

setup

creating a project	<code>django -admin startproject roject mysite</code>
changing settings	<code>edit mysite/settings.py</code>
database setup	<code>python3 manage.py migrate</code>
run the server	<code>python3 manage.py runserver</code>
and in browser:	<code>http://127.0.0.1:8000/</code>
test shit in an interactive console	<code>python3 manage.py shell</code>

apps

creating an application	<code>python3 manage.py startapp blog</code>
add the app name to INSTALLED_APPS	<code>'blog.apps.BlogConfig'</code>
models are kept in models.py	
add, edit and delete model in admin.py	

django queriesets

getting the model classes	<code>from app_name.models import model_name</code>
getting the user model	<code>from django.contrib.auth.models import User</code>
see all objects of a certain model	<code>Class_name.objects.all()</code> (is iterable)
create an object	<code>Class_name.objects.create(atribute=)</code> don't have to save for it to be reflected
accessing the object via attribute	<code>Class_name.objects.get(atribute="value")</code>
filtering (can return multiple, iterable)	<code>Class_name.objects.filter(atribute="value")</code>
filtering (contains)	<code>Class_name.objects.filter(title__contains="phrase")</code> <code>published_date__lte=timezone.now()</code>

django queriesets (cont)

ordering	<code>Class_name.objects.order_by("attribute")</code> "-attr" does the reverse
you can combine ordering and filtering	<code>Class_name.objects.filter(cond).order_by()</code>
delete	<code>obj.delete()</code>

forms

import	<code>from django import forms</code> <code>from .models import model_name</code>
--------	--

security

login decorator	<code>from django.contrib.auth.decorators import login_required</code> <code>@login_required</code>
-----------------	--

model attributes

model attribute	description	args
<code>models.CharField</code>	text with a limited number of characters	<code>max_length</code>
<code>models.TextField</code>	text field with unlimited content	
<code>models.DateTimeField</code>	datetime field	<code>default, blank, null, id</code>

databases

creating a database	<code>python3 manage.py migrate</code>
update changes to database	<code>python3 manage.py makemigrations</code> <code>python3 manage.py migrate</code>
register database in admin	<code>admin.site.register(Model_name, app_name/admin.py)</code>

admin

register a model `admin.site.register(Post)`

create superuser `python3 manage.py create_superuser`

pythonanywhere deployment

1. create the api token
2. new bash console
3. in bash: `pip3 install --user python any where`
4. `python3 manage.py collectstatic`
5. to update website, go to your directory and pull code
6. reload at web
7. work on your directory
8. `python3 manage.py collectstatic`

basic styling

get bootstrap `<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">`

`<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap-theme.min.css">`

add static files make a *static* folder with css files inside *app_name* folder

`{% load static %}`

`<link rel="stylesheet" href="{% static 'css/blog.css' %}">`

create a block `{% block content %}`

`{% endblock %}`

block content `{% block content %}`

`{% endblock %}`

connect a block to the base `{% extends 'blog/base.html' %}`

link a url within a site `a href="{% url 'url_name' pk=post.pk %}"`

where pk stands for primary key

to use the link `path('whatever/<int:pk>', views.view_name, name='url_name')`

