



Páginas
multidispositivo

Enfoques de diseño

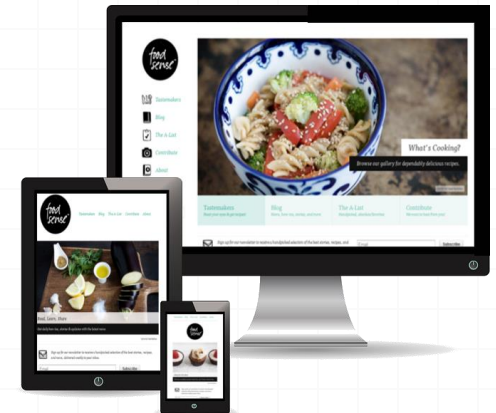
- o A la hora de diseñar una web multidispositivo podemos optar por diferentes enfoques:
 - o Diseño líquido o fluido (Liquid or fluid layout)
 - o Diseño Web Responsivo (Responsive Web Design)

Diseño líquido o fluido (Liquid or fluid layout)

- Consiste en conseguir que el tamaño de la web se ajuste al tamaño horizontal de la pantalla del dispositivo automáticamente y sin necesidad de una barra de desplazamiento horizontal (scroll).
- Este tipo de diseño genera problemas cuando el usuario está visualizando la web en pantallas muy grandes (por ejemplo una TV Full HD de 1.920 x 1080 px) o en pantallas con resoluciones inferiores a 1024 px de ancho.
- Para minimizar estos problemas, se pueden crear diferentes diseños según la pantalla del dispositivo donde se vaya a visualizar la web: pantalla de televisión, ordenador de escritorio, tablet, móvil, etc.
- Estos diferentes diseños lo conseguimos utilizando diferentes versiones de CSS, lo que hace que, a medida que el número de páginas del sitio va en aumento, el número de versiones CSS que hay que mantener va creciendo. Esto implica, que en algunos casos, el mantenimiento de las versiones CSS se vuelva un auténtico suplicio.

Diseño Web Responsivo (Responsive Web Design)

- o Este enfoque busca cubrir todas las resoluciones de pantalla con una única versión de HTML y CSS. Es la idea de “One Web” (Una Web)
- o Esto es posible gracias a las **media queries** que incorpora CSS3.
- o Estas *media queries* son una serie de instrucciones que se incluyen en la hoja de estilos y que indicará cómo debe mostrarse el documento HTML en función de la resolución de la pantalla, de la orientación del dispositivo (horizontal o vertical), la proporción entre ancho y alto de la pantalla, número de colores que es capaz de representar el dispositivo, etc.



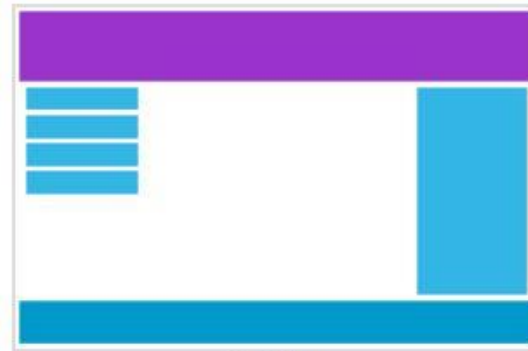


RWD

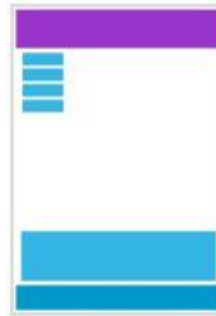
Introducción al Responsive Design

- o RWD = Responsive Web Design
- o Actualmente, los usuarios utilizan diferentes tipos de dispositivos para ver las páginas web: ordenadores de sobremesa, tablets, smartphones, etc.
- o Una página web debería verse correctamente independientemente del dispositivo que el usuario esté utilizando para visualizarla.
- o El Responsive Web Design hace que las páginas web se vean correctamente en todos los dispositivos.
- o El diseño Web Responsivo se consigue utilizando HTML y CSS

- Se llama diseño web sensible cuando se utiliza CSS y HTML para cambiar el tamaño, piel, reducir, ampliar, o mover el contenido para que se vea bien en cualquier pantalla.



Desktop



Tablet



Phone

Viewport

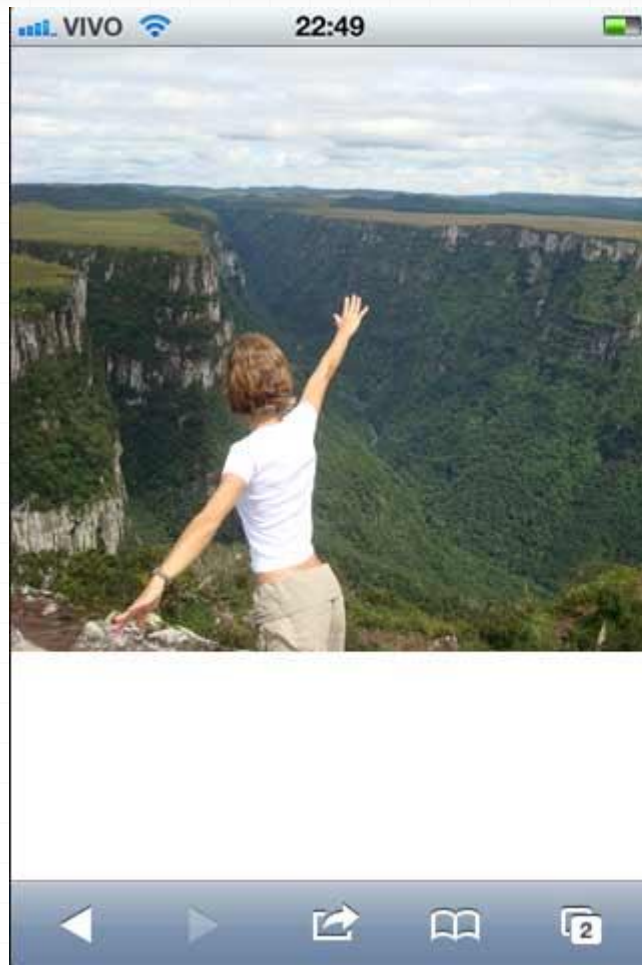
- o El viewport es el área útil de la ventana del navegador.
- o Este área útil varía con el dispositivo y será más pequeño en un smartphone que en una pantalla de ordenador.
- o Debería de incluirse en todas las páginas web la siguiente sentencia meta:

```
<meta name="viewport" content="width=device-width,  
initial-scale=1.0">
```




Etiqueta
viewport

- Esta etiqueta permite configurar cómo debe interpretar el navegador una web para móvil.
- Cuando nos referimos al viewport de un navegador nos estamos refiriendo al área útil de la ventana. Decimos útil porque hay que restar la parte que ocupa la interfaz del navegador (menús, botones, barras de direcciones, barras de desplazamiento, etc.)
- Cuando se visualiza una página web en un dispositivo móvil, habitualmente el área útil se reduce considerablemente.
 - Cuando se diseña una página web multidispositivo, y ésta se visualiza desde un dispositivo móvil, el navegador escalará esa web haciéndola más pequeña para conseguir mostrar toda la página en el espacio disponible de la pantalla del dispositivo.
- Para evitar este empequeñecimiento de las páginas web, el viewport es capaz de emular tener una resolución diferente de la que el dispositivo tiene realmente. Por ejemplo, en un iPhone que tiene realmente 320 píxeles, podemos emular un espacio de 980 píxeles.



- 0 Esta imagen tiene la misma foto que se muestra en la pantalla de un iPhone. Supongamos que la foto mide 320 píxeles de ancho. En la parte de la derecha tendríamos la foto a tamaño real, que es como se vería si tuviéramos un viewport configurado a 320 píxeles de ancho. Pero al verla en un iPhone con un viewport configurado a 980 píxeles de ancho, la imagen se verá bastante más grande.

Sintaxis de viewport

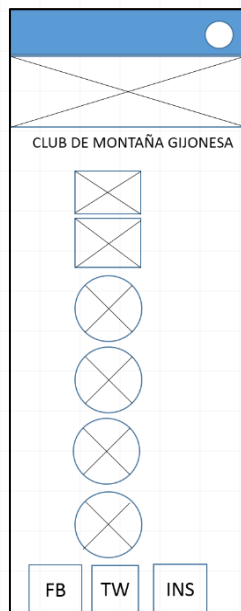
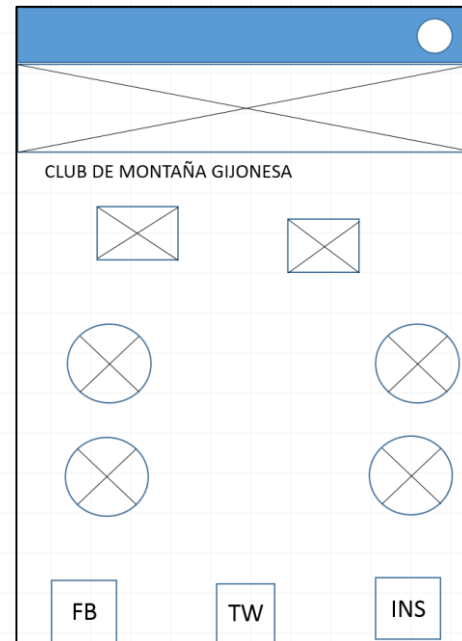
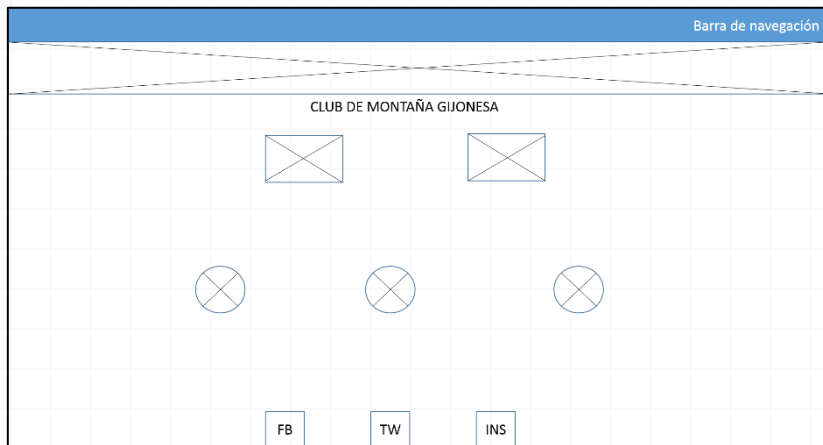
```
<meta name="viewport"  
      content="user-scalable=no,  
      width=device-width,  
      initial-scale=1.0">
```

- `user-scalable=no`. El usuario no podrá hacer zoom en la página. Esto hace que siempre se mantenga la escala como la hayamos definido en el parámetro `initial-scale`, lo que nos facilita el trabajo a la hora de determinar cómo se debe de ver nuestra web pero limita la posibilidad de que el usuario pueda hacer zoom para agrandar o empequeñecer alguna zona.
- `width=device-width`. Los parámetros `width` y `height` permiten especificar la anchura y altura virtual de la pantalla respectivamente.
 - Sin embargo es muy habitual que sólo se especifique la anchura.
 - Además, a la hora de especificar esta anchura es muy típico utilizar el valor `device-width` que quiere decir que iguale el viewport a la anchura real de la pantalla del dispositivo en el que se está visualizando la página.

- 0 `initial-scale=1`. No se hace zoom sobre la página. Es decir, el contenido no se agrandará ni se empequeñecerá.
- 0 Todos los parámetros de viewport:
 - 0 `width`: permite especificar la anchura virtual de la pantalla.
 - 0 `height`: permite especificar la altura virtual de la pantalla.
 - 0 `initial-scale`: permite indicar la escala inicial del documento.
 - 0 `minimun-scale`: escala mínima que se puede poner en el documento.
 - 0 `maximun-scale`: escala máxima que se puede poner en el documento.
 - 0 `user-scalable`: si se permite al usuario hacer zoom o no.



Página web
responsive



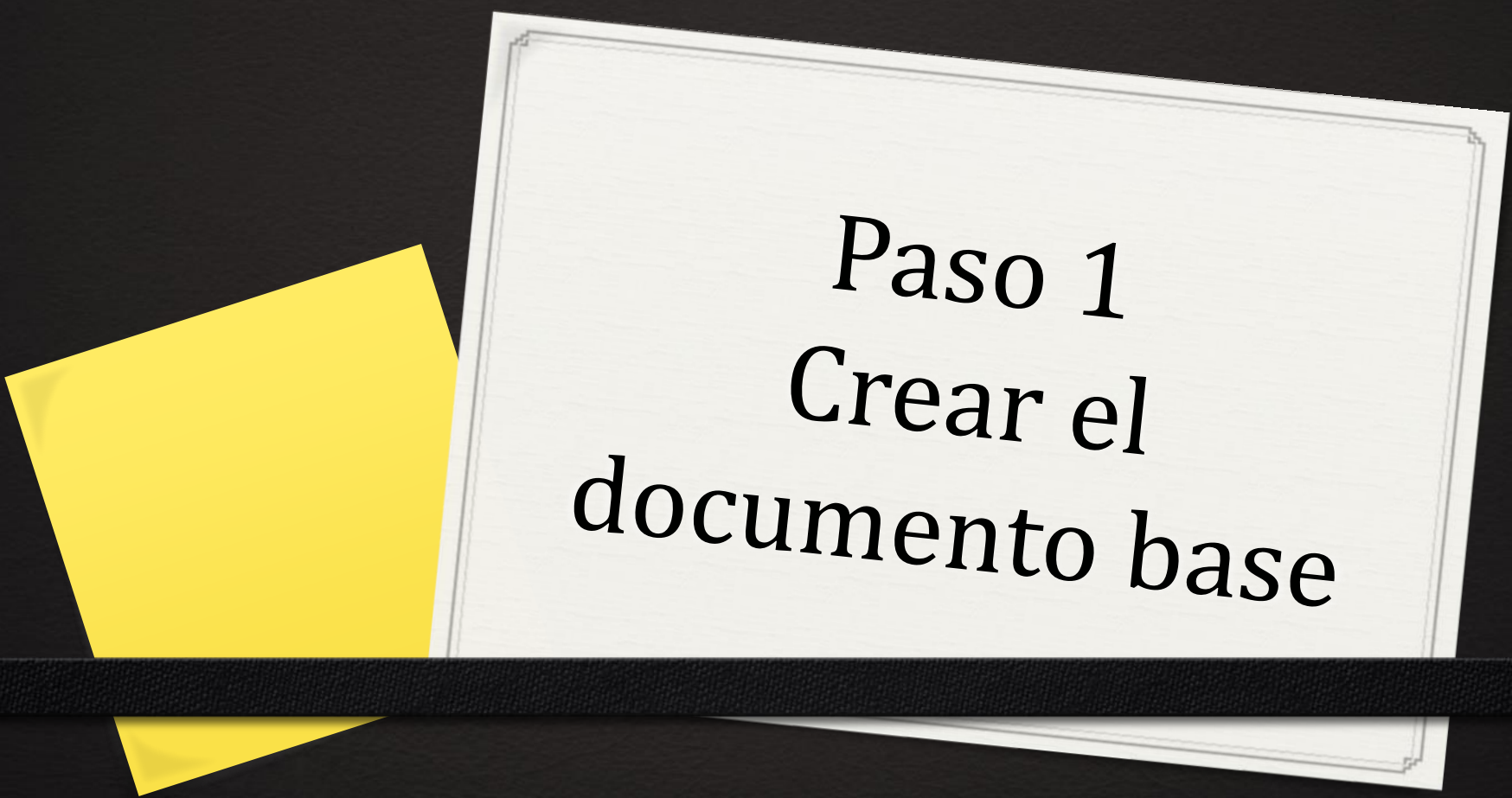
Estructura del sitio

- Para empezar, vamos a crear la estructura de nuestras carpetas.
- Crear una carpeta para contener todos los archivos del sitio
- Dentro de la carpeta anterior, crear una carpeta a la que llamaremos css para los archivos css y otra carpeta a la que llamaremos img para nuestras imágenes.

Obtener algún icono

- o Desde una página que nos permita descargar iconos reunir una serie de iconos.
 - o En nuestro caso vamos a utilizar fontello.com
- o Abrir la página web y buscar un icono de facebook, uno de twitter, uno de instagram y uno de un menu.
- o Descargar
- o Del archivo comprimido que se descarga, copiar en la carpeta de nuestro sitio la carpeta fonts.
- o En el archivo comprimido, localizar la carpeta css y copiar el archivo que se llama fontello.css en la carpeta css que tenemos dentro de nuestro sitio.

Podríamos usar icomoon



Paso 1
Crear el
documento base

- A la hora de plantearnos nuestra página web responsive, podemos tomar dos enfoques:
 - partir de la versión para escritorio e ir bajando a las resoluciones más pequeñas
 - o al revés, empezar creando nuestra versión para móvil e ir subiendo de resolución hasta llegar a la versión visible en pantallas de ordenadores de escritorio.
- En nuestro caso vamos a hacerlo de esta segunda forma.

Crear la estructura html básica para el documento

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>Club de montaña Gijonesa</title>
  <link rel="stylesheet" href="css/estilos.css">
  <link rel="stylesheet" href="css/fontello.css">
  <meta name="viewport" content="width=device-width,
    initial-scale=1">
</head>
<body>

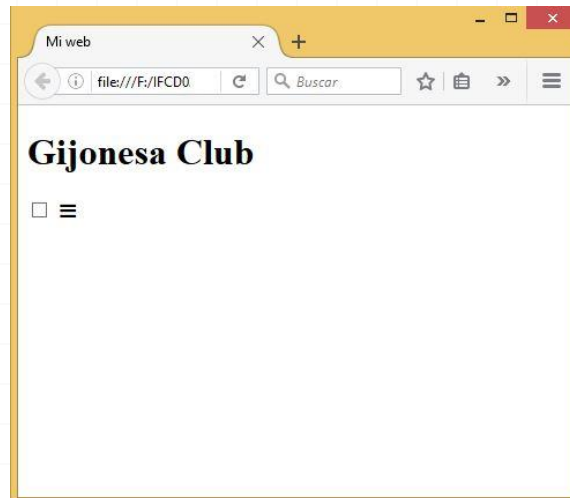
</body>
</html>
```

- o Cuando el usuario visualice la página web desde su tablet o smartphone, en lugar de un menú deberá mostrar un botón y cuando el usuario pinche sobre ese botón entonces se desplegará el menú.
- o Cuando el usuario visualice la página desde un monitor tradicional, se mostrará un menú más o menos tradicional.

Vamos a hacer el botón que desplegará el menú

- Para hacer el botón vamos a utilizar uno de los iconos que nos descargamos en uno de los pasos anteriores. En concreto, vamos a utilizar un botón del tipo icon-menu.
- Para ello, tenemos que poner un enlace al archivo css descargado
`<link rel="stylesheet" href="css/fontello.css">`
- Como queremos un botón que al picar sobre él se active el menú y al volver a picar sobre él se desactive, vamos a hacerlo con un campo de formulario de tipo checkbox.

```
<input type="checkbox" id="menu-bot">  
<label class="icon-menu" for="menu-bot"></label>
```



Definir el menú asociado al botón

- A continuación vamos a definir el menú que se deberá mostrar al usuario cuando éste pique sobre el botón.

```
<div>
```

```
...
```

```
<nav class="menu">
```

```
<a href="">Inicio</a>
```

```
<a href="">Actividades</a>
```

```
<a href="">Contacto</a>
```

```
</nav>
```

```
</div>
```

- En este caso, utilizamos directamente etiquetas <a> pero sería más correcto utilizar y

Primeros estilos

◦ Vamos a quitar el margen y el padding de todos los elementos

```
* {  
    margin:0;  
    padding:0;  
}
```

◦ Vamos a definir la fuente principal de nuestra página. Para ello, vamos a recordar cómo utilizar una fuente que está alojada en un servidor externo como google fonts.

◦ Vamos a google fonts y elegimos la fuente que nos guste.

◦ Hacemos clic sobre el botón uso rápido.

◦ Nos vamos a la pestaña @import y copiamos el import que nos propone google fonts.

```
@import url(https://fonts.googleapis.com/css?family=Titillium+Web);
```

◦ También podemos copiar el código css necesario para aplicarlo a los elementos:

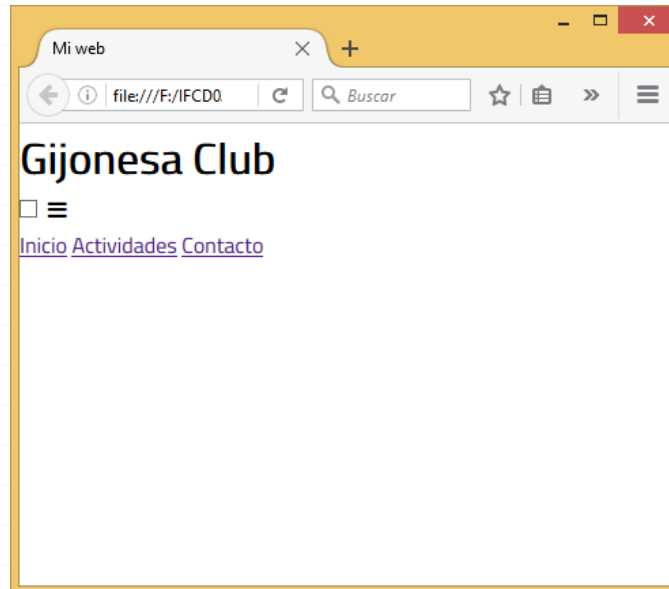
```
font-family: 'Titillium Web', sans-serif;
```



```
@import url(https://fonts.googleapis.com/css?family=Titillium+Web);
```

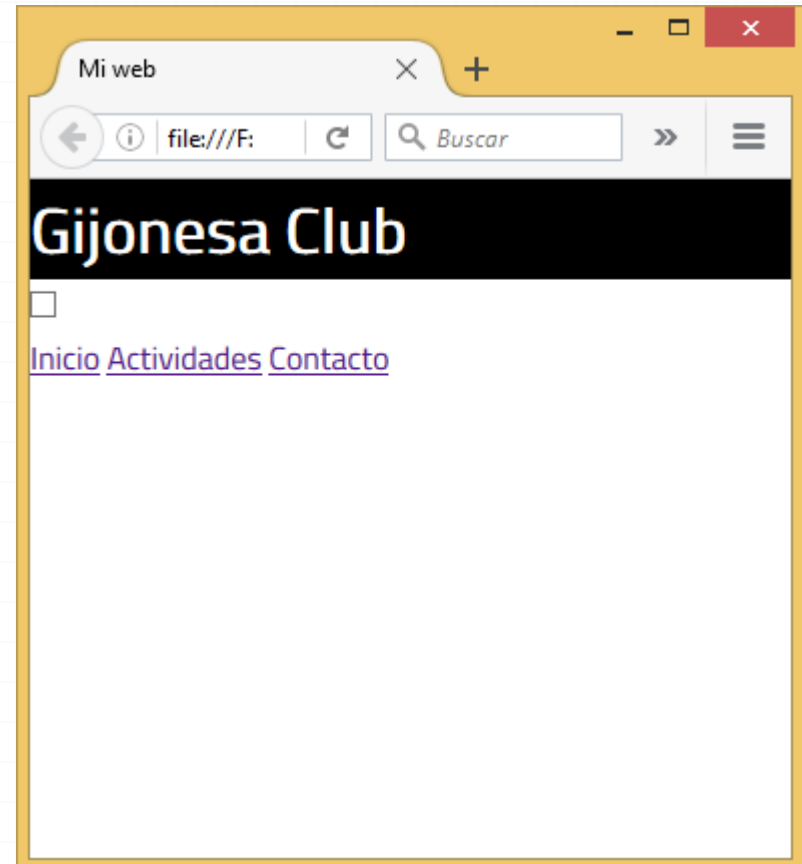
```
* {  
  margin:0;  
  padding:0;  
}
```

```
body {  
  font-family: 'Titillium Web', sans-serif;  
}
```



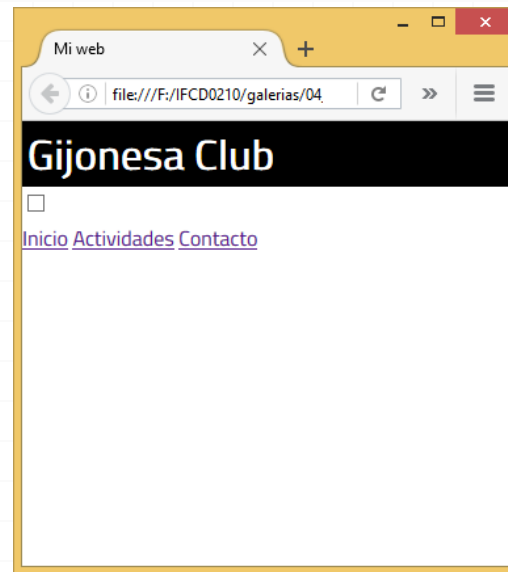
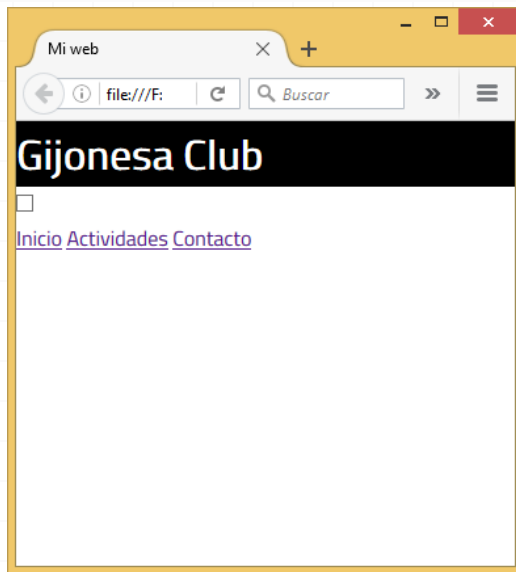
Estilos del header

```
header {  
  width: 100%;  
  height: 50px;  
  background: #000;  
  color: #fff;  
  
  position: fixed;  
}
```



Estilos del contenedor

```
.contenedor {  
  width: 98%;  
  margin: auto;  
}
```



Con el width del 98% y margin auto, centramos el div y hacemos que no ocupe el 100% de la ventana. Con eso conseguimos que el texto no esté pegado justo a la izquierda

Crear los estilos del menu

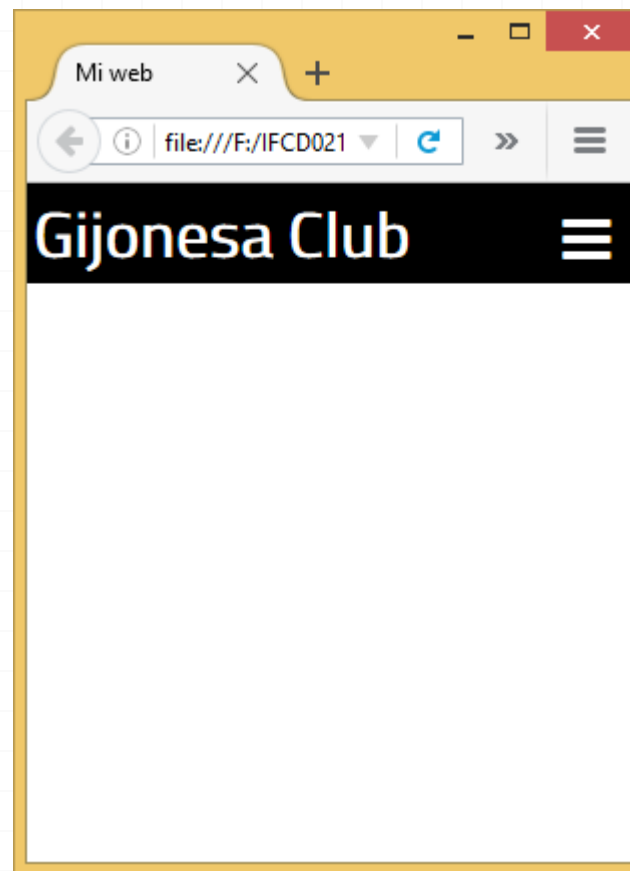
- o Vamos a crear los estilos css para el menú, pero en lugar de ponerlos en la misma hoja de estilos que hemos estado utilizando hasta ahora, vamos a colocarlos en otra hoja de estilos que llamaremos menu.css
- o Una vez creada la hoja de estilos, podemos vincularla con el archivo index.html con la etiqueta <link> igual que hemos hecho con estilos.css y con fontello.css, pero en lugar de hacerlo así vamos a incluirlo con una etiqueta @import dentro de la hoja principal (en nuestro caso estilos.css)

```
@import url(menu.css);
```
- o De esta manera, todos los estilos que definamos en menu.css se importarán en la hoja de estilos principal y en el documento .html sólo tendremos un link hacia dicha hoja de estilos principal.

Configurar el botón

◦ Vamos a dar los siguientes estilos:

```
#menu-bot{  
    display: none;  
}  
  
header label {  
    float: right;  
    font-size: 28px;  
    margin: 6px 0;  
}  
  
.menu {  
    display: none;  
}
```



◦ Además vamos a poner a h1 un float left para que nos quede a la izquierda

Formato del menú

- Vamos a dar formato al menú

```
.menu {  
  
}
```

- Vamos a hacer que el menú salga por debajo del header.

- Para ello vamos a ponerle una posición absoluta y como el header ocupa 50px, vamos a decirle que el menú lo desplace 50px desde arriba.

```
position: absolute;  
top: 50px;
```

- Queremos que cuando se abra el menú, ocupe toda la ventana del dispositivo:

```
width: 100%;  
height: 100vh; /*Para que ocupe todo el alto de la ventana*/
```

- Vamos a poner el color de fondo del menú:

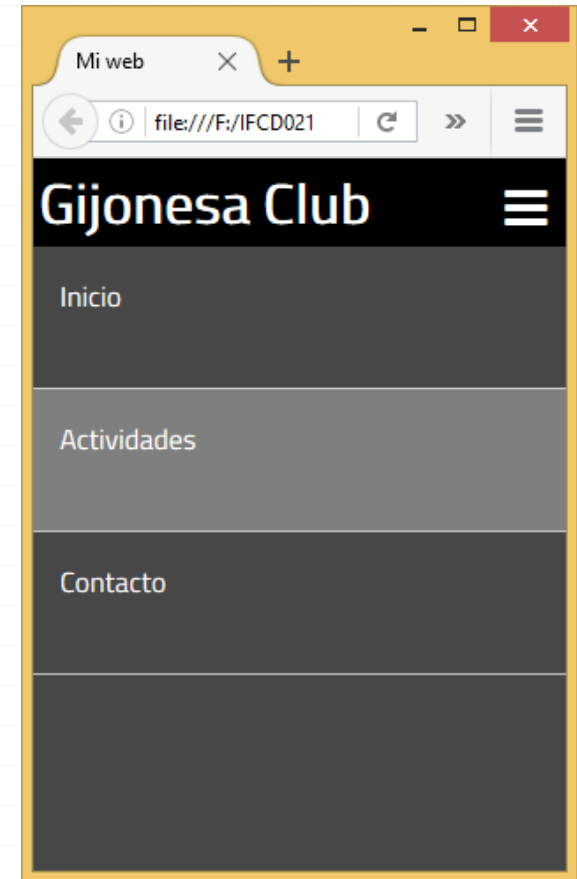
```
background: #666666;
```

```
.menu {  
  position: absolute;  
  top: 50px;  
  width: 100%;  
  height: 100vh;  
  background: #666666;  
}
```



Formato de los enlaces

```
.menu a{  
    display: block;  
    color: #fff;  
    height: 30px;  
    text-decoration: none;  
    padding: 15px;  
    border-bottom: 1px solid #999999;  
}  
  
.menu a:hover {  
    background: #999999;  
}
```



Ocultar el menú

- Lo que queremos conseguir ahora es que cuando el usuario pique sobre el botón el menú aparezca y cuando vuelva a picar sobre el botón desaparezca.
- Para ello, lo primero que vamos a hacer es poner una animación en .menú:

```
transition: all 0.5s;  
transform: translateX(-100%);
```
- Ahora queremos que cuando piquemos sobre el botón, el menú vuelva a su posición original.
 - Para ello vamos a indicar que cuando el campo de tipo checkbox exté activado (#menu-bot:checked) aplique los estilos a .menu
 - Para ello tenemos que utilizar un combinador css.

Combinadores css

- Existen varios caracteres CSS3 que nos permiten combinar los estilos css:
 - Selector de descendiente (espacio)
 - Selector de hijo (>)
 - Selector de hermanos adyacentes (+)
 - Selector general de hermanos (~)

Selector descendiente

- El selector descendiente aplica los estilos a aquellos elementos que son descendientes de un elemento especificado.

```
div p {  
    background-color: yellow;  
}  
...  
<div>  
    <p>Párrafo 1</p>  
    <p>Párrafo 2</p>  
    <span><p>Párrafo 3</p></span>  
</div>
```

```
<p>Párrafo 4</p>  
<p>Párrafo 5</p>
```

Párrafo 1

Párrafo 2

Párrafo 3

Párrafo 4

Párrafo 5

Selector de hijo

- Aplica los estilos a todos los elementos que son hijos directos del elemento especificado.

```
div > p {  
    background-color: yellow;  
}  
...  
<div>  
    <p>Párrafo 1</p>  
    <p>Párrafo 2</p>  
    <span><p>Párrafo 3</p></span>  
</div>  
  
<p>Párrafo 4</p>  
<p>Párrafo 5</p>
```

Párrafo 1

Párrafo 2

Párrafo 3

Párrafo 4

Párrafo 5

Selector general de hermanos

- El selector general de hermanos selecciona todos los elementos que son hermanos de un elemento especificado.

```
div ~ p {  
    background-color: yellow;  
}
```

...

```
<div>  
    <p>Párrafo 1</p>  
    <p>Párrafo 2</p>  
</div>
```

```
<p>Párrafo 3</p>  
<p>Párrafo 4</p>
```

Párrafo 1

Párrafo 2

Párrafo 3

Párrafo 4

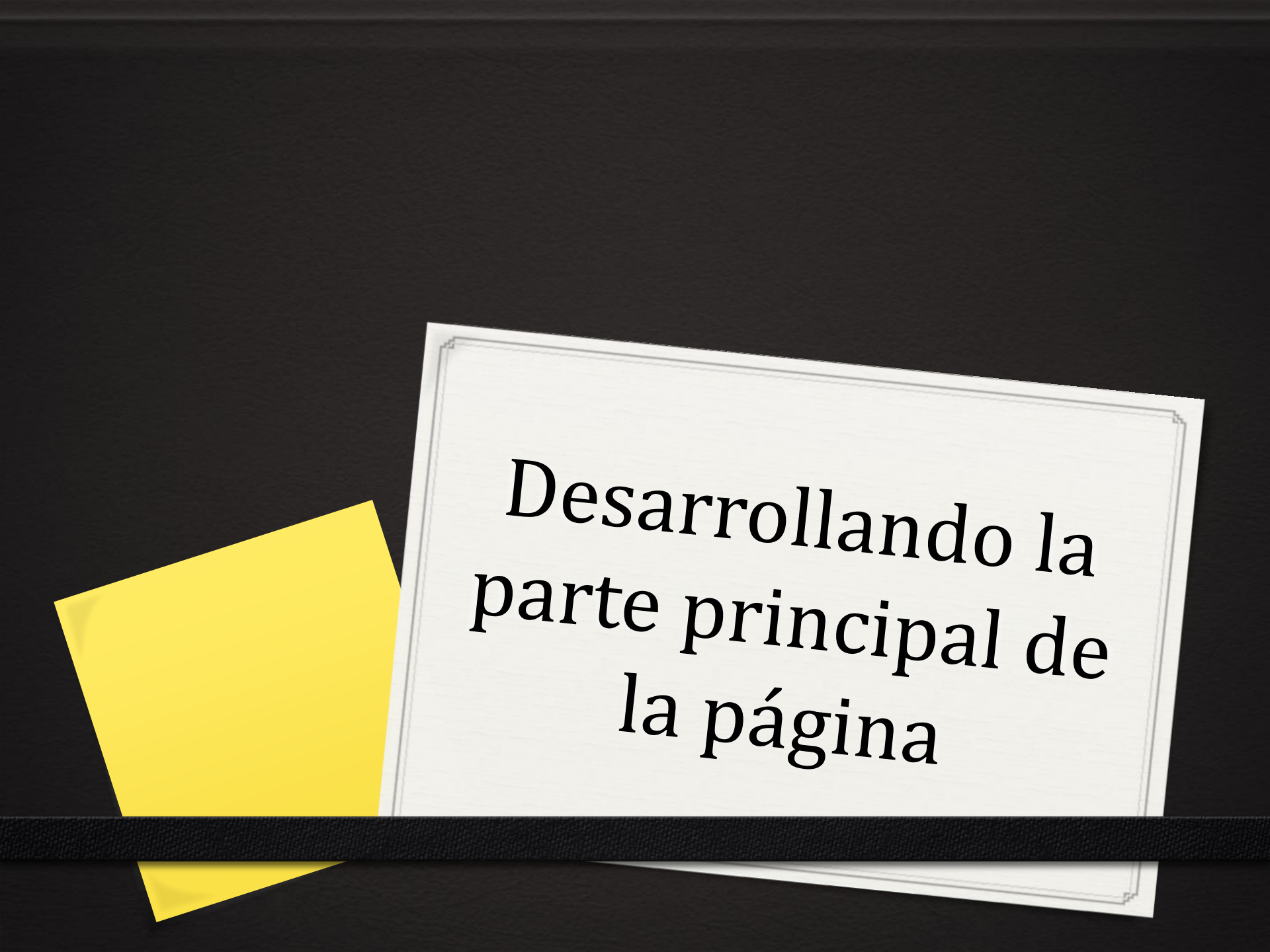
~ = alt+126

- En nuestro caso el combinador que tenemos que usar será el selector general de hermanos:

```
#menu-bot:checked ~ .menu {  
    transform: translateX(0%);  
}
```

Cuando el elemento menu-bot que es de tipo checkbox esté activado aplicar una transformación sobre el elemento menu

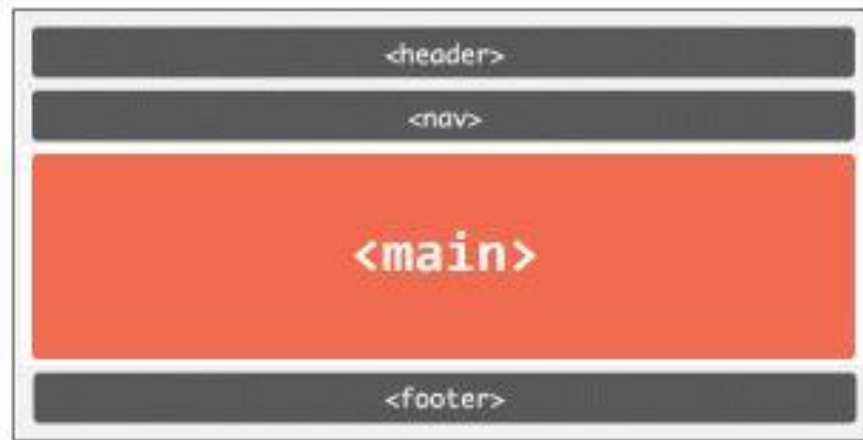
- Con esto ya tenemos nuestro menú.
- Como es responsive, al agrandar la ventana, el menú se irá adaptando a la ventana.



Desarrollando la
parte principal de
la página

Etiqueta <main>

- Hace poco, el elemento main fue incluido formalmente en la especificación de HTML5 de la W3C.
- Si bien ya teníamos elementos como header, section, article o footer, no existía una etiqueta precisa que describiese el contenido principal de una página.
- Hasta ahora, generalmente se utilizaba una etiqueta div para englobar el contenido primario de un documento, asignándole al mismo un id de valor “main” (que significa principal en inglés).
- Ahora el id puede seguir siendo el mismo, pero la etiqueta debería cambiar a main si queremos maquetar correctamente.
- El propósito principal de esta etiqueta es desde un punto de vista de accesibilidad, ya que ayuda a que los screen readers (lectores de pantalla) y otras tecnologías asistenciales puedan identificar donde comienza el contenido principal de la página y donde termina.
- Según la especificación: el elemento main representa el contenido principal del cuerpo (body) de un documento.



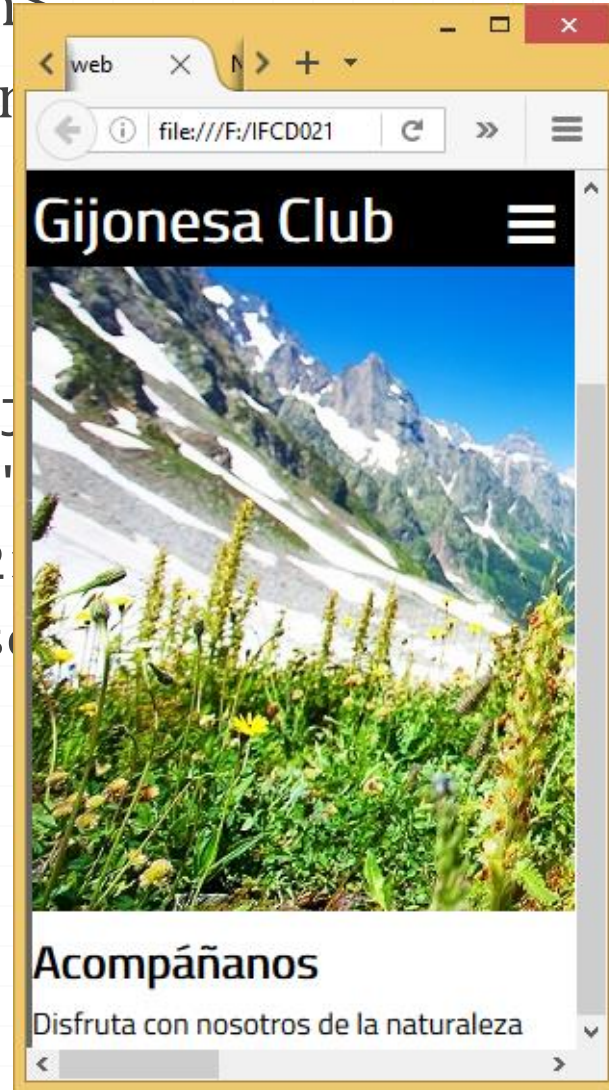
- La etiqueta main se puede utilizar una sola vez por página.
 - Usar más de un elemento main, hará que nuestro HTML sea inválido para la W3C.
- `<main>` no puede ser hijo de ninguno de los siguientes: header, nav, article, aside y footer.
- Algunos navegadores todavía no reconocen a main ni tampoco tienen estilos por defecto para el mismo
 - Esto se puede solucionar incluyendo el archivo HTML5 shiv en el `<head>` de nuestro proyecto

```
<!--[if lt IE 9]>
<script
  src="http://html5shim.googlecode.com/svn/trunk/html5.js">
</script>
<![endif]-->
```

Crear el contenido principal

- Vamos a definir la parte principal de nuestra página. Para ello, vamos a incluir la etiqueta `<main>`
- Dentro del main vamos a crear un `<section>` para el banner. Lo vamos a poner en un `<div>`.

```
<main>  
  <section id="banner">  
      
    <div class="contenedor">  
      <h2>Acompáñanos</h2>  
      <p>Disfruta con nosotros de la naturaleza</p>  
    </div>  
  </section>  
</main>
```



Dar estilos al banner

- o Como vamos a utilizar varios <section>, en el archivo de estilos principal vamos a darle un estilo a los section con un ancho del 100% y un margen inferior de 25px para que cuando haya contenido en la parte inferior nos lo separe un poco

```
section{  
    width: 100%;  
    margin-bottom: 10px;  
}
```

- o Vamos a crear un nuevo archivo de estilos que se va a llamar banner.css y lo vamos a importar en el archivo de estilos principal.

- Al contenedor que está dentro de banner vamos a posicionarlo absolutamente y a banner vamos a darle un position relative

```
#banner {  
    margin-top:50px;  
    position: relative;  
}  
#banner .contenedor {  
    position:absolute;  
    top: 35%;  
    left: 2%;  
}
```

- Al ponerles posicionamiento a los dos, esto hace que #banner .contenedor se posicione respecto del contenedor que lo contiene y está posicionado (#banner)

- Ahora vamos a colocar el texto en el centro de la imagen, para eso vamos a darle un top del 50% y un left del 0%.

top: 50%;

left: 2%;

- Por último, vamos a poner el texto en blanco.

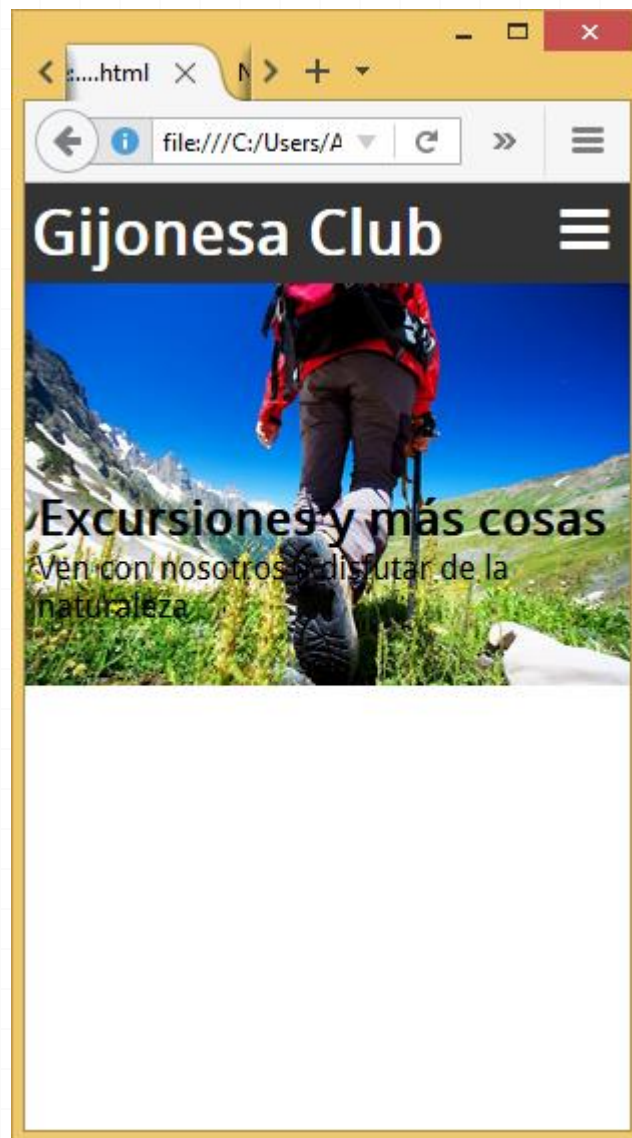
color: #ffffff;

- Al texto que está dentro del h2 de banner, vamos a darle un tamaño de 25px

```
#banner h2 {
```

```
font-size: 25px;
```

```
}
```



- Vamos a añadir otro section que contenga un texto de bienvenida a la página web

```
<section id="bienvenida">  
  <h2>Bienvenidos</h2>  
  <p>Lorem ipsum... </p>  
</section>
```

- A esta section le vamos a dar un estilo que nos centre el texto. Como es un estilo muy sencillo y breve, lo defino en la hoja de estilos general: estilos.css

```
#bienvenida {  
  text-align: center;  
}
```


- Vamos a añadir otro section con un id=blog
- Dentro de esta section vamos a poner tres articles

```
<section id="blog">
  <h3>Nuestro blog</h3>
  <div class="contenedor">
    <article>
      
      <h4>Rutas de montaña </h4>
    </article>
    <article>
      
      <h4>Paseos por la playa </h4>
    </article>
    <article>
      
      <h4>Ruta hacia Bueño</h4>
    </article>
  </div>
</section>
```

○ Ahora vamos a darle estilos al blog. Para eso creamos un nuevo archivo de estilos css al que podemos llamar blog.css e importamos este archivo en la hoja de estilos general.

- Al contenedor que se encuentra dentro de la section con id=blog, vamos a darle un display flex para que las imágenes se ubiquen todas en una línea y vamos a centrar su contenido.
- Además vamos a darle un flex-wrap:wrap; para que los elementos que no entren en una línea se pasen a la siguiente.

```
#blog .contenedor{  
    display: flex;  
    justify-content: center;  
    flex-wrap: wrap;  
}
```

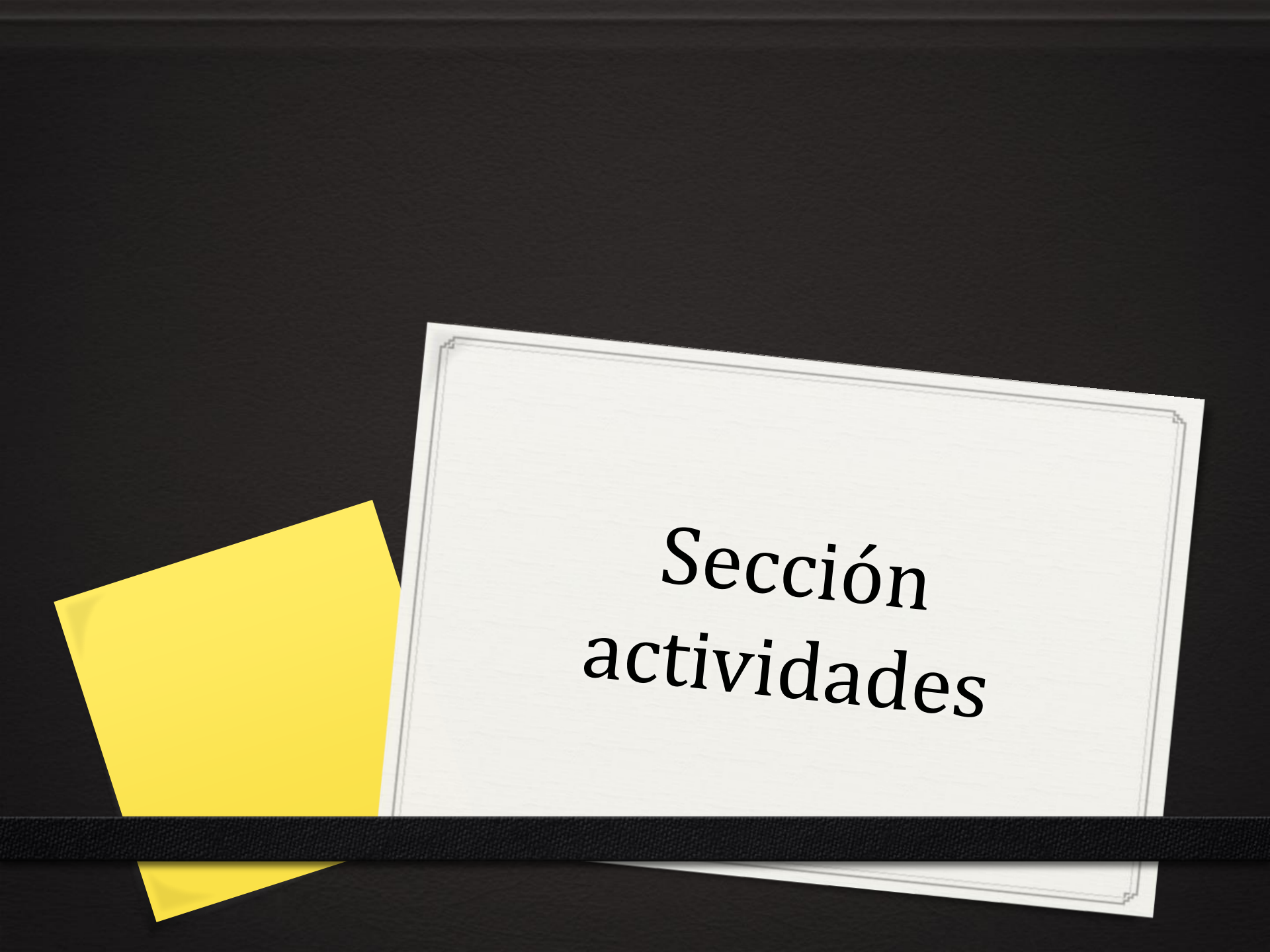
- A los artículos vamos a darles un margin de 15px; para que se separen.

```
#blog article{  
    margin: 15px;  
}
```

- A las imágenes que están dentro del blog voy a darles un width del 100% y un ancho máximo de 280px.

```
#blog img {  
    width: 100%;  
    max-width: 280px;  
}
```

- al h3 y al h4 vamos a alinearlos en el centro.



Sección actividades

Index.html

- Creamos una nueva section dentro del documento a la que vamos a dar el id = “actividades”.
- Luego dentro de esta section crearemos un div con class=“contenedor”.
- Dentro de este div contenedor vamos a poner un div para cada imagen (en este caso 4). A este div le vamos a dar la clase fotos y además, debajo de la imagen vamos a poner un rótulo con el nombre de cada una de las actividades (este rótulo lo pondremos con una etiqueta h4).

```
<section id="actividades">
  <h2>Actividades</h2>
  <div class="contenedor">
    <div class="fotos">
      
      <h3>Trekking</h3>
    </div>
    <div class="fotos">
      
      <h3>Rutas en btt </h3>
    </div>
    <div class="fotos">
      
      <h3>Escalada</h3>
    </div>
    <div class="fotos">
      
      <h3>Rutas a caballo</h3>
    </div>
  </div>
</section>
```

Actividades



Trekking



Estilos css

- Crearemos un archivo css al que llamaremos actividades y que luego importaremos en nuestro archivo de estilos general:

```
#actividades {  
    background: #666666;  
    color: #ffffff;  
    text-align: center;  
    padding: 20px;  
}  
#actividades .contenedor {  
    display: flex;  
    flex-wrap: wrap;  
    justify-content: center;  
}  
.fotos {  
    margin: 20px;  
}  
.fotos img {  
    width: 180px;  
    height: 180px;  
    border-radius: 50%;  
    border: 7px solid #ffffff;  
}
```

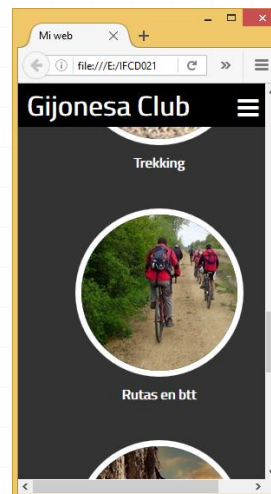
Pantalla grande




Tablet



Móvil





Configurar el
footer

- Vamos a configurar el pie de nuestra página:
 - Pondremos una línea con el mensaje de copyright
 - Pondremos otra línea con el enlace a las redes sociales.
 - A los enlaces de las redes sociales les asignaremos la clase con el nombre del botón que nos bajamos de fontello

```
<main>
```

```
...
```

```
</main>
```

```
<footer>
```

```
  <div class="contenedor">
```

```
    <p class="copy">&copy; Gijonesa Club 2016</p>
```

```
    <div class="sociales">
```

```
      <a class="icon-facebook" href="#"></a>
```

```
      <a class="icon-twitter" href="#"></a>
```

```
      <a class="icon-instagram" href="#"></a>
```

```
    </div>
```

```
  </div>
```

```
</footer>
```

< web



www.ejemplo.com



Gijonesa Club



Paseos por la playa



Ruta hacia Bueño

© Gijonesa Club 2016




- Crear una hoja de estilos a la que llamaremos pie.css y en la que definiremos los estilos para nuestro footer.

```
footer .contenedor {  
    display: flex;  
    flex-wrap: wrap;  
    justify-content: center;  
    padding-bottom: 25px;  
}  
.copy{  
    font-size: 20px;  
}  
.sociales {  
    width: 100%;  
    text-align: center;  
    font-size: 28px;  
}  
.sociales a {  
    color: #333;  
    text-decoration: none;  
}
```



- Importar la hoja de estilos anterior, en nuestra hoja de estilos.css



Media queries

- Vamos a empezar completando el diseño reponsive de la web, empezando por el menú.
- La forma de mostrar el menú que hemos diseñado es válida para móvil y tablet pero no para pantalla.
- Abrimos la hoja de estilos menu.css y vamos a añadir la configuración para nuestra pantallas

```
@media (min-width:1024px) {
```

```
  .menu{  
    position: static;  
    width: auto;  
    height: 30px;  
    transform: translateX(0%);  
    float: right;  
    display: flex;  
  }
```

En el menú de móvil el menú estaba desplazado -100%, así que en este lo quitamos. Le ponemos float right para que se ponga a la derecha y display flex para que se pongan unos al lado de los otros

```
  .menu a {  
    border: none;  
    height: 20px;  
  }
```

Quitamos el borde a los enlaces

```
  header label {  
    display:none;  
  }
```

Quitamos el botón que nos abre el menú

```
}
```

- Si queremos por ejemplo que en las pantallas, el logotipo y el menú no quede tan pegado al borde podemos cambiar el tamaño del contenedor que en la hoja de estilos.css hemos definido con un ancho del 100% y ponerle menos ancho.

```
@media (min-width:1024px) {
```

```
  .contenedor{
```

```
    width: 90%;
```

```
}
```

También se podría especificar en píxeles:
Width: 1000px;

- Respecto al banner, el texto que aparece a la izquierda que le habíamos puesto un desplazamiento del 40%, en pantallas pequeñas se ve bien, pero en las grandes se ve mal porque queda justo donde está lo blanco de la imagen.

```
@media (min-width:1024px) {  
  #banner .contenedor {  
    top: 50%;  
  }  
}
```



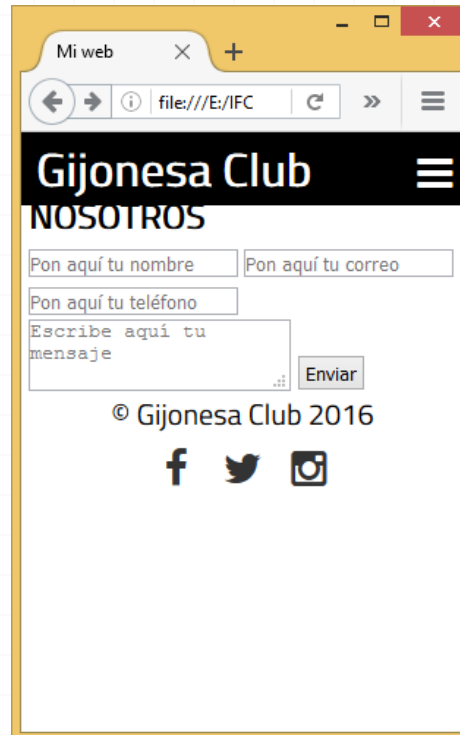

Formulario
responsive

- Vamos a crear la parte de contacto con un pequeño formulario de contacto
- Para ello, copiamos el fichero index.html y al fichero copia le cambiamos el nombre y le asignamos contacto.html.
- En el archivo contacto.html, borramos todo lo que está dentro de <main> y ponemos el código html que nos permita generar el formulario de contacto.

```
<main id="main_form">
  <form action="" method="post">
    <h2>PONTE EN CONTACTO CON NOSOTROS</h2>
    <input type="text" name="nombre"
      placeholder="Pon aquí tu nombre" required>
    <input type="email" name="correo"
      placeholder="Pon aquí tu correo" required>
    <input type="text" name="telefono"
      placeholder="Pon aquí tu teléfono" required>
    <textarea name="mensaje"
      placeholder="Escribe aquí tu mensaje"
      required>
    </textarea>

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

© Gijonesa Club 2016



Estilos para el formulario

- Vamos a aplicar los primeros estilos para el formulario.
- Vamos a empezar por el contenedor main del formulario al que le hemos dado el id `main_form`:
 - Lo primero que vamos a hacer es ponerle un margen superior de 50px, para que no lo pise la cabecera. Ya que la cabecera tiene un tamaño de 50px, le ponemos ese margen para que empiece justo debajo.

`margin-top: 50px;`

- A continuación vamos a ponerle una imagen de fondo que se llama `fondo1.jpg` y que está almacenada en la carpeta `img`.

`background-image: url(../img/fondo1.jpg);`

- Luego vamos a darle el tamaño a esa imagen. Para eso vamos a utilizar `background-size` pero en lugar de dar porcentajes o píxeles le vamos a indicar que tome el ancho y el alto de la ventana

`background-size: 100vw 100vh;`

- Queremos que cuando la ventana no sea lo suficiente para contener todos los campos de formulario y el usuario tenga que desplazar la ventana que la imagen permanezca fija, que no se mueva por lo que le pondremos

`background-attachment: fixed;`

- Por último, vamos a ponerle un padding de 0 por si arrastra el padding de algún otro estilo del sitio



- En vez de al main, podríamos poner el identificado en el body del fichero estilos y al aplicar los estilos se consigue un efecto algo diferente.

Estilos de nuestro formulario

```
form {  
  width: 40%;  
  margin: auto;  
  background: rgba(0,0,0,0.4);  
  padding: 10px 20px;  
  border-radius: 10px;  
}
```

```
form h2 {  
  color: #fff;  
  text-align: center;  
  margin: 5px 0px 20px;  
  font-size: 20px;  
}
```