Django – The best Of

Co najbardziej podoba mi się w Django.

# ORM

Jeden z podstawowych elementów, który wbudowany jest w Django. Z wykorzystaniem DjangoORM możliwe jest, podobnie jak w Hibernate, utworzenie pełnego modelu danych, który zostanie następnie odzwierciedlony w bazie danych. Na tym jednak podobieństwa się kończą, ponieważ DjangoORM potrafi o wiele więcej, a raczej Django poprzez zastosowanie własnego silnika ORM potrafi więcej z niego wyciągnąć.

Największą korzyścią jest wbudowany, wewnętrzny, silnik zapytania dostępny w dwóch wersjach. W pierwszej z nich możemy, korzystając ze statycznych metodach na klasach lub metod dostępnych na obiekcie (domniemane jest tutaj wykorzystanie relacji klucz główny - klucz obcy oraz łączenia tabel), wykonywać tak zwane **type-safe[[1]](#endnote-1)** kwerendy.

Ponadto warto tutaj wspomnieć również o kolejnych elementach, składających się na cały framework Django, jakimi są:

* widoki,
* formularze,
* panel administracyjny;

gdzie, wewnętrzne mechanizmy Django są w stanie:

* zbudować panel administracyjny dla danych modeli w sposób OOTB[[2]](#endnote-2), tj. pozwalać administrować stanem bazy w sposób natychmiastowy,
* automatycznie generować kontrolki HTML zgodne z typem pól w bazie danych,
* wykonywać zapytania na bazie danych

DjangoORM posiada także mechanizm migracji danych ze starszych do nowszych wersji odpowiadającego im modelu. Nie jest to funkcja zbyt często spotykana w frameworkach, a w Django była dostępna jeszcze zanim została w nim bezpośrednio osadzona.  
Ostatecznie nie jest trudnością zmiana bazy danych. Model danych stoi na wysokim poziomie abstrakcji w stosunku do silnika, a motto rw

# MVC [ MVT ]

MVC w wykonaniu Django jest bardzo proste i łatwe do przyswojenia, a w wykonaniu omawianego szkieletu aplikacji znany jest, jako MVT.

Model, implementowany na poziomie DjangoORM, jest interfejsem do bazy danych, mimo faktu, że wiele z jego elementów Django potrafi użyć w warstwie przejściowej między logiką a faktycznym widokiem.

Widok należy rozumieć, jako warstwę logiki. Na tej płaszczyźnie możliwe jest wyciągania danych z bazy, ich wstępna obróbka, walidacja i ostatecznie przekazanie do warstwy danych celem wykonania jednej z 4 operacji CRUD. Część z tych operacji można pominąć (tj. nie implementować ich), ponieważ Django posiada [zestaw klas](https://docs.djangoproject.com/en/1.8/ref/class-based-views/generic-editing/), które automatycznie się tym zajmują. Developer musi się jedynie zatroszczyć o wskazanie, dla jakiego modelu buduje widok, jakie pola mają być wyświetlone oraz dodania URL.

Ostatecznie szkielet [z angielskiego template] jest to warstwa dynamicznego języka HTML (lub innego wspieranego przez Django i tłumaczonego później do HTML). Niektóre z możliwych operacji do operacje warunkowe, pętle itp. Dobrą analogią jest tutaj język szkieletów zaimplementowany w AngularJS z tą różnicą, że w Django mamy możliwość operowania na zmiennych pochodzących bezpośrednio z serwera, a nieprzesłanych w odpowiedzi HTTP.

## Automatyczna walidacja danych

Jest to jedna z funkcji, o których wspomniałem wcześniej, a o których chciałbym napisać więcej. Powodem tego jest jest ponownie DjangoORM. Walidacja odbywa się powiem na podstawie ograniczeń, jakie nałożone zostały na bazę danych i co więcej jest ona całkowicie transparentna dla użytkownika i developera. Ponadto błędy są dostępne, po fazie walidacji, w widoku i co więcej są one dodatkowe powiązane z polami, dla których zostały wykryte.

# Admin Panel

Panel administracyjny jest często takim elementem aplikacji dla utworzenia, którego przewiduje się całe sprinty. W Django taki element jest dostępny od samego początku. Ponadto dokładnie odzwierciedla on model danych i pozwala w pełni go edytować. Jeśli nie jest to wystarczająco możemy napisać własne modyfikacji odnoszące się do tego:

* jakie pola, w jakim miejscu będą wyświetlana dla danego modelu danych,
* jakie pole będą edytowalne,
* w jakim porządku można obiekty sortować i wyszukiwać,
* zmiany domyślnych kontrolek wyświetlanych dla pól modelu danych,
* dodatkowego stylowania;

Istnieje również możliwość skorzystania z innych paneli administracyjnych, takich jak np. [ten](http://djangosuit.com/).

# Podsumowanie

Wymieniłem jedynie kilka elementów, dla których warto zainteresować się Django, a który mnie osobiście przyciągnęły do niego w pierwszej kolejności. Jeśli pomyśleć by o złotej zasadzie DRY, Django jest książkowym przykładem jej realizacji. W bardzo dużej ilości przypadków developer nie musi robić praktycznie nic albo musi zrobić bardzo mało, jeśli jego celem jest napisania trochę mniej niż trywialnej aplikacji. Co nie zmienia faktu, że i dla dużych projektów Django jest również doskonałym fundamentem. Warto tutaj wspomnieć chociażby o [Horizon](https://github.com/openstack/horizon/), realizującym panel administracyjny dla chmur OpenStack.

1. Np. zapytanie na rzecz klasy D zwróci obiekty klasy D. [↑](#endnote-ref-1)
2. Out-Of-The-Box [↑](#endnote-ref-2)