[OSAKamp] ASOCIACIÓN DE EMPRESAS DE TECNOLOGÍAS LIBRES Y CONOCIMIENTO ABIERTO DE EUSKADI

01 – HTML y CSS

Teoria

CODE4JOBS

Bizkaia toru aldundia diputación foral

[esle]

HTML

Formularios

Los formularios se usan para recoger datos de varios inputs de nuestra página web. Por ejemplo, durante el registro de un usuario querremos recoger información como nombre, email, password, etc. Y que toda esa información se envíe de una vez a una aplicación de back-end, la cual tratará esos datos y devolverá una respuesta. Hay varios elementos de formularios que podemos utilizar, cada uno tiene su función dentro de los formularios.

La etiqueta de un formulario es **<form></form>** y sigue la siguiente sintaxis:

```
<form action="direccionUrl" method="GET|POST">
    <!-- Aquí van los elementos como inputs, botones, etc. -->
</form>
```

Como veis, la etiqueta form acepta varios atributos (y alguno más que no está puesto), veamos por el momento esos dos que tenemos ahí.

- **Action:** la dirección a la que enviaremos los datos del formulario. Si lo dejamos vacío, por defecto nos cargará otra web en el navegador
- **Method:** Los formularios por defecto solo manejan peticiones tipo GET y POST. Las peticiones de tipo GET se usan principalmente para pedir datos al back-end, y las peticiones de tipo POST se usan para enviarle nosotros datos al back-end (más adelante trataremos esto en profundidad). Si lo dejamos vacío, por defecto será de tipo GET

Elementos de los formularios

Hay diferentes elementos de formularios que podemos utilizar para recoger datos:

- Text input
- Checkbox input
- Radio Box input
- Select Box input
- File Upload input
- Button input

Todos los elementos tienen unas propiedades comunes, que nos sirven para poder interactuar con ellos y que el formulario, al enviarse, sepa qué está enviando. Esas propiedades son

- Type: indica el tipo de input (text, password, radio, etc.)
- Name: indica el nombre que va a tener el input
- Value: indica el valor que va a contener el input

Vamos a verlos en detalle

Text input

Los text input aceptan un string o texto y hay varios que podemos usar: input de una línea, input de múltiples líneas e input de tipo password.

Los inputs de una línea se usan para recoger datos simples, como un nombre, y para los cuadros de búsqueda (Google usa este input, por ejemplo).

```
<form>
      First name: <input type = "text" name = "first_name" />
      <br>
      Last name: <input type = "text" name = "last_name" />
   </form>
```

Los inputs de múltiples líneas se usan para recoger strings más elaborados, como pudiera ser una opinión, un comentario, etc (podemos pensar en Twitter). Los atributos rows y cols sirven para determinar el tamaño en filas y columnas del textarea.

```
<textarea rows="5" cols="50" name="description">
      Enter description here...
   </textarea>
```

Los inputs de password se usan para esconder la contraseña. En vez de aparecer lo que escribimos, aparecerán los clásicos puntos de cualquier campo contraseña.

```
<form>
      User ID : <input type="text" name="user_id" />
      <br />
      Password: <input type="password" name="password" />
    </form>
```

CheckBox Input

Los checkbox se usan cuando queremos que el usuario seleccione varias opciones entre todas las que ofrecemos. Si en el input anterior el type era de tipo text, en este el type será de tipo checkbox. Los checkbox tienen la propiedad checked, que puede usarse para tener una de las opciones seleccionada por defecto.

```
<form>
      <input type = "checkbox" name = "maths" value = "on"> Maths
      <input type = "checkbox" name = "physics" value = "on"> Physics
   </form>
```

Radio Button Input

Estos se usan para que el usuario elija una de las opciones de entre todas las que le ofrecemos. Aquí la propiedad type tendrá el valor radio. También cuenta con la propiedad checked.

```
<form>
    <input type = "radio" name = "subject" value = "maths"> Maths
    <input type = "radio" name = "subject" value = "physics"> Physics
 </form>
```

Select Box Input

Un select, también llamado dropdown, nos permite tener una lista desplegable con varias opciones, de las cuales un usuario puede seleccionar una de ellas. Tiene su propia etiqueta (**<select></select>**). En un select, las diferentes opciones las mostraremos como etiquetas **<option></option>**, las cuales tienen, además del resto de propiedades, las propiedades selected (para dejar seleccionada una opción por defecto) y disabled (para que una opción no se pueda seleccionar).

```
<select name="dropdown">
      <option value="Maths" selected>Maths</option>
      <option value="Physics" disabled>Physics</option>
    </select>
```

File Upload Input

SI queremos permitir la carga de un archivo, necesitaremos usar el input con type file. Tiene la propiedad específica accept, la cual indica qué tipo de archivo es aceptado. Tambien tiene la propiedad multiple, para indicar que vamos a aceptar múltiples archivos en vez de uno solo.

```
<form>
      <input type = "file" name = "fileupload" accept = "image/*"
multiple />
</form>
```

Button Input

Existen dos tipos de botones. Unos ya los hemos visto, los que son **<button></button>**. Los otros son inputs, y en el type pondríamos submit (para realizar el envío del formulario) y reset (para resetear el formulario). Existe alguno más, pero nos centraremos en estos dos.

```
<form>
      <input type="submit" name="submit" value="Submit" />
      <input type="reset" name="reset" value="Reset" />
    </form>
```

Hidden Input

Un input de type hidden no aparecerá en el navegador, pero al enviar el formulario también se tendrá en cuenta y su valor será enviado al back.

```
<form>
      <input type = "hidden" name = "pagename" value = "10" />
    </form>
```

CSS

Background

Background hace referencia, en los elementos HTML, al fondo de cada uno de los elementos. Cuando utilizamos la propiedad **background-color** y seleccionemos un color, el elemento HTML pasará a tener ese color de fondo.

Html:

```
<body>
  <div id="back-color"></div>
</body>
```

Css:

```
#back-color {
  height: 2000px;
  background-color: indigo;
}
```

También podemos poner una imagen de fondo en vez de un color. Para ello, usaremos **background-image: url(“”)** y dentro de las comillas pondremos el enlace a una imagen tanto de internet como de nuestro propio proyecto.

```
#back-color {
  height: 2000px;
  background-color: indigo;
  background-image: url('https://qph.cf2.quoracdn.net/main-qimg-
32f8e54649d71cf48b4890606c428721-lq');
}
```

SI la imagen es más pequeña que el elemento HTML, la replicará hasta cubrir al completo el elemento. Esto está bien si queremos un patrón, pero si es una imagen única podemos usar **background-repeat** con el valor **no-repeat**. De esta forma evitaremos la repetición.

Si lo que queremos es que esa imagen se repita sólo en el eje horizontal, usaremos el valor **repeat-x**; y si lo queremos en el eje vertical, usaremos **repeat-y**.

```
#back-color {
  height: 2000px;
  background-color: indigo;
  background-image: url('https://qph.cf2.quoracdn.net/main-qimg-
32f8e54649d71cf48b4890606c428721-lq');
  background-repeat: repeat-x;
  background-repeat: repeat-y;
  background-repeat: no-repeat;
}
```

Está puesto aquí para que lo tengáis, pero no podéis usarlas todas juntas porque se pisarían.

Si la queremos centrada, podemos usar la propiedad **background-position** con uno de sus valores (center, por ejemplo) para ubicar nuestra imagen en un lugar específico del fondo.

```
#back-color {
  height: 2000px;
  background-color: indigo;
  background-image: url('https://qph.cf2.quoracdn.net/main-qimg-
32f8e54649d71cf48b4890606c428721-lq');
  background-repeat: no-repeat;
  background-position: center;
}
```

También podemos darle un tamaño a nuestra imagen de fondo. Para eso, la propiedad **background-size** será la que usemos. Podemos darle un tamaño, o podemos usar el valor **cover** para que ocupe todo el elemento. Este valor siempre hará que la imagen se ajuste hasta alcanzar la altura del elemento HTML, y la cortará o dejará hueco vacío en los costados si la anchura es menor a la del elemento.

```
#back-color {
  height: 2000px;
  background-color: indigo;
  background-image: url('https://qph.cf2.quoracdn.net/main-qimg-
32f8e54649d71cf48b4890606c428721-lq');
  background-repeat: no-repeat;
  background-position: center;
  background-size: cover;
}
```

Por último, si queremos que la imagen se quede fija aunque nosotros hagamos scroll en la pantalla, podemos usar la propiedad **background-attachment** con el valor **fixed**.

```
#back-color {
  height: 2000px;
  background-color: indigo;
  background-image: url('https://qph.cf2.quoracdn.net/main-qimg-
32f8e54649d71cf48b4890606c428721-lq');
  background-repeat: no-repeat;
  background-position: center;
  background-size: cover;
  background-attachment: fixed;
}
```

Y así tendríamos nuestro elemento con una imagen de fondo centrada, escalada y fijada.

Google Fonts

Cuando queremos cambiar la tipografía de nuestro proyecto, normalmente las tipografías que podremos usar serán las que tengamos por defecto en nuestro ordenador. Para utilizar otras, Google nos ofrece Google Fonts, que es un servicio de fuentes gratuito al que accedemos desde https://fonts.google.com

Aquí podremos seleccionar una fuente, copiar el link e incluirlo en la etiqueta **<head></head>** de nuestro HTML, de la siguiente forma:

```
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <title>Document</title>
    <link href="https://fonts.googleapis.com/css?family=Notable"
rel="stylesheet" />
    <link rel="stylesheet" href="style.css" />
  </head>
```

Aquí nos hemos importado la fuente 'Notable'. Tras esto, podremos usar la fuente como valor en la propiedad font-family

HTML:

```
<div id="back-color">
      <div id="fuente-interna">
        <p>
          Lorem ipsum dolor sit amet consectetur adipisicing elit.
Provident
          fugiat ab ducimus temporibus qui repudiandae laudantium at
veritatis
          nulla explicabo. Cupiditate, voluptas! Consequuntur cupiditate
          repellendus, autem ratione aliquam molestiae neque?
        </p>
      </div>
      <div id="fuente-google">
        <p>
          Lorem ipsum dolor sit amet consectetur adipisicing elit.
Provident
          fugiat ab ducimus temporibus qui repudiandae laudantium at
veritatis
          nulla explicabo. Cupiditate, voluptas! Consequuntur cupiditate
          repellendus, autem ratione aliquam molestiae neque?
        </p>
      </div>
    </div>
```

CSS:

```css
#fuente-interna {
  background-color: green;
}

#fuente-google {
  background-color: blue;
  font-family: 'notable', sans-serif;
}
```

He creado dos divs con un párrafo cada uno para que se vea la diferencia de fuentes. El primero es la estándar, si no usamos Font-family esa es la que por defecto usará nuestro proyecto. Y el segundo utiliza la fuente recién importada desde Google Fonts.

Media Queries

Los media queries son puntos de ruptura en nuestros estilos. En ellos podremos indicar desde qué tamaño a qué tamaño (en pixels) de pantalla queremos que se apliquen nuestros estilos. Para el ejemplo que venimos trabajando de la foto y los textos, podremos usar por ejemplo un media query que, desde los 767px de pantalla hacia abajo, modifique los estilos de nuestro HTML.

```css
@media (max-width: 767px) {
  #back-color {
    display: flex;
    justify-content: center;
    align-items: center;
    background-size: 100%;

  }
}
```

Entre 0 y 767px de tamaño, se aplicarán estos estilos. Y a partir de los 767px, se aplicarán los estilos que hemos determinado antes.

Aunque ahora no lo estemos haciendo, por norma el desarrollo de estilos se hace primero para pantallas pequeñas y después se añaden los estilos para pantallas más grandes. Lo haremos más adelante. Por último, tenéis que tener en cuenta que para cambiar un estilo o propiedad de un elemento con media query, tenéis que sobreescribirlo, como el **background-size** que hemos sobreescrito ahí arriba. Ahora la imagen no cubre todo el elemento, sino que mantiene su tamaño original.

Hay varios puntos de ruptura a tener en cuenta: 500, 600, 767, 1024 y 1400 pixels.

Pseudo-clases

Las pseudo-clases se utilizan para definir un estado especial de un elemento. Por ejemplo, un enlace **<a>** tiene su estado normal y su estado visitado. Vamos a jugar con eso para crear diferentes estilos dependiendo de cómo esté el enlace. Añadiremos una **<a>** después de la etiqueta p del primer div (id fuente-interna), y después le daremos estilos para su estado normal y para cuando hayamos hecho click en él (visitado).

HTML:

```
<div id="fuente-interna">
        <p>
          Lorem ipsum dolor sit amet consectetur adipisicing elit.
Provident
          fugiat ab ducimus temporibus qui repudiandae laudantium at
veritatis
          nulla explicabo. Cupiditate, voluptas! Consequuntur cupiditate
          repellendus, autem ratione aliquam molestiae neque?
        </p>
        <a href="https://google.com">Google</a>
      </div>
```

CSS:

```
a {
  color:red;
}

a:visited {
  color: orange;
}
```

Podemos darle también una pseudo-clase cuando pasemos el ratón sobre el enlace con **hover**:

```
a:hover {
  color: greenyellow;
}
```

Hover sirve para cualquier elemento del HTML y cambiará la apariencia del mismo cuando pasemos el puntero sobre él.

Podemos seleccionar también determinados elementos hijos de un elemento padre. Con **:first-child** modificaremos el comportamiento del primer hijo. **:nth-child(n)** (**n** hace referencia a la posición del elemento hijo) elegiremos el de la posición indicada. **:nth-child(odd)** y **:nth-child(even)** selecciona los elementos impares y pares respectivamente. Estas pseudo-clases se ponen en los elementos que se repiten, no en el padre de esos elementos. En este ejemplo, las pseudo-clases van sobre los **
**, no sobre la **
**

HTML:

```
<ul>

        <li>Audi</li>
        <li>BMW</li>
        <li>Mercedes</li>
        <li>Masserati</li>
      </ul>
```

CSS:

```
li:first-child {
  font-size: 50px;
}
```

```
li:nth-child(even) {
  font-size: 50px;
}
```

```
li:nth-child(odd) {
  font-size: 50px;
}
```

Aquí también, si utilizáis todos a la vez se pisarán, así que usadlos cada uno por separado para ver el resultado.

Pseudo-elementos

Los pseudo-elementos se usan para seleccionar una parte de un elemento. Por ejemplo, si quisiéramos cambiar la primera línea de una ** ** usaremos **::first-line**

CSS:

```
p::first-line {
  font-size: 50px;
}
```

Si queremos cambiar también la primera letra de esa **p**, usaremos **::first-letter**

CSS:

```
p::first-letter {
  font-size:100px;
}
```