Course Introduction

Maxim Mozgovoy

Administrative Issues

- Course home page: http://tinyurl.com/seia2019
  (Get new course materials there; PDFs are printer-friendly)

- Instructor:
  Maxim Mozgovoy (mozgovoy@u-aizu.ac.jp)

- Grading scheme:

–Assignments	50%
–Quizzes	20%
–Final examination	30%

Course Goal

The main goal is to form solid understanding of a pragmatic process for engineering Web-based systems and applications.

We will follow a project-based approach: each student will have to design a simple but complete web application with all necessary components.

I will be doing the same job in real time, so just follow my example and pass the course ☺

What is an “Internet Application”

By “internet applications” people usually mean “web applications” or “rich internet applications”.

Web applications are client-server systems, where the client is a conventional web browser.

Traditionally, rich internet applications relied on specialized clients, often implemented as browser plugins (e.g., Adobe Air and JavaFX).

As browsers became more advanced, the trend is to use browser-only capabilities without any plugins (HTML5).

Prerequisites

Knowledge of the following concepts is necessary:

- Concepts of object-oriented programming.
- Python programming
  (we will use only quite basic constructions)
- Web technologies / Web programming
- Basics of JavaScript
  (but we won’t use it much)

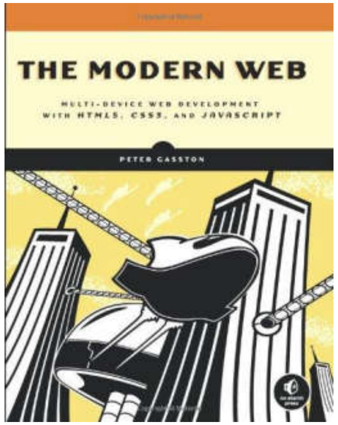
Principal Topics

- Web app architectures
- Introduction to Django framework
- Storing web app data
- Developing main logic of an app
- Developing user interface

Literature

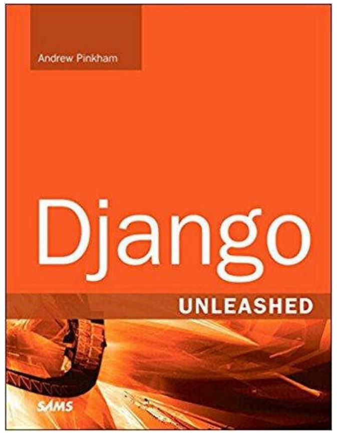
Peter Gasston
The Modern Web
No Starch Press, 2013.

nostarch.com/modernweb


THE MODERN WEB
MULTI-DEVICE WEB DEVELOPMENT WITH HTML5, CSS3, AND JAVASCRIPT
PETER GASSTON

Literature

Andrew Pinkham
Django Unleashed
SAMS, 2015



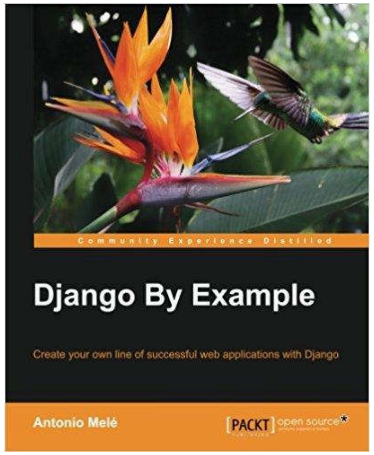
https://www.amazon.com/Django-Unleashed-Andrew-Pinkham/dp/0321985079

Literature

Antonio Mele

Django by Example

Packt, 2015


Community Experience Distilled
Django By Example
Create your own line of successful web applications with Django
Antonio Melé
PACKT open source

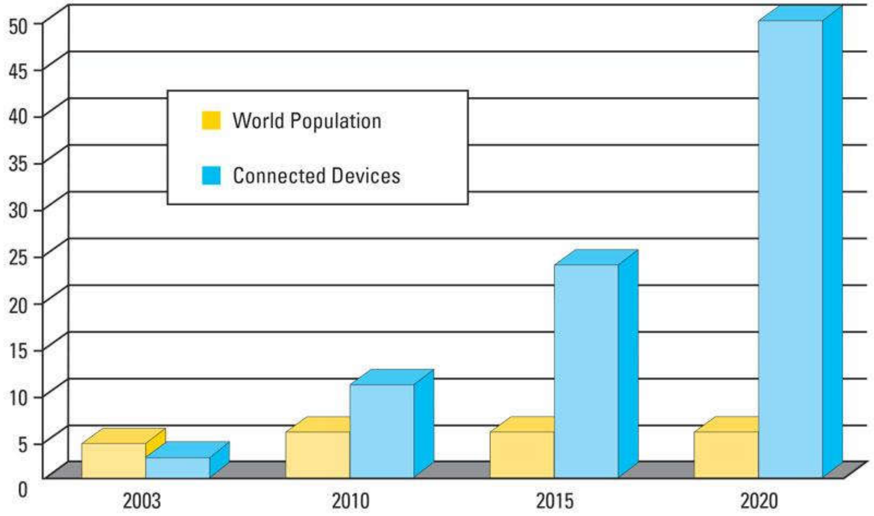
https://www.amazon.com/Django-Example-Antonio-Mele/dp/1784391913/

Motivation

- Many desktop applications migrate to the Web (e.g., Google Docs, Office 365)
- Hi-speed internet access gave birth to new types of Web apps (Dropbox, YouTube)
- Most mobile apps rely on interaction with internet, and mobile market is steadily growing.
- More specialized “smart devices” connect to the internet yearly, leading to the emergence of “internet of things”.

- Newest trend: popular web application frameworks are now used to develop mobile and desktop apps as well!

Internet Connected Devices



From Websites to Web apps

Websites (in their basic form) merely store hypertext documents and serve them to users.

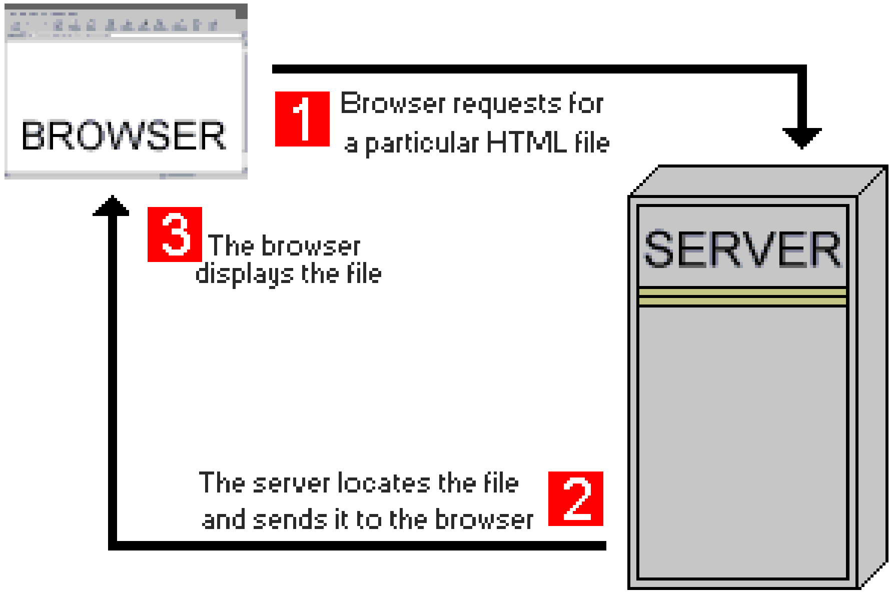
•Hypertext (cross-linked HTML documents): basic format of human-readable information.

•Browser: displays hypertext documents.

•Web server: stores hypertext documents.

•HTTP: protocol for data communication.

Static Websites



BROWSER

1 Browser requests for a particular HTML file

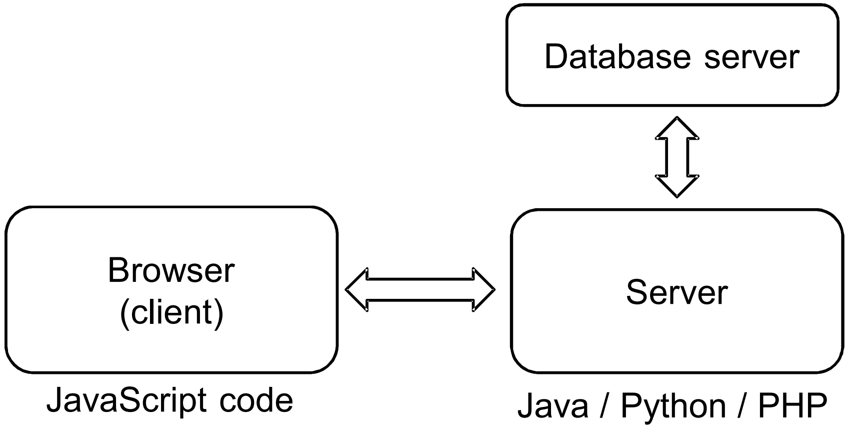
SERVER

2 The server locates the file and sends it to the browser

3 The browser displays the file

Web Apps

Both client and server have logic implemented in some programming language.



Database server

Browser (client)

Server

JavaScript code

Java / Python / PHP

Web Apps Attributes

- Network intensiveness: Every WebApp resides on a network and serves the needs of clients.
- Concurrency: A large number of users may access the WebApp at one time.
- Unpredictable load: The number of users of the WebApp may change unpredictably.
- Performance: If a WebApp users have to wait too long, they may decide to go elsewhere.
- Availability: Users of popular WebApps demand access on a “24/7/365” basis.
- Data driven: The key function of many WebApps is to use hypermedia to present text, graphics, audio, and video content.

Web Apps Attributes

- Content sensitive: The quality and aesthetic nature of content remains an important factor of the quality of a WebApp.
- Continuous evolution: Content of WebApps may be updated on minute-by-minute schedule.
- Immediacy: The strong need to get software to market quickly.
- Security: To protect sensetive content and provide secure data transmission.
- Aesthetics: When an a WebApp is designed to market or sell progucts or ideas or provide services, aesthetics may equal to technical design.

General advices

- Take the time to understand business needs and product objectives, even if the details of the WebApp are vague.
- Describe how users will interact with the WebApp using a scenario-based approach.
- *Always d*evelop a project plan, even if it’s very brief.
- Spend some time modeling what it is that you’re going to build.
- Review the models for consistency and quality.
- Use tools and technology that enable you to construct the system with as many reusable components as possible.
- Don’t reinvent when you can reuse.
- Don’t rely on early users to debug the WebApp—design and use comprehensive tests before releasing the system.

How we will work

- During the next lecture, we will discuss general architecture of web applications and example projects.
- You will have to select one project for implementation.
- We will discuss how to separate the project into modules.
- During lectures, I will explain how to implement the next module and show the real process.
- During exercises, you will try to do the same work for the project of your choice.
- In the end we will evaluate how complete is your work, and this will be the basis for your grade.

Basic Component of Web Application Architectures

- **Client**: Generally a browser (user agent) is controlled by a user to operate the Web application. The client's functionality can be expanded by installing plug-ins and applets.
- **Firewall**: A piece of software regulating the communication between insecure networks (e.g., the Internet) and secure networks (e.g., corporate LANs). This communication is filtered by access rules.
- **Proxy**: A proxy is typically used to temporarily store Web pages in a cache. However, proxies can also assume other functionalities, e.g., adapting the contents for users (customization), or user tracking.

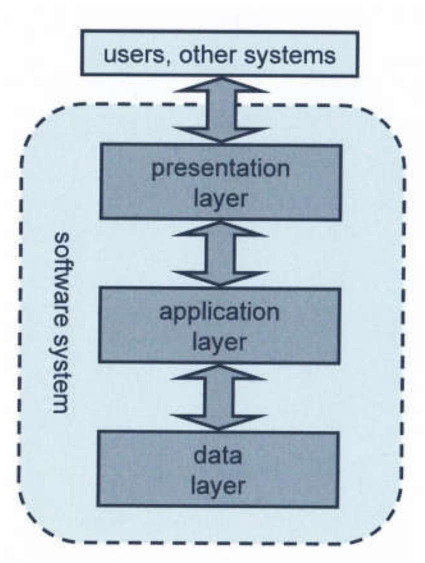
Basic Component of Web Application Architectures

- **Web server**: A Web server is a piece of software that supports various Web protocols like HTTP, and HTTPS, etc., to process client requests.
- **Database server**: This server normally supplies an organization's production data in structured form, e.g., in tables.
- **File/media server**: This component is primarily used for content delivery of non-structured bulk data (e.g., audio or video).
- **Application server**: An application server holds the functionality required by several applications, e.g., workflow or customization.

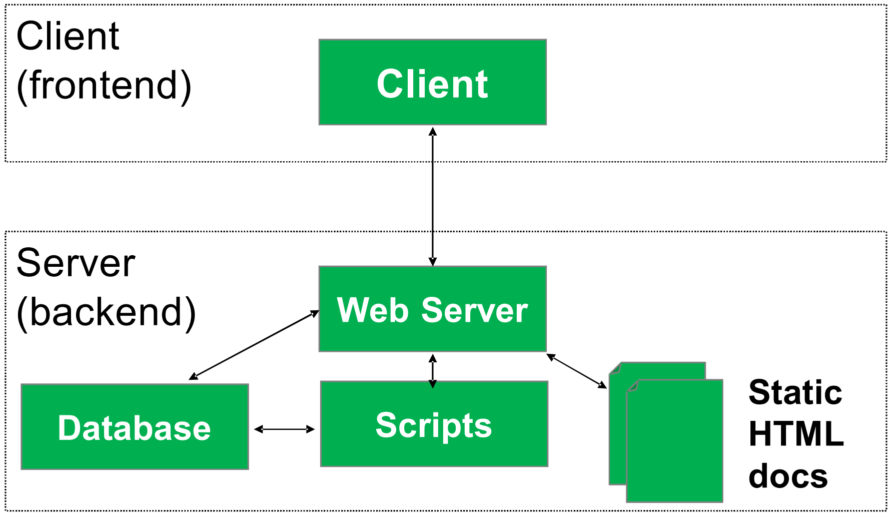
Layers of a Software System

Most of software systems are designed around three layers:

- Presentation layer: encapsulates interactions with users or other systems.
- Application (business) layer: determines what the system actually does. It takes care of business rules and processes.
- Data layer: deals with the organization (storage, indexing, and retrieval) of the data to support the application layer.



Basic Client/Server (2-Layer)



Client (frontend)

Client

Server (backend)

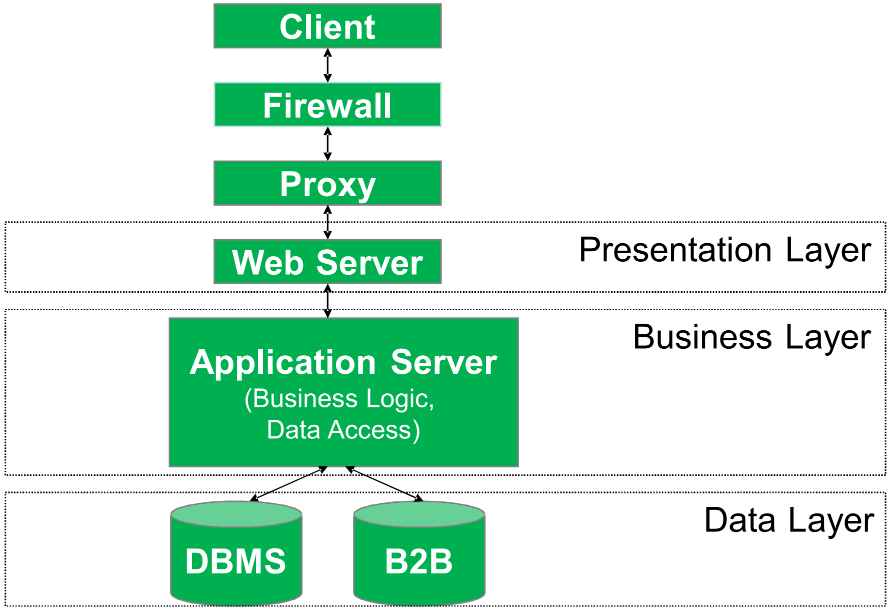
Web Server

Database

Scripts

Static HTML docs

N-Layer Architectures



Client

Firewall

Proxy

Web Server

Presentation Layer

Application Server (Business Logic, Data Access)

Business Layer

DBMS

B2B

Data Layer

Why an N-Layer Architecture?

- Separating services in business layer promotes re-use different applications
  - Loose coupling – changes reduce impact on overall system.
  - More maintainable (in terms of code)
  - More extensible (modular)
- Trade-offs
  - Higher complexity
  - More points of failure

Conclusion

The quality of a Web application is considerably influenced by its underlying architecture.

- Incomplete or missed architectural aspects make it difficult to meet the quality requirements of Web applications.
- Poor performance, insufficient maintainability and expandability, and low availability of a Web application are often caused by inappropriate architecture.

Next Exercise

Study "Dentistry" document to understand how to write a simplified work plan sketch.

Study the topics suggested in the "Topics" document and write your own sketch of the work plan, using "Dentistry" as an example. You can select any existing topic or suggest your own application.

For the rest of the course, you will be working on the application according to your own plan. The application should contain three mandatory layers:

1) client (presentation layer);
2) application layer;
3) database layer.