ÉCOLE SUPÉRIEURE DES SCIENCES ET DE LA TECHNOLOGIE DE HAMMAM SOUSSE			
Matière : Java EE	**Date :** Mars 2019	**Durée :** 2 Heures	**CORRIGÉ**
Enseignant : Ala Eddine KHARRAT	**Niveau :** 3LFSI	**Session :** Principale	

Exercice 1 : Configuration du fichier « *web.xml* » (3 points)

1) Dès la version ***Servelet 3.0***, le fichier ***Web.xml*** devient optionnel. Il suffit d'utiliser l'annotation ***@WebServlet*** avant la classe Servlet en question. ***(0.5 point)***

2)

a) ***(0.5 point)***

```
<servlet>
	<servlet-name>profil</servlet-name>
	<servlet-class>controllers.Profil</servlet-class>
</servlet>
```

b) ***(0.5 point)***

```
<servlet-mapping>
	<servlet-name>profil</servlet-name>
	<url-pattern>/profil/*</url-pattern>
	<url-pattern>/Profil/*</url-pattern>
</servlet-mapping>
```

Signification : La servlet ***« Profil.java »*** peut être appelée par l'une des deux URLs ***« /profil/ »*** et ***« /Profil/ »*** suivie de n'importe quelle combinaison de caractères alphanumériques. ***(0.5 point)***

c) ***(0.5 point)***

```
<error-page>
	<error-code>404</error-code>
	<location>/404.jsp</location>
</error-page>
```

3) ***(0.5 point)***

```
@WebServlet({ "/Profil", "/profil" })
```

Exercice 2 : Connexion et déconnexion d'un étudiant (7 points)

1)

a) Naturellement, la connexion à la base de données doit être faite qu'une seule fois. Il est judicieux donc, d'instancier l'objet ***« StudentDAOImpl »*** dans la méthode ***« init »*** puisque cette dernière n'est appelée par le serveur qu'une seule fois à l'exécution de l'application. ***(0.5 point) : Toute autre justification correcte sera acceptée.***

b) ***(3 points)***

```
String login = request.getParameter("login");
String password = request.getParameter("password");

int statut = studentDAOImpl.verifyLogin(login, password);

if(statut==1) {

    Student student = studentDAOImpl.getStudentByLogin(login);

    HttpSession session = request.getSession();
    session.setAttribute("etudiant", student);
    rd = request.getRequestDispatcher("index.jsp");

} else if (statut==0) {
    request.setAttribute("error", extra.Strings.ERROR_LOGIN);
    rd = request.getRequestDispatcher("login.jsp");
} else {
    request.setAttribute("error", extra.Strings.ERROR_DB_PROBLEM);
    rd = request.getRequestDispatcher("login.jsp");
}

rd.forward(request, response);
```

c) ***(0.5 point)***

```
action="login" method="POST"
```

d) ***(1 point)***

```
<%
   String errorMessage = (String) request.getAttribute("error");
   if(errorMessage!=null){
      out.println("<b style='color:red;'>"+errorMessage+"</b>");
   }
%>
```

e) ***(1 point)***

```
<%
   Student student = (Student) session.getAttribute("etudiant");
   if(student!=null){
      response.sendRedirect("index.jsp"); // Toute autre solution cohérente sera
      acceptée.
   }
%>
```

2)

a. ***(0.5 point)***

```
HttpSession session = request.getSession();
session.invalidate(); // ou bien session.removeAttribute("etudiant") ;
```

b. Logiquement les lignes de code permettant de faire la déconnexion doivent être décrites dans la méthode ***« doGet »*** afin que l'étudiant puisse se déconnecter directement en utilisant une URL simple. ***(0.5 point)***

Exercice 3 : Gérer la fiche des résultats d'un étudiant (3 points)

```
HttpSession session = request.getSession();
Student student = (Student) session.getAttribute("etudiant");

List<Mark> marksList = markDAOImpl.getMarksByUserId(student.getIdStudent());

if(marksList.size()!=0) { // ou bien (marksList==null) : ça dépend votre vision de
                                                          la question.
     response.setContentType("application/vnd.ms-excel");
     PrintWriter out = response.getWriter();
     DecimalFormat df = new DecimalFormat("00"); // ça dépend aussi de votre vision.

     out.println("Etudiant\t" + student.getFullName());
     out.println("Date\tNote /20");

     for(int i=0; i<marksList.size(); i++) {
         float currentMark = marksList.get(i).getMark();
         out.println(marksList.get(i).getDate() + "\t" + df.format(currentMark));
     }

     out.println();
     out.print("Moyenne\t=AVERAGE(B4:B" + (3+marksList.size()) + ")");
     out.close();
}
```

Exercice 4 : Chargement de questions (7 points)

1) ***(2 points) : Toute autre solution qui n'entre pas en contradiction avec les contraintes de l'exercice sera acceptée.***

```
List<Integer> questionsList = (List<Integer>) request.getAttribute("questionsList");
List<Integer> topicsList = (List<Integer>) request.getAttribute("topicsList");
boolean goodQuestion = true;
Question newQuestion;
do {
     Random random = new Random();
     int id = random.nextInt(500) + 1;
     newQuestion = questionDAOImpl.getQuestionById(id);

     for(int i=0; i<questionsList.size(); i++) {
        if(questionsList.get(i)==id) {
           goodQuestion = false;
           break;
        }
     }

     if(goodQuestion) {
        int nbTopics=0;
        for(int i=0; i<topicsList.size(); i++) {
           if(topicsList.get(i)==newQuestion.getIdTopic())
              nbTopics++;
           if(nbTopics>=5) {
              goodQuestion = false;
              break;
           }
        }
     }
}while(!goodQuestion);

return newQuestion;
```

2) ***(0.5 point)***

```
if(question.getSolution().equals(proposition))
	return 1;
else
	return 0;
```

3) ***(3 points) : Toute autre solution qui respecte la démarche de l'exercice sera acceptée.***

```
Integer finalMark = (Integer) request.getAttribute("finalMark");
List<Integer> questionsList = (List<Integer>) request.getAttribute("questionsList");
List<Integer> topicsList = (List<Integer>) request.getAttribute("topicsList");
Question savedQuestion = (Question) request.getAttribute("question");

String proposition = request.getParameter("proposition");
finalMark+= VerifyQuestion(savedQuestion, proposition);

if(questionsList.size()<20) {
		Question newQuestion = getNewQuestion();
		questionsList.add(newQuestion.getIdQuestion());
		topicsList.add(newQuestion.getIdTopic());

		request.setAttribute("questionsList", questionsList);
		request.setAttribute("topicsList", topicsList);
		request.setAttribute("question", newQuestion);
		request.getRequestDispatcher("test.jsp").forward(request, response);
}else {
		HttpSession session = request.getSession();
		Student student = (Student) session.getAttribute("etudiant");
		Mark newMark = new Mark();
		newMark.setMark(finalMark);
		newMark.setIdStudent(student.getIdStudent());
		markDAOImpl.addNewMark(newMark);

		PrintWriter out = response.getWriter();
		out.println("<h1>"+ Strings.RESULT_MSG + finalMark + "/20</h1>");
		out.println("</br>"+Strings.REDIRECTION_MSG);
		response.setHeader("Refresh", "5;URL=index.jsp");
}
```

Cette partie n'est pas demandée dans l'exercice. Elle sera notée comme BONUS.

4) ***(1.5 points)***

```
<%Question question = (Question) request.getAttribute("question");%>
// On peut ajouter un test : (question !=null)
<form action="test" method="POST">
   <input name="proposition" type="radio" value="A" /><%= question.getPropositionA()%>
   </br>
   <input name="proposition" type="radio" value="B" /><%= question.getPropositionB()%>
   </br>
   <input name="proposition" type="radio" value="C" /><%= question.getPropositionC()%>
   </br>
   <input name="proposition" type="radio" value="D" /><%= question.getPropositionD()%>
   </br>
   <input type="submit" value="Suivant" />
</form>
```