



**¡Les damos la
bienvenida!**

¿Comenzamos?

Esta clase va a ser

- grabada
a

Semana 11. PYTHON

Playground Intermedio – Parte II

Objetivos de la clase

- Heredar templates.
- Procesar contextos.
- Crear formularios para insertar datos.
- Crear formularios de búsqueda en la BD.

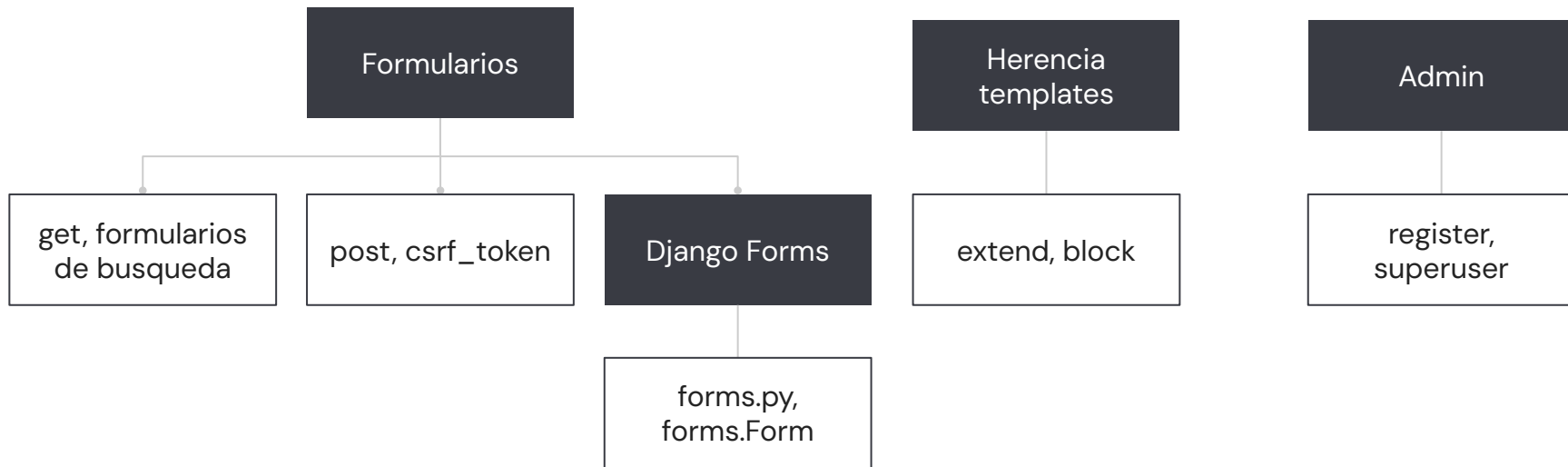


¡Recuerda esto!

Antes de iniciar esta clase,
debes abrir VSC.



MAPA DE CONCEPTOS



Repasemos...



REPASO

Semana 11

En los videos de esta semana aprendiste sobre:

- ✓ Herencia de templates, extends, block
- ✓ Apartado de admin, register, superuser
- ✓ Django Forms
- ✓ ventajas, validación

- ✓ Formularios html
- ✓ get, post, csrf_token
- ✓ Acceso a la info desde la vista
- ✓ Formularios de búsqueda



VIDEO N°11.1 – HERENCIA DE TEMPLATES

Recuperamos el tema visto

Al igual que con las clases **podemos utilizar la herencia en los templates**, permitiéndonos ahorrarnos el uso de código repetido.

Para esto, en el template que queremos usar como padre, debemos generar el bloque que queremos que sea accesible para los demás templates. Y, por medio del `extends`, hacer uso del template padre.

Recuerda que el `load static` es necesario utilizarlo en el template que utilice alguno de los archivos que se encuentren en `static` solamente.





VIDEO N° 11.2 – PANEL DE ADMINISTRACIÓN

Recuperamos el tema visto

Para **que nuestros modelos aparezcan en el apartado de admin** debemos registrarlos (`admin.site.register`) en el archivo `admin.py` de la app.

Para acceder a este apartado debemos generar un super usuario la primera vez, después pueden con este crearse otros usuarios para acceder a este apartado también.

En este apartado aparecen listados de los modelos por app de nuestro proyecto. Accediendo a alguno de estos modelos tenemos la posibilidad de ver, crear, modificar, eliminar los objetos del tipo de modelo seleccionado.





VIDEO N°11.3 – FORMULARIOS

Recuperamos el tema visto

Los **formularios** pueden ser creados desde HTML y pueden utilizar los métodos get y post. Aparte de esto, debemos agregar los inputs que necesitemos para el mismo y un botón para enviar la info.

En caso de enviarlo por post es requerido agregarle al form el csrf_token.

Para extraer la info enviada a la vista se accede al request y luego a GET o POST, los cuales se trabajan como diccionarios donde los names de los inputs son sus claves.

El **framework** nos brinda una forma más fácil y segura con sus Django forms, ya que contienen validaciones por defecto, detección de errores, entre otras.



CODERHOUSE

Contenido pregrabado

Si en algún momento quieres buscar un fragmento de contenido pregrabado y no recuerdas dónde hacerlo, puedes consultar [este resumen](#). Luego, podrás ir directo a buscar el video o podcast que necesites.

¿Preguntas?

Te invitamos a dejar tu
pregunta a través de/del
[chat](#)



Puesta en común microdesafío

¡Vamos a recuperar lo trabajado durante la semana!

Duración: **10 minutos.**



PUESTA EN COMÚN – MICRODESAFÍO

Heredemos

Termina de configurar y heredar el template de entregables.html.

El contenido del HTML, de cada archivo queda a tu criterio ¡Sé creativo!

¡Buen trabajo! 🧐



Ejemplo en vivo

Terminar de configurar y heredar el resto de los templates de nuestro proyecto AppCoder:

- `estudiantes.html`
- `profesores.html`.

El contenido del HTML, de cada archivo queda a criterio del alumno, seamos creativos



Duración: **5 minutos**







Ejemplo en vivo

Ahora veremos cómo agregar instancias de alguna clase desde el admin.

Duración: **10 minutos**

Paso a paso

1


APPCODER		
Cursos	+ Add	 Change
Entregables	+ Add	 Change
Estudiantes	+ Add	 Change
Profesors	+ Add	 Change

Paso a paso

2

Add curso

Nombre:

Camada: 

Paso a paso

3

Select curso to change

Action: 0 of 6 selected

- ☐ CURSO
- ☐ Curso object (6)
- ☐ Curso object (5)
- ☐ Curso object (4)
- ☐ Curso object (3)
- ☐ Curso object (2)
- ☐ Curso object (1)

6 cursors

Paso a paso

4

Curso object (6)

Nombre:	<input type="text" value="Programación Python"/>
Camada:	<input type="text" value="12345"/>

Podemos ver todos los cursos creados, el de arriba es el último en crearse, vemos también que en éste caso, habían otros 6 creados (eso *cambiará*).



Agregar datos a nuestro model

Duración: 10 min



ACTIVIDAD

Agregar datos a nuestro model

Descripción de la actividad.

Crear un superusuario para poder ingresar al panel de administración.
Una vez ingresados, agregar datos a dos modelos distintos (botón add).



Puesta en común

Duración: 5 min



Break

¡En 10 minutos volvemos!



Ejemplo en vivo

Ya vimos como se usan los formularios trabajando con los cursos.

Ahora implementemos lo mismo, pero para algo que tenga más datos. Con la práctica se hace el maestro 😊

Duración: **10 minutos**



Ejemplo en vivo

```
def profesorFormulario(request):  
    if request.method == 'POST':  
        miFormulario = ProfesorFormulario(request.POST) #aquí mellega toda la información del html  
        print(miFormulario)  
  
        if miFormulario.is_valid: #Si pasó la validación de Django  
            informacion = miFormulario.cleaned_data  
  
            profesor = Profesor (nombre=informacion['nombre'], apellido=informacion['apellido'],  
                                email=informacion['email'], profesion=informacion['profesion'])  
  
            profesor.save()  
  
            return render(request, "AppCoder/inicio.html") #Vuelvo al inicio o a donde quieran  
        else:  
            miFormulario= ProfesorFormulario() #Formulario vacio para construir el html  
  
    return render(request, "AppCoder/profesorFormulario.html", {"miFormulario":miFormulario})
```

Veamos la carga de nuevos datos

```
class ProfesorFormulario(forms.Form):  
    nombre= forms.CharField(max_length=30)  
    apellido= forms.CharField(max_length=30)  
    email= forms.EmailField()  
    profesion= forms.CharField(max_length=30)
```



Ejemplo en vivo

```
<h1>Formulario - Agregar Profesor</h1>

{% if miFormulario.errors %}

<p style="color: red;"> Están mal los datos, revisar</p>

{% endif %}

<form action="" method="POST">{% csrf_token %}

  <!-- Acá está la magia de Django-->

  <table>

    {{ miFormulario.as_table }}

  </table>

  <input type="submit", value="Enviar">

</form>
```

Estamos en la mitad del camino

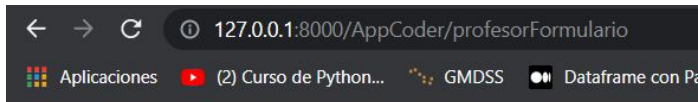
```
urlpatterns = [

    path('', views.inicio, name="Inicio"), #esta era nuestra primer view
    path('cursos', views.cursos, name="Cursos"),
    path('profesores', views.profesores, name="Profesores"),
    path('estudiantes', views.estudiantes, name="Estudiantes"),
    path('entregables', views.entregables, name="Entregables"),
    path('cursoFormulario', views.cursoFormulario, name="CursoFormulario"),
    path('profesorFormulario', views.profesorFormulario, name="ProfesorFormulario"),

]
```



Ejemplo en vivo



Formulario - Agregar Profesor

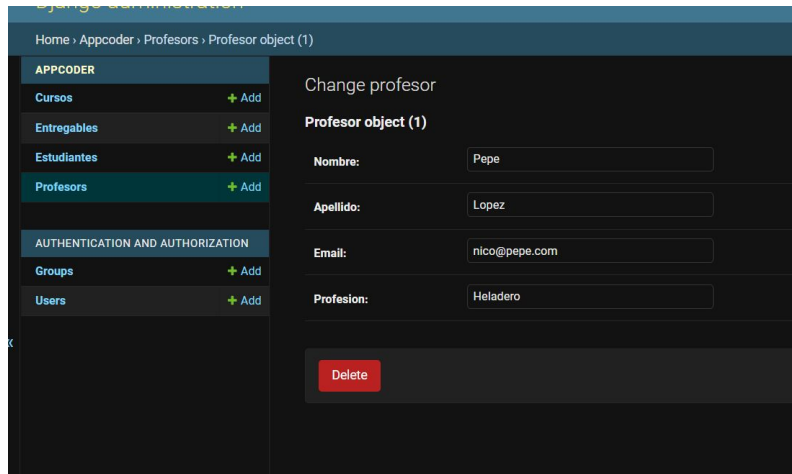
Nombre:

Apellido:

Email:

Profesion:

¡Increíble!
¿Sabrían que hemos conseguido?





Agregar con un API form

Agregar datos a una tabla de BD por medio de un Api Form.

Duración: **10 min**



ACTIVIDAD

Agregar con un API form

Descripción de la actividad.

Crear un Api Form que permita insertar datos a la base de datos de tu proyecto.
Una vez terminado, crear 2 instancias/objetos de ese modelo y verificar dichos datos en la BD.



Puesta en común

Duración: 5 min



Completemos

Duración: **15 min**



ACTIVIDAD

Completemos

Descripción de la actividad.

Usando los formularios brindados por Django, agrega los templates y/o vistas de los modelos que resten (cursos, entregables, estudiantes, profesores).

Ten en cuenta no solo los formularios de creación sino también los de búsqueda.



Puesta en común

Duración: 5 min



Para pensar

Teniendo en cuenta la forma en que hacemos las búsquedas hasta el momento.

¿Cómo harías para usar el mismo HTML para buscar y para ver los resultados de la búsqueda? ¿Crees que es posible?

Contesta mediante el chat de Zoom

Respuesta

¡Claro que sí!

Solo resta hacer que la misma plantilla resuelva los POST afirmativos, nulos o incorrectos, así se podría modificar.

```
{% if cursos %}

<p>Estamos buscando el: {{camada}}</p>

<ul>
{% for curso in cursos %}
    <li> {{curso.nombre}} </li>
    <li> {{curso.camada}} </li>
{% endfor %}
</ul>

{% else %}
<p>No hay datos con esa descripción.</p>

{% endif %}

<p style="color: red;">{{respuesta}}</p>
```

```
def buscar(request):

    if request.GET["camada"]:

        #respuesta = f"Estoy buscando la camada nro: {request.GET['camada']}"
        camada = request.GET['camada']
        cursos = Curso.objects.filter(camada__icontains=camada)

        return render(request, "AppCoder/inicio.html", {"cursos":cursos, "camada":camada})

    else:

        respuesta = "No enviaste datos"

        #No olvidar from django.http import HttpResponseRedirect
        #return HttpResponseRedirect(respuesta)
        return render(request, "AppCoder/inicio.html", {"respuesta":respuesta})
```

¿Preguntas?

Te invitamos a dejar tu
pregunta a través de/del
[chat](#)



Pre-entrega 3

Tu primera página



Tu primera página

Crea una web en Django utilizando Herencia de plantillas, con un modelo de por lo menos 3 clases, un formulario para ingresar datos a las 3 clases y un formulario para buscar algo en la BD, no hace falta que sea sobre las tres clases, con realizar búsqueda sobre una alcanzará. Te sugerimos que hagas una web estilo blog para ir preparando en la entrega final.

Debes entrar al link para acceder a la [consigna completa](#).

Objetivos

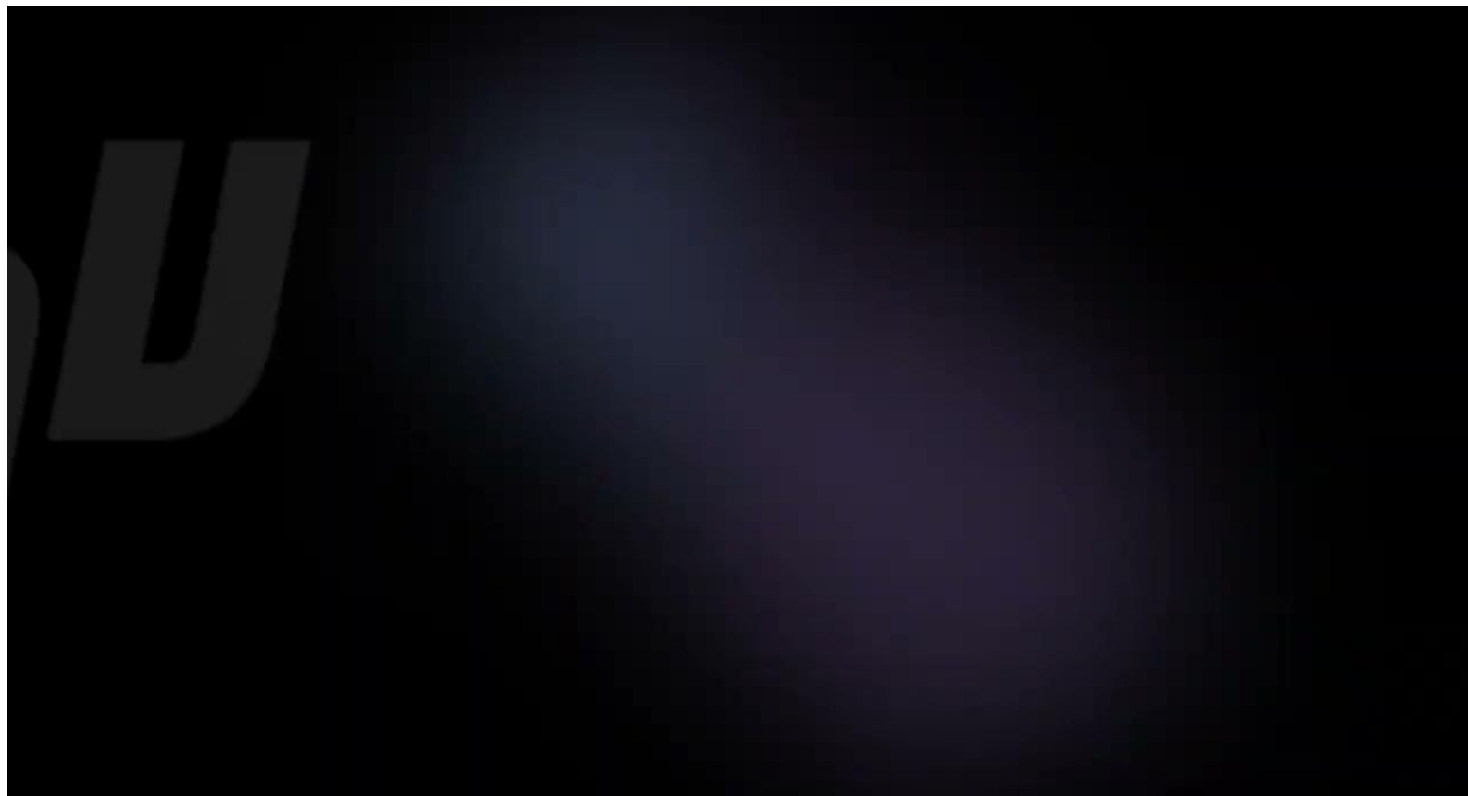
- ✓ Desarrollar tu primer WEB en Django utilizando patrón MVT

Formato

- ✓ **Link al repositorio de GitHub** con el nombre **"TuPrimeraPagina+Apellido"** por ejemplo **"TuPrimeraPagina+Fernandez"**



PRE-ENTREGA N° 3



CODERHOUSE

¿Preguntas?

Te invitamos a dejar tu pregunta
a través del [chat](#)

Resumen de la clase hoy

- ✓ Heredar contextos entre HTML
- ✓ Navegar entre distintos .html
- ✓ Ingresar al panel de admin
- ✓ Cargar datos desde el panel
- ✓ Formularios de alta de BD.
- ✓ Formularios de búsqueda.
- ✓ Formularios en HTML heredados.

La próxima semana

Los próximos temas que vamos a ver



Contenido Pregrabado

- ✓ Video 12.1 - CRUD
- ✓ Video 12.2 - Clases basadas en vistas
- ✓ Video 12.3 - Login - registro -logout
- ✓ Microdesafío - Listame



Clase en vivo (2h)

- ✓ Actividad individual: Mejor con CBV
- ✓ Mixin y decoradores
- ✓ Actividad individual: Realizar CRUD
- ✓ Actividad individual: Agregar login

La **próxima** semana

Recuerda que, a partir de ahora, tienes disponible el contenido pregrabado en la plataforma y que **es necesario que lo veas en forma previa a la próxima clase.**

Opina y valora
esta clase

Muchas gracias.

#DemocratizandoLaEducación

**¡Gracias por estudiar
con nosotros! ✨**