Cours Java 16

Application web servlet

**Diapo 2**

Les api servlet viennent de JEE (javax.servlet). Servlet conçu pour gérer des requêtes réponse.

**Diapo 3**

Frontend serveur apache. Contenu statique le serveur renvoie la bonne page. Si page dynamique traité par l’objet servlet, construction réponse, renvoie par objet servlet.

Servlet ne fait pas que gérer du html. Servlet containers est un serveur dédié pour pouvoir gérer les servlet (on utilisera le conteneur tomcat). Tomcat gère à la fois la partie statique et dynamique. Généralement fontend apache et backend tomcat. Tomcat facilite le développement d’appli web sans avoir à déployer et paramétrer un serveur http Apache.

**Diapo 4**

Servlet instancié une fois, et peut répondre à des requêtes venant de plusieurs utilisateurs en ayant des thread différent (transparent pour nous).

Req2 arrivant deux fois pour deux clients, on utilise le même servlet (car instancié une fois). Possibilité de charger et instancier dynamiquement par le conteneur. Donc plus de main c’est le conteneur qui s’exécute qui gère.

**Diapo 5**

Servlet toute petite partie de JEE. Servlet partie web. Remarque JDBC et passé dans JSE.

**Diapo 9**

Init() permet de faire des choses quand le servlet est créé. Le package Servlet.http spécialise le package Servlet. Les parties dans les entêtes réponse des requêtes http, ce sont les méthodes de la partie servlet.http qui nous permettrons de le faire.

**Diapo 10**

Pour faire un servlet on hérite de HttpServlet et on fait des overides sur les méthodes de base. Si on a des exceptions non gérées par la méthode redéfinie, il faut que l’on fasse le try catch nous.

**Diapo 12**

L’URL de la servlet est définie dans une annotation avant le début de la classe. L’url que l’on donne pour urlPatterns est un chemin relatif en fonction du conteneur de l’application. On peut avoir plusieurs URL qui sont gérés par un même servlet.

Redéfinition du doGet, on met ce que l’on veut faire dans le get.

**Diapo 18**  
C’est le conteneur qui gère la servlet, request crée par le navigateur et response par le conteneur. Servlet communique avec le client par l’url relative (dans @WebServlet).

Remarque sur les fichiers dans netbeans pour une appli web :

Dans projet dossier WebPages (retrouve toutes les ressources que l’on trouve dans un site standar, dont js script).

Libraries : .jar du class path du projet (driver JDBC)

Source package : code de l’appli

Vision fichier des choses

Src, les classes

Web, les pages internet

Test les executables de test

Le repertoire web de build c’est ce que l’on donne pour le déploiement sur un serveur. On le met en .jar pour l’échange. Netbeans le fait seul, clic droit clean and build. On trouve dans le rep dist, l’appli en .war (équivalent à un .jar, mais pour les web application).

Retour diapo 18

La partie action de <form> de la page html doit correspondre à l’url donné dans le servlet.

Diapo 21

La variable nombre est partagée par les deux clients, il y a une servlet (instancié une fois) pour les deux requêtes. Le servlet utilise des threads différent pour les requêtes des clients, mais ici nombre est commun pour les deux threads d’où le bug. La variable nombre étant l’attribut de l’objet servlet elle est partagée par tous les threads. Et donc peut être modifiée lors d’accès simultané de client (apparition d’un effet de bord).

Diapo 23

Attention aux attributs de classes !!!!

Diapo 24

Le conteneur (serveur) prépare des threads à l’avance (pool). Les servlets peuvent être chargées au démarrage ou au moment de la requête. Pour faire des actions au démarrage d’une servlet, c’est la méthode init() qu’il faut redéfinir.

Redéfinition de la méthode destroy() permet de pouvoir récupérer des traces de ce qui a été fait.

Diapo 25

Bloc synchronized n’est exécuté que pour un thread. Si une méthode peut poser des problèmes de concurrence de thread, on met synchronized dans sa signature. Elle ne pourra pas être appelée tant qu’elle est déjà dans un thread. Synchronized peut aussi être juste affecté à une variable.

La servlet peut aussi implémenter SingleThreadModel, mais alors une nouvelle instance de servlet est instancié. Ca peut être lourd si bcp de client (déprécié).

Diapo 26

Les cookies peuvent être crée par la servlet et mis dans la réponse, ou bien récupéré dans une requête. Pour envoyer un cookie, il faut le mettre dans l’entête (donc avant les out. … c’est-à-dire la partie données pour la création de la page).

Diapo 27

Attention cookie est stocké coté client, donc éviter informations critiques dans le cookie, sinon risque de modification par le client.

La session crée automatiquement un cookier permettant d’identifier le client.

Diapo 28

Si pas de cookie de présent dans la requête quand getSession(), le cookie est créé. Sinon on récupère les « infos » de session (deux navigateurs permettent de vérifier les sessions, car chaque navigateur gère ses propres cookies).

Diapo 29

Projet voir si messages pour avertir l’utilisation de cookie est utile ou non ?

Par défaut les sessions durent 30 min.

Diapo 30

Cast de getAttribute car cette fonction renvoie un object.

Diapo 31

Informations partagées au niveau de l’application utilisation de l’objet ServletContext

Diapo 32

ServletContext, c’est comme une session mais qui est partagé par tout le monde. Dans la méthode init() de la servlet on peut récupérer des paramètres de la servlet. Les paramètres que l’on récupère c’est ceux qui sont dans les fichiers de configuration de l’appli (à vérifier).