Convention de nommage

Table des matières

[Indentation : 2](#_Toc440826785)

[Règle de nommage pour Symfony 2 : 2](#_Toc440826786)

[Les Bundles : 2](#_Toc440826787)

[Les entités (classe) : 2](#_Toc440826788)

[Les view (Twig) : 3](#_Toc440826789)

[Règle de nommage pour Android : 4](#_Toc440826790)

[Hiérarchie des fichiers : 4](#_Toc440826791)

# Indentation :

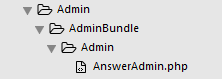
L’indentation par défaut du projet sera d’une tabulation ou 4 espaces selon les fichiers.

# Règle de nommage pour Symfony 2 :

## **Les Bundles :**

Les bundles sont structurés de façon hiérarchique avec des règles précises :

* **Nom principal :** Première lettre en majuscule le reste en minuscule, ne se compose que d’un mot.
* **Nom secondaire :** Première lettre en majuscule le reste en minuscule, se compose d’un ou plusieurs mots qui résument l’utilité de celui-ci. Se termine par le mot « Bundle ». Aucun caractères spéciaux ou espace n’est accepté.
* **Dossier interne au Bundle :** Première lettre en majuscule le reste en minuscule, possibilité d’avoir plusieurs mots collés en mettant la première lettre de celui-ci en majuscule.



## Les entités (classe) :

* **Nom :** 
  + Première lettre en majuscule le reste en majuscule :

« Question.php ».

* + Dans le cas où deux mots serait nécessaires pour caractériser une entité, une majuscule à la première lettre du deuxième mot est nécessaire :

« GroupUser.php ».

* + Enfin si cette entité est une entité de liaison (n ; n), une séparation avec un Under-Score est nécessaire :

« User\_Qcm.php ».

* **Namespace :** Le Namespace indique l’emplacement du fichier il s’inscrit d’une façon simple, nom des dossiers avec comme séparation un antislash :

« namespace QCM\WebserviceBundle\Entity; »

* **Nom de la classe :** Elle est identique au nom mais ne prends pas le type de fichier :

« Question ».

* **Nom d’une propriété :** Commence en minuscule et majuscule seulement à la première lettre du deuxième mot si celui-ci est composé :

« name » ou « isAvailable ».

* **Nom de la colonne :** Tous les mots en minuscule. Si deux mots sont présents un Under-Score est nécessaire entre le premier et le deuxième mot mais ils restent en minuscule :

« name » ou « is\_available ».

* **Accesseur :** Le nom des méthodes commencent par une minuscule et ajoute une majuscule à la première lettre de chaque nouveau mot de celle-ci :

«  public function setIsAvailable() »

## Les view (Twig) :

* **Nom :** Le nom des view est inscrit en minuscule et séparé par un Under-Score quand il est composé de plusieurs mots. Il est ajouté à la fin « .html.twig » afin de préciser le type de fichier :

«  view\_substance.html.twig »

# Règle de nommage pour Android :

## Hiérarchie des fichiers :

* **Nom du projet :** Première lettre en majuscule puis première lettre de chaque mot complémentaire également en majuscule :

« IiaFrameworks »

* **Sous-dossier :** Les dossiers ou packages à l’intérieur du projet sont tous écrit en minuscule et ne contiennent pas d’Under-Score :

« push »

* **Classe :** Le nom des classes commencent par une majuscule et chaque mot complémentaire également. Il est souvent significatif de ce qu’il y a l’intérieur de celui-ci :

« GcmBroadcastReceive »

* **Nom des méthodes et nom des paramètres :** Le nom des méthodes commence par une minuscule et chaque première lettre d’un mot supplémentaire est inscrite en Majuscule :

« onCreate(Bundle savedInstanceState) »

* **Nom de variable :** Le nom des variables commence par une minuscule et chaque première lettre d’un mot supplémentaire est inscrite en majuscule.

« private TextView longView »

* **Nom des constantes :** Le nom des constantes est inscrit en majuscule et lorsqu’il est composé de plusieurs mots ils sont séparés par un Under-Score :

« private static final String PROJECT\_ID = ‘’123456’’; »

# Règle de nommage pour IOS :

En attente du cours d’IOS.

# Règle de nommage pour Windows Phone :

En attente de cours de Windows Phone.