**Web Technologies Assignment**

**17251A05B0**

|  |  |
| --- | --- |
| **1a)** | **What does Servlet Config interface do? Which objects it receives, when the servlet accepts a call from client?**  An object of servlet config is created by the web container for each servlet. This object can be used to get information from a web.xml file. The core advantage of servlet config is that you need not edit the servlet file, if information is modified in web.xml file.  **Methods of ServletConfig:**   * **String getInitParameter(String name):** returns a String value initialized parameter, or NULL if the parameter does not exist. * **Enumeration 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. * **ServletContextgetServletContext():** returns a reference to the ServletContext * **String getServletName():** returns the name of the servlet instance   When a servlet accepts a service call from a client, it receives two objects, ServletRequest and ServletResponse . The ServletRequest class encapsulates the communication from the client to the server, while the ServletResponse class encapsulates the communication from the servlet back to the client. |
| **1b)** | **What is session tracking? Explain. Write a session tracker that tracks the number of accesses and last access data of a particular web page.**  Session Tracking is a way to maintain state (data) of an user. It is also known as session management in servlet.  Http protocol is a stateless so we need to maintain state using session tracking techniques. Each time user requests to the server, server treats the request as the new request. So we need to maintain the state of an user to recognize to particular user.  HTTP is stateless that means each request is considered as the new request. Session tracking is used to recognize the user It is used to recognize the particular user.  **Session Tracking Techniques**  **There are four techniques used in Session tracking:**   1. Cookies 2. Hidden Form Field 3. URL Rewriting 4. HttpSession   **Cookies in Servlet**  A cookie is a small piece of information that is persisted between the multiple client requests.  A cookie has a name, a single value, and optional attributes such as a comment, path and domain qualifiers, a maximum age, and a version number.  There are 2 types of cookies in servlets.   1. Non-persistent cookie 2. Persistent cookie   Non-persistent cookie:    It is valid for single session only. It is removed each time when user closes the browser.  Persistent cookie  It is valid for multiple session . It is not removed each time when user closes the browser. It is removed only if user logout or signout.  **2)Hidden Form Field**  In case of Hidden Form Field a hidden (invisible) textfield is used for maintaining the state of an user.  In such case, we store the information in the hidden field and get it from another servlet. This approach is better if we have to submit form in all the pages and we don't want to depend on the browser.  **Syntax:**  <input type="hidden" name="uname" value="web technologies ">  **3)URL Rewriting**  In URL rewriting, we append a token or identifier to the URL of the next Servlet or the next resource. We can send parameter name/value pairs using the following format:  url?name1=value1&name2=value2&??  A name and a value is separated using an equal = sign, a parameter name/value pair is separated from another parameter using the ampersand(&). When the user clicks thehyperlink, the parameter name/value pairs will be passed to the server. From a Servlet, we can use getParameter() method to obtain a parameter value.  **HttpSession interface**  In such case, container creates a session id for each user. The container uses this id to identify the particular user. An object of HttpSession can be used to perform two tasks:   1. bind objects 2. view and manipulate information about a session, such as the session identifier, creation time, and last accessed time.   The HttpServletRequest interface provides two methods to get the object of HttpSession :   1. **public HttpSessiongetSession():**Returns the current session associated with this request, or if the request does not have a session, creates one. 2. **public HttpSessiongetSession(boolean create):**Returns the current HttpSession associated with this request or, if there is no current session and create is true, returns a new session.   **PROGRAM:**  import java.io.\*;  import javax.servlet.\*;  import javax.servlet.http.\*;  public class SessionTracker extends HttpServlet{  public void doGet(HttpServletRequestreq, HttpServletResponse res) throws  ServletException, IOException{  res.setContentType(“text/html”);  PrintWriter out = res.getWriter();  HttpSession session = req.getSession(true);  Integer count=(Integer)session.getValue(“tracker.count”);  if(count==null)           count=new Integer(1);  else           count=new Integer(count.intValue()+1);  session.putValue(“tracker.count”,count);  out.println(“<html><head><title>SessionTracker</title></head>”);  out.println(“<body><h1>Session Tracking Demo</h1>”);  out.println(“You have visisted this page ”+count+((count.intValue()==1)?”time.”:”times.”));  out.println(“<p>”);  out.println(“<h2>Here is the session data:</h2>”);  String[ ] names = session.getValueNames( );  for(int i=0;i<names.length;i++){  out.println(names[i]+”:”+session.getValue(names[i])+<br>”);  }  out.println(“</p></body></html>”);  }  } |

|  |  |
| --- | --- |
| **2)a** | **Write a JSP to demonstrate the page and include directives. How the data can be shared between JSP pages**  AJSP page may contain instructions to be used by the JSP container to indicate how this page is interpreted and executed. Those instructions are called directives.  They are enclosed within <%@ and %> tags.  **Page directive:**  The page directive defines attributes that apply to an entire JSP page.  **Attributes of JSP page directive:**  **Import:** Consists of list of class names separated by commas.  Eg: <%@ page import="java.util.Date,java.io.\*" %>  **contentType:**  The contentType attribute defines the MIME(Multipurpose Internet Mail Extension) type of the HTTP response.The default value is "text/html;charset=ISO-8859-1".  **extends:**  The extends attribute defines the parent class that will be inherited by the generated servlet.It is rarely used.  **Info:**  This attribute simply sets the information of the JSP page which is retrieved later by using getServletInfo() method of Servlet interface.  **Buffer:**  The buffer attribute sets the buffer size in kilobytes to handle output generated by the JSP page.The default size of the buffer is 8Kb.  **Language:**  The language attribute specifies the scripting language used in the JSP page. The default value is "java".  **isThreadSafe:**  Servlet and JSP both are multithreaded.If you want to control this behaviour of JSP page, you can use isThreadSafe attribute of page directive.The value of isThreadSafe value is true.If you make it false, the web container will serialize the multiple requests, i.e. it will wait until the JSP finishes responding to a request before passing another request to it.If you make the value of isThreadSafe attribute like:  <%@ page isThreadSafe="true| false" %>  **autoFlush:**  This specifies if buffer should be flushed automatically(true) when it is full.  **session:**  This attribute indicates if JSP page requires a HTTP session  **errorPage:**  The errorPage attribute is used to define the error page, if exception occurs in the current page, it will be redirected to the error page.  **isErrorPage:**  The isErrorPage attribute is used to declare that the current page is the error page.  **Multiple attributes** can also be used but in this case no attribute other than import should appear twice.  **Eg:**  <%@ page  [language=”java”]  [import=”java.io.\*,java.util.\*”]  [session=”true”]  [buffer=”8kb”]  [autoFlush=”true”]  [info=”text”]  [isThreadSafe=”true”]  [isErrorPage=”falsee”]  %>  **Include directive:**  The include directive includes the original content of the included resource at page translation time (the jsp page is translated only once so it will be better to include static resource).  Eg: <%@ include file="resourceName" %>  **Sharing data between JSP pages:**  All JSP pages participate in an HTTP session unless the session attribute is set to false. An HTTP session is represented by implicit object session.  For examplelogin.jsp page may store the user name in session while pages like home.jsp can use it.  **Consider the code below:**  **Login.jsp:**  String user=request.getParameter(“user”);  Session.setAttribute(“user”,user);  **Home.jsp:**  String user=(String)session.getAttribute(“user”); |
| **2)b** | **What are the problems with servlet? Write in brief about JSP anatomy?**    **Problems with Servlet:**   1. Not useful for generating presentational content such as HTML. 2. We need to recompile the source ourselves if any modification is done 3. If presentation logic is changed the servlet code has to be modified, recompiled and redeployed. 4. Contains both presentation and processing logic which makes the code difficult to understand.   **Anatomy of JSP:**  A JSP page basically consists of two parts: HTML/XML markups and JSP constructs.  A large part of JSP pages consists of static HTML/XML contents known as template text.  We use 3 types of JSP constructs in a typical JSP page: scripting elements, directives and actions.  Java code that will become integral part of the resultant servlet is inserted using scripting elements. directives let us control overall structure and behaviour of resultant servlet. Action allows us to use existing components and otherwise control the behaviour of JSP engine.  There are again 3 types: scriptlets, declarations and expressions.  Scriptlets allows us to insert any Java-server relevant API in the HTML/XML page. Declarations allow us to declare methods and variables. Expressions are used to print the values of Java expressions. |
| **3)a** | **How a servlet does communicate with a JSP page? Explain.**  When a servlet jsp communication is happening, it is not just about forwarding the request to a JSP from a servlet. There might be a need to transfer a string value or on object itself.  getServletConfig().getServletContext().getRequestDispatcher(“jspfilepathtoforward”).forward(request, response). This line is the essence for communication.  Following is a servlet and JSP source code example to perform Servlet JSP communication. Wherein an object will be communicated to a JSP from a Servlet.  **Following are the steps in Servlet JSP Communication:**  1. Servlet instantiates a bean and initializes it.  2. The bean is then placed into the request  3. The call is then forwarded to the JSP page, using request dispatcher.  **Example Servlet Source Code: (ServletToJSP.java)**  public class ServletToJSP extends HttpServlet {                   public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException{                 String message = "Example source code of Servlet to JSP communication.";                 request.setAttribute("message", message);  Vector vecObj = new Vector();                          vecObj.add("Servlet to JSP communicating an object");  request.setAttribute("vecBean",vecObj);                         RequestDispatcherreqDispatcher =                          getServletConfig().getServletContext().getRequestDispatcher("/jsp/java.jsp");  reqDispatcher.forward(request,response);     }}  **Example JSP Source Code: (java.jsp)**  <html>  <body>  <%  String message = (String) request.getAttribute("message");  out.println("Servlet communicated message to JSP: "+ message);  Vector vecObj = (Vector) request.getAttribute("vecBean");  out.println("Servlet to JSP communication of an object: "+vecObj.get(0));  %>  </body>  </html> |
| **3)b** | **What is a bean? Discuss how to create beans in JSP.**  Java beans are reusable Java components that allows us to separate business logic from presentational logic. Technically a Java bean class meets the following requirements.   * It is public, no argument constructor. * It implements java.io.Serializable or java.io.Externalizable interface. * Its properties are accessible using methods that are written following a naming convention.   The three action elements used are:  1)useBean:  Initializes javabean object into JSP page.  <jsp: useBean id = "object\_name" class=”class\_name”scope = "page| request| session| application" />  2)setProperty:  assigns a new value to the specified property of the specified bean object  <jsp:setProperty name=”obj\_name” property = "property name" value=”prop\_value”/>  3)getProperty:  retrieves value to the specified property of the specified bean object  <jsp:getProperty name=”obj\_name” property = "property name" />  **Example:**  StudentsBean.java  public class StudentsBean implements java.io.Serializable {  private String firstName = null;  private String lastName = null;  private int age = 0;  public StudentsBean() {  }  public String getFirstName(){  return firstName;  }  public String getLastName(){  return lastName;  }  public int getAge(){  return age;  }  public void setFirstName(String firstName){  this.firstName = firstName;  }  public void setLastName(String lastName){  this.lastName = lastName;  }  public void setAge(Integer age){  this.age = age;  }  }  Sample.jsp  <html>  <head>  <title>get and set properties Example</title>  </head>  <body>  <jsp:useBean id = "students" class = "com.tutorialspoint.StudentsBean">  <jsp:setProperty name = "students" property = "firstName" value = "Zara"/>  <jsp:setProperty name = "students" property = "lastName" value = "Ali"/>  <jsp:setProperty name = "students" property = "age" value = "10"/>  </jsp:useBean>  <p>Student First Name:  <jsp:getProperty name = "students" property = "firstName"/>  </p>  <p>Student Last Name:  <jsp:getProperty name = "students" property = "lastName"/>  </p>  <p>Student Age:  <jsp:getProperty name = "students" property = "age"/>  </p>  </body>  </html>  **Output is:**  Student First Name: Zara  Student Last Name: Ali  Student Age: 10 |
| **4)a** | **Discuss Event handlers in javascript. Validate registration page( name, password ,phno, email) with Regular expressions.**  JavaScript is used to add Interactivity to web pages. This interactivity is provided by responding to user actions called events so Javascript identifies these events and takes an action by executing some piece of code. The procedure of taking actions is called Event Handling. The specific piece of code that takes the action is called event handler. This event handler can be interactive or non-interactive based on the events it handles. Event acts like a trigger so when that occurs the corresponding event handler is executed.  The name of the property for the event evnt takes the from*onevnt(case insensitive).*   <input type=”button” value=”clickme**” onClick***=”alert(‘You Clicked’*);”>  *”alert(‘You Clicked’);  -> Event Handler* **onClick-Event**  **Example demonstrates how to work with a load Event:**  <html>  <head><title> On Load Event</title></head>  <body onloaded=”frmLogin.login.focus();”>  <form name=”frmLogin”>  Login:<input type=”text” name=”login”><br>  Password:<input type=”password” name=”password”><br>  <input type=”submit” name=”submit” value=”Submit”>  <input type=”reset” name=”reset” value=”Reset”>  </form>  </body>  </html>   |  |  | | --- | --- | |  | **Event Object:**  This is available only when a event occurs. This event object encapsulates the state of an event such as location of mouse pointer and left or right button pressed for a mouse event, Unicode of the key pressed for a key event, and so on. So only the event handlers can access this event object.  In case of Nested events the order in which the outer and inner event handlers are triggered depends on event Propagation Model - Event bubbling and event capturing.  **Two phases of Propagation Model:**  Event Capturing - It starts from outermost source element and propagates downwards until it reaches innermost source element.  After reaching the innermost source element it changes the direction and  moves up until it reaches the outermost element. – Event Bubbling  All Browsers support registering a handler for bubbling phase.Code registers someHandler for the event someEvent on someElement for the bubbling phase. Similarly the following code does essentially the same thing.  <someElementonsomeEvent=”someHandler”>  addEventListener() on an element that can be used to register a handler for both the capturing and bubbling phases.  someElement.addEventListener(‘eventName’,eventHandler,flag)  if the flag==true  it is capturing phase else it is bubbling phase.  Event Capturing Example: |       <html>      <head>      <title>Event Capturing Demo</title>       <script language=”JavaScript”>        function handler\_body()       {  alert(‘You clicked on body’);}  function handler\_p(){  alert(‘You clicked on p’);  }  function handler\_div(){  alert(‘You  clicked on div’);}  function register(){  document.getElementById(“para1”).addEventListener(‘click’,handler\_p,true);  document.getElementById(“div1”).addEventListener(‘click’,handler\_div,true);  document.body.addEventListener(‘click’,handler\_body,true);  }To prevent the event propagation inorder to turn off to avoid interference among methods by setting cancelBubble property of the event object to true i.e; event.cancelBubble =true; |
| **4)b** | **Explain AJAX working process with a block diagram and example**  **BLOCK DIAGRAM:**  AJAX stands for Asynchronous JavaScript and XML. It is a set of web development techniques which allow web applications to work asynchronously – processing any requests to the server in the background.  how ajax works, flow of ajax  In the above image, XMLHttpRequest object plays a important role.  1. User sends a request from the UI and a javascript call goes to  XMLHttpRequest object.  2. HTTP Request is sent to the server by XMLHttpRequest object.  3. Server interacts with the database using JSP, PHP, Servlet, ASP.net  etc.  4. Data is retrieved.  5. Server sends XML data or JSON data to the XMLHttpRequestcallback  function.  6. HTML and CSS data is displayed on the browser.  The best part of AJAX is that it allows users to fetch data from the web server asynchronously,without any need to reload the page.  The HTTP protocol follows the client-server model and is a request-response protocol.The main aim of a server is to allocate resources to the client on the request.  The main advantage is that a client can keep on sending requests to the server and is interrupted only when it receives a response.So,at the client’s side, a small javScript code is returned which on receiving the data from the server,modifies that part of the webpage only.  The 3 steps to be followed to write an AJAX based application are:  1.Create an XMLHttpRequest object:  It forms the heart of the AJAX application.  The code is as below:  try {//for non-IE browsers  xmlHttp=new XMLHTTPRequest();  } catch(e1)  { xmlHttp=new ActiveObject(“Microsoft.XMLHTTP”);  }  2.Specify a handler:  The handler will receive data sent back by the server,extract it and perform the desired function.  Code:  Function handler(){//some code}  xmlHttp.onreadystatechange=handler();  3.AJAX readyState property:  The diff. States with their values are:  i)uninitialised:0  ii)connection established:1  iii)request sent:2  iv)processing:3  v)complete and response is ready:4  Code:  Function handler()  { if(xmlHttp.readyState==4)  {//some code}  Else  {//some code}  } |
|  |  |
| **5a** | **Illustrate about cookies with example. Give the limitations**  Cookies are small files which are stored on a user’s computer. They are used to hold a modest amount of data specific to a particular client and website and can be accessed either by the web server or by the client computer  When cookies were invented, they were basically little documents containing information about you and your preferences. For instance, when you select your language in which you want to view your website, the website would save the information in a document called a cookie on your computer, and the next time when you visit the website, it would be able to read a cookie saved earlier. That way the website could remember your language and let you view the website in your preferred language without having to select the language again.  A cookie can contain any type of information such as the time when you visited the website, the items that you added into your shopping basket, all the links you clicked in website, etc.  The **cookie constructor** is as:  Cookie(string key,String value)  **Program:**  import java.io.IOException;  import java.io.PrintWriter;  import javax.servlet.ServletException;  import javax.servlet.annotation.WebServlet;  import javax.servlet.http.Cookie;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  @WebServlet("/CookieDemo")  public class CookieDemo extends HttpServlet  {  public void doGet(HttpServletRequest request,  HttpServletResponse response) throws ServletException, IOException  {  response.setContentType("text/html");  PrintWriter out = response.getWriter();  //Creating a cookie object.  Cookie cookie = new Cookie("user","1234");  //Setting the maximum age to 1 hour  cookie.setMaxAge(60\*60);  //Send the cookie to the client  response.addCookie(cookie);  out.println("Cookie created");  }  This program allows you to add cookies to the client using servlets.  **Limitations of cookies:**  ● A cookie cannot be used by any other server as the id saved in your cookie  is directly mapped to the website’s database.  ● A cookie can never be used to access any information saved in your system,  browser or hard disk.  ● A cookie cannot be used to deliver viruses or any other threats. |
| **5)b** | **Read the journal paper titled”**[**Impact**](http://ieeexplore.ieee.org/document/5359753/) **of AJAX in web applications.” and write short notes on it.**  AJAX is mainly used to allow asynchronous communication i.e loading time of a page is drastically reduced. AJAX stands for Asynchronous JavaScript and XML.It is popular with Google,Amazon,etc;  AJAX applications eliminate the start-stop start-stop nature of traditional web pages .AJAX allows pages to request small bits of information from the server instead of entire pages.  The main ideas behind AJAX:  1.Data Object model  2.data interchange through XML.  3.Asynchronous communication with XMLHttpRequest object  The writer talks about how they developed an application harnessing various features of AJAX n a login page which is responsive and had password security,similarly,it provides text formatting for web applications.It also makes the look and feel of the web application good.  AJAX provides instantaneous reactions and is a true application model.The main advantage is it provides flexibility to users as well as developers.The paper concludes that AJAX has a very high impact on Rich Client applications. |