ELEMENTOS DE UN JSP

ELEMENTOS DE UN JSP

- Expressions:
  - Sintaxis: <%= expresion %>
  - La expresión se evalúa y ser inserta en la salida del servlet
  - Es equivalente a out.println(expresion)
- Scriptlets:
  - Sintaxis: <% codigoJava %>
  - El código Java se inserta en el método service() del Servlet generado
  - Puede ser cualquier código Java válido
- Declaraciones:
  - Sintaxis: <%! codigoJava %>
  - Se utiliza para agregar código a la clase del Servlet generado
  - Se pueden declarar variables o métodos que pertenecen a la clase
- Sintaxis XML:
  - Cada elemento JSP tiene su equivalente en sintaxis XML
  - Esta sintaxis se utiliza para tener una mayor compatibilidad, por ejemplo, con herramientas visuales

Vamos a revisar los elementos básicos de un JSP. Un JSP se va compone de Expresiones, Scriptlets, Declaraciones y también existe una característica en la cual vamos a manejar una sintaxis equivalente en XML.

Una Expresión tiene la siguiente sintaxis: Sintaxis <%=expresión%> Lo que vamos a hacer es utilizar la sintaxis **<%=** para abrir nuestra expresión, posteriormente agregamos una expresión que puede ser cualquier expresión valida que a final de cuentas va a ser equivalente a poner la sintaxis out.println y lo que tengamos en nuestra expresión lo va a mandar al cliente. Finalmente cerramos la expresión con **%>** Este tipo de sintaxis **<%=** expresión **%>** se conoce como un tag o etiqueta de expresión. La expresión puede ser un valor como puede ser una cadena, una expresión matemática, una suma, una resta etc. o también puede ser el resultado de una llamada a una función, pero si mandamos a llamar una función el resultado de esta función no puede ser de tipo void, debe de regresar algún valor ya que como podemos observar el resultado de evaluar esta expresión se va a mandar a nuestro cliente entonces por ello esta expresión debe de regresar algún resultado.

Otro elemento de los JSPs son los Scriptlets. Un Scriptlet contiene código Java a diferencia de una expresión que únicamente la utilizamos para imprimir información a nuestro cliente, un Scriptlets puede contener código Java que no necesariamente va a visualizar nuestro cliente debido a que si este código Java no manda imprimir nada al cliente, entonces nuestro cliente no va a visualizar la información que este procesando. Este código Java puede ser bastante robusto y por ello hay que tener cuidado respecto al código que manejemos con los Scriptlets, ya que aunque nos permiten agregar mucho código y puede ser muy poderoso para agregar código Java eso también puede ser contraproducente sobre todo para el mantenimiento de un JSPs, por lo que demasiado código en un scriptlet se considera una mala práctica.

La sintaxis de un Scriptlet es **<%** codigoJava **%>** a diferencia de una expresión no tiene un signo de igual. Inicia de igual manera con **<%** y el cierre para el tag de un Scriptlets es el mismo, entonces la sintaxis es: **<%** codigoJava **%>** Es importante mencionar que el código generado dentro de un Scriptlets se inserta dentro del método service() del servlet generado a partir del JSP que se haya compilado, posteriormente vamos a ver algunos ejemplos para observar y poder entender este párrafo. Los Scriptlets pueden contener cualquier código Java válido, es decir, puede tener declaraciones de variables, llamadas a funciones y en general puede tener cualquier lógica siempre y cuando respetemos que este código está dentro de otro método, en este caso del método service().

En los JSPs también vamos a manejar los tags de declaraciones y la sintaxis es: **<%!** codigoJava **%>** . En las declaraciones también es código Java, pero este código lo vamos a utilizar para declarar ya sea variables o métodos que pertenezcan a la clase del servlet generado, y a diferencia del código Java que está en un Scriptlets en el cual todas las variables que declaremos son locales al método service(), en el caso de las declaraciones el código que agreguemos, si es que declaramos una variable, se vuelve una variable de instancia debido a que se está agregando como una variable de la clase del servlet generado y no únicamente como una variable local a cierto método. También los métodos que agreguemos se vuelven métodos que son parte de la clase del servlet generado aunque no es muy utilizado este concepto de declaraciones ya que lo más común es que desde un Scriptlets mandemos a llamar funciones o utilicemos variables que están definidas en otras clases y no dentro de nuestro JSP, entonces en el caso de las declaraciones lo vamos a utilizar siempre y cuando necesitemos agregar ya sea variables o métodos que pertenezcan a nuestro servlet generado a partir del JSP.

Por último, cada uno de nuestros componentes, como expresiones, Scriptlets y las declaraciones tienen su equivalente en sintaxis XML. Esto vamos a ver que tiene varias aplicaciones en el sentido de que vamos a tener mayor compatibilidad si es que nuestros JSPs los generamos con sintaxis XML y no con la sintaxis estándar. Cuando revisemos el tema de las sintaxis XML vamos a ver las equivalencias para cada uno de estos elementos.

VARIABLES IMPLÍCITAS EN LOS JSP´S

- **request**:
  - Es el objeto HTTPServletRequest
- **response**:
  - Es el objeto HttpServletResponse
- **out**:
  - Es el objeto JspWriter (equivalente a PrintWriter)
- **session**:
  - Es el objeto HttpSession asociado con el objeto request
  - Se puede deshabilitar el uso de sesiones en un JSP
- **application**:
  - Es el objeto ServletContext que se obtiene a partir del método getServletContext() en un Servlet

Vamos a revisar a continuación el tema de las variables implícitas en un JSP dentro del uso de los Scriptlets.

Dentro de un JSP podemos utilizar objetos que ya están instanciados para que podamos usarlos de manera inmediata. Por ejemplo tenemos los objetos request, response, out, el objeto sesión, application entre otros. El objeto request es el mismo objeto que hemos trabajado por ejemplo en los Servlets cuando sobreescribimos el método doGet o doPost recibimos tanto el objeto request como el objeto response como parámetros. Entonces estos objetos ya los tenemos disponibles en un JSP y no tenemos que instanciarlos, simplemente ponemos el nombre de nuestra variable y por medio del operador punto (.) podemos acceder a los atributos y métodos del objeto del tipo HTTPServletRequest. Lo mismo sucede con el método response, simplemente con este nombre de variable podemos acceder a los atributos y métodos de este objeto HTTPServletResponse.

La variable out es equivalente al objeto PrintWriter que hemos instanciado dentro de los Servlets, en este caso existe una pequeña variante ya que esta variable out es de tipo JspWriter.

La variable session es una variable de tipo HttpSession y con esta variable podemos obtener información que ya hemos agregado a nuestra sesión, por ejemplo por medio de un Servlet. Vamos a ver más adelante por medio de un concepto llamado directivas que podemos deshabilitar el acceso a la sesión en caso de que no queramos que los JSPs manipulen directamente los atributos de una sesión o simplemente que no queramos que un JSP acceda a los atributos o modifique los elementos de una sesión.

Tenemos también la variable llamada application, esta es equivalente al objeto ServletContext, que en un servlet podemos obtener a partir del método getServletContext(). Este concepto lo vamos a ver posteriormente a más detalle.

En resumen, existen variables ya instanciadas que podemos acceder directamente en un JSP. Vamos a revisar a continuación algunos ejercicios para revisar estos conceptos básicos de los JSP’s.