X33EJA – Enterprise Java

Petr Šlechta
Sun Microsystems
petr.slechta@sun.com

Servlety

Cíl: Vytvořit jednoduchou web aplikaci

- Tenký klient (webový prohlížeč)
- AS: pouze web kontejner
  - Servlety
  - JSP stránky
- Bez business logiky a persistence
- Viz též
  - JSF
  - Web Frameworks

Dokumentace

- Java EE 5 API
  - http://java.sun.com/javaee/5/docs/api/
- NetBeans již tuto dokumentaci obsahují
  - http://www.netbeans.org/

Servlet

- Objekt spravovaný (managed) web kontejnerem
- Obecný servlet (javax.servlet.*)
- HTTP servlet (javax.servlet.http.*)
- Odstíněn od některých detailů komunikace (listenery, cache) a HTTP protokolu (bezestavový)
- Na základě vstupních informací (HttpServletRequest) generuje výstupní informace (HttpServletResponse, HTML stránku)

Mapování servletů

- Konfigurační data pro servlety (a ostatní komponenty web aplikace) jsou uloženy v souboru web.xml (descriptor)
- Mapování servletů na URL
  - Aplikace sama má kontext
    - http://localhost:8080/**HelloWorld**
  - V rámci web aplikace je nutno určit, které dotazy budou zpracovány daným servletem
    - http://localhost:8080/HelloWorld/**SayHello**

WAR soubor

- Celá web aplikace je zabalena do WAR (Web Archive) souboru
- WAR = JAR (Java ARchive) s určitou strukturou (některé položky jsou stejné jako v JAR souboru)
  - JAR = ZIP s určitou struktorou
- WAR může obsahovat Java kód, JSP, HTML, resources, konfigurační informace (descriptors), ...

Ukázka – vstup a výstup

- Servlet provede
  - Zpracování vstupních parametrů
  - Zápis do logu
  - Generování výstupní stránky
- Ukázka ladění na úrovni HTTP protokolu

Lifecycle servletu

- AS má obvykle více threadů zpracovávajících požadavky klientů
  - Viz konfigurace AS
- Lifecycle: init(), service(), service(), ..., service(), destroy()
- Kód servletu není synchronizován (!)
  - Metoda service() může být prováděna několika thready najednou
  - Nutno psát reentrantní kód (pouze s lokálními proměnnými) a synchronizovat všechny přístupy ke sdíleným zdrojům

Uchování kontextu

- HTTP je bezestavový protokol
- Kontext lze uchovat na klientu či na serveru
  - Klient: cookies
  - Server: sessions
- Různé rozsahy (scopes)
  - Application (javax.servlet.ServletContext)
  - Session (javax.servlet.http.HttpSession)
  - Request (javax.servlet.ServletRequest)
  - Page (javax.servlet.jsp.PageContext)

Ukázka – Cookies

- Využití API
- HTTP ladění

Konfigurace servletu

- web.xml může obsahovat konfigurační data pro web aplikaci (app scope)
  - Java EE web aplikace by neměly být závislé na externích zdrojích (soubory, atd.)
  - Měly by být nezávislé na OS a AS

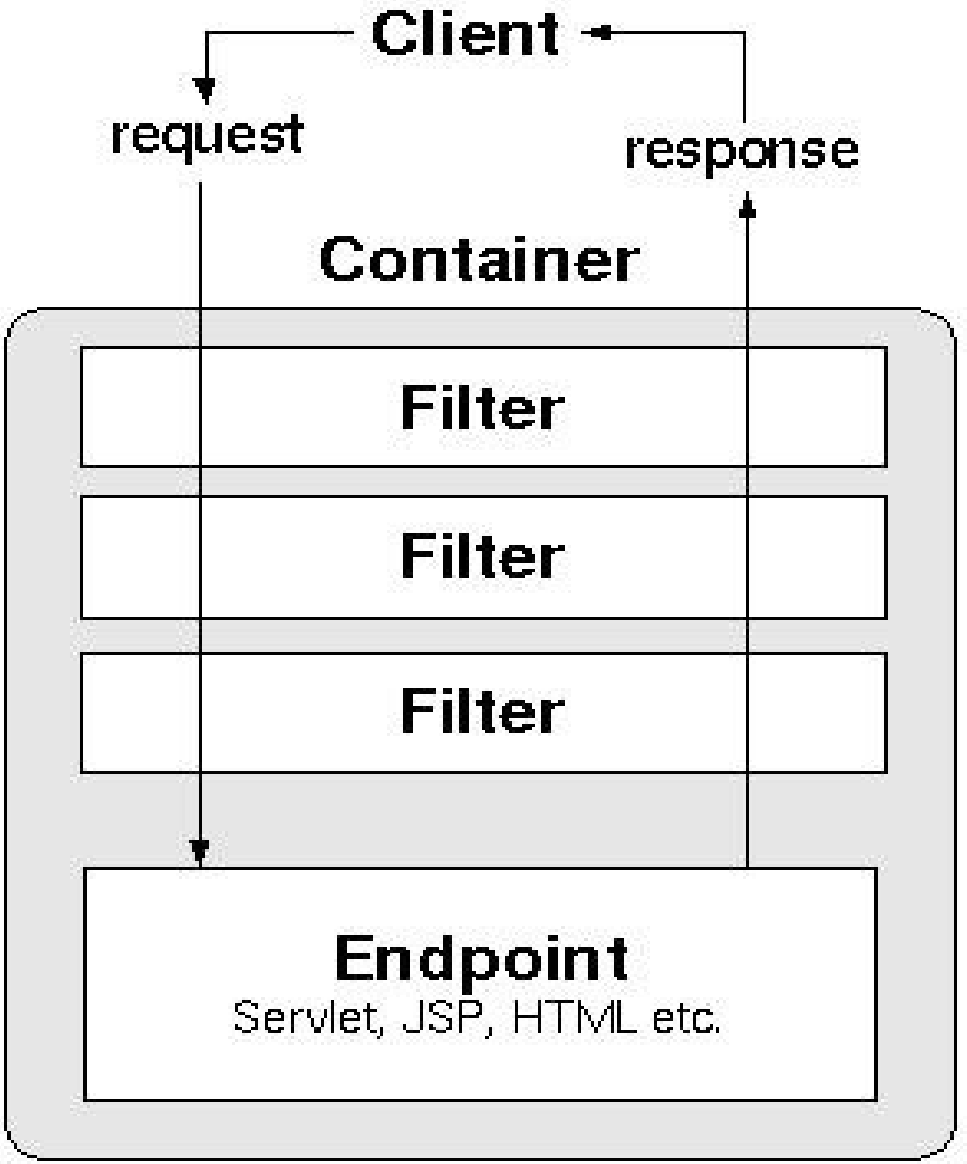
Ukázka

- Zpřístupnění souborů přes web
  - Různí uživatelé, různé sessions
  - Volba jak zakódovat parametry do URL (REST styl vs. klasický styl)
  - Konfigurace aplikace

Filtry (filters)

- Umožňují vstoupit mezi klienta a servlet a změnit data
  - Komprese, kódování obrázků, ...
  - Pozměňování výstupu servletu (logo, ...)
  - Bezpečnost (odmítnutí přístupu)
  - Integrace web aplikací (SSO)
  - ...
- Konfigurace filtru pomocí web.xml (záleží na pořadí položek)
- Řetězení filtrů do filtrovacího řetězce (filter chain)

Řetězení filtrů



Implementace filtru

- doFilter(...) metoda
- Návrat z metody = prázdná odpověď
- Lze přesměrovat klienta na jiný zdroj (redirect)
- Pomocí “zabalení” (wrapping) request a response objektů lze dosáhnout pozměnění vstupů či výstupů pro další filtr v řetězci (HttpServletRequestWrapper a HttpServletResponseWrapper)
- Ukázka: kontrola přístupu k informacím podle přihlášení uživatele

Přesměrování a vložení zdroje

- HttpServletResponse.sendRedirect(String location)
- Pomocí RequestDispatcher (metody forward a include):

```
RequestDispatcher dispatcher =
   session.getServletContext().getRequestDispatcher("/banner");

if (dispatcher != null)
   dispatcher.include(request, response);
```

Listeners

- Umožňují reagovat na změnu v lifecycle servletu či session
  - ServletContextListener
  - ServletContextAttributeListener
  - ServletRequestListener
  - ServletRequestAttributeListener
  - HttpSessionListener
  - HttpSessionActivationListener
  - HttpSessionAttributeListener
  - HttpSessionBindingListener

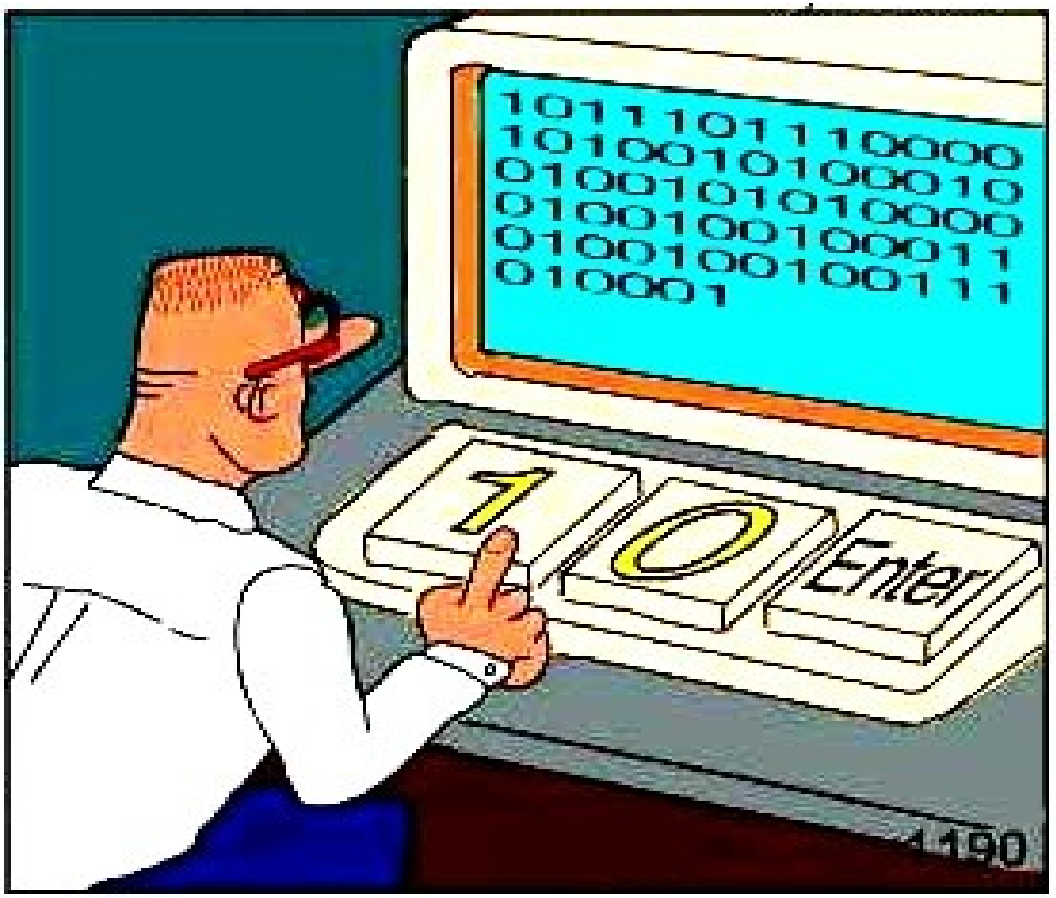
Ošetření chyb

- Web kontejner generuje standardní stránku v případě chyby
- Lze definovat chybovou stránku pro danou vyjímku
  - Ve web.xml, tag <error-page>

SingleThreadModel

- SingleThreadModel Interface pro servlety se stavem
  - Třída je deprecated, takže raději nepoužívat
  - Lépe je použít synchronize blok
  - SingleThreadModel může způsobovat potíže s výkoností AS

Otázky?



REAL Programmers code in BINARY.