



# Django: El framework web pels exigents amb deadlines

Capacitació Tecnològica per a Professionals i Empreses

The Django logo, which consists of the word "django" in a white, lowercase, sans-serif font, centered within a dark green rectangular box.

---



## Temes per la sessió 2

- 1 – South
- 2 – Admin
- 3 – Formularis
- 4 – Tags i filtres
- 5 – Projecte 2: Un diari



## 1 – South

- 1.1 – Per què South?
- 1.2 – Què son les migracions?
- 1.3 – Integració amb Django
- 1.4 – Exemples



## 1.1 – Per què South?

- South és un sistema de migracions
- Exten el model de Django i permet iterar sobre la BBDD amb canvis sense tocar SQL
- És un paquet third-party
- Té una gran comunitat darrere millorant-lo i donant-li suport
- És OpenSource
- S'integra molt fàcilment amb Django



## 1.2 – Què son les migracions?

- Evolució de l'esquema o “mutacions”
- Canvi de versió de la BBDD
- Exemple:





## 1.2 – Què son les migracions?

- South ens proporciona les migracions en dos simples processos:
  - Generació de l'esquema de migració: en aquest moment, South mirarà els canvis, i ens generarà un fitxer incremental amb la migració pertinent.
  - Aplicació de les migracions: en aquest segon punt, apliquem les migracions anteriors per a que siguin vigents a la BBDD



## 1.3 – Integració amb Django

- Afegir app a settings.py  
“south”
- És necessari que precedeixi totes les aplicacions en les que es vulgui aplicar les migracions
- Ús principal de dues tasques:  
*schemamigration*  
*migrate*



## 1.4 – Examples

```
python manage.py schemamigration app -init  
python manage.py schemamigration app -auto  
python manage.py migrate app1  
python manage.py migrate  
python manage.py migrate 0001
```





## 2 – Admin

- 2.1 – Descobrint l'Admin de Django.
- 2.2 – Utilitats
- 2.3 – Configuració bàsica
- 2.4 – Configuració avançada
- 2.5 – Personalització



## 2.1 – Descobrint l'Admin de Django

- És una aplicació opcional, que ens dona accés a una interfície d'administració generada semiautomàticament
- Configuració bàsica aplicada rapidament
- Molt potent com a prototipatge
- Molt senzilla de personalitzar
- Permet tenir un site a punt en qüestió de minuts



## 2.1 – Descobrint l'Admin de Django

- Per iniciar la interacció, només cal:
  - Afegir l'app `django.contrib.admin` a `settings.py`
  - Afegir la URI dins `urls.py`
  - Afegir un fitxer `admin.py` dins cada app amb la definició dels models accessibles

\* Els dos primers punts ja estan fets en qualsevol projecte Django. Només cal descomentar les línies indicades en els fitxers `settings.py` i `urls.py`



## 2.1 – Descobrint l'Admin de Django

← localhost:8010/admin/ ☆ Google 🔍 🏠 ⚙️ 🔄

**Administració de Django** Benvingut/da, **Isaac**. [Canviar contrasenya](#) / [Finalitzar sessió](#)

### Administració del lloc

Auth	
<b>Grups</b>	<a href="#">+ Afegir</a> <a href="#">✎ Modificar</a>
<b>Users</b>	<a href="#">+ Afegir</a> <a href="#">✎ Modificar</a>

Blog	
<b>Posts</b>	<a href="#">+ Afegir</a> <a href="#">✎ Modificar</a>

Sites	
<b>Llocs</b>	<a href="#">+ Afegir</a> <a href="#">✎ Modificar</a>

**Accions recents**  
**Les meves accions**  
[+ t2](#)  
Post



## 2.2 – Utilitats

- L'admin bàsic es compona de tres tipus de pàgina:
  - Dashboard
  - Llistat
  - Formulari



## 2.2 – Utilitats

# Dashboard



The screenshot shows the Django Admin interface in a web browser. The address bar displays 'localhost:8010/admin/'. The page title is 'Administració de Django'. The user is logged in as 'isaac', with links to 'Canviar contrasenya' and 'Finalitzar sessió'. The main content area is titled 'Administració del lloc' and contains three sections: 'Auth', 'Blog', and 'Sites'. Each section has a table with 'Grups' and 'Users' (or 'Posts' and 'Llocs') and buttons for 'Afegir' (Add) and 'Modificar' (Modify). A sidebar on the right shows 'Accions recents' (Recent actions) with a single entry: 't2 Post'.

Auth	
Grups	<a href="#">+ Afegir</a> <a href="#">Modificar</a>
Users	<a href="#">+ Afegir</a> <a href="#">Modificar</a>

Blog	
Posts	<a href="#">+ Afegir</a> <a href="#">Modificar</a>

Sites	
Llocs	<a href="#">+ Afegir</a> <a href="#">Modificar</a>

**Accions recents**  
**Les meves accions**  
[+ t2](#)  
Post



## 2.2 – Utilitats

# Llistat

← localhost:8010/admin/blog/post/ ☆ Google 🔍 🏠 ⚙️ 📧

**Administració de Django** Benvingut/da, **isaac**. [Canviar contrasenya](#) / [Finalitzar sessió](#)

[Inici](#) > [Blog](#) > [Posts](#)

### Selecioneu post per modificar

Acció:   0 de 2 seleccionats

<input type="checkbox"/>	Title	Created at	Published at	Visits
<input type="checkbox"/>	t2	22 de març de 2013 a les 9:42	22 de març de 2013 a les 9:42	5
<input type="checkbox"/>	t1	8 de març de 2013 a les 10:09	8 de març de 2013 a les 10:09	10

2 posts

**Filtre**  
**Per published at**  
[Qualsevol data](#)  
[Avui](#)  
[Últims 7 dies](#)  
[Aquest mes](#)  
[Aquest any](#)

+



## 2.2 – Utilitats

# Formulari

localhost:8010/admin/blog/post/3/ Google

Administració de Django Benvingut/da, Isaac. Canviar contrasenya / Finalitzar sessió

Inici > Blog > Posts > Post example 1

✓ El post "Post example 1" fou canviat satisfactòriament. Pot editar-lo un altra vegada a continuació.

### Modificar post

Històric Veure al lloc

**Title:** Post example 1

**Short body:** Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras convallis porta augue, ut ullamcorper quam dictum eu. Phasellus ultrices interdum massa a lacinia. Nulla tempor lorem sit amet diam malesuada eu euismod enim mollis. Phasellus justo sapien, ultrices id cursus non, rutrum in massa. Sed rhoncus, nunc tincidunt congue molestie, eros neque venenatis magna, sed suscipit arcu sem ac odio. Donec eu ligula purus, nec mattis neque. Proin sodales quam sed urna luctus a adipiscing velit sollicitudin.

**Body:** Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras convallis porta augue, ut ullamcorper quam dictum eu. Phasellus ultrices interdum massa a lacinia. Nulla tempor lorem sit amet diam malesuada eu euismod enim mollis. Phasellus justo sapien, ultrices id cursus non, rutrum in massa. Sed rhoncus, nunc tincidunt congue molestie, eros neque venenatis magna, sed suscipit arcu sem ac odio. Donec eu ligula purus, nec mattis neque. Proin sodales quam sed urna luctus a adipiscing velit sollicitudin.

Cras hendrerit quam nec metus lacinia porta. Morbi volutpat lectus sed erat adipiscing ac auctor enim vestibulum. Cras iaculis vulputate dapibus. Nulla et velit mauris. Sed eleifend, massa nec facilisis pretium, lectus lorem egestas lorem, in mollis mi risus et nulla. Sed vehicula dictum mi in porttitor. Fusce aliquet consequat elit, et hendrerit orci ornare ac. Donec adipiscing, tortor ut vulputate mattis.

**Created at:** Data: 28/03/2013 Avui | Hora: 19:46:45 Ara |

**Published at:** Data: 30/03/2013 Avui | Hora: 20:46:51 Ara |

**Visits:** 0

#### Links

URI	Name	Eliminar?
Google Inc. Actualment: http://www.google.com/ Canviar: http://www.google.com/	Google Inc.	<input type="checkbox"/>

Afegir un/a altre/a Link.

Eliminar Desar i afegir-ne un de nou Desar i continuar editant Desar





## 2.3 – Configuració bàsica

- Unicament necessitem registrar el model

```
admin.site.register(models.Post)
```

- Si volem augmentar la funcionalitat per defecte, tenim disponible els ModelAdmin

```
class LinkInline(admin.TabularInline):
    extra = 0
    model = models.Link

class PostAdminModel(admin.ModelAdmin):
    inlines = [
        LinkInline
    ]
    list_display = ("title", "created_at", "published_at", "visits")
    list_filter = ("published_at",)
    search_fields = ["title"]

    class Meta:
        model = models.Post

admin.site.register(models.Post, PostAdminModel)
```



## 2.4 – Configuració avançada

- Fem ús de l'estructura ModelAdmin
- Tenim múltiples possibilitats de personalització
  - exclude
  - fields
  - list\_display
  - Inlines
  - list\_filter
  - search\_fields
  - etc.



## 2.4 – Configuració avançada

<https://docs.djangoproject.com/en/dev/ref/contrib/admin/>

```
class LinkInline(admin.TabularInline):
    extra = 0
    model = models.Link

class PostAdminModel(admin.ModelAdmin):
    inlines = [
        LinkInline
    ]
    list_display = ("title", "created_at", "published_at", "visits")
    list_filter = ("published_at",)
    search_fields = ["title"]

    class Meta:
        model = models.Post

admin.site.register(models.Post, PostAdminModel)
```



## 2.5 – Personalització

- Mètodes de personalització
  - Sobreescriptura de plantilles de l'admin
    - Per app o per model
      - app\_index.html
      - change\_form.html
      - change\_list.html
      - delete\_confirmation.html
      - object\_history.html
    - La resta, son generals a tot el projecte
    - Un exemple:
      - templates/admin/change\_form.html
      - templates/admin/my\_app/change\_form.html



## 2.5 – Personalització

- Mètodes de personalització
  - Reemplaçar els templates
  - AdminSite
    - S'encarrega de donar les directives sobre els paths per defecte
    - Exemple:
      - AdminSite.login\_template
    - Útil si un mateix projecte vol tenir diferents admins a diferents URIs
      - Per exemple, la construcció d'un site de minisites



## 3 – Formularis

- 3.1 – Treballar amb Forms
- 3.2 – Formsets



## 3.1 – Treballar amb Forms

- Què son els Forms?
  - Son abstraccions dels propis formularis, que:
    - Encapsulen la lògica del formulari
    - Permeten un rendering a diferents nivells
  - Automatitzen el processament de dades d'entrada
  - Son molt recomanables per mantenir el codi net

<https://docs.djangoproject.com/en/dev/topics/forms/>



## 3.1 – Treballar amb Forms

- Tipus de formularis disponibles
    - Form
      - Es tracta d'un formulari de propòsit genèric, al qual s'assignen els atributs arbitràriament
    - ModelForm
      - Subclasse de l'anterior, que s'acopla al model designat, i genera automàticament un camp de formulari per cada camp del model assignat
      - Altament configurable
- <https://docs.djangoproject.com/en/dev/topics/forms/modelforms/>





## 3.1 – Treballar amb Forms

```
from django import forms

class ContactForm(forms.Form):
    subject = forms.CharField(max_length=100)
    message = forms.CharField()
    sender = forms.EmailField()
    cc_myself = forms.BooleanField(required=False)
```

```
from django.forms import ModelForm

class ArticleForm(ModelForm):
    class Meta:
        model = Article
```



## 3.1 – Treballar amb Forms

### Instanciació, validació i ús

```
form = ContactForm()  
  
form = ContactForm(request.POST)
```

```
form.is_valid()  
  
form.cleaned_data
```

```
{{ form.as_p }}  
  
{{ form.as_table }}  
  
{{ form.subject.errors }}  
{{ form.subject.label_tag }}  
{{ form.subject }}  
  
{% for field in form.fields %}
```

```
form = ArticleForm()  
  
form = ArticleForm(instance=article)  
  
form = ArticleForm(request.POST)  
  
form = ArticleForm(request.POST, instance=article)
```



## 3.2 – Formsets

- Què és un FormSet?
  - És un mètode disenyat per treballar amb diversos formularis d'un mateix tipus
  - S'utilitza de manera anàloga als formularis de Django ja que, d'una manera abstracta, son simplement agregacions d'aquests
  - Útils per expressar relacions 1 – N

<https://docs.djangoproject.com/en/dev/topics/forms/formsets/>



## 4 – Tags i filtres

- 4.1 – Què son?
- 4.2 – Estudi dels built-ins
- 4.3 – Creació de tags pròpis



## 4.1 – Què son?

- Què son els filtres?
  - Funcions simples sobre un valor, que el converteixen en quelcom més elaborat
  - S'utilitzen principalment a templates
  - Ens ajuden a donar format i aplicar canvis controlats a les dades d'una manera visual i no intrusiva



## 4.1 – Què son?

- Què son els tags?
  - Son funcions que s'apliquen generalment als templates
  - Les funcions solen retornar quelcom
  - El retorn pot ésser tant elaborat com es vulgui
    - Fins i tot, una part del pròpi template
  - En la seva forma genèrica consten de:
    - Compilació: obté la informació necessària per la funció i aplica un parser als paràmetres d'entrada
    - Renderitzat: s'encarrega del rendering de la funció



## 4.2 – Estudi dels built-ins

- Functionals
  - for
  - if
  - extends
  - include
- Altres
  - escape
  - slugify
  - date

<https://docs.djangoproject.com/en/dev/ref/templates/builtins/>



## 4.3 – Creació de tags pròpis

- Podem crear la part del compilat i del renderitzat per separat
  - Tag estàndar
- Tenim disponibles classes elaborades per tal de facilitar casos d'ús comuns
  - Simple tags
  - Inclusion tags





## 4.3 – Creació de tags pròpis

- Per crear nous tags, únicament és necessari
  - Carpeta templatetags dins una de les nostres apps
  - Definir register

```
from django import template  
register = template.Library()
```

- Implementar els tags



## 4.3 – Creació de tags pròpis

```
@register.filter(is_safe=True)
def custom_float_format(value):
    try:
        return "%.2f" % float(value)
    except:
        return value

@register.simple_tag
def print_a_number():
    return unicode(random.random()*100)

@register.inclusion_tag('public/outstandings.html', takes_context = True)
def set_sidebar_outstandings(context, num_outstandings):
    request = context.get("request")
    outstandings = models.Outstanding.objects.all()[0:num_outstandings]
    return {
        'current_id': request.user.id,
        'outstandings': outstandings
    }
```



## 4.3 – Creació de tags pròpis

```
{% load my_tags %}

{{ my_var|custom_float_format }}
{% print_a_number %}
{% set_sidebar_outstandings 8 %}
```



## 5 – Projecte 2: Un Diari

- Crearem un nou site per un negoci d'un diari que vol generar el seu contingut online
- Descarreguem el projecte de la següent URI

[https://mega.co.nz/#!sNZTmRgl!MvKOsi\\_pxyPHR2x9h15hLUCHoHwk8eZ9inghyvmneac](https://mega.co.nz/#!sNZTmRgl!MvKOsi_pxyPHR2x9h15hLUCHoHwk8eZ9inghyvmneac)

- El vostre projecte ve preconfigurat per facilitar la feina



## 5 – Projecte 2: Un diari

- Models:
  - Noticia: contingut d'una notícia. Ha de tenir tota la informació, tant per llistats com per interiors, així com informació estadística.
  - Anunci: model simple per els anuncis del diari.
  - Secció: seccions en les que s'agrupen les notícies. Hauria de ser un model jeràrquic (una secció pot ser pare d'altres seccions).
- \* És requerit l'ús de migracions per generar el model de dades



## 5 – Projecte 2: Un diari

- Vistes:
  - Pàgina d'entrada: Ha de contenir una capçalera amb totes les seccions, i un llistat de les últimes notícies.
  - Llistat de notícies per secció: Ha de contenir un llistat de les subseccions, així com les notícies de la secció i subseccions, per ordre de novetat.
  - Interior de notícia: Ha de contenir tota la informació de la notícia. A més, es mostrarà un formulari per afegir comentaris, així com la llista dels comentaris de la pròpia notícia.
  - Contacte: Formulari per contactar amb el diari.



## 5 – Projecte 2: Un diari

- Plantilles
  - La plantilla principal ha de contenir, com a mínim:
    - Una capçalera, amb totes les seccions, i destacar la actual seleccionada
    - Un sidebar, a l'esquerra, amb els anuncis (3, mostrats aleatoriament)
    - Un sidebar a la dreta, amb els següents continguts:
      - 3 notícies més vistes
      - 3 notícies més comentades
  - Tot aquest contingut ha de ser renderitzat amb template tags



## 5 – Projecte 2: Un diari

- Admin
  - Cal crear una interfície d'administració utilitzant l'admin de Django
  - Volem un modeladmin per cada model existent, i un inline si hi ha jerarquia.
  - Els llistats han de contenir:
    - Cercador
    - Filtres
    - Diversos camps en el llistat





## 5 – Projecte 2: Un diari

- Ampliació (Opcional):
  - Afegir estadístiques als anuncis
  - Crear validacions pròpies als camps dels formularis
  - Afegir fieldsets pròpis a l'Admin



# Django

Gràcies i fins la propera sessió!



Barcelon**a**ctiva



Ajuntament  
de Barcelona

[bcn.cat/barcelonactiva](http://bcn.cat/barcelonactiva)  
[bcn.cat/cibernarium](http://bcn.cat/cibernarium)