Web Frameworks

web development done right

Course of Web Technologies
A.A. 2010/2011

Valerio Maggio, PhD Student
Prof.ssa Anna Corazza
Outline

- Web technologies evolution
- Web frameworks
  - Design Principles
- Case Study
  - Django and GAE (Google App Engine)
- Working Example
Nowadays frameworks has become a buzzword

- Software framework
- Web framework
- Development framework
- ...

So, what do you expect a web framework is?
1. Web Technologies Evolution
Web: Evolution Roadmap

HTML Pages:

Web developers wrote every page “by hand”

Update a web site means editing HTML

“redesign” involved redoing every page
  - One at a time

Solution not scalable
Web: Evolution Roadmap

CGI – Common Gateway Interface

► (+) Pages intended as resources
  ○ Pages are generated dynamically on demand
  ○ Raise of (so called) server side technologies

► (-) Code reuse difficult
  ○ Lot of “boilerplate” code
► (-) High learning curve
What are pros and cons of these two CGI examples?
Web: Evolution Roadmap

PHP *like* solutions

► (+) Learning curve extremely shallow
  ○ Code directly embedded into HTML

► (-) No security and/or protection mechanism provided

► (?) Bunch of HTML, (Business Logic) Code, (Data) SQL code *all together*
<%@ page import="java.util.Hashtable, java.sql.*, javax.naming.*, javax.sql.*" %>
<html><head><title>Database Query in WebLogic</title></head>
<body>
<h2>Querying a database with a JSP</h2>
<String sql = "select * from user">
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    ResultSetMetaData rsm = null;
    <table border='1'><tr>
        <% try{
            conn = pool.getConnection( );
            stmt = conn.createStatement( );
            rs = stmt.executeQuery(sql);
            rsm = rs.getMetaData( );
            int colCount = rsm.getColumnCount( );
            for (int i = 1; i <=colCount; ++i) {
                <th>%=rsm.getColumnName(i)%> </th>
            }
        } %>
    </tr>
    <% while( rs.next( )){ %>
        <tr>
            <% for (int i = 1; i <=colCount; ++i) { %>
                <td> %= rs.getString(i) %></td>
            %>/for %>
        </tr>
    <%} //while
    } catch (Exception e) {
        throw new JspException(e.getMessage( ));
        ....
    %>
</table></body></html>
RIA and “Integrated Solutions”

- **RIA**: Rich Internet Applications
  - **Q**: Do you know what RIA means?
  - **A**: Desktop-like web applications
    - (Ajax and javascript intensive web apps)

- **A.k.a.**: Solutions battery included

- CMS and Web Frameworks
Aim to manage work-flows and contents in a collaborative environment

Designed to simplify the publication of contents to web sites and mobile devices

Examples: Joomla, Drupal, Wordpress, ....
Web frameworks

- **Aim to** alleviate the overhead associated with common Web development
  - Databases, templates, sessions, ...

- **Designed to** support the development of dynamic websites, web applications and web services

- Examples: Struts, Spring, Ruby on Rails, Django, Google App Engine, ...
So, What is a Web Framework?

```python
#!/usr/bin/env python
import MySQLdb

print "Content-Type: text/html\n" 
print "<html><head><title>Books</title></head>"
print "<body>"
print "<h1>Books</h1>"
print "<ul>
connection = MySQLdb.connect(user='me', passwd='letmein', db='my_db')
cursor = connection.cursor()
cursor.execute("SELECT name FROM books ORDER BY pub_date DESC LIMIT 10")
for row in cursor.fetchall():
    print "<li>%s</li>" % row[0]
print "</ul>"
print "</body></html>"
connection.close()
```

- What does this code do?
- What happens when multiple pages need to connect to database?
- Should a developer *really* have to worry about printing the `Content-type`?
- Is this code reusable in multiple environments with different DB connection parameters?
- What happens when a web designer have to redesign the page?
Web Frameworks in a nutshell

- These problems are exactly what a web frameworks tries to solve

- Web frameworks provides a programming infrastructure for applications

- Focus on developing code without having to reinvent the wheel
2. Web Frameworks Design Principles
CGI Architecture Model

Presentation and Visualization

Browser

HTTP request

HTML page

Web Server

exec w/environ

stdout

CGI Program

Business Logic

Database

Data and Models
Task centric architecture (a.k.a. Model 1)
- Difficult reusability and maintenance of code
- Requires different skill-sets

High coupling among:

Presentation (View)
- How to show data

Processing (Controller)
- What information to show

Data Acquisition (Model)
- What information to extract (from DB)
Model 1 Architecture (Java)

- Processing delegates as JSP and Servlets
- Is there any difference between CGI and Servlet?
MVC Architecture Model

- **Model:**
  - Manages domain and data acquisition

- **View:**
  - Manages the visualization of data

- **Controller:**
  - Manages domain and data processing

Q: Do you think this model is feasible to be used on the web as is?
Web-MVC Architecture Model

- **Model:**
  - Manages domain and data acquisition

- **View:**
  - Manages the visualization of data

- **Controller:**
  - Manages domain and data processing

A: No (direct) relationship between the view and the model
Model 2 Architecture (Java)

► **Model:**
  - EJB and Javabeans

► **View:**
  - JSP and JFaces

► **Controller:**
  - Servlets
From Python.org wiki:

[...] frameworks provide support for a number of activities such as interpreting requests, producing responses, storing data persistently, and so on. [...] those frameworks [...] are often known as full-stack frameworks in that they attempt to supply components for each layer in the stack.

So, what are such components?
Web Frameworks Capabilities

► **View**
  - JavaScript Library
  - Template Engine and View Composition
  - Development Server

► **Controller**
  - URL Routing
  - Controller-view Association

► **Model**
  - Database Abstraction
  - ORM (Object Relational Mapping)
Database Access

- Distributed Access
  Logic (JSP, Servlets)

- Centralized Access
  Logic

Q: How easy is modify the db schema?
Active Record pattern

- An object encapsulates both data and behavior

- Put data access logic in the domain object

http://martinfowler.com/eaaCatalog/activeRecord.html
Heavy-weight vs Light-weight Frameworks

Heavy-weight frameworks:
- (Mostly) Java Based
- Based on Model 2 Architecture
- High learning curve
- Bunch of (XML) Configuration Files

Light-weight frameworks:
- Convention over Configuration and DRY Principles
- Shallow learning curve
- Use of Dynamic Languages
  - Python, Ruby, Groovy, Scala
H-W Java frameworks: Struts

```xml
<?xml version="1.0" encoding="ISO-8859-1" ?>
....
<struts-config>
  <form-beans>
    <form-bean name="LoginForm"
      type="com.oreilly.jent.struts.library.ui.LoginActionForm"/>
    <form-bean name="AddBookForm"
      type="org.apache.struts.validator.DynaValidatorForm"/>
    <form-property name="title" type="java.lang.String"/>
    <form-property name="author" type="java.lang.String"/>
    <form-property name="isbn" type="java.lang.String"/>
    <form-property name="addBookAction" type="java.lang.String"/>
  </form-bean>
  </form-beans>
  <global-forwards>
    <forward name="home" path="/home.do"/>
    <forward name="login" path="/login/index.jsp"/>
  </global-forwards>
  <action-mappings>
    <action path="/home" parameter="/home.jsp"
      type="org.apache.struts.actions.ForwardAction" />
    <action path="/login" name="LoginForm"
      scope="request" validate="true" input="/login/index.jsp"
      type="com.oreilly.jent.struts.library.ui.LibraryRequestProcessor">
      <forward name="success" path="/login/success.jsp" redirect="false"/>
      <forward name="failure" path="/login/index.jsp" redirect="false"/>
    </action>
  </action-mappings>
  <controller processorClass="com.oreilly.jent.struts.library.ui.LibraryRequestProcessor"/>
  <message-resources parameter="application"/>
</struts-config>
```
H-W Java frameworks: Hibernate

```xml
<hibernate-mapping>
  <class name="User" table="users">
    <id name="ID" column="id" type="string">
      <generator class="assigned"/>
    </id>
    <property name="password" column="password" type="string"/>
  </class>
</hibernate-mapping>
```

```sql
CREATE TABLE users (  
id VARCHAR(20) NOT NULL,  
password VARCHAR(20),  
PRIMARY KEY(id)  );
```
Design Principles

► Convention over configuration
  ○ “Convention over Configuration is a programming design that favors following a certain set of programming conventions instead of configuring an application framework. [...]”

► DRY (Don't repeat yourself)
  ○ “DRY is a principle that focuses on reducing information duplication by keeping any piece of knowledge in a system in only one place.”
3. Case Study: Django and Google App Engine
Frameworks and Languages

- Rails
- Ruby
- Django
- Google App Engine
- Python
“Speed and flexibility of development are critical. Dynamic languages let you get more done with less lines of code (which means less bugs)”

- Object oriented languages
- Clean and simple syntax
  - Strong Typed
  - Dynamic Typed
Python Programming Language

Is there someone that uses python in professional projects?
- IBM, Google, Sun, HP, Industrial Light and Magic, NASA, Microsoft

Goggle it:
- `site:microsoft.com python`
- You'll get more than 9 thousands results
Programming language Python has become programming language of 2010. This award is given to the programming language that gained most market share in 2010. Python grew 1.81% since January 2010. This is a bit more than runner up Objective-C (+1.63%).

Objective-C was favorite for the title for a long time thanks to the popularity of Apple's iPhone and iPad platforms. However, it lost too much popularity the last couples of months.

Python has become the "de facto" standard in system scripting (being a successor of Perl in this), but it is used for much more different types of application areas nowadays. Python is for instance very popular among web developers, especially in combination with the Django framework.

Since Python is easy to learn, more and more universities are using Python to teach programming languages.

Source: tiobe.com
TIOBE: Programming Languages ranking

<table>
<thead>
<tr>
<th>Year</th>
<th>Winner</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010</td>
<td>Python</td>
</tr>
<tr>
<td>2009</td>
<td>Go</td>
</tr>
<tr>
<td>2008</td>
<td>C</td>
</tr>
<tr>
<td>2007</td>
<td>Python</td>
</tr>
<tr>
<td>2006</td>
<td>Ruby</td>
</tr>
<tr>
<td>2005</td>
<td>Java</td>
</tr>
<tr>
<td>2004</td>
<td>PHP</td>
</tr>
<tr>
<td>2003</td>
<td>C++</td>
</tr>
</tbody>
</table>

### Category

<table>
<thead>
<tr>
<th>Category</th>
<th>Ratings Jan 2011</th>
<th>Delta Jan 2010</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object-Oriented Languages</td>
<td>55.8%</td>
<td>+1.4%</td>
</tr>
<tr>
<td>Procedural Languages</td>
<td>39.2%</td>
<td>-2.1%</td>
</tr>
<tr>
<td>Functional Languages</td>
<td>3.5%</td>
<td>+0.4%</td>
</tr>
<tr>
<td>Logical Languages</td>
<td>1.5%</td>
<td>+0.2%</td>
</tr>
</tbody>
</table>

http://www.tiobe.com
“Duck Typing”

Walks like a duck?

Quacks like a duck?

It's a duck!

def half(n):
    return n/2.0

Q: What is the type of variable n?
Django web framework

Design characteristics

- Model-View-Controller for the Web
- Written in Python
- Explicit instead of implicit
- Loose Coupling
- Don't repeat yourself
Django Architecture Model

- **Django** is based on a slightly different version of MVC
  - a.k.a. **MVT**: Model View Template

- **Model**: Domain Objects
  - Python Classes

- **View**: contains business logic for the pages
  - *Callback* as python functions

- **Templates**: describes the design of the page
  - Template Language HTML based
Django Stack

- Database wrapper (ORM)
- URL dispatcher
- Template system
- Admin Framework
- i18n & l10n
- Authentication
- RSS
- .....

......
Projects and Applications

Projects:
- Composed by different applications
- Glued together by unique configuration file

Applications:
- Set of **portable** functionalities
- Code is more reusable
- *Django Plugables*
  - (djangoplugables.com)
Loose coupling principle between URLs and Views

- Based on regular expressions

```python
from django.conf.urls.defaults import *
from mysite.views import current_datetime, hours_ahead

urlpatterns = patterns('',
    (r'^time/$', current_datetime),
    (r'^time/plus/\d+/$', hours_ahead),
)```
Template Language

- Very restrictive *specific*-language
  - Allows only *presentation operations*
  - No logic and/or processing allowed

- Less Pythonic
  - Oriented to web designers
  - HTML based

- Templates Inheritance Mechanism
  - *Code Reuse*
Template Language (2)

Template Inheritance

- Templates are composed by **Blocks**

```html
base.html

<html>
  <body>
    <h1>
      {% block title %}
      Index page
    {% endblock %}

    {% block content %}
    ...
    {% endblock %}
  </div>
</body>
</html>

index.html

{% extends 'base.html' %}
{% block title %}
Index page
{% endblock %}
{% block content %}
...
{% endblock %}
```
Variables: ```{{ variable_name }}```  

Tags: ```{% template_tag %}```  
  - Board definition: Tags tell the framework to do something

Filters: ```{{ variable|filter }}```  
  - Alters the formatting of variables
Django Admin Framework

- So called *Killer-application*

- Compliant with Active record Pattern
Django Included Apps

- **django.contrib.auth**
  - An authentication system.

- **django.contrib.contenttypes**
  - A framework for content types.

- **django.contrib.sessions**
  - A session framework.

- **django.contrib.sites**
  - A framework for managing multiple sites with one Django installation.

- **django.contrib.messages**
  - A messaging framework.
Google App Engine

► dynamic web serving *(built on top of Django)*
  ○ e.g. supports Django Templating Language
► persistent storage
► automatic scaling and load balancing
► APIs for authenticating
  ○ using Google Accounts
► a fully featured local development environment
► scheduled tasks for triggering events at specified times and regular intervals
References: Google App Engine

http://code.google.com/appengine/
References: Django

- http://www.djangoproject.com/
  - Sito del Progetto
- https://groups.google.com/forum/
  - Google group Italiano di Django

django
References: Python

- [http://www.python.org](http://www.python.org)
  - Sito ufficiale di Python

- [http://www.python.it](http://www.python.it)
  - Sito ufficiale Python Italia

- [https://groups.google.com/forum/#!forum/it.comp.lang.python](https://groups.google.com/forum/#!forum/it.comp.lang.python)
  - Google group Italiano di Python

- [http://forum.python-it.org](http://forum.python-it.org)
  - Forum (~)official Python Italia

- [http://www.pycon.it/](http://www.pycon.it/)
  - Python Italian Conference
  - EuroPython 2011 – Florence, IT – across spring
The definitive guide to Django
A. Holovaty and J.K. Moss, Apress
Sviluppare applicazioni web con Django, Marco Beri, APOGEO
References: Titles

- Python, Marco Beri, APOGEO Serie Pocket
References: Titles

► Programming Google App Engine, D. Sanderson, O'Reilley
And last...

► Want to get some actions?

► Let's do together a working example