

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |
| --- | --- |
| **Document title** | **PHPSTENCIL** |
| **Document reference number** | **UEL -** |
| **Project name** | **UNISA e-Content 20140363** |
| **Unisa Department** | **Department of Economics** |

Table of Contents

1 Description of the component 5

2 How to Install The component 5

3 how to use the component 7

Document sign-off

This document has been seen and accepted by the following people:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Title | Date | Signature |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Compiler | Purpose | Change Summary |
| <yyyymmdd> | Name Surname | Document Created,  Internal Review,  External Review,  etc | Initial release,  Updates based on internal review,  Updates to section xxx,  etc |
|  |  |  |  |
|  |  |  |  |

# Description of the component

PHPSTENCIL

* What is the PHPSTENCIL?
* Installation and configuration

## Description of the PHPSTENCIL?

The PHPstencil is the main container which is set up as main container and holding all the components developed in Java. The Stencil support as well Tsugi implementation compatible with the LTI 2.0.

# How to install it

## Installation:

### Step 1

Install Laravel framework. The following command will install the latest version of Laravel with the project name LaravelDefault.

composer create-project laravel/laravel LaravelDefault

### Step 2

Download Zip from [PHPStencilGit repo](https://github.com/EonConsulting/PHPStencil).

### Step 3

Create a folder called "Packages" inside of the root of the Laravel project.

### Step 4

Paste the folder from the Git repo (Step 1.2) inside the Packages folder.

### Step 5

Inside the root Laravel project, open the composer.json file. In that file, look for:

"autoload" : {

"psr-4": {

...

Inside of the psr-4 tag, load the following:

"EONConsulting\\PHPStencil\\": "Packages/PHPStencil/src"

Just as above, except in the components composer.json file:

"EONConsulting\\PHPStencil\\": "src/\*"

### Step 6

Use as a global function instead of by namespace (Optional)

Under the psr-4 but still inside of the autoload if the array files does not exist, create it like this:

"files": [

...

]

And enter into it:

"Packages/PHPStencil/src/Http/helpers.php"

And again, in the components composer.json file:

"src/Http/helpers.php"

### Step 7

Registering the Service Provider

Open up config/app.php. In the providers array, enter the following:

EONConsulting\PHPStencil\PHPStencilServiceProvider::class,

### Step 8

In the same file, in the aliases array, enter:

'PHPStencil' => \EONConsulting\PHPStencil\src\Facades\PHPStencil::class,

To be able to use the tsugi, the APP\_URL in your env file needs to be correct. So change it to the URL you are using for your testing environment.

### Step 9

In the command line / terminal, enter the following command in the root of the Laravel project:

composer dump-autoload -o

We have included a dummy\_table in the database for testing purposes. To have access to the migrations, you will need to perform the following commands:

php artisan vendor:publish

php artisan migration

# How to use the PHPStencil

Let's see this in action. Go to routes/web.php and where it says return view('welcome');replace that with

echo PHPStencil::output();

// or

echo phpstencil()->output(); // if you did step 2.4

To test this out, go to the URL for this Laravel site.

You should now see the result "Hello World" being printed out to the screen.

## Outputting Text

We have provided 3 standard output formats: JSON, CSV and XML.

## Using JSON

First we need to create the factory:

$factory = new TextFactory(new AdapterFactory);

Next we need to set the adapter using the make method and passing through which adapter we would like.

$text = $factory->make(TextEnum::JSON);

Now, just call output and pass through the data you need.

$text->output($data);

So this is all we have:

$factory = new TextFactory(new AdapterFactory);

$text = $factory->make(TextEnum::JSON);

$text->output($data);

The $data variable would be an array you wish to make into JSON.

## Using CSV

We need to do the exact same as above, except for this, we need to change the TextEnum::JSON to TextEnum::CSV. Like so:

$text = $factory->make(TextEnum::CSV);

So all the code is:

$factory = new TextFactory(new AdapterFactory);

$text = $factory->make(TextEnum::CSV);

$text->output($data);

THe $data variable expects an array of arrays. Each array represents a row, and each item in the array represents a column.

## Using XML

Again, do the same as above and just change the TextEnum::CSV to TextEnum::XML.

So all the code is:

$factory = new TextFactory(new AdapterFactory);

$text = $factory->make(TextEnum::XML);

$text->output($data);

Here, the $data variable will convert an array of objects into XML. Something like this:

$xml = [

'folders' => [

['name' => 'Folder 1', 'id' => '1', 'files' => [

['name' => 'File 1', 'id' => '1', 'type' => 'psd'],

['name' => 'File 2', 'id' => '2', 'type' => 'csv'],

['name' => 'File 3', 'id' => '3', 'type' => 'pdf'],

]],

['name' => 'Folder 2', 'id' => '2'],

['name' => 'Folder 3', 'id' => '3', 'files' => [

['name' => 'File 1', 'id' => '1', 'type' => 'txt'],

['name' => 'File 2', 'id' => '2', 'type' => 'csv'],

]],

],

'files' => [

['name' => 'My public file']

]];

## The GUI

We have provided two ways of getting a GUI. Either a form or a list. I will take you through how to add more as well.

## Form

To create a form element, we need to start by creating the factory.

$factory = new GUIFactory(new AdapterFactory);

Now we need to specify which type of view we would like to use. For form, we need to use the GUIEnum::FORM constant.

$gui = $factory->make(GUIEnum::FORM);

And finally, we need to return the view with the following line of code:

return $gui->render($data);

So all in all, to create a form, we have:

$factory = new GUIFactory(new AdapterFactory);

$gui = $factory->make(GUIEnum::FORM);

return $gui->render($data);

## List

We use the exact same code as above, except now, we will use the GUIEnum::UILISTconstant.

**Note: we use the name UILIST and not LIST because LIST is a used keyword.**

$factory = new GUIFactory(new AdapterFactory);

$gui = $factory->make(GUIEnum::UILIST);

return $gui->render($data);

## Output REST Web Services

The base route it /\_eon\_phpstencil/api/rest.

To create your own routes, place them in the routes\_rest.php file under the src/Factories/WebService/Routes/ directory.

Create your controller in the src/Factories/WebService/REST/Controllers and extend the RestBaseController class.

Extending this class will give you access to two methods:

return\_success

return\_error

## return\_success

The return\_success method has two parameters. $messages and $more\_data. The second parameter is optional. You can pass through any success messages you like for the $messages parameter, and then any data you wish to present alongside the response with the $more\_data parameter.

Note: Both $messages and $more\_data are arrays.

## return\_error

The return\_error method only accepts one parameter. $messages is an array that will hold all error messages in your response.

## Returning content to the user

To return content to the user (from the controller), you need to wrap the return\_successand return\_error methods in a json response. Like this:

return response()->json($this->return\_error(['Item does not exist.'])); // request error

return response()->json($this->return\_success(['Item deleted.'])); // request successful

## The Observer Pattern

### Introduction

The observer pattern is very useful, and you will use it a lot for certain tasks across all types of applications. An observer, is an object (called the subject) that maintains a list of all of its dependents (called observers).

If a state change occurs to the main object, it will notify the dependents by calling one of their methods.

To make use of this pattern, we have made it easy for you. There are 2 steps.

1. Create a class that extends the Event class in the EONConsulting\PHPStencil\Eventsnamespace.
2. Create your observer that implements the SplObserver class that is part of the [SPL (Standard PHP Library)](http://php.net/manual/en/book.spl.php)

### Step 1 - Extend Event

Create your normal class that will hold your object. All you need to do is extend the Eventclass. If you have a constructor, do not overwrite the parent constructor (the constructor in the Event class) by calling parent::\_\_construct(); at the top of your constructor.

### Step 2 - Implement \SplObserver

Create your observer class. Here you need to do 2 things.

1. Implement SplObserver using the following command: implements \SplObserver
2. Make sure you have a method called update that has a parameter of \SplSubject $event - you can label this $event variable whatever you like - as long as it has the typehint of SplSubject.

### Using the pattern

Now, create the object like you would any other.

$event = new ExampleClass;

Then we need to attach the observers you wish to call.

$event->attach(new ObserverClassOne);

$event->attach(new ObserverClassTwo);

$event->attach(new ObserverClassThree);

And finally, when you would like to call the observers, go ahead and execute the following line of code:

$event->notify();

Set it Tsugi with tolls compatible LTI 2.0

Check the code out from GitHub and put it in a directory where your web server can read it

git clone https://github.com/csev/tsugi.git

We have provided a very easy way to use Tsugi.

You just need to setup the keys in your .env file. Like so:

TSUGI\_DB=tsugi

TSUGI\_DB\_HOST=127.0.0.1

TSUGI\_DB\_USER=root

TSUGI\_DB\_PASS=root

TSUGI\_DB\_PREFIX=""

TSUGI\_ADMIN\_PASS=1234

TSUGI\_SERVICE\_NAME="TSUGI"

TSUGI\_SERVICE\_DESC=""

TSUGI\_OWNER\_NAME="Charles Severance"

TSUGI\_OWNER\_MAIL=""

TSUGI\_PROVIDE\_KEYS=false

TSUGI\_GOOGLE\_CLIENT\_ID=false

TSUGI\_GOOGLE\_CLIENT\_SECRET=false

TSUGI\_GOOGLE\_MAP\_API\_KEY=false

TSUGI\_BADGE\_ENCRYPT\_PASSWORD=false

TUSIG\_BADGE\_ASSERT\_SALT=false

TSUGI\_PRODUCT\_INSTANCE\_GUID="lti2.example.com"

TSUGI\_DEVELOPER=true

TSUGI\_COOKIE\_SECRET="warning:please-change-cookie-secret-a289b543"

TSUGI\_COOKIE\_NAME="TSUGIAUTO"

TSUGI\_COOKIE\_PAD="390b246ea9"

TSUGI\_MAIL\_DOMAIN=false

TSUGI\_MAIL\_SECRET="warning:please-change-mailsecret-92ds29"

TSUGI\_MAILEOL="\n"

TSUGI\_NONCE\_CHECK=100

TSUGI\_NONCE\_TIME=1800

TSUGI\_SESSION\_SALT="warning:please-change-sessionsalt-89b543"

TSUGI\_TIMEZONE="Pacific/Honolulu"

TSUGI\_OLD\_ANALYTICS\_KEY=false

TSUGI\_OLD\_ANALYTICS\_NAME=false

TSUGI\_UNIVERSAL\_ANALYTICS=false

TSUGI\_OFFLINE=false

Fill out these config details to best suit your project. If you do not know what the key means or what to change it to, leave it to its default value above.

## Configuration

### Tsugi Config

We have provided a very easy way to use Tsugi.

You just need to setup the keys in your .env file. Like so:

TSUGI\_DB=tsugi

TSUGI\_DB\_HOST=127.0.0.1

TSUGI\_DB\_USER=root

TSUGI\_DB\_PASS=root

TSUGI\_DB\_PREFIX=""

TSUGI\_ADMIN\_PASS=1234

TSUGI\_SERVICE\_NAME="TSUGI"

TSUGI\_SERVICE\_DESC=""

TSUGI\_OWNER\_NAME="Charles Severance"

TSUGI\_OWNER\_MAIL=""

TSUGI\_PROVIDE\_KEYS=false

TSUGI\_GOOGLE\_CLIENT\_ID=false

TSUGI\_GOOGLE\_CLIENT\_SECRET=false

TSUGI\_GOOGLE\_MAP\_API\_KEY=false

TSUGI\_BADGE\_ENCRYPT\_PASSWORD=false

TUSIG\_BADGE\_ASSERT\_SALT=false

TSUGI\_PRODUCT\_INSTANCE\_GUID="lti2.example.com"

TSUGI\_DEVELOPER=true

TSUGI\_COOKIE\_SECRET="warning:please-change-cookie-secret-a289b543"

TSUGI\_COOKIE\_NAME="TSUGIAUTO"

TSUGI\_COOKIE\_PAD="390b246ea9"

TSUGI\_MAIL\_DOMAIN=false

TSUGI\_MAIL\_SECRET="warning:please-change-mailsecret-92ds29"

TSUGI\_MAILEOL="\n"

TSUGI\_NONCE\_CHECK=100

TSUGI\_NONCE\_TIME=1800

TSUGI\_SESSION\_SALT="warning:please-change-sessionsalt-89b543"

TSUGI\_TIMEZONE="Pacific/Honolulu"

TSUGI\_OLD\_ANALYTICS\_KEY=false

TSUGI\_OLD\_ANALYTICS\_NAME=false

TSUGI\_UNIVERSAL\_ANALYTICS=false

TSUGI\_OFFLINE=false

Fill out these config details to best suit your project. If you do not know what the key means or what to change it to, leave it to its default value above.

Appendix A

For Appendices use the style “UEL Appendix”