

# Créer une application avec Laravel 5.5 – Les langues

Dans ce chapitre on va s'intéresser à l'aspect multi-langage. Pour le moment notre galerie est en français mais on a fait en sorte que les textes soient faciles à traduire en utilisant dans le code les helpers de Laravel. On va donc ajouter maintenant l'anglais à notre galerie. Ça ne concernera évidemment que l'interface et pas les données, ce qui serait une autre histoire...

## La configuration

Dans le fichier **config/app.php** on a des réglages pour les langues :

```
'locale' => 'fr',
```

```
'fallback_locale' => 'en',
```

On a fixé la locale au français (**fr**) et la langue par défaut en cas d'absence de traduction à l'anglais (**en**).

On va ajouter un réglage avec les langues qu'on va mettre en œuvre et qui devront avoir les traductions présentes :

```
'locales' => ['fr', 'en'],
```

## Route, contrôleur et middleware

### Route

On ajoute la route pour le changement de la langue :

```
Route::name('language')->get('language/{lang}',  
'HomeController@language');
```

# Contrôleur

Et on ajoute la fonction dans **HomeController** :

```
public function language(String $locale)
{
    $locale = in_array($locale, config('app.locales')) ? $locale :
config('app.fallback_locale');

    session(['locale' => $locale]);

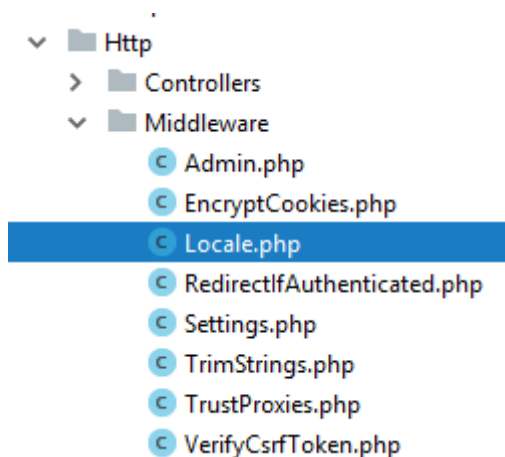
    return back();
}
```

On reçoit une locale en paramètre. Si elle est présente dans le tableau de la configuration on fixe cette locale en session, sinon on se rabat sur la locale par défaut.

# Middleware

Il nous faut maintenant un middleware qui va vérifier si on a une locale en session pour réellement l'affecter. Si ce n'est pas le cas essayer de définir la langue de l'utilisateur. On va aussi fixer la locale pour les dates.

php artisan make:middleware Locale



Et on code ainsi :

```
<?php
```

```
namespace App\Http\Middleware;
```

```

use Closure;

class Locale
{
    public function handle($request, Closure $next)
    {
        if(!session()->has('locale')) {
            session(['locale' =>
$request->getPreferredLanguage(config('app.locales'))]);
        }

        app()->setLocale(session('locale'));

        setlocale(LC_TIME, session('locale'));

        return $next($request);
    }
}

```

*Sur un serveur en production vous n'allez pas avoir une langue déclarée avec juste **fr** ou **en**. Ça ne marchera donc pas avec ce code. Il faut d'abord vérifier les langues installées avec **locale -a**. Ensuite on peut créer un tableau de conversion dans le *middleware* :*

```

$locale = session('locale');

$conversion = [
    'fr' => 'fr_FR',
    'en' => 'en_US',
];

$locale = $conversion[$locale];

setlocale(LC_TIME, $locale);

```

Et on le référence dans **app/Http/Kernel** :

```

protected $middlewareGroups = [
    'web' => [
        ...

        \App\Http\Middleware\Locale::class,
        \App\Http\Middleware\Settings::class,
    ],

```

```

    ],
    ...
];

```

## Le menu

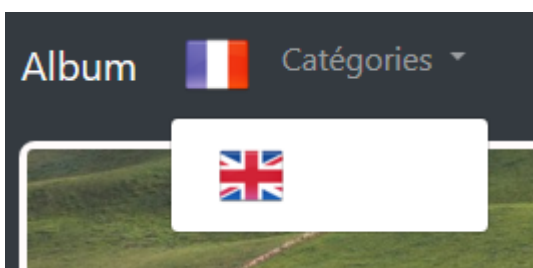
On va ajouter un menu déroulant pour le choix de la locale dans **views/layouts/app** :

```

<ul class="navbar-nav mr-auto">
  <li class="nav-item dropdown">
    <a class="nav-link" href="#" id="navbarDropdownFlag"
    role="button" data-toggle="dropdown" aria-haspopup="true" aria-
    expanded="false">
      
    </a>
    <div id="flags" class="dropdown-menu" aria-
    labelledby="navbarDropdownFlag">
      @foreach(config('app.locales') as $locale)
        @if($locale != session('locale'))
          <a class="dropdown-item" href="{{ {{
    route('language', $locale) }}">
            
          </a>
        @endif
      @endforeach
    </div>
  </li>

```

Et on va voir le résultat :

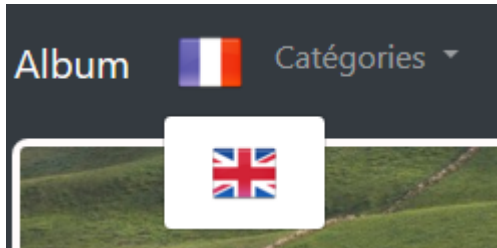


Je ne trouve pas trop élégant la largeur de la zone pour le drapeau. On rectifie ça dans **assets/css/app.css** :

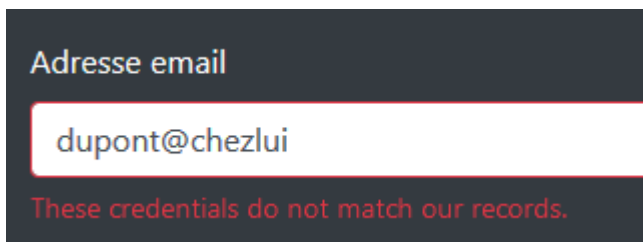
```
.dropdown-menu {  
    min-width: 5rem;  
}
```

On lance npm...

Et c'est maintenant plus équilibré :



Pour le moment le changement de langue ne se voit que dans les validations :



## Un package

Il nous faut créer un fichier JSON avec toutes les traductions pour l'anglais. On pourrait faire ça en explorant tout le code avec des copier/coller, ça serait vraiment laborieux !

On va plutôt utiliser un package pour nous aider. Comme je n'en ai pas vraiment trouvé un qui me plaise [j'en ai créé un](#). On va commencer par l'installer :

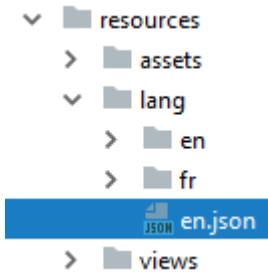
```
composer require bestmomo/laravel5-artisan-language --dev
```

On a maintenant 4 commandes de plus dans artisan :

```
language
language:diff      Show differences with locale
language:make      Create a new json language file
language:strings   List all default language strings
language:sync      Synchronise differences for the locale
```

On va utiliser la deuxième :

php artisan language:make en



Le fichier JSON a été créé et on a tous les textes par ordre alphabétique qui attendent leur traduction :

```
{
  "Administration": "",
  "Adresse email": "",
  "Ajouter une catégorie": "",
  "Ajouter une image": "",
  "Catégorie": "",
  ...
}
```

On ajoute donc les traductions :

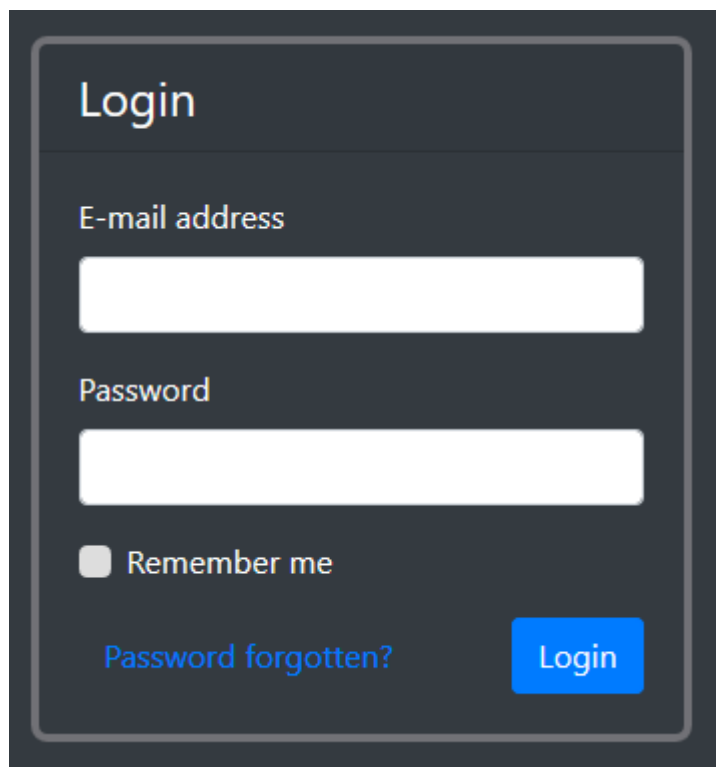
```
{
  "Administration": "Administration",
  "Adresse email": "E-mail address",
  "Ajouter une catégorie": "Add a category",
  "Ajouter une image": "Add an image",
  "Catégorie": "Category",
  "Catégories": "Categories",
  "Cette page n'existe pas": "This page doesn't exist",
  "Confirmation du mot de passe": "Password confirmation",
  "Connexion": "Login",
  "Déconnexion": "Logout",
  "Description (optionnelle)": "Description (optional)",
  "Envoi de la demande": "Send demand",
  "Envoyer": "Send",
}
```

"Erreur 403": "Error 403",  
"Erreur 404": "Error 404",  
"Erreur 503": "Error 503",  
"Gestion des catégories": "Categories gestion",  
"Gérer les catégories": "Categories gestion",  
"Il semble y avoir une erreur sur le serveur, veuillez réessayer plus tard...": "Looks like there is a server issue, please try later...",  
"image orpheline|images orphelines": "orphan image|orphans images",  
"images par page": "images per page",  
"Inscription": "Registration",  
"L'image a bien été enregistrée": "Image has been saved",  
"La catégorie a bien été enregistrée": "Category has been saved",  
"La catégorie a bien été modifiée": "Category has been updated",  
"Le profil a bien été mis à jour": "Profile has been updated",  
"Maintenance": "Maintenance",  
"Modifier le profil": "Profile update",  
"Modifier la catégorie": "Update the category",  
"Modifier une catégorie": "Update a category",  
"Mot de passe": "Password",  
"Mot de passe oublié ?": "Password forgotten?",  
"Nom": "Name",  
"Non": "No",  
"Oui": "Yes",  
"Pagination : ": "Pagination: ",  
"Photos de ": "Photos from ",  
"Profil": "Profile",  
"Renouvellement du mot de passe": "Password reset",  
"Renouveler": "Reset",  
"Se rappeler de moi": "Remember me",  
"Service temporairement indisponible ou en maintenance": "The server is currently unavailable",  
"Supprimer": "Delete",  
"Supprimer cette photo": "Delete this photo",  
"Supprimer la catégorie": "Delete the category",  
"Voir les photos de ": "Show photos from ",  
"Vos droits d'accès ne vous permettent pas d'accéder à cette ressource": "You might not have the necessary permissions for this resource",  
"Vraiment supprimer cette catégorie ?": "Really delete this

```
category?",
```

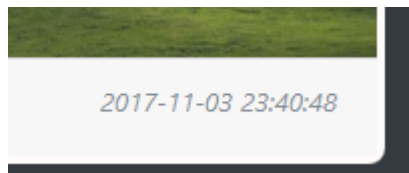
```
    "Vraiment supprimer toutes les photos orphelines ?": "Really  
delete all orphans images?"  
}
```

Maintenant si on passe à l'anglais on a bien les textes dans cette langue :



## Les dates

Pour le moment les dates ne sont pas très élégantes :



Et elles sont toujours au format américain. On a prévu ce code dans **views/home** :

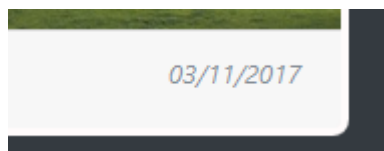
```
{{ $image->created_at }}
```

On va se servir [d'une méthode de Carbon](#) pour arranger ça (on va négliger l'heure) :

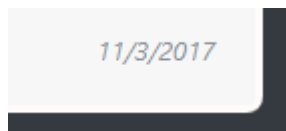
```
{{ $image->created_at->formatLocalized('%x') }}
```



Maintenant en français j'obtiens :



Et en anglais :



## Conclusion

Dans ce chapitre on a :

- prévu la configuration pour les locales
- ajouté la route, la fonction du contrôleur et un middleware pour le changement de locale
- ajouté un menu déroulant avec les drapeaux des langues disponibles
- installé un package pour créer le fichier de la nouvelle langue et ajouté ainsi les traductions
- adapté les dates à la locale dans la vue de la galerie

Pour vous simplifier la vie vous pouvez [charger le projet](#) dans son état à l'issue de ce chapitre.