TEMA IV

DESARROLLO DE APLICACIONES WEB EN BACKEND

"*Siempre parece imposible,*
hasta que se hace"
Nelson Mandela

FUNDAMENTOS DE LA PROGRAMACIÓN WEB

7.4 MÁS SOBRE LAS TEMPLATES

Con frecuencia, dentro de una aplicación front, hay partes de nuestro html que sean los mismos en varias páginas. Obviamente estos elementos comunes no los vamos a repetir en cada template, lo que haremos será crear ese html común a parte e incluirlo, ahora veremos como, en los templates necesarios.

Plantillas Incrustadas

Para incluir un template dentro de otro utlizamos la etiqueta

{% include “nombre.html” %}

Colocamos en la plantilla contenedora esa instrucción en el lugar en el que queremos que aparezca la otra plantilla

En una página “profesional” habrá muchas plantillas que a su vez incluyan otras, es muy importante organizarse bien en carpetas para encontrar lo que buscamos.

Tendremos que incluir en le etiqueta include además del nombre de la plantilla la ruta relativa al directorio templates.

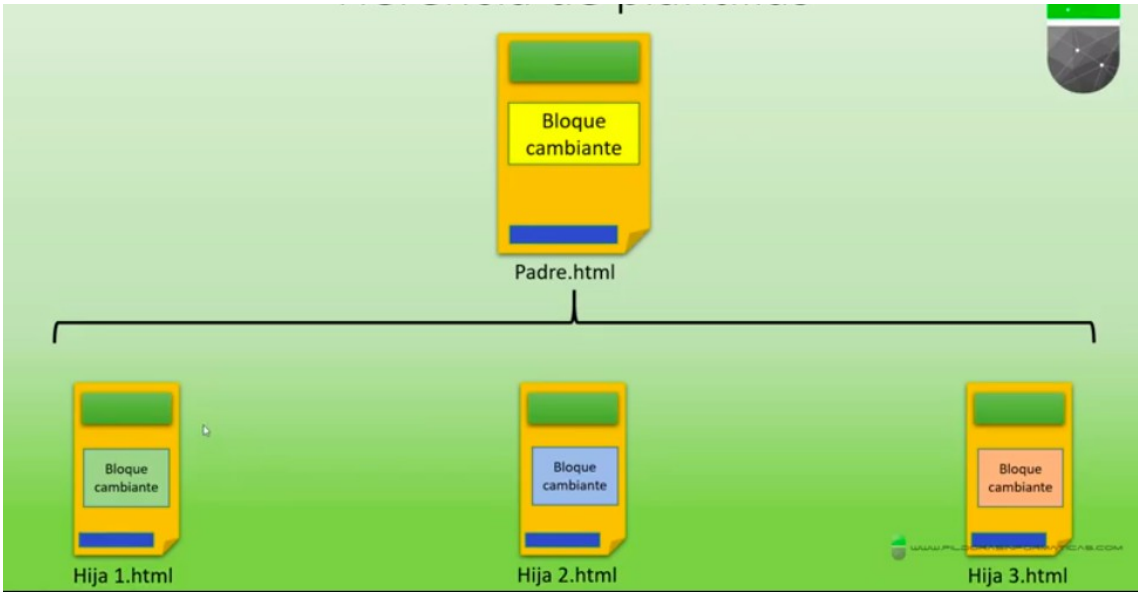
En una plantilla podemos incluir todas las que necesitemos

Hay otra forma “más elegante” de hacer esto y es utilizar la llamada herencia de plantillas.

Herencia de Plantillas

Es una forma más óptima de reutilizar plantillas. Se trata de crear una plantilla padre con el formato que tendrán las plantillas de nuestro sitio web (todas o las que necesitemos que sean comunes). Es decir tenemos una plantilla base “esqueleto” que contienen todos los bloques comunes a las páginas de mi sitio y además los bloques que van a cambiar y que serán los que las plantillas hija sobreescriban.

Por ejemplo, en la imagen de arriba supongamos que tanto el pie como la cabecera de todas mis páginas son iguales(algo bastante frecuente) pero el contenido cambia. Definimos la plantilla base con esos dos bloques comunes el bloque contenido será sobreescrito por las plantillas hijas.



Para definir esta herencia tenemos la etiqueta {% extends %}

{% extends padre.html %}

Cada plantilla que lleva esa etiqueta es “hija” de padre.html

Obviamente cualquier cambio que queramos hacer en los bloques que son comunes se harían sólo sobre la plantilla padre.

Realmente lo que vamos a definir son los bloques “diferentes”.

Para definir los bloques de contenido cambiante tenemos la etiqueta {% block nombre %}

Todas las etiquetas de este tipo que yo ponga en la plantilla base, le indican que quizás una de sus plantillas hija modifique esa parte de la plantilla

En resumen:

1. Diseño la plantilla base, incluyendo {% block nombre_del_bloque %} {%endblock%} en aquellas partes en las que quiero poner el contenido de las plantillas hijas (lo que cambia)

2. En las plantillas hijas:

    - La primera línea tiene que ser la etiqueta {% extends plantilla_base%}

- Defino todos los bloques que quiero incluir en la plantilla base, obviamente debe coincidir el nombre

No se pueden poner varias etiquetas block con el mismo nombre en una plantilla, si necesito utilizar varias veces un bloque puedo anidarlos.

Si hay contenido duplicado en varias plantillas, probablemente debas usar un bloque

No todas las plantillas hijas tienen que definir todos los bloques en la plantilla base. Si un bloque no está definido en una hija se visualiza el de la plantilla base.

Ejemplo: Veamos un sencillo ejemplo de dos plantillas para mis dos asignaturas

La primera página mostrará:

- Una barra de navegación con dos enlaces, común a todas las páginas
- Un pie también común
- Un contenido que cambiará en función de la opción elegida
- El título de la página también cambiará

Lo que recomiendan en la página oficial:

Puede utilizar tantos niveles de herencia como necesite. Una forma común de utilizar la herencia es el siguiente enfoque de tres niveles:

- Cree una **base.html** plantilla que mantenga la apariencia principal de su sitio.
- Cree una **base_SECTIONNAME.html** plantilla para cada "sección" de su sitio. Por ejemplo, **base_news.html**, **base_sports.html**. Todas estas plantillas amplían **base.html** e incluyen estilos / diseños específicos de la sección.
- Cree plantillas individuales para cada tipo de página, como un artículo de noticias o una entrada de blog. Estas plantillas amplían la plantilla de sección adecuada.

Hay otra etiqueta de plantilla que se utiliza con mucha frecuencia {% url %}

Es muy útil para generar dinámicamente los enlaces. Es decir, si nuestro enlace es estático href=’about’, si lo queremos cambiar en el archivo urls.py tendríamos que refactorizar toda la aplicación. Si lo generamos de forma dinámica, lo cambiamos sólo en el archivo y django genera la url nueva. Para ello añadimos un atributo a la función path y escribimos {% url nombre que aparece en el archivo %}

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path('',home),
    path('servidor/',servidor,name='serv'),
    path('moviles/',moviles,name='mov'),
]
```

En la template

```
<li class="nav-item">
    <a class="nav-link" href="{% url 'serv' %}">D. Servidor</a>
</li>
<li class="nav-item">
    <a class="nav-link" href="{% url 'mov' %}">D. App. Moviles</a>
</li>
```