



# Formation Python-Django - 1

## Base Django

*Résumé: Après le Python, Django !*

*Version: 1*

# Table des matières

<b>I</b>	<b>Préambule</b>	<b>2</b>
<b>II</b>	<b>Règles communes</b>	<b>3</b>
<b>III</b>	<b>Règles spécifiques de la journée</b>	<b>4</b>
<b>IV</b>	<b>Exercice 00</b>	<b>5</b>
<b>V</b>	<b>Exercice 01</b>	<b>6</b>
<b>VI</b>	<b>Exercice 02</b>	<b>8</b>
<b>VII</b>	<b>Exercice 03</b>	<b>10</b>
<b>VIII</b>	<b>Rendu et peer-évaluation</b>	<b>12</b>

# Chapitre I

## Préambule

Voici la liste des monstres qu'il est possible de rencontrer lors de l'exploration d'un donjon en Terre de Fangh :

- toutes sortes de morts vivants.
- des araignées géantes.
- des orcs et des gobelins.
- des trolls dans les souterrains.
- des sorciers.
- des guerriers maudits.
- des rats géants.
- une bouteille d'huile.
- du papier toilette.
- deux éponges.
- des raviolis.

# Chapitre II

## Règles communes

- Votre projet doit être réalisé dans une machine virtuelle.
- Votre machine virtuelle doit avoir tout les logiciels necessaire pour réaliser votre projet. Ces logiciels doivent être configurés et installés.
- Vous êtes libre sur le choix du systmème d'exploitation à utiliser pour votre machine virtuelle.
- Vous devez pouvoir utiliser votre machine virtuelle depuis un ordinateur en cluster.
- Vous devez utiliser un dossier partagé entre votre machine virtuelle et votre machine hôte.
- Lors de vos évaluations vous allez utiliser ce dossier partager avec votre dépôt de rendu.
- Vos fonctions de doivent pas s'arrêter de manière inattendue (segmentation fault, bus error, double free, etc) mis à part dans le cas d'un comportement indéfini. Si cela arrive, votre projet sera considéré non fonctionnel et vous aurez 0 au projet.
- Nous vous recommandons de créer des programmes de test pour votre projet, bien que ce travail **ne sera pas rendu ni noté**. Cela vous donnera une chance de tester facilement votre travail ainsi que celui de vos pairs.
- Vous devez rendre votre travail sur le git qui vous est assigné. Seul le travail déposé sur git sera évalué. Si Deepthought doit corriger votre travail, cela sera fait à la fin des peer-evaluations. Si une erreur se produit pendant l'évaluation Deepthought, celle-ci s'arrête.


# Chapitre III

## Règles spécifiques de la journée

- Les routes relatives à une application doivent être définies dans un fichier `urls.py` se trouvant dans le dossier de cette application.
- Tout formulaire (classe dérivée de `django.forms.Form`) doit se trouver dans le fichier `forms.py` de l'application à laquelle il est rattaché.
- Chaque page affichée doit être correctement formatée (présence d'un doctype, de couples de balises `html`, `body`, `head`), gestion correcte des caractères spéciaux, pas d'affichage bizarre.
- Le serveur utilisé pour cette journée est le serveur de développement par défaut de Django fourni avec l'utilitaire `manage.py`.
- Seules les URLs explicitement demandées doivent retourner une page sans erreur. Ainsi, si seule `/ex00` est demandée, `/ex00foo` doit retourner une erreur 404.
- Les URLs demandées doivent fonctionner avec et sans slash de fin. Ainsi, si `/ex00` est demandée, `/ex00` et `/ex00/` doivent toutes les deux fonctionner.

# Chapitre IV

## Exercice 00

	Exercice : 00
Exercice 00 : Première page statique.	
Dossier de rendu : <code>ex00/</code>	
Fichiers à rendre : <code>requirements.txt</code> , les fichiers et dossiers de votre projet et de votre application	
Fonctions Autorisées : L'ensemble des fonctionnalités de Django.	

Vous allez dans cet exercice faire votre première page statique avec Django.

Créez un `virtualenv` avec `python3`, installez Django, créez un fichier `requirements.txt` contenant les dépendances du projet (ce qui est installé dans le `virtualenv` avec `pip`) à la racine du rendu.

Démarrez un projet `d04`.


Démarrez une application `ex00`.

Réalisez une page qui sera consultable à l'URL `/ex00` de votre site. Rassemblez dans cette page les informations sur l'ensemble de la syntaxe du `Markdown`, et donnez pour titre à cette page “`Ex00 : Markdown Cheatsheet.`”.

Le template utilisé devra avoir pour nom `index.html`.

# Chapitre V

## Exercice 01

	Exercice : 01
Exercice 01 : Quelques pages de plus.	
Dossier de rendu : <i>ex01/</i>	
Fichiers à rendre : <code>requirements.txt</code> , les fichiers et dossiers de votre projet et de votre application.	
Fonctions Autorisées : L'ensemble des fonctionnalités de Django.	

Réalisez les pages suivantes dans une seconde application **ex01** :

- **Titre** : “Ex01 : Django, framework web.”

**URL** : `/ex01/django`.

**Description** : Faites dans cette page une brève présentation de Django et de son historique.

- **Titre** : “Ex01 : Processus d’affichage d’une page statique.”

**URL** : `/ex01/affichage`

**Description** : Décrivez dans cette page le processus entraînant l’affichage d’une page web statique à partir d’un simple template en passant par une vue, de la requête à la réponse.

- **Titre** : “Ex01 : Moteur de template.”

**URL** : `/ex01/templates`

**Description** : Décrivez dans cette page le fonctionnement du moteur de templates par défaut de Django ainsi que le fonctionnement :

- Des blocs.
- Des boucles `for ... in`.
- Des structures de contrôle `if`.
- De l’affichage de variables passées par contexte.

Le principe du *don't repeat yourself* fait partie intégrante de la philosophie de Django. Pour respecter ce principe, réalisez les pages demandées en utilisant un template de base ayant pour nom `base.html`.

Le template `base.html` doit comporter :

- Un bloc `content` dans le `body`.
- Un bloc `style` dans le `head`.
- Un bloc `title` dans le `head`.

Réalisez également un template `nav.html` contenant une barre de navigation listant les liens vers chacune des pages de l'exercice.

Toutes les pages de cet exercice doivent se baser sur le template `base.html`. Le template `nav.html` doit être inclus dans toutes les pages.

Créez également deux fichiers `style1.css` et `style2.css`. Le premier devra définir la couleur de texte comme étant du *bleu*, l'autre devra la définir comme étant du *rouge*. Vous devrez utiliser `style1.css` dans toutes les pages, sauf dans la page "Moteur de template" où vous utiliserez `style2.css`.



Vous ne devez utiliser chaque feuille de style qu'une seule et unique fois dans l'ensemble de vos templates pour cet exercice.




`collectstatic` sera à utiliser en correction.



# Chapitre VI

## Exercice 02

	Exercice : 02
Exercice 02 : Premier formulaire.	
Dossier de rendu : <i>ex02/</i>	
Fichiers à rendre : <code>requirements.txt</code> , les fichiers et dossier de votre projet et de votre application.	
Fonctions Autorisées : L'ensemble des fonctionnalités de Django.	

Réalisez une page accessible par l'URL `/ex02` dans une nouvelle application `ex02`.

Cette page contiendra deux parties :

- Un formulaire avec un champ de texte et un bouton `Submit`.
- Une partie pour contenir l'historique des entrées.

Voici la règle concernant le formulaire :


- Vous ne devez PAS coder le champ du formulaire en dur, mais utiliser la classe `django.forms.Form` que fournit Django pour créer un formulaire. Vous le passerez en tant que `contexte` à la fonction de rendu du template.

Voici les règles pour l'historique :

- L'historique des entrées sera initialement vide.
- À chaque soumission de texte par le formulaire, vous devrez :
  - Inscrire l'entrée et son horodatage à la fin d'un fichier de `logs`. Si ce fichier n'existe pas il doit être automatiquement créé. Ce fichier doit être créé dans le dossier de l'application `ex02`.
  - Ajouter l'entrée à l'historique de la page, avec l'heure et la date de soumission.
- Le chemin vers le fichier de `logs` (nom du fichier inclus) doit être défini dans le fichier `settings.py` du projet. Ainsi, chaque entrée soumise sera présente dans la partie historique de la page et dans le fichier de `log`, précédée de son horodatage.
- Les données doivent également être persistantes. Si le serveur de développement doit redémarrer pour une quelconque raison, les données ne devront pas être perdues.  
En règle générale, si vous réaffichez la page, les données sont réaffichées tant qu'elles sont conservées.

# Chapitre VII

## Exercice 03

	Exercice : 03
Exercice 03 : Fifty shades of bic.	
Dossier de rendu : <i>ex03/</i>	
Fichiers à rendre : <code>requirements.txt</code> , les fichiers et dossiers de votre projet et de votre application.	
Fonctions Autorisées : L'ensemble des fonctionnalités de Django.	

Créez une dernière application `ex03`.

Affichez une page contenant un tableau de 4 colonnes et de 51 lignes (dont une ligne pour les noms de colonnes).

Cette page sera accessible à l'URL `/ex03`.

Chaque colonne du tableau correspondra à une couleur parmi *noir*, *rouge*, *bleu*, *vert*.

Les cases du tableau doivent avoir pour caractéristiques :

**Hauteur** : 40 pixels.

**Largeur** : 80 pixels.

**Couleur de fond** : une nuance de la couleur à laquelle correspond la colonne.

Le tableau doit afficher les nuances de chaque couleur de manière à obtenir un dégradé sur 50 lignes.

Vous devez respecter les consignes suivantes :

- Votre template ne doit **PAS** contenir les couleurs en dur. Les différentes nuances doivent être générées dans une vue et être toutes différentes.
- 4 couples de balises `<td></td>` sont autorisés dans vos templates au total.
- 4 couples de balises `<th></th>` sont autorisés dans vos templates au total.
- Votre tableau doit être formaté de la sorte :
  - Un couple de balises `<th></th>` par case pour les noms de colonnes.
  - Un couple de balises `<tr></tr>` par ligne de nuance de couleur.
  - Un couple de balises `<td></td>` par case de nuance de couleur.

# Chapitre VIII

## Rendu et peer-évaluation

Rendez votre travail dans votre dépôt Git comme d'habitude. Seul le travail présent dans votre dépôt sera évalué en soutenance. Vérifiez bien les noms de vos dossiers et de vos fichiers afin que ces derniers soient conformes aux demandes du sujet.



L'évaluation se déroulera sur l'ordinateur du groupe évalué.