

Introducción a Quality Assurance

Formularios HTML

¿Qué es un formulario web?

Un formulario web, son esa serie de campos de texto, fechas, números, cajitas para tiquear y con un botón al final que tenemos que rellenar para registrarnos en un página, para reservar un fin de semana en cancan o para aprobar un examen.

Un formulario sin mucho diseño, normalmente lo vamos a ver así:

Nombre	Apellido
<input type="text"/>	<input type="text"/>
Fecha de nacimiento	Documento de identidad
<input type="text" value="dd/mm/aaaa"/>	<input type="text"/>
Dirección	País
<input type="text" value="Cabildo y Juramento"/>	<input type="text" value="Elija..."/>
Estado civil	
<input checked="" type="radio"/> Soltero/a	
<input type="radio"/> Casado/a	
<input type="button" value="Regístrate"/>	

¿Por qué vamos a analizar los formularios HTML?

El testing de formularios es un proceso que se realiza para probar la **calidad de un formulario en un página web**, verificando elementos como campos de texto, longitud y diseño en general. Uno de los propósitos de testear formularios es para

mejorar las **tasas de conversión**, lo que sería el porcentaje de gente que pasa de ser **visitantes de la página a consumidores/clientes** de dicha página.

Los formularios son herramientas importantes para muchos sitios web ya que sirven como **dispositivos de comunicación con los clientes**, creando una conexión entre los visitantes y las empresas.

Que los clientes puedan enviar la información que quieren de manera correcta a través del formulario es importante como, por ejemplo en el caso de un pedido o una consulta de soporte. Ahí tenemos información que se debe **representar de la manera que quiere el cliente** y por eso es importante que probemos el correcto funcionamiento del formulario.

¿Por qué es importante la prueba de formularios?

Ya vimos que es la prueba de formulario y algunas de las razones por las cuales deberíamos siempre hacer pruebas de formulario, ahora vamos a enumerar todas las **ventajas** que nos va a dar la prueba de formularios.

- **Impulsa el tráfico de usuarios:** Puede ayudar a impulsar el tráfico a su sitio web, mejorando tanto las variantes orgánicas (visitantes) como las pagas (clientes). Cuando sus formularios funcionan correctamente, es más probable que los clientes regresen a su sitio web y dejen comentarios.
- **Mejora las comunicaciones empresa-cliente:** Un formulario que funciona según lo previsto permite a los clientes comunicarse con la empresa para proporcionar la información necesaria para las operaciones comerciales.
- **Percepción del cliente:** Tener un formulario que funciona como corresponde puede darle mucha información a la empresa sobre cómo el cliente percibe el sitio web, que cosas le gustan, cuáles no, qué cosas están fallando, siempre desde la perspectiva de alguien que utiliza el sitio web.
- **Mejora la resolución de problemas:** El testeo de formularios es útil porque nos va a ayudar a descubrir todos los problemas que tenga nuestro formulario y cómo resolverlos.
- **Mejora las conversiones de clientes:** Testear nuestro formulario es importante porque puede ayudar a convertir a los visitantes en clientes.

Además, los formularios que funcionan correctamente pueden mejorar la percepción que tiene el cliente sobre nuestro sitio web.

- **Mejora la usabilidad del formulario:** La usabilidad es esencial para un formulario porque describe qué tan bien los clientes pueden usar y navegar a través de los distintos aspectos y detalles del formulario. Probar los formularios nos va a dejar un formulario sencillo de utilizar para el cliente.

¿Qué elementos componen a un formulario?

Los formularios están compuestos principalmente por campos de texto o en html (lenguaje de programación con el que se crean los formularios) conocidos como **inputs**, estos campos son los que deberemos completar con nuestra información.

Hay muchas convenciones que damos por sentado en los formularios, por ejemplo si me piden ingresar numeros, que solo pueda ingresar numeros o que si voy a ingresar mi contraseña no se pueda ver el texto y aparezcan círculos negros.

Pero esto hay que validarlo y para eso tenemos que entender cómo funcionan estos campos y qué tipos de campos existen. En esta guía veremos los distintos tipos de campos que existen y cuáles son algunas de las validaciones importantes que tenemos que hacer.

Inputs

HTML, que es el lenguaje de marcado de hipertexto usado para crear formularios, nos propone una gran diversidad de alternativas a la hora de crear nuestros formularios, es decir, una gran variedad de elementos para diferentes propósitos. Estas van desde la clásica caja de texto, hasta la lista de opciones en un menú desplegable, pasando por las cajas de validación, etc. Por ahora nos **concentramos en los inputs** y más adelante veremos las cajas o menús desplegables

Los campos de texto son generados en el código por medio de la etiqueta **<input>** por eso les decimos inputs. Con solo la etiqueta input decimos que el

formulario va a tener un campo de texto, pero, no tiene definido ningún tipo de dato concreto.

Tipo de inputs

Para poder especificar el tipo de dato que se va a ingresar en nuestro input se le agrega un atributo llamado type y a ese atributo se le especifica el tipo de dato, por ejemplo: `<input type="number">`. Esto haría un input de tipo numérico. Veamos entonces todos los tipos de datos que nos permite el input.

1. TEXT (Texto)

Este tipo permite al usuario ingresar texto.

Se vería así:

Para textos muy grandes usamos otro tipo de input llamado textarea, que lo veremos más adelante.

Validaciones a realizar en los campos de texto

Al encontrarnos con un campo de texto, nosotros como testers tenemos la tarea de validar los siguientes puntos, para que el formulario funcione de la manera esperada.

- **Mayúsculas, minúsculas y números:** Que el campo le permita al usuario escribir en mayúsculas, minúsculas y números. Imagínate que tienes que ingresar la dirección de tu calle, deberías poder ingresar mayúsculas, minúsculas y números.
- **Caracteres especiales :** Los caracteres especiales son **caracteres que no están presentes en el diseño del teclado**. Se pueden usar si se presiona una combinación de teclas en el teclado (ej: coma, ©,®, #)
- **Caracteres de control:** Un carácter de control es, en el ámbito de la informática, un carácter no imprimible que sirve para uso interno de la computadora (ej: null, ETX)

- **Máximo posible:** Usualmente nosotros validamos una longitud de caracteres que elija el programador, supongamos que el programador elige que se pueden ingresar 100 caracteres, deberemos probar qué más de 100 no se pueda. El máximo posible de caracteres que se puede ingresar por defecto en un campo de texto es **524,288 caracteres**.
- **No trunca la cadena ingresada:** Cuando decimos que no trunque la cadena ingresada es que no acorte el largo de la cadena, por ejemplo: "Mi nombre es M..."

2. NUMBER (Numérico)

Este tipo permite al usuario ingresar números. Los navegadores vienen con validaciones para evitar que el usuario ingrese algo que no sea números. Además, en los navegadores modernos, los campos numéricos suelen venir con controles que permiten a los usuarios cambiar su valor de forma gráfica.

Se vería así:

A screenshot of a web browser's numeric input field. The field is a rectangular box containing the number '0'. To the right of the box is a small square button with two upward-pointing triangles, which is a standard UI element for incrementing the value in a numeric field.

Validaciones a realizar en los campos numéricos

Al encontrarnos con un campo numérico, nosotros como testers tenemos la tarea de validar los siguientes puntos, para que el formulario funcione de la manera esperada.

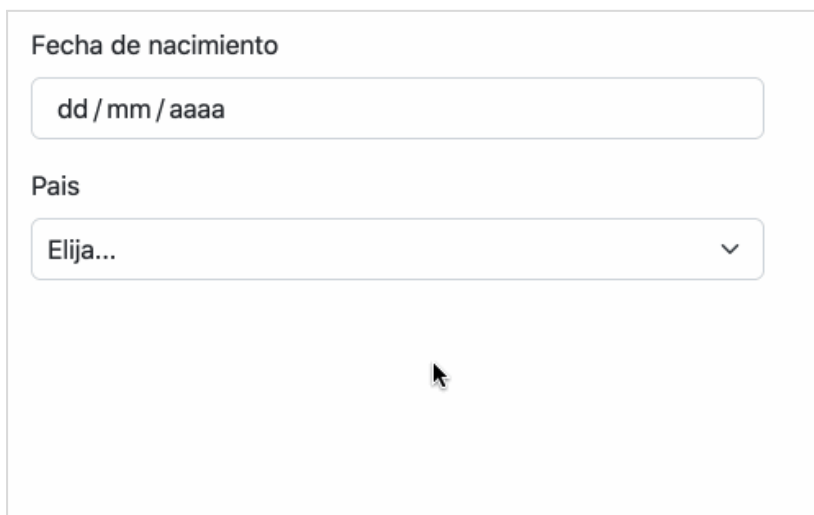
- **Sólo números (o caracteres como +, -, ., o el número e, entre otros):** Supongamos que tenemos que ingresar una suma o que tenemos que ingresar un dni con puntos entre cada número.
- **Máximo posible :**Usualmente nosotros validamos una longitud numérica que elija el programador, supongamos que el dato a ingresar es un año, el programador pondrá que no se puedan ingresar más de 4 caracteres y nosotros deberemos validar que el funcionamiento sea correcto. El máximo posible de caracteres numéricos que se puede ingresar por defecto es de 524,288 caracteres numéricos.

- **Números negativos, cero, decimales:** Que el campo le permita al usuario escribir números negativos, el número cero o escribir con decimales. Lo decimales en campos numéricos suelen escribirse con punto en vez de coma, ej: 10.5
- **No nulos:** Significa que no permite enviar el formulario con ese campo vacío, seguramente lo has visto en algún formulario de google que has completado en el pasado. Depende de como armemos el formulario y que consideremos importante, este tipo de restricciones pueden existir o no. Por ejemplo, muchas veces los datos personales son campos marcados como no nulos, ya que son fundamentales para poder procesar y gestionar los datos ingresados.

3. DATE (Fechas)

Este le permite al usuario ingresar una fecha, ya sea mediante una caja de texto o una interfaz gráfica con selector de fecha.

Se vería así:



Fecha de nacimiento

dd / mm / aaaa

Pais

Elija... ▼

El formulario muestra dos campos de entrada. El primero, etiquetado como 'Fecha de nacimiento', es un campo de texto con el placeholder 'dd / mm / aaaa'. El segundo, etiquetado como 'Pais', es un selector de lista desplegable con el texto 'Elija...' y un icono de flecha hacia abajo.

Validaciones a realizar en los campos de fechas

Al encontrarnos con un campo de fecha, nosotros como testers tenemos la tarea de validar los siguientes puntos, para que el formulario funcione de la manera esperada.

- **Sólo números, tipo de caracteres aceptados, combinado, formato predefinido:** Que el campo le permita al usuario solo escribir números o usar caracteres combinados como por ejemplo el guión para escribir una fecha (31-05-1997)
- **Formato: d/m/a . a/m/d . m/d/a . d-m-a . a-m-d . m-d-a:**
El formato en el que se representa la fecha depende del país en donde se utilice la web que aloja al formulario o la cultura a la que pertenecen los usuarios que lo van a completar. Supongamos que en el argentina usamos dia/mes/año, revisar que esté ese formato y no a/m/d
- **Caracteres especiales :** Los caracteres especiales son caracteres que no están presentes en el diseño del teclado. Se pueden usar si se presiona una combinación de teclas en el teclado (ej: coma, ©,®, #). Es necesario corroborar si el formulario acepta o no acepta este tipo de caracteres según los requerimientos planteados en el proyecto.
- **No nulos:** Significa que no permite enviar el formulario con ese campo vacío, seguramente lo has visto en algún formulario de google que has completado en el pasado.

4. Email

Este tipo permite al usuario ingresar un mail. Los navegadores vienen con validaciones para validar que se esté ingresando con el formato correcto de un mail. Este input se va a ver como un input de texto común y corriente.

Validaciones a realizar en los campos de email

Al encontrarnos con un campo de email, nosotros como testers tenemos la tarea de validar los siguientes puntos, para que el formulario funcione de la manera esperada.

- **Mayúsculas, minúsculas y números:** Que el campo le permita al usuario escribir en mayúsculas, minúsculas y números. Supongamos que tenemos un mail con todas esas combinaciones. Ej: T3St3r@egg.com

- **Caracteres especiales** : Qué podamos poner un mail con caracteres especiales, por ejemplo `**testers!!@egg.com`
- **Formato correcto de un mail**: Que el campo **solo** le permita al usuario escribir con el formato que corresponde de un mail, `texto@dominio.com`. Que no permita escribir algo así: `testers.com`

5. Texto oculto

Hay determinados casos en los que podemos desear esconder el texto escrito en el campo input, por medio de círculos negros, de manera que aporte una cierta confidencialidad. Para esto vamos a usar el type **password**.

Se vería así:

Email

Contraseña

Validación campo oculto

Con el campo oculto lo único que tenemos que validar es su correcto funcionamiento, si yo ingreso texto en dicho campo debería **mostrarlo oculto**.

6. TEXTAREA para texto largo

Si deseamos poner a disposición del usuario un campo de texto donde pueda escribir cómodamente sobre un espacio compuesto de varias líneas, hemos de invocar una nueva etiqueta: **<textarea>** y su cierre correspondiente.

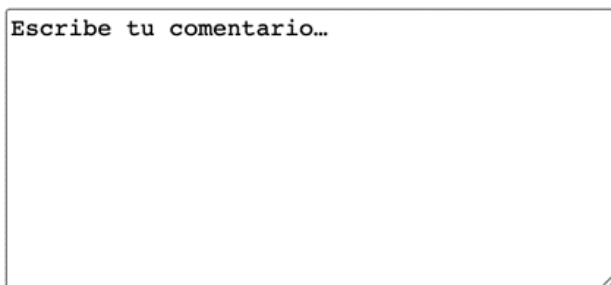
Este tipo de campos son prácticos cuando el contenido a enviar no es un nombre, teléfono, edad o cualquier otro dato breve, sino más bien, un

comentario, opinión, etc. En los que existe la posibilidad de que el usuario desee rellenar varias líneas.

El resultado es el siguiente:



Asimismo, es posible predefinir el contenido del campo.



Validaciones a realizar en los campos de texto largo

Al encontrarnos con un campo de texto, vamos a realizar las mismas validaciones que realizamos para los campos de texto normales.

- **Mayúsculas, minúsculas y números:** Que el campo le permita al usuario escribir en mayúsculas, minúsculas y números. Imagínate que tienes que ingresar la dirección de tu calle, deberías poder ingresar mayúsculas, minúsculas y números.
- **Caracteres especiales :** Los caracteres especiales son **caracteres que no están presentes en el diseño del teclado**. Se pueden usar si se presiona una combinación de teclas en el teclado (ej: coma, ©,®, #)
- **Caracteres de control:** Un carácter de control es, en el ámbito de la informática, un carácter no imprimible que sirve para uso interno de la computadora (ej: null, ETX)

- **Máximo posible:** Como ya vimos en el campo de texto, usualmente nosotros validamos una longitud numérica que elija el programador, supongamos que el dato a ingresar es un párrafo de 100 caracteres, el programador pondrá que no se puedan ingresar más de 100 caracteres y nosotros deberemos validar que el funcionamiento sea correcto.
- **No trunca la cadena ingresada:** Cuando decimos que no trunque la cadena ingresa es que no acorte el largo de la cadena de caracteres, por ejemplo: "Mi canción favorita es p..."

Configuraciones extras

Vamos a explicar algunas configuraciones que le podemos poner a nuestros campos de texto para que funcionen de manera distinta, esta información es por si llegan a encontrarse con campos con alguna de estas características, entiendan cuál es la configuración que se les ha asignado.

- **size:** define el tamaño de la caja de texto, en número de caracteres visibles. Si al escribir el usuario llega al final de la caja, el texto que escriba a continuación también cabrá dentro del campo, pero irá corriéndose a medida que se escribe, haciendo desaparecer la parte de texto que queda a la izquierda.
- **maxlength:** indica el tamaño máximo del texto, en número de caracteres, que puede ser escrito en el campo. En caso de que el campo de texto tenga el atributo maxlength, el navegador no permitirá escribir más caracteres que los que hayamos indicado.
- **placeholder:** este atributo especifica una pequeña pista que describe el valor esperado para el campo (input). La pequeña sugerencia se muestra en el campo de entrada antes de que el usuario ingrese un valor. Una vez que escriba, ese valor va a desaparecer. Genera un campo de este estilo:

Email

Ingresar aquí tu mail

- **value:** en algunos casos puede resultarnos interesante asignar un valor predefinido al campo en cuestión. Esto puede ayudar al usuario a rellenar más rápidamente el formulario o darle alguna idea sobre la naturaleza de datos que se requieren. *Es importante destacar que esto no significa que el usuario no pueda escribir en dicho campo, si quiere puede cambiar lo que está por defecto, por lo que él quiera*

Por ejemplo, estamos haciendo un formulario para enviar una encuesta a todos los estudiantes del curso de Egg Educación. Vamos a preguntar datos personales, curso y una pregunta de satisfacción. Como sabemos que la mayoría de los estudiantes van a completar el campo con “Egg Educación” lo dejamos por defecto para ahorrar tiempo al usuario. Pero en el caso de nombre y apellido, no proponemos un valor predefinido ya que cada respuesta de ese campo es única. Se vería así:

Curso

Egg Educacion

Camada

¿Qué validaciones podemos hacer?

Antes que nada, el equipo de desarrollo nos debe confirmar si es necesario validar estas configuraciones. Algunas de las formas de validar son:

- Qué el texto desaparezca al llegar al final de la caja pero podamos seguir escribiendo
- Qué los campos al cargar el formulario ya tengan valores
- Qué el texto en gris en un campo desaparezca al escribir en él

Título del campo (etiqueta label)

El campo de texto puesto solo en un formulario va a ser confuso para un usuario, ya que es un recuadro vacío esperando ser completado.

Con título

En este ejemplo vemos un campo con título, otro sin título que el usuario no entendería por qué debe llenarlo y con qué información debería llenarlo.

Es por eso que para indicarle al usuario qué información debería ingresar en el campo o input se incorpora un título a dicho campo, este título suele ponerse con una etiqueta de html llamada **<label>**, por lo que si en algún momento de la guía ven la palabra label, sepan que es el texto que aparece previo a un campo de texto.

Esto en una página se vería así:

Diagrama de un formulario:

- Un recuadro azul con el texto "Label" tiene una flecha que apunta hacia abajo a la palabra "Nombre".
- Debajo de "Nombre" hay un campo de texto (input).
- Debajo del campo de texto hay un recuadro azul con el texto "Campo de texto" que tiene una flecha que apunta hacia arriba al campo de texto.

La **validación** aquí sería que el título **se corresponda con el tipo de campo** a rellenar.

Otros elementos de formularios

Seguramente hayan notado que los inputs son una manera muy práctica de hacernos llegar la información del navegante. No obstante, en muchos casos, permitir al usuario que escriba cualquier texto permite demasiada libertad y

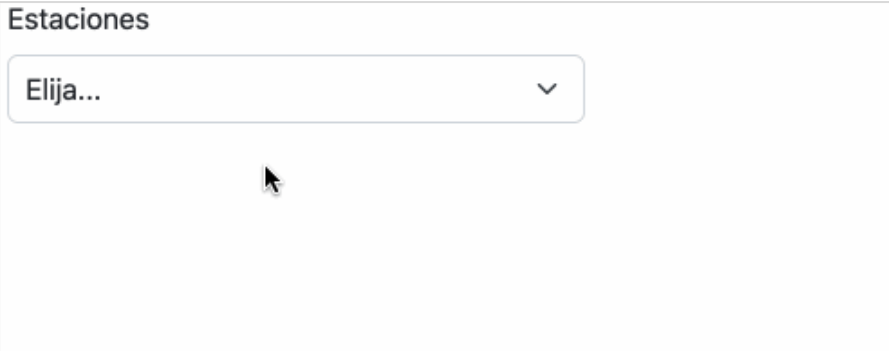
puede que la información que éste escriba no sea la que nosotros estamos necesitando.

Por ejemplo, pensemos que queremos que el usuario indique su país de residencia. En ese caso podríamos ofrecer una lista de países para que seleccione el que sea. Este mismo caso se puede aplicar a gran variedad de informaciones, como el tipo de tarjeta de crédito para un pago, la puntuación que da a un recurso, si quiere recibir o no un boletín de novedades, etc...

Este tipo de opciones predefinidas por nosotros pueden ser expresadas por medio de diferentes campos de formulario. Veamos a continuación cuales son:

1. Listas de opciones

Las listas de opciones son ese tipo de menús desplegables que nos permiten elegir una (o varias) de las múltiples opciones que nos proponen. Esto en una página se vería así:



Validaciones a realizar en las listas de opciones

Al encontrarnos con listas de opciones, nosotros como testers tenemos la tarea de validar los siguientes puntos, para que el formulario funcione de la manera esperada.

- **Datos requeridos:** Que la lista de opciones tenga todas las opciones necesarias, por ejemplo si la lista de opciones es sobre los meses del año, que tenga los 12 meses como opción.
- **Datos seleccionables:** Que la lista de opciones le permita al usuario seleccionar cualquiera de las opciones que se encuentran disponibles.

- **Modificar selección:** Que el usuario pueda cambiar la opción que eligió, por ejemplo, elegí el mes enero y después lo quiero cambiar, que la lista me dejé realizar ese pequeño cambio.
- **Tiempo de respuesta:** Validar que la lista de opciones no tarde demasiado tiempo en mostrar las opciones o en elegir opciones, sino el usuario puede interpretar que durante ese tiempo muerto el proceso se ha trabado
- **Mostrar todos los datos del listado:** Validar que la lista de opciones al abrirla/clickear, muestre todas las opciones, supongamos que si tiene los 12 meses del año pero al clickear no me sale dicha opción para usar.
- **Orden:** Validar que la lista de opciones muestre el orden correcto de las opciones por ejemplo, no puede mostrar los meses así:
 - A. Noviembre
 - B. Mayo
 - C. Enero
- **Selección múltiple:** En algunos casos en las listas de opciones se puede seleccionar más de una opción, validar si esto sería correcto o no según el campo que tiene que completar el usuario.

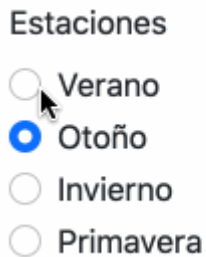
Por ejemplo, si estamos dando una lista de opciones con los meses del año para que ingrese en qué mes nació, no puede poner dos opciones pero, si le estamos preguntando en qué meses es invierno, debería poder elegir más de una opción.

2. Botones de opción

Existe otra alternativa para plantear una elección, en este caso, obligamos al usuario a elegir únicamente una de las opciones que se le proponen. Para esto usamos los botones de opción.

Los botones de opción pueden ser, por ejemplo, una lista de agujeros circulares que pueden contener un espacio blanco (para la opción de “no seleccionado”) o un punto (para la opción de “seleccionado”). Adyacente a cada botón de opción normalmente se muestra un texto que describe la opción que representa el botón de opción.

Un botón de opción se ve normalmente de la siguiente manera:



Validaciones a realizar en los botones de opción

Al encontrarnos con botones de opción, nosotros como testers tenemos la tarea de validar los siguientes puntos, para que el formulario funcione de la manera esperada.

- **Marcar y desmarcar:** Que el usuario pueda cambiar la opción que eligió, por ejemplo, elegí invierno y después lo quiero cambiar, que el botón de opción me dejé desmarcar la opción y elegir otra.
- **Funcionamiento de los botones de opción :** Que el usuario al elegir la opción se marque con algún color o algún cambio en el botón y que cuando desmarque la opción vuelva a su estado original o en blanco.
- **Opciones únicas o opciones múltiples :** En las listas de opciones vimos que dependiendo de lo que le fuéramos a pedir al usuario que rellene, podíamos tener la opción de que eligiera una o múltiples opciones, bueno, esto también es válido para los botones de opción. Por lo que, dependiendo del dato que se le pida al usuario validar si debería poder elegir una o múltiples opciones.
- **Datos requeridos:** Que los botones de opción tengan todas las opciones necesarias, por ejemplo si los botones de opciones son sobre los meses del año, que tenga los 12 meses como opción.

- **Alineación de los botones:** Revisar que la opción se alinee correctamente con la caja a seleccionar por ejemplo:

<input type="radio"/> Andy Weir	<input checked="" type="checkbox"/> The Princess Bride
<input type="radio"/> Matt Damon	<input type="checkbox"/> Star Wars
<input checked="" type="radio"/> Ridley Scott	<input type="checkbox"/> Forrest Gump

Botones

Como podremos imaginarnos, en formularios no solamente habrá elementos o campos donde solicitar información del usuario, sino también habrá que implementar otra serie de funciones. Concretamente, han de permitirnos su envío mediante un botón. También puede resultar práctico poder proponer un botón de borrado o bien un botón de volver.

Botón de envío de formulario (botón submit)

Para dar por finalizado el proceso de relleno del formulario y hacerlo llegar a su gestor, el usuario ha de enviarlo por medio de un botón previsto a tal efecto. Usualmente el botón se encuentra al final del formulario y tiene un texto acorde.


Supongamos que tenemos un formulario de registro, tendríamos un botón que diga “Registrarse”. Esto en la página se verá así:

Formulario de registro

Nombre

Apellido

País

Elija... 

Estado civil

☒ Soltero/a

☐ Casado/a

Registrarse

Validaciones a realizar en el botón de envío

Al encontrarnos con el botón de envío, nosotros como testers tenemos la tarea de validar los siguientes puntos, para que el formulario funcione de la manera esperada.

- **Verificar que el botón sea clickeable:** Que el usuario pueda hacer click en el botón a la hora de enviar el formulario. Y que al hacer click se ejecute la acción esperada, en este caso enviar el formulario.
- **Posición del botón :** Que el botón esté situado en un lugar lógico en el flujo del uso del formulario, por ejemplo, que se encuentre al final del formulario. Esto no es completamente necesario pero ayuda a que el usuario tenga una mejor experiencia.
- **No nulos :** En caso que tengamos datos obligatorios, el botón no nos debería dejar enviar el formulario sin antes llenar dichos campos o no debería mostrar un mensaje de error.
- **Redirección o mensaje de confirmación :** Validar que la página nos redirecciona a alguna página deseada o nos da una mensaje de confirmación ("registro con éxito!!") al hacer click en el botón.
- **Texto del botón :** Validar que el nombre del botón sea el correcto o que sea algo coherente, supongamos que queremos que el usuario se registre el boton debería decir, registrate o submit.

Botón de borrado (botón de reset)

Este botón nos permitirá borrar el formulario por completo, en el caso de que el usuario desee rehacerlo desde el principio.

A diferencia del botón de envío, indispensable en cualquier formulario, el botón de borrado resulta meramente optativo y no es utilizado frecuentemente. Hay que tener cuidado de no ponerlo muy cerca del botón de envío y de distinguir claramente el uno del otro, para que ningún usuario borre el contenido del formulario que acaba de rellenar por error.

Esto en una página se vería así:



Nombre

Apellido

Dirección

Registrarse Restaurar todos los campos

Validaciones a realizar en el botón de borrado

Al encontrarnos con el botón de borrado, nosotros como testers tenemos la tarea de validar los siguientes puntos, para que el formulario funcione de la manera esperada.

- **Verificar que el botón sea clickeable:** Que el usuario pueda hacer click en el botón.
- **Funcionamiento del botón:** Que al clicar el botón se borren los campos rellenados.
- **Posición del botón y visibilidad del botón:** Algunos detalles de posición y visibilidad que podemos validar son:
 - que el botón no se encuentra muy cerca del botón de envío para poder distinguir claramente el uno del otro, y así minimizar la posibilidad de que por error el usuario borre el contenido del formulario que acaba de rellenar.
 - que el botón no haya quedado oculto entre otros elementos del formulario.
- **Texto del botón :** Validar que el nombre del botón sea el correcto o que sea algo coherente, supongamos que queremos que el usuario pueda borrar el formulario el botón debería decir, borrar o resetear.

URL

Hay una cosa extra que podemos validar cuando hacemos click en el botón de envío y es cerciorarnos de que los datos sensibles no se envíen a través de la url.

¿Qué significa esto?

Cuando nosotros enviamos nuestro formulario, toda la información que estaba en los campos se envía al servidor de la página que se encarga de enviarlo a la base de datos y de esa manera estar registrados. Si el formulario no está bien programado, la información va a viajar a la base de datos pero, el usuario va a ver toda la información en la url de la página.

¿Qué peligro conlleva eso?

Además de que hace que el usuario probablemente no esté muy contento con esto, el riesgo real es que los usuarios pueden ver exactamente qué parámetros se envían a su servidor y no solo pueden guardar esa URL con un marcador (para volver a enviar) sino que también pueden modificar la URL para enviar otros parámetros, potencialmente sin sentido, a nuestra base de datos / servidor.

¿Por qué sucede esto?

Esto sucede porque a la hora de enviar nuestro formulario se envía a través de una petición HTTP, las peticiones HTTP son básicamente la manera en la que la página se comunica con el servidor.

Vamos a poner un ejemplo muy sencillo, el usuario teclea en su url **www.ejemplo.com**, el navegador en ese momento envía una petición HTTP al servidor para que traiga la página web. El servidor envía la petición con la página para que el navegador la cargue y por último la muestra.

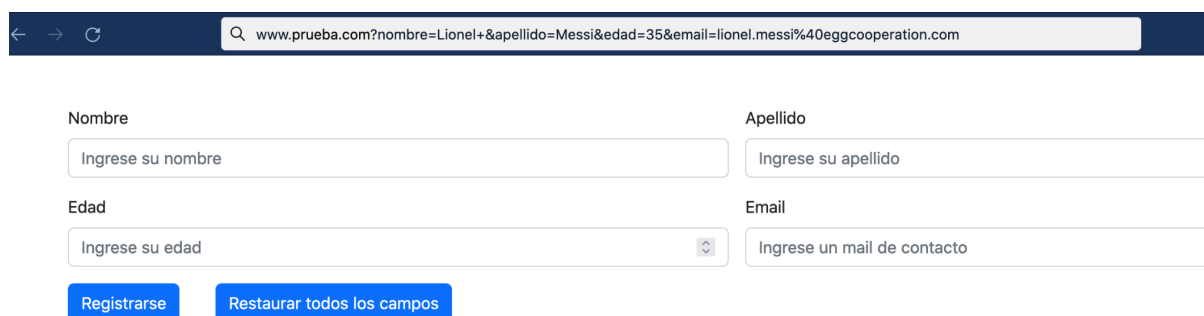
Estas peticiones también se usan para enviar la información, cuando nosotros le damos al botón de submit en nuestro formulario, enviamos una petición HTTP con toda nuestra información para que el servidor la guarde.

Get y Post

Como podemos ver hay dos tipos de peticiones, una que es de traer **(get)** información del servidor y otra que es enviar **(post)** información al servidor. Estos dos métodos, llamados get y post, nos ayudan a definir cada acción que se realiza en nuestro servidor, los programadores definen según necesidad si hacen que "x" acción sea get o post.

El problema con nuestro formulario aparece cuando en vez de poner el formulario con un método post, lo ponemos con un método get y ahí vamos a ver toda nuestra información en la url.

¿Cómo se vería esto?



The screenshot shows a web browser with the address bar containing the URL: `www.prueba.com?nombre=Lionel+&apellido=Messi&edad=35&email=lionel.messi%40eggcooperation.com`. Below the browser, there is a registration form with four input fields: "Nombre" (Ingrese su nombre), "Apellido" (Ingrese su apellido), "Edad" (Ingrese su edad), and "Email" (Ingrese un mail de contacto). At the bottom of the form are two buttons: "Registrarse" and "Restaurar todos los campos".

Como podemos ver en la url se puede ver que ingresó el usuario en cada campo, ahora podría guardar esa url y enviar otros parámetros sin sentido.



The screenshot shows a web browser with the address bar containing the URL: `www.prueba.com?nombre=Lionel+&apellido=Messi&edad=35&email=lionel.messi%40eggcooperation.com`.



Te dejamos [aquí un video](#) de ejemplo de como **validar un formulario HTML**