Uebung 8

1.

a.)

Entwicklung der ELs:

- in JSP < 2.0 (JSTL 1.0):
  - Syntax `${...}`
  - durfte nur innerhalb von Tags verwendet werden
  - JSP hatte kein Model-Konzept (auch in 2.0 nicht)
- in JSP 2.0
  - Syntax `${...}`
  - durfte auch ausserhalb von Tags verwendet werden
- Mit JSF (auf Basis von JSP 2.0):
  - Weitere EL
  - andere Syntax: `#{...}`
- JSP 2.0 + JSF 1-2
  - Zusammenführung der ELs zu Unified EL
- verschiedene Versionen:
  - standard: get und set
  - deferred: nur get
  - unified: Vereinigung der standard aus JSP und deferred aus JSF

Die verschiedenen ELs:

- JSF 2.0 EL:
  - `#{...}`
  - Verwendung in ganzer Facelet Page
- JSF 1.x EL (mit JSP):
  - `#{...}`
  - Verwendung nur in Attributen von JSF-Tags
- JSP 2.0 EL:
  - `${...}`
  - Verwendung in ganzer Page

Wozu?

- Zweck ist die Kommunikation der Präsentationsschicht mit der Anwendungslogik
- dynamischer Zugriff auf JavaBeans

b.)

Passiert in der Render-Response-Phase

Reihenfolge:

1. request
2. session
3. application

c.)

- Komponente verwenden wenn Ausgabeelement konfiguriert werden muss
- ansonsten egal

d.)

Iteration über die Rückgaben von p1, p2 und p3 und jeweiliger Bean-Lookup wie in *c.)*

e.)

- Anlegen eines Model-Objekts (oder einer Property)
- Verändern eines Model-Objekts (oder einer Property)
- Ausgeben einer Repräsentation eines Model-Objekts (oder einer Property)
- Actions und ActionListener

f.)

Objekte die im Zusammenhang stehen mit der Verarbeitung/Ausführung eines Requests oder mit der Umgebung.

Zur verfügung stehen:

1. facesContext: an instance of FacesContext. FacesContext contains all of the per-request state information related to the processing of a single JavaServer Faces request, and the rendering of the corresponding response.
2. application: an instance of the ServletContext. A ServletContext instance provides access to the execution environment i.e. the servlet container.
3. initParam: A Map of the initialization parameters of this web application.
4. session: an instance of HttpSession. A HttpSession can be used to bind objects, get information about a session, such as the session identifier, creation time, and last accessed time. Session information is scoped only to the current web application (ServletContext), so information stored in one context will not be directly visible in another
5. view: The current UIViewRoot for this view. UIViewRoot is the UIComponent that represents the root of the UIComponent tree.
6. component: The UIComponent instance being currently processed at the time of evaluation.
7. cc: The top-level composite component currently being processed.
8. request: an instance of HttpServletRequest.
9. applicationScope: A Map (name and value) of all application scope attributes.

10. sessionScope: A Map (name and value) of all attributes in the current session.
11. viewScope: A Map (name and value) of all attributes in the current view scope.
12. requestScope: A Map (name and value) of all attributes in the current request being processed.
13. flowScope:* #{flowScope}, for flow local storage. This maps to facesContext.getApplication().getFlowHandler().getCurren
14. flash: A Map for forwarding temporary objects between the user/next views generated by the faces lifecycle or the current view using #{falsh.now} (see flash.putNow(..)).
15. param: A Map (name and value) of HTTP request parameters, containing only the first value for each name.
16. paramValues: A Map of HTTP request parameters, yielding a String[] array of all values for a given name.
17. header: A Map of HTTP header parameters for the current request.
18. headerValues: A Map of HTTP header parameters for the current request, yielding a String[] array of all values for a given name.
19. cookie: A Map of the cookie names and values of the current request.
20. resource: A Map of application resources.
21. pageContext ${pageContext.request.contextPath}

g.)

Steht im Konflikt mit dem MVC-Modell. Man kann am vorgesehenen Lebenszyklus vorbei arbteiten.

h.)

1. Arithmetic
2. Logical
3. Relational
4. Conditional
5. (Empty)

i.)

- Da manche Operatorsymbole schon in der EL eine bedeutung haben, müssen diese escaped werden. Kann zu Fehlern führen.
- Übermäßige Nutzung widerspricht dem MVC-Model, Logik soll nicht in die Facelets.

j.)

Ab JSF 2.2 können bei Methodenaufrufen Parameter übergeben werden. Coole Sache. Verleitet allerdings wieder dazu Logik in den Facelets zu implementieren.

k.)

Servlet 3.0 / EL 2.2 capable container like Tomcat 7, Glassfish 3, JBoss AS 6, etc and your web.xml is been declared as per Servlet 3.0 specification.