



Développement web avec Java EE



Déroulé du cours

- 23 heures de cours
- 23 heures TD/TP
- Evaluation : 2 contrôles, 1 examen + 1 mini projet

Objectifs pédagogiques :

- Comprendre les bases du Java EE
- Disposer des bases nécessaires pour utiliser la plupart des technologies se basant sur Java EE

Compétences acquises :

- Capacité à développer un projet web dynamique
- Capacité à créer une application JEE qui respecte les standards reconnus dans le domaine

Des notions en développement Java sont nécessaires

- Java
- Java OO

Des notions en technologies Web

- HTML / HTML5
- CSS / CSS3

Des notions en langage SQL

- MySQL



Plan du cours

■ **Partie I: Les bases du Java EE**

- Cours 1: Introduction au Java EE
- Cours 2: Environnement de développement et structure d'un projet JEE

■ **Partie II: Ecosystème de la technologie Java EE**

- Cours 3: Les servlets et les JavaBeans
- Cours 4: La technologie JSP
- Cours 5: La bibliothèque JSTL
- Cours 6: Gestion des cookies et sessions
- Cours 7: Gestion des fichiers

■ **Partie II: Bases de données et JSF**

- Cours 8: MySQL et JDBC
- Cours 9: Framework JSF



Les bases du Java EE

Cours 1

Serveur d'application, protocole HTTP, modèle MVC,
outil et environnement de travail





L'écosystème de la technologie Java EE

- Un nombre conséquent de technologies et d'approches existent



spring



JSF



GlassFish



Portal Server

Qu'est-ce que Java EE

- **Java SE**

Le terme « Java » fait bien évidemment référence à un langage, mais également à une plate-forme : son nom complet est « **Java SE** » pour *Java Standard Edition*, et était anciennement raccourci « *J2SE* ». Celle-ci est constituée de nombreuses bibliothèques, ou API : citons par exemple `java.lang`, `java.io`, `java.math`, `java.util`, etc. Toutes ces bibliothèques que vous devez déjà connaître et qui contiennent un nombre conséquent de classes et de méthodes prêtes à l'emploi pour effectuer toutes sortes de tâches.

- **Java EE**

Le terme « **Java EE** » signifie *Java Enterprise Edition*, et était anciennement raccourci en « *J2EE* ». Il fait quant à lui référence à une extension de la plate-forme standard. Autrement dit, la plate-forme *Java EE* est construite sur le langage Java et la plate-forme Java SE, et elle y ajoute un grand nombre de bibliothèques remplissant tout un tas de fonctionnalités que la plate-forme standard ne remplit pas d'origine.



Qu'est-ce que Java EE

- **Objectif du JEE**

L'objectif majeur de Java EE est de faciliter le développement d'applications web robustes et distribuées, déployées et exécutées sur un serveur d'applications.

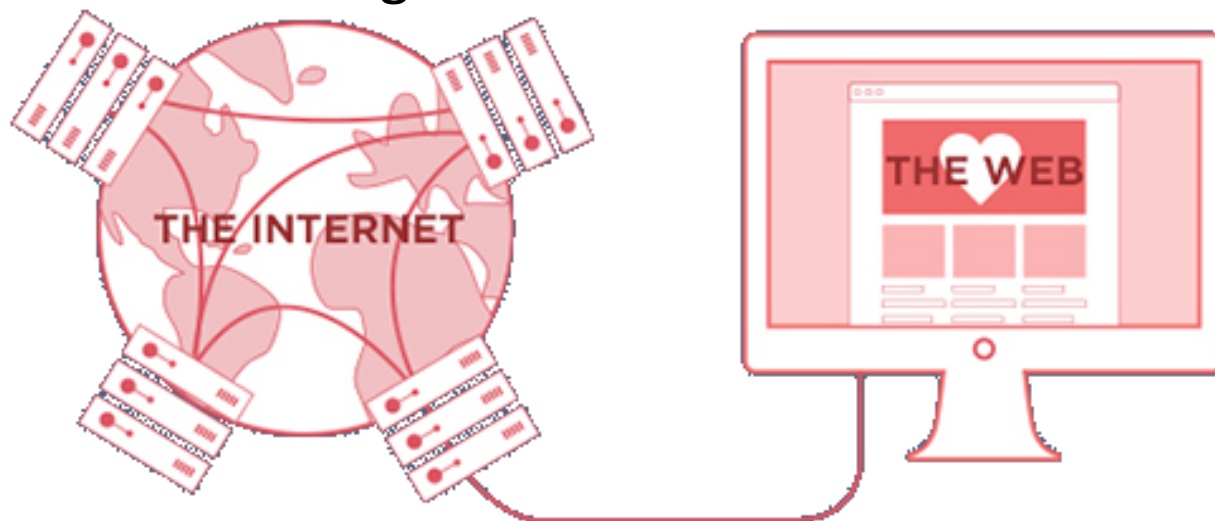
- **Java EE n'est pas Javascript**

S'il est vrai que Java EE permet la création d'applications web, il ne faut pas pour autant le confondre avec le langage Javascript, souvent raccourci en « JS », qui est lui aussi massivement utilisé dans les applications web. Ce sont là deux langages totalement différents, qui n'ont comme ressemblance que leur nom

Internet n'est pas le web

- **Ne pas confondre internet et web**

- L'internet est le réseau, le support physique de l'information. C'est un ensemble de machines, de câbles et d'éléments réseau en tout genre éparpillés sur la surface du globe.
- Le web constitue une partie seulement du contenu accessible sur l'internet. Il existe d'autres contenus, comme le courrier électronique ou encore la messagerie instantanée.



Internet désigne le réseau physique.

Le web désigne le contenu accessible à travers ce réseau.

Les sites internet

- **Site web**

Un site web est un ensemble constitué de pages web (elles-mêmes faites de fichiers HTML, CSS, Javascript, etc.). Lorsqu'on développe puis publie un site web, on met en réalité en ligne du contenu sur internet. On distingue deux types de sites internet: les sites statiques et les sites dynamiques

- **Site internet statique**

Les sites internet statiques : ce sont des sites dont le contenu est « fixe », il n'est modifiable que par le propriétaire du site. Ils sont réalisés à l'aide des technologies HTML, CSS et Javascript uniquement.

- **Site internet dynamique**

Les sites internet dynamiques : ce sont des sites dont le contenu est « dynamique », parce que le propriétaire n'est plus le seul à pouvoir le faire changer ! En plus des langages précédemment cités, ils font intervenir d'autres technologies : Java EE est l'une d'entre elles !

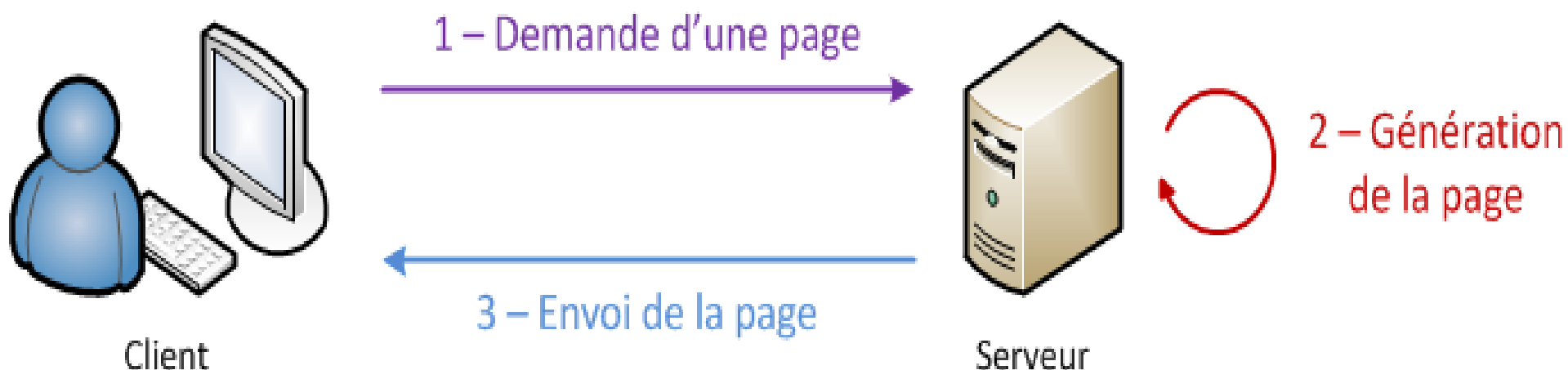
Interactions client-serveur

- **Le client**

Dans la plupart des cas, c'est le navigateur installé sur l'ordinateur. Retenez que ce n'est pas le seul moyen d'accéder au web, mais c'est celui qui nous intéresse dans ce cours.

- **Le serveur**

C'est la machine sur laquelle le site est hébergé, où les fichiers sont stockés et les pages web générées.



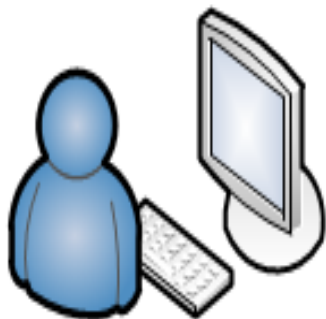
Le protocole HTTP

- **HTTP (HyperText Transfer Protocol)**

La communication qui s'effectue entre le client et le serveur est régie par des règles bien définies du **protocole HTTP**

1. l'utilisateur saisit une URL dans la barre d'adresses de son navigateur
2. le navigateur envoie alors **une requête HTTP** au serveur pour lui demander la page correspondante
3. le serveur reçoit cette requête, l'interprète renvoie une réponse au client par le biais d'une **réponse HTTP** ;
4. le navigateur reçoit, via cette réponse, la page web finale, qu'il affiche à l'utilisateur.

1 - Le client saisit une URL



Client

2 – Le navigateur envoie une
requête HTTP au serveur



Serveur



3 – Le serveur traite la requête et
génère la page web demandée

4 – Le serveur renvoie une
réponse HTTP au client



Les langages du web

Client (front-end)

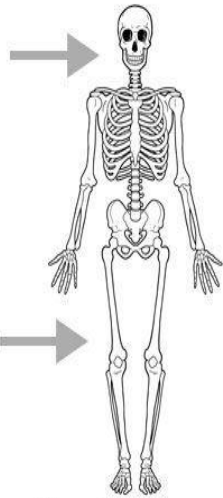
Interprété par le navigateur

HTML



HTML

HEAD



BODY

Structural Layer

CSS



HTML with CSS

← APPEARANCE



← LOOK

Presentation Layer tutorial.techaltum.com

Serveur (back-end)

Interprété par le serveur



Comment choisir la technologie la mieux adaptée à son projet ?

- **Votre Expérience**

Si vous avez déjà développé en Java, Python ou C# auparavant, il semble prudent de vous orienter respectivement vers Java EE, Django et .NET ;.

- **Les besoins du cahier de charge**

Rapidité de développement, faible utilisation des ressources sur le serveur, réactivité de la communauté soutenant la technologie, ampleur de la documentation disponible en ligne, coût, etc

- **Java EE une technologie fiable et adaptée aux besoins du marché**

- Java EE est une extension de la plate-forme standard Java SE, principalement destinée au développement d'applications web
- Pour interagir avec un site web (le serveur), l'utilisateur (le client) passe par son navigateur.
- À travers le protocole HTTP, le navigateur envoie des requêtes au serveur et le serveur lui renvoie des réponses

Le serveur d'applications

- **Serveur d'applications = serveur HTTP + conteneur**

Ce composant se charge d'exécuter le code en plus de faire le travail du serveur HTTP.

- **Serveur HTTP**

C'est une interface de communication avec le protocole HTTP. Il écoute tout ce qui arrive sur le port utilisé par le protocole HTTP, le **port 80**, et scrute chaque requête entrante. Apache HTTP Server et IIS (Microsoft) sont les deux plus connus.

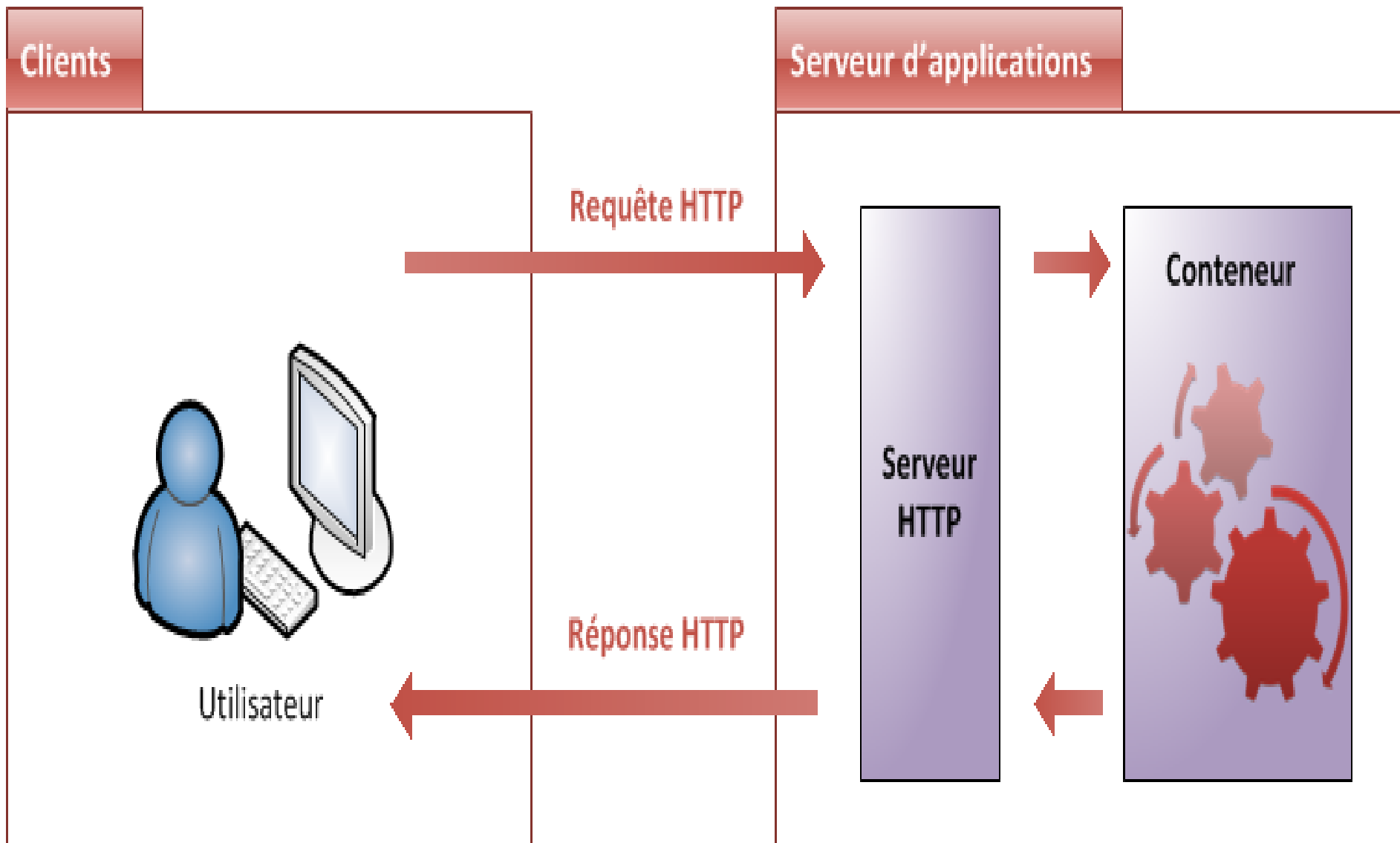


- **Le conteneur**

C'est un composant qui analyse et exécute le contenu du code et y ajoute la gestion d'objets de diverses natures.

Architecture serveur d'applications

- Architecture 2-tiers (Client - Serveur)



Le rôle du serveur d'applications

- **Le serveur d'applications est multifonctionnels**

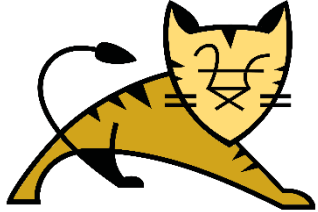
- Il récupère les requêtes HTTP issues des clients.
- Il les met dans des boîtes, des objets, que votre code sera capable de manipuler.
- Il fait passer ces objets dans l'application, via le conteneur.
- Il renvoie les réponses HTTP aux clients, en se basant sur les objets retournés par votre code.

- **Les solutions existants**

ORACLE[®]
WEBLOGIC SERVER



GlassFish



Apache Tomcat

IBM[®]
WebSphere.



JBoss[®]
by Red Hat



JOnAS

- **Le MVC**

En anglais **design pattern**, un modèle de conception (ou encore patron de conception) est une simple **bonne pratique**, qui répond à un problème de conception d'une application. C'est en quelque sorte une ligne de conduite qui permet de décrire les grandes lignes d'une solution. Le MVC est issu de l'expérience des concepteurs et développeur d'applications

- **Que recommandent les développeurs Java EE expérimentés ?**

Il est recommandé d'adopté le modèle MVC pour les raisons suivantes:

- Que l'on puisse être amené à travailler à plusieurs contributeurs sur un même projet ou une même application (travail en équipe).
- Que l'on puisse être amené à maintenir et corriger une application que l'on n'a pas créée soi-même.
- Que l'on puisse être amené à faire évoluer une application que l'on n'a pas créée soi-même.

Structure du MVC

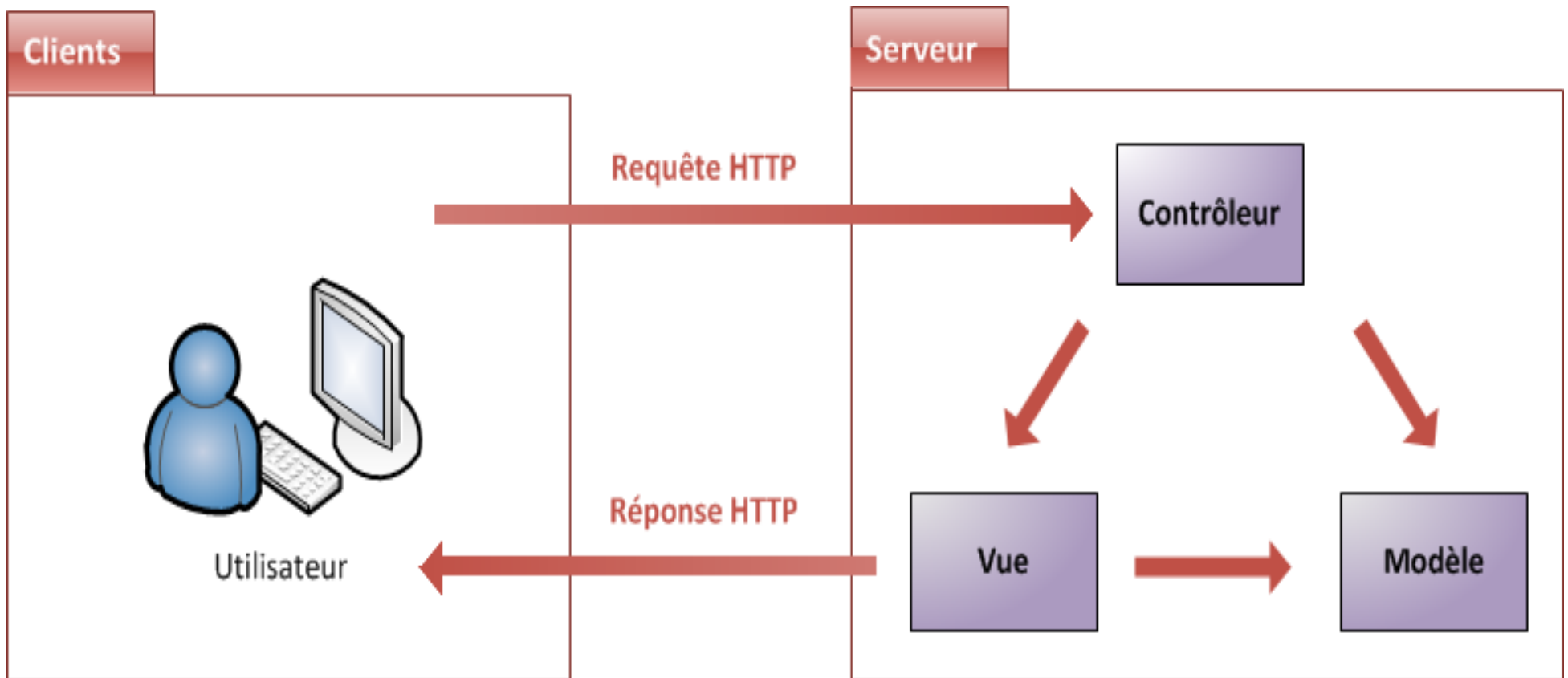
- **Le modèle MVC (Modèle-Vue-Contrôleur)**

Il découpe littéralement l'application en couches distinctes, et de ce fait impacte très fortement l'organisation du code.

- tout ce qui concerne le traitement, le stockage et la mise à jour des données de l'application doit être contenu dans la couche nommée "**Modèle**" (le M de MVC) ;
- tout ce qui concerne l'interaction avec l'utilisateur et la présentation des données (mise en forme, affichage) doit être contenu dans la couche nommée "**Vue**" (le V de MVC) ;
- tout ce qui concerne le contrôle des actions de l'utilisateur et des données doit être contenu dans la couche nommée "**Contrôle**" (le C de MVC).

Les composants du MVC

- Schéma conceptuel du MVC



- **Modèle: des traitements et données**

Il découpe littéralement l'application en couches distinctes, et de ce fait impacte très fortement l'organisation du code. Dans le modèle, on trouve à la fois les données et les traitements à appliquer à ces données. Ce bloc contient donc des objets Java d'une part, qui peuvent contenir des attributs (**données**) et des méthodes (**traitements**) qui leur sont propres, et un système capable de stocker des données d'autre part. Rien de bien transcendant ici, la complexité du code dépendra bien évidemment de la complexité des traitements à effectuer par votre application.



Java EE et MVC (2/3)

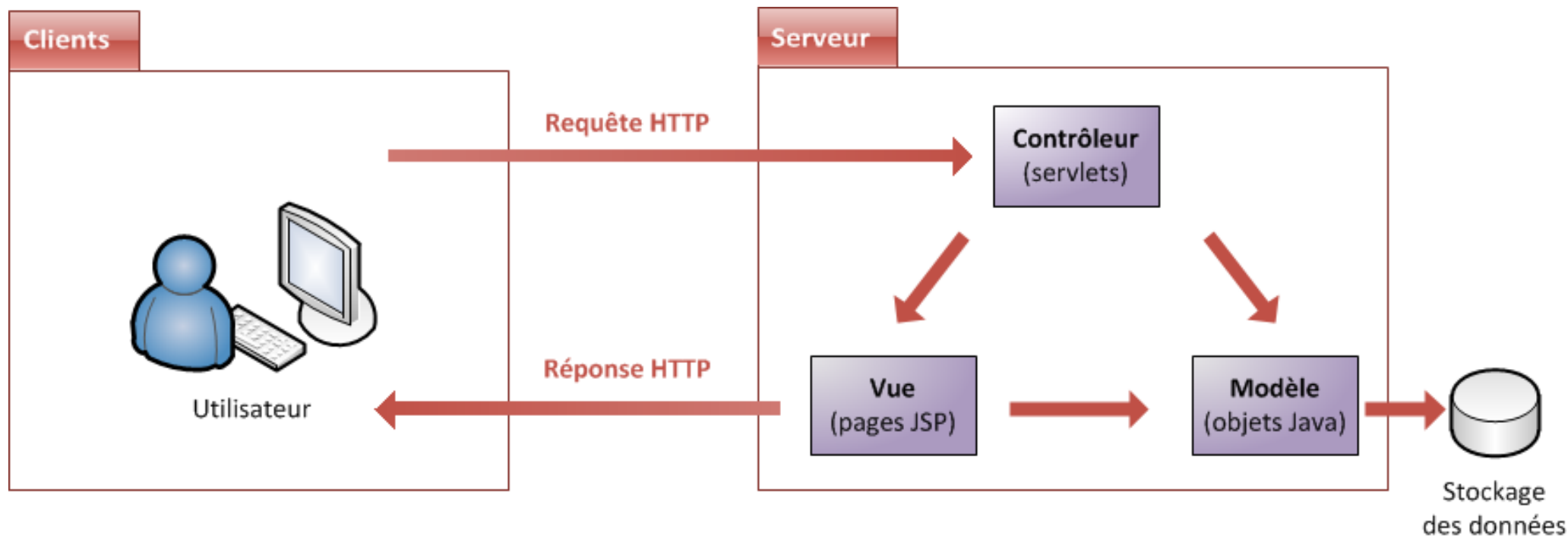
- **Vue: les pages JSP (Java Server Page)**

Une page JSP est destinée à la vue. Elle est exécutée côté serveur et permet l'écriture de gabarits (pages en langage "client" comme HTML, CSS, Javascript, XML, etc.). Elle permet au concepteur de la page d'appeler de manière transparente des portions de code Java, via des balises et expressions ressemblant fortement aux balises de présentation HTML.

- **Contrôleur: des Servlets**

Une servlet est un objet qui permet d'intercepter les requêtes faites par un client, et qui peut personnaliser une réponse en conséquence. Il fournit pour cela des méthodes permettant de scruter les requêtes HTTP. Cet objet n'agit jamais directement sur les données, il faut le voir comme un simple aiguilleur : il intercepte une requête issue d'un client, appelle éventuellement des traitements effectués par le modèle, et ordonne en retour à la vue d'afficher le résultat au client.

- MVC avec Java EE



- Dans une application Java EE sans Frameworks

- la couche **M**odèle est constituée d'objets Java ;
- la couche **V**ue est constituée de pages JSP ;
- la couche **C**ontrôle est constituée de servlets.

Résumé

- ❑ Java EE est une extension de la plate-forme standard Java SE, principalement destinée au développement d'applications web.
- ❑ Internet désigne le réseau physique ; le web désigne le contenu accessible à travers ce réseau.
- ❑ Pour interagir avec un site web (le serveur), l'utilisateur (le client) passe par son navigateur.
- ❑ À travers le protocole HTTP, le navigateur envoie des requêtes au serveur et le serveur lui renvoie des réponses :
 1. le travail du serveur est de recevoir des requêtes, de générer les pages web et de les envoyer au client.
 2. le travail du navigateur est de transmettre les actions de l'utilisateur au serveur, et d'afficher les informations qu'il renvoie.



QUESTIONS ?