9. Introduction au HTML

Vous vous demandez sûrement ce qu'est un template.

Un template est un fichier que vous pouvez réutiliser afin de présenter des informations différentes sous un seul et même format. Par exemple, vous pourriez avoir envie d'utiliser un template pour écrire une lettre : bien que son contenu varie ou qu'elle puisse être adressée à des personnes différentes, sa forme reste la même.

Le format d'un template Django est décrit grâce à un langage qui s'appelle HTML.

Qu'est-ce que le HTML ?

HTML est un code qui est interprété par votre navigateur (Chrome, Firefox ou Safari) et qui permet d'afficher une page web à l'utilisateur.

L'abréviation HTML signifie "HyperText Markup Language". "HyperText" signifie que c'est un type de texte qui supporte les hyperliens entre les pages. "Markup" signifie que nous avons pris un document et que nous l'avons balisé le code pour signifier (ici, au navigateur) comment il faut interpréter la page. Le code HTML est construit à l'aide de balises, chacune commençant par < et finissant par >. Ces balises représentent des "éléments" markup.

Votre premier template !

Créer un template signifie créer un fichier template. Et oui, encore des fichiers ! Vous aviez déjà probablement remarqué que tout tourne autour des fichiers.

Les templates sont sauvegardés dans le dossier blog/templates/blog. Tout d'abord, créons un dossier appelé "templates" à l'intérieur du dossier de notre blog. Ensuite, créez un autre dossier appelé "blog" à l'intérieur de votre dossier templates :

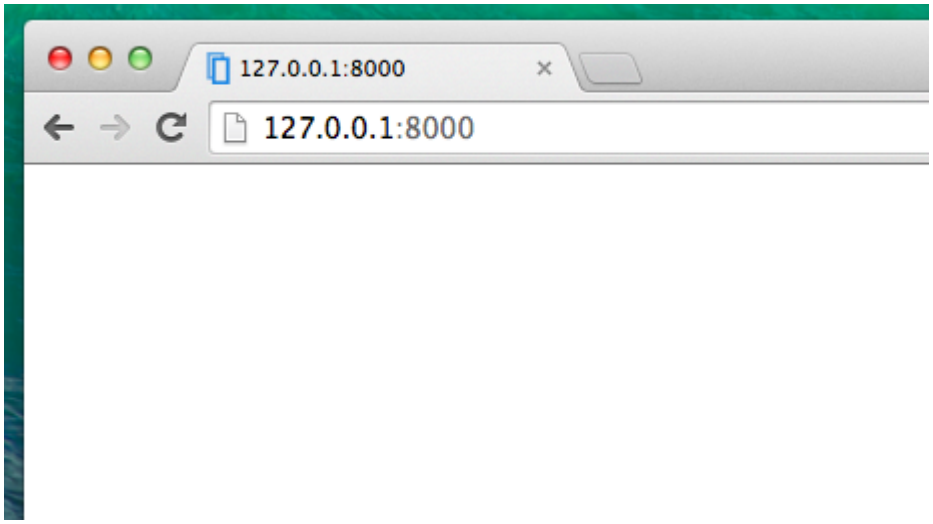
```
blog
└── templates
    └── blog
```

Vous pourriez vous demander pourquoi nous avons besoin de deux dossiers portant tous les deux le nom "blog". Comme vous le découvrirez plus tard, c'est une convention de nommage qui va nous faciliter la vie quand les choses vont commencer à devenir compliquées.

Et maintenant, créez un fichier "post_list.html" (laissez-le vide pour le moment) dans le dossier "templates/blog/blog".

Allons regarder à quoi ressemble notre site maintenant : http://127.0.0.1:8000/

Si vous avez encore l'erreur "TemplateDoesNotExist", essayez de redémarrer votre serveur. Allez sur votre ligne de commande et arrêtez votre serveur en appuyant simultanément sur Ctrl+C (les touches Control et C de votre clavier). Vous pouvez le relancer en tapant la commande "python manage.py runserver".



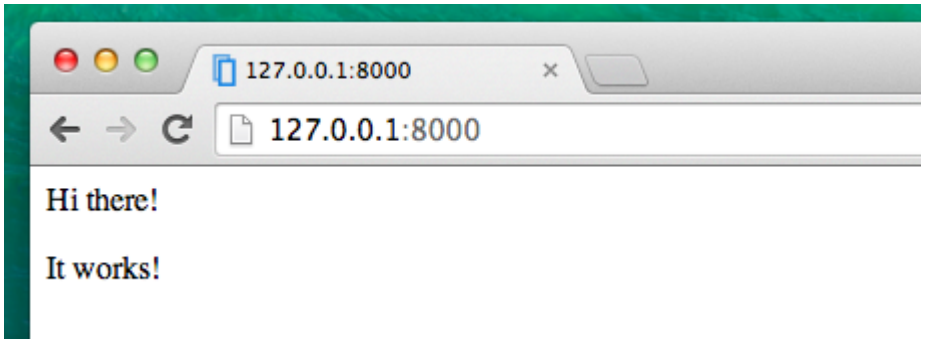
Et voilà, il n'y a plus d'erreurs ! Bravo :) Cependant, notre site ne peut rien faire d'autre pour le moment qu'afficher une page blanche. La faute à notre template que nous avons laissé vide. Allons corriger ça.

Ouvrez le nouveau fichier dans l'éditeur de code et ajoutez le morceau suivant :

`blog/templates/blog/post_list.html`

```
<html>
<body>
    <p>Hi there!</p>
    <p>It works!</p>
</body>
</html>
```

Alors, à quoi ressemble notre site web maintenant ? Allons le découvrir : http://127.0.0.1:8000/



Ça marche ! Bon boulot :)

- La balise la plus élémentaire, `, figure toujours au début de n'importe quelle page web tandis que` est toujours située à la fin. Comme vous pouvez le constater, l'intégralité du contenu de notre page web est située entre la balise de départ, `, et la balise fermante, .`
- est la balise pour les éléments de type paragraphe. permet de fermer chaque paragraphe.

Head et body

Chaque page HTML est divisée en deux éléments : head (entête) et body (corps).

- **head** est un élément qui contient des informations sur le document : son contenu ne s'affichera pas à l'écran.
- **body** est un élément qui contient tout le reste. Son contenu s'affichera à l'écran et constituera notre page web.

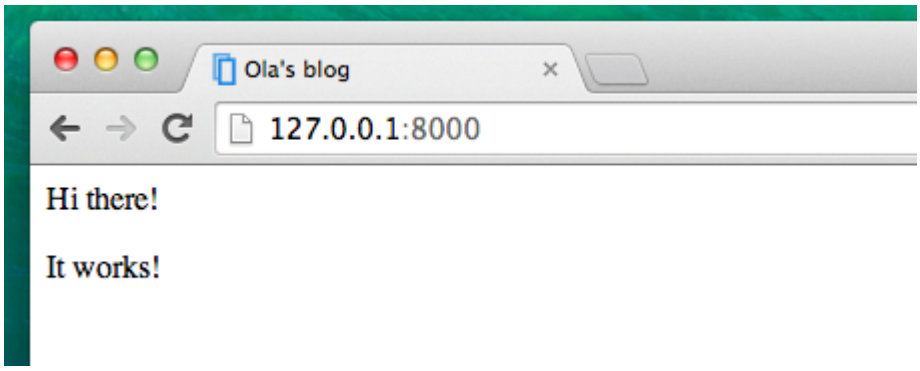
Nous utilisons <head> pour transmettre la configuration de la page au navigateur tandis que <body> l'informe sur le contenu de la page.

Par exemple, vous pouvez donner un titre à votre page web en utilisant l'élément titre dans le <head> :

`blog/templates/blog/post_list.html`

```
<html>
    <head>
        <title>Django Girls blog</title>
    </head>
    <body>
        <p>Hi there!</p>
        <p>It works!</p>
    </body>
</html>
```

Sauvegardez votre fichier et actualisez la page.



Vous avez vu comment le navigateur a compris que "Le Blog d'Ola" est le titre de votre page ? Il a interprété <title>Le blog d'Ola</title> et a placé ce texte dans la barre de titre de votre navigateur (c'est ce titre qui va être aussi utilisé lorsque vous créez un marque-page, etc.).

Vous avez aussi probablement remarqué que chaque balise ouvrante possède sa balise fermante, composée d'un /, est qu'elles encadrent les différents éléments. Cela signifie que vous ne pouvez pas fermer une balise si celles imbriquées à l'intérieur de celle-ci n'ont pas été fermées.

Pensez à lorsque vous mettez des choses à l'intérieur de boîtes. Vous avez une grosse boîte, ; à l'intérieur de celle-ci, on trouve une plus petite boîte, <body></body>, qui contient elle-même d'autres petites boîtes, .

Essayez de vous rappeler de cet exemple lorsque vous utilisez les balises fermantes et que vous avez des éléments imbriqués. Si vous ne suivez pas ces règles, votre navigateur risque de ne pas être capable d'interpréter votre code correctement et votre page web sera mal affichée.

Personnaliser votre template

Et si nous en profitions pour nous amuser un peu ? Essayons de personnaliser notre template ! Voici quelques balises que vous pouvez utiliser :

- <h1>Titre 1</h1> pour vos titres les plus importants
- <h2>Titre 2</h2> pour les sous-titres
- <h3>Titre 3</h3> ... et ainsi de suite jusqu'à <h6>
- Un paragraphe contenant du texte
- texte permet de mettre l'accent sur une partie du texte
- texte permet de mettre encore plus l'accent sur une partie de texte
-
 permet d'insérer un saut de ligne (vous ne pouvez rien mettre à l'intérieur d'un élément br et il n'y a pas de balise fermante)
- link permet de créer un lien
- premier item
second item permet de créer des listes, comme celle que nous sommes en train de faire !

- permet de créer une section au sein de la page

Voici un exemple d'un modèle complet, copiez et collez-le dans blog/templates/blog/post_list.html :

`blog/templates/blog/post_list.html`

```
<html>
    <head>
        <title>Django Girls blog</title>
    </head>
    <body>
        <div>
            <h1><a href="/">Django Girls Blog</a></h1>
        </div>

        <div>
            <p>published: 14.06.2014, 12:14</p>
            <h2><a href="">My first post</a></h2>
            <p>Aenean eu leo quam. Pellentesque ornare sem lacinia quam
venenatis vestibulum. Donec id elit non mi porta gravida at eget metus.
Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut
fermentum massa justo sit amet risus.</p>
        </div>

        <div>
            <p>published: 14.06.2014, 12:14</p>
            <h2><a href="">Mon second post</a></h2>
            <p>Aenean eu leo quam. Pellentesque ornare sem lacinia quam
```

```
venenatis vestibulum. Donec id elit non mi porta gravida at eget.</p>
        </div>
    </body>
</html>
```

Nous avons créé trois sections à l'aide de .

Le premier contient le titre de notre blogpost - c'est à la fois un titre et un lien

Les deux autres contiennent nos posts avec leur date de publication, un titre de post <h2> qui est cliquable ainsi que deux <p>s (paragraphe) de texte : un pour la date et l'autre pour notre post.

Ce qui nous donne :

Pour l'instant, notre template nous permet seulement d'afficher les mêmes informations alors que nous disions précédemment qu'il doit nous permettre d'afficher des informations différentes utilisant le même format.

Ce qu'on aimerait pouvoir faire maintenant, c'est d'afficher les posts que nous avons créés précédemment dans l'interface d'administration de Django. Penchons-nous là-dessus.

10. L'ORM de Django