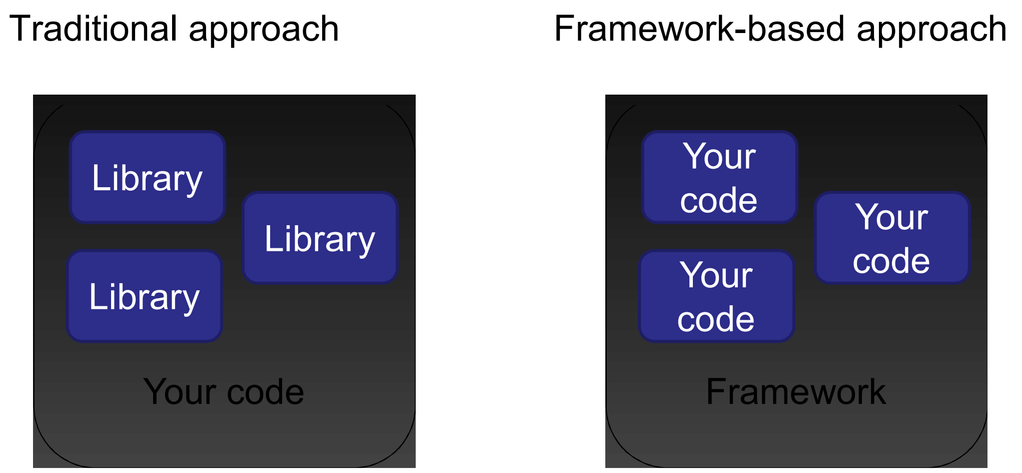
Django App Architecture Writing Specifications

Maxim Mozgovoy

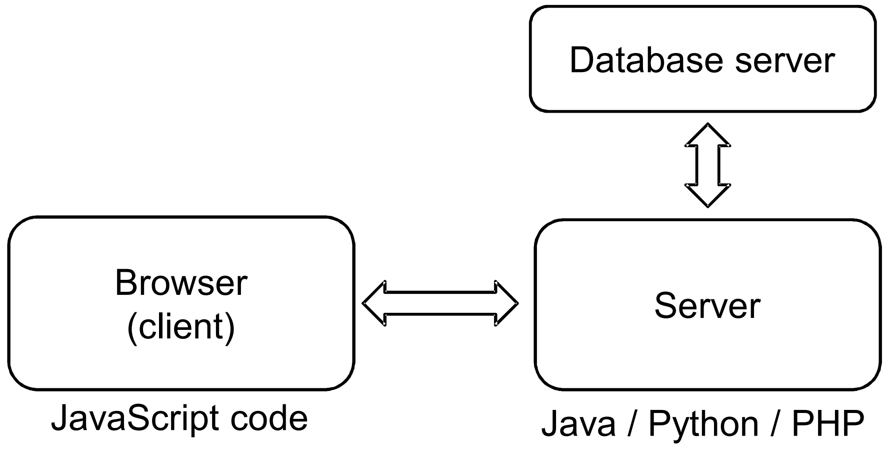
Libraries vs Frameworks



Development Process

Traditional approach: the components are developed separately and can be based on different technologies.

Framework approach: the framework provides a ready template for each component, and we customize them further.



What is Django

- Django is perhaps the most popular framework for webapp development (appeared in 2005, recent release: May, 2019)
- Pros: large community, great number of existing components for typical tasks, highly functional initial template.
- Cons: somewhat high learning curve and limited flexibility.
- Closest competitor: Flask (lower learning curve, lightweight, requires more effort to build a complete application).
- Many other frameworks are available: Pyramid, Ruby on Rails, Drupal, Symfony…

Django Setup

- Django works with Python 2 and Python 3.
  In our examples I will use Python 3.

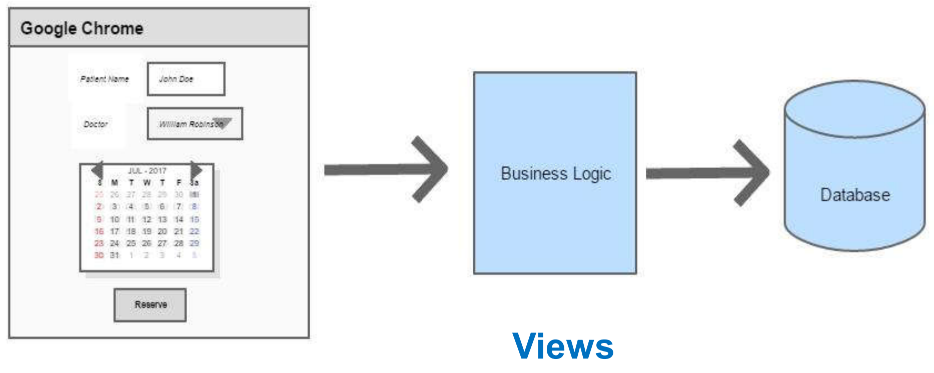
- Install Python 3 from
  `https://www.python.org`

- Install Django:
  ```
  python -m pip install django
  ```

- Check that everything is all right:
  ```
  python -m django --version
  ```

  (it says “2.2.1” on my machine)

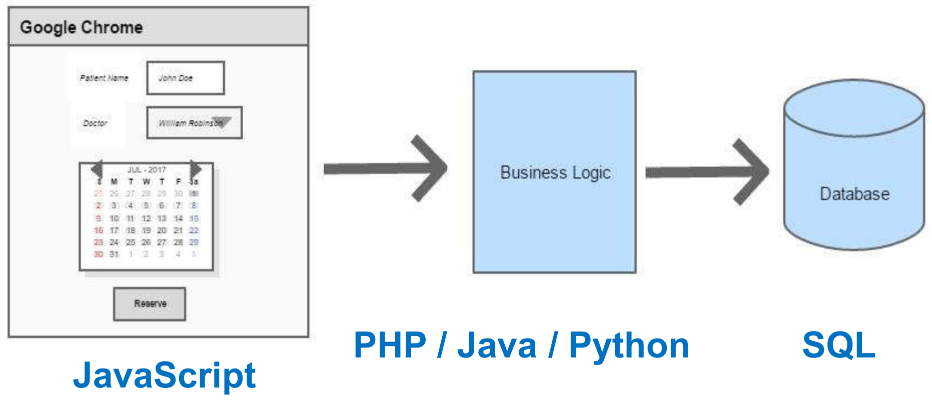
Django App Architecture



Conceptually, Django implements classical web app architecture, consisting of data, application, and presentation layers.

The core of Django application is a system of *views* that binds together database and presentation layers.

Django App Architecture

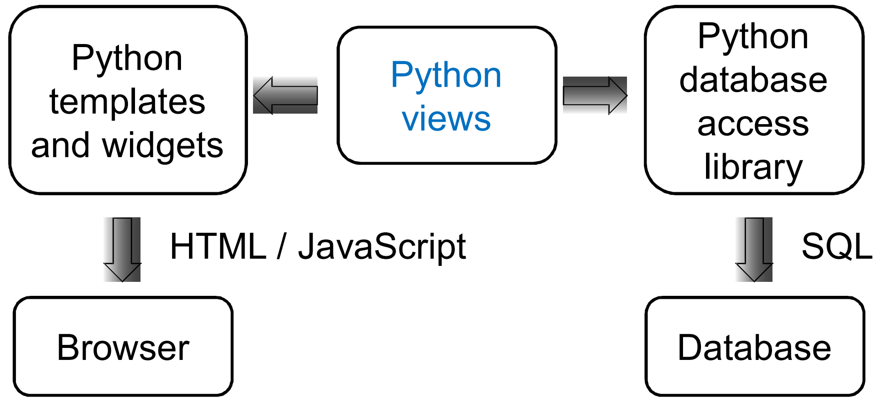


In classical workflow, all parts of an application are developed independently using different technologies.

In Django, everything is programmed in Python!

Django App Architecture

How it works?



A programmer writes queries and responses using Django Python objects, and Django transforms them into SQL or HTML.

Django App Architecture

- Large part of learning Django is to understand how it deals with databases and forms HTML documents.
- The set of Django widgets (user-input HTML elements) is rather limited.
- However, it is possible to use Django in combination with a front-end framework, such as Angular, React, or Vue.
- In any case, before writing an app, we'll need a specification ☺

How to Write a Spec

In real projects, there are two kinds of specifications:

•**Functional**: how a product works from the user's perspective. Concentrates on screens, menus, etc.

•**Technical**: how a product should be implemented. Concentrates on languages, algorithms, tools, database structure, etc.

How to Write a Spec

An easy way to write a functional specification is to prepare a number of sample scenarios.

E.g., "User John opens the website. He sees the page containing the input box *input your name* and the OK button…"

The interface and its behavior should be described in detail, possibly with pictures. However, no technical discussion is necessary at this point.

How to Write a Spec

Technical specification should discuss the project architecture. The system should be separated into modules, and each module discussed in detail.

Here all relevant points are important: the languages used, the toolset used, performance and space requirements, database structure, technical notes…

In our case, we will need a simplified document: just to make sure everyone understands what is being done.

How to Design and Test a Webapp

Show me your flowcharts and conceal your tables, and I shall continue to be mystified. Show me your tables, and I won’t usually need your flowcharts; they’ll be obvious. (Fred Brooks)

•Start with the database. Think which data structures you will need for the application. You can test your database without writing anything else!

•Write code that returns some results for fixed (predefined) user input values without actually processing user requests. Now you can test the logic of the app without input.

•Write the simplest possible user interface and test the correctness of whole application.

•Design a nice rich GUI.

My Own Homework

My example project is
Dentistry Reservation Website.
(Study my specification!)

Work plan:

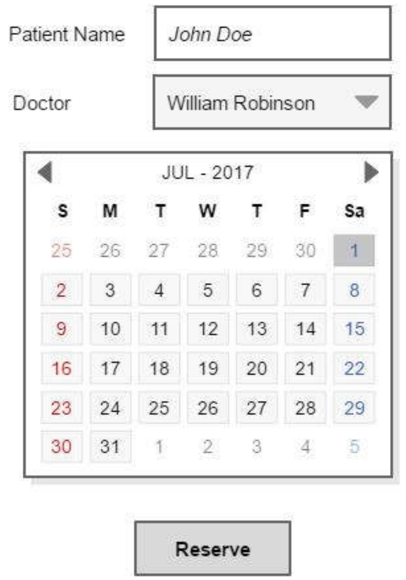
•Design the database.

•Design the application layer.

•Design the user interface.

•Improve the user interface.

I will also demonstrate Django concepts on *FirstApp* small demo application.



Patient Name: *John Doe*

Doctor: William Robinson

JUL - 2017

S	M	T	W	T	F	Sa
25	26	27	28	29	30	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

Coming Exercise

By the end of the next exercise you should finally choose your topic app and write the first draft of the specification.

Use my Dentistry document to write your own spec.

Start with the database.

There are two types of applications in the Topics document:

•“Reservations”. Similar to my Dentistry app.

•“Blogs/social”. Somewhat simpler, but with fewer similarities with Dentistry.