**Les Services**

1. Qu’est-ce qu’un service :

Un service est simplement un objet PHP qui remplit une fonction, associé à une configuration.

1. Comment créer un service :

Un service c’est une classe simple :

**<?php****namespace** CoreBundle\Service;  
  
**use** CoreBundle\Entity\Brand;  
**use** Doctrine\ORM\EntityManager;  
**use** Doctrine\ORM\EntityRepository;  
  
**class** BrandService  
{  
 */\*\*  
 \** ***@var*** *EntityRepository  
 \*/* **private** $repository;  
  
 **public function** \_\_construct(EntityManager $em)  
 {  
 $this->repository=$em->getRepository(Brand::*class*);  
 }  
 */\*\*  
 \* Retourner la liste des marques  
 \* return array  
 \*/* **public function** getBrands(){  
 **return** $this->repository->findAll();  
 }  
 */\*\*  
 \* Retourner la liste des marques selon la variable $active  
 \* $active prend 2 valeurs true ou false  
 \* return array  
 \*/* **public function** getActiveBrands($active){  
 **return** $this->repository->getActiveBrands($active);  
 }  
 */\*\*  
 \* Retourner un seul marque selon l'id  
 \*  
 \*/* **public function** getBrand($id){  
 **return** $this->repository->find($id);  
 }  
}

1. La configuration :

On doit configurer le service dans le fichier services.yml pour qu’on puisse l’utilisé dans le contrôleur et que le Service container connaitre le service a utilisé :

**services:  
 core.service.brand:  
 class:** CoreBundle\Service\BrandService  
 **arguments:** ['@doctrine.orm.entity\_manager']

Dans cette configuration :

**core.service.brand** : est le nom de notre service fraîchement créé. Essayez de respecter la convention en préfixant le nom de vos services par le nom du bundle, ici « core ».

**class** : est un attribut obligatoire de notre configuration, il définit simplement le namespace complet de la classe du service. Cela indique au conteneur de services quelle classe instancier lorsqu'on lui demandera le service.

**arguments :** est un tableau d'arguments

**Les arguments peuvent être :**

Des valeurs normales en YAML (des booléens, des chaînes de caractères, des nombres, etc.) ;

Des paramètres (définis dans le parameters.yml par exemple) : l'identifiant du paramètre est encadré de signes « % » **:%nomDuParametre%;**

Des services : l'identifiant du service est précédé d'une arobase **:@nomDuService**.

1. L’utilisation du service :

Dans un contrôleur juste on fait :

$brandService = $this->get('core.service.brand');