

erstes Django Projekt

petproject

neues Projekt
mit django-
admin
erstellen

`django-admin.py startproject petproject`

Projektname:

ohne Leerzeichen

ohne Umlaute

ohne Sonderzeichen

keine Zahlen

Dateistruktur

Django_projects

- petproject

 manage.py

- petproject

 __init__.py

 settings.py

 urls.py

 wsgi.py

 asgi.py

 db.sqlite3 (wird nach dem ersten Ausführen von python manage.py makemigrations erstellt)

manage.py

Ein Kommandozeilenprogramm um mit Django zu interagieren. Eigentlich ein Wrapperprogramm um django-admin, angereichert mit projektspezifischen Settings.

Mehr zu manage.py und django-admin:

<https://docs.djangoproject.com/en/3.1/ref/django-admin/>



Settings.py

Die Einstellungen für das Projekt. Datenbank, statische Dateien, installierte Apps, alles wird hier eingestellt

Mehr dazu:

<https://docs.djangoproject.com/en/3.1/topics/settings/>

urls.py

alle projektspezifischen URLs für die Web-Applikation werden hier eingetragen. Ein sauberes URL-Schema ist von enormer Wichtigkeit für ein Webprojekt

`http://example.com/pets/pet/2`

statt

`http://example.com/shopwebsite.php?productID=2&SHOW=2`

<https://docs.djangoproject.com/en/3.1/topics/http/urls/>

Was ist ein Projekt?

Ein Projekt ist nur der Rahmen für eine Webseite. Die eigentliche Funktionalität wird durch die Applikationen implementiert, die in das Projekt eingebunden werden.

Genau wie für Projekte bietet Django auch für Applikationen ein Werkzeug an, um das Grundgerüst einer Applikation zu erzeugen:

`manage.py`

eine neue App im Projekt erzeugen

mit `manage.py` lässt sich komfortabel eine `neue App` erstellen

```
python manage.py startapp pets
```

Für den Namen der App gilt das gleiche, wie für das Projekt.

Die App wird in Zukunft unter <http://127.0.0.1/pets> erreichbar sein

Dateistruktur

projekte

- petproject

 manage.py

 - petproject

 - pets

 admin.py

 models.py

 urls.py

 views.py

 tests.py

 __init__.py

 db.sqlite3

Die Datei urls.py ist nicht in der App automatisch angelegt worden. Das bitte nachholen!

urls.py (App)

Hier liegen alle app-spezifischen URLs. Sie ergänzen die projektspezifischen URLs.

`http://127.0.0.1/pets/hello`

Die Funktion `path()` erwartet drei Parameter:

`path('hello', views.helloview, name='helloview')`

der 1. (positionelle) Parameter ist der Teil der URL, die auf die Funktion mappen soll

der 2. (positionelle) Parameter ist die Funktionsreferenz aus der view

der 3. (Keyword) Parameter ist ein Name für diese Route (sollte gleich der View sein)

der Teil `pets` wird in den URLs für das Projekt behandelt



<https://docs.djangoproject.com/en/3.1/topics/http/urls/>

mehr über URLs

URLs für die App

```
from django.urls import path  
from . import views  
  
urlpatterns = [  
    path('hello', views.helloview, name='helloworld'),  
    path('check', views.check, name='check'),  
]
```

Das name-Argument ist für reverse-Lookup nutzbar, aber optional.

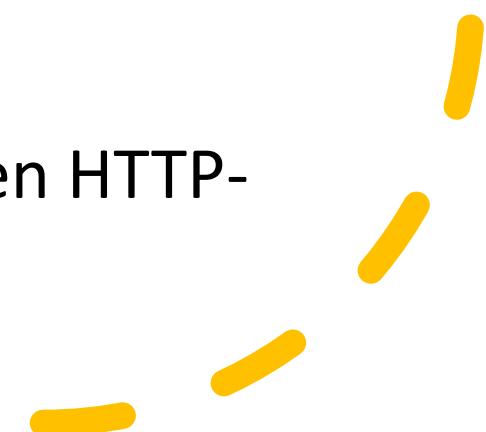
A large orange circle is positioned on the left side of the slide, covering approximately one-third of the vertical space.

views.py

Hier liegen die appspezifischen views. Views existieren nur in den Apps, nicht auf Projektebene.

Views im Kontext von Django sind Controller, die Userinput entgegennehmen, Daten vom Model holen oder im Model speichern und den HTTP-Response an den Server bzw. den User weitergeben.

Jede View benötigt zwingend einen HTTP-Response!

Three small, horizontal yellow bars of varying lengths are scattered in the bottom right corner of the slide.

ein einfaches Beispiel für eine View-Funktion

```
from django.http import HttpResponse

def hello(request):
    """http://127.0.0.1:8000/pets/hello"""
    return HttpResponse("Hello World!")

def second_view(request):
    """http://127.0.0.1/my_first_app/second"""
    return HttpResponse("noch eine zweite View!", content_type="text/plain")
```

Fazit

Unser erstes kleines Projekt ist nun fertig.
Wir hatten mit django-admin ein neues, leeres
Projekt namens **petproject** erstellt.

In diesem Projekt wurde dann mit manage.py
eine neue App names **pets** erstellt.

Abschließend haben wir zwei einfache Views
angelegt. Damit wir diese Views auch anzeigen
können, hatten wir die entsprechenden URLs
dafür definiert (einmal die **projektspezifische URL**
(zur App), einmal die **appspezifischen URLs** (zu
den Funktionen innerhalb der App)).