Java EE

-

Cours 1

Cours de 2e année ingénieur
Spécialisation « Génie Informatique »

Présentation du cours

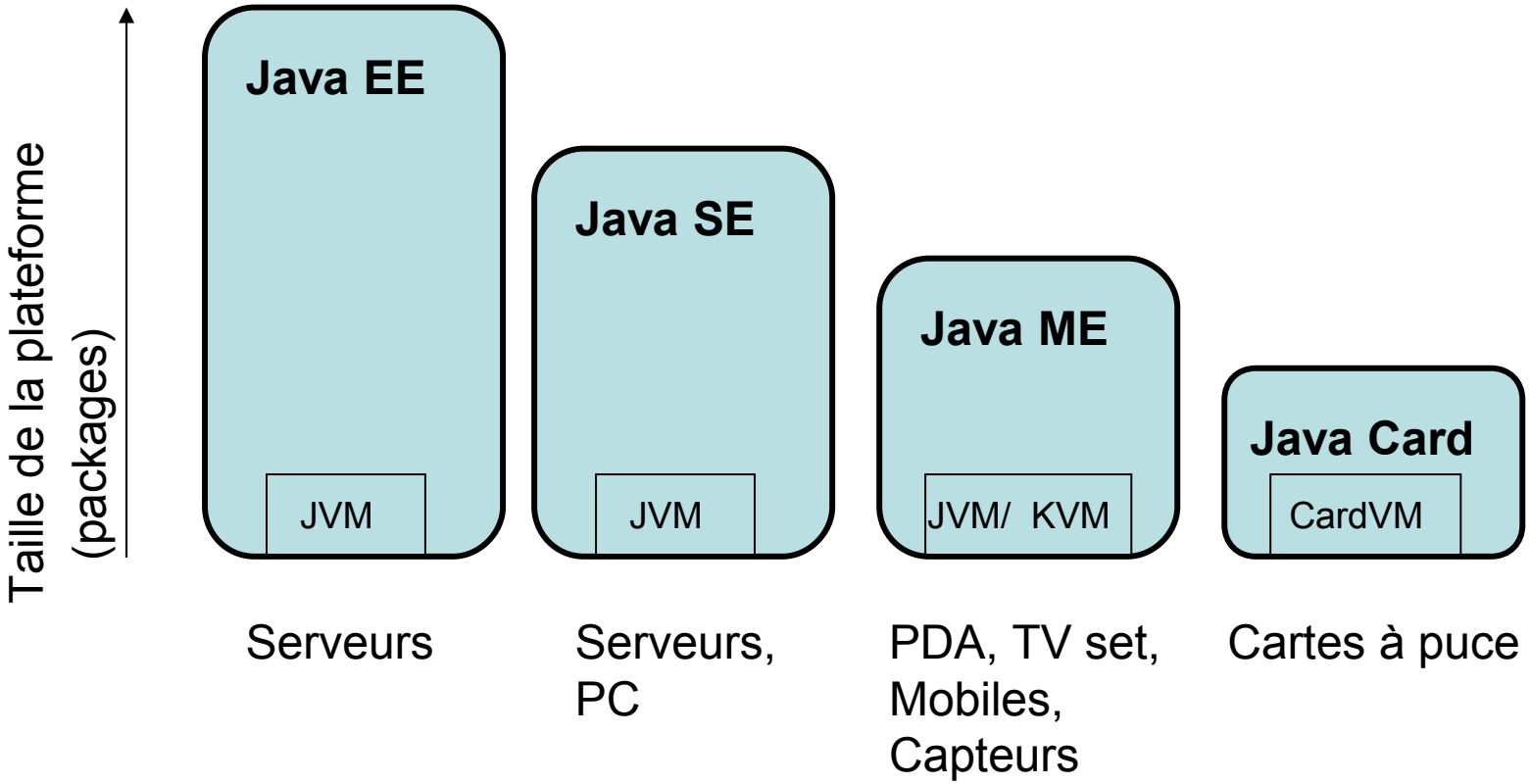
- Pôle GL-I2 « Génie logiciel »
  - 30h sur 10 semaines (3h de cours/TP)
- Objectifs
  - Développement d'applications Web ***robustes***
    - «*Ne pas réinventer la roue*» ⟶ utilisation d'un framework[1] (standard)
  - Apprentissage d'une partie de Java EE
    - Servlet, JSP et EL/JSTL
- Prérequis
  - Maîtrise du langage Java (Java SE)
  - Maîtrise du développement Web ***client***
    - XHTML (au moins balises de structure et formulaires)
    - CSS et XML sont un plus
  - Bases de Réseau (Architecture Client/Serveur)

[1] dans COO, utilisation des Design Patterns

Java EE?

- Java Enterprise Edition est un framework
  - riche (Java SE + nombreuses API)
  - ouvert (specs. du Java Community Process)
  - dédié au développement, au déploiement età l'exécution d'applications Internet ***modernes*** (nécessaires aux entreprises)
- Favorise la séparation des préoccupations
  - Code métier vs. Propriétés non-fonctionnelles
    - QoS, persistance (JPA), administration (JMX), sécurité, transaction (JTS/JTA),…

La galaxie Java



- Une édition n'inclut pas forcément la totalité d'une édition de plus « petite taille »

Le développement Web

- Le World Wide Web
  - Un SI public et universel déployé sur Internet
  - Un langage : HTML (ou XHTML)
  - Un modèle de communication : client-serveur
  - Un protocole : HTTP
- Pages statiques
  - Pages HTML préparées à l'avance
  - Le **serveur** renvoit les pages sans effectuer de traitement particulier
- Pages dynamiques
  - Pages HTML générées par le **serveur**
  - Le **serveur** construit la réponse en fonction de la requête de l'utilisateur

Serveur

• Serveur : un ordinateur disposant d'un certain nombre de ressources qu'il met à disposition d'autres ordinateurs (clients) via le réseau.

• Types de serveurs:
    • Serveur web
    • Serveur d'application
    • ...

Serveur web

- Programme s'exécutant sur une machine reliée à internet
- Protocole HTTP: répond aux requêtes des clients (navigateur web) et les traite
- Retourne des pages HTML au **Client**

Java et le développement Web

- Différentes technologies Java permettent de faire du développement Web à différents niveaux

	Développement	
	Côté Client	**Côté Serveur**
.class autonome	applet	servlet
Source Java mixé avec code html	JavaScript	JSP

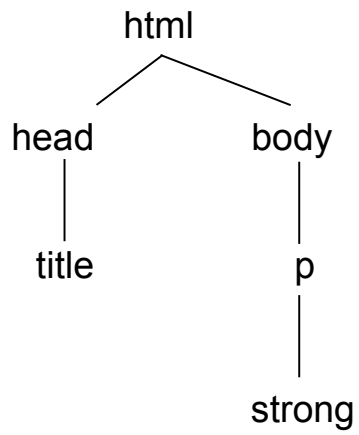
Pages statiques

Pages dynamiques

HTML (rappel)

- Langage de balisage, non propriétaire (W3C)
- Conçu pour afficher des documents sur le Web
- Liens hypertextes possibles entre les documents
- XHTML assure maintentant la compatibilité avec XML

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
 <head>
  <title>XHTML 1.0 valide !</title>
 </head>
 <body>
  <p>Une page XHTML 1.0 <strong>valide</strong>.</p>
 </body>
</html>
```



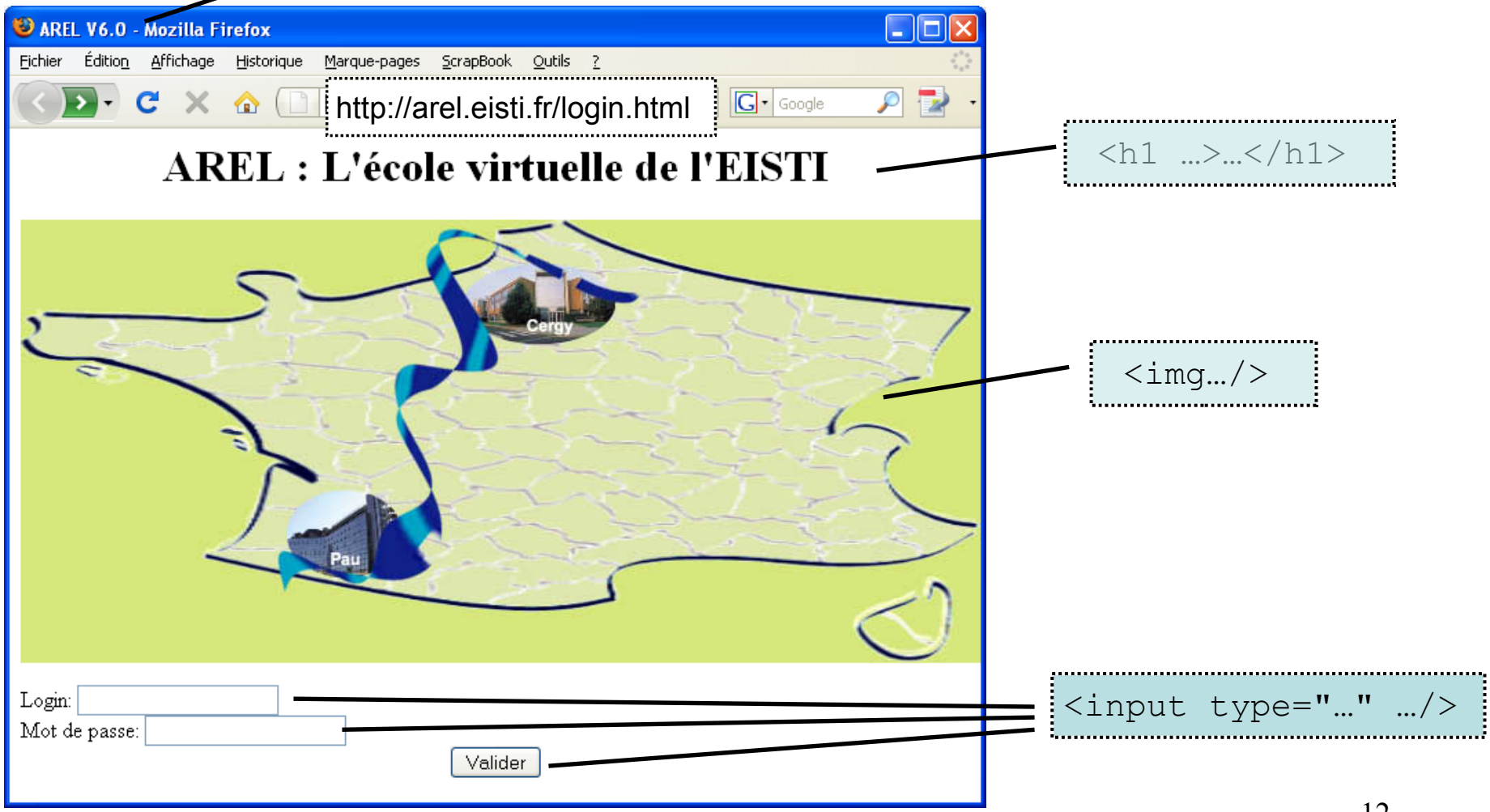
Quelques balises

- `<!--  -->` commentaires
- `<a>` ancre (hyperlien href)
- `<body>` corps du document
- `<br>` *line break*
- `<form>` formulaire
- `<h1>` titre1
- `<head>` entête
- `<html>` limite le document
- `<input type>` boutons et champs de saisie
- `<p>` paragraphe
- `<title>` titre

Ex : AREL V6 (code HTML)

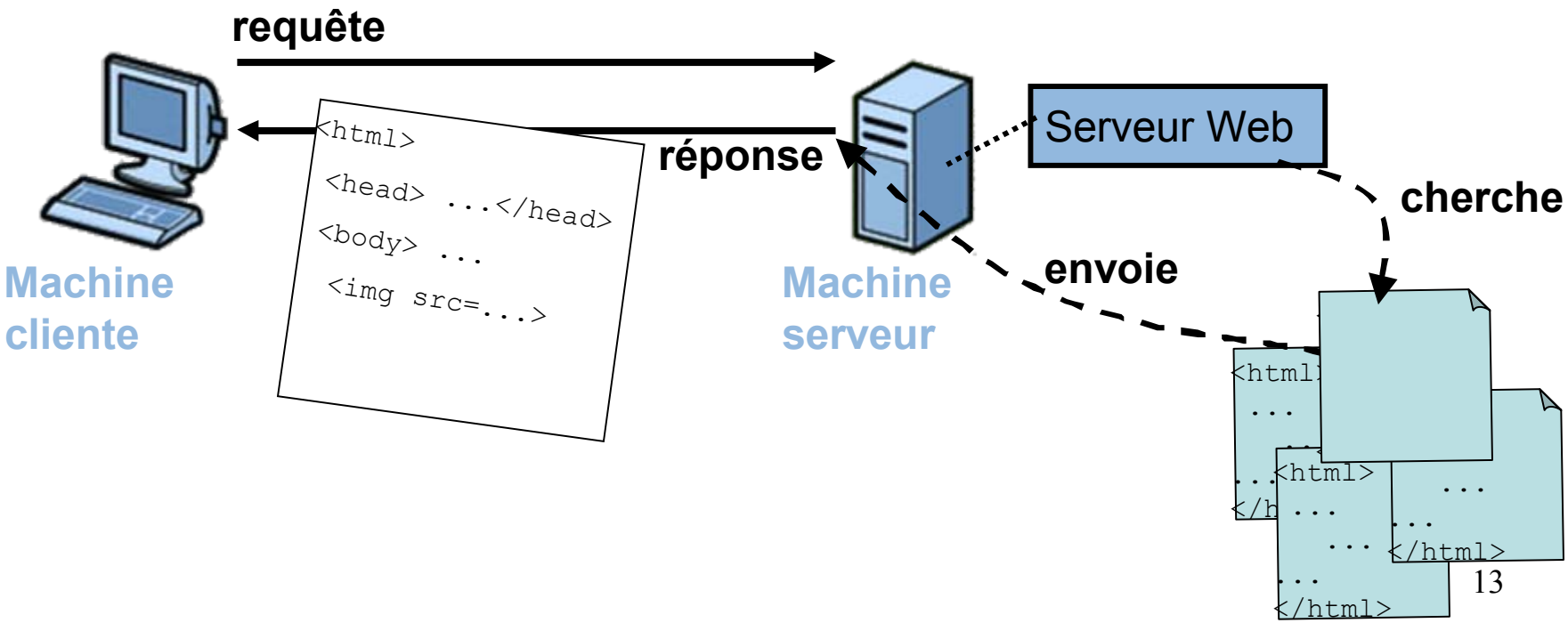
```
<html>
<!-- AREL V6.0 -->
<head>
  <title>AREL V6.0</title>
</head>
<body>
  <h1 align="center">AREL:L'école   virtuelle de l'EISTI</h1>
  <p><center>
    <img src="fondcergypau.jpg"/>
  </center></p>
  <form action="date2">
       Login: <input type="text" name="param1"/><br/>
Mot de passe: <input type="password" name="param2"/><br/>
   <center>
     <input type="submit" value="Valider"/>
   </center>
  </form>
</body>
</html>
```

Ex : AREL V6 (affichage)



Pages statiques

- Fonctionnement normal d'un serveur Web seul
  - Le serveur cherche la page dans le système de fichiers
  - La page est renvoyée au client **telle quelle**



Limites d'un serveur Web seul

- Pas de contenu dynamique

```
<html>
  <head>
    <title>Clock</title>
  </head>
  <body>
     Il est toujours 12:12.
  </body>
</html>
```

statique

VS.

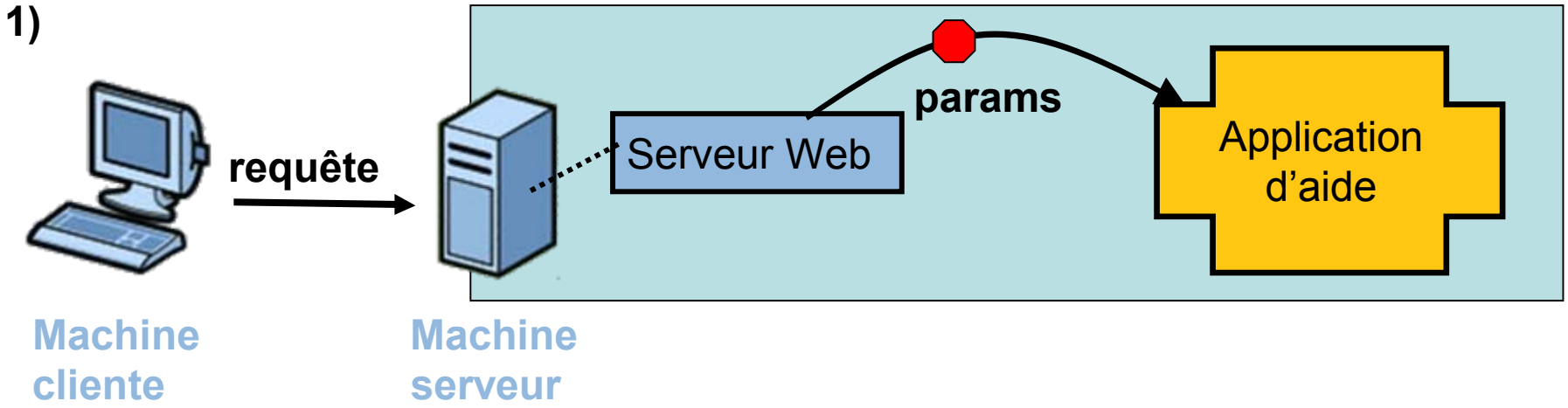
```
<html>
  <head>
    <title>Clock</title>
  </head>
  <body>
     Il est [getTimeOnServer].
  </body>
</html>
```

dynamique

- Pas de sauvegarde de données sur le serveur
  - Traitemenent de formulaires:
    besoin d'une application d'aide au serveur Web
    - pour évaluer les paramètres reçus
    - pour générer une réponse appropriée

Pages dynamiques

- Le serveur Web a besoin d'aide pour faire du dynamique
  - Autrefois, les CGI (Common Gateway Interface)
  - Aujourd'hui, un conteneur Java EE avec servlets (ex: Tomcat)



Présentation de Java EE

• Java EE est une plate-forme fortement orientée serveur pour le développement et l'exécution d'applications distribuées. Elle est composée de deux parties essentielles :

  • Un ensemble de spécifications pour une infrastructure dans laquelle s'exécute les composants écrits en Java : un tel environnement se nomme **serveur d'application**.

  • Un ensemble d'**APIs** qui peuvent être obtenues et utilisées séparément. Pour être utilisées, certaines nécessitent une implémentation de la part d'un fournisseur tiers.

Les APIs de Java EE

• Une API (*Application Programming Interface*) est une interface de programmation**.** C'est un ensemble de fonctions, procédures ou classes mises à disposition des programmes informatiques par une bibliothèque logicielle, un système d'exploitation ou un service.

• Les **composants :** permet un découpage de l'application et donc une séparation des rôles lors du développement :

- Les composants web : **Servlet** et **JSP**(Java Server Pages).
- Les composants métier : EJB (Enterprise Java Beans).

• Les **services :**

- Les **services d'infrastructures** : JDBC, JNDI, JTA, JCA, JMX
- Les **services de communication** : RMI-IIOP, JavaMail, JAAS

Architecture en Java EE

- JEE permet une grande flexibilité dans le choix de l'architecture de l'application en combinant les différents composants.

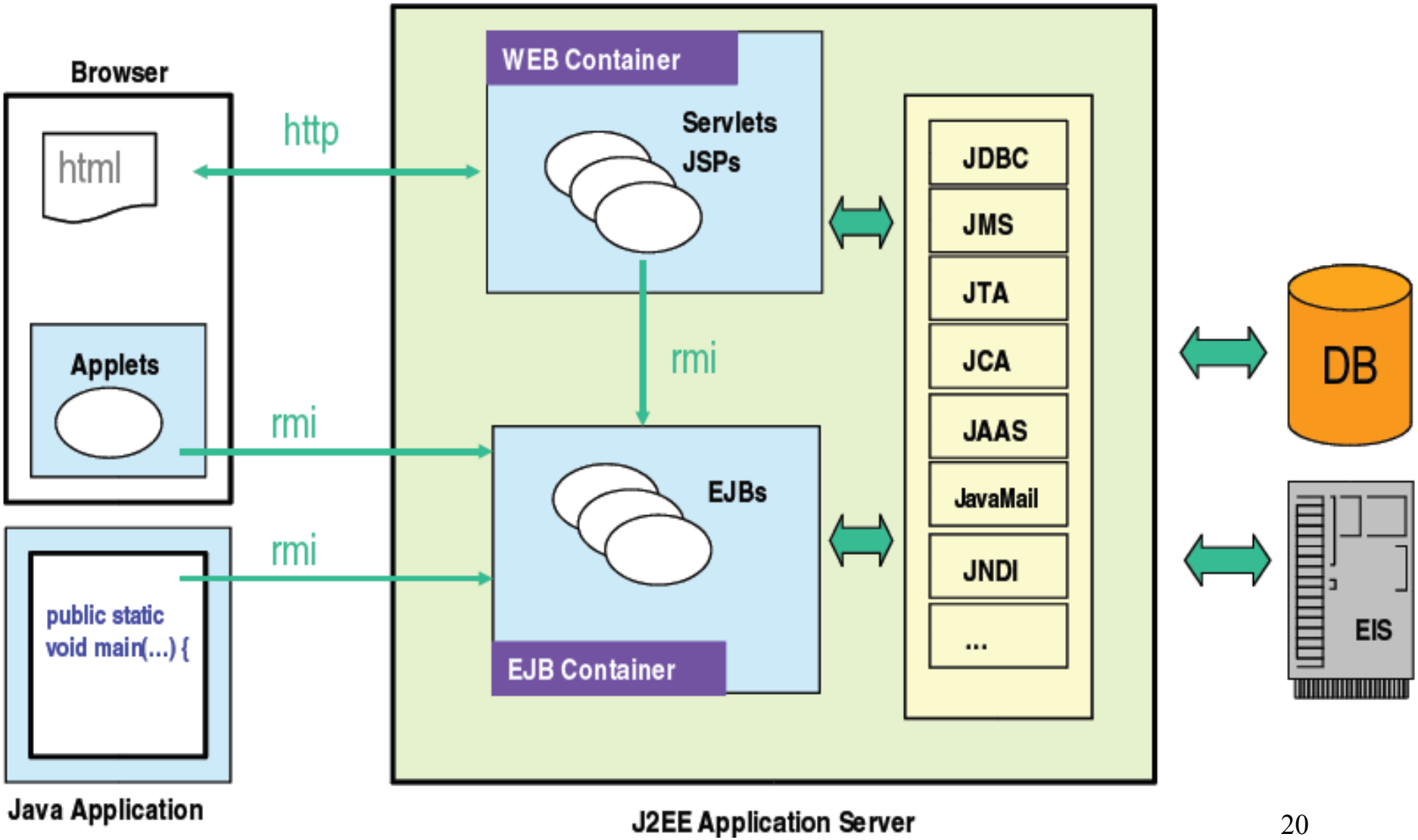
- L'architecture d'une application se découpe idéalement en au moins trois tiers :
  - **La partie cliente** : permet le dialogue avec l'utilisateur. Elle peut être composée :
    - d'une application stand-alone
    - d'une application web
    - d'applets
  - **La partie métier** : encapsule les traitements (dans des EJB ou des JavaBeans)
  - **La partie données** : stocke les données

Servlets et JSP

- Afin de réaliser des applications Web dynamique, nous réaliserons 2 grands type de « pages JEE » :
  - **Les Servlets** : qui sont des **classes Java** spécifiques pouvant être exécutées sur un serveur JEE. La méthode principale de ces classes sera appelée à chaque requête du client et recevra en paramètre la requête soumise. Après traitement (dans le corps de la méthode), elle renverra ensuite au client la page HTML générée.
  - **Les JSP** : qui ont le même but que les Servlets mais avec une syntaxe plus proche de l'HTML (comparable au PHP).

- Ces 2 types de programmation peuvent être utilisés de manière indépendante ou conjointe en fonction de l'application à réaliser.

Architecture en Java EE



Conteneur

• Les conteneurs assurent la gestion du cycle de vie des composants qui s'exécutent en eux. Les conteneurs fournissent des services qui peuvent être utilisés par les applications lors de leur exécution.

• La notion de conteneur se retrouve dans de nombreuses technologies :
  – Servlet, Applet, MIDlet, Xlet, (**-let*), EJB, …

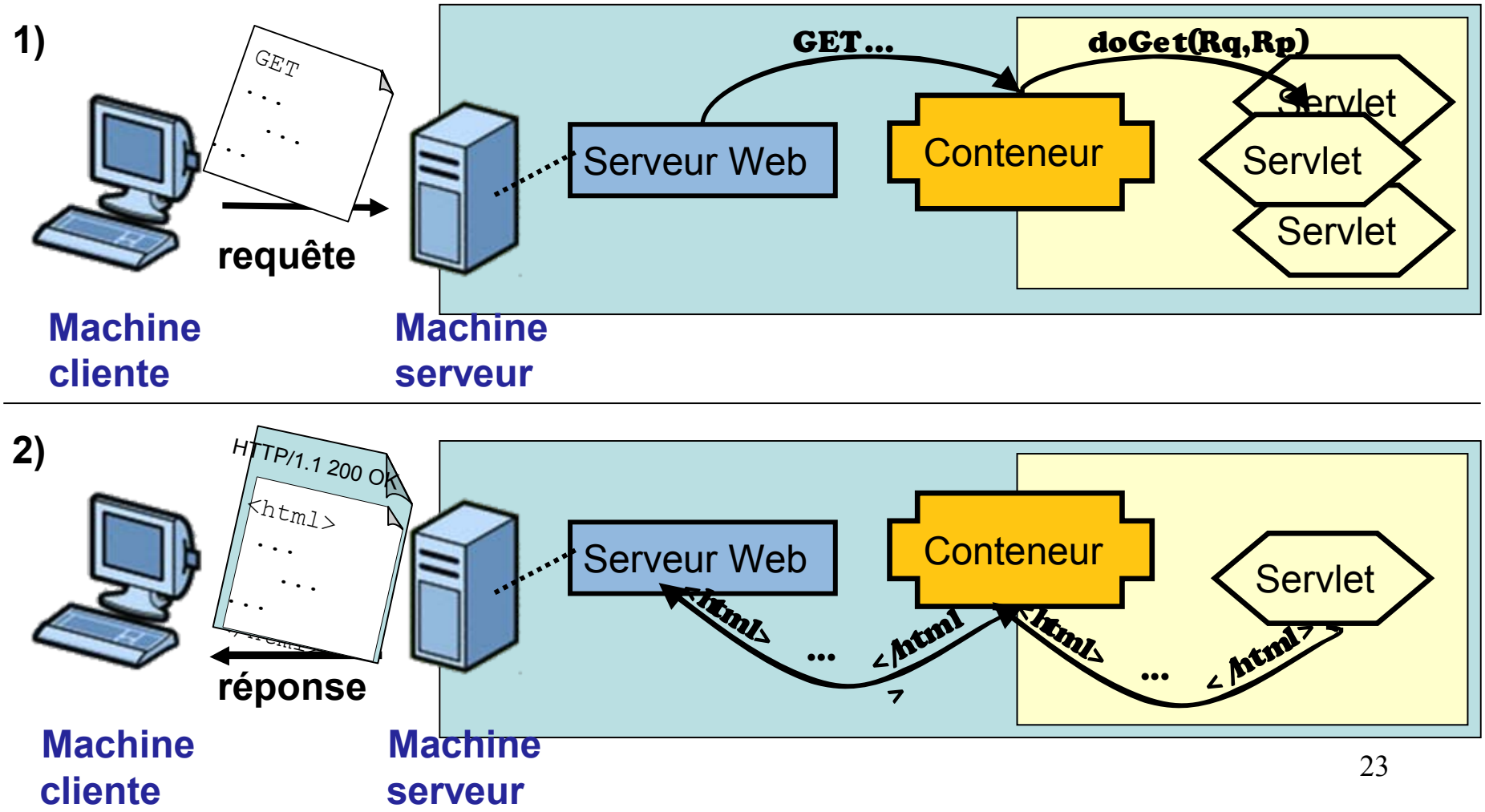
• Il existe plusieurs conteneurs définit par JEE:
  – Conteneur web : pour exécuter les Servlets et les JSP
  – Conteneur d'EJB : pour exécuter les EJB
  – Conteneur client : pour exécuter des applications stand-alone sur les postes qui utilisent des composants JEE

Conteneur

- Un conteneur est un composant logiciel *système* qui contrôle d'autres composants, dits *métier*
  - Tomcat est un exemple de conteneur
  - Les servlets n'ont pas de méthode **main()**, ils sont contrôlés par le conteneur Tomcat
  - Les requêtes ne sont pas adressées aux servlets mais au conteneur dans lequel ils sont *déployés*

Application Web avec un conteneur

- Le serveur Web a besoin d'aide pour faire du dynamique



Pourquoi un conteneur?

- Pour oublier le cours de « réseau » !
- Un conteneur fournit pour les Servlets :
  - Un support pour la communication
    - Pas besoin de ServerSocket, Socket, Stream,…
  - La gestion du cycle de vie
  - Un support pour le Multithreading
    - Création automatique des Threads
  - Un support pour la sécurité
  - Un support pour les JSP

Module Web

- Un servlet ne peut pas être déployé directement dans un conteneur, il doit faire partie d'un module Web.

- Un module Web est un ensemble de librairies, de fichiers de configurations, de code Java (bytecode des servlets…), …

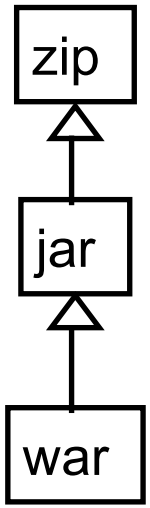
- Le module Web est l'unité de déploiement dans le conteneur.

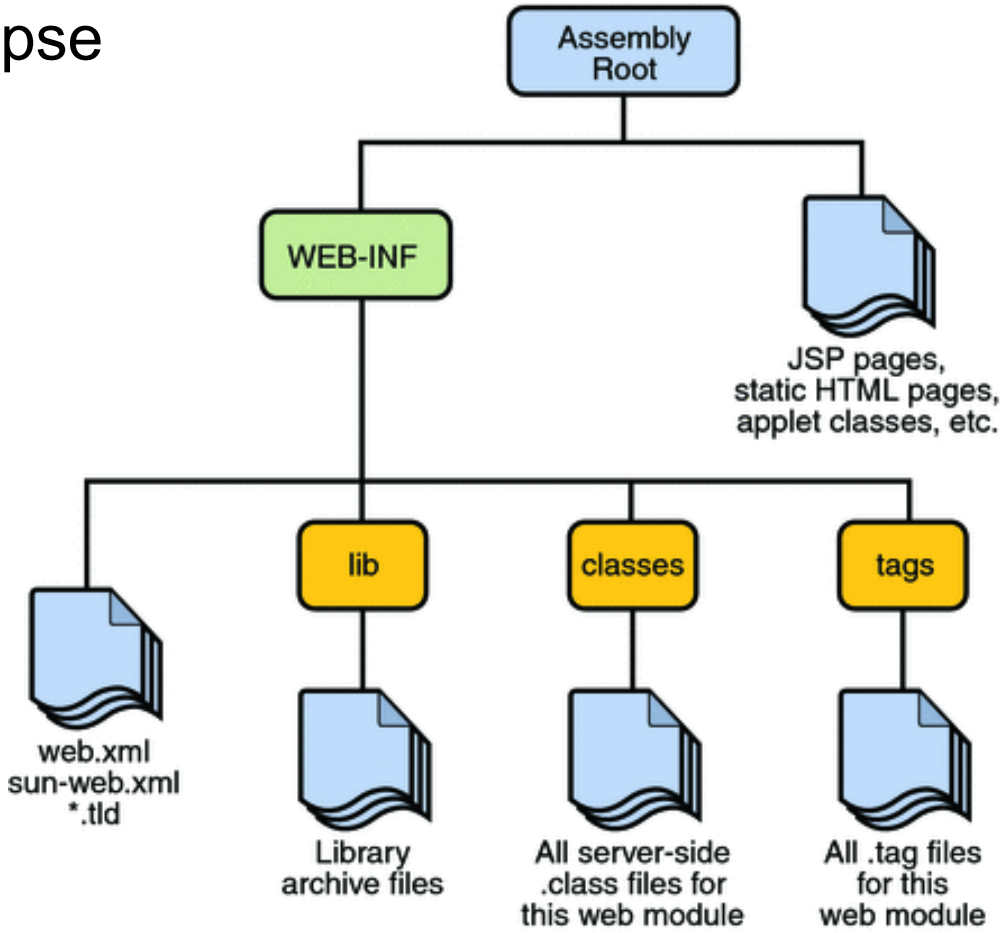
Module Web

- Pour déployer une application dans un conteneur, il faut lui fournir deux éléments :
  - L'application avec tous les composants (classes compilées, ressources ...) regroupée dans une archive ou module. Chaque conteneur possède son propre format d'archive.
  - Un fichier descripteur de déploiement contenu dans le module qui précise au conteneur des options pour exécuter l'application

Structure d'un module Web (.war)

- Automatisé dans Eclipse
  - File/ Export…
    Web/ WAR file





Les différents types d'archives

Archive / module	Contenu	Extension	Descripteur de déploiement
bibliothèque	Regroupe des classes	jar	
application client	Regroupe les ressources nécessaires à leur exécution (classes, bibliothèques, images, ...)	jar	application-client.jar
web	Regroupe les servlets et les JSP ainsi que les ressources nécessaires à leur exécution (classes, bibliothèques de balises, images, ...)	war	web.xml
EJB	Regroupe les EJB et leur composants (classes)	jar	28

Une application est un regroupement d'un ou plusieurs modules dans un fichier EAR (Entreprise ARchive). L'application est décrite dans un fichier application.xml lui même contenu dans le fichier EAR

Serveur d‘application

- Les serveurs d'applications peuvent fournir :
  - Un conteneur web uniquement (exemple : Tomcat) ou
  - Un conteneur d'EJB uniquement (exemple : JBoss, Jonas, ...) ou
  - Les deux conteneurs (exemple : Websphere, Weblogic, ...).

Les services proposés par la plate-forme JEE

Une plate-forme d'exécution JEE complète, implémentée dans un serveur d'application, propose les services suivants :

- service de nommage (naming service)
- service de déploiement (deployment service)
- service de gestion des transactions (transaction service)
- service de sécurité (security service)

Ces services sont utilisés directement ou indirectement par les conteneurs mais aussi par les composants qui s'exécutent dans les conteneurs grâce à leurs API respectives.

Environnements de développement

- Le cycle *Développement-Déploiement-Exécution* est trop complexe à votre goût?

- Les IDE sont là pour vous assister!

...