Formation Java EE

Web - Servlet

digitalent FORMATION

Introduction au Web

HTTP ?

HTTP = « HyperText Transfert Protocol »

Protocole de communication basé sur TCP/IP utilisé pour le Web

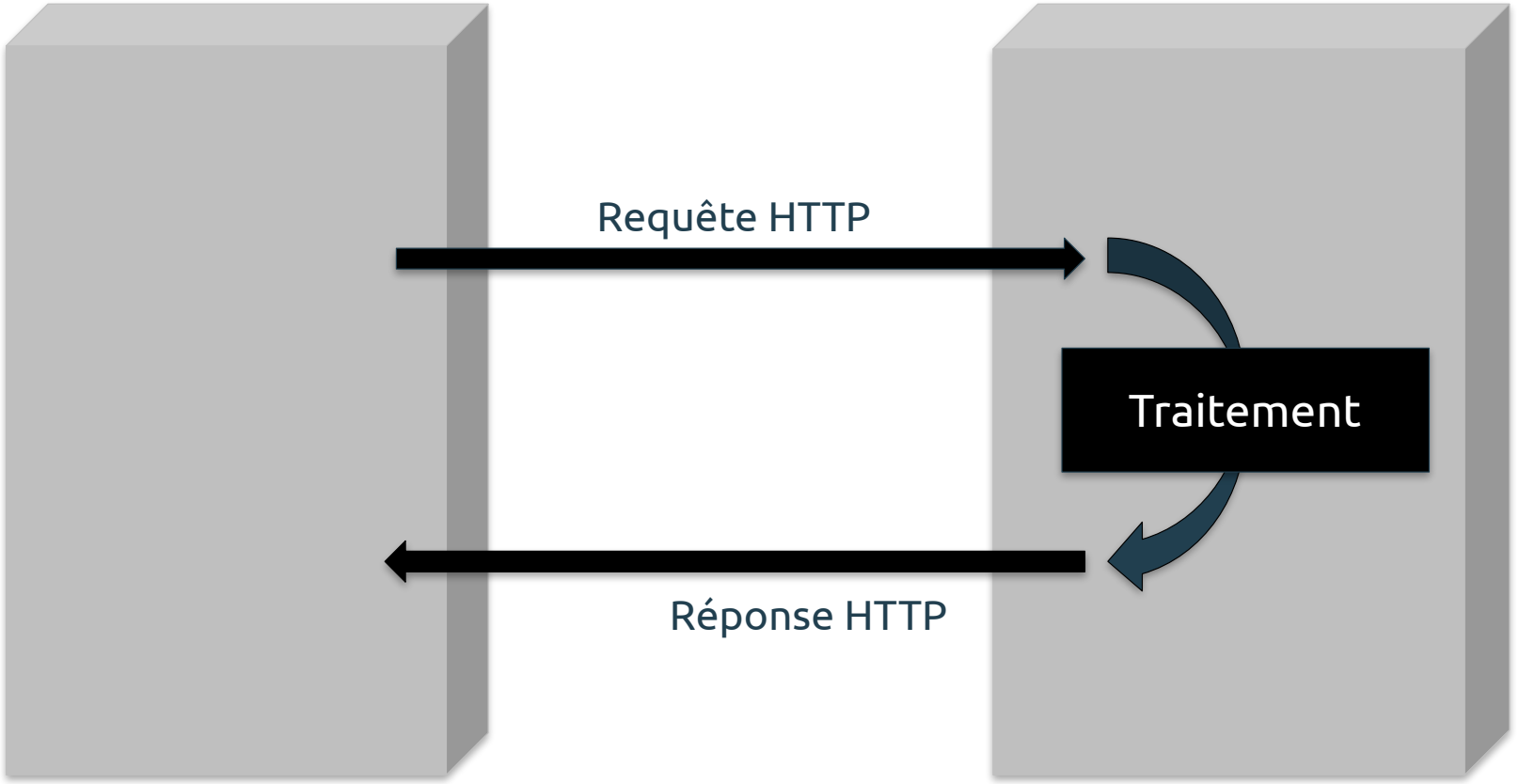
HTTP ne maintient pas de connexion entre un client et un serveur

HTTP est sans état (« stateless »)

HTTP est indépendant du type de données transportées

HTTP est synchrone => 1 requête attend 1 réponse

Communication HTTP



Machine dite « cliente »

Machine dite « serveur »

Requête HTTP

Format d'une requête HTTP

[METHODE] [URI] HTTP/[VERSION]
[CHAMPS ENTETE au format « CLE:VALEUR »]
Ligne blanche
[CORPS DE LA REQUETE (optionnel)]

Méthodes possibles

- GET, POST, HEAD, PUT, DELETE, OPTIONS, PATCH, TRACE,...

Exemple

GET http://google.fr/ **HTTP/1.1**
Accept:text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding:gzip, deflate, sdch
Accept-Language:en-US,en;q=0.8,fr;q=0.6
Connection:keep-alive
Host:clever-institut.com
User-Agent:Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/40.0.2214.115 Safari/537.36

Réponse HTTP

Format d'une requête HTTP

HTTP / [VERSION] [STATUT] [COMMENTAIRE STATUT]
Content-Type: [TYPE MIME DU CONTENU]
[CHAMPS ENTETE au format « CLE:VALEUR »]
Ligne blanche
[Document]

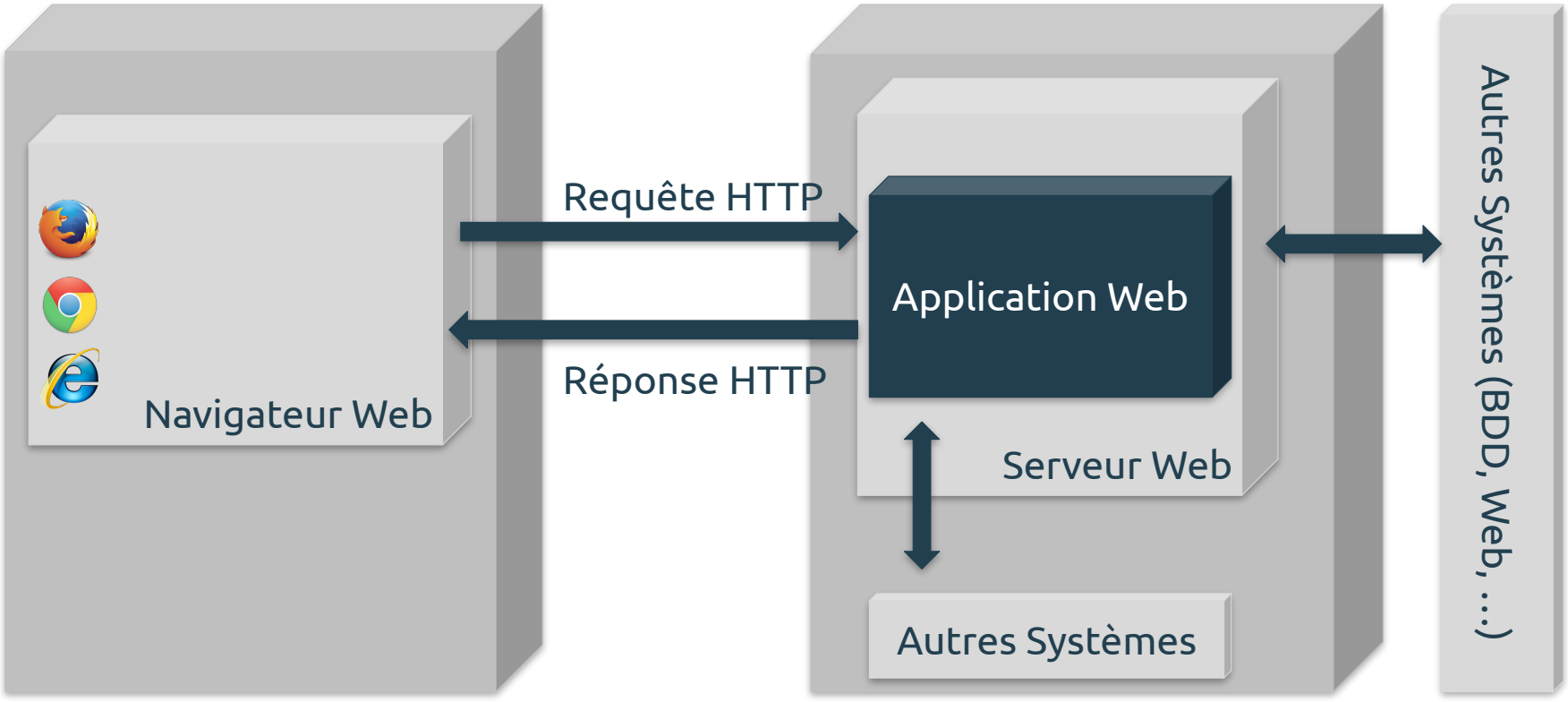
Statut d'une réponse

- 100 – 199 : Information
  100 : Continue, 101 : Switching Protocols, …
- 200 – 299 : Succès
  200 : OK, 201 : Created, 202 : Accepted, …
- 300 – 399 : Redirection
  301 : Moved Permanently, 304 : Not Modified, …
- 400 – 499 : Erreur dans la requête
  400 : Bad Request, 404 : Not Found, …
- 500 – 599 : Erreur côté serveur
  - 500 : Internal Server Error, 503 : Service Unavailable, …

HTTP/1.1 **200 OK**
Connection:Keep-Alive
Content-Encoding:gzip
Content-Language:en-US
Content-Length:6049
Content-Type:text/html;charset=UTF-8
Date:Sat, 21 Feb 2015 20:15:50 GMT
Keep-Alive:timeout=15, max=100
Server:Apache
Vary:Accept-Encoding

<!DOCTYPE html>
<html>

Application Web

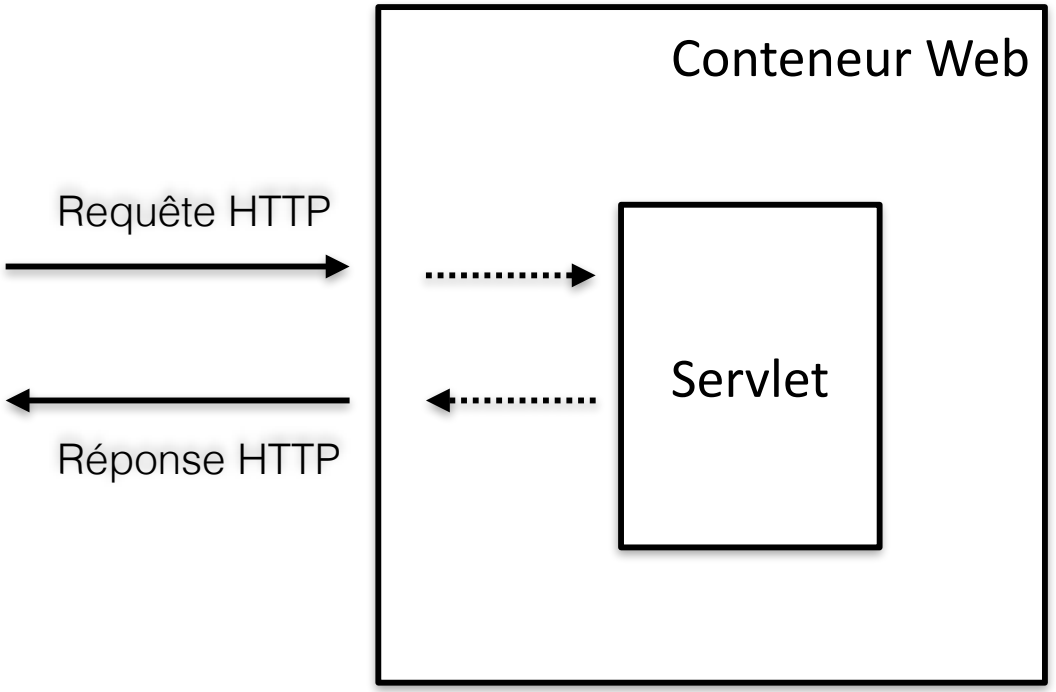


Servlet

Servlet

Servlet ?

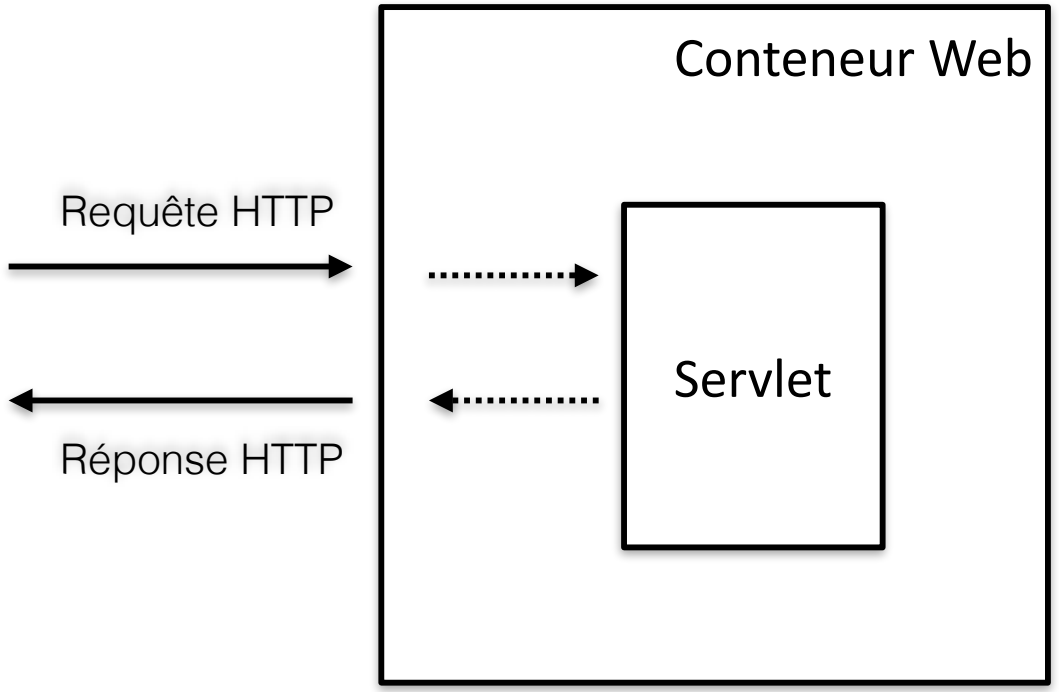
- Composant qui étend les fonctionnalités d'une serveur Web.
- Exécuté à l'intérieur d'un conteneur Web
- Reçoit des requêtes HTTP
- Génère une réponse



Servlet

Via des APIs permet la gestion

- des informations de requête HTTP
- de la construction de la réponse HTTP
- des cookies
- …



Servlet

javax.servlet.Servlet

- Une servlet doit implémenter l'interface java.servlet.Servlet soit directement soit en héritant d'une classe qui implémente cette interface

```
public class MaServlet implements Servlet {

  @Override
  public void init(ServletConfig config)
                              throws ServletException {
    // phase d'initialisation

  }

  @Override
  public void service(ServletRequest requete,
                                  ServletResponse reponse)
        throws ServletException, IOException {
    // phase de traitement des requêtes
  }

  @Override
  public void destroy() {
    // phase d'arrêt
  }

}
```

Servlet

Servlet Web

- Etend la classe java.servlet.http.HttpServlet
- Possède des méthodes spécifiques au protocole HTTP qui remplacent la méthode service()
  - doGet() —> GET
  - dotPost() —> POST
  - doPut() —> PUT
  - doDelete() —> DELETE
  - doHead() —> HEAD
  - doOptions() —> OPTIONS
  - doTrace() —> TRACE

```
public class MaServletWeb extends HttpServlet{

   @Override
   protected void doGet(HttpServletRequest req,
          HttpServletResponse resp)
          throws ServletException, IOException {
     // Traitement d'une requête avec la méthode GET
   }

   @Override
   protected void doHead(HttpServletRequest req,
          HttpServletResponse resp)
          throws ServletException, IOException {
     // Traitement d'une requête avec la méthode HEAD
   }

}
```

Servlet

javax.servlet.http.HttpServletRequest

- Chacune des méthodes doXXX() possède en paramètre un objet de type HttpServletRequest
- Cet objet contient les informations de la requête HTTP reçu par la servlet
  - Par exemple, la méthode getParameter() permet de récupérer les paramètres de la requête HTTP

```
public class MaServletWeb extends HttpServlet {

   @Override
   protected void doGet(HttpServletRequest req,
          HttpServletResponse resp)
          throws ServletException, IOException {
      // Récupération d'un paramètre d'une requête HTTP
      // http://monsite.com?langue=FR
      String langueParam = req.getParameter("langue");
   }

}
```

Servlet

javax.servlet.http.HttpServletResponse

- Chacune des méthodes doXXX() possède en paramètre un objet de type HttpServletResponse
- Cet objet contient le flux de sortie de la réponse HTTP
- La méthode getWriter() permet de récupérer le flux de sortie

```
public class MaServletWeb extends HttpServlet {

  @Override
  protected void doGet(HttpServletRequest req,
        HttpServletResponse resp)
        throws ServletException, IOException {

    // Récupérer le flux de sortie
    PrintWriter out = resp.getWriter();
    // Ecrire dans le flux de sortie
    out.write("Bonjour, je suis la servlet !");

  }

}
```

Servlet

Cookies

- La classe javax.servlet.http.Cookie permet de créer un Cookie
- La méthode **HttpServletRequest.getCookies**() retourne les cookies de la requête HTTP
- Le cookie créé est configurable via les méthodes :
  - **setName()** —> Nom du cookie
  - **setValue()** —> Valeur du cookie
  - **setMaxAge()** —> délai en seconde avant expiration du cookie
  - **setPath()** —> répertoire où s'applique le cookie
  - **setComment()** —> un commentaire

```
public class MaServletWeb extends HttpServlet {

   @Override
   protected void doGet(HttpServletRequest req,
         HttpServletResponse resp)
         throws ServletException, IOException {

      // récupérer les cookies présents dans la requête
      Cookie[] cookies = req.getCookies();

      // Création d'un objet Cookie
      Cookie unCookie = new Cookie("souvienstoidemoi", "oui");

      // ajout du cookie dans la réponse
      resp.addCookie(unCookie);

   }

}
```

Servlet

HttpSession

objet qui a une durée de vie d'une session de l'utilisateur

- Une API de gestion de session est mise à disposition
- Elle s'articule autour d'un objet de type java.servlet.http.HttpSession

```
public class MaServletWeb extends HttpServlet {

  @Override
  protected void doGet(HttpServletRequest req,
        HttpServletResponse resp)
        throws ServletException, IOException {
    // Récupérer la session existante ou création d'une session
    HttpSession session = req.getSession(true);

    // Stocker un utilisateur
    session.setAttribute("utilisateur", "rossi.oddet");

    // Récupérer une valeur stockée
    Object utilisateur = session.getAttribute("utilisateur");

    // Invalider une session
    session.invalidate();

    // Durée d'inactivité avant invalidation de la session
    session.setMaxInactiveInterval(10000);

    // Identifiant de la session
    String sessionID = session.getId();

  }
}
```

Servlet

Déploiement

- Le packaging WAR (Web Archive) permet de déployer une application dans un serveur d'application.
- Structure d'un WAR
  - **/WEB-INF :** Répertoire interne non exposé par le serveur Web.
  - **/WEB-INF/web.xml :** le fichier de configuration de l'application Web.
  - **/WEB-INF/classes** : les classes compilées de l'application Web.
  - **/WEB-INF/lib** : les librairies (fichiers JAR) utilisées.
  - **/** : en dehors du répertoire WEB-INF, toutes les ressources sont exposées par le serveur web

Tout est exposé sauf Web-inf--> si on veut deposé quelques choses qui ne doit pas s'excécuté c est sera a l'intérieur

Servlet

web.xml

- Configure l'application Web
- Exemple d'informations fournies au serveur d'application

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_3_1.xsd" version="“3.1">

  <display-name>Application Pizzeria</display-name>
  <description>
  Une application Web de gestion d'une pizzeria
  </description>

</web-app>
```

Servlet

web.xml

- Configurer la durée d'inactivité d'une session

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://
xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_3_1.xsd" version="3.1">
…
  <session-config>
    <session-timeout>600</session-timeout>
  </session-config>
…
</web-app>
```

Servlet

web.xml

- Mapping URL <-> Servlet

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/
javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/
web-app_3_1.xsd" version="3.1">
…
  <servlet>
    <servlet-name>MaServlet</servlet-name>
    <servlet-class>fr.test.MaServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>MaServlet</servlet-name>
    <url-pattern>/urlMaServlet</url-pattern>
  </servlet-mapping>
…
</web-app>
```

Servlet

web.xml

- Page d'accueil de l'application

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://
xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd" version="3.1">
…
 <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
  </welcome-file-list>
…
</web-app>
```

Travaux Pratiques

TP - Introduction aux servlets