



# D07 - Formation Python-Django

## Django - Advanced

Damien Cojan [dcojan@student.42.fr](mailto:dcojan@student.42.fr)  
42 Staff [pedago@staff.42.fr](mailto:pedago@staff.42.fr)

*Résumé: Aujourd'hui nous allons découvrir quelques fonctionnalités avancées de Django.*

# Table des matières

<b>I</b>	<b>Préambule</b>	<b>2</b>
<b>II</b>	<b>Consignes</b>	<b>5</b>
<b>III</b>	<b>Règles spécifiques de la journée</b>	<b>7</b>
<b>IV</b>	<b>Exercice 00</b>	<b>8</b>
<b>V</b>	<b>Exercice 01</b>	<b>10</b>
<b>VI</b>	<b>Exercice 02</b>	<b>12</b>
<b>VII</b>	<b>Exercice 03</b>	<b>14</b>
<b>VIII</b>	<b>Exercice 04</b>	<b>16</b>
<b>IX</b>	<b>Exercice 05</b>	<b>17</b>
<b>X</b>	<b>Exercice 06</b>	<b>18</b>

# Chapitre I

## Préambule

Le bonheur des frameworks

J'ai voulu construire une petite étagère pour y ranger les condiments.

Ayant fait un peu de menuiserie avant, j'avais une bonne idée de ce dont j'avais besoin : un peu de bois et quelques outils de base. Un mètre, une scie, un niveau et un marteau.

D'ailleurs, si je voulais construire toute une maison, j'en aurais besoin également. Du coup je suis allé dans une quincaillerie, et j'ai demandé au vendeur où je pouvais trouver un marteau.

“- Un marteau?”, me répondit-il. “Plus personne n'achète des marteaux de nos jours vous savez. Ils sont un peu vieux jeu.”

Surpris, je lui demande pourquoi.

– “Et bien, le problème avec les marteaux, c'est qu'il y en a plein de différents types. Des marteaux arrache-clou, des masses, des marteaux de tapissier... Que se passerait-il si vous achetiez un type de marteau et réalisiez que vous avez besoin d'un autre type plus tard? Vous devriez acheter un autre marteau pour votre prochaine tâche. Il se trouve que la plupart des gens veulent vraiment un seul marteau qui peut être utilisé pour la majorité des tâches qu'ils peuvent rencontrer dans leur vie.”

– “Ça me paraît logique. Pouvez-vous me dire où je peux trouver un marteau universel?”

– “Non, nous ne les vendons plus. Ils sont obsolètes.”

– “Vraiment? Je pensais que vous veniez de dire que le marteau universel était l'avenir.”

– “Il se trouve que, si vous faites un seul marteau qui puisse être utilisé pour toutes sortes de tâches, il n'est vraiment bon, à aucune d'entre elles. Enfoncer un clou avec une

masse n'est pas très efficace. Et pour tuer votre petite amie, rien ne vaut un marteau de tapissier."

– "C'est clair ! Donc, si plus personne n'achète des marteaux universels, et que vous ne vendez plus de marteaux à l'ancienne, quels marteaux vendez-vous ?"

– "En fait, nous n'en vendons pas."

– "Alors..."

– "D'après nos recherches, ce dont les gens ont besoin n'est pas un marteau universel du tout. Il vaut toujours mieux avoir le bon marteau pour le bon boulot. Donc, nous avons commencé à vendre des fabriques de marteau, capable de produire n'importe quel marteau qui pourrait vous intéresser. Tout ce dont vous avez besoin est de remplir la fabrique de travailleurs, lancer la machinerie, acheter les matériaux de base, payer les charges et hop, vous avez \*exactement\* le type de marteau dont vous avez besoin en un clin d'œil."

– "Mais je ne veux pas acheter une fabrique de marteaux..."

– "Parfait. Car nous n'en vendons plus."

– "Attendez, vous venez de me dire que..."

– "Nous avons découvert que la plupart des gens n'ont pas besoin d'une fabrique complète de marteaux. Certains, par exemple, n'auront jamais besoin d'un marteau de tapissier. (Peut être qu'ils n'ont pas d'ex. Ou peut être qu'ils les ont tué avec des pics à glace.). Donc il n'y a aucune raison pour quelqu'un d'acheter une fabrique de marteaux pour tous les types de marteaux."

– "Oui, c'est sûr."

– "Donc, à la place, on a commencé à vendre les plans de constructions de la fabrique de marteaux, afin que nos clients puissent construire leurs propres fabriques, complètement personnalisées pour produire uniquement les types de marteaux dont ils ont besoin."

– "Laissez-moi deviner. Vous ne les vendez plus."

– "Non. Bien entendu. Il se trouve que les gens ne veulent pas construire toute une fabrique juste pour faire quelques marteaux. Laissez la construction des fabriques aux experts de construction de fabriques, c'est ce que je dis toujours !!"

– "Et je vous approuve sur ce point."

– "Et oui. Donc nous avons arrêté de vendre ces plans et nous avons commencé à vendre des fabriques de fabriques de marteaux. Chacune d'elle est construite par nos experts dans le business de fabrique de fabrique de marteaux, afin que vous n'ayez pas à vous inquiéter des détails triviaux de la construction d'une fabrique. Malgré cela, vous

avez tous les bénéfices d'avoir votre propre fabrique personnalisée, produisant vos propres marteaux personnalisés, collant à vos designs spécifiques en matière de marteau.”

– “Heu, ça ne me semble pas vraiment...”

– “Je sais ce que vous allez dire!! ... et nous ne les vendons d'ailleurs plus. Apparemment, peu de gens achetaient ces fabriques de fabrique de marteaux, donc nous avons trouvé une solution à ce problème.”

– “Hum.”

– “Nous avons pris le temps de faire le bilan de notre infrastructure technique, et nous avons déterminé que les gens développaient une frustration à avoir à gérer et opérer une fabrique de fabrique de marteaux, tout comme la fabrique qu'elle produisait. Ce genre de contrainte additionnelle peut se révéler fastidieux quand vous vous retrouvez dans un scénario où vous utilisez également une fabrique de fabrique de mètres, une fabrique de fabrique de scies et une fabrique de fabrique de niveaux. Sans compter un conglomérat de transformation du bois. Nous avons objectivement évalué la situation, et déterminé que c'était trop complexe pour quelqu'un qui voulait juste créer une étagère pour condiments”.

– “Non, sans blague?”

– “Du coup cette semaine, nous mettons sur le marché une fabrique de fabrique de fabrique de création d'outils en tout genre, pour qu'ainsi vos différentes fabriques de fabrique à outils puissent être créées à partir d'une seule fabrique unifiée. La fabrique de fabrique de fabrique produira uniquement la fabrique de fabrique dont vous avez réellement besoin, et ainsi ces fabriques de fabrique produiront une seule fabrique basée sur vos spécifications d'outils personnalisés. Vous aurez *\*exactement\** le marteau dont vous avez besoin, et exactement le bon mètre pour votre tâche, juste en appuyant sur un bouton (même si vous aurez probablement quelques fichiers de configuration pour que tout fonctionne selon vos attentes).

– “Donc, vous n'avez pas de marteaux? Pas du tout?”

– “Non. Si vous voulez vraiment une étagère à condiments de haute qualité, de standard industriel, vous avez vraiment besoin de quelque chose de plus sophistiqué qu'un simple marteau acheté à la quincaillerie du coin.”

– “Ok... Bon. Il faut ce qu'il faut. Si c'est comme ça qu'on fait maintenant, il faut bien que je m'y mette.”

– “Excellent!!”

– “Ça vient avec une documentation, pas vrai?”

Source : [SametMax](#)

# Chapitre II

## Consignes

- Seule cette page servira de référence : ne vous fiez pas aux bruits de couloir.
- Le sujet peut changer jusqu'à une heure avant le rendu.
- Si aucune information contraire n'est explicitement présente, vous devez partir du principe que les versions des langages et framework utilisés sont les suivantes (ou ultérieures) :
  - Python 3
  - HTML5
  - CSS 3
  - Javascript ES6
  - Django 1.9
  - psycopg2 2.6
- Sauf indication contraire dans le sujet, les fichiers en python de chaque exercice sur Python seul (d01, d02 et d03) doivent comporter à leur fin un bloc `if __name__ == '__main__':` afin d'y insérer le point d'entrée dans le cas d'un programme, ou des tests dans le cas d'un module.
- Sauf indication contraire dans le sujet, chaque exercice des journées portant sur Django aura sa propre application dans le projet à rendre pour des raisons pédagogiques.
- Les exercices sont très précisément ordonnés du plus simple au plus complexe. En aucun cas nous ne porterons attention ni ne prendrons en compte un exercice complexe si un exercice plus simple n'est pas parfaitement réussi.
- Attention aux droits de vos fichiers et de vos répertoires.
- Vous devez suivre la procédure de rendu pour tous vos exercices : seul le travail présent sur votre dépôt GIT sera évalué en soutenance.
- Vos exercices seront évalués par vos camarades de piscine.
- Vous ne devez laisser dans votre répertoire aucun autre fichier que ceux explicitement spécifiés par les énoncés des exercices.
- Sauf indication contraire dans le sujet vous ne devez pas inclure dans votre rendu :

- Les dossiers `__pycache__`.
- Les éventuelles migrations.
- Le dossier créé par la commande `collectstatic` de `manage.py` (avec pour chemin la valeur de la variable `STATIC_ROOT`).
- Les fichiers en bytecode Python (Les fichiers avec une extension en `.pyc`).
- Les fichiers de base de donnée (notamment avec `sqlite`).
- Tout fichier ou dossier devant ou pouvant être créé par le comportement normal du travail rendu.

Il vous est recommandé de modifier votre `.gitignore` afin d'éviter les accidents.

- Lorsque vous devez obtenir une sortie précise dans vos programmes, il est bien entendu interdit d'afficher une sortie précalculée au lieu de réaliser l'exercice correctement.
- Vous avez une question ? Demandez à votre voisin de droite. Sinon, essayez avec votre voisin de gauche.
- Votre manuel de référence s'appelle `Google / man / Internet / ....`
- Pensez à discuter sur le forum Piscine de votre Intra !
- Lisez attentivement les exemples. Ils pourraient bien requérir des choses qui ne sont pas autrement précisées dans le sujet...
- Par pitié, par Thor et par Odin ! Réfléchissez nom d'une pipe !

# Chapitre III

## Règles spécifiques de la journée

- Durant cette journée de piscine vous allez développer un unique site. Celui-ci a pour objectif de proposer des articles à consulter. Il permet aux utilisateurs connectés de publier de nouveaux articles ou d'en sauvegarder sous forme de favoris.
- Vous avez la liberté de nommer ce site comme il vous plait et d'orienter sa thématique à votre gout (actualités, fanfictions, nouvelles érotiques, etc ...).
- Vous pouvez choisir la base de donnée que vous souhaitez, du moment qu'elle est compatible avec l'ORM natif de Django
- Votre rendu prendra la forme d'un unique projet Django. Il n'aura pas le découpage habituel en exercices. Chacun d'entre eux rajoutent au projet une fonctionnalité. C'est celle-ci, ainsi que son implémentation qui seront notés.
- Il est interdit d'implémenter la moindre url "en dur". Vous devez systématiquement vous référer au nom de l'url. Que ce soit dans les vues ou dans les templates.
- Il est interdit d'implémenter la moindre vue sous forme de fonctions. Vous devez utiliser des vues génériques.
- Souvenez vous que les exercices seront corrigés dans l'ordre.
- La langue par défaut de votre site est l'anglais. Le contenu de la base de donnée peut-être n'importe quoi. Du latin si cela vous arrange.
- Vous devez laisser l'application d'administration par défaut.




Ne perdez pas de temps sur ce qui ne vous est pas demandé !



# Chapitre IV

## Exercice 00

	Exercice : 00
Exercice 00 : Model building - Generic Class	
Dossier de rendu : ex00/	
Fichiers à rendre :	
Fonctions Autorisées :	
Remarques : n/a	

Créez un nouveau projet. Vous pouvez le nommer comme il vous plait et organiser la structure des applications comme il vous plait. Pensez qu'une structure logique et facile à comprendre facilitera les corrections.

Implémentez les modèles suivants :

- **Article** : Contenu des articles et quelques metadatas. Il doit contenir les champs suivants :
  - **title** : Titre de l'article. Chaîne de caractères d'une taille maximum de 64. Non null.
  - **author** : Auteur de l'article. Référence un enregistrement du model User. Non null.
  - **created** : Date et heure complète de création. Doit être automatiquement rempli à la création. Non null.
  - **synopsis** : Résumé de l'article. Chaîne de caractères d'une taille maximum de 312. Non null.
  - **content** : L'article lui-même. C'est un type texte. Non null.

La méthode `__str__` doit être 'override' pour renvoyer 'title'

- **UserFavouriteArticle** : Articles favoris enregistrés par un utilisateur. Doit contenir les champs suivants :
  - **user** : Référence un enregistrement du model User. Non null.
  - **article** : Référence un enregistrement du model Article. Non null.

La méthode `__str()` doit être 'override' pour renvoyer 'title' qui se trouve dans le modele Article.

Une fois ceci fait, nous pouvons passer à l'objectif réel de ce premier exercice.

En vous servant uniquement de vues génériques (à l'exception de '**View**' dont vous n'avez pas la permission d'hériter directement), vous devez implémenter les fonctionnalités suivantes dans votre site. Chacune de ces fonctionnalités doit avoir une URL qui lui est propre :

**Articles** : Page HTML affichant sous forme de table HTML tous les champs (à l'exception de **content**) de tous les articles enregistrés dans la table **Article**.

Le tableau doit disposer d'un **header** indiquant le titre de chaque colonne.

**Home** : Url imposée : '`127.0.0.1:8000`'. Redirige vers **Articles**

**Login** : Page HTML affichant un formulaire de type **POST**. Logue un utilisateur non logué grâce à un nom d'utilisateur et un mot de passe. En cas d'erreur, la page doit afficher un message décrivant cette erreur. En cas de succes, la vue doit rediriger vers '**Home**'.


Vous devez également fournir au moins cinq articles d'exemples depuis trois utilisateurs différents. Fournissez des fixtures si nécessaires. Le contenu des articles n'a aucune importance, du 'lorem ipsum' convient parfaitement.



Aucun formatage css n'est requis dans le cadre de cet exercice.

# Chapitre V

## Exercice 01

	Exercice : 01
Exercice 01 : Generic Class again	
Dossier de rendu : <i>ex01/</i>	
Fichiers à rendre :	
Fonctions Autorisées :	
Remarques : n/a	

En vous servant uniquement de vues génériques (à l'exception de '**View**' dont vous n'avez pas la permission d'hériter directement), vous devez implémenter les fonctionnalités suivantes dans votre site. Chacune de ces fonctionnalités doit avoir une URL qui lui est propre :

**Publications** : Page HTML affichant, sous forme de table HTML les champs '**title**', '**synopsis**' et '**created**', de tous les articles enregistrés dans le modèle '**Article**' dont l'auteur est l'utilisateur actuellement connecté.

Pour chaque article, vous devez également implémenter un lien dont l'url doit contenir l'identifiant du-dit article, menant à la fonctionnalité '**Detail**' de l'article en question.

Le tableau doit disposer d'un '**header**' indiquant le titre de chaque colonne.

**Detail** : Page HTML affichant tous les champs d'un article donné se trouvant en base de données. L'identifiant de cet article doit se trouver dans l'url.

La disposition des champs est libre.

Vous devez également rajouter pour chaque article présent dans la table HTML de la fonctionnalité **Articles** de l'exercice précédent, un lien vers le '**Detail**' du-dit article.

**Logout** : Lien déloguant un utilisateur logué. L'endroit où se trouve le lien pour se déloguer n'a pas d'importance dans cet exercice, du moment qu'il est visible et accessible. Une fois délogué, l'utilisateur est redirigé vers '**Home**'


**Favourites :** page HTML affichant, sous la forme d'une liste de liens, les titres des articles enregistrés en favori par l'utilisateur actuellement connecté.

Chaque lien, dont l'url doit contenir l'identifiant de l'article concerné, doit mener à la fonctionnalité '**Detail**' du-dit article.

Vous devez fournir au moins un utilisateur ayant au moins deux favoris différents.

# Chapitre VI

## Exercice 02

	Exercice : 02
Exercice 02 : Generic Class - CreateView	
Dossier de rendu : <i>ex02/</i>	
Fichiers à rendre :	
Fonctions Autorisées :	
Remarques : n/a	

En vous servant uniquement de **CreateView**, vous devez implémenter les fonctionnalités suivantes dans votre site. Chacune de ces fonctionnalités doit avoir une URL qui lui est propre :

**Register** : Page HTML contenant un formulaire de type 'POST' permettant à un utilisateur non logué de créer un nouveau compte utilisateur.

Le formulaire doit demander obligatoirement un login, un mot de passe et une confirmation de mot de passe. Ce formulaire doit être accessible à une URL qui lui est exclusivement dédiée et qui doit terminer par '**register**'.

**Publish** : Page HTML contenant un formulaire de type 'POST' permettant à un utilisateur loggé de publier un nouvel article. Le champ '**author**' ne doit pas être affiché, il doit être complété dans la vue lors du processus de validation. Vous devez impérativement utiliser un objet '**form**' créé par votre View pour générer votre formulaire (pas de balises `<input>` tapé à main pour les champs de votre formulaire!)

Rajoutez un lien vers cette fonctionnalité dans le template de la fonctionnalité Publications.

**Add to favourite** : Page HTML contenant un formulaire de type 'POST' se trouvant dans la page de détail d'un article. Aucun champs ne doit être visible. Le champ '**article**' doit être prérempli avec l'ID de l'article courant et lors de la validation, le champ '**user**' doit être rempli avec l'ID de utilisateur logué. Ce mécanisme permet d'ajouter l'article courant dans les favoris de l'utilisateur connecté.




Aucun formatage CSS n'est requis dans le cadre de cet exercice.



Vous saviez que Django propose des formulaires tout prêt à l'emploi ?

# Chapitre VII

## Exercice 03

	Exercice : 03
Exercice 03 : Template tags and Filters	
Dossier de rendu : <i>ex03/</i>	
Fichiers à rendre :	
Fonctions Autorisées :	
Remarques : n/a	

Dans cet exercice vous devez créer un menu qui doit être visible depuis TOUTES les pages du site.

Tous les liens de ce menu mènent vers des fonctionnalités que **VOUS AVEZ DEJA CRÉES**. S'il vous manque quelque chose... posez vous des questions.

Ce menu doit disposer des éléments suivants :

- **Home** : Lien vers la fonctionnalité 'Home' (qui elle-même redirige vers 'Articles', vous vous souvenez?).
- **Last Articles** : Lien vers la fonctionnalité 'Article'. Vous pouvez adapter le nom de ce lien en fonction de l'éventuelle thématique que vous avez choisi (mais toujours en anglais!).
- Si un utilisateur n'est pas connecté :
  - **Register** : Lien vers la fonctionnalité 'Register'
  - **Login** : Fonctionnalité 'Login'. Ici, c'est un peu différent car ce n'est pas un simple lien, vous devez mettre le formulaire entier DANS le menu.

Celui-ci sera donc disponible sur toutes les pages et non plus seulement sur la page dédiée que vous avez du créer.

Celle-ci doit cependant toujours afficher les erreurs en cas de formulaire inva-

lide.

- Si un utilisateur est connecté :
  - **Favourites** : Lien vers la fonctionnalité 'Favourites'
  - **Publications** : Lien vers la fonctionnalité 'Publications'
  - **Logged as <user>** : Simple texte indiquant que l'utilisateur est logué. <user> doit évidemment être remplacé par le nom de l'utilisateur connecté.
  - **Logout** : Fonctionnalité 'Logout'. Maintenant, vous avez un endroit précis où mettre ce lien.

En utilisant des tags et des filtres, modifiez le template listant tous les articles de sorte que :

- Le synopsis soit réduit à 20 caractères maximum, au delà desquels la suite doit être remplacée par trois points de suspension. Vous devez prévoir un exemple qui en fait la démonstration.
- La liste d'articles soit triée par date, de la plus récente à la plus vieille.
- Une colonne supplémentaire mentionne depuis combien de temps l'article a été publié.




Aucun formatage CSS n'est requis dans le cadre de cet exercice.



# Chapitre VIII


## Exercice 04

	Exercice : 04
Exercice 04 : Bootstrap	
Dossier de rendu : <i>ex04/</i>	
Fichiers à rendre :	
Fonctions Autorisées :	
Remarques : n/a	

En utilisant Bootstrap, donnez a votre menu le même formatage CSS que celui se trouvant dans l'image fournie dans les ressources de la journée.

# Chapitre IX

## Exercice 05

	Exercice : 05
Exercice 05 : Internationalization	
Dossier de rendu : <i>ex05/</i>	
Fichiers à rendre :	
Fonctions Autorisées :	
Remarques : n/a	

Traduisez toute la fonctionnalité **Articles** du site, ainsi que le menu (qui est visible depuis cet endroit également) en français en fonction du prefix se trouvant dans l'URL.

Par exemple :

Si mon URL est : `127.0.0.1:8000/en/articles`, alors tout son contenu sera en anglais.

Si cette URL est `127.0.0.1:8000/fr/articles`, alors tout son contenu sera en français.


Le contenu de la base de données et le nom du site ne sont pas à traduire.

Par défaut, le site doit etre en anglais.

Ajoutez un lien, permettant de passer d'une langue à l'autre sur cette même page.

# Chapitre X

## Exercice 06

	Exercice : 06
Exercice 06 : Testing	
Dossier de rendu : <i>ex06/</i>	
Fichiers à rendre :	
Fonctions Autorisées :	
Remarques : n/a	

En vous servant du framework de test intégré à Django, créez les tests permettant de vérifier que le site à bien les comportements suivants :

- Les vues **favorites**, **publications** et **publish** ainsi que leur templates ne sont accessibles qu'aux utilisateurs connectés.
- Il n'est pas possible pour un utilisateur connecté d'avoir accès au formulaire de création d'un nouvel utilisateur.
- Un utilisateur ne peut pas mettre deux fois le meme article en favoris.

Vos tests doivent impérativement porter un nom explicite, indiquant exactement ce qu'ils verifient.

Corrigez toutes les erreurs potentiellements révélées par ces tests