displayed. To describe the error, you also have to use a small template replacements and these are below:

- {errorName} - the title, name or small description of the error
- {errorCode} - the error number given
- {errorDescription} - a full description of the error

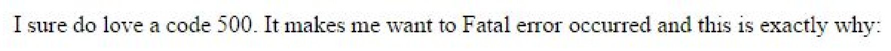
All you have to do is change the ErrorHandler HTML with the following command

```
ErrorHandler::setErrorHTML($html);
```

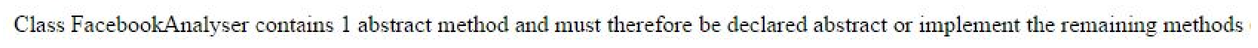
Here’s what the error handler code would look like in PHP Code with text example.

```
ErrorHandler::setErrorHTML(“<p>I sure do love a code {errorCode}. It makes me want to
{errorName} and this is exactly why: </p><br/><p>{errorDescription}</p>”);
```

What it looks like...



```
I sure do love a code 500. It makes me want to Fatal error occurred and this is exactly why:
```



```
Class FacebookAnalyser contains 1 abstract method and must therefore be declared abstract or implement the remaining methods
```

How should I present my (web)page?

The page should use a set of template viewports that can be included when required, then the website is as simple as swapping various blocks of HTML code. Then the design can be created meanwhile development is taking place.

DBConnection

How do I use DBConnection

The DBConnection class uses a singleton instance so that the connection to the database is seamless. Therefor when you want to use the database, it will available to use anytime you want.

N.B. I'm thinking about switching to PDO soon though as it allows us to focus on object-oriented models easily

Do I need to create a new instance?

No, as a singleton instance, we can access the instantiated object without constantly creating new database connections.

DBObject

How do I use DBObject?

You can derive the database model you want to create so that you can either use the built in update and select functions, or you can make your own selects after exporting the model.

Do I have to use DBObject?

Nope, it's totally up to you what you use to get Database results back, DBObject is just a way of trying to make it easier for us when we want a basic select