Dynamic Web Sites

COMP 8347

Slides prepared by Dr. Arunita Jaekel
arunita@uwindsor.ca

Dynamic Web Sites

- Topics
  - Introduction
  - Communication
    - HTTP, URL
  - Presentation
    - HTML/CSS, templates
  - Overall Structure (MVC)
    - First look at Django

Client Server Model

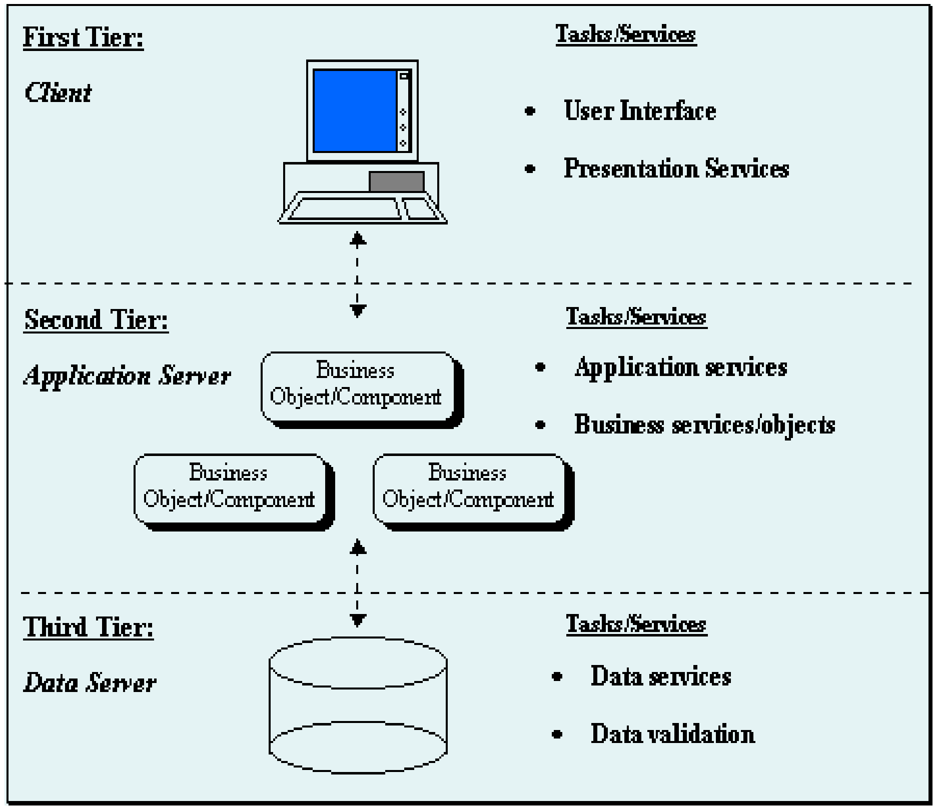


*Fig. taken from [1]

- User requests document from the Web server.
- Web server fetches and/or generates the necessary document.
- The result is returned to the user's browser.
- The browser renders the document.

Static vs. Dynamic Web Pages

- *Static web page*: requests to the same URL always return the same information.
  - Content consists of HTML text files stored on the server.
  - URL typically does not contain parameters; simply a ‘path’
  - Primarily informational
- *Dynamic web page*: Data returned by a given URL can vary significantly.
  - generates content and displays it based on actions the users make on the page
  - Functional and informational

HTTP

- *HTTP*: Hyper-Text Transfer Protocol
  - Encapsulates the process of serving web pages
  - Protocol for client-server communication
  - Most clients/servers use version 1.1; HTTP 2 gaining popularity.
  - ***A network protocol*****:** defines rules and conventions for communication between network devices.
- HTTP is stateless
  - Server maintains no information on past client requests.

HTTP

- Application level protocol
  - Client sends request
  - Server responds with reply
  - Other application level protocols are FTP, SMTP, POP etc.
- Almost always run over TCP
  - Uses ‘well known’ port 80 (443 secure)
  - Other ports can be used as well
  - Can support multiple request-reply exchanges over a single connection

URL

- *URL*: Uniform Resource Locator
- General Format: *<scheme> : //<host> :<port> /<path> ;<parameters> ?<query>*
  - – Scheme: Protocol being used (e.g. http)
  - – Host: host name or IP address
  - – Port: TCP port number used by the server (if not specified, defaults to 80 for http)
  - – Query passes parameters
  - – Example:
    - https://www.google.ca/?gfe_rd=cr&ei=XzYUVceeHayC8QfamoGgDw&gws_rd=ssl#q=http

HTTP Message

- A start line: can be request or status line
  - **GET /hello.htm HTTP/1.1 (e.g. of request line from client)**
  - **HTTP/1.1 200 OK (e.g. of status line from server)**
- Zero or more header fields followed by CRLF
  - Provide information about the request or response, or about the object sent in the message body
  - Format for message-header = field-name ":" [field-value]
  - Examples:
    - » **Host: www.example.com (Host required for requests in Ver 1.1)**
    - » **Server: Apache**
    - » **Content-Length: 51**
- An empty line indicating the end of the header fields
- Optionally a message-body
  - If present, carries the actual data
    - <html> <body> <h1>Hello, World!</h1> </body> </html>

HTTP Methods

- GET: Used to retrieve information from the given server using a given URI.
  - should only retrieve data and should have no other effect on the data.
- POST: Used to send data to the server, e.g. customer info, using HTML forms.
- Other methods: PUT, DELETE, TRACE etc

HTTP Requests

- Request-Line = Method SP Request-URI SP HTTP-Version CRLF
  - Method indicates method to be performed; should always be uppercase.
  - Request-URI identifies the resource on which to apply request [2]

```
GET /hello.htm HTTP/ 1.1
User-Agent: Mozilla/4.0
Host: www.tutorialspoint.com
Accept Language: en-us
Connection: Keep-Alive

POST /cgi-bin/process.cgi HTTP/ 1.1
User-Agent: Mozilla/4.0 compatible; MSIE5.01; Windows NT)
Host: www.tutorialspoint.com
Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
```

HTTP Responses

- Status-Line = HTTP-Version SP Status-Code SP Reason-Phrase CRLF

```
HTTP/1.1 200 OK
Date: Mon, 27 Jul 2009 12:28:53 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Wed, 22 Jul 2009 19:15:56 GMT
Content-Length: 88
Content-Type: text/html
Connection: Closed

<html>
<body>
<h1>Hello, World!</h1>
</body>
</html>
```

Status Codes

- **1xx: Informational:** request received and continuing process.
  - 100 Continue
  - 101 Switching protocols
- **2xx: Success:** action was successfully received, understood, and accepted.
  - 200 OK
- **3xx: Redirection:** further action must be taken in order to complete the request.
  - 301 Moved Permanently
  - 307 Temporary redirect

Status Codes

- **4xx: Client Error:** request contains bad syntax or cannot be fulfilled
  - 400 Bad Request
  - 401 Unauthorized
  - 403 Forbidden
  - 404 Not Found
  - 408 Request Timeout
- **5xx: Server Error**: server failed to fulfill an apparently valid request
  - 500 Internal Server Error
  - 503 Service Unavailable
  - 504 Gateway Timeout
  - 505 HTTP Version Not Supported

What Is HTML?

- HTML is a markup language used to describe webpages.
  - HTML stands for HyperText Markup Language. When a web browser displays a webpage:
    - it is reading and interpreting a HTML document.
  - Used for structuring and presenting content on the World Wide Web.
  - Some related standards include CSS3

Basic Structure

- *DOCTYPE*: Tells browsers *how* to read your document.
  - Forces browsers to use ‘standard mode’.
  - Using standard mode, most browsers will read your document the same way.
- *<head>*: Contains information about your page.
- *<body>*: The actual content of your page.

```
<!DOCTYPE html>
<html>
  <head>
        <title>My first Webpage</title>
  </head>
  <body>
        <h1>This is a Heading</h1>
<p>Hello World!</p>
  </body>
</html>
```

Elements

- HTML elements are marked up using start tags and end tags.
  - Tags are delimited using angle brackets with the tag name in between.
    - End tags include a slash before the tag name.
    - Some elements require only a single tag, e.g. <br>, <img>
  - HTML tag names are case insensitive.
    - Recommended: use lowercase.
  - Most elements contain some content
    - e.g. <p>...</p>
  - Elements may contain attributes
    - Used to set various properties of an element.

Attributes

- Attributes: provide additional information about the specific element
  - Always specified in the opening tag.
  - The pattern for writing attributes: attribute="value".
  - Examples:
    - <a href="http://www.myurl.com">This is tag content</a>
    - <img src="my-pic.jpg" alt="This is a picture ">
    - <div class="example">...</div>.
    - <a href="http://www.myurl.com">This is a link</a>

Links

- *Link*: Some text or image you can click to jump to another document or a specific part of the current document.
  - *<a>*: element for links (internal and external).
  - *href*: A required attribute that specifies the destination address
  - *Link text*: The visible part.
    - Click on link text sends you to the specified address.

```
<a href="http://www.mypage.com">Link text</a>
```

  - You can also use an image as a link.

```
<a href="default.html">
<img src="smiley.gif" alt="HTML tutorial"
            style="width:42px;height:42px;border:0">
</a>
```

HTML Forms

- HTML forms are used to collect user input.
  - The **<form>** tag is used to create an HTML form.
  - HTML forms contain **form elements**.
  - The **<input>** element is the most important **form element**.
    - has many variations, depending on the **type** attribute.
  - *Text* Defines normal text input
    - Default width is 20 characters.
  - *Radio* Defines radio button input (for selecting one of many choices)
  - *Submit* Defines a submit button (for submitting the form)

```
<form action="/url_for_processing/" method="post" >
Username: <input type="text" name=“username"><br>
<input type="radio" name="gender" value="male" >Male<br>
<input type="radio" name="gender" value="female" >Female<br>
  <input type="submit" value="Submit now" >
  </form>
```

- Other elements: Reset button, Textarea, Checkbox. Dropdown list etc

Web Framework

- *Web framework*: a software framework designed to support development of dynamic websites and services.
  - Alleviate overhead with associated activities
- Frameworks standardize the ‘boilerplate’ parts.
  - Provide pre-built components so you can focus on unique parts of your project.
  - Repetitive parts handled by framework.
  - Code you use will be well tested, and have less bugs than what you write from scratch.
  - Enforce good development practices.
  - Security features (login, sessions etc) often better implemented in frameworks.
- Limitations:
  - May restrict you in terms of coding paradigms.
  - Steep learning curve.

Which Framework?

- Many different frameworks are available:
  - ASP.NET using C#, Struts in J2EE, Ruby on Rails, other frameworks using PHP, Perl etc.
- Django is a high-level Python Web framework
  - Encourages rapid development and clean, pragmatic design.
  - Build high-performing, elegant Web applications quickly.
  - Adhere to DRY (Don't Repeat Yourself) principle.

Django Framework

- Web framework for perfectionists with deadlines
  - Main focus
    - Dynamic and database driven websites
  - Automate repetitive stuff
  - Rapid development
  - Follow best practices
  - Free
  - Easy to learn
  - Powerful

- Powerful object-relational mapper (ORM)
  - Data models defined entirely in Python
- Automatic admin interface
  - Eliminates tedious work of creating interfaces to add and update content.
- Elegant URL design
  - Flexible, cruft-free URLs
- Template system
  - powerful, extensible template language to separate design, content and Python code

Sites Using Django

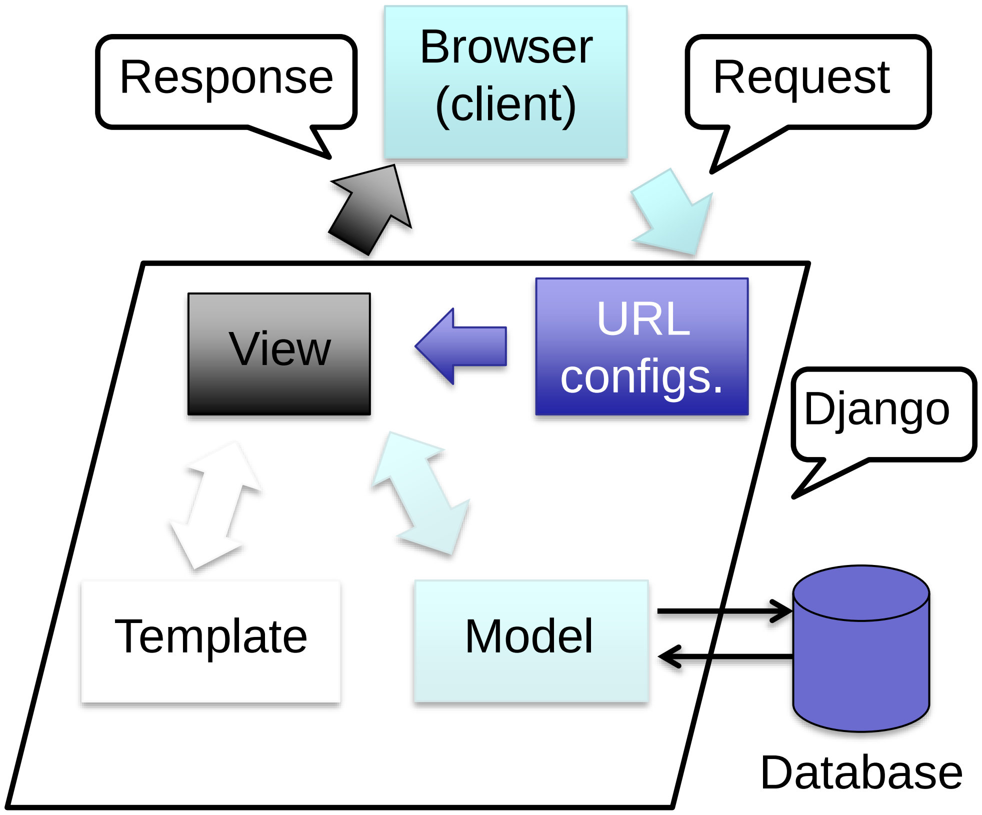
- Disqus
- Instagram
- Mozilla
- NASA
- National Geographic
- OpenStack
- Pinterest

MVC Paradigm

- MVC (Model-View-Controller) paradigm: The application is separated into 3 main layers.
  - Model: Deals with the data
  - View: Defines how to display data
  - Controller: Mediates between the two, allows user to request and manipulate data.
- Allows code reuse
- Increases flexibility
  - E.g. single set of data can be displayed in multiple formats.

Django’s MTV Architecture

- MVC → MTV
- *Model*:
  - Deals with data representation/access.
- *Template*:
  - Describes **how** data is represented.
  - Same as ‘view’ in MVC
- *View*:
  - Describes **which** data is presented.
  - Same as ‘controller’ in MVC.



MTV Architecture

- Model: Represents the data that will be gathered, stored and presented.
  - Changing the models changes underlying database schema. This can have significant side effects.
- Template: Template language used to render the html
  - Simple logic constructs such as loops
  - HTML most common format, but templates can be used to create any text format, e.g. csv
- View: Describes which data you see.
  - Responsible for much (often most) of the logic.
  - Linked to one or more URLs; return a HTTP response object.
  - Django provides useful shortcuts and helper functions for common tasks.
    - Helps in rapid development.
  - For full flexibility you can write your own custom functions.

Project Directory

Create a new Django project:

- outer mysite/
  - container for project; can be renamed.
- manage.py
  - command-line utility to interact with your project.
- inner mysite/
  - actual python package for project
- __init.py__
  - empty file, indicates this dir is a package
- settings.py
  - settings/configuration for the project
- urls.py
  - URL declarations for the project
- wsgi.py
  - entry-point for WSGI-compatible web servers to serve your project

```
mysite/
    manage.py
    mysite/
        __init.py__
        settings.py
        urls.py
        wsgi.py
```

Settings

- Settings.py: Python module with variables for Django settings.
  - update DATABASES ‘default’ item
  - ‘ENGINE’ : 'django.db.backends.sqlite3‘
    - 'django.db.backends.postgresql_psycopg2',
    - ‘django.db.backends.mysql', or
    - 'django.db.backends.oracle‘
- By default, following apps are installed
  - django.contrib.admin – The admin site.
  - django.contrib.auth – An authentication system.
  - django.contrib.contenttypes – A framework for content types.
  - django.contrib.sessions – A session framework.
  - django.contrib.messages – A messaging framework.
  - django.contrib.staticfiles – A framework for managing static files.

Summary

- Dynamic Web
  - Client Server Model
  - HTTP Protocol
  - HTML
    - Forms
- Web Frameworks
  - Django philosophy
    - Don’t Repeat Yourself (DRY)
    - Rapid Development
    - MTV Architecture

References

- [1] http://edn.embarcadero.com/article/10343
- [2] www.tutorialspoint.com/http/
- [3] Python web development with Django by Jeff Forcier, Paul Bissex and Wesley Chun. Addison Wesley 2009.
- [4] https://flatworldbusiness.wordpress.com/flat-education/previously/web-1-0-vs-web-2-0-vs-web-3-0-a-bird-eye-on-the-definition/