Templateless Django Cheat Sheet

Command Line

Create Project
```
$ django-admin startproject {project-name}
```

Create App
```
$ python manage.py startapp {app-name}
```

Create Migration Files (DB not yet affected)
```
$ python manage.py makemigrations {app-name}
```

Perform migration on DB
```
$ python manage.py migrate
```

Run tests
```
$ python manage.py test
```

Run Single Test
```
$ python manage.py test {app_name}.tests.{TestClass}.{test_method}
```

Run server
```
$ python manage.py runserver
```

Run server allowing all IPs
```
$ python manage.py runserver 0.0.0.0:{port_number}
```

Important Files

Web app settings (database, apps, ...)	`{project_name}/settings.py`
Urls/Routes (maps "views" to urls)	`{project_name}/urls.py` `[can also have sub-urls files via "include(*)"]`
Views/Controllers (retrieve data via "models" and renders response)	`{app_name}/views.py`
Models (creates columns for tables)	`{app_name}/models.py`
Test	`{app_name}/tests.py`

Notable Settings

```
INSTALLED_APPS, DATABASES, BASE_DIR, LOGGING, DEBUG,
ALLOWED_HOSTS, DATABASE_ROUTERS, APPEND_SLASH,
CORS_ORIGIN_ALLOW_ALL, AUTHENTICATION_BACKENDS
```

Urls

GET, POST Url	`url(r'^smth/?$', views.method)`
GET, PUT, DELETE Url	`url(r'^smth/(?P[0-9]+)/?$', views.method)`
Import CRSF Exempt (not recommended for production)	`from django.views.decorators.csrf import csrf_exempt`

Views

View Declaration	`def view_method(request):`
JSON Request Data	`request.data['json_field']`
Get user	`request.user`
Request Method	`request.method`
Request Cookies	`cookies_dict = request.COOKIES`
Response	`return HttpResponse(data, content_type={..}, status_code={..})`
Response Header	`response = HttpResponse(..); response['header'] = value`
Response's Cookie	`HttpResponse(..).set_cookie ('key', 'value')`
JSON Response (subclass of HTTP Response)	`return JsonResponse({'key': 'val'})`

Models

Model Declaration	`from django.db import models`
	
	`class SomeStuff(models.Model):`

Fields Declaration	`models.SomeField(param=val)`
Common Field Options	`null = True; blank = True; choices = (('CD','Code')); default = "default_value"; primary_key = True; unique = True;`
Boolean Field	`BooleanField(**options)`
Char Field	`CharField(max_length={..}, ..)`

Models (cont')

Text Field	`TextField(..)`
Email Field	`EmailField(max_length=300, ..)`
URL Field	`URLField(max_length={..},..)`
Date Field	`DateField(auto_now=True, ..); DateTimeField(..)`
Integer Field	`IntegerField(..)`
Decimal Field	`DecimalField(max_digits=3, decimal_place=2, ..)`
Float Field	`FloatField(..)`
One-to-one relationship	`OneToOneField(someModel, on_delete={..}, ..)`
Many-to-one relationship	`ForeignKey(someModel, on_delete={..}, ..)`
Many-to-Many relationship	`ManyToManyField(someModel, ..)`

Making Queries

Create object	`obj = Model.objects.create(field=val,..); obj.save()`
Get single object	`Model.objects.get(field=val,..)`
Filter objects (returns queryset)	`Model.objects.filter(field=val,..)`
Advance filtering	`Model.objects.filter(field__{whereop}=..)`
Get all objects of a model	`Model.objects.all()`
Exclude objects	`Model.objects.filter(..).exclude(field=val,...)`
Check objects exists	`Model.objects.filter(..).exists()`
Update object	`obj['key'] = 'new_val'; obj.save()`
Add many-to-many objects	`obj.many_fields.add(some_obj)`
Add many-to-one objects	`one_obj.manymodelname_set.add(some_obj)`
Delete object	`obj.delete()`

BUILD-2-MASTER

Created on: April9, 2016