****

**Manera incorrecta de validar un formulario**

Veamos cómo generar una validación que funcione desde el punto de vista de javascript, pero que no cumpla con las normas de accesibilidad ya que si el usuario desactiva javascript el formulario nunca será enviado:

<script>  
**function validar(f)**{  
if(document.getElementById('mail').value.length<3){//si lo escrito tiene menos de 3 caracteres  
document.getElementById('mail').focus();//enfocar el campo  
alert('complete el campo mail');//lanzar mensaje  
return;//suspender la ejecución de la función   
}  
if(document.getElementById('mail').value.indexOf('.')==-1){//si lo escrito no tiene punto  
document.getElementById('mail').focus();//enfocar el campo  
alert('complete correctamente el campo mail');//lanzar mensaje  
return;//suspender la ejecución de la función  
}  
if(document.getElementById('mail').value.indexOf('@')==-1){//si lo escrito no tiene arroba  
document.getElementById('mail').focus();//enfocar el campo  
alert('complete correctamente el campo mail');//lanzar mensaje  
return; //suspender la ejecución de la función  
}  
//si no se cumplió ninguna de las condiciones, llegamos a este bloque:  
f.submit();//enviamos el formulario usando su método submit  
}  
</script>  
<form id="form1" name="form1" method="post" action="">  
<label>E-mail: <input name="mail" type="text" id="mail" /> </label>  
<input type="button" name="Submit" value="Enviar" onclick="validar(this.form)" />  
</form>

**Manera correcta de validar un formulario**

La manera correcta de validar un formulario es, en lugar de usar el evento onclick sobre un botón tipo button, es usar un botón tipo submit y disparar la función de validación usando el evento onsubmit del tag form.

<script>  
function validar(f){   
if(document.getElementById('mail').value.length<3){//si lo escrito tiene menos de 3 caracteres  
document.getElementById('mail').focus();//enfocar el campo  
alert('complete el campo mail');//lanzar mensaje  
return false;//suspender la ejecución de la función  
}  
if(document.getElementById('mail').value.indexOf('.')==-1){//si lo escrito no tiene punto   
document.getElementById('mail').focus();//enfocar el campo  
alert('complete correctamente el campo mail');//lanzar mensaje  
return false;//suspender la ejecución de la función  
}   
if(document.getElementById('mail').value.indexOf('@')==-1){//si lo escrito no tiene arroba   
document.getElementById('mail').focus();//enfocar el campo  
alert('complete correctamente el campo mail');//lanzar mensaje  
return false;//suspender la ejecución de la función  
}  
//si no se cumplió ninguna de las condiciones, llegamos a este bloque:  
return true;  
}   
</script>  
<form id="form51" name="form51" method="post" action="" onsubmit="return validar(this)">  
<label>E-mail: <input name="mail" type="text" id="mail" /> </label>  
<input type="submit" name="Submit" value="Enviar" />  
</form>

**El código cambió bastante poco, sin embargo, vemos cómo ahora la referencia al formulario, pasada como parámetro de la función de validación, se hace con la palabra this, ya que quien dispara el evento ahora es el propio elemento form. También hay que notar que en la llamada a la función de validación usamos return validar(this), y el sentido del return es el siguiente: cualquier evento declarado en línea con javascript puede ser detenido, en espera de lo que retorne una función, si así lo indicamos con la palabra return antes del llamado a dicha función. Sin embargo, eso nos obliga a que la función retorne un valor Booleano (true o false), que indicará si detener o no el evento disparado. Y eso es precisamente lo que hacemos. En cada condicional, retornamos false, ya que no queremos que el formulario se envíe si se cumple la condición evaluada, y al final (llegar al final implica no haberse cumplido ninguna de las condiciones que llevaban dentro un return) retornamos verdadero para que el formulario se envíe normalmente.**