Struts2 keretrendszer

Áttekintés

Áttekintés

- Bevezetés
- Struts2 keretrendszer
- Action osztály
- Interceptor-ok
- OGNL

Áttekintés

- Bevezetés
- Struts2 keretrendszer
- Action osztály
- Interceptor-ok
- OGNL

Áttekintés

- Bevezetés
- Struts2 keretrendszer
- Action osztály
- Interceptor-ok
- OGNL

Áttekintés

- Bevezetés
- Struts2 keretrendszer
- Action osztály
- Interceptor-ok
- OGNL

- Már a megjelenésük kezdetén bebizonyosodott a **Servletek** hasznos volta.
- A CGI-vel szemben gyorsabbak voltak, hatékonyabbak, hordozhatók és bővíthetők.
- A HTML kód beágyazása `println()` metódusokon keresztül fárasztó volt és problematikus.
- A választ erre a **JSP** adta meg, mely a nézet-generálást jelentős módon megkönnyítette.
  A fejlesztők simán keverhették a HTML kódot Java kóddal, megtartva a Servlet összes előnyét.
- A java alapú Web-alkalmazások először JSP-központúak lettek, azaz keveset tettek a vezérlés megoldására. Más modellre volt szükség.

- Rájöttek, hogy a JSP-k és Serveltek együtt jól használhatók a Web-alkalmazásokban: a Serveletek gondoskodnak a vezérlésről a JSP-k pedig a megjelenítésről.
- Ezt a modellt nevezték el `Model2`-nek (A JSP-k kizárólagos alkalmazása volt a `Model1`).
- Ez a `Model2` nagyon hasonlít a klasszikus **MVC** modellhez (*Model-View-Controller*), és ma már ugyanarra a modellre mindkét nevet használják.

A **Struts2** keretrendszer megvalósítja az MVC elvet.

Keretrendszerek

két véglet:

- semmiféle rendszer – káosz
- túl merev struktúra – nem enged sok szabadságot a programozónak

egy jó keretrendszer jellemzője:

- egy bizonyos struktúra betartására kötelez anélkül, hogy túlzottan korlátozná a programozó lehetőségeit

- a Web-es keretrendszerek bizonyos alaposztályok használatára ösztönöznek, elemkönvtárakat kínálnak fel

Struts2:

- a Struts és a WebWork ötvözete
- http://struts.apache.org/2.x/

Az MVC elvet illetően a Struts2

- a vezérlés réteget valósítja meg
- a megjelenítés réteget saját elemkönyvtárakkal segíti.
- a modell réteg megvalósítása a Struts2 szempontjából lényegtelen.

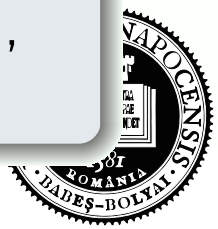
Struts2 vezérlés:

A Struts2 több komponenst biztosít a kontroller réteg megvalósítására:

- egy kontroller Servletet (konfigurációs állomány alapján bekonfigurált)
- a fejlesztő által megírt kérés kezelőket (Action osztályok)
- interceptor-ok (előre definiált vagy a fejlesztő által megírt)

Struts2 nézet:

- A Struts2-s elemkönyvtár közvetlen módon támogatja a nézet réteget (tipikusan JSP).
- Természetesen más elemkönyvtárakat (pl. JSTL) is használhatunk a Struts2-vel.
- A JSP-n kívül más nézet-technológiák is használhatók Struts2-vel, (pl. `FreeMarker`, `Velocity`, `JasperReports`, `XSLT`, stb.).



Modell:

- A modell réteg mindig projekt-specifikus.

A modell lehet:

- az `Action` osztályokban megvalósított logika (nem ajánlott)
- külön üzleti logika réteg:
  - üzleti logika osztályok
  - EJB-k (bonyolultabb üzleti logika esetén)

Hogyan működik mindez együtt:

- Inicializáláskor a kontroller Servlet feldolgozza (parse) a konfigurációs fájlt (`struts.xml`),
- Ennek alapján tudni fogja, hogy melyik URL esetén melyik `Action` osztályhoz irányítson.
- Még mielőtt meghívódik az Action osztály megfelelő metódusa (tipikusan `execute()`), meghívódnak az interceptor-veremben lévő interceptor-ok a megfelelő sorrendben
- Az `Action` különböző ellenőrzések, illetve az üzleti logika meghívásának eredményeképpen a megfelelő nézethez továbbít.
- Az `Action` a hibakezelésre és a vezérlésátadásra koncentrál. Nem implementál üzleti logikát, csak meghívja azt.
- A felhasználó által bevitt adatok az Action osztály mezőit (ezek lehetnek pl. JavaBean-ek) állítják be.
- Ellenőrzési hiba esetén a Struts2-nek van egy mechanizmusa a megfelelő hibagenerálásra ill. annak a megmutatására a JSP-ben.

web.xml konfigurálása

```
<web-app ...>
  <filter>
    <filter-name>struts</filter-name>
    <filter-class>org.apache.struts2.dispatcher.FilterDispatcher</filter-
class>
  </filter>
  <filter-mapping>
    <filter-name>struts</filter-name>
    <url-pattern>/ * </url-pattern>
  </filter-mapping>
...
</web-app>
```

struts.xml

örökölheti a `struts-default.xml`-ben előre definiált elemeket:

- result-típusok
- interceptorok
- előre definiált interceptor-vermek

definiálhatók:

- új (nézetre vonatkozó) result-típusok
- új interceptorok
- interceptor-vermek
- **action-ok**

struts.xml

Pl.

```
<?xml version="1.0" encoding="UTF-8" ?>
...
<struts>
  <package name="struts" namespace="/"
extends="struts-default">
    <action name="updateStudent"
class="edu.prg.StudentAction">
      <result name="success" type="redirectAction">
        studentList
      </result>
      <result name="input">student.jsp</result>
      <interceptor-ref name="paramsPrepareParamsStack"/>
    </action>
  </package>
</struts>
```

Az Action-t implementáló osztály

követelmény:

- a Struts2 action osztályai a `com.opensymphony.xwork2.Action` interfészt kell implementálják.
- az alapértelmezés szerint meghívott metódus fejléce:
  `public String execute() throws Exception`
  (bármilyen más nevű, de hasonló fejlécű metódus is használható)

segédosztály:

- az `Action` implementálása helyett tipikusan az `ActionSupport` osztályt bővítjük ki.

ActionSupport osztály

az `ActionSupport` az alábbi interfészeket implementálja:

- Validatable – egy validate() metódust bocsát rendelkezésre, mely a meghívódik, ha a *workflow* interceptor része az interceptor-veremnek
- ValidationAware – hibakezelést szolgáló metódusokat bocsát rendelkezésre (pl. addFieldError(), addActionError())
- TextProvider – a nyelvi beállításnak megfelelő szöveg erőforrásfájlokból való kinyerését segíti elő `getText()` metódusokkal
- LocaleProvider – `getLocale()` metódus a locale lekérdezésére

- az action osztály további opcionális interfészeket implementálhat, melyek különböző interceptorokkal működnek együtt. Pl.
  - Preparable – a prepare interceptorral működik együtt;
  - SessionAware, RequestAware a servletConfig interceptorral működnek együtt

Modell objektumok

A felhasználó által bevitt adatok beállítják az Action osztály mezőit (melyek lehetnek pl. JavaBean-ek):

- Amennyiben params interceptor része az interceptor-veremnek, meg fognak hívódni az action osztály megfelelő set/get metódusai

Pl. – tekintsük az alábbi form-ot

```
<s:form>
  <s:textfield name="user.firstName" label="Firstname" />
  <s:textfield name="user.lastName" label="Lastname" />
  <s:submit value="Submit" action="setUser" />
</s:form>
```

submit-ra meghívódnak az action osztály alábbi metódusai:

- getUser().setFirstName(...);
- getUser().setLastName(...);

az action osztály:

```
public class SetUserAction extends ActionSupport{
  private UserBean user;
...
  public UserBean getUser();{
    return user;
  }
...
  public void setUser(UserBean user);{
     this.user=user;
  }
}
```

- automatikus típuskonverzió történik, ahol szükséges

Hozzáférés a szesszióhoz, illetve más Servlet-specifikus adatokhoz

hozzáférés a szesszió attribútumokhoz `ActionContext` osztályon keresztül:

```
Map session = ActionContext.getContext().getSession();
session.put("user", user);
```

kérés objektum lekérése

```
ActionContext ctx= ActionContext.getContext();
HttpServletRequest req =
  ctx.get(ServletActionContext.HTTP_REQUEST);
HttpSession session = req.getSession();
```

- ajánlottabb ehelyett a `SessionAware` illetve `ServletRequestAware` interfészek kiterjesztése

hozzáférés a szesszió attribútumokhoz a `SessionAware` interfész segítségével:

```
public class SetUserAction extends ActionSupport implements
SessionAware {
  Map session;
...
  public void setSession(Map session){
    this.session=session;
  }
  public String execute() throws Exception {
    //uzleti logika meghívás
    if(hiba){
      return INPUT;
  }else
    session.put("user",user);
    return SUCCESS;
  }
}
```

Interceptorok

Az interceptorok

- a Servlet-nél használt szűrőkhöz (filter) hasonlíthatóak – ugyanaz a logika szerint hívódnak meg
- elő-/utófeldolgozást végeznek az action osztály meghívása előtt/után
- tipikus feladatok: pl. adatellenőrzés, paraméterek beállítása, stb.

előre definiált interceptorok:

- *logger* – naplózást végez az action lefutása előtt és után
- *params* – a kérés paraméterek alapján beállítja az action osztály tulajdonságait (megfeleltetés a név alapján, az értékek a megfelelő típusra lesznek konvertálva) – set/get metódushívások
- *conversionError* – egy field error üzenetet ad az action osztályhoz minden egyes sikertelen típuskonverzió esetén

előre definiált interceptorok (folyt.):

- *servletConfig* – beállítja az action osztályban a `HttpServletRequest`, `HttpServletResponse`, parameter map, session map illetve application map-et, amennyiben az implementálja a `ServletRequestAware`, `ServletResponseAware`, `ParameterAware`, `SessionAware`, illetve `ApplicationAware` interfészeket.
- *workflow* – egy alapértelmezett munkamenetet határoz meg az action-ok számára. Ha az action implementálja a `Validatable` interfészt, az interceptor meghívja a **validate()** metódust. Ha implementálja a `ValidationAware` interfészt, ellenőrzi, hogy az action osztályban be van-e állítva valamilyen hibaüzenet a **hasErrors()** metódus segítségével. Ha van valamilyen hiba, akkor az interceptor az `input`-ra irányít, anélkül, hogy az action végrehajtódna.
- *prepare* – meghívja a **prepare()** metódust az action-re, amennyiben az implementálja a `Preparable` interfészt.

Előre definiált interceptor vermek

- interceptorok valamilyen sorozatához nevet rendelhetünk: *interceptor-verem*
- a `struts-default.xml` tartalmaz néhány előre definiált interceptor vermet

pl. – lásd `struts-default.xml`:

- defaultStack (alapértelmezett)
- basicStack
- paramsPrepareParamsStack

```
<interceptor-stack name="stackName">
  <interceptor-ref name="interceptor1" />
  <interceptor-ref name="interceptor2" />
  ...
</interceptor-stack>
```

lehetőség saját interceptor megírására:

- kódismétlés kiküszöbölése
- pl. annak ellenőrzése, hogy be van-e jelentkezve a felhasználó vagy megvannak-e a megfelelő jogai
- `AroundInterceptor` osztály kibővítésével
- állapot nélküli

OGNL

- OGNL – Object Graph Navigation Language
  http://commons.apache.org/ognl/

- a Struts2 kényelmes hozzáférést biztosít az action által előkészített adatokhoz
- ennek alapja egy *kifejezés-nyelv* (OGNL), melynek segítségével könnyen hivatkozhatunk JavaBean-ek tulajdonságaira, kollekciókra, metódusokra

- kontextus-al dolgozik – a Struts2 esetében ez megegyezik az ActionContext objektummal
- kontextus gyökér tartalma: általunk létrehozott lokális objektumok, nyilvános hatókörű objektumok, érték-verem (value stack)

OGNL kifejezések használata:

- a Struts2 elemkönyvtár elemeiben: %{...}
- ha bizonyos Struts2 tag-ek által bevitt, illetve nyilvános hatókörben tárolt objektumokra hivatkozunk, akkor:%{# ...}
  (pl. %{#session.user.name})
- az OGNL érték-vermében egy konkrét elemre az alábbi szintaxissal hivatkozhatunk:
  %{[level]....}
  (pl. %{[1].name })

JavaBean tulajdonságokhoz való hozzáférés:

- a JavaBean tulajdonságok alakja: getXxx(), setXxx(), isXxx(), hasXxx() (utóbbi kettő boolean tulajdonságok esetén)
- az ezekhez való hozzáférés **xxx**-el történik

- OGNL-ben haszálhatjuk a Java-ból ismert matematikai operátorokat

metódushívás:

- meghívhatunk bármilyen metódust, nem csak get vagy set metódusokat

Struts2 tag-ek-ről bővebben:

- http://struts.apache.org/2.x/docs/struts-tags.html