JEE Quiz 11042016

1. JSP : Java Server Pages. JSP permet de créer du code pour afficher une page web, qui peut être dynamique. JSP fonctionne avec des balises pour délimiter le code dynamique.
   JSTL : Java Server Page Standard Library. C'est un ensemble de tags permettant d'exploiter les fonctionnalités des JSP.
   Servlet : un servlet est une classe permettant de créer des données dans un serveur http au sein d'une classe Java.

2. Il existe trois types de portée de variable :
   - Page : n'existe que sur une JSP
   - Requête : peut exister d'une JSP à l'autre en cas de JSP forward
   - Session : persistante entre les pages

3. Il y a trois façons de développer un servlet :
   - Créer une classe implémentant la classe de base Servet et toutes ses méthodes
   - Créer une classe étendant la classe générique GenericServlet et implémenter la méthode service()
   - Créer une classe étendant la classe httpServlet et implémenter les méthodes nécessaires – c'est la méthode la plus fréquemment utilisée

4. S'il n'existe qu'une instance par servlet dans le cas par défaut, chaque requête adressée au servlet vit dans un thread séparé. Le servlet peut donc effectuer plusieurs tâches à la fois.

5. Les fonctions doGet et doPost ont les signatures suivantes :
   doGet(HttpServletRequest, HttpServletResponse);
   doPost(HttpServletRequest, HttpServletResponse);

6. Les JSP ont des objets implicites, parmi lesquels on peut trouver, entre autres, « request » pour la requête et « session » pour la session.

7. Pour récupérer les données de JSP au servlet, on peut écrire le code suivant dans le servlet :

```
protected void doPost(HttpServletRequest request, HttpServletResponse
response)
      throws ServletException, IOException
{
    String background = request.getParameter("fond");
    String text = request.getParameter("texte");
}
```

8. Un JavaBean est une classe respectant un ensemble de conventions. Elles sont créées pour encapsuler un objet (le « bean »). Ainsi, la classe doit être « Serializable », avoir des getters et setters pour ses attributs privés, avoir un constructeur par défaut (aucun paramètre n'étant requis), et ne doit pas être déclarée « final ».

9. Pour récupérer des données du servlet dans une JSP, on peut procéder de la sorte :

   Dans le servlet :

```
    request.setAttribute("nom", getNom());
    request.getRequestDispatcher("mypage.jsp").forward(request, response);
```

Puis dans le JSP :

```
<%
     Object nom = request.getAttribute("nom");
%>
```

Dans le JSP, on pourrait aussi utiliser le code suivant (ce qui est mieux, car les scriptlets deprecated) :

```
${nom}
```

10.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>
    <servlet>
        <servlet-name>login</servlet-name>
        <servlet-class>servlet.Login</servlet-class>
    </servlet>
    <servlet>
        <servlet-name>liste</servlet-name>
        <servlet-class>servlet.ListeMembre</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>liste</servlet-name>
        <url-pattern>/maListe</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>login</servlet-name>
        <url-pattern>/login</url-pattern>
    </servlet-mapping>
    <context-param>
        <param-name>login</param-name>
        <param-value>admin</param-value>
    </context-param>
    <init-param>
        <param-name>titrePage</param-name>
        <param-value>Bonjour admin</param-value>
    </init-param>
</web-app>
```

Init-param doit être dans une balise servlet.