

Le développement Web côté serveur avec Java EE

Module 2 – Les servlets



Objectifs

- Comprendre le rôle d'une servlet
- Comprendre le cycle de vie d'une servlet
- Savoir exploiter une requête HTTP
- Savoir générer une réponse HTTP

Les servlets

La spécification

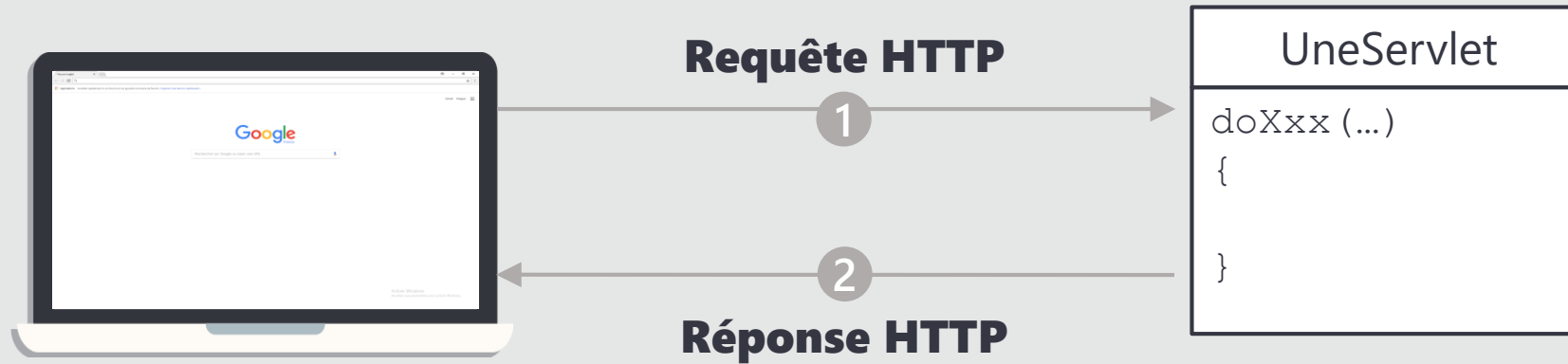
Java Servlet 3.1



JSR 340

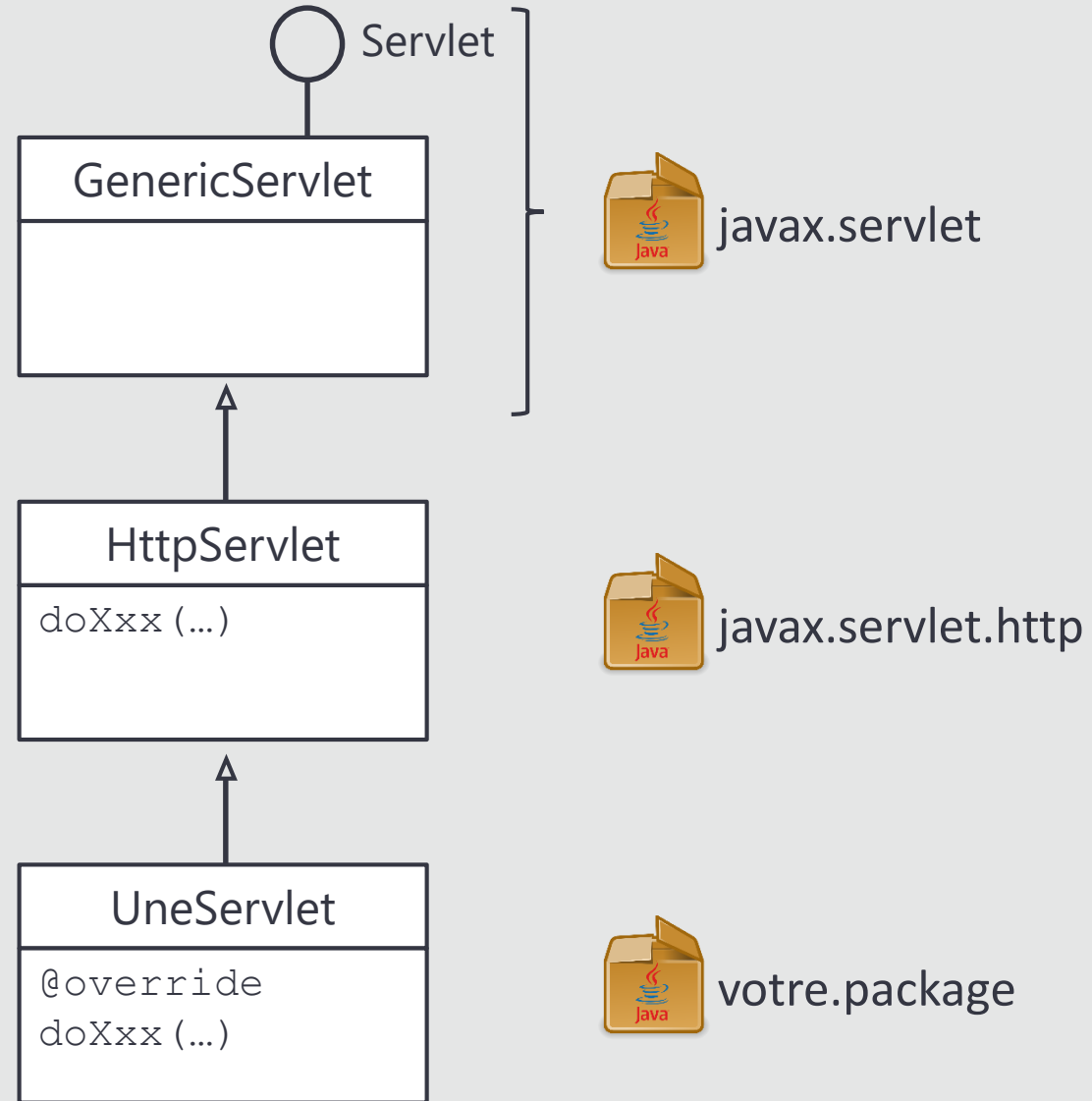
Les servlets

Qu'est-ce que c'est ?



Les servlets

Modèle objet



La création d'une servlet

```
public class UneServlet extends HttpServlet
{
    @Override
    protected void doGet(...) throws ServletException, IOException
    {
        //Générer la réponse à une requête de type GET
    }
    @Override
    protected void doPost(...) throws ServletException, IOException
    {
        //Générer la réponse à une requête de type POST
    }
}
```

Les servlets

Le paramétrage de l'URL



web.xml



web.xml

|
Servlet 3.0



Le paramétrage dans le web.xml

```
<!-- Déclaration d'une servlet -->
<servlet>
  <servlet-name>UneServlet</servlet-name>
  <servlet-class>fr.eni.demo.servlets.UnesServlet</servlet-class>
</servlet>

<!-- Association de la servlet à une ou des URL -->
<servlet-mapping>
  <servlet-name>UneServlet</servlet-name>
  <url-pattern>/url/de/la/servlet</url-pattern>*
  <url-pattern>...</url-pattern>
</servlet-mapping>
```



* utilisation du caractère joker

```
<url-pattern>/debut/url/*</url-pattern>
```


Le paramétrage par annotation

```
@WebServlet
(
    name="UneServlet",
    urlPatterns={"/url/de/la/servlet","..."}
)
public class UneServlet extends HttpServlet
{
    ...
}

@WebServlet("/debut/url/*")
public class UneServlet extends HttpServlet
{
    ...
}
```

Les servlets

La première servlet

Démonstration



Les servlets

Installation de la Javadoc

Démonstration

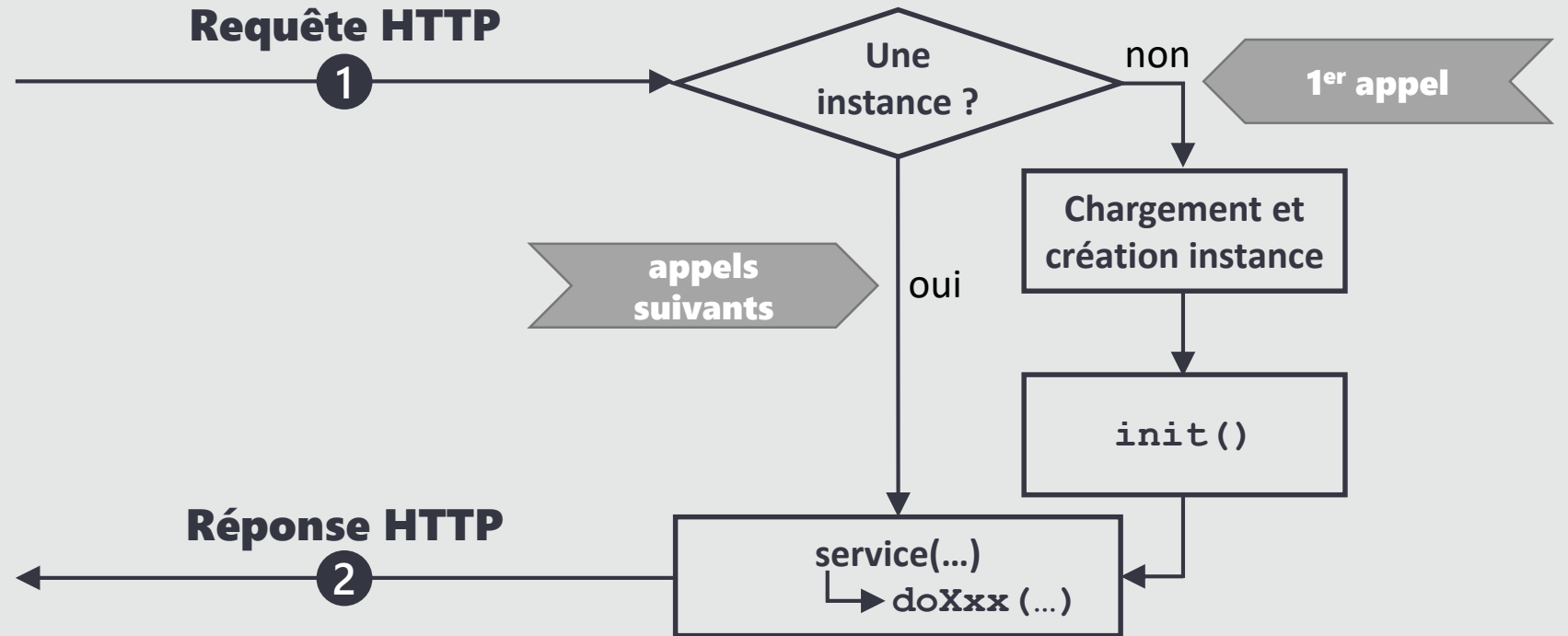


Le cycle de vie d'une servlet

Gérer par le
conteneur web

une seule
instance

Pool de
threads



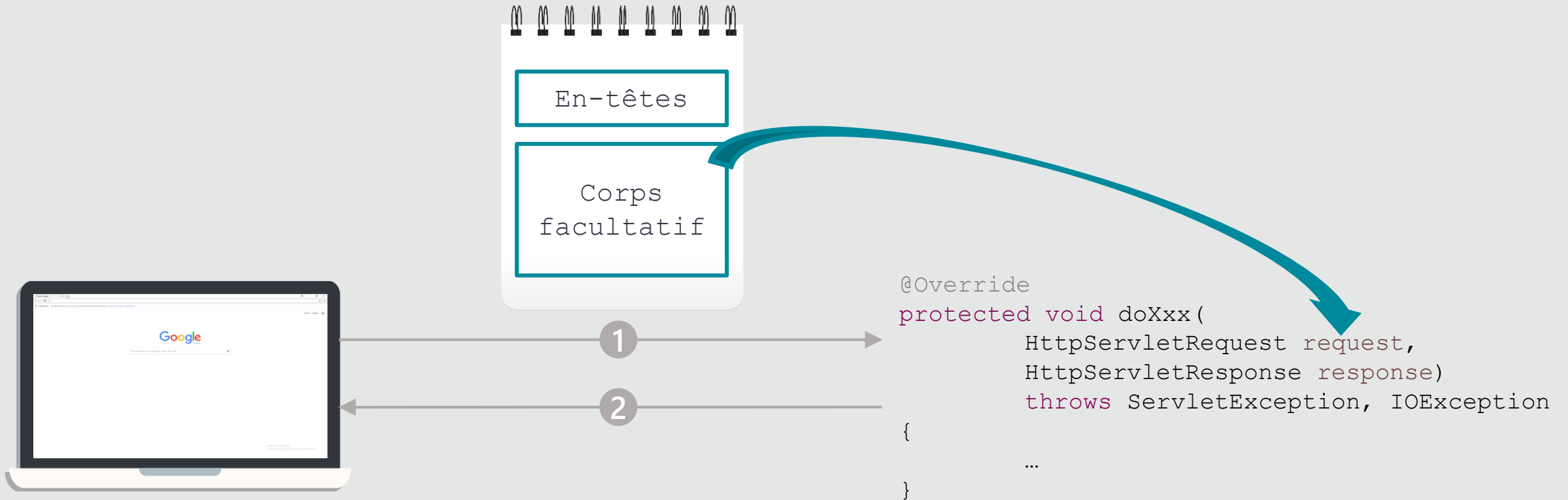
Les servlets

Le cycle de vie

Démonstration



Lien requête HTTP → Servlet




La lecture de la requête : l'URL

`http://www.exemples.fr:8080/DemoJavaEE/url/de/la/servlet`

HttpServletRequest
<<interface>>

```
getScheme():String  
getServerName():String  
getServerPort():int  
getContextPath():String  
getServletPath():String
```

La lecture de la requête : les en-têtes principaux




```
POST /docs/ouvrage HTTP/1.1
Host: www.exemples.fr
Accept-Language: en-US
...
```

```
nom=Java%20EE&auteur=ENI%20
Ecole
```

HttpServletRequest
<<interface>>

```
getCharacterEncoding():String
getContentType():int
getContentType():String
getLocale():Locale
getMethod():String
...
```


La lecture de la requête : tous les en-têtes




```
POST /docs/ouvrage HTTP/1.1
Host: www.exemples.fr
Accept-Language: en-US
...
```

```
nom=Java%20EE&auteur=ENI%20
Ecole
```

HttpServletRequest <<interface>>

```
getHeader(String name):String
getDateHeader(String name):Date
getIntHeader(String name):int
getHeaders(String name)
    :Enumeration<String>
getHeaderNames()
    :Enumeration<String>
...
```

La lecture de la requête : les paramètres



```
POST /docs/ouvrage HTTP/1.1
Host: www.exemples.fr
Accept-Language: en-US
...
```

```
nom=Java%20EE&auteur=ENI%20
Ecole
```

HttpServletRequest
<<interface>>

```
getParameter(String name)
    :String
getParameterValues(String name)
    :String[]
getParameterNames()
    :Enumeration<String>
getParameterMap()
    :Map<String, String[]>
```

*quel que soit
le type
de la requête*

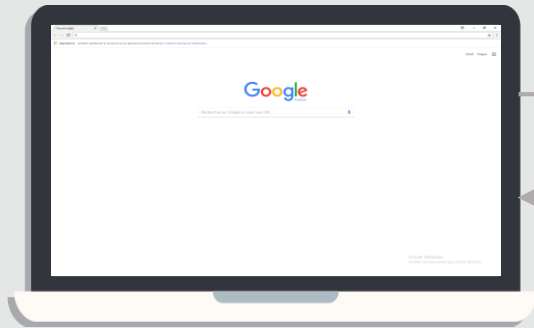
Les servlets

La lecture de la requête

Démonstration



Lien servlet → réponse HTTP




1

2



```
@Override
protected void doXXX(
    HttpServletRequest request,
    HttpServletResponse response)
    throws ServletException, IOException
{
    ...
}
```

L'écriture de la réponse : les en-têtes principaux



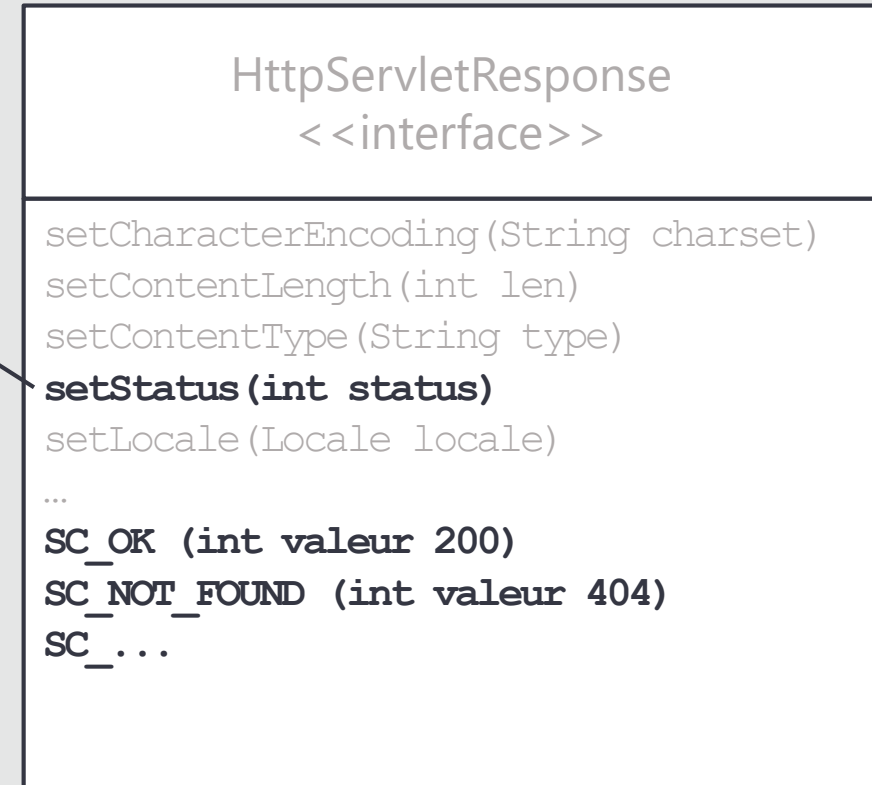
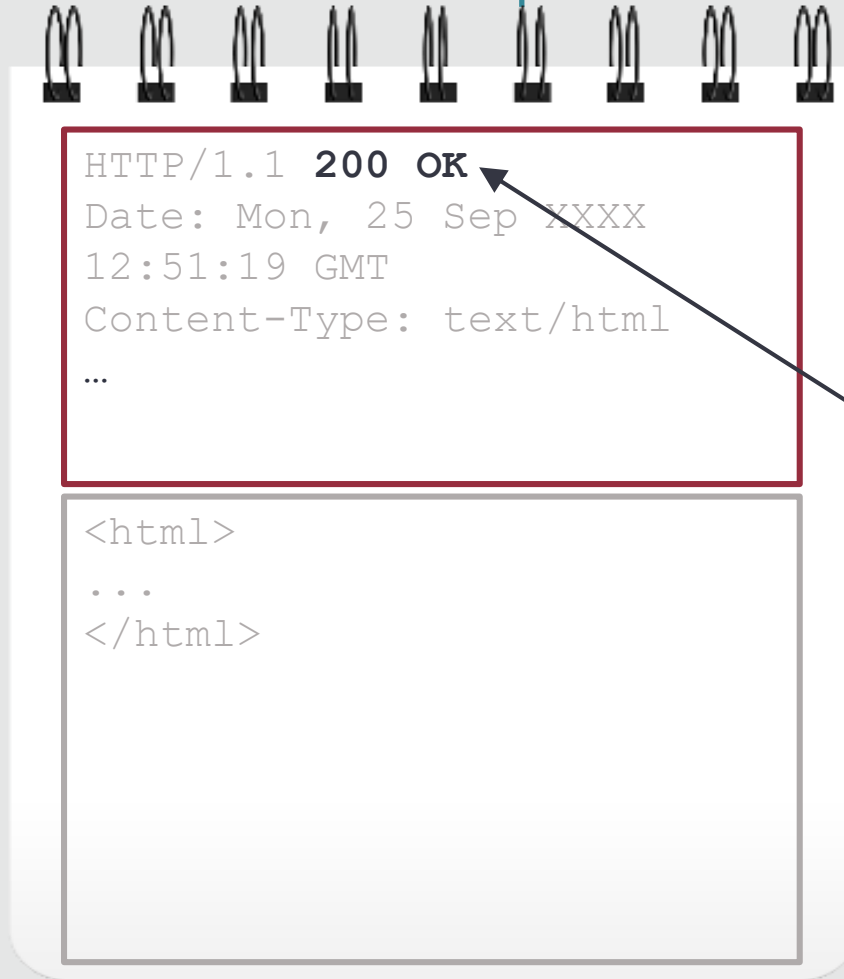
```
HTTP/1.1 200 OK
Date: Mon, 25 Sep XXXX
12:51:19 GMT
Content-Type: text/html
...
```

```
<html>
...
</html>
```

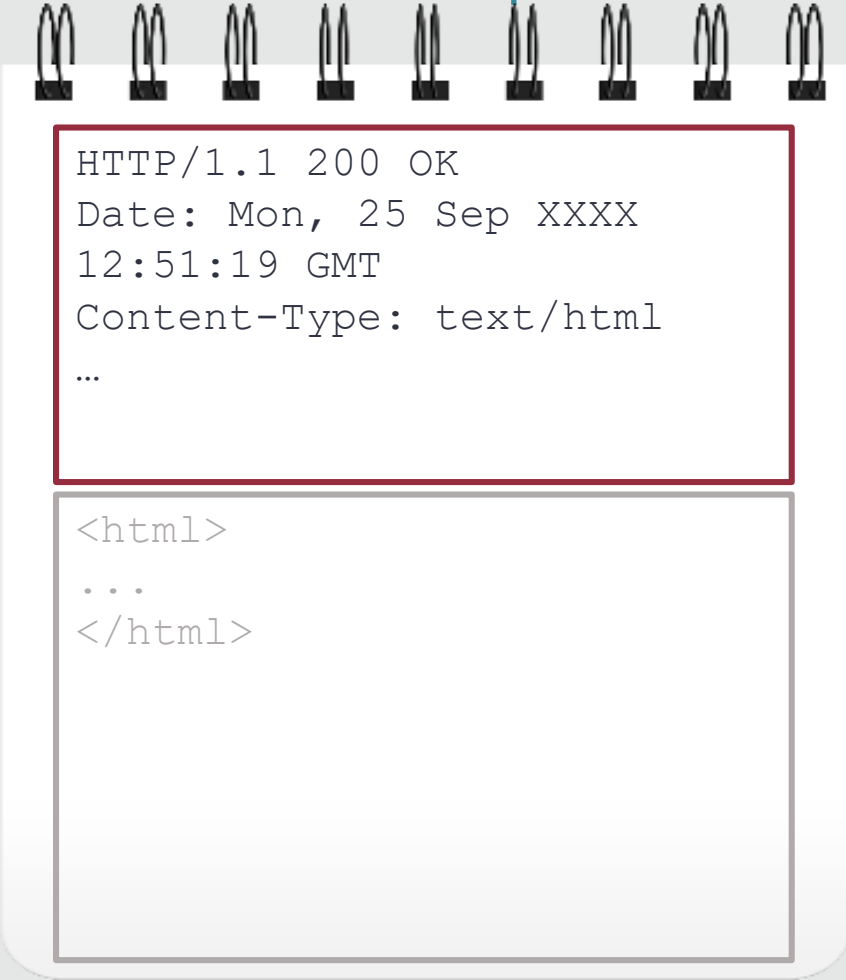
HttpServletResponse <<interface>>

```
setCharacterEncoding(String charset)
setContentLength(int len)
.setContentType(String type)
setStatus(int status)
setLocale(Locale locale)
...
```

L'écriture de la réponse : focus - le code de statut



L'écriture de la réponse : tous les en-têtes



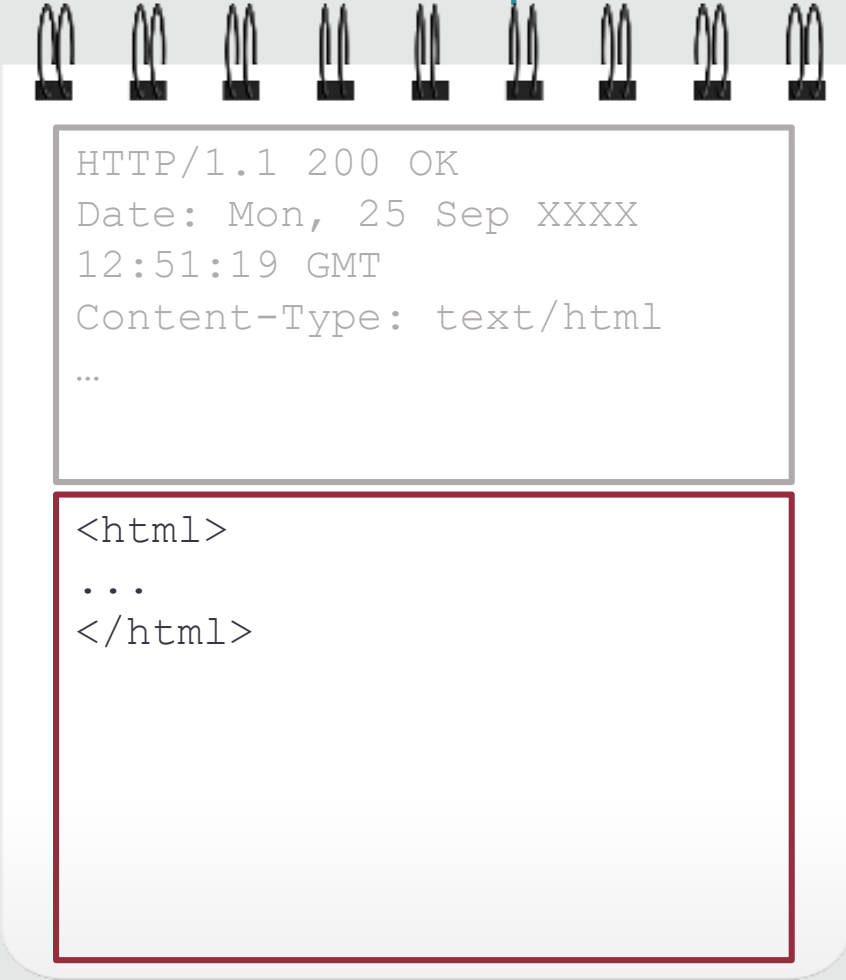
```
HTTP/1.1 200 OK  
Date: Mon, 25 Sep XXXX  
12:51:19 GMT  
Content-Type: text/html  
...
```

```
<html>  
...  
</html>
```

HttpServletResponse
<<interface>>

```
setHeader(String name, String value)  
setDateHeader(String name, long date)  
setIntHeader(String name, int value)  
...
```

L'écriture de la réponse : le corps



```
HTTP/1.1 200 OK
Date: Mon, 25 Sep XXXX
12:51:19 GMT
Content-Type: text/html
...
```

```
<html>
...
</html>
```

HttpServletResponse
<<interface>>

getWriter() : PrintWriter

Flux texte

getOutputStream() : ServletOutputStream

Flux binaire

L'écriture de la réponse : le corps au format texte

Utilisation

```
PrintWriter out = response.getWriter();  
out.println("Ecriture du corps de la réponse HTTP");  
//Suite écriture...  
out.close();
```

Manipulation du tampon

```
//Taille du tampon  
response.setBufferSize(size);  
//Vidage du tampon  
response.reset();  
//Forcer l'envoi du tampon  
out.flush();
```

Les servlets

L'écriture de la réponse

Démonstration



L'écriture de la réponse : la redirection permanente



HttpServletResponse
<<interface>>

```
setStatus(int status)  
setHeader(String name, String value)  
SC_MOVE_PERMANENTLY
```

```
response.setStatus(HttpServletResponse.SC_MOVED_PERMANENTLY);  
response.setHeader("Location", "Nouvelle URL");
```

L'écriture de la réponse : la redirection temporaire



HttpServletResponse <<interface>>
<pre>setStatus(int status) setHeader(String name, String value) SC_MOVE_TEMPORARILY ou sendRedirect(String location)</pre>

```
response.sendRedirect("URL Temporaire");
```

L'écriture de la réponse : répondre une erreur



web.xml

```
<error-page>
    <error-code>500</error-code>
    <location>/erreur500.html</location>
</error-page>
```

HttpServletResponse <<interface>>
<pre>sendError(int status) sendError(int status, String message)</pre>

```
response.sendError(HttpServletResponse.SC_INTERNAL_SERVER_ERROR);
```

Les servlets

Rediriger la réponse

Démonstration



Les servlets


Rechercher le nombre tiré au sort

TP




L'écriture de la réponse : IllegalStateException

1^{er} CAS

```
//traitement  
response.getWriter();  
//traitement  
response.getOutputStream();   
//traitement
```

2nd CAS

```
PrintWriter out = response.getWriter();  
out.println("Début réponse");  
//...  
//envoi réponse a débuté  
//traitement  
response.sendRedirect("URL");  *  
//traitement
```

***response.sendError(x); **

Les servlets

L'exception `IllegalStateException`



Démonstration



Les paramètres d'initialisation dans le web.xml



web.xml

```
<servlet>
    <servlet-name>UneServlet</servlet-name>
    <servlet-class>...</servlet-class>
    <init-param>
        <description>...</description>
        <param-name>NOM_PARAMETRE</param-name>
        <param-value>VALEUR_PARAMETRE</param-value>
    </init-param>
    ...
</servlet>
```

Les paramètres d'initialisation par annotation



```
@WebServlet
(
    name="UneServlet",
    urlPatterns="/url/de/la/servlet",
    initParams=
    {
        @WebInitParam(description="...",
                        name="NOM_PARAMETRE",
                        value="VALEUR_PARAMETRE"),
        ...
    }
)
```

Utilisation des paramètres d'initialisation

```
public class UneServlet extends HttpServlet
{
    private String valeurParametre;

    @Override
    public void init() throws ServletException
    {
        this.valeurParametre=this.getInitParameter("NOM_PARAMETRE");
    }
}
```

une seule
lecture

Les servlets

Les paramètres d'initialisation

Démonstration



Les servlets

Rechercher le nombre tiré au sort (évolution)

TP



Conclusion

- Vous savez maintenant générer une réponse adaptée avec la technologie des servlets