FrontEndII
Profesor: Martín Bonino

Módulo 4. Validación del lado del Cliente

Sesión 13. Formularios II - ¿Cómo validar?
Marzo 29 de 2022

Eventos de formulario

focus
Sucede cuando el usuario entra con el cursor dentro de un campo input.

blur
El cursor abandona el campo en el que se encuentra.

change
Permite identificar que el valor de un campo ha cambiado.

submit
Identifica el momento en que se hace click en un botón o un input de tipo submit.

this
Esta palabra reservada permite identificar el elemento sobre el que se ejecuta un evento determinado.

this.value
Retorna la infromación dentro del campo que se está analizando.

Validación del lado del Cliente

index.html:

```
<section class="errores">
        <ul></ul>
</section>
// Se crea un elemento tipo ul para desplegar los errores encontrados en la validación del formulario.
```

script.js

```
window.addEventListener(“load”, function() {
// Garantiza que el código solo se ejecuta una vez se haya cargado el formulario

	let formulario = document.querySelector(“form.reservation”);

	formulario.addEventListener("submit", function(event){

		let errores = [];

		let campoNombre = document.querySelector("input.nombre");
		let campoMensaje = document.querySelector(“input.mensaje”);

		if(campoNombre.value == ""){
			errores.push("El campo nombre está vacío");
		}
		if(campoMensaje.value == ""){
			errores.push("El campo mensaje está vacío");
		}
		// Otras validaciones…

		if(errores.length > 0){

			event.preventDefault(); // Detiene el envío del formulario
			let ulErrores = document.querySelector(".errores ul");

			errores.forEach(error => {
				ulErrores.innerHTML += `<li>${error}</li>`
			}); // Muestra los errores detectados en el formulario

		}
	});
}
```

Objeto location

location.href; // Devuelve la url completa del sitio.
https://www.youtube.com/results?search_query=hugo

location.search; // Permite acceder a los parámetros de la búsqueda
search_query=hugo

location.reload(); // Se recarga el sitio web

Query String
Cuando se realiza una petición GET en una búsqueda, location provee el atributo search que permite obtener el query string entero. Para utilizar esos datos se utiliza la interfaz de URLSearchParams.

Ejemplo:

```
let query = new URLSearchParams(location.search);

if(query.has('search_query')){
	let search = query.get('search_query');
	console.log(search)
};
```

Sincrónico

stoppropagation: Detiene la propagación hacia arirba del evento

Eventos:

- blur: al abandonar el control
- change: cambia al abandonar el control
- input: A medida que se va escribiendo

Normalización después de validar.

Expresiones regulares:
https://regexr.com/

Sesión 14. JSON y Storage
Marzo 31 de 2022

JSON
Es un formato de texto sencillo para el intercambio de datos, cuya implementación proviene de la notación de objetos de JavaScript. Está compuesto por parejas clave-valor, en las que las propiedades van siempre entre comillas dobles.

JSON vs Objeto literal:

Objeto literal	JSON
Admite comillas simples y dobles	Las claves van entre comillas
Las claves del objeto van sin comillas	Sólo se pueden usar comillas dobles
Podemos escribir métodos sin problemas	No admite métodos, sólo propiedades y valores
Se recomienda poner una coma en la última propiedad	No podemos poner una coma en el último elemento

Métodos de conversión:

JSON.parse();
Analiza una cadena de texto JSON, la cual recibe por parámetro. Retorna un objeto de JS que se corresponde con el texto parseado.

```
JSON.parse(‘{“nombre”:”Mauricio”}’); // Devuelve {nombre: ‘Mauricio’}
JSON.parse('{}'); // {}
JSON.parse('true'); // true
JSON.parse('"hola!"'); // "hola!"
```

JSON.stringify();
Recibe un objeto o valor de JavaScript y lo convierte. Retorna una cadena de texto con el formato correspondiente a la notación JSON.

```
JSON.stringify({nombre:’Mauricio’}); // Devuelve ‘{“nombre”: “Mauricio”}’
JSON.stringify({}); // '{}'
JSON.stringify(true); // 'true'
JSON.stringify('chau!'); // '"chau!"'
```

SessionStorage y localStorage

Session Storage
Permite guardar información en la sesión. Si se cierra el navegador, la información almacenada se pierde.

sessionStorage.setItem(‘key’, ‘valor’); // Solo se almacenan strings en formato string
sessionStorage.getItem(‘key’);
sessionStorage.removeItem(‘key’);

Local Storage
Los datos almacenados en localStorage no tienen fecha de expiración.

localStorage.setItem(‘key’, ‘valor’); // Solo se almacenan strings en formato string
localStorage.getItem(‘key’);
localStorage.removeItem(‘key’);
localStorage.clear();

Sincrónico

cookies: Pequeños archivos que se guardan en el computador. Pesan varios kV. Normalmente se manejan desde el BackEnd.
Local Storage se maneja desde el navegador.
El localStorage solo es accesible por la páginma que lo creó. Es más seguro que las cookies.

El sessionStorage persiste dentro de la pestaña, pero no en una pestaña diferente.

Sesión 15
Abril 1 de 2022.

Sincrónico

El último botón de un formulario por defcto se toma como submit (si no tiene definida la propiedad)

heatmap js: Librería que permite detectar el movimiento del mouse.