GRADO EN INGENIERÍA INFORMÁTICA

CURSO 2018-2019

UNIVERSIDAD DE GRANADA

PROGRAMACIÓN WEB

PRÁCTICA EVALUABLE 1

David Peinado Perea

Se pretende hacer un sitio web para la recomendación de libros utilizando únicamente HTML5 y CSS3. Este desarrollo no debe incluir librerías tipo Bootstrap, tampoco se puede usar JavaScript de modo que todo el contenido dinámico debe ser desarrollado usando sólo estas tecnologías.

El sitio web tiene una estructura clásica, con un archivo principal index.html y desde él se podrá acceder al resto, una vez el usuario se registre en la página o se autentifique se le dará acceso a la totalidad del sitio.

Este sitio tiene secciones como:

- Foro, en el cual se podrán ver los hilos más recientes, así como crear nuevos hilos.
- Recomendaciones, un sistema de recomendación implementado en la siguiente práctica .
- Mis libros, los libros que haya marcado como leídos un usuario, así como los últimos libros introducidos en la aplicación y un formulario para dar de alta nuevos libros.
- Mis datos, donde el usuario podrá cambiar sus datos públicos.

Primero voy a explicar los elementos comunes a las páginas y luego me centraré en elementos específicos de cada una.

Las páginas se estructuran de la siguiente manera:

- Hay un header o cabecera en la parte superior de la página, dividido en 3 partes, el logo, el nombre de la aplicación y el nombre del usuario y un enlace para desconectarse (en caso de index es un formulario de identificación y un enlace de registro).
- Salvo en index, por lo anteriormente mencionado, en todas las páginas hay una barra de navegación o nav, esta tiene las 4 secciones ya dichas.
- El contenido principal de cada página.
- Un footer o pie de página con el contacto y el enlace a este pdf.

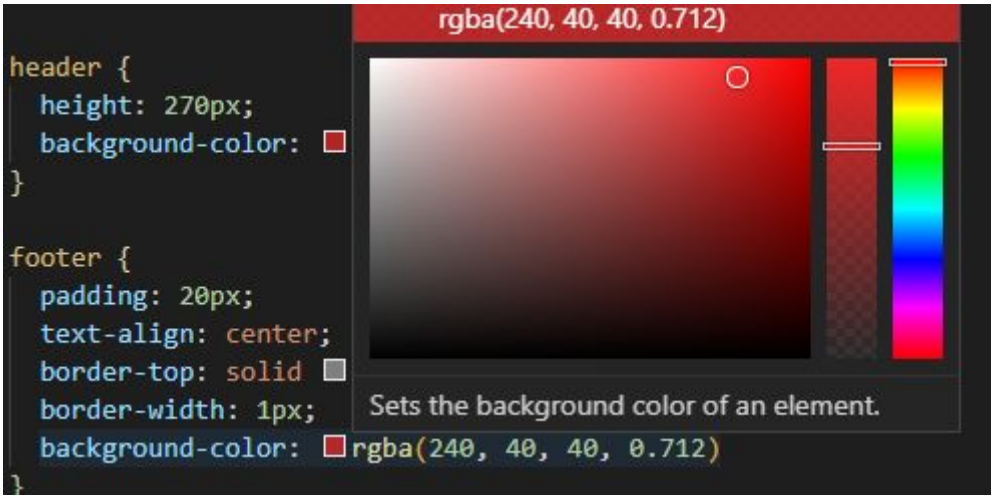
Se ha recurrido a skeleton para hacer más fácil el desarrollo.
Se utilizan mucho los bordes redondeados como se puede observar.
Se ha modificado según la página y las necesidades que esta tenga.
Los botones, salvo en la página foro y los de valoración de libros, siguen las reglas CSS de Skeleton.
En el header se usan las clases three columns y six columns para posicionar el logo, nombre de la aplicación e información de usuario.

Para la barra de navegación se hace una lista con las secciones a mostrar.
Se aplican animaciones cuando el ratón pasa por encima de una sección.
Para ello se usan propiedades como transition y el pseudo elemento after.

```css
nav li.different:hover {
  border: none;
}

nav li:hover {
  border-bottom: 5px solid #FFFFFF;
}

.different::after {
  content: '';
  position: absolute;
  width: 0px;
  height: 5px;
  left: 50%;
  bottom: 0;
  background-color: white;
  transition: all ease-in-out .2s;
}

.different:hover::after {
  width: 100%;
  left: 0;
}
```

index.html

Para los colores, tanto de esta página como del resto, se usa mayoritariamente la notación rgba, ya que Visual Studio Code implementa unas características que hacen la selección de colores muy fácil. Al poner el nombre de un color, si se mantiene el ratón sobre él aparece una ventanita que permite modificar fácilmente el color. Se usa la misma gama cromática para toda la web, excepto para los hilos del foro.



alta_libro.html

Se importan fuentes de google para esta y otras páginas.
La gran novedad que se introduce, es la de los botones para valorar la satisfacción de un libro. Para ello se han utilizado múltiples reglas CSS, todas en la familia de la clase .btns.

Para que queden así los colores se ha utilizado la función linear-gradient().
Se han utilizado sombras, tanto para el texto como para las cajas, y se ha usado la función cubic-bezier() junto a la propiedad color(reutilizada) y al intervalo de tiempo para hacer la transición.
cubic-bezier() hace que una transición no sea lineal, si no que sea personalizable, puede ser muy lenta al principio y al final y muy rápida en su parte media por ejemplo.
Más información sobre transiciones aquí.
Para poner el check se ha utilizado la propiedad `content: '\2714';` este es el valor que representa el check, se puede ver la lista aquí.

El resultado es bastante llamativo y vistoso.

Para hacer obligatorios todos los campos simplemente se añade la propiedad required en cada input.

También se ha introducido una simple transición para que cuando un campo de texto se selecciona, éste se alargue, para eso basta con añadir estas líneas:

```
input[type="email"], input[type="number"], input[type="searc
  padding: 6px 10px;
  /* The 6px vertically centers text on FF, ignored by Webki
  background-color: #fff;
  border: 1px solid #d1d1d1;
  border-radius: 4px;
  box-shadow: none;
  float: left;
  width: 55%;
  transition: width 0.4s ease-in;
}

input[type="email"]:focus, input[type="number"]:focus, input
  border: 1px solid #33c3f0;
  outline: 0;
  width: 75%;
}
```

Los valores predeterminados como ease-in, ease-out, etc. actúan como funciones cubic-bezier() predeterminadas, para que el usuario no tenga que centrarse en la parte técnica y sólo tenga claro el concepto que quiere realizar

Estas mismas novedades, aunque no todas, se incluyen en otras páginas como alta_usuario.html, crear_hilo.html, libroleido1.html, mis_datos.html y valoracion_libro1.html

foro.html

Para el foro se ha implementado la característica de que la vista de los hilos cambie si se pasa el ratón sobre ellos, con una simple transición y un cambio de color de fondo y cambio de tamaño de fuente.

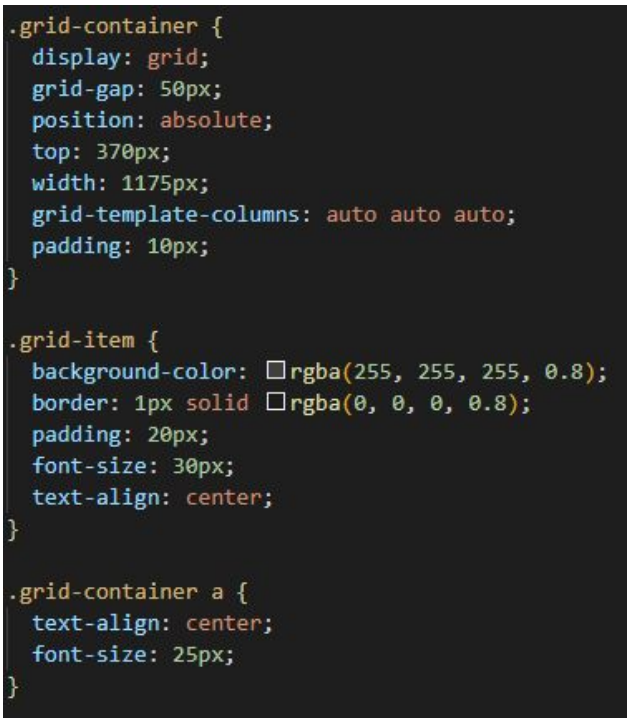
hilo1.html

Para los hilos se han realizado diversas novedades, como en otras páginas, la propiedad `cursor: pointer;` hace que cuando se pasa el ratón por el elemento, aparezca como clickable. Se utiliza la propiedad flex, pero esta se explicará con más detalle en mis_libros.html.

Se obliga al botón de respuesta a estar siempre en mayúsculas con `text-transform: uppercase;`. Cuando se clicka, obtiene una opacidad de 0.7.
Se ha optado por hacer ese diseño de fotos de usuario simplemente añadiendoles un borde circular de 50px.

libro1.html

Se ha tomado la decisión de usar grid en vez de tablas HTML para la “tabla” de opiniones de usuarios, consideré que es más personalizable la primera opción y más versátil. Aquí se puede ver más sobre estos elementos.



```css
.grid-container {
  display: grid;
  grid-gap: 50px;
  position: absolute;
  top: 370px;
  width: 1175px;
  grid-template-columns: auto auto auto;
  padding: 10px;
}

.grid-item {
  background-color: rgba(255, 255, 255, 0.8);
  border: 1px solid rgba(0, 0, 0, 0.8);
  padding: 20px;
  font-size: 30px;
  text-align: center;
}

.grid-container a {
  text-align: center;
  font-size: 25px;
}
```

Con estas reglas estaría hecho todo lo relacionado con la tabla de opiniones de usuarios.

recomendaciones_u1.html

Se utiliza `overflow: auto;` para poder desplazarse con la rueda del ratón por la parte izquierda de la pantalla.

mis_libros.html

La página responsive. Para hacerla así se ha utilizado mayormente Flexbox.

Para hacer responsive la navbar, se ha dispuesto la siguiente media query:

```css
nav a {
  text-decoration: none;
  color: #fff;
  display: block;
  transition: 0.3s background-color;
}

@media screen and (min-width: 1000px) {
  nav li {
    width: 230px;
    border-bottom: none;
    height: 50px;
    line-height: 50px;
    font-size: 1.4em;
  }
  nav li {
    display: inline-block;
    margin-right: -4px;
  }
}
```

Que dispone los 4 elementos como bloques, a 1 por cada línea, si la pantalla tiene menos de 1000px, y los pone todos en una línea(inline-block) si supera ese umbral, también reduce el tamaño de la fuente. A su vez, hace que la animación ocupe toda la pantalla, ya que el elemento en el que está así lo hace.

Para el header se hace `display: flex;` en el header y a cada uno de los 3 elementos que lo componen se le da una anchura de 40%, luego con ayuda de la media query `@media screen and (max-width: 1000px)` se ponen los elementos con un 100% de anchura y `flex-direction: column;` Así los elementos se apilan verticalmente, en el eje Y.

También se ajustan los tamaños de las fuentes y el tamaño de la imagen, se centran haciendo uso de la palabra clave !important, ya que había conflicto con la herencia de la propiedad padding en estos elementos.
!important hace que prime el valor que se está dando en ese momento a la propiedad en la cual se llama

Se ajustan los tamaños de fuente en el footer.

Para hacer lo mismo con el contenido principal se crean 2 columnas distintas, izquierda y derecha. El elemento main queda así:

```css
main {
  margin: 0px auto;
  display: flex;
  justify-content: space-around;
}
```

Para que se apilen una encima de otra cuando se llegue a una anchura determinada(flex) y se distribuyan con espacio entre ellas y entre los bordes de la pantalla(space-around). Asimismo se ajustan los tamaños de fuente también.