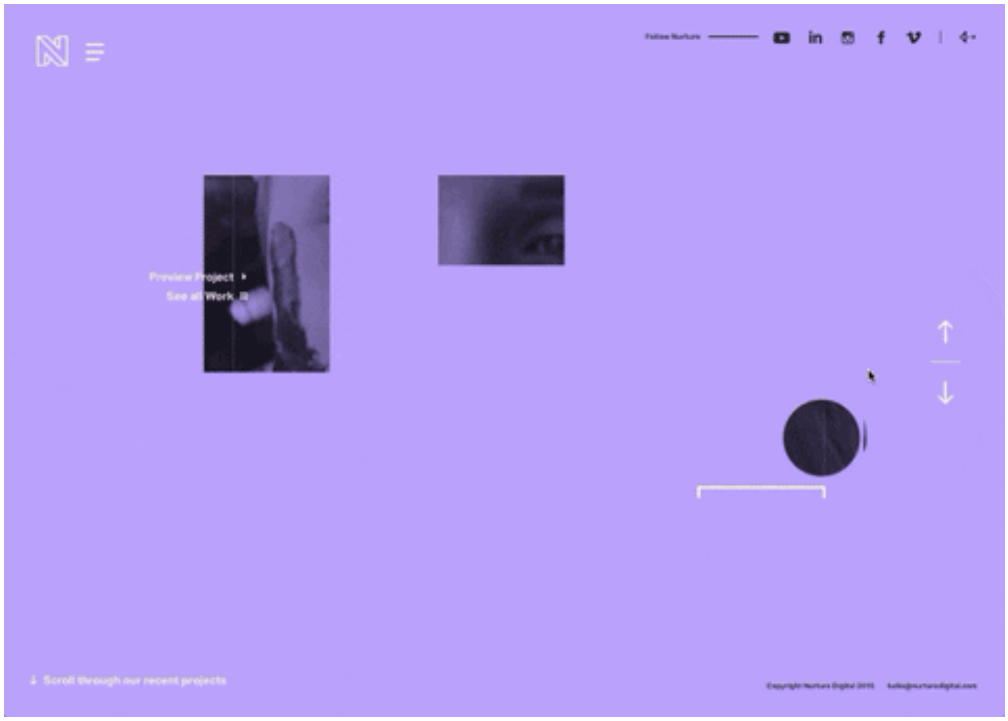
UT 7.- Contenidos multimedia en la Web: Animaciones

7.1. ¿Qué son las animaciones web?

Las animaciones web, como su propio nombre indica, son una serie de efectos visuales que puedes aplicar en los diferentes elementos que componen una web (bloques de texto, botones, banners, etc.) creando una sensación en ella de movimiento.

Por ejemplo, lo que puedes ver en la siguiente imagen son varias animaciones web en una slider de imágenes. Cada diapositiva de la slider contiene a su vez otras animaciones en botones, imágenes, textos, etc.



Por regla general, las animaciones web que se implementan en la actualidad se crean con una o varias de las siguientes tecnologías y/o lenguajes de programación:

- HTML5
- CSS3
- Javascript / Jquery
- Imágenes animadas (GIFS)

Hay más posibilidades. Pero estas cuatro son las que se utilizan más y con más frecuencia.

IMPORTANTE: Como puedes ver, no se ha incluido en ellas Flash.

Esta es una tecnología obsoleta que no se debe utilizar para nada a la hora de crear un blog o web. Existen varios motivos, pero los más importantes son:

1. El contenido generado en flash no se muestra en algunos dispositivos móviles, los cuales en la actualidad son muy utilizados.

2. No es un estándar ni cumple con los estándares web.

3. El flash no es visible de cara al SEO y, en consecuencia, a los buscadores.

4. Es pesado y aumentará los tiempos de carga de tu web considerablemente.

5. Ha dado problemas de seguridad.

7.1.1. Cómo utilizar animaciones en tu web correctamente

A. Las animaciones web son necesarias y útiles en algunos casos muy concretos.

Siempre que pienses en utilizar una animación web, piensa primero por qué introducirla, que vas a conseguir con ella o qué te aporta de cara a los objetivos que te has propuesto alcanzar.

Esta afirmación puede ser muy abstracta y difícil de entender, así que veremos ejemplo práctico:

Imagina que estás pensando en meter en una página de un servicio que quieres vender en tu web una serie de animaciones que son:

1. Una imagen de una mariposa de colores volando por toda tu web.

2. Una animación en un texto en el que hablas del problema que afecta a tu cliente y la solución que tú le aportas para arreglar este problema.

3. Una animación en el botón de compra de tu servicio.

Ahora te tienes que preguntar…

1. ¿Por qué razón meter la mariposa de colores?
2. ¿Por qué motivo meter una animación en el texto?
3. ¿Qué conseguiré introduciendo una animación en el botón de compra?

B. Utiliza las animaciones web con criterio y sentido.

Una vez respondida la pregunta de por qué meter esa animación web que tienes en mente, ya tienes un análisis en tu poder de lo que dichas animaciones pueden aportar.

Por tanto, ahora puedes decidir siguiendo un criterio lógico qué animaciones web utilizar en el diseño de tu blog o web y cuáles debes desechar.

O lo que es lo mismo: puedes utilizarlas con sentido para conseguir un objetivo concreto o mini objetivos que te ayuden a lograr tu objetivo principal.

Vamos a seguir con el ejemplo anterior…

1. **La mariposa de colores** puede ser muy bonita a nivel estético, pero pensándolo detenidamente no aporta nada más que eso: estética

   Por tanto, en principio se descartaría.

2. **La animación de entrada en el texto** sí que parece más importante.

   Quieres que la persona que entra en esta página de tu servicio lea este texto sí o sí para que sea consciente tanto del problema que tiene como de la solución que tú le ofreces.

   Al introducir una animación de entrada en este texto concreto, si esa persona hace scroll y ve un texto que entra moviéndose, casi inconscientemente se parará a ver que pone.

3. **En cuanto a la animación del botón**, al igual que ocurría con el texto, también puede ser importante introducirla para que este sea más visible de cara al usuario que está leyendo la página de tu servicio.

   Sin embargo, por otro lado, puedes destacar este botón de otra manera en un primer punto de vista (por ejemplo, con un color diferente que destaque) y luego utilizar la animación cuando el usuario pase el ratón por encima de él (dejando más claro aún que es un botón, una llamada a la acción).

   En este caso, queda un poco más a criterio personal.

   Si quieres destacar el botón desde el primer momento, mete la animación directamente al cargarse el botón.

   Si en cambio, crees que no es tan importante esta animación de entrada y puedes destacarlo con el color, la segunda opción es mejor. Además,

con ella conseguirás ahorrar un poco de tiempo de carga, que nunca viene mal.

C. No abuses de las animaciones web

Quizás al hacer el análisis, bajo tu criterio, te parece que hay muchas animaciones importantes.

Como resultado, al implementarlas en tu web o blog… eso parece un tiovivo que no para de moverse.

Es importante hacer un segundo filtro y dejar sólo las imprescindibles.

Abusar de las animaciones web puede generar el efecto contrario que buscas.

No sólo puedes hacer que la experiencia de usuario dentro de tu plataforma sea peor, sino que además puedes conseguir que con tanta animación realmente nada destaque.

No debes pasarte poniendo animaciones web a todo lo que tú consideras importante. Una vez más, simplificar todo lo máximo posible es la clave.

7.2. Animaciones con CSS

Las animaciones son transiciones entre elementos que se definen mediante reglas de estilo en CSS. Se distinguen dos partes fundamentales en la creación de dichas animaciones, por un lado, los fotogramas, que indican el estado inicial y final de estas, y, por otro, el estilo CSS, que define la transición de la animación. Hasta la aparición de este nuevo mecanismo, las técnicas de animación estaban basadas en scripts descritos en lenguaje Java Script. Una de las ventajas fundamentales es que la animación se muestra eficientemente en cualquier tipo de dispositivos.

7.2.1. La propiedad `animation'

La propiedad `animation' presenta diversos parámetros que permiten configurar su funcionamiento. Cabe mencionar que, con esta propiedad, solo se determinan las propiedades de la animación, pero la apariencia de esta se define a través de @keyframes. Los parámetros más habituales son:

1. animation-delay. Define el tiempo que transcurre entre el momento de carga del elemento y el momento de inicio la animación.

2. animation-direction. Define la dirección de reproducción de la animación tras finalizar. Existen dos opciones: que retroceda hasta el fotograma de inicio al finalizar la secuencia o que comience siempre desde el principio

3. animation-duration. Tal y como su nombre indica, este parámetro define el tiempo que la animación tarda en completar su ciclo de acción.

4. animation-iteration-count. Define el número de veces que se repite la animación; para indicar que se reproduzca indefinidamente toma el valor infinite.

5. animation-play-state. Al utilizar esta propiedad es posible parar y reanudar la secuencia de la animación

6. animation-tíming-function. Define el ritmo con el que la animación se muestra; es decir, la velocidad de los fotogramas de la animación.

7. animation-fill-mode. Define qué valores toman las propiedades de la animación tras su finalización.

8. animation-name. Como se ha visto, las características de distribución y apariencia de la animación se definen mediante la regla en CSS @keyframes; ahora bien, este atributo de la propiedad `animation' permite dar nombre a la regla a la animación.

Dentro de las reglas de apariencia definidas por @keyframes se aconseja definir el tiempo y el ritmo de la animación; para ello se utiliza percentage, para indicar el momento en el que tiene lugar la secuencia de la animación:

Inicio: 0 % (from).

Fin: 100 % (to)

En el siguiente ejemplo, el elemento de texto “Hola” se desplaza por la pantalla comenzando, aproximadamente, en la zona central de la pantalla (200%) y terminando en el extremo izquierdo. A través del parámetro animation-duration se establece que la animación tendrá una duración de 6 segundos desde el inicio al fin. Mediante el parámetro animation-name se define el nombre de la regla @keyframes que describe la apariencia de los fotogramas de la animación.

Además, en la mitad de la secuencia, se indica que el valor de la letra aumente hasta un 200 %, y, a continuación, que decrezca hasta llegar al tamaño inicial. El regreso al valor original del tamaño de letra se produce cuando el elemento llega a la posición marcada por margin-left.

• Ejemplo:

```
<!DOCTYPE html>

<html>
```

```
<head>
  <style>
    h1 {
      animation-duration: 6s;
animation-name: estiloAnimacion;
    }

    @keyframes estiloAnimacion {
      from {
margin-left: 50%;
width: 200%;
      }
      50% {
font-size: 200%;
margin-left: 10%;
width: 150%;
      }
      to {
        margin-left: 0%;
        width: 100%;
      }
    }
  </style>
</head>
<body>
  <h1>Hola</h1>
</body>
</html>
```

7.2.2. La propiedad ‘transition'

Aquí se verá otra de las propiedades fundamentales de los elementos de animación en CSS, la propiedad `transition', que permite modificar el valor de la velocidad, es decir, el ritmo al que se producen las transiciones entre los estados definidos para cada animación y que aportan la sensación de movimiento.

• Sintaxis para la propiedad `transition':

transition <propiedad> <duracion> <funcion-tiempo> <retraso>

• Ejemplo en CSS:

```
etiquetaHTML{ transition<propiedad> <duracion> <funcion-tiempo>
        <retraso>
}
```

Esta propiedad está formada por varios parámetros:

1. transition-property. Especifica el nombre o nombres de las propiedades CSS a las que deberían aplicarse las transiciones.

2. transition-duration. Define el valor de la duración en la que sucederán las transiciones. Es posible definir tantos valores corno se desee para cada una de las propiedades de transición.

3. transition-timing-function. Define cómo se computan los valores intermedios para las propiedades. Puede ser:

    a) ease. Valor por defecto. Comienza lento, luego aumenta su velocidad y, finalmente, concluye lentamente.

    b) linear. Presenta la misma velocidad durante toda la duración de la transición.

    c) ease-in. Se define como comienzo lento.

    d) ease-out. Comienzo rápido y final lento.

    e) ease-in-out. El comienzo es lento, a continuación, aumenta de velocidad y, concluye lento de nuevo.

4. transition-delay. Define el tiempo de espera entre el momento en que se cambia una propiedad y el inicio de la transición.

En el siguiente ejemplo, se aplica la regla de color general a azul y, cuando el cursor del ratón pase por encima del elemento marcado, el color pasará a ser verde. La duración de la animación, es decir, lo que tarda el elemento en volver a cambiar de color será de 2 segundos, y la velocidad del cambio será constante, definida por el valor linear.

• Ejemplo de animación de texto:

```
<!DOCTYPE html>
<html>
<head>
  <style>
    a {
      text-decoration: none;
      color: blue;


    }


    a:hover {
color: green;
    }
a {
 transition: color 2s linear;
 }


  </style>
</head>
<body>
  <a href="#">Soy un camaleón.</a>
</body>
</html>
```

Es posible animar casi cualquier elemento que se pase por la imaginación, no solo el texto. En este último ejemplo, al pasar el cursor del ratón sobre una determinada zona de la imagen donde hay situado un cuadrado de color rojo, se produce la animación. Como se ha dicho, se parte de un elemento cuyas dimensiones son de 100 px de altura por 100 px de ancho, de color rojo. Cuando se pasa el ratón por encima del elemento, se desea que su apariencia cambie. Estas propiedades se recogen en otra regla.

El cambio, el paso entre estos dos estados se define a través del atributo `transition', que, como ya se ha visto, indica las propiedades que varían (width, height y margin) y que la duración de este cambio, 2s.

• Ejemplo de animación de un elemento:

```
<!DOCTYPE html>
<html>
<head>
        <style>
                .transition {
                        width: 100px;
                        height: 100px;
                        background: red;
                    transition: width 2s, height 2s, margin 2s;
                      margin: 50px auto 0;
                }
                .transition:hover {
                        width: 100%;
                        height: 200px;
                        margin: 0 auto;
                }
        </style>
</head>
<body>
      <div class="transition"></div>
</body>
</html>
```

7.2.3. La propiedad steps ()

Finalmente, se hablará de una última función, llamada step(), que permite controlar el valor de las transiciones de los diferentes segmentos que componen una animación. Esta propiedad hace que sea posible trabajar con

diferentes momentos de la animación, ampliando así el espectro de posibilidades de diseño del movimiento y la animación.

```
<!DOCTYPE html>
<html>
<head>
 <style>
section
.circle {
      background: rgba(191, 63, 116, 1);
          width: 400px;
          height: 400px;
          border-radius:
200px;          margin:
100px auto;  display:
block;
      transition: background 1s steps(4);
 }
 section .circle:hover {
      background: rgba(83, 180, 191, 1);
 }
 </style>
</head>
<body>
 <section>
 <div class="circle"></div>
 </section>
</body>
</html>
```

En el ejemplo anterior, se representa un círculo que cambia de color cuando el cursor del ratón pasa por encima de este elemento. En la regla marcada mediante la expresión section .circle {} se definen los parámetros de diseño del

círculo (color, tamaño, radio,...) -y„ a través de la propiedad `transition', se indica que la animación va a modificar el color de fondo del circulo (background) en cuatro momentos, es decir, en cuatro pasos (`steps(4)').

7.3. Elementos interactivos con CSS

Los elementos interactivos permiten acercar la comunicación entre el sitio web y el usuario, esto eso, hacerlo partícipe y que consiga una experiencia de navegación única, como si la página estuviera diseñada para él, creando una web más dinámica. Por lo tanto, para conseguir un sitio interactivo, el foco principal del diseño de la web siempre será el usuario. Importante, no deben perderse de vista la accesibilidad ni la usabilidad a costa de ganar interactividad.

7.3.1. ¿Qué son los elementos interactivos?

Se define como elemento interactivo aquel que cambia su estado o su valor cuando el usuario interactúa con él o sobre él. Existen multitud de elementos interactivos; a continuación, se verán algunos ejemplos:

a) **Enlaces.** Cuando el usuario actúa sobre ellos, se produce un cambio en la página. bien por el redireccionamiento hacia a otro sitio o página web, bien por la descarga de algún, tipo de fichero (con la propiedad `download'), etc. En función de si un enlace ha sido visitado o no, se puede modelar su apariencia.

b) **Botones.** La acción de pulsar un botón puede implicar múltiples acciones, desde el
envío de un formulario al borrado de la información contenida en un formulario, la selección de una opción, etc. De nuevo, al igual que ocurre con los enlaces, el estado de un botón cambia en función de si ha sido pulsado o no.

c) **Cuadros de texto**. Son elementos en los cuales el usuario puede escribir una
determinada información, por lo tanto, su aspecto va cambiando.

d) **Casillas de verificación**. Más conocidas como cuadros check, permiten escoger una
serie de opciones; dependiendo de si están seleccionados o no, su estado es diferente.

7.3.2. Comportamientos interactivos

Los elementos anteriores, de por sí, no implican un comportamiento interactivo; este debe ser añadido a través de los elementos, propiedades y atributos que se han visto a lo largo de este módulo. En concreto, bien a través de reglas de estilo, bien a través de los lenguajes de programación dinámicos.

En cuanto a las reglas de estilo, se utilizan las pseudoclases: link, visited, hover, active y focus.

En HTML5 hay algunos elementos cuyo funcionamiento, sin utilizar reglas de estilo, ya proporcionan la interactividad que se recoge en este apartado; estos son details, summary y dialog.

Algunos de estos elementos ya se vieron en el capítulo dedicado al lenguaje de marcas HTML, pero se desarrollan con mayor en profundidad a continuación.

A) Etiqueta <details>

Esta etiqueta mantiene oculta una zona de la página web a través de un icono en forma de triángulo, que se sitúa en la parte derecha del elemento. Si pulsa sobre este, se hace visible la parte recogida dentro de esta etiqueta y, cuando se vuelve a pulsar, se oculta o viceversa.

• Sintaxis para la etiqueta <details>:

oculto-visible:

<details> …. </details>

Visible-oculto: <details open>… </details>

Sera el atributo 'open', situado en la etiqueta <details>, el que determine si el elemento objeto de la interacción aparece visible al inicio de la carga de la pagina o no.

• Ejemplo sin atributo 'open'; no aparece visible.

```
<p>Si no aparece el atributo open incluido en la etiqueta details_</p>

 <details>

        <p>... este texto aparece oculto hasta que se despliega pulsando sobre
        Detalles</p>
</details>
```

• Ejemplo con atributo 'open'; aparece visible:

```
<p>Si el atributo open está incluido en la etiqueta details...</p>

 <details open>

        <p>... este texto aparece visible hasta que se pulsa sobre Detalles</p>

</details>
```

B) La propiedad 'summary'

La propiedad `summary' siempre aparece contenida dentro de una etiqueta <details>. Se trata del elemento que modifica el valor de la etiqueta de texto que aparece junto al icono de triangulo sobre el que se debe pulsar para

mostrar o ocultar el texto. Si no se utiliza este elemento, se muestra el valor por defecto "detalles".

• Sintaxis para la propiedad 'summary':

```
<details >
        <summary>  ….
        </summary>
</details>
```

•Ejemplo con atributo 'open"; aparece visible:

```
<p> Si el atributo open está incluido en la etiqueta details..</p>
<details open>
        <summary> Pulsa </summary>
         <p>… este texto aparece visible hasta que se pulsa sobre Detalles</p>
</details>
```

C) La etiqueta ‘dialog’

Por último, la etiqueta <dialog> define el contenido de una ventana de diálogo independiente de la página web, es decir, abre una nueva ventana que se superpone a la principal. Puede incorporar el atributo `open', que indica que el cuadro de diálogo está activo y el usuario puede interactuar con él. La sintaxis es muy sencilla.

• Sintaxis para la propiedad `dialog':

```
<dialog>…</dialog>
```

La posición por defecto en la que se mostrará este elemento es centrada horizontalmente. De la misma forma, la apariencia por defecto será con color de fondo blanco. Si se desean modificar estas y otras propiedades, es posible hacerlo utilizando el pseudoelemento ::backdrop.

```
<!DOCTYPE html>
<html>
<head>
  <title>Ejemplo de cuadro de diálogo con HTML y CSS</title>
        <style>
        dialog::backdrop {
                background-color: rgba(255, 255, 0, 0.7);
            }
        </style>
</head>
```

```
<body>
        <button onclick="document.querySelector('dialog').show()">Abrir cuadro
de diálogo</button>
        <dialog>
                <p>Cuadro de diálogo</p>
        </dialog>
</body>
</html>
```