



**M E A N**

**WEB FULL STACK DEVELOPER**

*Germán Caballero Rodríguez*  
*germanux@gmail.com*



# Formularios en HTML5

# HTML



# INDICE

- 1) Introducción
- 2) Elemento `<form>`
- 3) Elemento `<input>` y nuevos tipos
- 4) Nuevos atributos en `<input>`
- 5) Nuevos elementos para `<form>`

# Introducción

- La Web 2.0 está completamente enfocada en el usuario.
- Y 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.

# Introducción

- 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 completa.
- Ahora la capacidad de procesamiento de información insertada en formularios en tiempo real ha sido incorporada en los navegadores y completamente estandarizada.

## 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.



# Elemento <form>

- Nuevos atributos para el elemento <form>:

## **Autocomplete**

- Este es un viejo atributo que se ha vuelto estándar en esta especificación.
- Puede tomar dos valores: on y off.
- El valor por defecto es on.

# Elemento <form>

- Nuevos atributos para el elemento <form>:

## **Autocomplete**

- Cuando es configurado como off los elementos <input> pertenecientes a ese formulario tendrán la función de autocompletar desactivada, sin mostrar entradas previas como posibles valores.
- Puede ser implementado en el elemento <form> o en cualquier elemento <input> independientemente.



## Elemento <form>

- Nuevos atributos para el elemento <form>:  
**novalidate**
- Una de las características de formularios en HTML5 es la capacidad propia de validación.
- Los formularios son automáticamente validados.
- Para evitar este comportamiento, podemos usar el atributo novalidate.

## Elemento <form>

- Nuevos atributos para el elemento <form>:  
**novalidate**
- Para lograr lo mismo para elementos <input> específicos, existe otro atributo llamado **formnovalidate**.
- Ambos atributos son booleanos, ningún valor tiene que ser especificado (su presencia es suficiente para activar su función).

# 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**.

# Elemento <form> e <input>

- Práctica formulario :

```
<!DOCTYPE html>
<html lang="es">

<head>
  <title>Formularios</title>
</head>

<body>
  <section>
    <form name="miform" id="miform" method="get">
      <input type="text" name="nombre" id="nombre">
      <input type="submit" value="Enviar">
    </form>
  </section>
</body>

</html>
```

## Elemento <input>

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

# Elemento <input>

- 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.

# Elemento `<input>`

- Lista de tipos de elementos inputs:

<code>&lt;/&gt;</code> <u>text</u>	<code>&lt;/&gt;</code> <u>password</u>	<code>&lt;/&gt;</code> date
<code>&lt;/&gt;</code> <u>search</u>	<code>&lt;/&gt;</code> <u>number</u>	<code>&lt;/&gt;</code> <u>month</u>
<code>&lt;/&gt;</code> <u>url</u>	<code>&lt;/&gt;</code> <u>range</u>	<code>&lt;/&gt;</code> <u>week</u>
<code>&lt;/&gt;</code> tel	<code>&lt;/&gt;</code> <u>checkbox</u>	<code>&lt;/&gt;</code> time
<code>&lt;/&gt;</code> <u>email</u>	<code>&lt;/&gt;</code> file	<code>&lt;/&gt;</code> <u>datetimelocal</u>
<code>&lt;/&gt;</code> color	<code>&lt;/&gt;</code> <u>datetime</u>	



# Elemento <input>

## Tipo email

- Ofrece un campo para ingresar una dirección de email

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

# Elemento <input>

## Tipo search

- El tipo search (búsqueda) no controla la entrada, es solo una indicación para los navegadores.
- Al detectar este tipo de campo algunos navegadores cambiarán el diseño del elemento para ofrecer al usuario un indicio de su propósito.

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

# Elemento <input>

## 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">
```

# Elemento <input>

## Tipo tel

- Este tipo de campo es para números telefónicos.
- A diferencia de los tipos email y url, el tipo tel no requiere ninguna sintaxis en particular.
- Es solo una indicación para el navegador en caso de que necesite hacer ajustes de acuerdo al dispositivo en el que la aplicación es ejecutada.

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

# Elemento <input>

## Tipo number

- El tipo number es sólo válido cuando recibe una entrada numérica.
- Atributos nuevos 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.

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

# Elemento <input>

## Tipo range

- 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.
- El tipo range usa los atributos **min** y **max** estudiados previamente para configurar los límites del rango.
- También puede utilizar el atributo step

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

# Elemento <input>

## Tipo date

- Ofrece una mejor forma de ingresar fechas.
- 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.

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



# Elemento <input>

## 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">
```

# Elemento <input>

## 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">
```

# Elemento <input>

## 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">
```

# Elemento <input>

## Tipo datetime

- El tipo de campo datetime es para ingresar fecha y hora completa, incluyendo la zona horaria.

```
<input type="datetime" name="fechahora" id="fechahora">
```

# Elemento <input>

## 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">
```

# Elemento <input>

## Tipo color

- Provee una interface predefinida para seleccionar colores.
- Normalmente el valor esperado para este campo es un número hexadecimal, como #00FF00.

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

# Elemento <input>

## Práctica: Crea un HTML con el formulario

```
<form id="miFormulario" action="#">
  <div id="rosa">
    <p>Nombre: <input type="text" name="nombre"/></p>
    <p>E-mail: <input type="email" name="email"></p>
    <p>Color: <input type="color" name="color"></p>
    <p>URL: <input type="url" name="url">
    <p>Edad: <input type="number" name="edad" min="0" max="100" value="0"></p>
    <p>Rango: <input type="range" name="rango" min="-50" max="50"></p>
    <p>Fecha: <input type="date" name="date_control"/>
    <p>hora: <input type="time" name="date_control2"/>
    <p>Fecha y hora: <input type="datetime-local" name="date_control3"/>
    <p>Mes: <input type="month" name="date_control"/>
    <p>Semana: <input type="week" name="date_control"/>
    <p>Telefono: <input type="tel" name="tlf"/>
    <p>Buscar: <input type="search" name="busca_control"/>
  </div>
  <p><input type="submit" value="enviar"></p>
</form>
```



## Atributos <input>

- 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.

## Atributos <input>

- Ya vimos algunos de ellos, como **novalidate** para evitar que el formulario completo sea validado o **formnovalidate** para hacer lo mismo con elementos individuales.
- El atributo **autocomplete**, también estudiado anteriormente, provee medidas de seguridad adicionales para el formulario completo o elementos individuales.
- Estos atributos no son los únicos incorporados por HTML5.

## Atributos <input>

- Ya vimos algunos de ellos, como **novalidate** para evitar que el formulario completo sea validado o **formnovalidate** para hacer lo mismo con elementos individuales.
- El atributo **autocomplete**, también estudiado anteriormente, provee medidas de seguridad adicionales para el formulario completo o elementos individuales.
- Estos atributos no son los únicos incorporados por HTML5.

# Atributos <input>

## 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.
- 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.

# Atributos <input>

## 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.

# Atributos <input>

## Atributo multiple

- Es otro atributo booleano que puede ser usado en algunos tipos de campo (por ejemplo, email o file) para permitir el ingreso de entradas múltiples en el mismo campo.
- Los valores insertados deben estar separados por coma para ser válidos.

# Atributos <input>

## Atributo autofocus

- Esta es una función que muchos desarrolladores aplicaban anteriormente utilizando el método focus() de Javascript.
- Este método era efectivo pero forzaba el foco sobre el elemento seleccionado, incluso cuando el usuario ya se encontraba posicionado en otro diferente.
- Este comportamiento era irritante pero difícil de evitar hasta ahora.
- El atributo autofocus enfocará la página web sobre el elemento seleccionado pero considerando la situación actual.
- No moverá el foco cuando ya haya sido establecido por el usuario sobre otro elemento.



# Atributos <input>

## Atributo pattern

- El atributo pattern es para propósitos de validación.
- Usa expresiones regulares para personalizar reglas de validación.
- El atributo pattern nos permite crear nuestro propio tipo de campo para controlar esta clase de valores no ordinarios.

# Atributos <input>

## Atributo pattern

- Puede incluso incluir un atributo title para personalizar mensajes de error.
- Por ejemplo, para hacer campo de código postal que consiste en 5 números.

```
<input name="codigopostal" id="codigopostal"  
       pattern="[0-9]{5}"  
       title="inserte los 5 números del C.P.">
```

# Atributos <input>

## Atributo form

- El atributo form es una adición útil que nos permite declarar elementos para un formulario fuera del ámbito de las etiquetas <form>.
- Hasta ahora, para construir un formulario teníamos que escribir las etiquetas <form> de apertura y cierre y luego declarar cada elemento del formulario entre ellas.
- En HTML5 podemos insertar los elementos en cualquier parte del código y luego hacer referencia al formulario que pertenecen usando su nombre y el atributo form:

# Atributos <input>

## Atributo form

```
<!DOCTYPE html>
<html lang="es">
<head>
  <title>Formularios</title>
</head>
<body>
  <nav>
    <input type="search" name="busqueda"
           id="busqueda" form="formulario">
  </nav>
  <section>
    <form name="formulario" id="formulario" method="get">
      <input type="text" name="nombre" id="nombre">
      <input type="submit" value="Enviar">
    </form>
  </section>
</body>
</html>
```

## Atributos <input>

**Ejercicio:** Formulario anterior con los siguientes inputs y sus atributos:

- Nombre: Con una expresión regular para su validación, a elegir
- Email: Que muestre por defecto un email de ejemplo
- Con varios campos obligatorios

# Nuevos elementos para <form>

## 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.

# Nuevos elementos para <form>

## Elemento <datalist>

- Este elemento utiliza el elemento <option> en su interior para crear la lista de datos a sugerir.

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

- Con la lista ya declarada, lo único que resta es referenciarla desde un elemento <input> usando el atributo list:

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

# Nuevos elementos para <form>

## Elemento <progress>

- Este elemento no es específico de formularios, pero debido a que representa el progreso en la realización de una tarea, y usualmente estas tareas son comenzadas y procesadas a través de formularios, puede ser incluido dentro del grupo de elementos para formularios.
- El elemento <progress> utiliza dos atributos para configurar su estado y límites.
  - El atributo **value** indica qué parte de la tarea ya ha sido procesada,
  - y **max** declara el valor a alcanzar para que la tarea se considere finalizada.



# Nuevos elementos para <form>

## Elemento <meter>

- Similar a <progress>, el elemento <meter> es usado para mostrar una escala, pero no de progreso.
- Este elemento tiene la intención de representar una medida, como el tráfico del sitio web, por ejemplo.
- El elemento <meter> cuenta con varios atributos asociados:
  - **min y max** configuran los límites de la escala,
  - **value** determina el valor medido,
  - y **low, high y optimum** son usados para segmentar la escala en secciones diferenciadas y marcar la posición que es óptima.

# Nuevos elementos para <form>

## Elemento <output>

- Este elemento representa el resultado de un cálculo.
- Normalmente ayudará a mostrar los resultados del procesamiento de valores provistos por un formulario.
- El atributo for asocia el elemento <output> con el elemento fuente que participa del cálculo, pero este elemento deberá ser referenciado y modificado desde código Javascript.
- Su sintaxis es <output>valor</output>.

# Nuevos elementos para <form>

**Ejercicio:** Formulario con una caja de texto que dé a elegir entre varios cafés (<datalist>)

Lista:

Fecha:

hora:

Fecha:

Café solo

Café con leche

Café descafeinado

Carajillo

Latte Macchiato

Chocolate