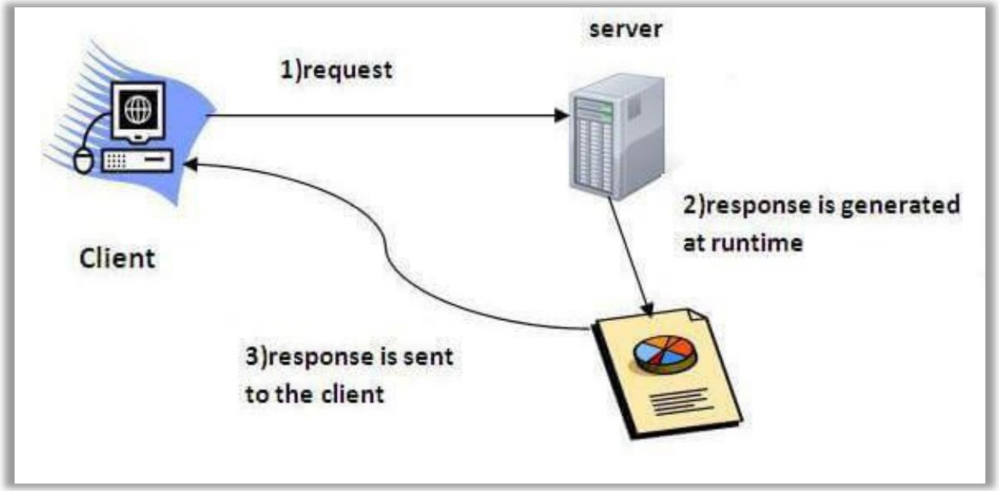
جامعة طرابلس
University of Tripoli
1957

Servlet API

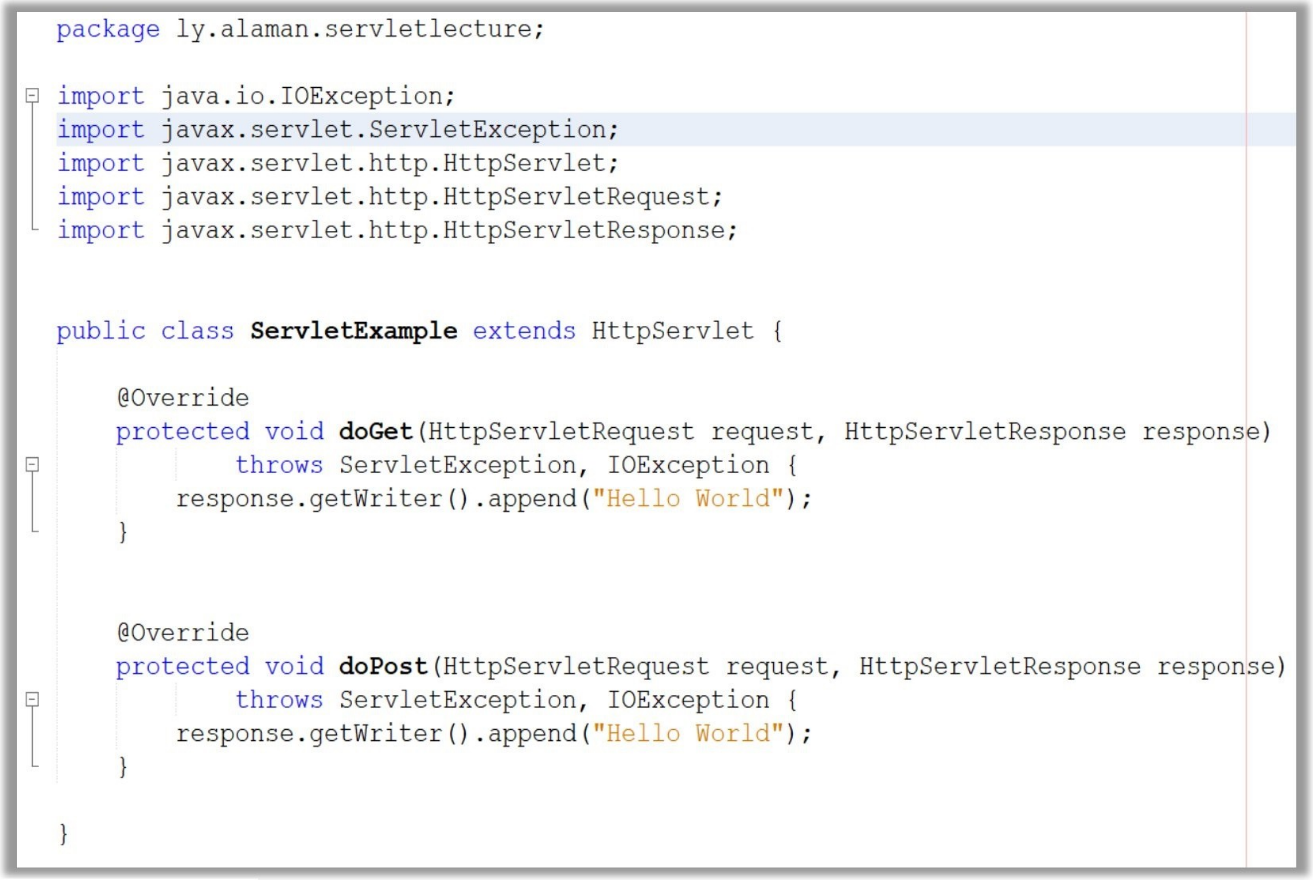
Java EE

What is Servlet?

- هي عبارة عن مكون برمجي بلغة الجافا تستخدم لاضافة بعض القدرات للسيرفر الموجودة به وذلك للتعامل مع request-response programming model.
- احد مكونات Java EE و عبارة عن برنامج جافا يستخدم في انشاء تطبيقات ويب من خلال انشاء dynamic web pages وهي موجودة في server side.
- Servlet API عبارة عن مجموعة من classes و interfaces لانشاء تطبيق ويب.
- هي web component تعمل على السيرفر لانشاء dynamic web pages.

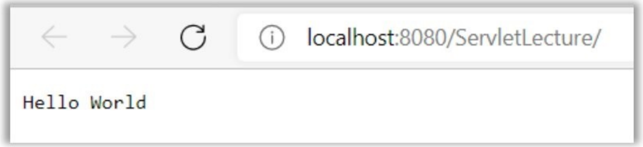

server
1)request
Client
2)response is generated at runtime
3)response is sent to the client

Example



```java
package ly.alaman.servletlecture;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


public class ServletExample extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.getWriter().append("Hello World");
    }


    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.getWriter().append("Hello World");
    }

}
```

البرنامج التالي يقوم بطباعة Hello World في المتصفح.



localhost:8080/ServletLecture/

```
Hello World
```

Servlet API

تحتوي على مجموعة من classes و interfaces الموجودة في كل من :

- Javax.servlet pckage وهي تستخدم في حالة لم يتم تحديد البروتوكول المستخدم.
- Javax.servlet.http package وهي تستخدم مع البروتوكول HTTP .

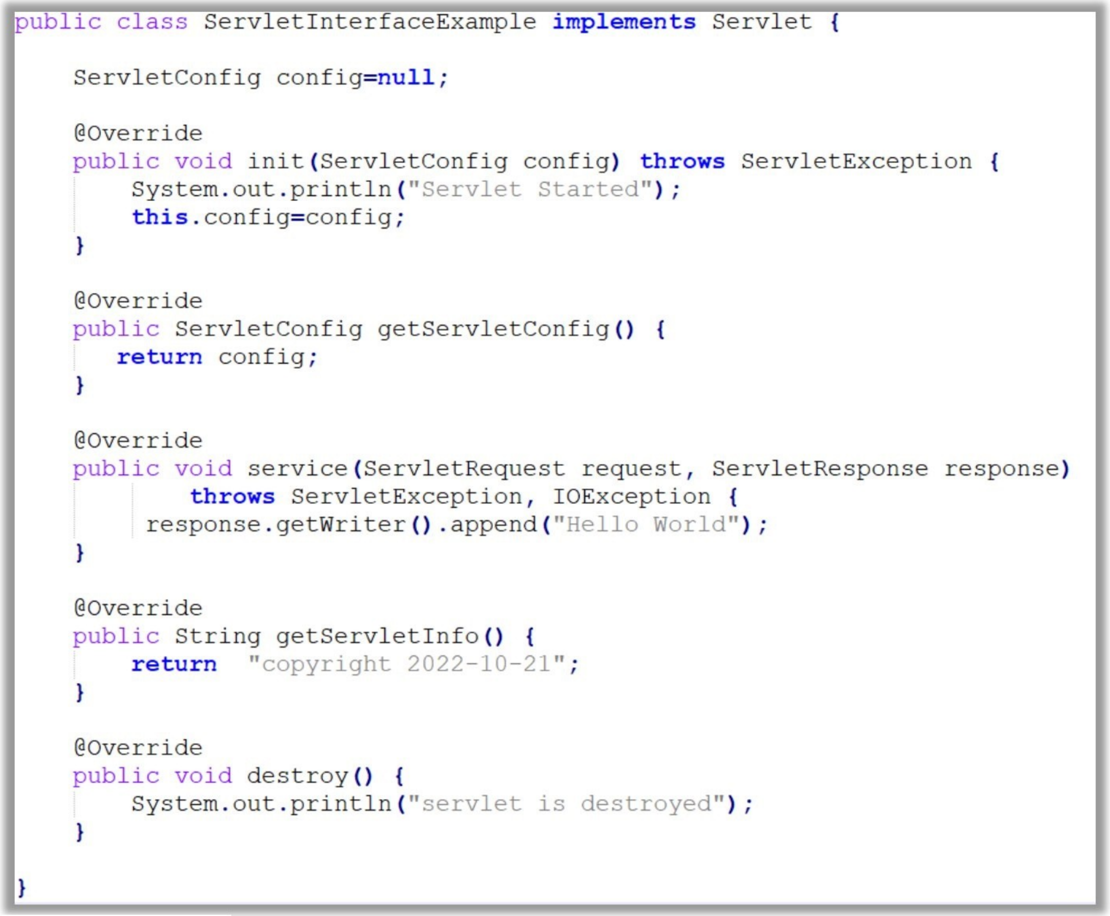
Servlet Interface

يقدم مجموعة من الدوال التي يجب على كل servlet تطبيقها لتقوم بالعمليات حسب الجدول التالي:

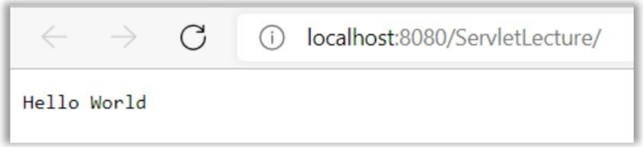
Method	Description
public void init(ServletConfig config)	تستخدم لتهيئة servlet ويتم استدعائها من container واحدة فقط.
public void service(ServletRequest request,ServletResponse response)	تستخدم للرد على الطلبات القادمة container وهي تستعدى عند كل طلب جديد.
public void destroy()	تستدعى مرة واحدة لانهاء عمل servlet .
public ServletConfig getServletConfig()	تقوم برد object من النوع ServletConfig.
public String getServletInfo()	تقوم باعطاء معلومات عن servlet مثل حقوق الطيع ورقم النسخة.

Example

البرنامج التالي يبين استخدام servlet interface .



```
public class ServletInterfaceExample implements Servlet {

    ServletConfig config=null;

    @Override
    public void init(ServletConfig config) throws ServletException {
        System.out.println("Servlet Started");
        this.config=config;
    }

    @Override
    public ServletConfig getServletConfig() {
       return config;
    }

    @Override
    public void service(ServletRequest request, ServletResponse response)
            throws ServletException, IOException {
         response.getWriter().append("Hello World");
    }

    @Override
    public String getServletInfo() {
        return  "copyright 2022-10-21";
    }

    @Override
    public void destroy() {
        System.out.println("servlet is destroyed");
    }

}
```

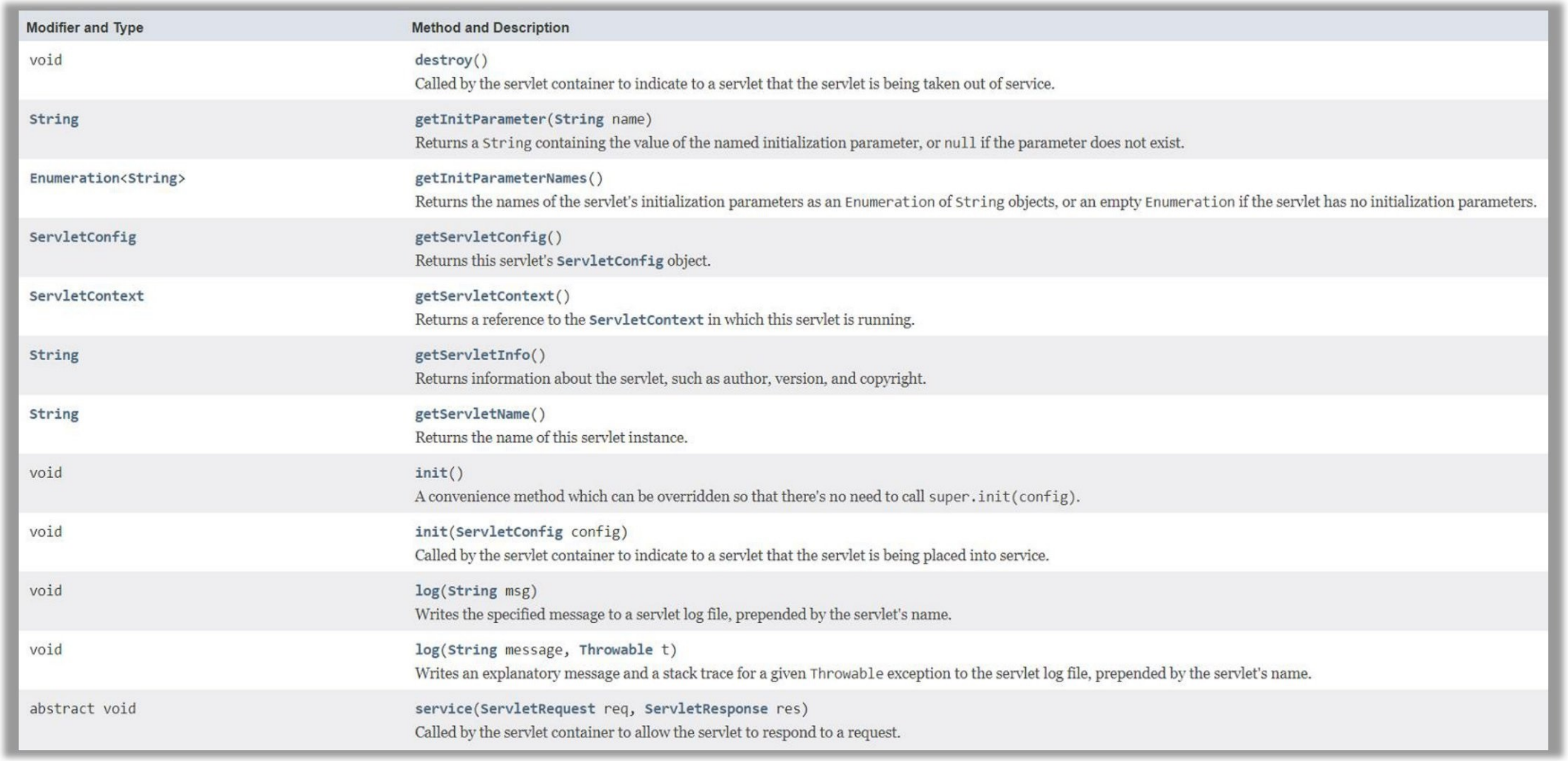


localhost:8080/ServletLecture/

```
Hello World
```

GenericServlet class

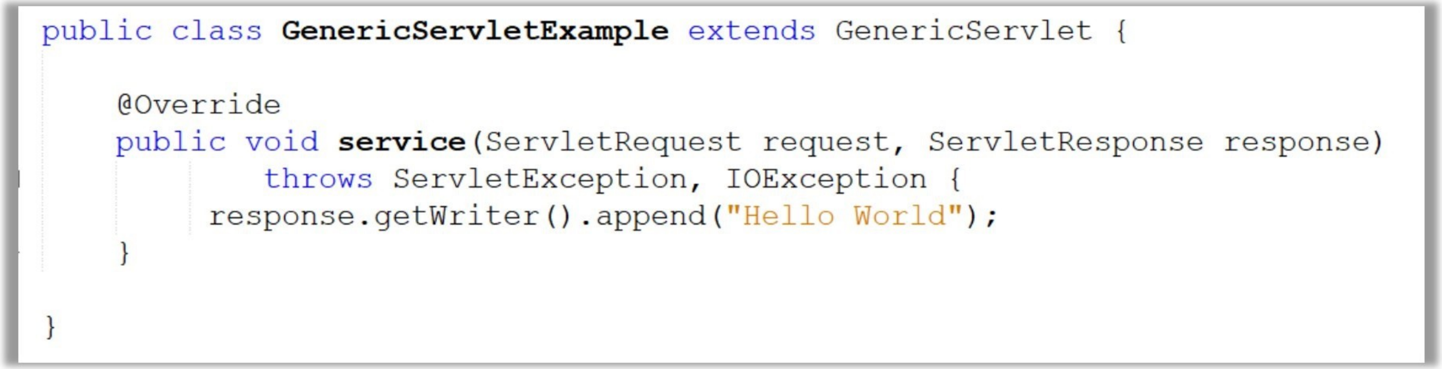
هي protocol independent servlet تستخدم للرد على طلبات المستخدمين من خلال عمل override للدالة service .



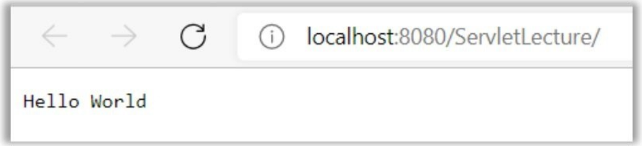
Modifier and Type	Method and Description
void	`destroy()` Called by the servlet container to indicate to a servlet that the servlet is being taken out of service.
String	`getInitParameter(String name)` Returns a `String` containing the value of the named initialization parameter, or `null` if the parameter does not exist.
Enumeration<String>	`getInitParameterNames()` Returns the names of the servlet's initialization parameters as an `Enumeration` of `String` objects, or an empty `Enumeration` if the servlet has no initialization parameters.
ServletConfig	`getServletConfig()` Returns this servlet's `ServletConfig` object.
ServletContext	`getServletContext()` Returns a reference to the `ServletContext` in which this servlet is running.
String	`getServletInfo()` Returns information about the servlet, such as author, version, and copyright.
String	`getServletName()` Returns the name of this servlet instance.
void	`init()` A convenience method which can be overridden so that there's no need to call `super.init(config)`.
void	`init(ServletConfig config)` Called by the servlet container to indicate to a servlet that the servlet is being placed into service.
void	`log(String msg)` Writes the specified message to a servlet log file, prepended by the servlet's name.
void	`log(String message, Throwable t)` Writes an explanatory message and a stack trace for a given `Throwable` exception to the servlet log file, prepended by the servlet's name.
abstract void	`service(ServletRequest req, ServletResponse res)` Called by the servlet container to allow the servlet to respond to a request.

Example

البرنامج التالي يبين استخدام GenericServlet class .



```
public class GenericServletExample extends GenericServlet {

    @Override
    public void service(ServletRequest request, ServletResponse response)
            throws ServletException, IOException {
        response.getWriter().append("Hello World");
    }

}
```

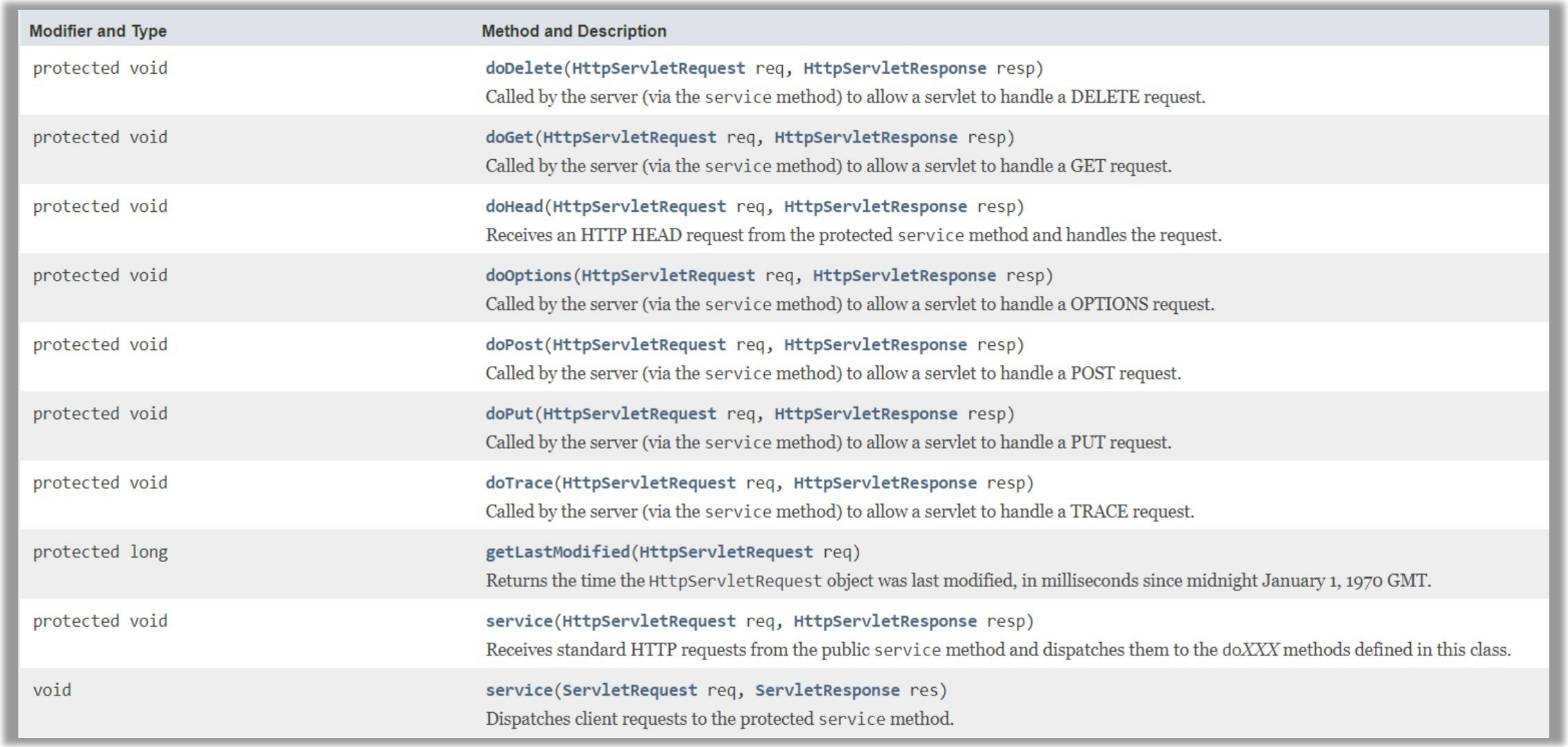


localhost:8080/ServletLecture/

```
Hello World
```

HttpServlet class

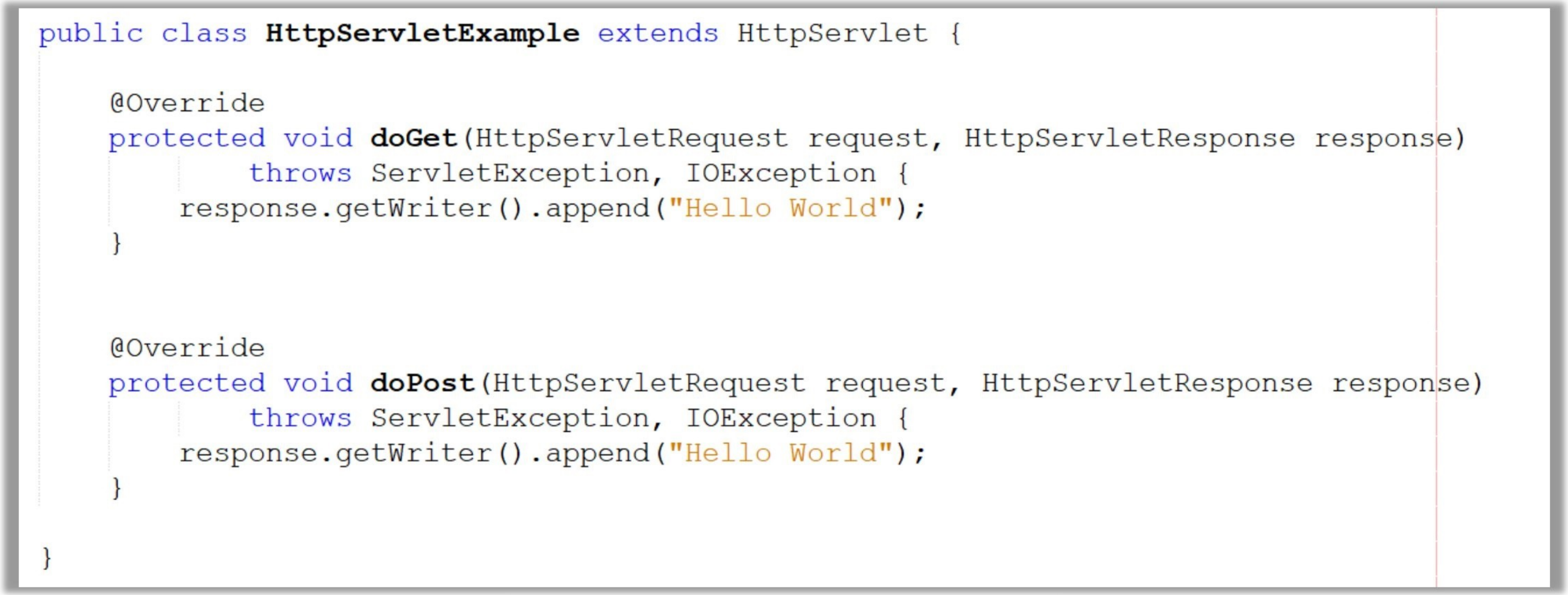
تستخدم للرد على الطلبات التي تستخدم HTTP protocol من خلال استخدام دوال خاصة بكل HTTP method .



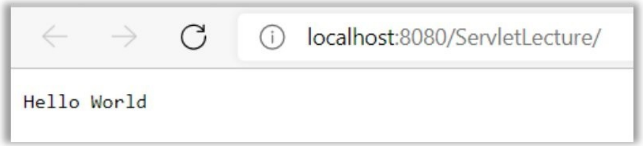
Modifier and Type	Method and Description
protected void	`doDelete(HttpServletRequest req, HttpServletResponse resp)` Called by the server (via the `service` method) to allow a servlet to handle a DELETE request.
protected void	`doGet(HttpServletRequest req, HttpServletResponse resp)` Called by the server (via the `service` method) to allow a servlet to handle a GET request.
protected void	`doHead(HttpServletRequest req, HttpServletResponse resp)` Receives an HTTP HEAD request from the protected `service` method and handles the request.
protected void	`doOptions(HttpServletRequest req, HttpServletResponse resp)` Called by the server (via the `service` method) to allow a servlet to handle a OPTIONS request.
protected void	`doPost(HttpServletRequest req, HttpServletResponse resp)` Called by the server (via the `service` method) to allow a servlet to handle a POST request.
protected void	`doPut(HttpServletRequest req, HttpServletResponse resp)` Called by the server (via the `service` method) to allow a servlet to handle a PUT request.
protected void	`doTrace(HttpServletRequest req, HttpServletResponse resp)` Called by the server (via the `service` method) to allow a servlet to handle a TRACE request.
protected long	`getLastModified(HttpServletRequest req)` Returns the time the `HttpServletRequest` object was last modified, in milliseconds since midnight January 1, 1970 GMT.
protected void	`service(HttpServletRequest req, HttpServletResponse resp)` Receives standard HTTP requests from the public `service` method and dispatches them to the do*XXX* methods defined in this class.
void	`service(ServletRequest req, ServletResponse res)` Dispatches client requests to the protected `service` method.

Example

البرنامج التالي يبين استخدام HttpServlet class .



```java
public class HttpServletExample extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.getWriter().append("Hello World");
    }


    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.getWriter().append("Hello World");
    }

}
```



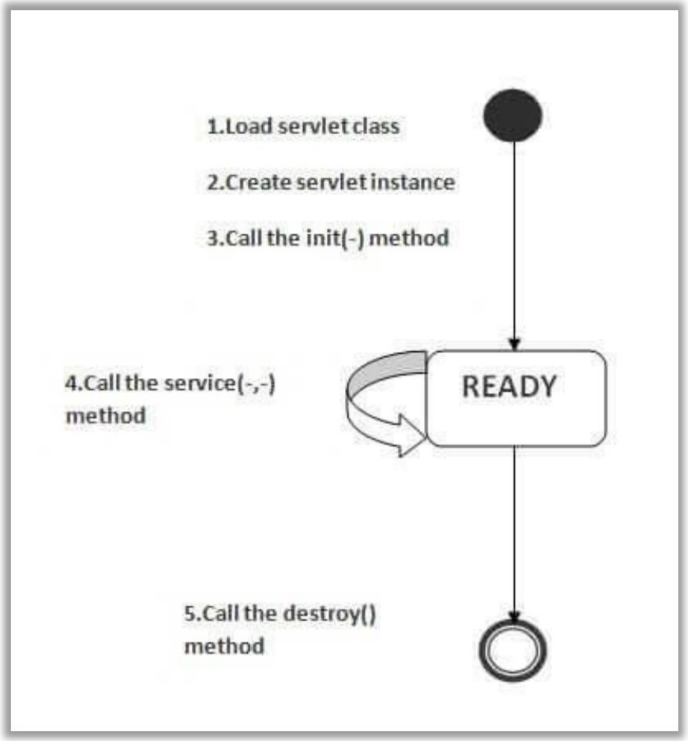
localhost:8080/ServletLecture/

```
Hello World
```

Servlet Life Cycle

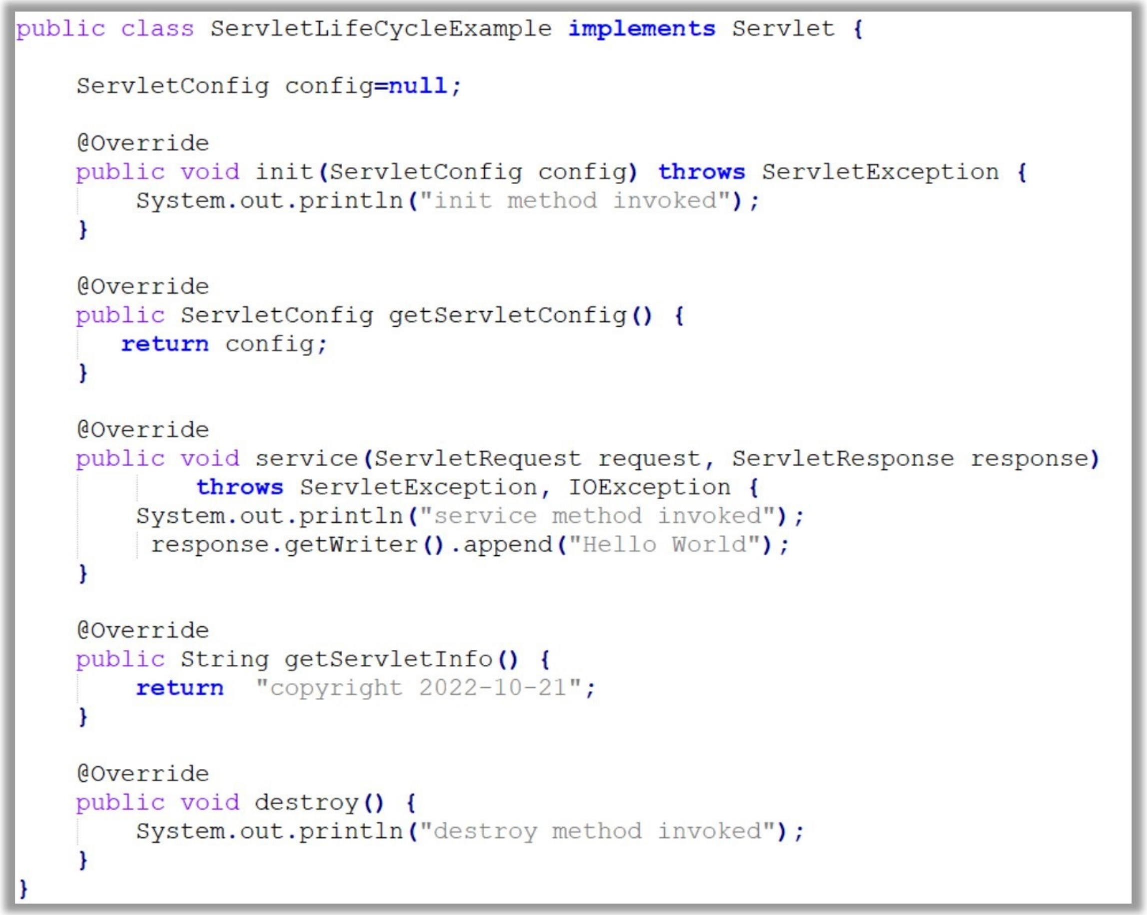
تقوم web container بإدارة دورة حياة servlet object وذلك على النحو التالي:

- تحميل Servlet class.
- انشاء Servlet object .
- استدعاء الدالة init.
- استدعاء الدالة service.
- استدعاء الدالة destroy.


1.Load servlet class
2.Create servlet instance
3.Call the init(-) method
4.Call the service(-,-) method
READY
5.Call the destroy() method

Example

البرنامج التالي يبين استدعاء life cycle methods .



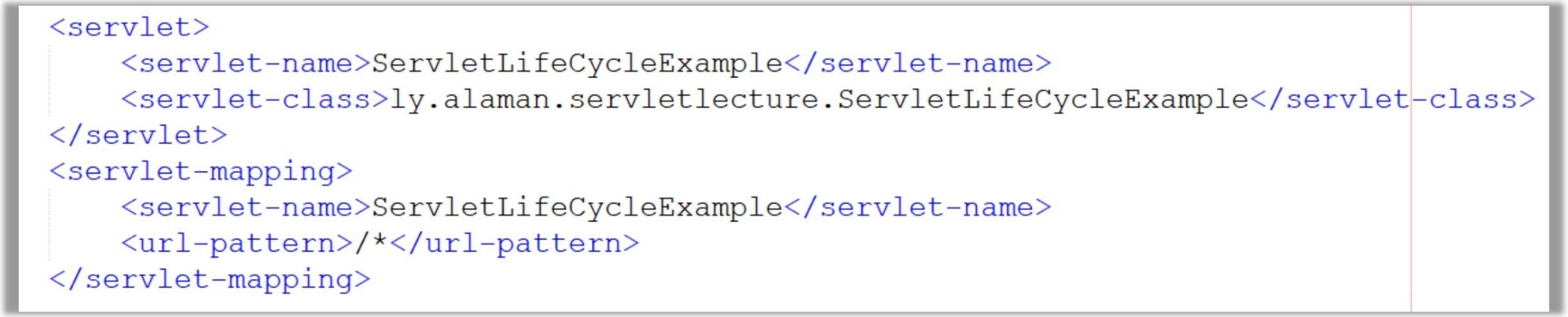
```java
public class ServletLifeCycleExample implements Servlet {

    ServletConfig config=null;

    @Override
    public void init(ServletConfig config) throws ServletException {
        System.out.println("init method invoked");
    }

    @Override
    public ServletConfig getServletConfig() {
       return config;
    }

    @Override
    public void service(ServletRequest request, ServletResponse response)
                throws ServletException, IOException {
        System.out.println("service method invoked");
         response.getWriter().append("Hello World");
    }

    @Override
    public String getServletInfo() {
        return  "copyright 2022-10-21";
    }

    @Override
    public void destroy() {
        System.out.println("destroy method invoked");
    }
}
```

Web.xml

بعد انشاء Servlet باستخدام احد الطرق التالية:

- By implementing Servlet interface,
- By inheriting GenericServlet class
- By inheriting HttpServlet class

يعرف ب deployment descriptor وهو ملف xml تستخدمه web container للحصول على معلومات بخصوص Servlet التي سيتم استدعاءها وذلك من خلال Tags التالية:



```xml
<servlet>
    <servlet-name>ServletLifeCycleExample</servlet-name>
    <servlet-class>ly.alaman.servletlecture.ServletLifeCycleExample</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>ServletLifeCycleExample</servlet-name>
    <url-pattern>/*</url-pattern>
</servlet-mapping>
```

ServletRequest interface

Method	Description
public String getParameter(String name)	is used to obtain the value of a parameter by name.
public String[] getParameterValues(String name)	returns an array of String containing all values of given parameter name. It is mainly used to obtain values of a Multi select list box.
java.util.Enumeration getParameterNames()	returns an enumeration of all of the request parameter names.
public int getContentLength()	Returns the size of the request entity data, or -1 if not known.
public String getCharacterEncoding()	Returns the character set encoding for the input of this request.
public String getContentType()	Returns the Internet Media Type of the request entity data, or null if not known.
public ServletInputStream getInputStream() throws IOException	Returns an input stream for reading binary data in the request body.
public abstract String getServerName()	Returns the host name of the server that received the request.
public int getServerPort()	Returns the port number on which this request was received.

بستخدم في الحصول على معلومات من client request مثل:

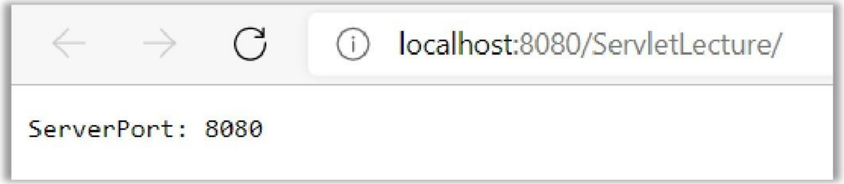
- content type
- content length
- parameter names and values
- header information
- attributes

يتم ذلك عن طريق مجموعة من الدوال أهمها :

Example

البرنامج التالي يبين استخدام ServletRequest

```java
public class ServletRequestExample extends GenericServlet {

    @Override
    public void service(ServletRequest request, ServletResponse response)
            throws ServletException, IOException {
        response.getWriter().append("ServerPort: "+request.getServerPort());
    }

}
```



```
ServerPort: 8080
```

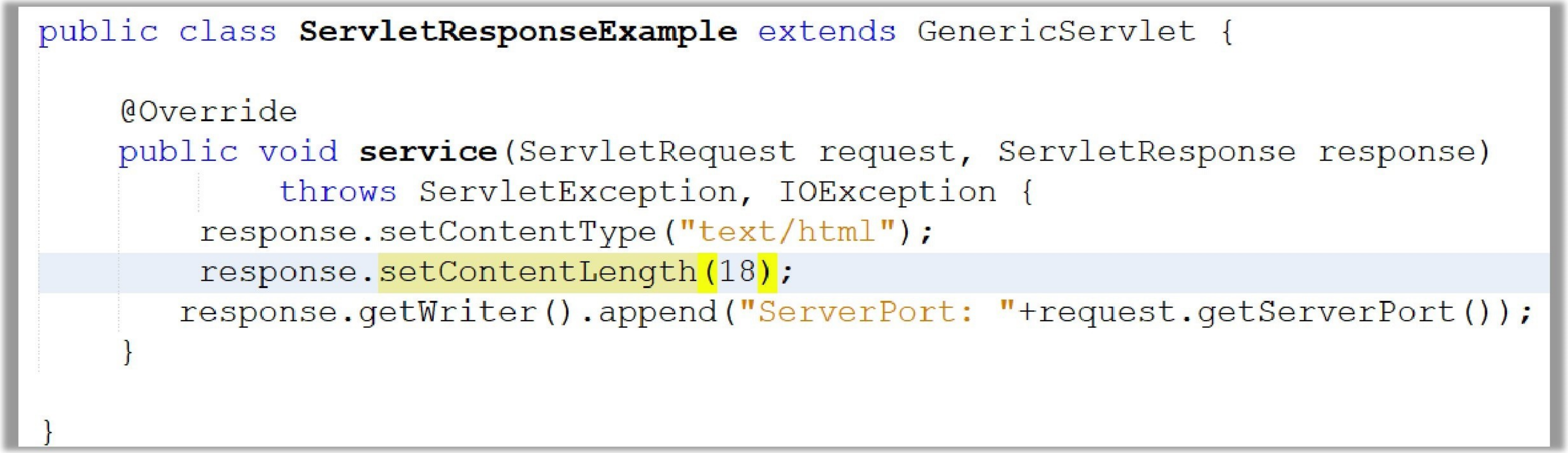
ServletResponse interface

بستخدم في تهيئة الرد قبل ارساله إلى clientمن خلال مجموعة منالدوال أهمها :

Methods	Description
PrintWriter getWriter()	returns a PrintWriter object that can send character text to the client.
void setBufferSize(int size)	Sets the preferred buffer size for the body of the response
void setContentLength(int len)	Sets the length of the content body in the response In HTTP servlets, this method sets the HTTP Content-Length header
void setContentType(String type)	sets the content type of the response being sent to the client before sending the respond.
void setBufferSize(int size)	sets the preferred buffer size for the body of the response.
boolean isCommitted()	returns a boolean indicating if the response has been committed
void setLocale(Locale loc)	sets the locale of the response, if the response has not been committed yet.

Example

البرنامج التالي يبين استخدام ServletResponse



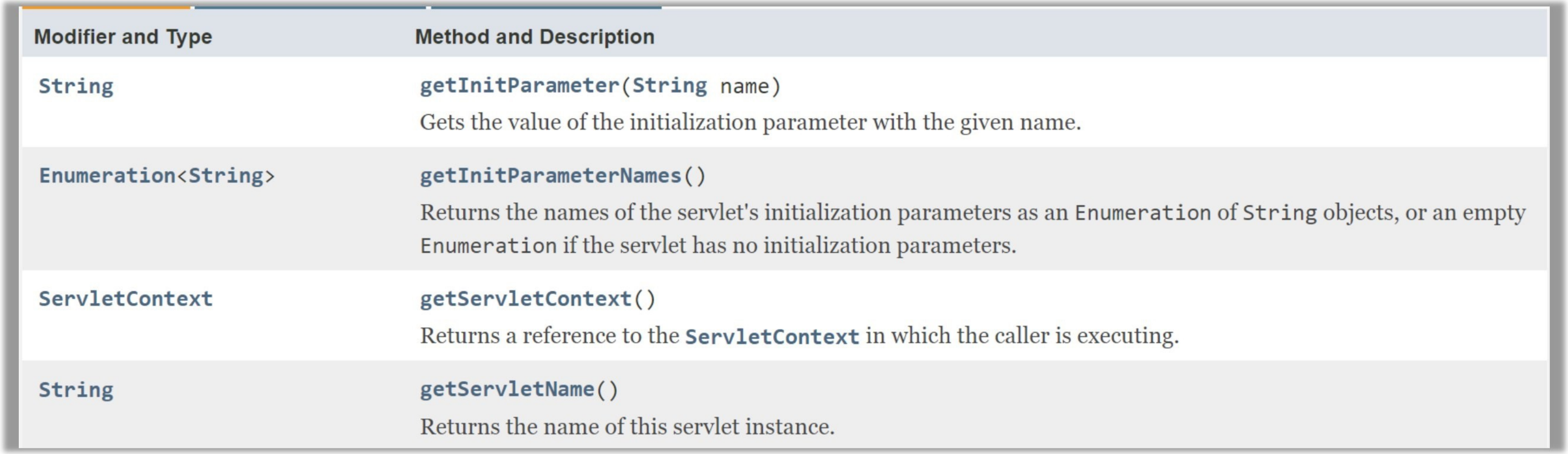
```
public class ServletResponseExample extends GenericServlet {

    @Override
    public void service(ServletRequest request, ServletResponse response)
            throws ServletException, IOException {
         response.setContentType("text/html");
         response.setContentLength(18);
        response.getWriter().append("ServerPort: "+request.getServerPort());
    }

}
```

ServletConfig Interface

تستخدم في الحصول على object عن طريق web container لكل servlet يمكن استخدامه في الحصول configuration information من ملف web.xml .

يمكن استخدامه لتمرير معلومات إلى servlet عتد الحاجة.

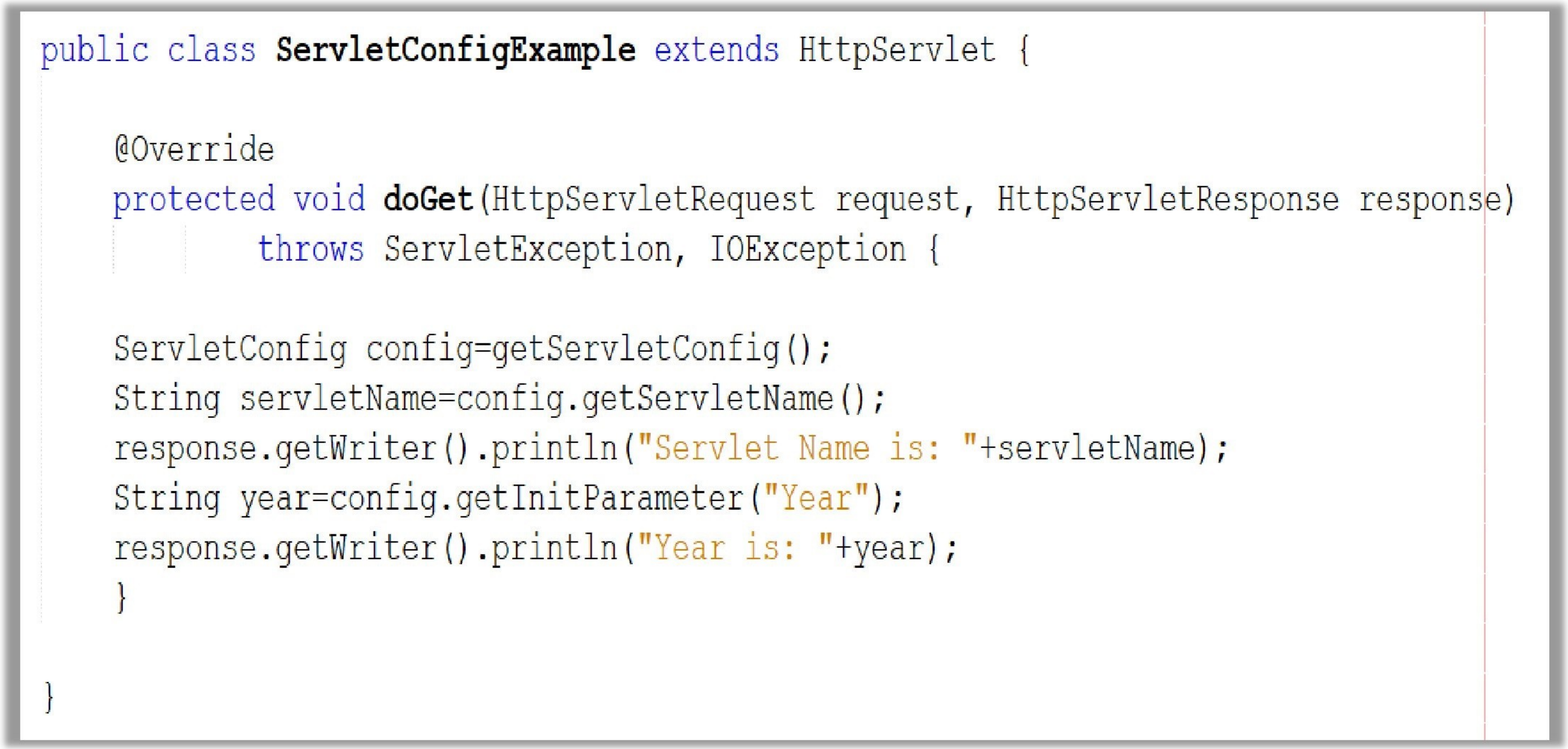
يوفر مجموعة من الدوال للقيام بذلك منها:



Modifier and Type	Method and Description
`String`	`getInitParameter(String name)` Gets the value of the initialization parameter with the given name.
`Enumeration<String>`	`getInitParameterNames()` Returns the names of the servlet's initialization parameters as an `Enumeration` of `String` objects, or an empty `Enumeration` if the servlet has no initialization parameters.
`ServletContext`	`getServletContext()` Returns a reference to the `ServletContext` in which the caller is executing.
`String`	`getServletName()` Returns the name of this servlet instance.

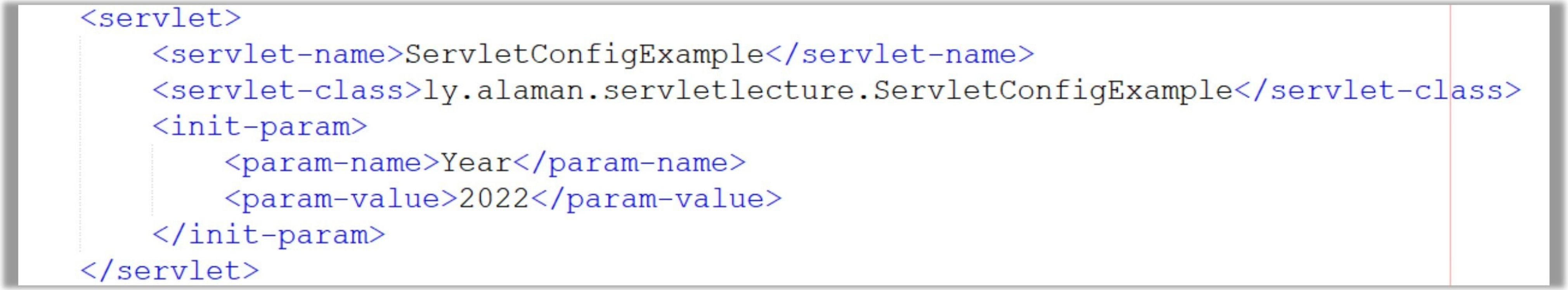
Example

البرنامج التالي يبين استخدام ServletConfig Interface

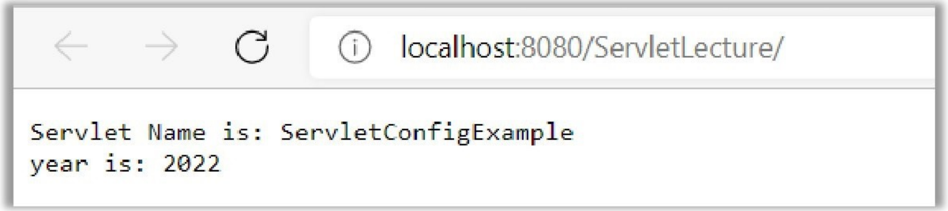


```
public class ServletConfigExample extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

    ServletConfig config=getServletConfig();
    String servletName=config.getServletName();
    response.getWriter().println("Servlet Name is: "+servletName);
    String year=config.getInitParameter("Year");
    response.getWriter().println("Year is: "+year);
    }

}
```

يتبع



```xml
<servlet>
    <servlet-name>ServletConfigExample</servlet-name>
    <servlet-class>ly.alaman.servletlecture.ServletConfigExample</servlet-class>
    <init-param>
        <param-name>Year</param-name>
        <param-value>2022</param-value>
    </init-param>
</servlet>
```



```
Servlet Name is: ServletConfigExample
year is: 2022
```

ServletContext Interface

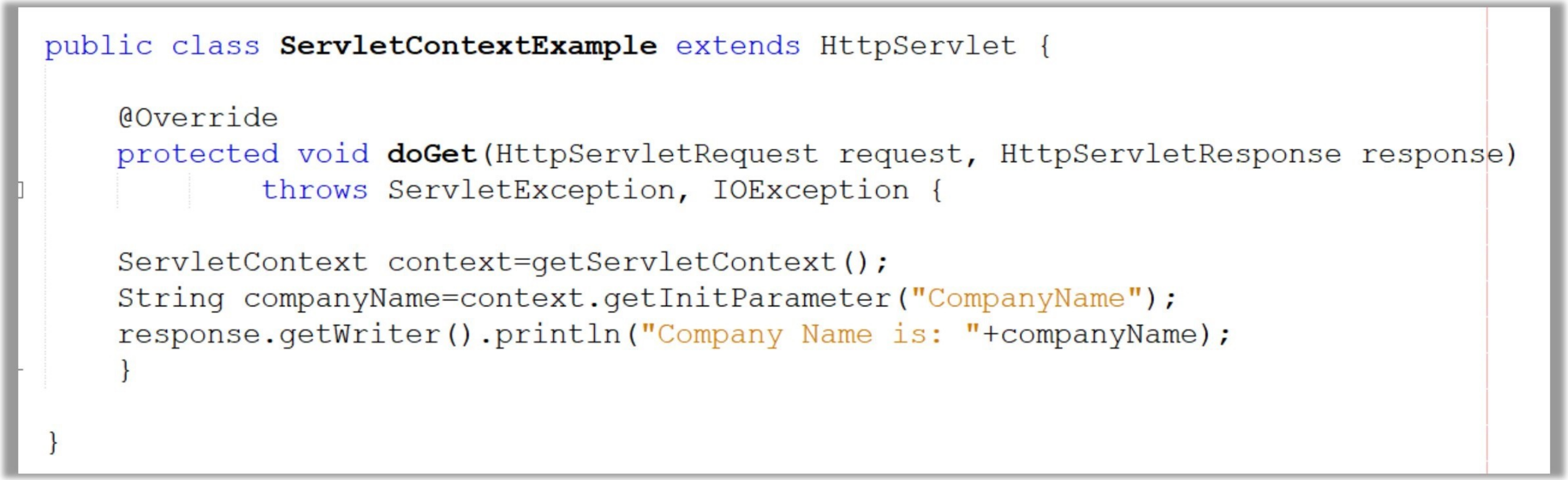
تستخدم في الحصول على object عن طريق web container يمكن استخدامه في الحصول configuration information من ملف web.xml .

يمكن استخدامه لتمرير معلومات المشتركة بين كل servlets عتد الحاجة.

Example

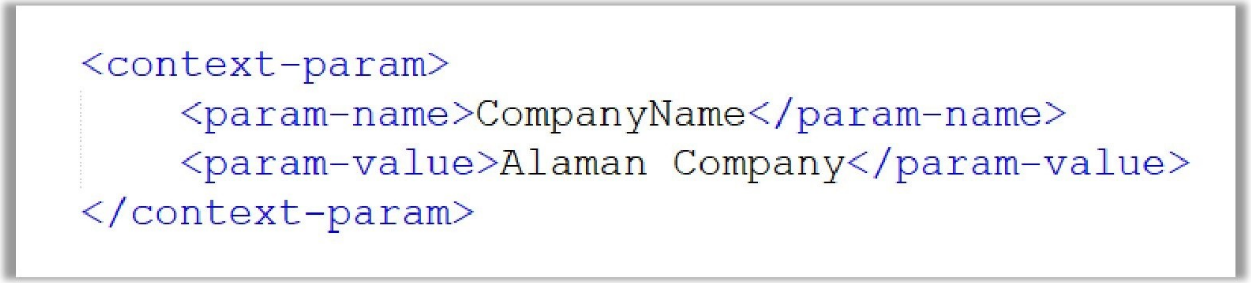
البرنامج التالي يبين استخدام ServletContext Interface

.

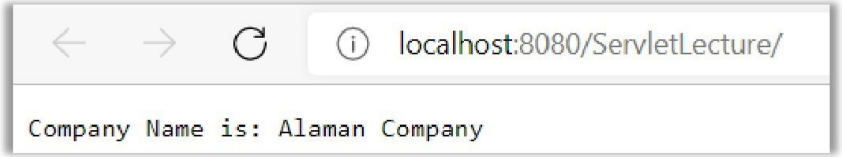


```
public class ServletContextExample extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

    ServletContext context=getServletContext();
    String companyName=context.getInitParameter("CompanyName");
    response.getWriter().println("Company Name is: "+companyName);
    }

}
```

Java EE يتبع



```
<context-param>
    <param-name>CompanyName</param-name>
    <param-value>Alaman Company</param-value>
</context-param>
```

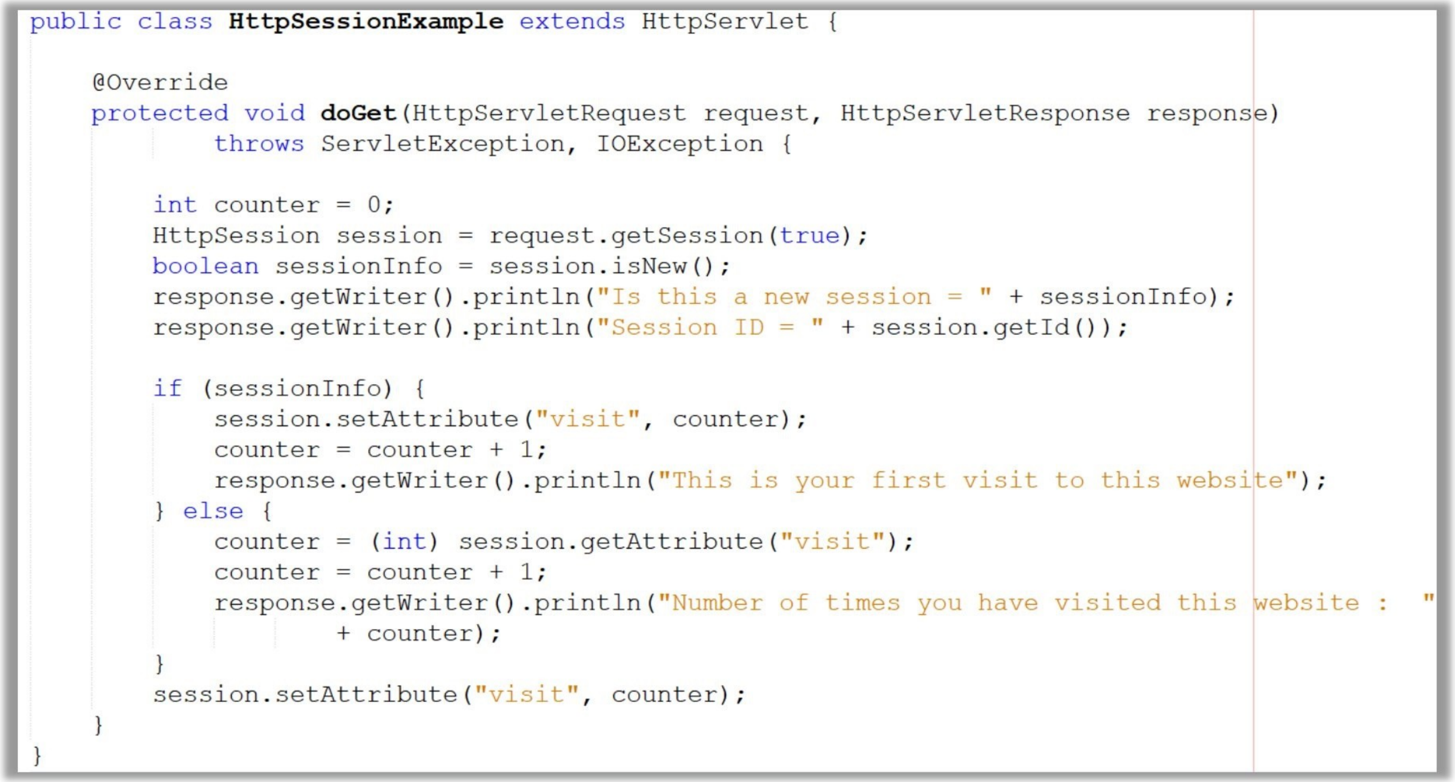


```
Company Name is: Alaman Company
```

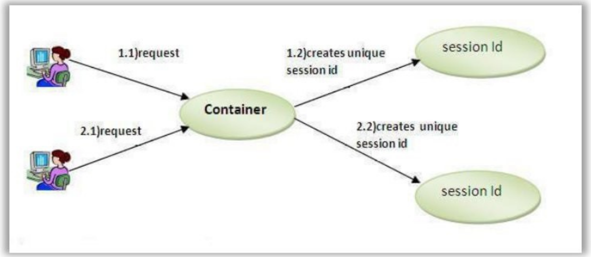
HttpSession interface

HTTP يعتبر stateless protocol وللحفاظ على user session يمكن استخدام HttpSession interface .

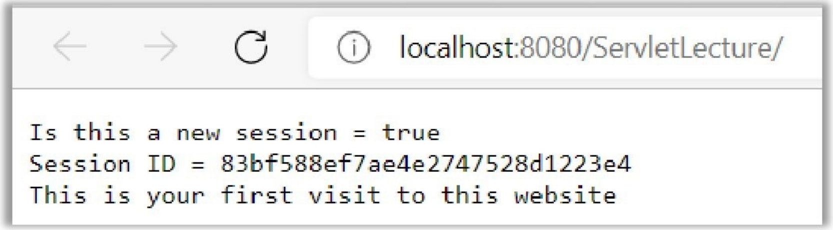
للحصول على HttpSession Object يمكن استخدام HttpServletRequest كالتالي:



```java
public class HttpSessionExample extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        int counter = 0;
        HttpSession session = request.getSession(true);
        boolean sessionInfo = session.isNew();
        response.getWriter().println("Is this a new session = " + sessionInfo);
        response.getWriter().println("Session ID = " + session.getId());

        if (sessionInfo) {
            session.setAttribute("visit", counter);
            counter = counter + 1;
            response.getWriter().println("This is your first visit to this website");
        } else {
            counter = (int) session.getAttribute("visit");
            counter = counter + 1;
            response.getWriter().println("Number of times you have visited this website :  "
                    + counter);
        }
        session.setAttribute("visit", counter);
    }
}
```

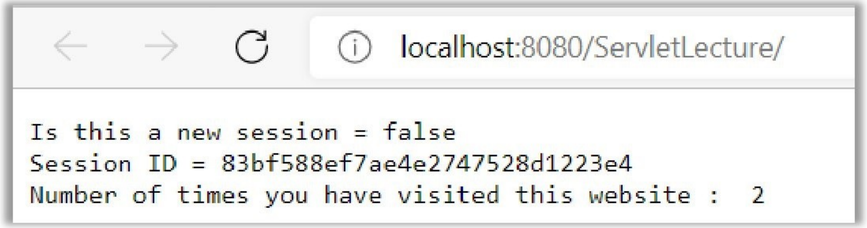


يتبع



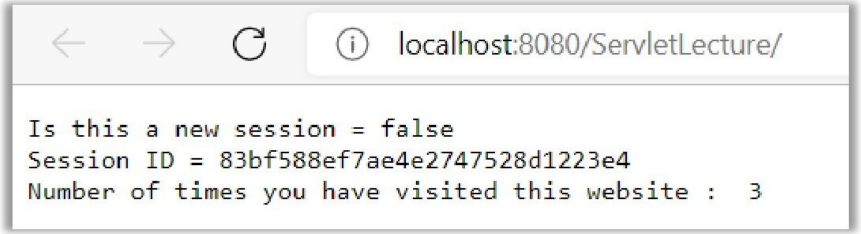
localhost:8080/ServletLecture/

```
Is this a new session = true
Session ID = 83bf588ef7ae4e2747528d1223e4
This is your first visit to this website
```



localhost:8080/ServletLecture/

```
Is this a new session = false
Session ID = 83bf588ef7ae4e2747528d1223e4
Number of times you have visited this website :  2
```

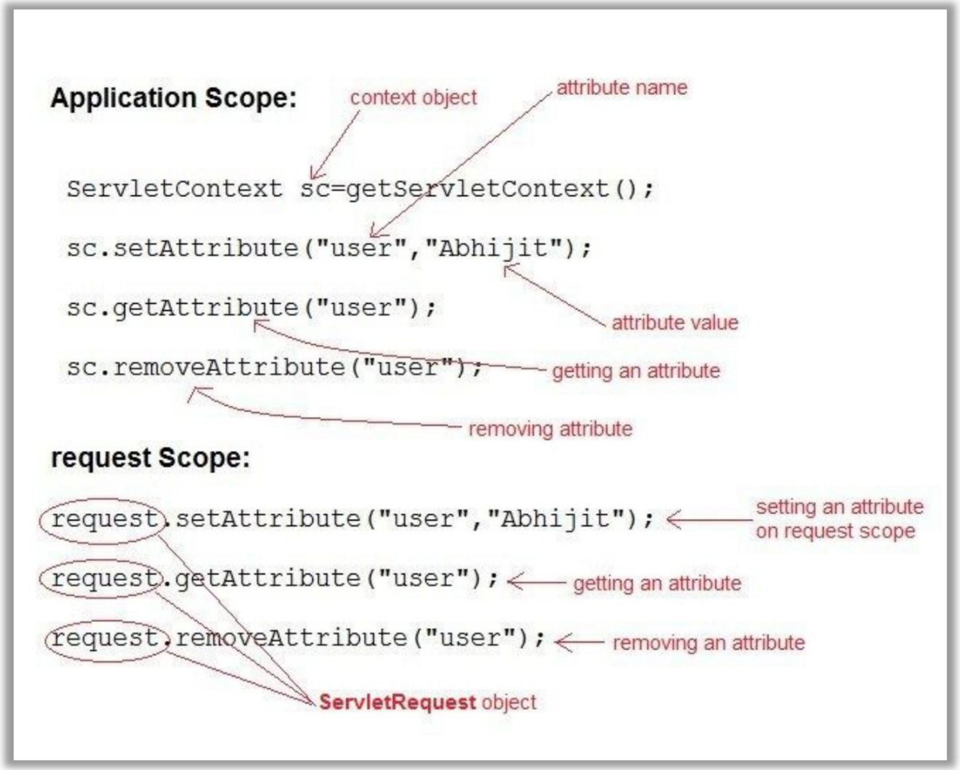


localhost:8080/ServletLecture/

```
Is this a new session = false
Session ID = 83bf588ef7ae4e2747528d1223e4
Number of times you have visited this website :  3
```

Attribute in Servlet

- عبارة عن object يستخدم لمشاركة البيانات بين servlets ، يمكن عمل set او get او remove له في scopes المختلفة مثل:
- Request scope
- Session scope
- Application scope



Application Scope:

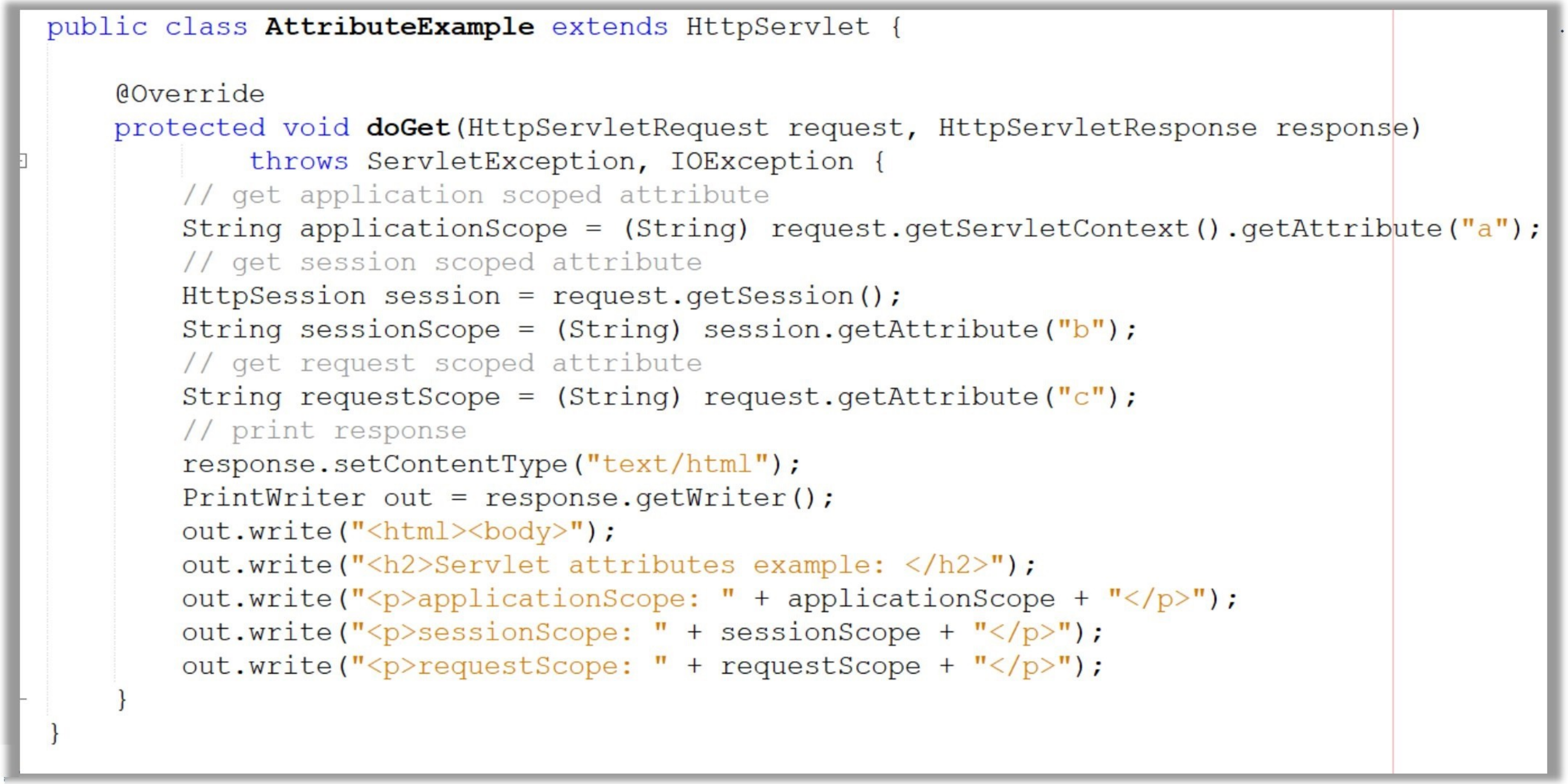
```
ServletContext sc=getServletContext();
sc.setAttribute("user","Abhijit");
sc.getAttribute("user");
sc.removeAttribute("user");
```

request Scope:

```
request.setAttribute("user","Abhijit");
request.getAttribute("user");
request.removeAttribute("user");
```

Example

البرنامج التالي يبين استخدام الحصول على attribute باستخدام scopes مختلفة.

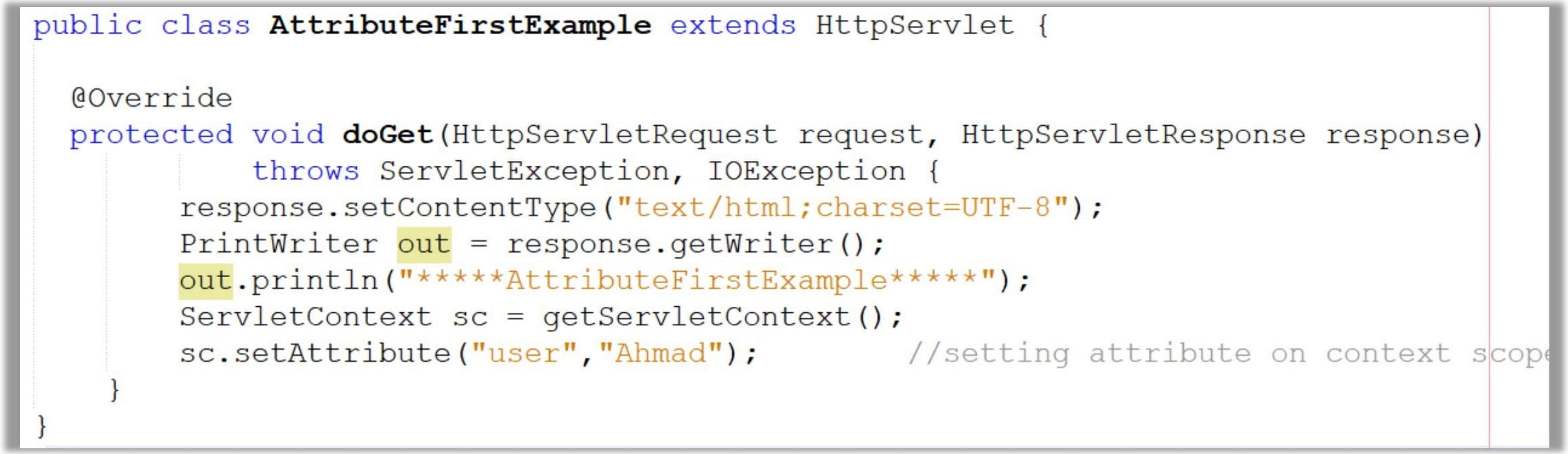


```
public class AttributeExample extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        // get application scoped attribute
        String applicationScope = (String) request.getServletContext().getAttribute("a");
        // get session scoped attribute
        HttpSession session = request.getSession();
        String sessionScope = (String) session.getAttribute("b");
        // get request scoped attribute
        String requestScope = (String) request.getAttribute("c");
        // print response
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.write("<html><body>");
        out.write("<h2>Servlet attributes example: </h2>");
        out.write("<p>applicationScope: " + applicationScope + "</p>");
        out.write("<p>sessionScope: " + sessionScope + "</p>");
        out.write("<p>requestScope: " + requestScope + "</p>");
    }
}
```

Example

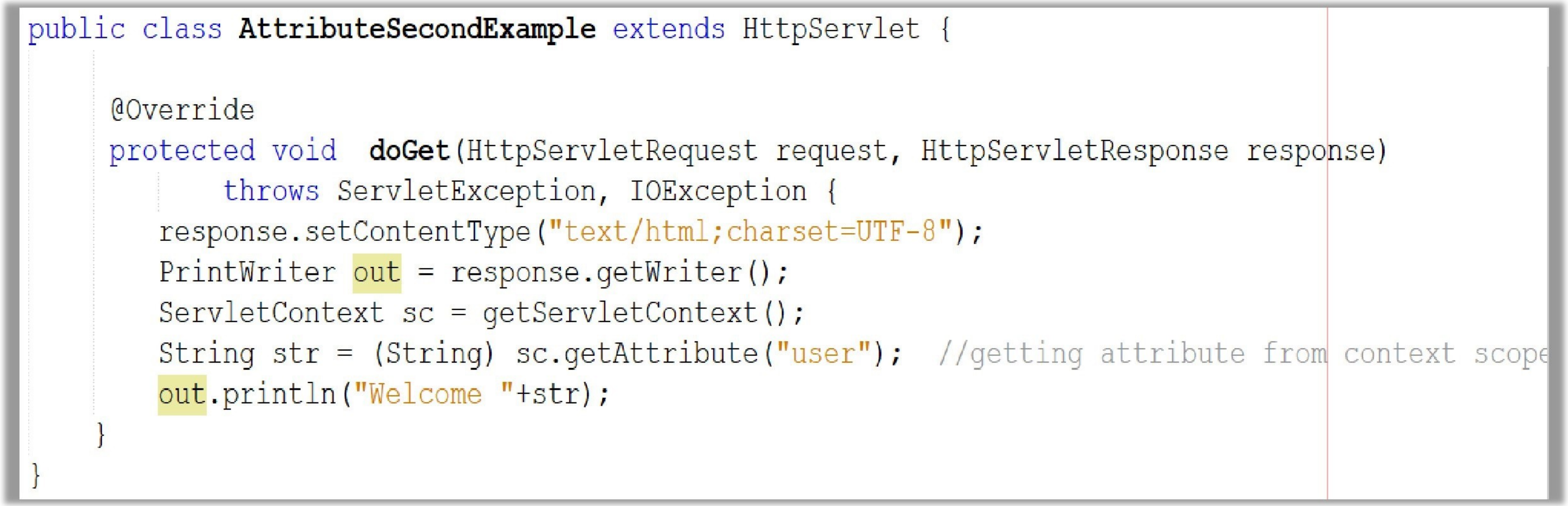
البرنامج التالي يبين استخدام attribute وذلك من خلال تمرير اسم المستخدم بين First servlet و Second servlet .

.

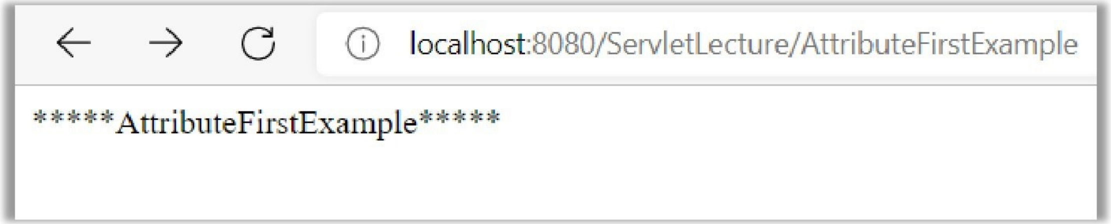


```java
public class AttributeFirstExample extends HttpServlet {

  @Override
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
          throws ServletException, IOException {
      response.setContentType("text/html;charset=UTF-8");
      PrintWriter out = response.getWriter();
      out.println("*****AttributeFirstExample*****");
      ServletContext sc = getServletContext();
      sc.setAttribute("user","Ahmad");        //setting attribute on context scope
    }
}
```

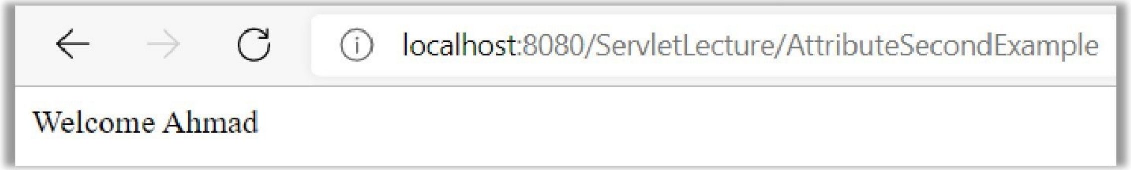
يتبع



```
public class AttributeSecondExample extends HttpServlet {

    @Override
    protected void  doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        ServletContext sc = getServletContext();
        String str = (String) sc.getAttribute("user");  //getting attribute from context scope
        out.println("Welcome "+str);
    }
}
```



```
localhost:8080/ServletLecture/AttributeFirstExample
*****AttributeFirstExample*****
```



```
localhost:8080/ServletLecture/AttributeSecondExample
Welcome Ahmad
```