Modelos

Los modelos son una forma sencilla pero potente para interactuar con las base de datos en Django al usar Python.

¿Cómo definir un modelo?

En la carpeta de tu aplicación en el archivo **models.py** se definen los modelos con los que vas a trabajar.

1. Debes importar la librería que te permite trabajar con los objetos de *models*:

```
from django.db import models
```

2. Defines tu modelo. Cada clase de tu modelo representa una tabla en tu base de datos y los atributos de tu clase son los campos que tiene tu tabla. Las clases de los modelos son objetos de *models.Model.* Los atributos de tus clases tienen un tipo de campo, Django tiene múltiples tipos, por ejemplo:

* *CharField*
* *IntegerField*
* *DateField*
* *DateTimeField*
* *EmailField*
* *URLField*

Aquí puedes consultar un poco más acerca de los tipos de campos que encuentras en Django.

Las tablas en bases de datos están relacionadas entre ellas, en Django en tus modelos también se definen las relaciones entre ellos, ya sea:

* *OnetoOne()*
* *ManytoMany()*

O por medio de *ForeingKey()* para establecer la relación entre las tablas.

Ejemplo:

```
from django.db import models
class Categoria(models.Model):
    titulo = models.CharField(max_length = 140)
```

Crear los modelos

1. En **setttings.py** en **DATABASES** configuras el motor de bases de datos con el que vas trabajar:

En **ENGINE** se especifica el motor de bases de datos

```
    'ENGINE': 'django.db.backends.sqlite3'
```

En **Name** pones el nombre que va a tener la base de datos

```
    'NAME': 'db'
```

2. Construir la base de datos: desde la consola a través de **manage.py** creas la base de datos con sus tablas que están referenciadas en tu modelo.

```
$ python manage.py syncdb
```

Con **syncdb** haces una sincronización entre los modelos y la base de datos. Al ejecutarlo te preguntará el usuario, el correo y la contraseña que vas a usar para administrar la base de datos y listo, tu base de datos está creada.

Si luego ejecutas la siguiente línea puedes ver en lenguaje SQL todo lo que contiene tu base de datos.

```
$ python manage.py sqlall app
```

Ejemplo:

A partir de lo definido en tu modelo Django pasas de

```
class Categoria(models.Model):
    titulo = models.CharField(max_length = 140)
```

a

```
CREATE TABLE Categoria(
     "id" serial NOT NULL PRIMARY KEY,
     "titulo" varchar(40) NOT NULL,
);
```

Nota: por *default* Django crea un *id* como *primary key* de cada tabla.

Usar los modelos en las vistas

Cuando ya tienes creados tus modelos es muy fácil usarlos en las vistas: importas a **views.py** todos tus modelos.

```
    from models import *
```

¡Listo! ya puedes comenzar a usar tus modelos en las vistas.

Ejemplo:

```
def home(request):
    categorias = Categoria.objects.all()
    enlaces = Enlace.objects.order_by("-votos").all()
    template = "index.html"
    return render_to_response(template,locals())
```

ORM → Consultas a las bases de datos

Django tiene una forma mucho más fácil para hacer las consultas que se hacen en SQL.

Ejemplo:

* Un **SELECT:**

```
    name_class.objects.all() → SELECT * from table_name
```

* Un **INSERT**:
Django:

```
name_class = Name_Class() → Instanciar objeto
name_class.value1 = "Valor"
name_class.save()
```

SQL:

```
INSERT INTO table_name VALUES (value1,value2,value3,...);
```

* Un **SELECT** con una condición.
Django:

```
name_class.objects.get(pk = id)
```

SQL:

```
 SELECT * from table_name WHERE id = pk
```

*Un **ORDER BY**
Django:

```
name_class = Name_Class.objects.order_by("-|+campo").all()
```

SQL:

```
SELECT column_name FROM table_name ORDER BY column_name ASC|DESC
```

Activa el administrador de Django en tres líneas

La gestión de dichos modelos se puede simplificar desde el Panel de Control integrado en la misma aplicación una vez modelada.

Django te permite habilitar un panel de administración en tan solo tres sencillos pasos:

1. En **urls.py** descomentas la url para habilitar el panel de administración:

```
url(r'^admin/', include(admin.site.urls)),
```

2. Instalas el administrador en **settings.py**, es decir en **INSTALLED_APPS;** descomentas la línea para instalar el administrador:

```
'django.contrib.admin',
```

3. Registras los modelos en **admin.py**; en la carpeta de tu aplicación creas un archivo **admin.py** al que importas la librería de administración y todos tus modelos; después registras los modelos a tu administrador:

```
from django.contrib import admin
from models import *
admin.site.register(_Modelo1_)
admin.site.register(_Modelo2_)
```

Después escribes en la consola:

```
$ python manage.py syncdb
$ python manage.py runserver
```

Vas a la url del administrador:

http://127.0.0.1:8000/admin/

¡Y ahora tienes un panel de administración en menos de cinco líneas de código!

Nota: el usuario y contraseña del *admin* por defecto es el que creas con la base de datos.