Alumno: Uriel Santarelli

Curso: 4º CUATRIMESTRE

Turno: Vespertino

Año Lectivo: 2022

Asignatura: PROG. WEB

TP. Nº3

1. ¿Qué es un formulario web? Enumere todos sus atributos y para qué sirve cada uno de ellos

Los formularios se utilizan para recolectar información, hay de varios tipos, de registro o *login,* de restablecimiento de contraseña, para completar en mayor profundidad un perfil luego del registro, sus atributos son:

## **El formulario:** <form>

Un formulario es un conjunto de controles (botones, cajas de texto, casillas de verificación, botones radio, etc) que permiten al usuario introducir datos y enviarlos al servidor web para su procesamiento.

La etiqueta que delimita un formulario es la etiqueta *<form> ...</form>*. Los atributos más importantes de la etiqueta *<form>*son:

* action: contiene el nombre del agente que procesará los datos remitidos al servidor (por ejemplo, un script de PHP)
* method: define la manera de enviar los datos al servidor. Los valores posibles son:
  + get: los valores enviados se añaden a la dirección indicada en el atributo *action*
  + post: los valores se envían de forma separada

Si el atributo *method* no está establecido, el formulario se comporta como si el valor fuera *get*.

La etiqueta *<form>* es un elemento de bloque. En su interior puede haber cualquier elemento típico de una página web (párrafos, imágenes, divisiones, listas, tablas, etc.), además de las etiquetas que crean los controles.

Las etiquetas que crean los controles en los formularios son *<input>, <button>, <select>, <optgroup>, <option> y <textarea>*. Además, se pueden estructurar los controles con las etiquetas *<fieldset>* y *<legend>*. Por último, la etiqueta *<label>* permite mejorar la accesibilidad de los controles.

El navegador envía únicamente los datos de los controles contenidos en el formulario. En una misma página puede haber varios formularios que envíen datos al mismo o a diferentes agentes

### **El atributo type**

El atributo *type* de la etiqueta *<input>* indica el tipo de control de que se trata.

En HTML 4.01 los tipos de *<input>* eran: botón para enviar *[submit](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html" \l "control-input-button-submit)*, botón para reiniciar *[reset](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html" \l "control-input-button-reset)*, texto *[text](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html" \l "control-input-text)*, contraseña *[password](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html" \l "control-input-password)*, archivo [*file*](https://www.mclibre.org/consultar/htmlcss/html/formularios-2.html#control-input-file), imagen *[image](https://www.mclibre.org/consultar/htmlcss/html/formularios-2.html" \l "control-input-image)* y oculto *[hidden](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html" \l "control-input-hidden)*.

En HTML 5 se añadieron los tipos de <input>: botón button, número [number](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html" \l "control-input-number), búsqueda [search](https://www.mclibre.org/consultar/htmlcss/html/html-formularios-2.html" \l "control-input-search), teléfono [tel](https://www.mclibre.org/consultar/htmlcss/html/html-formularios-2.html" \l "control-input-tel), dirección web [url](https://www.mclibre.org/consultar/htmlcss/html/html-formularios-2.html" \l "control-input-url), dirección de correo email, fecha date, hora [time](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-input-time), color [color](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html" \l "control-input-color) y rango [range](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html" \l "control-input-range).

En HTML 5.1 se añadieron los tipos de <input>: mes [month](https://www.mclibre.org/consultar/htmlcss/html/html-formularios-2.html" \l "control-input-date), semana week y fecha y hora local datetime-local.

### **El atributo name**

Identifica al control. Todos los controles de un formulario deben tener el atributo name porque, en general, sólo se envían los controles que lo tienen (aunque hay excepciones, como los controles de tipo imagen o los botones, que no los necesitan).

En general, los atributos name no se deben repetir, porque no se recibiría toda la información al enviarla. La única excepción es el botón radio, como se explica en el apartado correspondiente.

### **El atributo value**

Permite establecer el valor inicial de un control, aunque cada control lo utiliza de una forma ligeramente distinta y algunos controles no lo admiten (área de texto, archivo o imagen).

En algunos casos (botones) el valor se muestra al usuario y el usuario no puede modificarlo. En otros casos (cajas de texto, contraseña) el valor se muestra en la página y el usuario puede modificarlo. En otros casos (casillas de verificación, botones radio, opciones de menú, oculto) el valor no se muestra en la página y el usuario no puede modificarlo.

En algunos casos (botones, oculto, botones radio) el atributo value es necesario. En otros casos (casilla de verificación, opciones de menú) es conveniente. En otros casos (cajas de texto, contraseña, imagen) puede omitirse. En el caso del selector de archivo, los navegadores no lo admiten por motivos de seguridad.

### **El atributo required**

Permite indicar qué controles es obligatorio rellenar para enviar el formulario.

Si alguna de las opciones de un botón radio tiene establecido el atributo required, el control es obligatorio.

### **El atributo placeholder**

Permite mostrar en los controles de texto un texto que desaparece al escribir en el control.

### **El atributo size**

Permite establecer la longitud de los controles de texto (*text, password, search*, etc.). Por omisión, las cajas suelen tener 20 caracteres de longitud.

### **Los atributos maxlength y minlength**

Permiten establecer la longitud máxima y mínima, respectivamente, que puede escribir el usuario en un control de texto (*text, password, search*, etc.).

### **El atributo autofocus**

Permite indicar el control que debe tener el foco al cargarse el formulario.

Este atributo facilita la accesibilidad del formulario pues permite que el usuario pueda empezar a rellenar el formulario sin necesidad de hacer clic en el elemento. Como el navegador desplaza automáticamente la página hasta la posición del elemento, se recomienda que el elemento con atributo autofocus se encuentren al principio de la página, para que el desplazamiento no confunda al usuario.

1. ¿Qué tipo de controles existen en HTML? enumere todos los controles y ejemplifique para que caso se debe usar cada uno de ellos

Los controles son los elementos que permiten al usuario introducir información.

La mayoría de controles se obtienen mediante la etiqueta vacía <input>.

En HTML 4.01, los controles disponibles eran: botón para enviar [*<input type="submit">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-input-button-submit), botón para reiniciar [*<input type="reset">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-input-button-reset), botón [*<button>*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-botones), texto [*<input type="text">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-input-text), contraseña [*<input type="password">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-input-password), archivo [*<input type="file">*](https://www.mclibre.org/consultar/htmlcss/html/formularios-2.html#control-input-file), imagen [*<input type="image">*](https://www.mclibre.org/consultar/htmlcss/html/formularios-2.html#control-input-image), oculto [*<input type="hidden">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-input-hidden), área de texto [*<textarea>*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-textarea) y menú [*<select>*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-select).

En HTML 5 se añadieron los controles: botón general [*<input type="button">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-input-button-button), número [*<input type="number">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-input-number), búsqueda [*<input type="search">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios-2.html#control-input-search), teléfono [*<input type="tel">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios-2.html#control-input-tel), dirección web [*<input type="url">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios-2.html#control-input-url), dirección de correo [*<input type="email">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios-2.html#control-input-email), fecha [*<input type="date">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-input-date), hora [*<input type="time">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios-2.html#control-input-time), color [*<input type="color">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-input-color), rango [<input type="range">](https://www.mclibre.org/consultar/htmlcss/html/html-formularios.html#control-input-range), opciones *<option>* (para *<datalist>*), lista de valores *<datalist>*, salida *<output>*, barra de progreso *<progress>* y medidor <meter>.

En HTML 5.1 se añadieron los controles: mes [*<input type="month">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios-2.html#control-input-month), semana [*<input type="week">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios-2.html#control-input-week) y fecha y hora local [*<input type="datetime-local">*](https://www.mclibre.org/consultar/htmlcss/html/html-formularios-2.html#control-input-datetime-local).

1. Para enviar ciertos tipos de información al servidor, el protocolo HTTP provee diferentes **métodos de petición**.Los dos más importantes son GET y POST, los cuales, aunque entregan los mismos resultados, revelan algunas diferencias entre ellos.

## **Métodos de petición HTTP: GET**

Con el método GET, los datos que se envían al servidor **se escriben en la misma dirección URL**. En la ventana del navegador, se encuentra así:

www.ejemplo.com/registrarse.php?nombre=pedro&amp;apellido=perez&amp;edad=55&amp;genero=hombre

Toda la información introducida por el usuario (los llamados “parámetros URL”) se transmiten tan abiertamente como el URL en sí mismo. Esto tiene ventajas y desventajas.

### Ventajas de GET

Los parámetros URL se pueden **guardar junto a la dirección URL como marcador**. De esta manera, puedes introducir una búsqueda y más tarde consultarla de nuevo fácilmente. También se puede volver a acceder a la página a través del historial del navegador.

Esto resulta práctico, por ejemplo, si visitas con asiduidad un mismo lugar en Google Maps o si guardas páginas web con configuraciones de filtro determinadas.

### Desventajas de GET

La mayor desventaja del método GET es su **débil protección de los datos**. Los parámetros URL que se envían quedan visibles en la barra de direcciones del navegador y son accesibles sin clave en el historial de navegación, en el caché y en el log de los servidores.

Otra desventaja es que su **capacidad es limitada**: dependiendo del servidor y del navegador, no es posible introducir más de 2000 caracteres. Además, los parámetros URL solo pueden contener caracteres ASCII (letras, números, signos, etc.) y no datos binarios como archivos de audio o imágenes.

## **Métodos de petición HTTP: POST**

El método POST introduce los parámetros **en la solicitud HTTP**para el servidor. Por ello, no quedan visibles para el usuario. Además, 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 usuario 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 pueden enviar textos cortos, sino también otros tipos de información, como fotos o vídeos.

### Desventajas de POST

Cuando una página web que contiene un formulario se actualiza (por ejemplo, cuando se retrocede a la página anterior) **los datos del formulario se deben transferir de nuevo**(puede que alguna vez hayas recibido una de estas advertencias). Por este motivo, existe el riesgo de que los datos se envíen varias veces por error, lo que, en el caso de una tienda online, puede dar lugar a pedidos duplicados. No obstante, las webs modernas de las tiendas suelen estar preparadas para evitar este tipo de problemas.

Además, los datos transferidos con el método POST no pueden **guardarse junto al URL como marcador**.

## ¿Cuándo usar uno u otro?

El método **POST**es aconsejable cuando el usuario debe enviar datos o archivos al servidor, como, por ejemplo, cuando se rellenan formularios o se suben fotos.

El método **GET**es adecuado para la 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.

A modo de resumen:

* **GET** para la configuración de páginas web (filtros, ordenación, búsquedas, etc.)
* **POST**para la transferencia de información y datos

1. Crea un formulario utilizando la librería de bootstrap utilizando un proyecto con .NET framework C# MVC teniendo en cuenta lo siguiente:
2. El formulario debe ser para el registro de alumno
3. Contiene los siguientes campos Email, clavé de acceso, nombre, apellido, DNI, fecha de nacimiento, Carrera (Informática, Psicología, Medicina), Sexo y dirección.
4. El formulario se envía por post y se guarda en la base de datos.