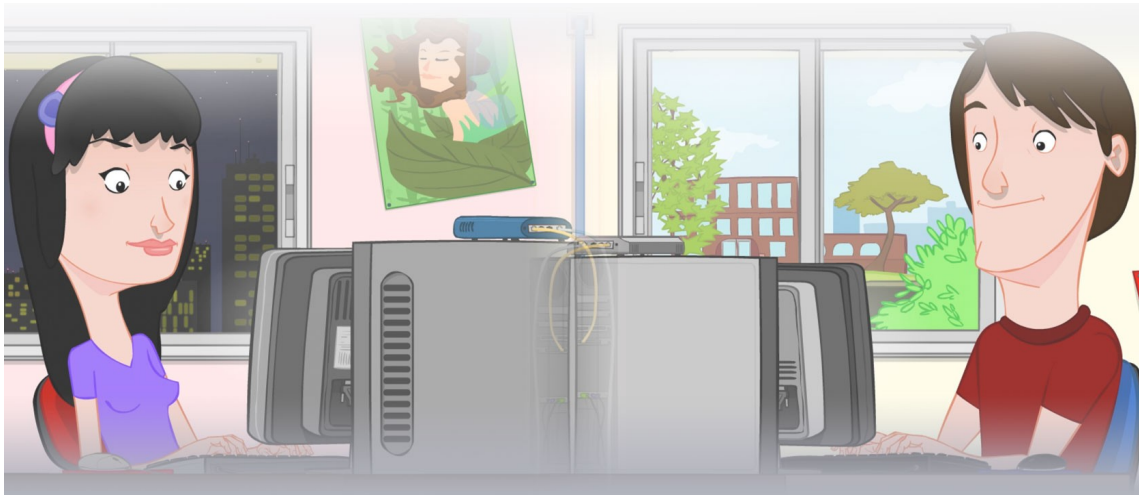
6. PRAKTIKA

JSF



Universidad del País Vasco
Euskal Herriko Unibertsitatea

OCW 2015

UPV/EHU

ZERBITZU TELEMATIKO AURRERATUAK: 6. PRAKTIKA

AURKIBIDEA

6. PRAKTIKA: JSF

1 Helburua

Aurreko praktiketan dendako administrari zein erabiltzaile arruntentzako aurkezpena, Java Servlet teknologiarekin garatu dugu. Aurkezpenetarako HTML kodea horrela sortzea nahiko neketsua denez, beste teknologia egokiagoak daude gaur egun JAVA EE barruan eta oraingo praktikan horietatik JSF eta Facelets direlakoak landuko ditugu.

2 Proiektua

Eclipse IDEan proiektu berri bat sortu behar dugu, Dynamic Web Project motakoa eta `web.xml` fitxategia duena. Bere propietateetan aldaketa hauek egin:

- Properties→Project Facets: JSF 2.2
  - Further configuration available...
    - Type: User Library
    - Download library...→JSF 2.2 (Mojarra 2.2.0)→Next→Lizentzia onartu→Finish
    - JSF Implementation Library: Mojarra liburutegi berria onartu
    - `URL Mapping Patterns`: `Add→*.xhtml`

Azkenik, REST Web Zerbitzuekin lan egiten jarraitzeko, hauek egin:

- Aurreko praktikako Jersey JAX-RS eta JACKSON liburutegiak gehitu Java Build Path eta Deployment Assembly ataletan.
- `web.xml` fitxategian REST Web Zerbitzuak erabiltzeko beharrezkoa gehitu, aurreko praktikan bezala.

Proiektu mota honetan jadanik MVC patroia jarraituko dugu, 3 mailetako softwarea programatuta.

3 Model

3.1 Datuen Atzipena

Dendako produktuek XML fitxategi batean egoten jarraituko dute. Horretarako, orain arte egin den bezala egingo dugu, `datuak` izeneko pakete batekin. Beraz, nahikoa da aurreko praktikako `datuak` paketea inportatzea oraingo proiektuan, eduki hauek dituena:

- XML fitxategiaren edukia zehazten duten JavaBean klaseak, `@Xml` oharrekin apaindutakoak
- XML fitxategitik irakurri/idazteko metodoak ditu(zt)en klasea(k)

3.2 Negozioaren logika

Dendan produktu berriak sartu eta daudenen zerrenda lortzeko, aurreko praktikan REST Web Zerbitzuak

programatu genituen. Oraingoan ere horiek erabiliko ditugu, beraz, `logika` paketea egin beharko dugu eta bertara aurreko praktikako `logika` paketearen edukia kopiatu.

Oraingo bertsioan aldaketa bakarra egin behar dugu: aurrekoan produktu bat jaso eta XML fitxategian sartzen zuen zerbitzuak, oraingoan produktuen zerrenda jasoko du eta berau sartu beharko du fitxategian, aurretik zegoen guztia ezabatuz.

4 View

JSF teknologia darabilten proiektuetan, View atala 2 osagai mota ezberdinekin osatzen da. Alde batetik, `ManagedBean` motako JavaBean klaseak programatu behar dira eta bestetik berriz, HTML kodea nola sortu behar den adierazten duten `.xhtml` fitxategiak.

4.1 ManagedBeans

Esandako `ManagedBeans` klase horiek, `aurkezpena` paketearen edukia izango dira; hortaz, kasu honetan ez dugu aurreko praktikaren bertsioko ezer kopiatu behar, bertsio hartan servlet-ak baitzeuden pakete horretan eta oraingoan servlet horiek JSFrekin ordezkatuko baititugu (hau da, ManagedBeans eta .xhtml fitxategiekin).

1. `aurkezpena` paketean, `ProduktuaMB` klasea sortu, `ManagedBean` motako JavaBean bat izango dena:
   - Esparrua: `HTTP-request` prozesamendua
   - Produktu baten atributu guztiak edukiko ditu: kodea aurrezteko, `datuak` paketean produktu bat adierazten duen JavaBean-aren eratorria bezala deklaratu.
   - Atributu berri bat ere edukiko du, `boolean` motakoa. Produktu berri bat zerrendan sartzea lortu den ala ez adierazteko balioko du.
   - Lehenetsitako eraikitzailea idatzi, produktuaren erreferentzia sar dezan (aurreko praktiketan bezala).
   - Atributu berriaren `getter` eta `setter` metodoak idatzi.
   - Gainera, `Object`-etik heredatutako `toString()` metodoa birdefinitu, produktuaren datuekin eratutako `String` bat itzuli dezan.
2. `aurkezpena` paketean, `ProduktuenZerrendaMB` klasea sortu, `ManagedBean` motako beste JavaBean bat izango dena:
   - Esparrua: aplikazioa abiatuta dagoen denbora guztia.
   - Atributu bezala dendako produktu guztien zerrenda izango du: kodea aurrezteko, `datuak` paketean produktuen zerrenda bat atribututzat duen JavaBean-aren eratorria bezala deklaratu.
   - Lehenetsitako eraikitzailea birdefinitu behar da, produktuen zerrenda XML fitxategitik irakur dezan. Oraingoz, hutsik utzi.
   - Metodo berri bat egin, parametro bezala `ProduktuaMB` motako objektu bat hartzen duena eta zerrendan sartzen duena. Sartzea lortzen badu, `ProduktuaMB` motako objektuaren `boolean`

atributuan islatuko du.

- Metodo berri bat egin, dendako produktuen zerrenda XML fitxategitik irakurri eta itzultzen duena. Horretarako, metodoak `logika` paketean idatzitako REST zerbitzuetako bat erabiliko du, hau da, metodoan REST zerbitzu horrekiko bezero bat programatu beharko dugu.

  Lehenetsitako eraikitzailearen birdefinizioa amaitu daiteke orain, bere zeregin bakarra metodo honek itzultzen duen zerrendarekin klasearen atributua betetzea delarik.

- Azkenik, beste metodo berri bat egin, atribututzat dugun produktuen zerrenda XML fitxategian idatziko duena. Horretarako, metodoak `logika` paketean idatzitako REST zerbitzuetako bat erabiliko du, hau da, metodoan REST zerbitzu horrekiko bezero bat programatu beharko dugu.

`datuak`, `logika` eta `aurkezpena` paketeak esan bezala programatuta, produktuen zerrenda bi modutara mantentzen da gure aplikazioan. Alde batetik, XML fitxategian (orain arte bezala) eta bestetik, zerbitzariaren memorian `ProduktuenZerrendaMB` motako objektu batean. Banaketa horrek zera ahalbidetzen digu:

- Model bloke osoa beste zerbitzari batean edukitzea, dendako produktuen XML fitxategia beste zerbitzari horretan egongo delarik. Oraingoan, ez dugu egingo.
- View eta Controller blokeak dituen zerbitzarian berriz, produktuen zerrenda memorian edukitzea, aplikazioak abiatuta irauten duen denbora guztian.

  Zerrendaren bertsio hau administrariak produktuak sartzen dituen heinean aldatuz joango da, eta uneren batean (ez produktu berri bakoitzagatik), XML fitxategian gorde ahalko da. Horrela, administrariari erabakitzeko aukera eman ahalko zaio memoriako zerrenda dendan (hau da, XML fitxategian) benetan noiz egotea nahi duen.

4.2 xhtml

View atalarekin amaitzeko, HTML kodea sortzea eragingo duten `xhtml` fitxategiak idaztea falta da. Fitxategi horiek, Facelets txantiloi bat erabiliko dute:

1. Proiektuaren `WebContent` karpetan, `EDUKIAK/TXANT` karpetak egin eta bertan `dendakoTxantiloia.xhtml` fitxategia sortu:
   - Dendako orrialde guztietan berdin sortu behar den HTML kodea adieraziko da bertan (CSS estilo fitxategiarekiko `link` elementua, `head` elementu barruko `title`, `body` elementuko `h1`...).
   - Aldagarriak izango diren zatiak `ui:insert` modura adierazi.
2. Proiektuaren `WebContent` karpetan, `EDUKIAK/ERAB` karpeta egin eta bertan `erabiltzailea.xhtml` fitxategia sortu:
   - Aurreko txantiloia erabiliko du (`ui:composition`), orrialdearen izenburua eta edukia zehaztuz (`ui:define`).
   - `h:dataTable` elementu bat izango du, XML fitxategian dauden produktuen zerrenda bistaratuko duena. Horretarako, `ProduktuenZerrendaMB` motako objektutik dagokion metodoak itzulitako zerrenda erabiliko du.
   - Taula ondo bistaratu dadin, `h:column` elementu bakoitzean `f:facet` eta `h:outputText` egokiak jarri.

3. Proiektuaren `WebContent` karpetan, `EDUKIAK/ADMIN` karpeta egin eta bertan `administraria1.xhtml` fitxategia sortu:
   - Aurreko txantiloia erabiliko du (`ui:composition`), orrialdearen izenburua eta edukia zehaztuz (`ui:define`).
   - Formulario bat bistaratuko du, produktu berri baten datuak sartzea ahalbidetzeko (erreferentzia izan ezik), JSF elementuekin (`h:form`, `h:outputLabel`, `h:selectOneMenu`, `f:selectItem`, `h:inputText`,...).
     - Datu guztiak sartzea derrigorrezkoa izango da eta mezu bat bistaratuko da (`h:message`) hutsik uzten den eremu bakoitzeko.
     - Formularioko datuekin betetako produktu berri bat aplikazioko `ProduktuenZerrendaMB` motako objektuan gorde dadin, `h:commandButton` elementu bat jarri behar da. Bere `actionListener` atributuak, objektu horretako metodo egokia deituko du produktu berria memoriako zerrendan gera dadin (baina XML fitxategira pasatu gabe).
   - Produktu berri bat memoriako zerrendan sartzea lortu den kasuetarako, `h:panelGroup` elementu baten bidez mezu bat bistaratuko zaio administrariari. Elementu hau beraz produktua benetan sartzea lortzen denean bakarrik bistaratuko da, baldintza hori `rendered` atributuarekin adierazi behar dugularik `h:panelGroup` elementuan.
   - Azkenik, `h:outputLink` elementu bat jarri ere, `administraria2.xhtml` orrialdera joateko, administrariak denda nola geratuko litzatekeen ikusi ahal izateko.
4. `EDUKIAK/ADMIN` karpetan bertan `administraria2.xhtml` fitxategia sortu:
   - Aurreko txantiloia erabiliko du (`ui:composition`), orrialdearen izenburua eta edukia zehaztuz (`ui:define`).
   - `h:dataTable` elementu bat izango du, aplikazioaren memorian dagoen produktuen zerrenda bistaratuko duena. Horretarako, `ProduktuenZerrendaMB` motako objektutik dagokion metodoak itzulitako zerrenda erabiliko du.
   - Azkenik, `h:form` elementu bat jarri, 2 `h:commandButton` elementu dituena
     - Batak, `ProduktuenZerrendaMB` motako objektuko zerrenda XML fitxategira pasatuko du eta administrariari `erabiltzailea.xhtml` orrialdea bistaratuko dio denda nola geratu den jakin dezan.
     - Besteak, berriz, administraria `administraria1.xhtml` orrialdera itzuliko du, memoriako zerrendan produktu berriak sartzen jarrai dezan, XML fitxategira ezer pasa gabe.

5 Controller

Amaitzeko, MVCko Model eta View blokeak komunikatzeaz arduratzen den Controller atala falta zaigu. JSF motako proiektuetan badago jadanik programatuta eta erabilgarri, `FacesServlet` izeneko servlet-arekin. Controller modura lan egiten duen servlet horren funtzionamendua web.xml fitxategiarekin konfiguratzen da. Proiektu honetarako, web.xml fitxategian aldaketa hauek egin:

1. Proiektuaren `welcome-file` modura `erabiltzailea.xhtml` fitxategia adierazi.
2. `*.xhtml` fitxategiak `FacesServlet`-era mapeatu: proiektua prestatzean jadanik egin dugu `URL`

`Mapping Patterns` bidez.

3. REST Web Zerbitzuetarako: aurreko praktikan sartutako elementuak oraingoan ere idatzi.