Comunicación entre formularios HTML Y JSP

INTEGRANTES:
HERMILO ALMARAZ VARGAS
ALEXI DANIEL RAMIREZ RUIZ
GAMALIEL SILVA LOPEZ
MARIO HECXAI VALENCIA REYES
LUIS LENNIN VILLARREAL CASTRO

Formulario HTML

Un formulario HTML tiene la forma:

```
< form action =”destino” method=
“método”>
 elementos de formulario
 </form>
```

GET

Cuando se ocupa el método get la información se codifica directamente en la URL.

Con GET no podemos manejar grandes cantidades de información

JSP formulario de datos

getParameter (): Método de uso request.getParameter () para obtener el valor del parámetro de forma.

getParameterNames (): Este método se puede obtener los nombres de todas las variables

getInputStream (): Este método se llama para leer el flujo de datos binarios desde el cliente.

POST

Con el método post la información se envía directamente al servidor, no se codifica en la URL, y además permite el envío de grandes cantidades de información, como podrían ser archivos.

El método post introduce los parámetros en la solicitud HTTP para el servidor. Por ello, no quedan visibles para el usuario, la capacidad del método post es ilimitada.

VENTAJAS DE POST

En lo relativo a los datos, como, por ejemplo al rellenar formularios con nombres de usuarios y contraseñas, el método Post ofrece mucha discreción, los datos no se muestran en el caché ni tampoco en el historial de navegación. La flexibilidad del método Post también resulta muy útil: no solo se puede enviar textos cortos, sino también otros tipos de información, como fotos o vídeos.

DESVENTAJAS DE POST

Cuando un página web que contiene un formulario se actualiza (por ejemplo cuando se retrocede a la página anterior) los datos del formulario deben transferirse de nuevo. Por este motivo, existe el riesgo de que los datos se envíen varias veces por error, lo que en caso de una tienda online, puede dar a pedidos duplicados

Cuando utilizar Post o Get

El método post es aconsejable cuando el usuario debe enviar datos o archivos al servidor, como , por ejemplo, cuando se rellena un formulario o se suben fotos.

El método Get es adecuado para las personalización de páginas web: el usuario puede guardar búsquedas, configuraciones de filtros y ordenaciones de listas junto al URL como marcadores, de manera que en su próxima visita la página web se mostrará según sus preferencias.

En resumen

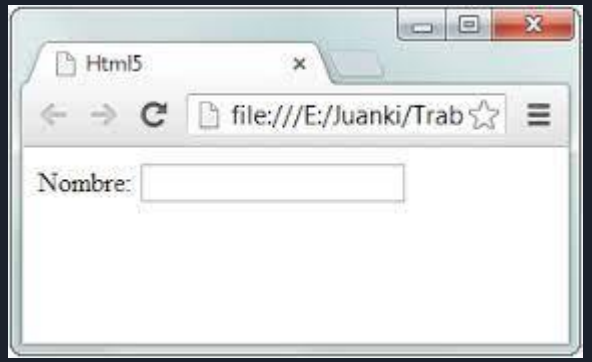
Get para la configuración de páginas web (filtros, ordenación, búsqueda, etc.)

Post para la transferencia de información y datos.

Elementos de formulario

Campos de texto

Son los tipos que se envían como texto simple.
La etiqueta <input> puede tomar varios valores diferentes en su atributo “type” para permitir al usuario introducir información de texto, además de otra etiqueta denominada <textarea> para cantidades de texto más grandes como varios párrafos.



```
    </p>

    <p><input type="submit" value="Enviar datos"></p>

</form>-->
<form name="form1" action="newjsp.jsp" method="POST">
        txt:
<input type="text" name="txt1" maxlength="10" size="15" />
        <br/>

<textarea name="textarea1" rows="4" cols="20">
        </textarea>
          <input value="Enviar" type="submit" /><!-- Enviar formulario -->
<br/>
```

```
<%
 //las variables que nos llegan son: txt1, txt2, radio1, ck1, select1, textarea1
 String txt1 = request.getParameter("txt1");
 out.println("text1: "+txt1+"<br/>");
 String txtarea = request.getParameter("textarea1");
 out.println("textarea1: "+txtarea+"<br/>");

 %>
```

Checkbox

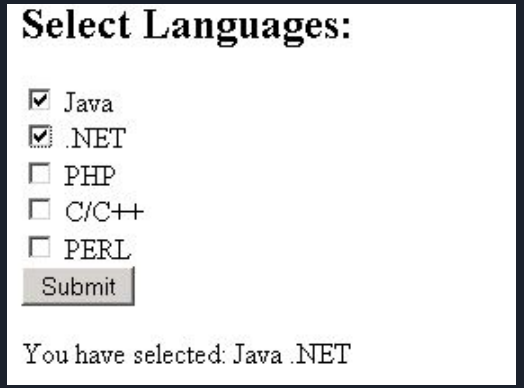
Es un método para que el usuario pueda interaccionar con el material de una página Web, activando un icono u otro elemento de entrada, consiste en unas cajas que permiten marcarlas o no para verificar alguna cosa en un formulario.

Los checkbox se consiguen con la etiqueta <INPUT type="checkbox">.

Con el atributo NAME de la etiqueta INPUT le podemos dar un nombre para luego acceder a ella con JSP.

Con el atributo CHECKED indicamos que el campo debe aparecer chequeado por defecto.

El atributo CHECKED es booleano, si colocamos ese atributo como indica que el elemento estará chequeado inicialmente, si el atributo no se encuentra en la etiqueta el elemento aparecerá desmarcado.



```
<form ACTION="newjsp.jsp">
<input type="checkbox" name="id" value="first"> first<BR>
<input type="checkbox" name="id" value="second"> second<BR>
<input type="checkbox" name="id" value="third"> third<BR>
<input type="checkbox" name="id" value="fourth"> fourth<BR>
<input type="checkbox" name="id" value="five"> fifth <BR>
<input type="submit" value="Submit">
</form>
```

```
<%
String s[] = request.getParameterValues("id");
if (s != null && s.length != 0) {
out.println("You have selected the option is: ");
for (int i = 0; i < s.length; i++) {
out.println(s[i] + "\n" + "Thank you");
}
}
%>
```

Implementación

Botón Radio

Botón radio: <input type="radio">

Los botones radio se crean mediante una etiqueta <input> cuyo atributo type tiene el valor radio.

Los botones radio que tiene el mismo atributo name forman un grupo, es decir, que si se marca uno de ellos se desmarca automáticamente el resto.

```
<input type="radio" name="boton" value="1">Opción 1<br>
<input type="radio" name="boton" value="2">Opción 2<br>
<input type="radio" name="boton" value="3">Opción 3
```

Si los atributos name no tiene el mismo valor, los botones radio son independientes (al marcar uno se desmarca el otro)

El atributo value contiene el valor que envía el formulario si el botón radio está marcado. Si el atributo value no está establecido el formulario envía on.

```
<input type="radio" name="boton" value="hombre"> Hombre<br>
<input type="radio" name="boton" value="mujer"> Mujer
```

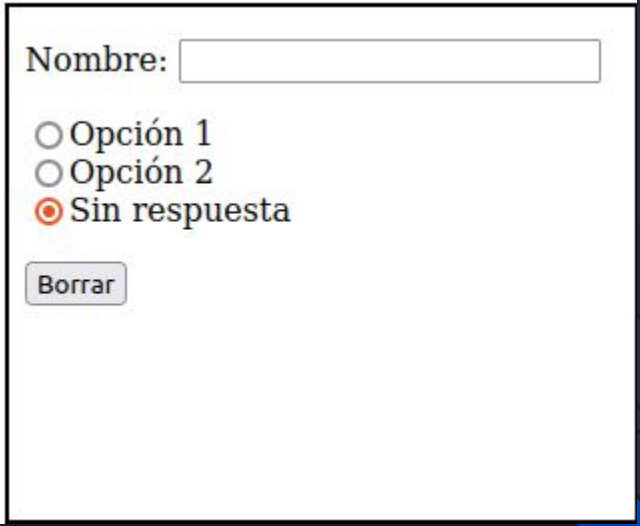
Si uno de los botones tiene el atributo checked con el valor checked, el botón aparece marcado de forma predeterminada.

```
<input type="radio" name="boton" value="1">Opción 1<br>
<input type="radio" name="boton" value="2" checked>Opción 2
```

Para desactivar una opción seleccionada desde un bloque de de botones radio es necesario utilizar un botón Reset.
El inconveniente es que el botón Reset reinicia todos los controles del formulario.
Recomendación: incluir un botón radio como opción neutra

```
<p>Nombre: <input type="text"
name="nombre"></p>

<p>
  <input type="radio" name="boton"
value="1">Opción 1<br>
  <input type="radio" name="boton"
value="2">Opción 2<br>
  <input type="radio" name="boton"
value="0" checked>Sin respuesta
</p>
```



<datalist>

La <datalist> etiqueta especifica una lista de opciones predefinidas para un <input>

```
<input list="browsers">

<datalist id="browsers">
  <option value="Internet Explorer">
  <option value="Firefox">
  <option value="Chrome">
  <option value="Opera">
  <option value="Safari">
</datalist>
```

Atributo type

El atributo type va dentro de la etiqueta <input> y su valor indicará que tipo de campo se está creando.

Valores de atributo type en formularios HTML

- Text: para campos de texto
- Password: para campos de contraseña
- Checkbox: para casillas de verificación
- Radio: para casillas de selección
- Submit: para botones de envío
- Reset: para botones de resetear
- File: para campos de selección de archivo

Ejemplo

HTML

```html
<form>
Nombre:<input type="text"/>  <br/><br/>
Apellidos:<input type="text"/>   <br/><br/>
Contraseña: <input type="password"/>  <br/><br/>
casilla de verificación: <input type="checkbox"/> <br/><br/>
casilla de selección: <input type="radio"/>  <br/><br/>
selección de archivo:  <input type="file">  <br/><br/>
botón de envio: <input type="submit">
</form>
```