Estilos CSS en formularios

Gracias a los estilos podemos romper el diseño gris con líneas negras de los formularios y convertirlos en una parte más que se integra perfectamente en nuestro diseño web.

Como sabemos, a la hora de crear nuestros formularios, seguiremos buenas prácticas CSS, no crearemos clases extra innecesarias y utilizaremos, en la medida de lo posible, selectores descendientes y etiquetas predefinidas.
A las etiquetas de los formularios se les puede dar estilo CSS al igual que al resto de elementos de nuestro documento. Sin embargo, debemos conocer una serie de propiedades que nos ayuden a elaborar nuestros diseños de la forma más óptima posible.

1. Propiedad *box-sizing*

Por defecto en el modelo de cajas de CSS, el ancho y alto asignado a un elemento es aplicado solo al contenido de la caja del elemento. Si el elemento tiene algún borde (border) o relleno (padding), este es entonces añadido al ancho y alto del tamaño de la caja o contenedor. Esto significa que cuando se define el ancho y alto, se tiene que ajustar el valor para permitir cualquier borde o relleno que se pueda añadir.

La propiedad `box-sizing` puede ser usada para ajustar el siguiente comportamiento:

- `content-box` es el comportamiento CSS por defecto para el tamaño de la caja (box-sizing). Si se define el ancho de un elemento en 100 pixeles, la caja del contenido del elemento tendrá 100 pixeles de ancho, y el ancho de cualquier borde o relleno será añadido al ancho final desplegado.
- `border-box` tiene en cuenta cualquier valor que se especifique de borde o de relleno para el ancho o alto de un elemento. Es decir, si se define un elemento con un ancho de 100 pixeles. Esos 100 pixeles incluirán cualquier borde o relleno que se añada, y la caja de contenido se encogerá para absorber ese ancho extra. Esta propiedad es especialmente útil para redimensionar cualquier elemento.

Vamos a ver un ejemplo: en el caso de los inputs, normalmente les añadimos un padding para darle algo de aire. Sin embargo, si después centramos el contenido y le dotamos de un «width:100%», este ancho no tendrá en cuenta el padding añadido al input y el elemento no se encontrará perfectamente redimensionado. Realiza la prueba quitando y añadiendo la propiedad «box-sizing: border-box» en el código del formulario básico creado a continuación.

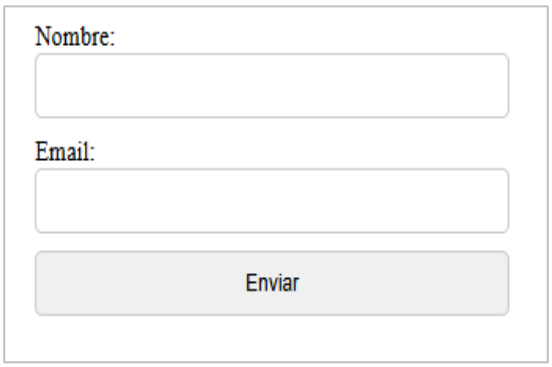
HTML:

```
<form>
    <label for="nombre">Nombre:</label>
    <input type="text" id="nombre" name="nombre">

    <label for="email">Email:</label>
    <input type="email" id="email" name="email">

    <input type="submit" value="Enviar">
</form>
```

CSS:

```css
input {
    -webkit-box-sizing: border-box;
    box-sizing: border-box;
    padding: 10px;
    width: 100%;
    margin-bottom: 10px;
    border: 1px solid #ccc;
    border-radius: 5px;
}
form{max-width: 500px; margin: 0 auto;}
```

Resultado:



Como puedes ver en el código, se han añadido los prefijos para navegadores necesarios para esta nueva propiedad.

```
-webkit-box-sizing: border-box;
box-sizing: border-box;
```

2. Propiedad *resize*

Por defecto, los elementos `<textarea>` permiten cambiar el tamaño por el usuario. Podemos anular este comportamiento utilizando la propiedad «*resize:none*«.
Valores: **none** | **both** | **horizontal** | **vertical**

Valores	Descripción
`none`	Permite cambiar el tamaño del elemento
`horizontal`	Permite cambiar el tamaño del elemento horizontalmente
`vertical`	Permite cambiar el tamaño del elemento verticalmente
`both`	Permite cambiar el tamaño del elemento en horizontal y vertical

Ejemplo:

HTML:

```html
<h2>Ejemplo de Textarea con resize</h2>
    <textarea placeholder="Escribe aquí...">
</textarea>
```

CSS:

```css
textarea {
            width: 300px;
            height: 150px;
            padding: 10px;
            border: 1px solid #ccc;
            border-radius: 5px;
            resize: none; /* Evita que el usuario
pueda redimensionar el textarea */
```

Resultado:

3. Estilos en el input: pseudo-clase `:focus`

Cuando un usuario interactúa con un input, ya sea para ingresar texto o seleccionar una opción, es importante proporcionar retroalimentación visual para mejorar la experiencia del usuario. Una forma de lograr esto es aplicando estilos específicos cuando el input está en estado `:focus`.
El pseudo-clase `:focus` se utiliza para aplicar estilos a un elemento cuando recibe el foco de atención. Por ejemplo, cuando un usuario hace clic en un input, este entra en estado `:focus`.
Veamos algunas técnicas comunes para estilizar inputs en estado `:focus`:

3.1. Eliminar el contorno predeterminado

Por defecto, los navegadores suelen agregar un contorno alrededor del input cuando está en estado `:focus`. Este contorno puede no ajustarse al diseño de tu sitio web o aplicación. Para eliminarlo, puedes usar la propiedad `outline`:

```
1.  input:focus {
2.      outline: none;
3.  }
```

2. Estilizar el fondo y el borde

Puedes personalizar el fondo y el borde del input cuando está en estado `:focus` para resaltar su estado activo. Por ejemplo:

```
1.  input:focus {
2.      background-color: #f0f0f0;
3.      border: 1px solid #ccc;
4.  }
```

3. Animaciones y transiciones

Agregar animaciones o **transiciones** sutiles puede hacer que la interacción con los inputs sea más atractiva. Por ejemplo, puedes animar el cambio de color del borde cuando el input está en estado `:focus`:

```
input {
    transition: border-color 0.3s ease;
}
input:focus {
    border-color: #007bff;
}
```

Ejemplo:

En el siguiente ejemplo se muestra cómo aplicar estilos a un input cuando está en estado `focus` y cómo incluir una transición para el color del borde.

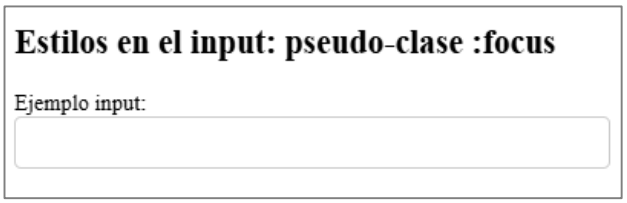
Html:

```
<h2>Estilos en el input: pseudo-clase :focus</h2>

<div class="input-container">
  <label for="example">Ejemplo input:</label>
  <input type="text" id="example" name="example">
</div>
```

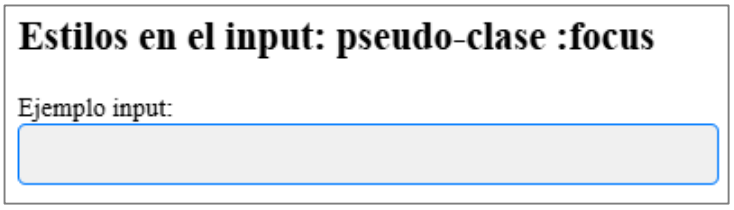
CSS:

```
.input-container {
    margin-bottom: 20px;
}
input {
    padding: 10px;
    width: 100%;
    border: 1px solid #ccc;
    border-radius: 5px;
    transition: border-color 0.3s ease; /* Transición para el color del borde */
    box-sizing: border-box;
}
input:focus {
    background-color: #f0f0f0;
    border-color: #007bff;
    outline: none;
}
```

Resultado inicial:



Resultado con focus:



4. Estilos para inputs

Los estilos en los inputs pueden mejorar significativamente la estética y la experiencia de usuario de tus formularios. Veamos algunos estilos creativos que puedes aplicar para personalizar tus inputs:

4.1. Estilo básico

El estilo básico es simple y funcional, ideal para una apariencia clásica y limpia.

HTML

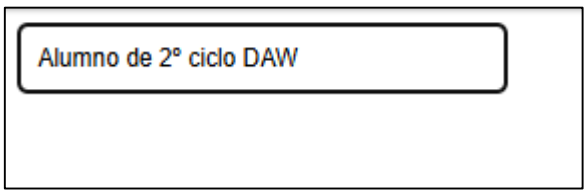
```html
<input type="text" class="basic-input" placeholder="Nombre">
```

CSS:

```css
.basic-input {
    border: 1px solid #ccc;
    border-radius: 5px;
    padding: 10px;
    width: 250px;
}
```

Resultado inicial:

Resultado final:



4.2. Bordes redondeados

Los bordes redondeados pueden hacer que tus inputs se vean más suaves y modernos.

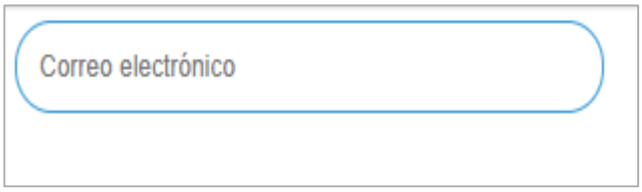
Html:

```html
<input type="text" class="rounded-input" placeholder="Correo electrónico">
```

Css:

```css
.rounded-input {
    border: 2px solid #3498db;
    border-radius: 20px;
    padding: 12px;
    width: 300px;
}
```

Resultado:



4.3. Iconos en inputs

Incorporar iconos o etiquetas dentro del input puede mejorar la usabilidad y el diseño general.

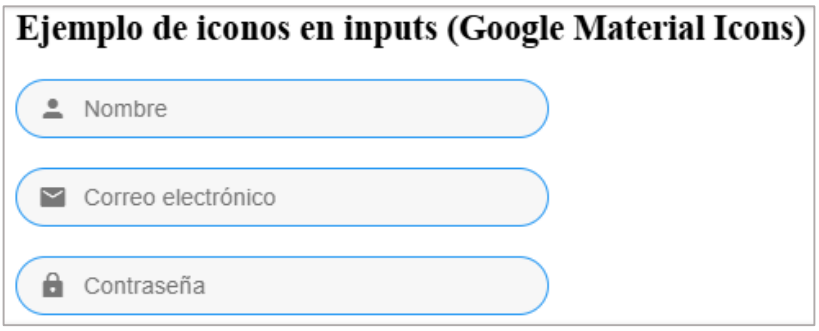
Html:

```html
<div class="input-container">
  <i class="material-icons input-icon">person</i>
  <input type="text" class="icon-input" placeholder="Nombre">
</div>

<div class="input-container">
  <i class="material-icons input-icon">email</i>
  <input type="text" class="icon-input" placeholder="Correo electrónico">
</div>

<div class="input-container">
  <i class="material-icons input-icon">lock</i>
  <input type="password" class="icon-input" placeholder="Contraseña">
</div>
```

Css:

```css
.input-container {
  position: relative;
  margin-bottom: 20px;
}
.icon-input {
  padding: 10px 15px 10px 45px;
  border: 2px solid #2196F3;
  border-radius: 25px;
  width: 300px;
  font-size: 16px;
  color: #333;
  background-color: #f7f7f7;
  outline: none;
}
.input-icon {
  position: absolute;
  top: 50%;
  left: 15px;
  transform: translateY(-50%);
  color: #777;
  font-size: 20px;
}
```

Navegador:



En este ejemplo, se ha utilizado los iconos de Google Material Icons. Asegurate que esté correctamente incluido en la sección <head> de tu documento HTML. Para ver cómo hacerlo ir a la página de google iconos (también puedes ver NOTA1. Al final de este archivo)

5. Estilos para inputs requeridos, asterisco en label

Veamos ahora cómo aplicar estilos específicos a los inputs que son marcados como requeridos en el formulario. Esto se hace mediante la adición de un indicativo, por ejemplo un asterisco, al final de las etiquetas `<label>` correspondientes a los campos obligatorios. Estos estilos ayudan a resaltar visualmente los campos que deben ser completados por el usuario.
En el siguiente ejemplo se usa el **pseudoelemento** `:after` y la propiedad `content` para insertar un asterisco al final de los label que tienen la clase `required`.

```
label.required::after {
    content: "*";
    color: red;
    margin-left: 5px;
}
```

Ejemplo:

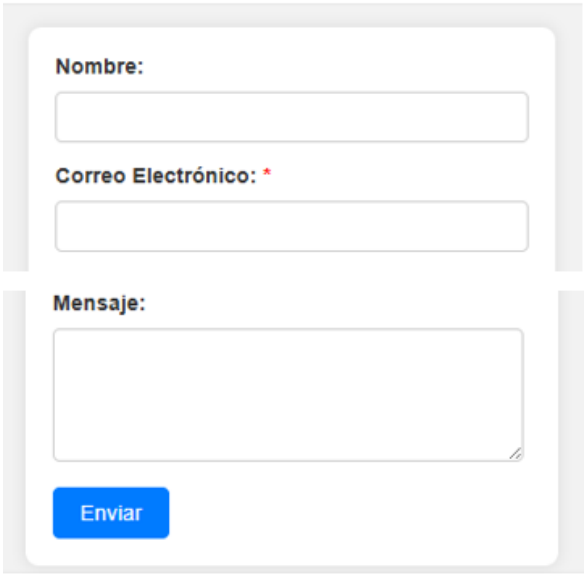
Html

```
<form>
  <div>
    <label for="nombre">Nombre:</label>
    <input type="text" id="nombre">
  </div>
  <div>
    <label for="email" class="required">Correo Electrónico:</label>
    <input type="email" id="email">
  </div>
  <div>
    <label for="mensaje" class="inline">Mensaje:</label>
    <label for="mensaje" class="inline">Mensaje:</label>
    <textarea id="mensaje" rows="4" cols="50"></textarea>
  </div>
  <button type="submit">Enviar</button>
</form>
```

Css: (parte del código)

```
label {
  display: block;
  font-weight: bold;
  margin-bottom: 5px;
  color: #333;
  font-size: 16px;
  padding-bottom: 5px;
  font-family: 'Arial', sans-serif;
}
label.required::after {
  content: "*";
  color: red;
  margin-left: 5px;
}
label.inline {
  display: inline-block;
  margin-right: 10px;
}
```

Navegador:



NOTA 1. Como incluir un icono de google material icono en un archivo html-css

Paso 1: Enlaza los Material Icons a tu HTML

Añade la siguiente línea en la sección `<head>` de tu documento HTML para incluir la biblioteca:

```
<link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
```

Paso 2: Estructura el HTML

Crea un `div` contenedor con `position: relative` para poder posicionar el icono de forma absoluta dentro del input. Luego, añade el `<input>` y el `<span>` con el ícono.

html

```
<div class="input-container">
  <input type="text" placeholder="Buscar...">
  <span class="material-icons icon-inside">search</span>
</div>
```

Paso 3: Aplica estilos con CSS

Usa CSS para posicionar el icono dentro del input. El `div` contenedor debe tener `position: relative` y el ícono debe tener `position: absolute`, `margin-left` y `margin-top` para ubicarlo correctamente.

css

```
.input-container {
  position: relative;
  display: inline-block; /* Para que el div no ocupe todo el ancho */
}

.input-container input {
  padding-left: 30px; /* Espacio para el icono */
}

.icon-inside {
  position: absolute;
  left: 10px;
  top: 50%;
  transform: translateY(-50%); /* Centra verticalmente el icono */
  color: #777;
}
```