Desarrollo Basado en Plataformas

Dr. Edward Hinojosa Cárdenas
ehinojosa@unsa.edu.pe

Formularios Web

- La Web 2.0 está completamente enfocada en el usuario.

- Cuando el usuario es el centro de atención, todo está relacionado con interfaces, en cómo hacerlas más intuitivas, más naturales, más prácticas, y por supuesto más atractivas.

- Los formularios web son la interface más importante de todas, permiten a los usuarios insertar datos, tomar decisiones, comunicar información y cambiar el comportamiento de una aplicación.

Formularios Web

- Durante los últimos años, códigos personalizados y librerías fueron creados para procesar formularios en el ordenador del usuario.

- HTML5 vuelve a estas funciones estándar agregando nuevos atributos, elementos y una API (interfaz de programación de aplicaciones - Application Programming Interface) completa. Ahora la capacidad de procesamiento de información insertada en formularios en tiempo real ha sido incorporada en los navegadores y completamente estandarizada.

El elemento <form>

- Los formularios en HTML no han cambiado mucho.

- La estructura sigue siendo la misma, pero HTML5 ha agregado nuevos elementos, tipos de campo y atributos para expandirlos tanto como sea necesario y proveer así las funciones actualmente implementadas en aplicaciones web.

El elemento <input>

- El elemento más importante en un formulario es <input>. Este elemento puede cambiar sus características gracias al atributo type (tipo).

- Este atributo determina qué clase de entrada es esperada desde el usuario. Los tipos disponibles hasta el momento eran el multipropósitos text (para textos en general) y solo unos pocos más específicos como password o submit.

- HTML5 ha expandido las opciones incrementando de este modo las posibilidades para este elemento.

El elemento <input>

- En HTML5 estos nuevos tipos no solo están especificando qué clase de entrada es esperada sino también diciéndole al navegador qué debe hacer con la información recibida.

- El navegador procesará los datos ingresados de acuerdo al valor del atributo type y validará la entrada o no.

- El atributo type trabaja junto con otros atributos adicionales para ayudar al navegador a limitar y controlar en tiempo real lo ingresado por el usuario.

El elemento <input>

```
<!DOCTYPE html>
<html lang="es">
<head>
    <title>Formularios</title>
</head>
<body>
    <section>
        <form name="miformulario" id="miformulario" method="get">
            <label for="nombre">Nombre</label><br>
            <input type="text" name="nombre" id="nombre"><br>
            <input type="submit" value="Enviar">
        </form>
    </section>
</body>
</html>
```

El elemento <input>

Tipo email

- Casi todo formulario en la web ofrece un campo para ingresar una dirección de email, pero hasta ahora el único tipo de campo disponible para esta clase de datos era text.

- El tipo text representa un texto general, no un dato específico, por lo que teníamos que controlar la entrada con código Javascript para estar seguros de que el texto ingresado correspondía a un email válido. Ahora el navegador se hace cargo de esto con el nuevo tipo email:

Tipo email

- El texto insertado en el campo generado por el siguiente código será controlado por el navegador y validado como un email. Si la validación falla, el formulario no será enviado.

- Cómo cada navegador responderá a una entrada inválida no está determinado en la especificación de HTML5. Por ejemplo, algunos navegadores mostrarán un borde rojo alrededor del elemento <input> que produjo el error y otros lo mostrarán en azul.

```
<input type="email" name="miemail" id="miemail">
```

Tipo url

- Este tipo de campo trabaja exactamente igual que el tipo email pero es específico para direcciones web. Está destinado a recibir solo URLs absolutas y retornará un error si el valor es inválido.

```
<input type="url" name="miurl" id="miurl">
```

Tipo number

- Como su nombre lo indica, el tipo number es sólo válido cuando recibe una entrada numérica. Existen algunos atributos nuevos que pueden ser útiles para este campo:
  - min El valor de este atributo determina el mínimo valor aceptado para el campo.
  - max El valor de este atributo determina el máximo valor aceptado para el campo.
  - step El valor de este atributo determina el tamaño en el que el valor será incrementado o disminuido en cada paso. Por ejemplo, si declara un valor de 5 para step en un campo que tiene un valor mínimo de 0 y máximo de 10, el navegador no le permitirá especificar valores entre 0 y 5 o entre 5 y 10.

Tipo number

- No es necesario especificar ambos atributos (min y max), y el valor por defecto para step es 1.

```
<input type="number" name="numero" id="numero" min="0" max="10"
                                                    step="5">
```

Tipo range

- Este tipo de campo hace que el navegador construya una nueva clase de control que no existía previamente.

- Este nuevo control le permite al usuario seleccionar un valor a partir de una serie de valores o rango. Normalmente es mostrado en pantalla como una puntero deslizable o un campo con flechas para seleccionar un valor entre los predeterminados, pero no existe un diseño estándar hasta el momento.

Tipo range

- El tipo range usa los atributos min y max estudiados previamente para configurar los límites del rango.

- También puede utilizar el atributo step para establecer el tamaño en el cual el valor del campo será incrementado o disminuido en cada paso.

```
<input type="range" name="numero" id="numero" min="0" max="10"
                                                    step="5">
```

Tipo date

- Este es otro tipo de campo que genera una nueva clase de control. En este caso fue incluido para ofrecer una mejor forma de ingresar una fecha.

- Algunos navegadores muestran en pantalla un calendario que aparece cada vez que el usuario hace clic sobre el campo. El calendario le permite al usuario seleccionar un día que será ingresado en el campo junto con el resto de la fecha.

Tipo date

- Un ejemplo de uso es cuando necesitamos proporcionar un método para seleccionar una fecha para un vuelo o la entrada a un espectáculo.

- Gracias al tipo date ahora es el navegador el que se encarga de construir un almanaque o las herramientas necesarias para facilitar el ingreso de este tipo de datos.

Tipo date

- La interface no fue declarada en la especificación. Cada navegador provee su propia interface y a veces adaptan el diseño al dispositivo en el cual la aplicación está siendo ejecutada. Normalmente el valor generado y esperado tiene la sintaxis año-mes-día.

```
<input type="date" name="fecha" id="fecha">
```

Tipo week

- Este tipo de campo ofrece una interface similar a date, pero solo para seleccionar una semana completa. Normalmente el valor esperado tiene la sintaxis 2011-W50 donde 2011 es al año y 50 es el número de la semana.

```
<input type="week" name="semana" id="semana">
```

Tipo month

- Similar al tipo de campo previo, éste es específico para seleccionar meses. Normalmente el valor esperado tiene la sintaxis año-mes.

```
<input type="month" name="mes" id="mes">
```

Tipo time

- El tipo de campo time es similar a date, pero solo para la hora. Toma el formato de horas y minutos, pero su comportamiento depende de cada navegador en este momento.

- Normalmente el valor esperado tiene la sintaxis hora:minutos:segundos, pero también puede ser solo hora:minutos.

```
<input type="time" name="hora" id="hora">
```

Tipo datetime-local

- El tipo de campo datetime-local es como el tipo datetime sin la zona horaria.

```
<input type="datetime-local" name="tiempolocal" id="tiempolocal">
```

Tipo color

- Además de los tipos de campo para fecha y hora existe otro tipo que provee una interface predefinida similar para seleccionar colores.

- Normalmente el valor esperado para este campo es un número hexadecimal, como #00FF00.

```
<input type="color" name="micolor" id="micolor">
```

Atributos útiles

- Algunos tipos de campo requieren de la ayuda de atributos, como los anteriormente estudiados min, max y step

- Otros tipos de campo requieren la asistencia de atributos para mejorar su rendimiento o determinar su importancia en el proceso de validación.

Atributo placeholder

- Especialmente en tipos de campo search, pero también en entradas de texto normales, el atributo placeholder representa una sugerencia corta, una palabra o frase provista para ayudar al usuario a ingresar la información correcta.

- El valor de este atributo es presentado en pantalla por los navegadores dentro del campo, como una marca de agua que desaparece cuando el elemento es enfocado.

```
<input type="search" name="busqueda" id="busqueda"
placeholder="escriba su búsqueda">
```

Atributo required

- Este atributo booleano no dejará que el formulario sea enviado si el campo se encuentra vacío.

- Por ejemplo, cuando usamos el tipo email para recibir una dirección de email, el navegador comprueba si la entrada es un email válido o no, pero validará la entrada si el campo está vacío.

Atributo required

- Cuando el atributo required es incluido, la entrada será válida sólo si se cumplen las dos condiciones: que el campo no esté vacío y que el valor ingresado esté de acuerdo con los requisitos del tipo de campo.

```
<input type="email" name="miemail" id="miemail" required>
```

Atributo multiple

- El atributo multiple es otro atributo booleano que puede ser usado en algunos tipos de campo (por ejemplo, email) para permitir el ingreso de entradas múltiples en el mismo campo.

- Los valores insertados deben estar separados por coma para ser válidos. El siguiente código permite la inserción de múltiples valores separados por coma, y cada uno de ellos será validado por el navegador como una dirección de email.

```
<input type="email" name="miemail" id="miemail" multiple>
```

Atributo autofocus

- El atributo autofocus enfocará la página web sobre el elemento seleccionado.

```
<input type="search" name="busqueda" id="busqueda" autofocus>
```

Atributo pattern

- El atributo pattern es para propósitos de validación. Usa expresiones regulares para personalizar reglas de validación. Algunos de los tipos de campo ya estudiados validan cadenas de texto específicas, pero no permiten hacer validaciones personalizadas, como por ejemplo un código postal que consiste en 5 números. No existe ningún tipo de campo predeterminado para esta clase de entrada.

- El atributo pattern nos permite crear nuestro propio tipo de campo para controlar esta clase de valores no ordinarios. Puede incluso incluir un atributo title para personalizar mensajes de error.

Atributo pattern

- Expresiones regulares son un tema complejo y no relacionado directamente con HTML5. Tarea: estudiar sobre expresiones regulares.

```
<input pattern="[0-9]{5}" name="codigopostal" id="codigopostal"
                title="inserte los 5 números de su código postal">
```

Elemento <datalist>

- El elemento <datalist> es un elemento específico de formularios usado para construir una lista de ítems que luego, con la ayuda del atributo list, será usada como sugerencia en un campo del formulario.

- Este elemento utiliza el elemento <option> en su interior para crear la lista de datos a sugerir. Con la lista ya declarada, lo único que resta es referenciarla desde un elemento <input> usando el atributo list.

Elemento <datalist>

```
<datalist id="informacion">
  <option value="123123123" label="Teléfono 1">
  <option value="456456456" label="Teléfono 2">
</datalist>
```

```
<input type="tel" name="telefono" id="telefono" list="informacion">
```

Laboratorio 6 – Grupo C

- Cree el siguiente formulario.
- Se recomienda usar
 como salto de línea o tablas.
- Tipo de DNI debe tener dos opciones: DNI Amarillo y DNI Azul.
- Considere los placeholders mostrados.
- Coloqué el foco en el text Nombre.
- Antes de presentarlos complete el formulario.

Mi Primer Formulario: NombreAlumno

Nombre:

Escriba su nombre completo

email:

Página web:

Utilice http:\\

Tipo de DNI:

DNI:

Nivel de Inglés:

Fecha de Nacimiento:

dd/mm/aaaa

Semana Actual:

Semana --, ----

Mes Actual:

--------- de ----

Hora Actual:

--:-- -----

Fecha y Hora Actual:

dd/mm/aaaa --:-- -----

Color Favorito:

Enviar

Gracias

Dr. Edward Hinojosa Cárdenas
ehinojosa@unsa.edu.pe