Prezentační vrstva javových webových aplikací. Přístupy - šablony, dekorátory. Velocity, WebMacro, Sitemesh, Freemarker. Nové přístupy - AJAX

Obsah

Prezentační vrstva webových aplikací

Prezentační vrstva webových aplikací

Prezentační vrstva webových aplikací

Sitemesh

Motivace

Sitemesh

- Moderní a používaný J2EE rámec, který byl pro svoji popularitu přenesen i na jiné platformy (PHP, .NET)
- Řeší některé problémy návrhu vzhledu webových aplikací. Řeší problém se stejným HTML kódem, který je uložen ve více stránkách. Příkladem jsou menu, hlavičky, patičky apod.
- Založen na šablonovacím systému, kdy výběr určité šablony zavisí na kontextu aplikace

Sitemesh I.

- Aplikační rámec umožňující dekorování výsledků HTTP požadavků šablonou. Výdledkem požadavku musí být HTML kód.
- Ve skutečnosti filter (javax.Filter), který je aplikován na určitý kontext aplikace
- Zpracovává výsledky požadavků a dekoruje je na základě některých vlastností prostředí a definovaných pravidel
- Dekorátory mohou být vybírány na základě hlavičky USER-LANGUAGE, na základě cookies, parametru v session či na základě uživatelsky definovaného mapování

```
<decorator name="skin-pro-administraci" page="edit.jsp">
   <pattern>/secure/edit/*</pattern>
</decorator>
```

Sitemesh II.

Příklad

Příklad 1. Dekorátor

```
01  <%@ taglib uri="http://www.opensymphony.com/sitemesh/decorator" prefix="decora
02  <html>
03  <head>
04
05    <decorator:head />
06
07    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
08  </head>
09  <body>
10    <div id="zahlavi">záhlaví</div>
11    <div id="menu">menu</div>
12
13    <decorator:body />
14
15    <div id="zapati">zápatí</div>
16  </body>
17 </html>
```

Příklad 2. Stránka

```
01 <html>
02  <head>
03     <title>Sitemesh</title>
04  </head>
05  <body>
06    <div id="content">
07      <p>Odstavec 1</p>
08      <p>Odstavec 2</p>
09    </div>
10  </body>
11 </html>
```

Příklad 3. HTML odezva na požadavek

```
01  <html>
```

```
02  <head>
03
04    <title>Sitemesh</title>
05
06    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
07  </head>
08  <body>
09    <div id="zahlavi">záhlaví</div>
10    <div id="menu">menu</div>
11
12    <p>Odstavec 1</p>
13    <p>Odstavec 2</p>
14
15    <div id="zapati">zápatí</div>
16  </body>
17 </html>
```

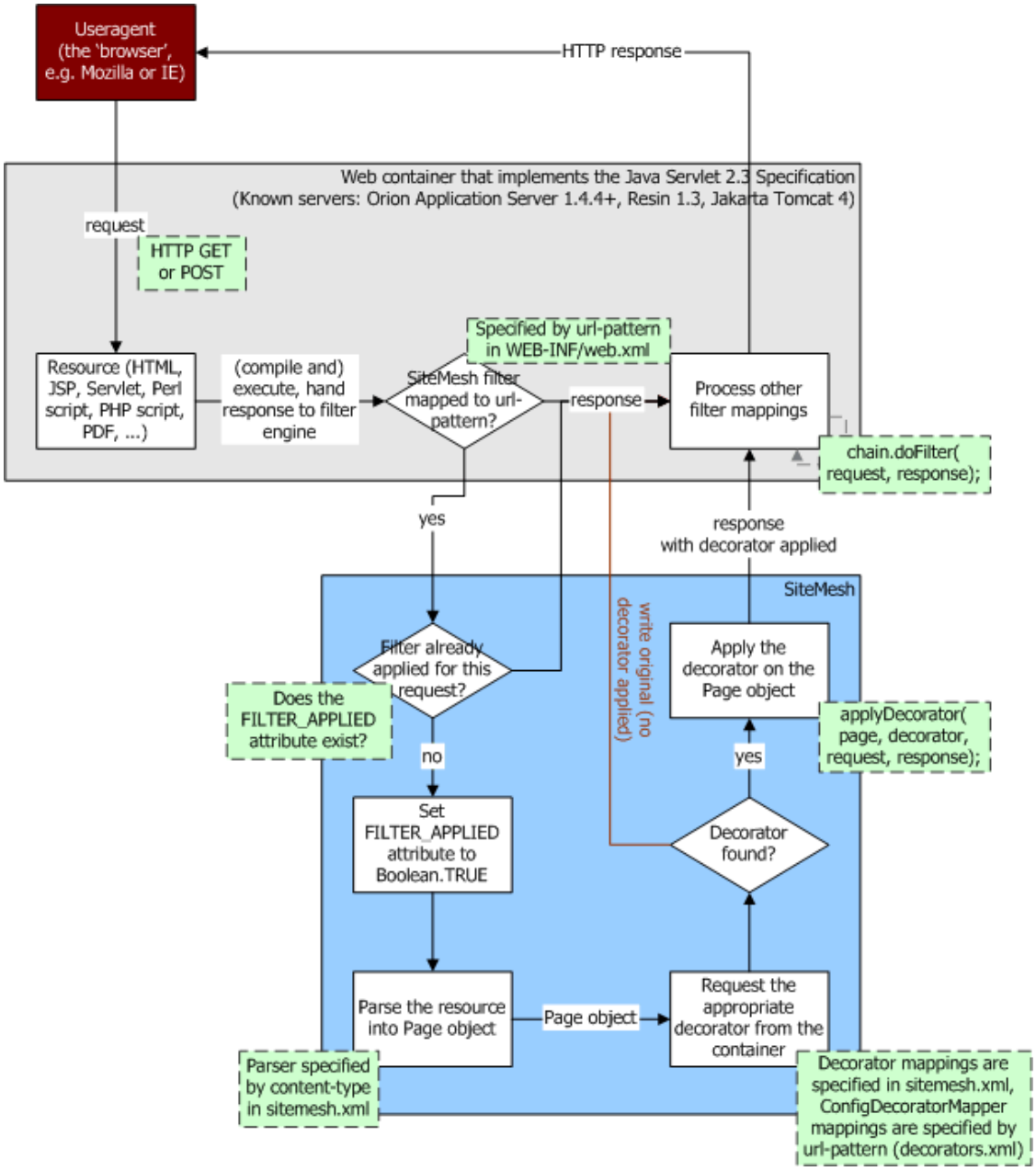
Sitemesh III.

Dekorátor

- Je šablona pro odezvu. Lze říci, že se jedná o skin
- Může být psán pomocí jazyků JSP, Velocity či Freemarker
- Specialní značky <decorator:body /> a <decorator:head />, apod. jsou nahrazeny odpovídajícími částmi požadované html stránky
- V aplikaci může být více dekorátorů.

Sitemesh IV.

Sitemesh webflow



Sitemesh V.

Výhody

- Lze jednoduše zaintegrovat do již hotových projektů (narozdíl od Tiles)
- Snadná instalace
- Možnost napsat si vlastní parser pro jiné odezvy než HTML
- Snadná správa a aplikace dekorátorů

Nevýhody

- V současné verzi 2.2.1 nelze použít v aplikacích navržených podle modelu MVC

Sitemesh VI.

Zdroje:

Weblog na Javaweb.biz [http://javaweb.biz/blog/sitemesh]

Sitemesh homepage [http://www.opensymphony.com/sitemesh/index.html]

Freemarker

Motivace

Freemarker

- Stejně jako JSP a známého Velocity je i Freemarker šablonovací jazyk
- Narozdíl od JSP nejsou šablony Freemarkeru kompilovány. Běh aplikací je tedy relativně rychlejší.
- Nutí oddělit prezentační vrstvu od aplikační vrstvy. (Neexistují v něm jazykové konstrukce typu JSP skripletů, i když možnost pracovat přímo v Java kódu tu existuje)
- Ve FTL (Freemarker template) lze využít Expresion language pro přístup k vlastnostem objektů uložených v modelu

Freemarker I.

- Je nástroj pro generování textového výstupu (nemusí být pouze HTML) na základě definovaných šablon
- Nicméně, Freemarker byl navržen pro generování HTML stránek v J2EE aplikacích navržených podle modelu MVC
- Striktně odděluje aplikační logiku od prezentační vrstvy
- Freemarker není aplikační rámec. Je navržen pouze pro zobrazení dat v modelu MVC
- Umí nativně pracovat s XML dokumenty, které reprezentují model
- Jeho velká obliba zapříčinila portaci do prostředí desktopového ve formě Freemarker Preprocesoru

Freemarker II.

Příklad

Příklad 4. FTL šablona

```
01  <html>
02  <head>
03    <title>Freemarker</title>
04    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
05  </head>
06  <body>
07     Ahoj, mé jméno je ${user.jmeno} ${user.prijmeni}
08  </body>
09 </html>
```

${user.jmeno} a ${user.prijmeni} jsou značky, které umožnují přístup k vlastnostem instance třídy User, který programátor uložil do modelu.

V šabloně lze využít FTL značky pro cykly, podmínky, existenci vlastností apod. Je možno přistoupit k jednotlivým prvkům v polích či přístup k metodám objektů uložených v modelu.

Pro často používané konstrukce lze využít parametrizovatelných maker

Freemarker III.

Příklad

Příklad 5. Model

```
01  (root)
02  + -- user
03    + -- jmeno
04    + -- prijmeni
05  + -- jinyObjekt
```

Do modelu lze vložit Java objekty, kolekce objektů, skalární proměnné, metody objektů apod.

Freemarker IV.

Jak to funguje

Pokud máme nadefinovanou šablonu a připravená data, která umístíme do modelu, musíme vytvořit jednotku Freemarkeru která se postará o zpracování šablony a připravených dat.

```
import freemarker.template.*;
import java.util.*;
import java.io.*;

public class Test {

  public static void main(String[] args) throws Exception {

    Configuration cfg = new Configuration();
          //zdroj sablon
    cfg.setDirectoryForTemplateLoading(new File("cesta/k/adresari/se/sablonami/"))
   //nastaveni wrapperu pro data pridavana do modelu
   cfg.setObjectWrapper(new DefaultObjectWrapper());
   //nacteni sablony
  Template temp = cfg.getTemplate("test.ftl");

   /* Vytvoreni modelu */
   Map root = new HashMap();
   root.put("user", "Big Joe");
   Map latest = new HashMap();
   root.put("latestProduct", latest);
   latest.put("url", "products/greenmouse.html");
   latest.put("name", "green mouse");

   /* Zpracování dat v modelu proti sablone */
   temp.process(root, out);

   //vytisteni dat
  Writer out = new OutputStreamWriter(System.out);
  out.flush();

  }

}
```

Freemarker V.

Freemarker vs. Velocity

Freemarker má v současnosti několik vlastností navíc oproti Velocity 1.2. Zde jsou některé vyjmenované.

- Internacionalizace. Formátování čísel, data a času podle definovaných vzorů.
- Práce s čísly a datem. Porovnávání čísel a dat, apod.
- Cykly. Lze přerušit cyklus, testovat zda-li je poslední prvek cyklu
- Lze získávat data z polí na základě indexu
- Podpora vlastních maker
- a další .... [http://freemarker.sourceforge.net/fmVsVel.html]

Freemarker VI.

Zdroje:

Freemarker Homepage [http://freemarker.sourceforge.net/]

AJAX - Asynchronous JavaScript and XML

Webové aplikace - důvody pro

Rozšíření webových aplikací na úkor desktopových má své důvody:

- Snadnější údržba (opravy či aktualizace verzí pocítíme jako uživatelé, ne jako správci).
- Většinou snadnější vývoj díky platformové nezávislosti kombinací HTTP, HTML, grafických formátů (a příp. JavaScriptu).
- Vše dobře ladí s centralizací dat a jejich zpracování na serveru.
- I starosti se zabezpečením (safety & security) jsou centralizovány.

Webové aplikace - důvody proti

Tradiční webové aplikace trpěly typickými nedostatky oproti klasickým desktopovým aplikacím:

- komunikace klient-server byla vždy inicována ze strany klienta a to většinou akcí uživatele - klikem na odkaz, stiknem tlačítka...

- většina operací (aplikační logiky) musela být na serveru (s drobnými výjimkami - validace vstupu, drobné změny v GUI...)
- zvyšovalo to zátěž sítě a hlavně ztěžovalo práci
- jsou starosti s nekompatibilitou interpretace HTML a zejména skriptů (i JavaScriptu) na různých platformách i prohlížečích.

Přístup AJAX - Asynchronous JavaScript and XML

Koncepce AJAX [http://en.wikipedia.org/wiki/AJAX] je vybudována na principech webových aplikací, které generují a používají na straně klienta (tj. v prohlížeči) speciální javascriptové techniky, jež:

- asynchronně komunikují se stranou serveru a
- vyměňují si XML data
- na straně klienta je skriptem modifikován DOM model dokumentu

Výhody

- Uživatel *nemusí stále klikat*, aby získal odezvu na své akce: výběry ze seznamů, vpisování do textových polí, pohyby po obrázcích (např. mapách) atd.
- Není nutné při sebemenší aktualizaci generovat na straně serveru a následně načítat a zobrazovat celou stránku.

Kde je použito?

Nejznámnějšími místy intenzivně využívajícími principy AJAX jsou služby Google:

- Google Suggest [en.wikipedia.org/wiki/Google_Suggest] (FAQ [http://labs.google.com/suggestfaq.html])- při vyhledávání jsou uživateli nabízeny tipy (klíčová slova), na co se může ptát
- Google Gmail [http://mail.google.com] (popis [http://www.google.com/search?hl=cs&hs=I2r&lr=&client=firefox-a&rls=org.mozilla:cs-CZ:official&oi=defmore&q=define:Gmail])- jsou nabízeny adresy příjemců zpráv, automaticky je aktualizován obsah mailboxu...
- Google Maps [http://www.google.com/url?sa=X&oi=def&q=http://en.wikipedia.org/wiki/Google_Maps] - při pohybu v mapě se automaticky objevují nové části mapy.

To ale nejsou aplikace jediné, mezi další patří i české:

- slovnik.seznam.cz [http://slovnik.seznam.cz] - při vkládání slova pro překlad se objevují tipy, co zadat.
- Manuel Kiessling's open-source ARSC [http://manuel.kiessling.net/projects/software/arsc/] (A Really Simple Chat) - chat založený na HTTP komunikaci
- KnowNow's SpeedReader [http://www.google.com/url?sa=t&ct=res&cd=1&url=http%3A//rss.knownow.com/&ei=k-gJQ7naDqDAQoDZhMsO]

Jak pracuje?

AJAX není jednou technologií, ale spočívá v použití kombinace (známých a zavedených) technologií a standardů:

- HTML / XHTML a CSS pro prezentaci informací
- JavaScript pro modifikaci Document Object Modelu (DOM) na klientovi
- XMLHttpRequest [http://en.wikipedia.org/wiki/XMLHttpRequest] pro (asynchronní) výměnu dat se serverem.

AJAX ve spojení s J2EE

Server dev.java.net [http://dev.java.net] nabízí v rámci Java BluePrints Solutions Catalog sekci:

- Asynchronous JavaScript and XML (AJAX) with Java 2 Enterprise Edition [https://bpcatalog.dev.java.net/nonav/ajax/index.html]