Cours Java AI

HTTP

**Diapo 3**

Protocol HTTP, pour que des programmes distants parlent entre eux.

Autres protocoles pour autres types d’échanges.

**Diapo 5**

Étape de chargement d’une page internet, demande de connexion au serveur, le serveur répond je suis prêt. Demande pour la page html. Renvoie la réponse HTTP. Et on continue les demandes sur la page pour l’afficher entièrement et le serveur nous renvoies les données.

Serveur les plus connus Apache et la version de windows (IIS). Serveur tomcat peut envoyer des pages internet mais gère aussi java. Apache est un serveur HTTP, on peut ajouter un module php si on veut. Apache est plus robuste et sécuritaire et on peut faire qu’Apache disctute avec Tomcat. Apache est en frontend et tomcat en backend

**Diapo 7**

Pour accéder à une application particulière sur le serveur, il faut connaitre le port.

**Diapo 9**

Port d’écoute sont les ports sur lesquels on écoute (au niveau du serveur). Quand requête de demande de connexion arrive sur le port standard (web : 80, ssh : 22, …) le serveur envoie un paquet indiquant qu’il accepte la connexion. Ce paquet contient un port différent du port standard, on a ainsi une connexion unique entre le serveur et le client. Si un deuxième client fait une demande de connexion on aura le même processus et le serveur utilisera un autre port libre pour communiquer avec ce nouveau client.

**Diapo 10**

URL permet d’accéder à un endroit spécifique dans le serveur.

**Diapo 13**  
Dans l’entête des requetés vers le serveur et les entêtes des réponses permettent d’avoir des informations spécifiques sur l’échange.

**Diapo 14**  
Détails de la requête depuis la machine cliente.

User agent permet au développeur de voir la proportion des browsers utilisés et donc on peut adapter le code pour qu’il soit plus optimal pour certain navigateur.

**Diapo 15**

Détails de l’entête de la réponse du serveur.

Content-type : permet au browser du client de savoir quel type de contenu il va recevoir.

**Diapo 16**

Configuration générale des entêtes requête et réponse.

**Diapo 17**

Head, on n’aura que les entêtes de la réponse, pas le code de la page.

Post, pour faire un traitement de ressources

**Diapo 20**

Referer, permet de savoir depuis quel lien la page a été demandée. Surtout quand elle vient d’un autre site, permet de savoir comment on est référencé.

**Diapo 21**

Accept langage permet de savoir les langues « favorites » de l’utilisateur et donc permet de fournir la page avec la bonne langue.

Codage faire en UTF-8 (pour éviter au maximum les problèmes de codage/décodage.

**Diapo 22**

If modified since, on prend en local les informations d’un site que l’on a déjà visité pour ne pas avoir à recharger les infos non modifiées.

Effacer le cache permet si on développe une appli d’être sûr de voir la modification que l’on a réalisée et pas tourner avec ce que l’on a en local dans notre cache.

**Diapo 24-28**

Le code réponse permet d’indiquer si tout s’est bien passé ou s’il y a eu un problème.

Pour voir les codes on passe par l’outils développeur de Firefox dans l’onglet réseau.

**Diapo 31**

Le cache récupère les informations des pages que l’on visite

**Diapo 33**

On peut changer la réponse par défaut de l’erreur 404 si on le souhaite.

**Diapo 34**

On peut faire en sorte qu’une partie du site nécessite une identification. On a un code 511 pour dire qu’il faut une identification. Apparition fenêtre login / password chez le client si l’authentification est bonne, on aura alors en réponse la page souhaitée.

**Diapo 36**

Formulaire html pour dialoguer avec le serveur.

La partie <form> contient les différents types de zone de saisie. En paramètre de <form> pour savoir où va aller le formulaire et quel sera son utilisation.

Html 5 implémentation de nouveau type d’entrée (calendrier …)

**Diapo 37**

Les données sont envoyées en tant que couple clé valeur. Clé nom du champ et valeur ce qu’il y a été entré dans le champ.

**Diapo 38**

Post envoie de données au serveur. Les paramètres ne sont pas transmis directement dans l’adresse, car le paramètre « method » de <form> est de type POST. Les données sont alors dans le corps de la requête. Dans les données envoyées on voit qu’il y a codage des accents et des espaces (URL encodage et décodage).

**Diapo 39**

Si on fait un GET, les paramètres sont dans l’url. Mais ils sont tout de même encodé en URL. Ils sont décodés à la réception.

**Diapo 41**

Cookie un petit fichier texte que le serveur renvoie au navigateur et sauvegarder par le navigateur. Permet de mémoriser un état entre un client et le serveur. Le cookie peut contenir un identifiant et ainsi maintenir une connexion ouverte avec le serveur.

**Diapo 42**

Le serveur a pas de moyen de relié plusieurs requêtes avec un même client. Pour se faire il envoie un cookie et ceux cookie sera renvoyé lors des prochains échanges. Et donc la réponse est adaptée.

**Diapo 43**

Cookie permettent la gestion de session. Personnalisation de page, du pistage pour afficher les bonnes pubs … Le cookie fait 4ko au max.

Cookie c’est intrusif, donc il faut se méfier.

**Diapo 44**

Cookie peut être dans l’entête de la réponse HTTP, ou bien directement dans le code de la page ( en javascript ou autre).

**Diapo 45-46**

Toutes les requêtes faites sur le domaine : .comp.com, auront le cookie qui partira avec.

**Diapo 49**

Il y a un certain nombre d’attribut possible dans les cookies. On peut avoir plusieurs set cookies dans un même entête.

**Diapo 52**

Webstrorage permet de mémoriser les actions hors ligne coté client et tout est envoyé au niveau du serveur une fois qu’une connexion internet est possible.

**Diapo 53**

Une requête passe par le serveur intermédiaire (proxy) qui le redirige vers le serveur http

**Diapo 54**

Le proxy peut faire du cache et ainsi accélère la connexion au site. Le proxy peut laisser passer le trafic que dans un sens. Filtrage de certains sites pour les gens du réseau.

Masque des informations provenant de notre machine.

**Diapo 55**

Le proxy connait tt ce que l’on fait. Modif de pages à la volée.