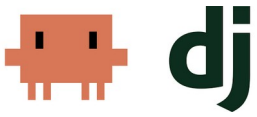
Corso Django con Claude Code

dr. Alessandro Dentella

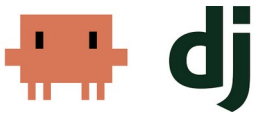
slides: https://corsi.e-den.it



Alessandro Dentella
adentella@thux.it

Django

The web framework for perfectionists with deadlines.



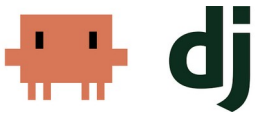
Alessandro Dentella
adentella@thux.it

Programma del corso

- Struttura progetto e glossario
  - urls.py
  - Settings
  - Templates
  - Apps
- Interazione
  - python manage.py shell/runserver
  - dj / dj r
- Modelli
  - fields
  - migration
  - ORM
  - QuerySet
- Admin
  - ModelAdmin
  - Inlines (Stacked, Tabular Ajax)
  - Advanced Search
- Viste
  - Flusso info da URL a response
  - Forms & fields
- Template & templatetags
- Commands
- Translations
- Configurazione:
  - Statici
  - Middleware
- Presentazione tool Thunder (jmb-*, dj)
- Deploy (uwsgi + nginx)

Glossario

- **Project**: collezione di setting per una istanza di django. Un progetto è composto di tante applicazioni dichiarate in una variabile chiamata `INSTALLED_APPS` (una lista)
- **App**: un modulo che normalmente ha dei modelli e delle viste. Ogni app è contraddistinta da una `app_label` unica in tutto il progetto: es.: `auth`
- **Model**: una classe che descrive una tabella del database e che viene definita ereditando da `django.db.models.Model`

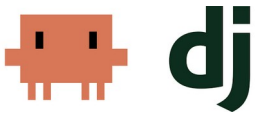


Alessandro Dentella
adentella@thux.it

Progetto di studio: Fellini

Per questo corso svilupperemo un progetto chiamato “Fellini” dove ovviamente avremo

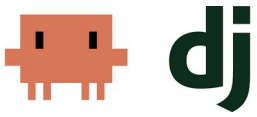
- una **app principale** chiamata “film” dall’ovvio contenuto: avrà modelli per film, registi, attori, gneri.
- Un’**altra app** potrebbe essere “Cinema” intendendo le sale cinematografiche.
- Useremo db sqlite che ha supporto integrato in Python.
- Vogliamo popolare velocemente i film usando il db pubblico TMDB
- Suggerisco di usare bene git per creare commit (claude code lo può fare bene) in modo da tracciare tutte le modifiche



Alessandro Dentella
adentella@thux.it

Claude Code

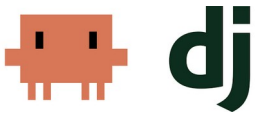
- Claude Code può aiutarci **molto** a creare il codice, ma abilitiamo almeno l'/output-style explanatory per comprendere cosa fa
- Considerate anche l' /outpuyt-style learning che lascia dei TODO allo user
- In queste slide suppongo che abbiate installato uv come package manager e lavoriate in una cartella locale con pieni privilegi.



Alessandro Dentella
adentella@thux.it

CLAUDE.md

- Cominciamo a raccontare a Claude cos’è questo progetto e cosa vogliano realizzare
- Aggiorneremo questo file che sarà inviato come contesto ad ogni nuova chat. Potremo anche chiedere a Claude di aggiornarlo se dimentichiamo di farlo



Alessandro Dentella
adentella@thux.it

Creazione venv

Recentemente sta imponendosi `uv` com package manager preferito. È scritto in rust ed ha una velocità esagerata rispetto agli altri package manager. Permette sia

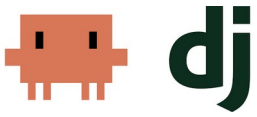
- `uv pip install` …
- `uv add  pkg` (usa il nuovo pyproject.toml, preferibile)

Quindi un sessione diventa:

```
uv init my-proj-dir
cd my-project-dir
uv venv (crea .venv)
uv add django  / uv pip install django (alternative)
uv run
```

Il fatto di preporre `uv` ai comandi garantisce l'uso dell'interprete corretto, quello in .venv

Non è l'unico modo ma è quello più intuitivo.



Alessandro Dentella
adentella@thux.it

Creare il progetto django e venv

```
~/src/DiangoProjects$ uv init fellini
Initialized project `fellini` at
`/home/sandro/src/DiangoProjects/fellini`
~/src/DiangoProjects$ ls -la film
totale 16
drwxrwxr-x 1 sandro sandro 118 nov 23 12:40 .
drwxrwxr-x 1 sandro sandro  18 nov 23 12:40 ..
drwxrwxr-x 1 sandro sandro  98 nov 23 12:40 .git
-rw-rw-r-- 1 sandro sandro 109 nov 23 12:40 .gitignore
-rw-rw-r-- 1 sandro sandro  82 nov 23 12:40 main.py
-rw-rw-r-- 1 sandro sandro 150 nov 23 12:40 pyproject.toml
-rw-rw-r-- 1 sandro sandro   5 nov 23 12:40 .python-
version
-rw-rw-r-- 1 sandro sandro   0 nov 23 12:40 README.md
~/src/DiangoProjects$ cd fellini
~/src/DiangoProjects/film$ uv add django
Using CPython 3.13.7
Creating virtual environment at: .venv
Resolved 5 packages in 13ms
Installed 3 packages in 170ms
 + asgiref==3.11.0
 + django==5.2.8
 + sqlparse==0.5.3
```

.git creato implicitamente

.venv creato implicitamente

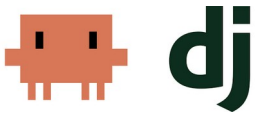
```
~.../fellini$ uv run django-admin startproject core .
~.../fellini$ tree
.
├── core
│   ├── asgi.py
│   ├── __init__.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
├── main.py
├── manage.py
├── pyproject.toml
├── README.md
└── uv.lock
```

Configurazione dell'URL dispatcher. Possiamo considerarlo l'indice del progetto

manage.py: handler per django: uv run manage.py

nuovo modo Python di configurare i progetti

Tiene traccia dei pacchetti python installati e della loro versione (miglioramento di requirements.txt)



Alessandro Dentella
adentella@thux.it

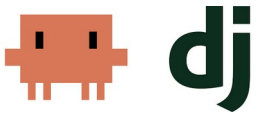
manage.py

- È la script principale di interazione con django, ha molti sottocomandi e tanti possono essere creati facilmente

  `shell`: apre una shell python (ipython se disponibile)
  `check`: esegue una serie di controlli che tutti funzioni
  `runserver`: lancia il web server
  `collectstatic`: copia gli statici dove saranno usati
  `migrate`: crea / migra il db
  `makemigrations`: crea una script per le migrazioni

- Io uso un alias che ho fatto io chiamato `dj` che ha anche il completamento… ne esiste uno chiamato `dj-cmd`, suggerisco di installarlo:

```
pip install dj-cmd
```



Alessandro Dentella
adentella@thux.it

Struttura di una app

```
(test):/tmp/mioweb$ ./manage.py startapp myapp

(test):/tmp/mioweb$ tree myapp
myapp
├── admin.py
├── __init__.py
├── migrations
│   └── __init__.py
├── models.py
├── tests.py
└── views.py
```

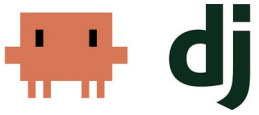
Modulo di conf dell'admin

Cartella dello storico delle modifiche ai modelli

Modulo di definizione dei modelli

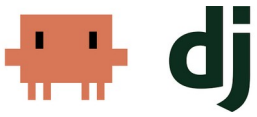
Modulo contenente le viste,
ovvero funzioni e classi che ritornano una “Response”

Modulo dei test



Alessandro Dentella
adentella@thux.it

- Creare un progetto in un virtualenv (per semplicità usate `uv`)
- `./manage.py shell`. Verificate che non è **ipython**
- Installate `ipython` e verificate che **django** lo usi
- Analizzate il modulo `settings` che viene creato per default e cercate di intuire il significato delle variabili
- Analizzate `django.apps.apps` e cercate di capire quali apps sono installate



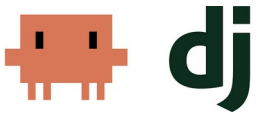
Alessandro Dentella
adentella@thux.it

apps

- creare una cartella apps e crearvi dentro una app chiamata `myapp`
- verificare se è importabile dalla shell di django, ed eventualmente capire come fare a fare sì che sia importabile con il comando:

```
import myapp
```

suggerimento: modificate sys.path in manage.py



Alessandro Dentella
adentella@thux.it

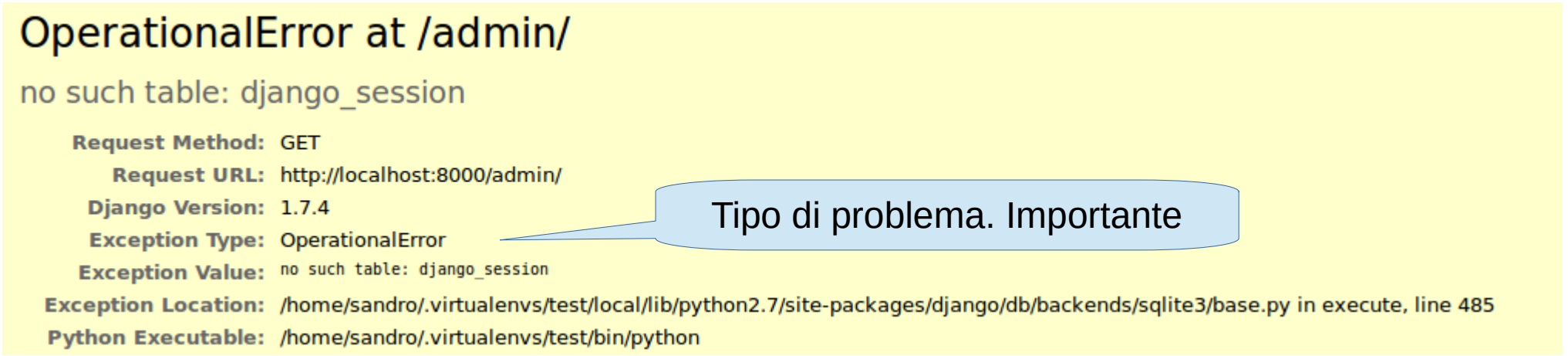
runserver

Django ha un server per sviluppo già compreso (batterie incluse) che può essere avviato con un comando della utility `manage.py`:

```
(test)sandro@bluff:/tmp/mioweb$ ./manage.py runserver
Performing system checks...

System check identified no issues (0 silenced).

You have unapplied migrations; your app may not work properly until
they are applied.
Run 'python manage.py migrate' to apply them.

November 06, 2017 - 08:20:12
Django version 1.11.7, using settings 'web.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Alessandro Dentella
adentella@thux.it

- Possiamo quindi visitare
  - http://localhost:8000
  - http://localhost:8000/admin/
- Il secondo link ci dà errore



Alessandro Dentella
adentella@thux.it

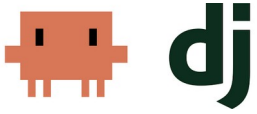
Creare il database

- Non dovrebbe stupirci che non trovi alcuna tabella visto che non abbiamo creato alcun db...

```
(test)sandro@bluff:/tmp/mioweb$ ./manage.py migrate
Operations to perform:
  Apply all migrations: admin, contenttypes, auth, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying sessions.0001_initial... OK
```

- Questo crea un db con sqlite, configurazione di default. Possiamo arrivarci (richiede forse `apt-get install sqlite3`) direttamente con:

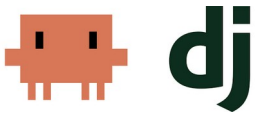
```
(test)sandro@bluff:/tmp/mioweb$ ./manage.py dbshell
SQLite version 3.8.2 2013-12-06 14:53:30
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> .tables
auth_group                  auth_user_user_permissions
auth_group_permissions      django_admin_log
auth_permission             django_content_type
auth_user                   django_migrations
auth_user_groups            django_session
```



Alessandro Dentella
adentella@thux.it

Interfaccia admin

- Possiamo quindi rivisitare http://localhost:8000 e ci accorgiamo che ci serve un utente che ancora non è stato creato: usiamo `manage.py` per creare un ***superuser***. A voi di capire come...
- Una volta entrati provate a creare altri utenti superuser e non, ***staff*** e non. Verificate che un utente non staff non può loggarsi da admin.
- Verificare che anche un utente della staff non si può loggare se non è ***attivo***.

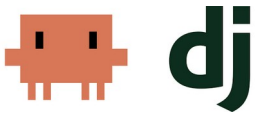


Alessandro Dentella
adentella@thux.it

Popoliamo il db

Per varie attività ci fa comodo avere il db pieno, facciamo quindi delle script per importare dati da un db pubblico TMDB. Possiamo usare questa API KEY: 377425889a46752b65961aa7938dcedb

Una script completamente integrata in django che può utilizzare i suoi modelli, il suo ORM (Object Relational model, che studieremo a breve) ed ogni altro codice nostro o scritto da django è un “command”



Alessandro Dentella
adentella@thux.it

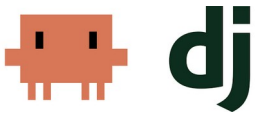
Commands

Esiste un modo di invocare django da riga di comando. Lo abbiamo usato molte volte con tutti i comandi tipo:

- dj migrate
- dj compilemessage
- dj createsuperuser

È possibile crearli custom, ad esempio per importare dati da cron, inviare mail, reimpostare una password...

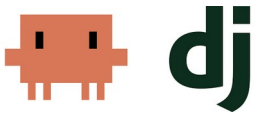
Occorre creare uno script nella cartella `management/commands` della nostra app. Claude Code sa cosa fare...



Alessandro Dentella
adentella@thux.it

Struttura

- Nei commands è possibile mettere **opzioni** usando il pacchetto di libreria argparse
- L'azione che dobbiamo fare è contenuta nel metodo **handle**



Alessandro Dentella
adentella@thux.it

import_film

- Creiamo un command import_film, capace di importare film da TMDB. Vogliamo avere opzioni per:
  - Max num film
  - Quali generi
  - Che periodo
  - Quale nazione
- Gestire doppioni

Usiamo git
Verifichiamo cosa ha prodotto!

Admin

- Chiediamo a Claude di abilitare l’app film nell’admin
- Verifichiamo come l’ha fatto
- Chiediamo di aggiungere le locandine dei film non come link ma come visualizzazione
- Verifichiamo come ha fatto