Introduction à Composer et SF 5

composer, autoload, routage

Dans ce TD, vous allez vous familiariser avec les composants du standard des Frameworks PHP : Symfony en commençant par quelques composants de base.

Une introduction générale à la version actuelle de symfony se trouve ici :
`http://symfony.com/doc/current/index.html`

Exercice 1. composer, class-loader et http-foundation

Installons d'abord composer s'il n'est pas déjà présent sur notre OS. On va d'abord créer un répertoire `bin` à la racine de notre HOME :

```
cd
mkdir bin
cd bin
curl -s https://getcomposer.org/installer | php
```

Vérifiez la définition de vos variables d'environnement http_proxy et https_proxy dans votre `.bashrc` Ajoutez également la ligne suivante à votre .bashrc :

```
export PATH=$PATH:~/bin
```

de manière à ce que tous les programmes installés dans le répertoire bin de votre HOME soient accessibles de n'importe où. Puis créez un répertoire de travail sf4-test dans votre dossier /www avec à l'intérieur un fichier composer.json :

```
{
    "require": {
        "symfony/class-loader": "4.2.*",
        "symfony/http-foundation": "4.2.*"
    }
}
```

Ceci indique que nous n'installons pour l'instant que ces 2 composants. Puis utiliser composer pour installer les composants demandés :

```
composer update
```

ou

```
composer update -o
```

L'utilisation de l'option -o de composer pour optimize-autoloader qui optimise « au mieux » le chargement automatique des classes.

On peut aussi simplement taper :

```
composer require symfony/http-foundation
```

L'autoloader et les 2 principaux composants de HttpFoundation à savoir `Request` et `Response` sont prêts à l'emploi. Remarquez l'usage des espaces de nommages en PHP semblables à ceux du C++ ou aux import de packages en java.

```php
<?php
require_once __DIR__.'vendor/autoload.php';

use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;

//$request = Request::createFromGlobals();

$request = Request::create('/essai.php?name=Zozo');

// URI demandee (sans les parametres)
$path=$request->getPathInfo();

// recup GET de variables
$nom=$request->query->get('name','World');
$prenom=$request->query->get('surname','Joe');
echo "Bonjour $surname $name<br/>";
```

On peut aussi récupérer d'autres informations sur le Client et fabriquer une réponse :

```php
<?php
// recup  variables SERVER
$host=$request->server->get('HTTP_HOST');

// recup  COOKIES
$request->cookies->get('PHPSESSID');

// HTTP headers
$headers=$request->headers->get('host');
$content_type=$request->headers->get('content_type');

```

```
12 $method=$request->getMethod();  //GET, POST, PUT, DELETE ou
     HEAD
13 $langs=$request->getLanguages();
14 $IP == $request->getClientIp();
15 $response =
16 new Response($IP." ".$host." ".$path." ".$headers."
17 ".$content_type." ".$method." ".$nom." ".$langs[0]);
18 $response->send();
```

Exercice 2. installer twig

Pour installer twig, on fait simplement :

```
composer require twig
```

Voir TD5 pour son utilisation.

Exercice 3. créer une application SF5

Pour créer un squellette d'application avec un nombre réduit de dépendances :

```
composer create-project symfony/skeleton hello-sf5
```

Pour une application plus complète (ne le faites pas pour le moment)

```
composer create-project symfony/website-skeleton sf5-full-
   project
```