SERVLET

1.
2. Describe the classes and interfaces of javax.servlet package.
3. Describe the classes and interfaces of javax.servlet.http package.

1) Interfaces in `javax.servlet` package:

Interface	Description
Servlet	Declares life cycle methods for a servlet.
ServletConfig	Allows servlets to get initialization parameters.
ServletContext	Enables servlets to log events and access information about their environment.
ServletRequest	Used to read data from a client request.
ServletResponse	Used to write data to a client response.

2) Classes in `javax.servlet` package:

Class	Description
GenericServlet	Implements the Servlet and ServletConfig interfaces.
ServletInputStream	Provides an input stream for reading requests from a client.
ServletOutputStream	Provides an output stream for writing responses to a client.
ServletException	Indicates a servlet error occurred.
UnavailableException	Indicates a servlet is unavailable.

Servlet

Method	Description
void destroy( )	Called when the servlet is unloaded.
ServletConfig getServletConfig( )	Returns a **ServletConfig** object that contains any initialization parameters.
String getServletInfo( )	Returns a string describing the servlet.
void init(ServletConfig *sc*) throws ServletException	Called when the servlet is initialized. Initialization parameters for the servlet can be obtained from *sc*. An **UnavailableException** should be thrown if the servlet cannot be initialized.
void service(ServletRequest *req*, ServletResponse *res*) throws ServletException, IOException	Called to process a request from a client. The request from the client can be read from *req*. The response to the client can be written to *res*. An exception is generated if a servlet or IO problem occurs.

TABLE 31-1 The Methods Defined by **Servlet**

ServletConfig

Method	Description
ServletContext getServletContext( )	Returns the context for this servlet.
String getInitParameter(String *param*)	Returns the value of the initialization parameter named *param*.
Enumeration getInitParameterNames( )	Returns an enumeration of all initialization parameter names.
String getServletName( )	Returns the name of the invoking servlet.

ServletContext

Method	Description
Object getAttribute(String *attr*)	Returns the value of the server attribute named *attr*.
String getMimeType(String *file*)	Returns the MIME type of *file*.
String getRealPath(String *vpath*)	Returns the real path that corresponds to the virtual path *vpath*.
String getServerInfo( )	Returns information about the server.
void log(String *s*)	Writes *s* to the servlet log.
void log(String *s*, Throwable *e*)	Writes *s* and the stack trace for *e* to the servlet log.
void setAttribute(String *attr*, Object *val*)	Sets the attribute specified by *attr* to the value passed in *val*.

TABLE 31-2 Various Methods Defined by **ServletContext**

ServletRequest

Method	Description
Object getAttribute(String *attr*)	Returns the value of the attribute named *attr*.
String getCharacterEncoding( )	Returns the character encoding of the request.
int getContentLength( )	Returns the size of the request. The value –1 is returned if the size is unavailable.
String getContentType( )	Returns the type of the request. A **null** value is returned if the type cannot be determined.
ServletInputStream getInputStream( ) throws IOException	Returns a **ServletInputStream** that can be used to read binary data from the request. An **IllegalStateException** is thrown if **getReader( )** has already been invoked for this request.
String getParameter(String *pname*)	Returns the value of the parameter named *pname*.
Enumeration getParameterNames( )	Returns an enumeration of the parameter names for this request.
String[ ] getParameterValues(String *name*)	Returns an array containing values associated with the parameter specified by *name*.
String getProtocol( )	Returns a description of the protocol.
BufferedReader getReader( ) throws IOException	Returns a buffered reader that can be used to read text from the request. An **IllegalStateException** is thrown if **getInputStream( )** has already been invoked for this request.
String getRemoteAddr( )	Returns the string equivalent of the client IP address.
String getRemoteHost( )	Returns the string equivalent of the client host name.
String getScheme( )	Returns the transmission scheme of the URL used for the request (for example, "http", "ftp").
String getServerName( )	Returns the name of the server.
int getServerPort( )	Returns the port number.

TABLE 31-3 Various Methods Defined by **ServletRequest**

ServletResponse

Method	Description
String getCharacterEncoding( )	Returns the character encoding for the response.
ServletOutputStream getOutputStream( ) throws IOException	Returns a **ServletOutputStream** that can be used to write binary data to the response. An **IllegalStateException** is thrown if **getWriter( )** has already been invoked for this request.
PrintWriter getWriter( ) throws IOException	Returns a **PrintWriter** that can be used to write character data to the response. An **IllegalStateException** is thrown if **getOutputStream( )** has already been invoked for this request.
void setContentLength(int *size*)	Sets the content length for the response to *size*.
void setContentType(String *type*)	Sets the content type for the response to *type*.

TABLE 31-4 Various Methods Defined by **ServletResponse**

S

GenericServlet

The GenericServlet Class

The **GenericServlet** class provides implementations of the basic life cycle methods for a servlet. **GenericServlet** implements the **Servlet** and **ServletConfig** interfaces. In addition, a method to append a string to the server log file is available. The signatures of this method are shown here:

```
void log(String s)
void log(String s, Throwable e)
```

Here, *s* is the string to be appended to the log, and *e* is an exception that occurred.

ServletInputStream

The ServletInputStream Class

The **ServletInputStream** class extends **InputStream**. It is implemented by the servlet container and provides an input stream that a servlet developer can use to read the data from a client request. It defines the default constructor. In addition, a method is provided to read bytes from the stream. It is shown here:

```
int readLine(byte[ ] buffer, int offset, int size) throws IOException
```

Here, *buffer* is the array into which *size* bytes are placed starting at *offset*. The method returns the actual number of bytes read or –1 if an end-of-stream condition is encountered.

ServletOutputStream

The ServletOutputStream Class

The **ServletOutputStream** class extends **OutputStream**. It is implemented by the servlet container and provides an output stream that a servlet developer can use to write data to a client response. A default constructor is defined. It also defines the **print( )** and **println( )** methods, which output data to the stream.

ServletException	**The Servlet Exception Classes** **javax.servlet** defines two exceptions. The first is **ServletException**, which indicates that a servlet problem has occurred. The second is **UnavailableException**, which extends **ServletException**. It indicates that a servlet is unavailable.
UnavailableException	

1. Describe the classes and interfaces of javax.servlet.http package.

3) Interfaces:

Interface	Description
HttpServletRequest	Enables servlets to read data from an HTTP request.
HttpServletResponse	Enables servlets to write data to an HTTP response.
HttpSession	Allows session data to be read and written.
HttpSessionBindingListener	Informs an object that it is bound to or unbound from a session.

4) Classes:

Class	Description
Cookie	Allows state information to be stored on a client machine.
HttpServlet	Provides methods to handle HTTP requests and responses.
HttpSessionEvent	Encapsulates a session-changed event.
HttpSessionBindingEvent	Indicates when a listener is bound to or unbound from a session value, or that a session attribute changed.

T

Interface
HttpServletRequest
HttpServletResponse

Method	Description
String getAuthType( )	Returns authentication scheme.
Cookie[ ] getCookies( )	Returns an array of the cookies in this request.
long getDateHeader(String *field*)	Returns the value of the date header field named *field*.
String getHeader(String *field*)	Returns the value of the header field named *field*.
Enumeration getHeaderNames( )	Returns an enumeration of the header names.
int getIntHeader(String *field*)	Returns the **int** equivalent of the header field named *field*.
String getMethod( )	Returns the HTTP method for this request.
String getPathInfo( )	Returns any path information that is located after the servlet path and before a query string of the URL.
String getPathTranslated( )	Returns any path information that is located after the servlet path and before a query string of the URL after translating it to a real path.
String getQueryString( )	Returns any query string in the URL.
String getRemoteUser( )	Returns the name of the user who issued this request.
String getRequestedSessionId( )	Returns the ID of the session.
String getRequestURI( )	Returns the URI.
StringBuffer getRequestURL( )	Returns the URL.
String getServletPath( )	Returns that part of the URL that identifies the servlet.
HttpSession getSession( )	Returns the session for this request. If a session does not exist, one is created and then returned.
HttpSession getSession(boolean *new*)	If *new* is **true** and no session exists, creates and returns a session for this request. Otherwise, returns the existing session for this request.
boolean isRequestedSessionIdFromCookie( )	Returns **true** if a cookie contains the session ID. Otherwise, returns **false**.
boolean isRequestedSessionIdFromURL( )	Returns **true** if the URL contains the session ID. Otherwise, returns **false**.
boolean isRequestedSessionIdValid( )	Returns **true** if the requested session ID is valid in the current session context.

TABLE 31-5 Various Methods Defined by **HttpServletRequest**

Method	Description
void addCookie(Cookie *cookie*)	Adds *cookie* to the HTTP response.
boolean containsHeader(String *field*)	Returns **true** if the HTTP response header contains a field named *field*.
String encodeURL(String *url*)	Determines if the session ID must be encoded in the URL identified as *url*. If so, returns the modified version of *url*. Otherwise, returns *url*. All URLs generated by a servlet should be processed by this method.
String encodeRedirectURL(String *url*)	Determines if the session ID must be encoded in the URL identified as *url*. If so, returns the modified version of *url*. Otherwise, returns *url*. All URLs passed to **sendRedirect( )** should be processed by this method.

TABLE 31-6 Various Methods Defined by **HttpServletResponse**

Method	Description
void sendError(int *c*) throws IOException	Sends the error code *c* to the client.
void sendError(int *c*, String *s*) throws IOException	Sends the error code *c* and message *s* to the client.
void sendRedirect(String *url*) throws IOException	Redirects the client to *url*.
void setDateHeader(String *field*, long *msec*)	Adds *field* to the header with date value equal to *msec* (milliseconds since midnight, January 1, 1970, GMT).
void setHeader(String *field*, String *value*)	Adds *field* to the header with value equal to *value*.
void setIntHeader(String *field*, int *value*)	Adds *field* to the header with value equal to *value*.
void setStatus(int *code*)	Sets the status code for this response to *code*.

TABLE 31-6 Various Methods Defined by **HttpServletResponse** *(continued)*

HttpSession

Method	Description
Object getAttribute(String *attr*)	Returns the value associated with the name passed in *attr*. Returns **null** if *attr* is not found.
Enumeration getAttributeNames( )	Returns an enumeration of the attribute names associated with the session.
long getCreationTime( )	Returns the time (in milliseconds since midnight, January 1, 1970, GMT) when this session was created.
String getId( )	Returns the session ID.
long getLastAccessedTime( )	Returns the time (in milliseconds since midnight, January 1, 1970, GMT) when the client last made a request for this session.
void invalidate( )	Invalidates this session and removes it from the context.
boolean isNew( )	Returns **true** if the server created the session and it has not yet been accessed by the client.
void removeAttribute(String *attr*)	Removes the attribute specified by *attr* from the session.
void setAttribute(String *attr*, Object *val*)	Associates the value passed in *val* with the attribute name passed in *attr*.

TABLE 31-7 The Methods Defined by **HttpSession**

HttpSessionBindingListener

Z

Class

Cookie

Method	Description
Object clone( )	Returns a copy of this object.
String getComment( )	Returns the comment.
String getDomain( )	Returns the domain.
int getMaxAge( )	Returns the maximum age (in seconds).
String getName( )	Returns the name.
String getPath( )	Returns the path.
boolean getSecure( )	Returns **true** if the cookie is secure. Otherwise, returns **false**.
String getValue( )	Returns the value.
int getVersion( )	Returns the version.
void setComment(String *c*)	Sets the comment to *c*.
void setDomain(String *d*)	Sets the domain to *d*.
void setMaxAge(int *secs*)	Sets the maximum age of the cookie to *secs*. This is the number of seconds after which the cookie is deleted.
void setPath(String *p*)	Sets the path to *p*.
void setSecure(boolean *secure*)	Sets the security flag to *secure*.
void setValue(String *v*)	Sets the value to *v*.
void setVersion(int *v*)	Sets the version to *v*.

TABLE 31-8 The Methods Defined by **Cookie**

HttpServlet

Method	Description
void doDelete(HttpServletRequest *req*, HttpServletResponse *res*) throws IOException, ServletException	Handles an HTTP DELETE request.
void doGet(HttpServletRequest *req*, HttpServletResponse *res*) throws IOException, ServletException	Handles an HTTP GET request.
void doHead(HttpServletRequest *req*, HttpServletResponse *res*) throws IOException, ServletException	Handles an HTTP HEAD request.
void doOptions(HttpServletRequest *req*, HttpServletResponse *res*) throws IOException, ServletException	Handles an HTTP OPTIONS request.
void doPost(HttpServletRequest *req*, HttpServletResponse *res*) throws IOException, ServletException	Handles an HTTP POST request.
void doPut(HttpServletRequest *req*, HttpServletResponse *res*) throws IOException, ServletException	Handles an HTTP PUT request.
void doTrace(HttpServletRequest *req*, HttpServletResponse *res*) throws IOException, ServletException	Handles an HTTP TRACE request.
long getLastModified(HttpServletRequest *req*)	Returns the time (in milliseconds since midnight, January 1, 1970, GMT) when the requested resource was last modified.
void service(HttpServletRequest *req*, HttpServletResponse *res*) throws IOException, ServletException	Called by the server when an HTTP request arrives for this servlet. The arguments provide access to the HTTP request and response, respectively.

TABLE 31-9 The Methods Defined by **HttpServlet**

HttpSessionEvent

The HttpSessionEvent Class

HttpSessionEvent encapsulates session events. It extends **EventObject** and is generated when a change occurs to the session. It defines this constructor:

HttpSessionEvent(HttpSession *session*)

Here, *session* is the source of the event.

HttpSessionEvent defines one method, **getSession( )**, which is shown here:

HttpSession getSession( )

It returns the session in which the event occurred.

HttpSessionBindingEvent

The HttpSessionBindingEvent Class

The **HttpSessionBindingEvent** class extends **HttpSessionEvent**. It is generated when a listener is bound to or unbound from a value in an **HttpSession** object. It is also generated when an attribute is bound or unbound. Here are its constructors:

HttpSessionBindingEvent(HttpSession *session*, String *name*)
HttpSessionBindingEvent(HttpSession *session*, String *name*, Object *val*)

Here, *session* is the source of the event, and *name* is the name associated with the object that is being bound or unbound. If an attribute is being bound or unbound, its value is passed in *val*.

The **getName( )** method obtains the name that is being bound or unbound. It is shown here:

String getName( )

The **getSession( )** method, shown next, obtains the session to which the listener is being bound or unbound:

HttpSession getSession( )

The **getValue( )** method obtains the value of the attribute that is being bound or unbound. It is shown here:

Object getValue( )