7. Elementos Interactivos

Elementos Interactivos

- Formularios
  - Manipulación
  - Validación
- Crear y destruir nodos
- Eventos
  - Ratón
  - Teclado
  - Formularios (`change` y `submit`)

Formularios

Para acceder o modificar el valor de un campo:

```
$("selector input").val()
```

Devuelve el valor del campo como una cadena

```
$("selector input").val("valor")
```

Cambia el valor del campo por "valor"

Formularios

Ejercicio: tema7/ejercicio1/index.html

- Click en “alert”: se muestra un alert con el contenido del input
- Click en “borrar”: se vacía el campo de texto

Formularios

Ejercicio: tema7/ejercicio2/index.html

- Se mostrará el mensaje en “#salida”
- Dependiendo del valor de “#color”, se modificarán los estilos de “#salida” para que se muestre del color adecuado

Formularios

Validación...

- ¡Es más difícil de lo que parece!
- Necesitamos un nuevo artefacto: Expresiones Regulares
- Un lenguaje arcano y críptico (¡mucho más que javascript!)
- Inagotable fuente de dolor de cabeza

Expresiones Regulares

Describir el formato de una cadena de texto

- Se formulan como patrones
- Tiene su propio vocabulario
- Se define una estructura
- Y después se comprueba si una cadena la satisface o no la satisface

Expresiones Regulares

Malas noticias:

- Muy complicado
- Muy proclive a errores
- Muy necesario

Expresiones Regulares

Malas noticias:

- Muy complicado
- Muy proclive a errores
- Muy necesario

Buenas noticias:

- Rara vez tendremos que escribirlas nosotros
- Solo por esta vez: ¡Copia y pega sin vergüenza!

Expresiones Regulares

```
var regexEmail = /^(([^<>()[\]\\.,;:
\s@\"]+(\.[^<>()[\]\\.,;:\s@\"]+)*)|
(\".+\"))@((\[[0-9]{1,3}\.[0-9]
{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|
(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/;
```

Expresiones Regulares

```
var noVacioRegex = /[\w|\d]+/;

var numeroRegex = /^\d+$/;
```

Expresiones Regulares

Comprobar si una cadena satisface una E. R.

```
"una cadena".match(regEx);
```

Expresiones Regulares

```
> var noVacioRegex = /[\w|\d]+/
  undefined
> "".match(noVacioRegex)
  null
> "asdf".match(noVacioRegex)
  ["asdf"]
```

Expresiones Regulares

Ejercicio: tema7/ejercicio3/index.html

- Al hacer click en `#validar`
  - Se comprobará que el primer campo sea un email
  - Que el segundo no sea vacío
  - Que el tercero sea un número
- En caso de error:
  - Se añadirá la clase “error” al `<label>` y al `<input>` correspondiente
  - Se mostrará el `<small>` correspondiente
- Si es correcto, informar al usuario

Crear y destruir nodos

Para crear un nuevo nodo:

1. Pasar una cadena de texto con el HTML a jQuery
2. jQuery devuelve el nodo creado
3. Lo podemos añadir a la página con:

```
$("selector").append(nodo);
```

Al final de los hijos de “selector”

```
$("selector").prepend(nodo);
```

Antes del primer hijo de “selector”

Crear y destruir nodos

```
> var nodo = $("<h1>")
  undefined
> nodo.html("Stairway to Heaven")
  [<h1>Stairway to Heaven</h1>]
> $("body").prepend(nodo)
  [▼<body>                                ]
      <h1>Stairway to Heaven</h1>
    ▶<div id="container">…</div>
    </body>
```

Crear y destruir nodos

Ejercicio: tema7/ejercicio4/index.html

- Al hacer click en `#anadir` se añade una nueva tarea a `#lista` y se borra el input
- Construye la tarea utilizando el contenido de `#template` como plantilla
- Si te sobra tiempo: ¡valida que no se puedan crear tareas vacías!

Crear y destruir nodos

Destruir un nodo es muy sencillo:

```
$("selector").remove();

$(nodo).remove();
```

Crear y destruir nodos

Ejercicio: tema7/ejercicio4/index.html

- Modifica el código del ejercicio anterior
- Para que, al hacer click en las "**x**" de las tareas
- La tarea sea eliminada
- Extra: Que haga un `fadeOut` y, al terminar, sea eliminada

Eventos

Además de `click`, jQuery nos permite capturar muchos otros eventos, por ejemplo:

- Ratón:

  `mouseenter` y `mouseleave`

  `mousemove`

  `dblclick`

- Teclado:

  `keypress, keyup, keydown`

- Formularios:

  `change, focus, blur`

  `submit`

Eventos

Ejercicio: tema7/ejercicio5/index.html

- Experimenta con diferentes eventos
  - focus y blur
  - keypress, keydown y keyup
  - focus, blur y change

Eventos

Cuando le pasamos un manejador a jQuery...

- Manejador = la función que le pasamos al evento
- Cuando el evento se dispara, jQuery invoca a nuestro manejador
- Y le pasa un parámetro: **e**
- **e** contiene información adicional sobre el evento

Eventos

Prueba a ejecutar algo así:

```
$("#boton").click(function (e) {
  console.log(e);
});
```

Eventos

```
$("#boton").click(function (e) {
  console.log(e);
});
[<a id="boton" class="button twelve columns" href="#"> Botón </a>]
▶ jQuery.Event {attrName: undefined, srcElement: <a>,
  relatedNode: undefined, screenY: 303, fromElement: null…}
```

Eventos

El parámetro e tiene muchos datos. Cosas importantes:

- `e.currentTarget`: el nodo que ha disparado el evento
- `e.type`: el tipo de evento
- `e.keyCode`: el código de la tecla que se ha pulsado (si es un evento de teclado)
- `e.clientX, e.clientY`: las coordenadas en las que se ha producido el evento

Eventos

Ejercicio: tema7/ejercicio5/index.html

- Modifica el código para que muestre información relevante sobre el evento (e.type, e.keyCode, etc...) en #info

Eventos

Ejercicio: tema7/ejercicio6/index.html

- Lee el código y fíjate cómo se está utilizando..
  - `e.currentTarget` para sacar el nodo que originó el evento
  - `parent()` para navegar por el DOM

Eventos

Muchos eventos tienen un significado predefinido

Por ejemplo:

- Hacer click en un `<a>` hace que el navegador navegue a su `href`
- Hacer click en el botón `submit` de un formulario hace que el formulario se envíe

Eventos

jQuery nos permite abortar el comportamiento predefinido de un evento con:

```
e.preventDefault()
```

Eventos

Ejercicio: abre una página con jQuery...

- Teclea en la consola:

```
$("a").click(function (e) {
  console.log("bloqueado!");
  e.preventDefault();
});
```

- ¡Ahora haz click en cualquier link!