Web Frameworks

MMIS 2 VU SS 2011 - 707.025

Denis Helic

KMI, TU Graz

March 10, 2011
Web Application Frameworks

- MVC Frameworks for Web applications
- Struts, Stripes (Java)
- Rails (Ruby)
- Grails (Groovy)
- AJAX, e.g. GWT
Web Services Frameworks (REST)

- REST Frameworks
- Restlet
- Rails plugins e.g.
  http://rubyforge.org/projects/restful-rails/
Struts

- Official project homepage: http://struts.apache.org/
- Java Server-Side Web Framework
- One of the first Web frameworks
- Currently in version 2.x.x
- Follows a MVC design pattern to separate content, logic, presentation
Struts

- Controller is already available
- You need to configure it (a lot of XML configuration files)
- Model is a collection of application specific Java classes
- You need to implement them
- View is a collection of JSP pages
- You need to implement them
- Heavyweight
- A lot to learn before you can start with development
- But still very popular with active community
- Documentation, books are available
Official project homepage:  
http://www.stripesframework.org/display/stripes/Home

Zero external configuration per page/action

Dynamic features of Java to map URLs to actions

You map parts of URL to a Java class

The rest is mapped to a method of that class
Ruby on Rails

- Official project homepage: http://www.rubyonrails.com/
- Ruby is an object-oriented dynamic programming language
- Similar to Smalltalk, Python
Ruby on Rails

- Rails is an MVC Web Framework
- Principles behind Rails
- Less software (a lot of source code is generated - data driven)
- Convention over configuration (e.g., follow naming conventions and no need to configure anything)
Ruby on Rails

- Typical Ruby on Rails work-flow
- Create a database schema
- Follow convention about naming tables and columns
- Invoke a Rails script to generate model/controller classes
- Start Rails server
- Open your browser, access your application, and enjoy!
Ruby on Rails

- Very productive
- E.g., a small Web application with 2 related tables only 50 LOC
- Disadvantage: you need to learn Ruby
- I personally recommend it: you will for sure have fun with Rails!
- Educational purpose: very clean implementation of MVC
- Official project homepage: http://www.grails.org/
- Groovy on Rails
- Groovy is a dynamic scripting language for the JVM - it allows you to use all the Java libraries
- Disadvantage: you need to learn Groovy but if you know Java you are all set up
- I personally recommend it!
You structure your system around resources
You can think of a resource as a record in a database table
You provide different representation of a resource
  e.g. XHTML, XML, JSON
You define a URL pattern for resources
  e.g. /student/id
You have the complete HTTP semantics provided by the framework
- e.g. GET /student/1 retrieves default representation
- PUT /student/1 + representation in the body
- A Java method is called where you need to insert this new student
- In essence it behaves like a controller
- You need to implement data management
- You need to implement user interface (e.g. with AJAX)
- Be careful not to brake addressability through JavaScript call sequences
- RESTLET + HIBERNATE + AJAX is also a recommendation!
Transparent AJAX calls
Browser compatibility
Some of them are Java-based
You write code in Java, JavaScript+XHTML automatically generated
- Prototype: http://www.prototypejs.org/
- Scriptaculous: http://script.aculo.us/
- Dojo: http://dojotoolkit.org/
- jQuery: http://jquery.com/
Google Web Toolkit

- **GWT**: [http://code.google.com/webtoolkit/](http://code.google.com/webtoolkit/)
- Is a JavaScript based MVC toolkit
- You write the code in Java that is then compiled in JavaScript
- View and controller in the browser, model on the server-side