

Hardware Planet

Daniel Ramos Rodríguez

Desarrollo de Aplicaciones Web

Universidad de Santiago de Compostela

Índice

Índice	2
Introducción	3
Diseño	4
Inventario de contenido	4
Diagrama de despliegue	5
Prototipo	6
Diagrama de casos de uso	10
HTML	11
Página principal	11
Plataformas	13
Arquitecturas	15
Contribuir	17
CSS	19
Página principal	19
Plataformas	21
Arquitectura	22
Contribuir	22
JavaScript	24
Modo noche	24
Barra de navegación colapsable	26
Previsualización de enlaces	28
JQuery	31
Menús laterales ocultables	31
ES6	33
Aviso de cookies	33
Expresiones regulares	35
Validación de elementos en formularios	35
AJAX	37
Carga de datos de plataformas leídos en XML	37
Carga de datos de arquitecturas leídos en JSON	39

Introducción

Se propone como proyecto para la creación de un sitio web un lugar de referencia sobre hardware tanto para aficionados como para programadores que busquen información a fin de desarrollar emuladores.

En lo relativo al contenido, la página contará con información sobre consolas, microordenadores de los años 80 y 90, ISAs (arquitecturas computacionales) así como particularidades de las mismas, especificaciones técnicas, señales de audio, vídeo, datos históricos para entender el contexto tecnológico.

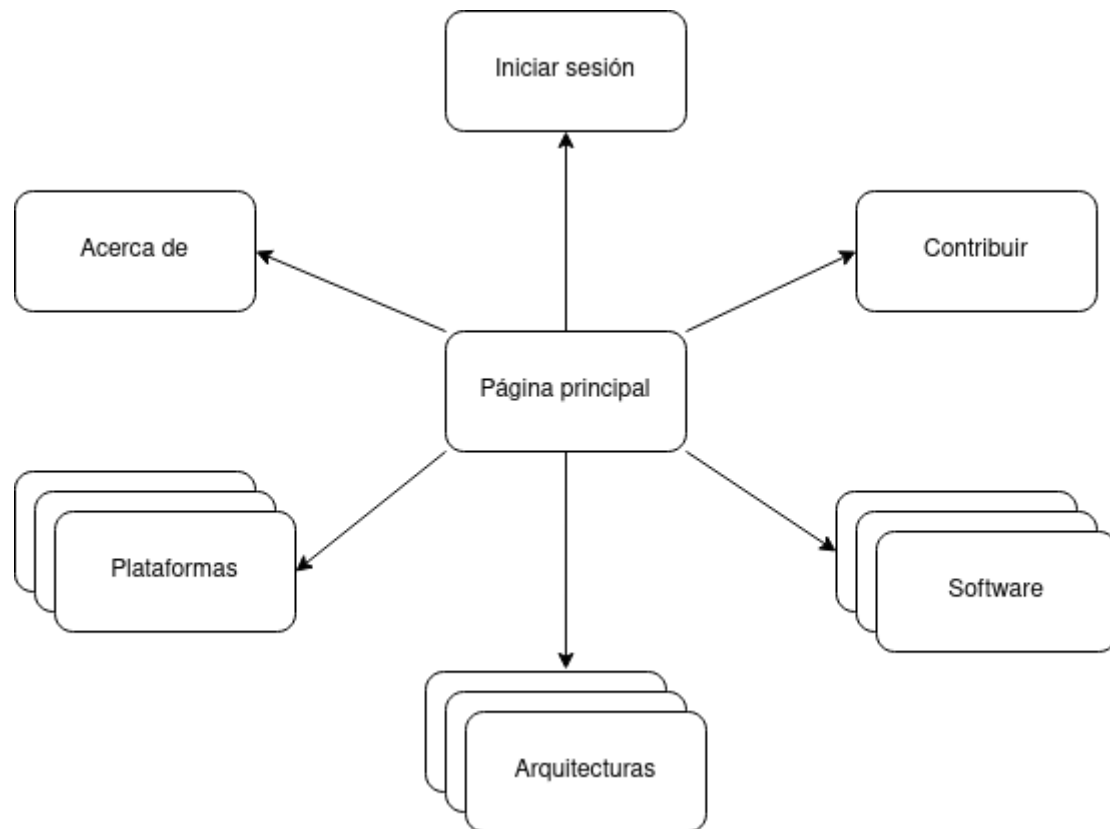
Además, se desea contar con elementos visuales, como imágenes de las plataformas y sus componentes que clarifiquen las explicaciones o esquemas lógicos acerca de su funcionamiento.

Por otra parte, se incluirán datos acerca de software, pero solo aquellos relacionados directamente con la emulación, la simulación o la preservación de hardware, así como sobre los formatos en que se codifican los datos extraídos de los medios originales (cartuchos, CDs, cassettes...).

Los visitantes podrán contribuir de dos modos diferentes. El primero es crear una cuenta cuyas ediciones estarán supervisadas durante el primer mes y deberán ser aprobadas previamente por los moderadores, y el segundo es enviar un mensaje a la administración del sitio web para que se encargue personalmente de subsanar errores o incluir nueva información. Será importante en ambos casos proporcionar fuentes.

Diseño

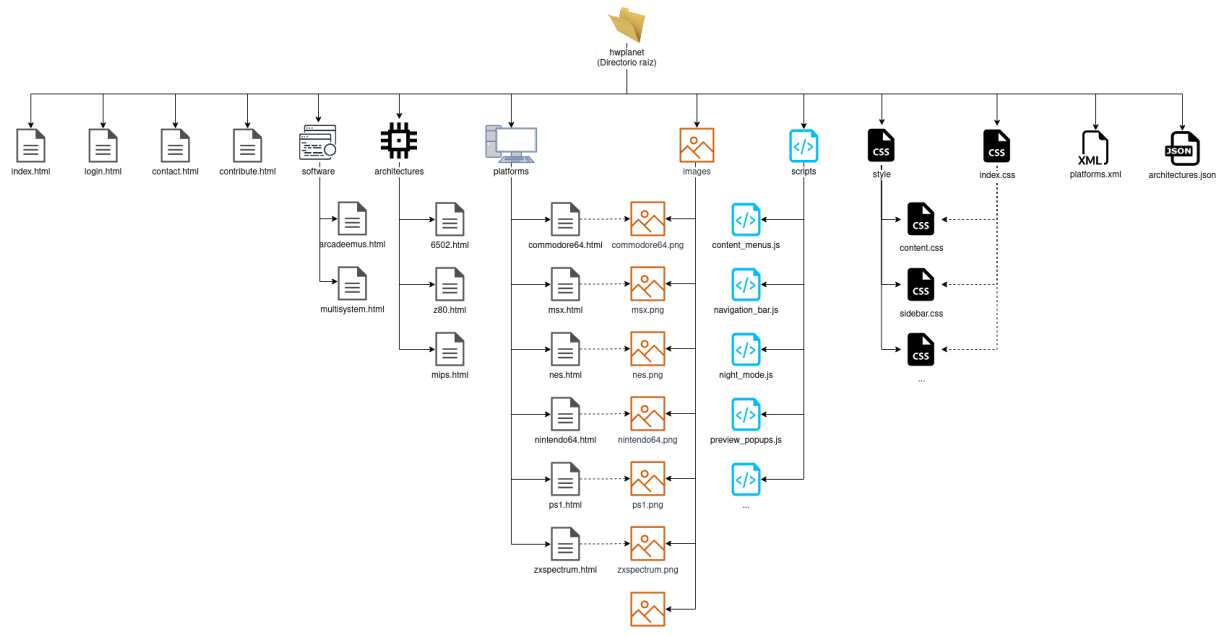
Inventario de contenido



En el diagrama podemos apreciar el inventario de contenido del proyecto, que será empleado como base a la hora de definir la estructura de directorios del sitio web, facilitando la organización de la información y otros recursos.

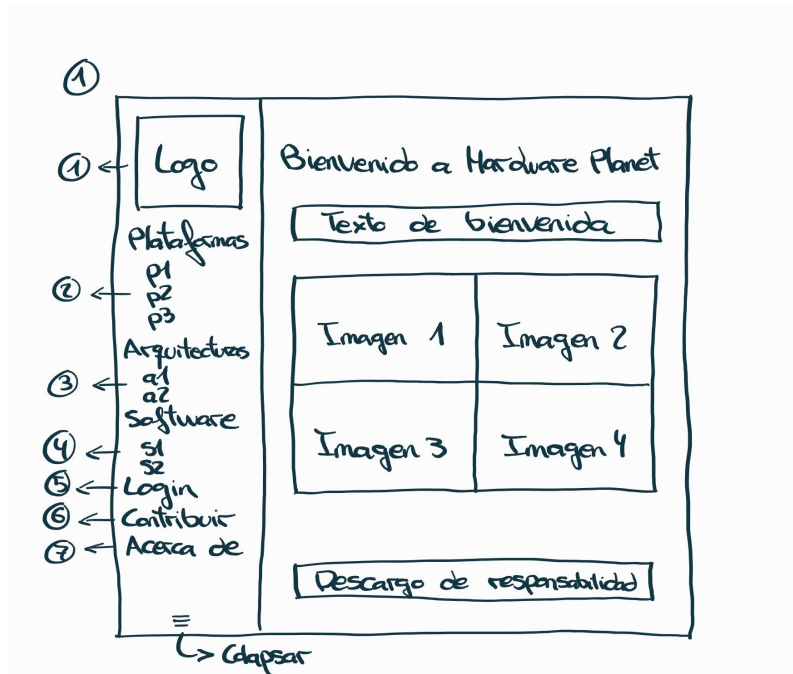
Como el énfasis del proyecto está puesto en su carácter enciclopédico, resulta preferible contar con un menor número de apartados y desarrollar en mayor profundidad aquellos que estén presentes, redactando más artículos y ampliando la información que ofrecen.

Diagrama de despliegue

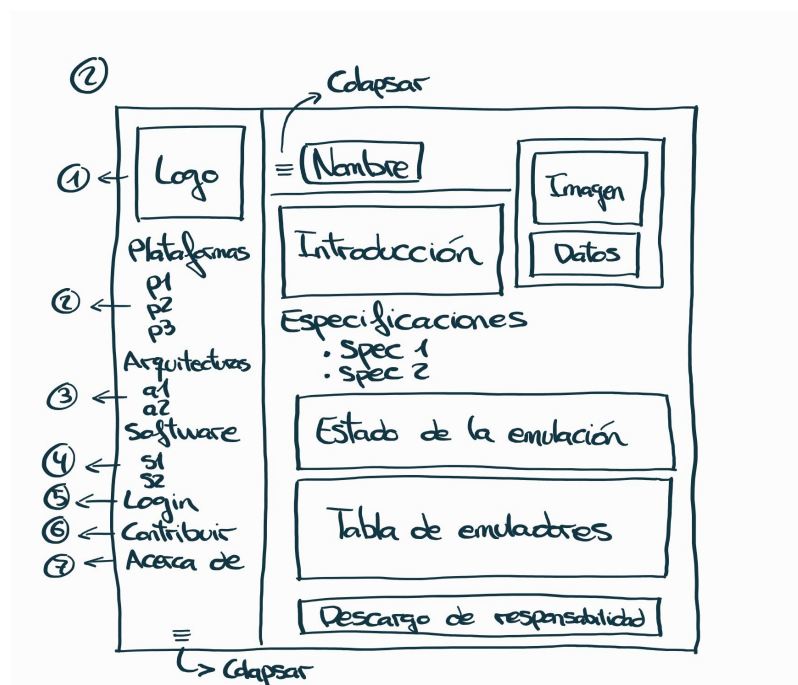


En la imagen se muestran los ficheros y directorios que conforman el sitio web, con flechas continuas que indican las relaciones de pertenencia y discontinuas que muestran las dependencias entre los archivos. Por ejemplo, los ficheros HTML de la sección de plataformas han de mostrar imágenes, contenidas en un apartado distinto. Además, el fichero index.css se encarga de incluir todos los demás, ubicados en el directorio “styles”.

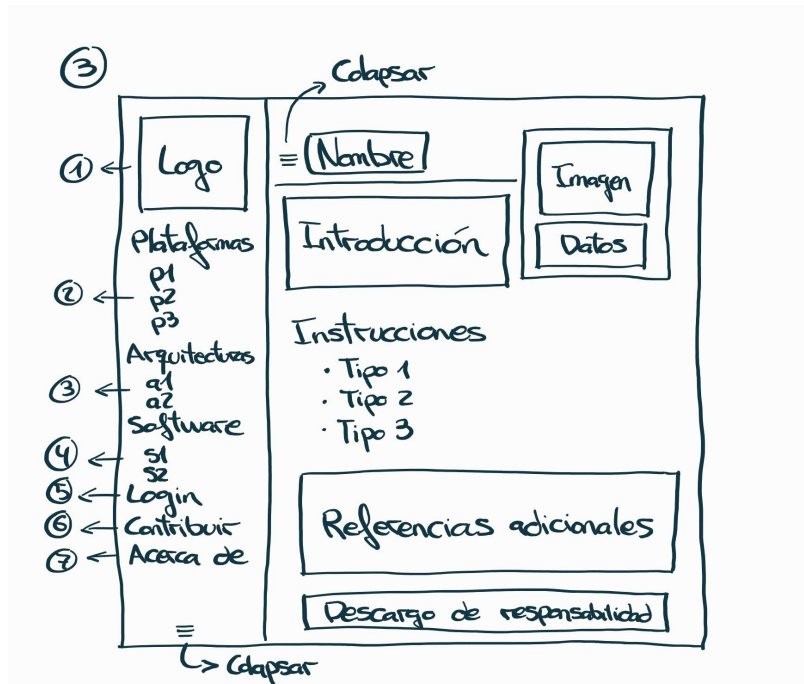
Prototipo



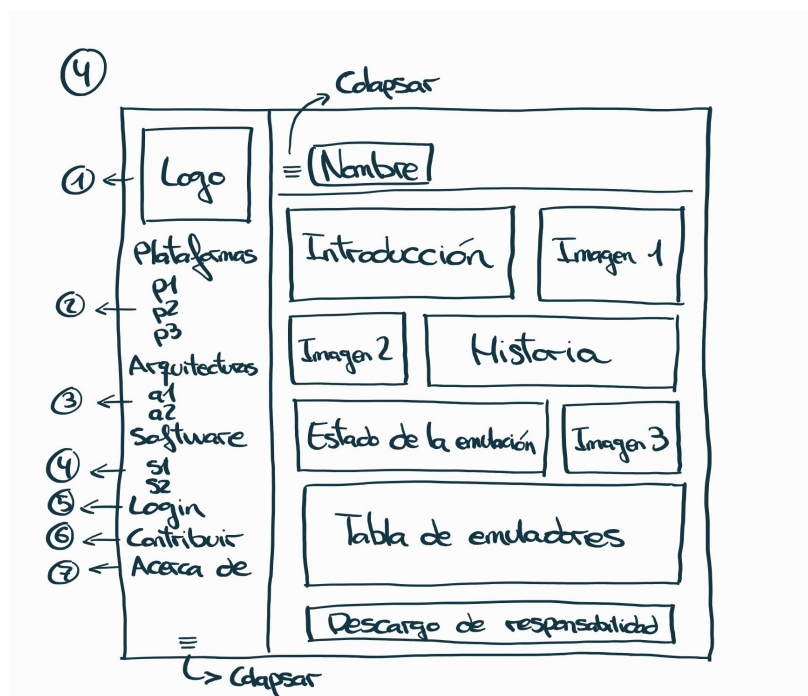
Página principal



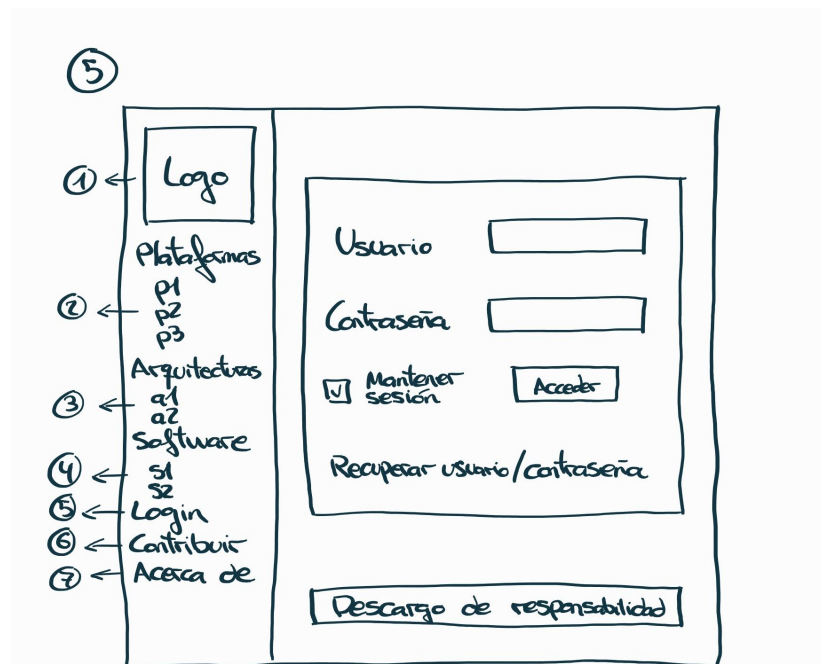
Plataformas



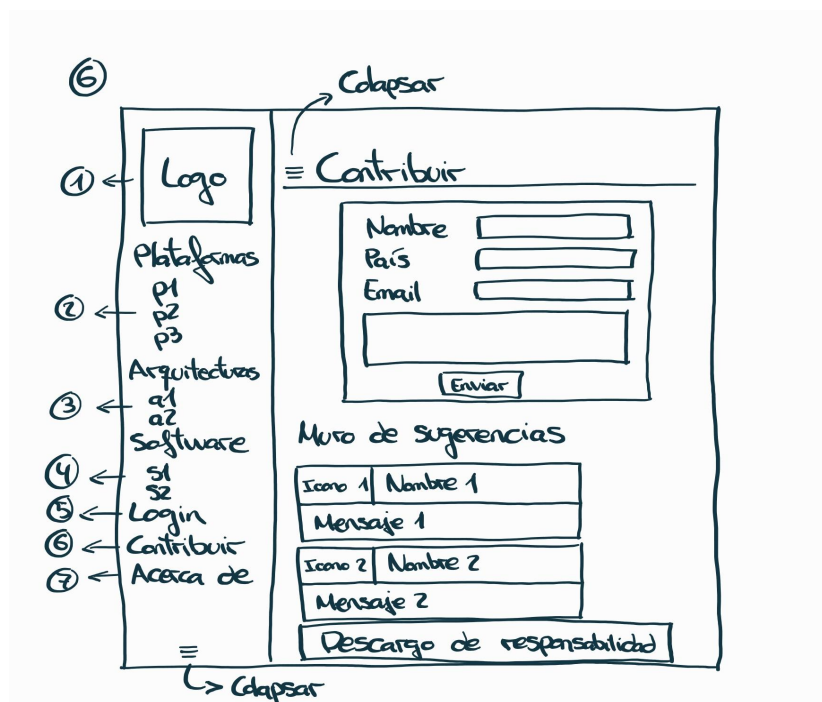
Arquitecturas



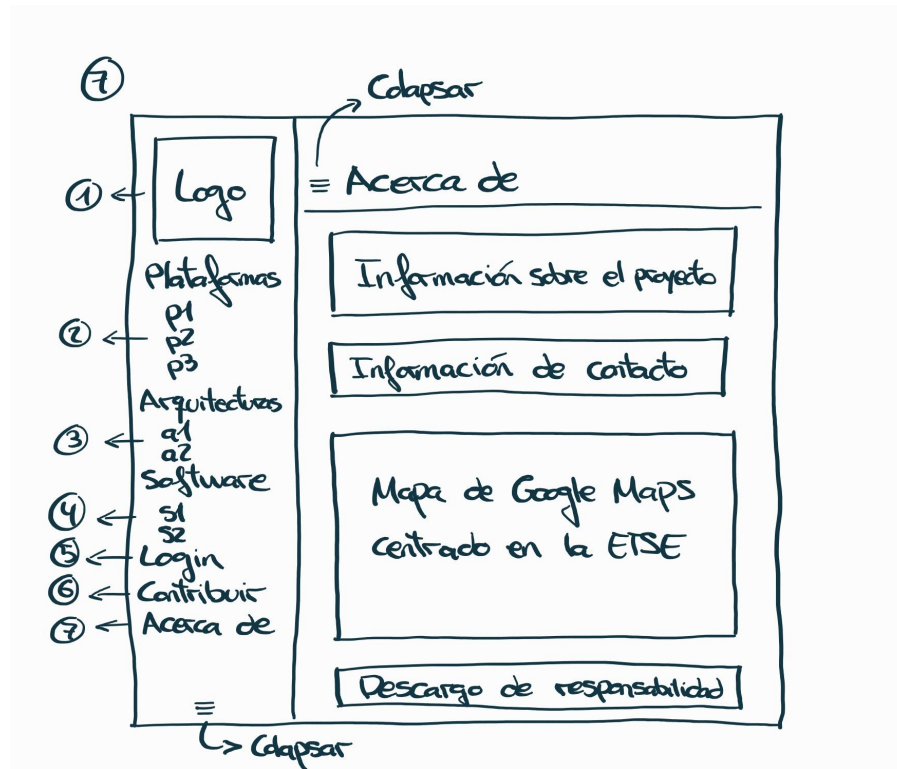
Software



Login

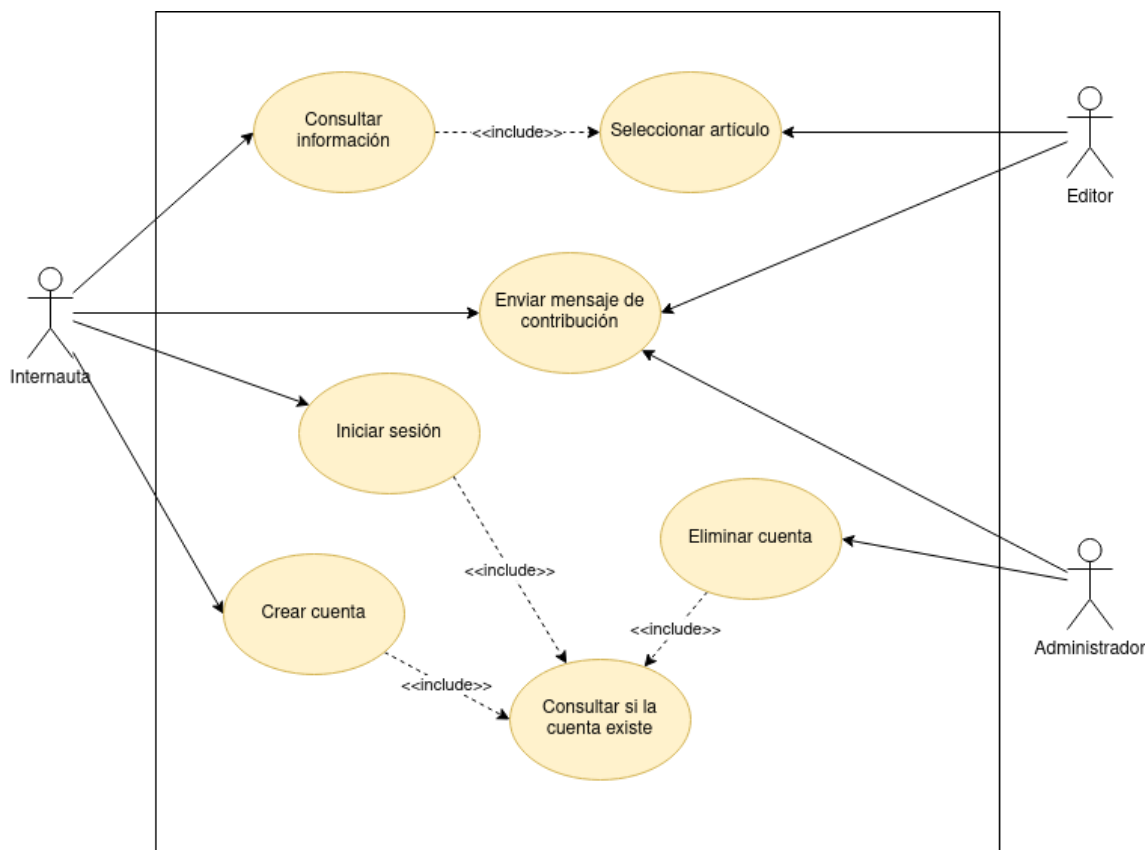


Contribuir



Acerca de

Diagrama de casos de uso



El diagrama de casos de uso nos permite representar las formas más comunes de emplear nuestro sitio web, lo que posibilita a su vez establecer perfiles de usuario personalizados que garanticen una mejor experiencia a los mismos, en función de sus necesidades y requisitos.

En este caso, los “clientes” serán los internautas que accedan a la web para consultar información, pudiendo crear una cuenta o iniciar sesión si ya lo han hecho. En ambos casos será necesario comprobar si la cuenta existe. Por otra parte, también podrán escribir mensajes en que expresar sus sugerencias.

Los mensajes, y las cuentas, serán gestionados por administradores que podrán eliminarlos si consideran que el responsable de los mismos no se ha comportado de manera cívica y educada.

Además, los editores de la web también tendrán la posibilidad de consultar las sugerencias y se encargarán de redactar los artículos que el público lea.

HTML

Página principal

```
10 <body>
11   <aside id="navigation_bar">
12     <figure>
13       <a href="index.html"></a>
14     </figure>
15     <dl>
16       <dt class="content_menu">Plataformas</dt>
17       <dd><a href="platforms/commodore64.html">Commodore 64</a></dd>
18       <dd><a href="platforms/msx.html">MSX</a></dd>
19       <dd><a href="platforms/nas.html">NES</a></dd>
20       <dd><a href="platforms/nintendo64.html">Nintendo 64</a></dd>
21       <dd><a href="platforms/ps1.html">PlayStation 1</a></dd>
22       <dd><a href="platforms/zxspectrum.html">ZX Spectrum</a></dd>
23       <dt class="content_menu">Arquitecturas</dt>
24       <dd><a href="architectures/6502.html">6502</a></dd>
25       <dd><a href="architectures/z80.html">Z80</a></dd>
26       <dd><a href="architectures/mips.html">MIPS</a></dd>
27       <dt class="content_menu">Software</dt>
28       <dd><a href="software/arcadeemul.html">Arcades</a></dd>
29       <dd><a href="software/multisystem.html">Multisistema</a></dd>
30       <dt><a href="login.html">Login</a></dt>
31       <dt><a href="contribute.html">Contribuir</a></dt>
32       <dt><a href="about.html">Acerca de</a></dt>
33     </dl>
34   </aside>
35   <div id="content">
36     <main class="welcome">
37       <h2><input type="image" id="header_toggle" class="navigation_toggle" src="images/hamburger.svg"/>Bienvenido a Hardware Planet</h2>
38       <h3>Una web de referencia sobre...</h3>
39       <div class="image_row">
40         <figure>
41           
42           <figcaption>Hardware</figcaption>
43         </figure>
44         <figure>
45           
46           <figcaption>Arquitecturas</figcaption>
47         </figure>
48       </div>
49       <figure>
50         
51         <figcaption>Emulación</figcaption>
52       </figure>
53       <h3>
54         ...en que todo el mundo puede <a href="contribute.html">colaborar</a>
55       </h3>
56     </main>
57     <footer>
58       <div>Hardware Planet 2023. <span id="disclaimer">La finalidad de esta página es exclusivamente informativa.</span></div>
59     </footer>
60   </div>
61   <script src="https://code.jquery.com/jquery-3.6.4.js" integrity="sha256-a9jBBryygX1Bh51t8GZjXDzyOB+bWve9E10tR0Utj/E=" crossorigin="anonymous"></script>
62   <script src="https://code.jquery.com/ui/1.13.2/jquery-ui.js" integrity="sha256-xL77nh162fcscZK2/v8LsBcb41G7dgULkuXoXB/j91c=" crossorigin="anonymous"></script>
63   <script src="scripts/content_menus.js"></script>
64   <script src="scripts/navigation_bar.js"></script>
65   <script src="scripts/night_mode.js"></script>
66 </body>
```

Sección <body> de la página principal.

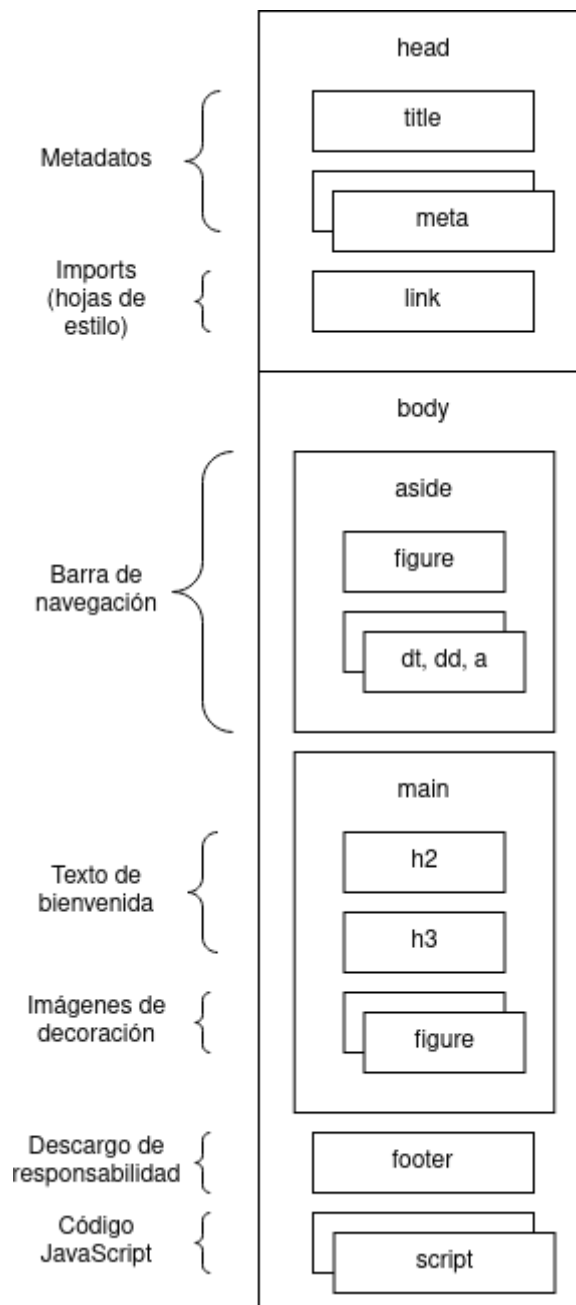
El proceso de diseño de la página principal ha permitido sentar las bases sobre las que se asientan el resto de documentos HTML del proyecto, que solamente varían la parte del <body>, y, especialmente, la que se encuentra dentro de <main>.

Inicialmente tenemos una sección <head> en que se especifican metadatos como el título, las palabras clave, la codificación o el viewport. También se importan las hojas CSS empleadas para dar estilo a la página.

Después, en la zona del <body>, se define un <aside> que servirá como barra de navegación para que el usuario pueda acceder a los diversos contenidos alojados en otros documentos, junto a un logo que permite volver a la página principal.

La sección <main>, a su vez, muestra un texto de bienvenida junto a imágenes que hacen referencia a los conceptos alrededor de los cuales gira todo el proyecto: sistemas y arquitecturas computacionales y emulación.

Por último, se incluye un <footer> que actúa como descargo de responsabilidad avisando de que la única finalidad del sitio web es informativa, así como las etiquetas necesarias para cargar el código JavaScript que proporcione interactividad.



Mapa de etiquetas de la página principal.

Plataformas

```
<main>
<table class="platform_data" platform="PlayStation"></table>
<h1>PlayStation</h1>
<p>
  La PlayStation, a menudo abreviada como PS1, es una consola de quinta generación lanzada por Sony a finales de 1994, contando con un precio de salida de $299.99 en EEUU. Sus principales competidores fueron la Nintendo 64 y la Sega Saturn.
</p>
<p>
  Sony comenzó a desarrollar la plataforma tras una fallida empresa comercial con Nintendo que tenía por objetivo crear un periférico que permitiese el uso de CDs como medio de almacenamiento para los juegos de la SNES. Tras ese evento, Sony decidió apostar por una consola basada en CDs, con énfasis en gráficos poligonales (3D) y que contase con apoyo masivo por parte de <i>third parties</i>, estudios de desarrollo de videojuegos que no estuviesen ligados a la propia compañía.
</p>
<p>
  Precisamente de mano de terceros llegaron algunas de las sagas más populares de esta consola: Crash Bandicoot (Naughty Dog), Spyro (Insomniac), Tomb Raider (Core Design), Metal Gear (Konami) o Final Fantasy (Square). El número de títulos lanzados a lo largo de su vida se estima en 3000, con unas ventas totales de 967 millones de copias.
</p>
<p>
  Gracias a la relativa facilidad de desarrollo que ofrecía, a comparación del resto de consolas del momento, amplio catálogo de juegos, bajo precio en relación a sus especificaciones y agresiva campaña de marketing orientada a adolescentes y jóvenes adultos, la consola se convirtió en un éxito comercial sin precedentes en la industria.
</p>
<h2>Especificaciones</h2>
<ul>
<li>
    CPU: R3000 de 32 bits, siguiendo la arquitectura <a href="../../architectures/mips.html">MIPS</a> (RISC).
  </li>
<li>
    Caché L1: 5 KiB, de los cuales 4 constituyen la caché de instrucciones y el resto se emplea como caché de datos SRAM no asociativa.
  </li>
</ul>
```

Datos de plataforma, introducción y comienzo de la sección de especificaciones.

```
<div class="table_wrapper">
<table class="emulator_table">
<tr>
  <th>Nombre</th>
  <th>Plataformas</th>
  <th>Lenguaje de programación</th>
  <th>Precisión</th>
  <th>Código abierto</th>
  <th>En desarrollo</th>
  <th>Recomendado</th>
</tr>
<tr>
  <td><a href="https://github.com/stenzek/duckstation">DuckStation</a></td>
  <td>  </td>
  <td>C++, C</td>
  <td>Alta</td>
  <td class="affirmative_table_row">✓</td>
  <td class="affirmative_table_row">✓</td>
  <td class="affirmative_table_row">✓</td>
</tr>
<tr>
  <td><a href="https://mednafen.github.io">Mednafen</a></td>
  <td>   </td>
  <td>C++, C</td>
  <td>Alta</td>
  <td class="affirmative_table_row">✓</td>
  <td class="affirmative_table_row">✓</td>
  <td class="affirmative_table_row">✓</td>
</tr>
<tr>
  <td><a href="http://drhell.web.fc2.com/ps1/">XE8RA</a></td>
  <td></td>
  <td>Desconocido</td>
  <td>Alta</td>
  <td class="negative_table_row">X</td>
  <td class="affirmative_table_row">✓</td>
  <td class="negative_table_row">X</td>
</tr>
</table>
```

Tabla de emuladores.

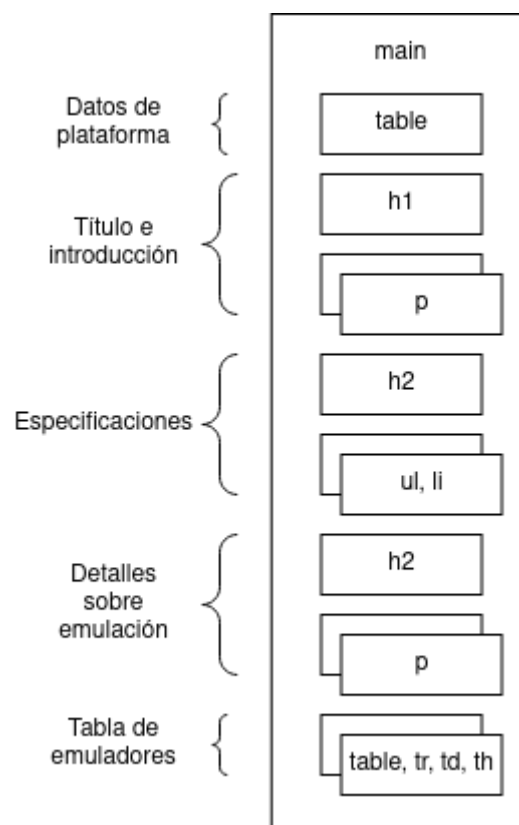
El apartado de las plataformas, al igual que el resto, difiere únicamente de la página principal en la sección del contenido (delimitada por las etiquetas <main>).

En este caso, contamos con una tabla que muestra los datos principales de la plataforma: nombre, fecha de lanzamiento, fecha de discontinuación, arquitectura computacional...

Después, hay una breve introducción acerca del contexto histórico en que fue lanzada la misma y por qué se volvió popular o falló en términos de ventas. También en este apartado se mencionan de forma somera algunas de las características principales de su hardware, que se desarrollarán posteriormente en la sección de especificaciones, mucho más precisa a ese respecto.

A continuación, se comenta el estado general en la emulación del sistema, si hay emuladores que funcionan, cuáles son sus principales problemas, de existir, o si por el contrario la plataforma lleva siendo emulada satisfactoriamente mucho tiempo.

Finalmente, se muestra una tabla que pretende servir de comparativa entre el software disponible para dicha emulación, ofreciendo enlaces a sus páginas oficiales o al código fuente, señalando el lenguaje de programación empleado y los sistemas operativos en que se puede ejecutar, e indicando algunas otras características como si es de código abierto o sigue en desarrollo activo.



Mapa de etiquetas de una página de plataforma.

Arquitecturas

```
<main>
<table class="platform_data" platform="MIPS"></table>
<h1>MIPS</h1>
<p>
  MIPS (Microprocessor without Interlocked Pipelined Stages) es una familia de arquitecturas computacionales (ISAs) de categoría RISC desarrollada por la empresa estadounidense MIPS Computer Systems, hoy en día MIPS Technologies.
</p>
<p>
  Existen múltiples versiones de MIPS, incluyendo MIPS I, II, III, IV y V, así como múltiples implementaciones de 32 y 64 bits (MIPS32/64, respectivamente). Las primeras ediciones eran exclusivamente de 32 bits, llegando las de 64 bits posteriormente. MIPS32/64 difiere de las ediciones I-V al incorporar el System Control Coprocessor como elemento hardware que actúa en modo kernel, en adición a la arquitectura de modo usuario.
</p>
<p>
  Por otra parte, la arquitectura MIPS cuenta con múltiples extensiones opcionales: MIPS-3D, que proporciona instrucciones SIMD de punto flotante destinadas a tareas 3D comunes, MDMX, similar a la anterior pero con operaciones de enteros, MIPS16e, que añade compresión al flujo de instrucciones para que los programas ocupen menos memoria, o MIPS MT, que añade capacidades de multithreading.
</p>
<p>
  Las materias relativas al hardware en los currículos de universidades y escuelas técnicas suelen incluir el estudio de la arquitectura MIPS, a causa de su simpleza e influencia en otras arquitecturas RISC como Alpha. En marzo de 2021, MIPS Technologies anunció el fin del desarrollo de la arquitectura, a causa del interés de la compañía en transicionar a RISC-V.
</p>
<h2>Instrucciones</h2>
<p>
  Al ser una arquitectura RISC, todas las instrucciones de MIPS cuentan con el mismo número de bits (32), y la mayoría de ellas son muy simples conceptualmente, a cambio de ejecutarse en menos ciclos de reloj, o el mismo número de estos pero siendo cada uno más breve. Por otra parte, gracias al elevado número de registros de propósito general y de punto flotante (generalmente 32 de ambos tipos), el hecho de no operar directamente en memoria no supone un problema tan notorio como representaría en arquitecturas CISC.
</p>
<p>
  Los tipos principales de instrucción, junto a un ejemplo de cada uno, son los siguientes:
</p>
<ul>
<li>
      Tipo R, excluyendo saltos (fundamentalmente aritmético-lógicas) (ADD): Modifican únicamente el contenido de los registros a través del uso de la ALU. Esta clase de instrucciones cuenta con 3 registros: destino y dos operandos.
    </li>
</ul>
</main>
```

Información de arquitectura, introducción y sección de instrucciones.

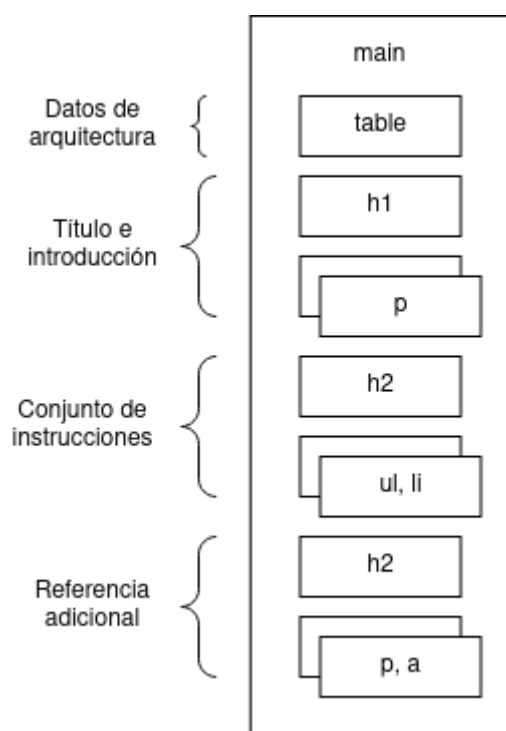
```
<h2>Referencia adicional</h2>
<p>
  Debido fundamentalmente a su interés pedagógico, así como a su empleo en hardware comercial como videoconsolas, la arquitectura MIPS se encuentra especialmente bien documentada, tanto en webs de terceros como por parte de la propia empresa encargada de su desarrollo. Es sencillo encontrar información detallada de cada una de las instrucciones, así como de otras particularidades de la arquitectura.
</p>
<p>
  Algunas herramientas de relevancia son
  <a href="https://s3-eu-west-1.amazonaws.com/downloads-mips/documents/MIPS_Architecture_MIPS64_InstructionSet_%20AFP_P_MD00087_06_05.pdf">el manual oficial de MIPS64</a> y
  <a href="https://spimsimulator.sourceforge.net/">SPIM, un simulador de MIPS32</a>
</p>
</main>
```

Apartado de referencia adicional.

Los artículos dedicados a las arquitecturas computacionales siguen un esquema muy similar a los de las plataformas: tabla con datos e imagen, título, introducción y desarrollo posterior.

Cabe destacar, eso sí, que en lugar de hablar sobre especificaciones se comenta en mayor profundidad el conjunto de instrucciones de la ISA y, además, hay una sección adicional en que se menciona si la arquitectura está bien documentada y se proporcionan enlaces a recursos que puedan ofrecer información complementaria.

Por otra parte, tampoco hay una tabla de emuladores al no tener sentido en este contexto. Sin embargo, una futura ampliación del sitio web podría incluir información sobre simuladores de las arquitecturas computacionales.



Mapa de etiquetas de una página de arquitectura.

Contribuir

```
<main>
<h1 class="notable">Contribuir</h1>
<p>
  Por favor, antes de contribuir echa un vistazo al muro de sugerencias y comprueba que la tuya no ha sido publicada antes por otra persona,
  ya que nos facilita mucho la labor de administración. Gracias.
</p>
<form class="contribute_form">
  <label for="fname"><b>Nombre (o apodo)</b></label>
  <input type="text" id="fname" name="nombre" required/>

  <label for="lname"><b>Apellidos</b></label>
  <input type="text" id="lname" name="apellidos"/>

  <label for="country"><b>País</b></label>
  <input type="text" id="country" name="pais"/>

  <label for="email"><b>Correo electrónico</b></label>
  <input type="text" id="email" name="correo" required/>

  <div class="contribute_form_row"><textarea id="mensaje" name="mensaje"></div>
  <div class="contribute_form_row"><input class="form_submit" type="submit" value="Enviar"></div>
</form>
```

Mensaje de advertencia y formulario de contribución.

```
<div class="message_board">
  <h2>Muro de sugerencias</h2>
  <div class="message_entry">
    
    <h3 class="message_name">Pedro</h3>
    <p class="message_body">
      Pienso que sería buena idea añadir una sección separada para cada emulador en que se explicaran las ideas tras su diseño
      y las características que ofrecen, así como su compatibilidad con diferentes juegos.
    </p>
  </div>
  <div class="message_entry">
    
    <h3 class="message_name">César</h3>
    <p class="message_body">
      Muy buena página en lo relativo al contenido, aunque faltan datos sobre consolas actuales.
    </p>
  </div>
  <div class="message_entry">
    
    <h3 class="message_name">Sonia</h3>
    <p class="message_body">
      Me gustaría que quienes tenemos una cuenta en la página pudiésemos marcar artículos a seguir y recibiésemos notificaciones
      cuando son editados por alguien.
    </p>
  </div>
</div>
</main>
```

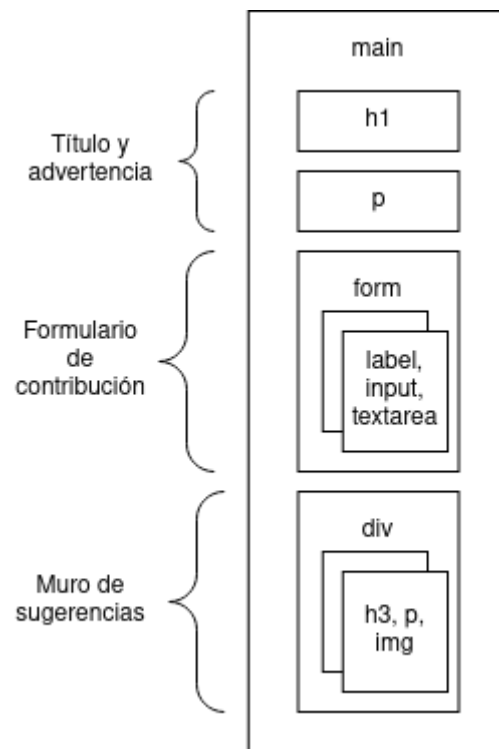
Muro de sugerencias.

La página de contribución está compuesta por dos apartados principales: el formulario de contribución y el muro de sugerencias.

Antes del primero, hay un mensaje que avisa a los usuarios de que lean aquellas ideas ya publicadas por otras personas para evitar repetirlas y facilitar la labor de quien mantenga la web.

El formulario en sí está compuesto por etiquetas (<label>) que nombran los campos a cubrir y elementos de entrada (<input>) en que se escribe. Algunos de estos últimos son obligatorios y deben seguir un formato particular que se comprueba a través del uso de expresiones regulares. También se emplea una etiqueta <textarea> para definir la región en que se ha de redactar el mensaje.

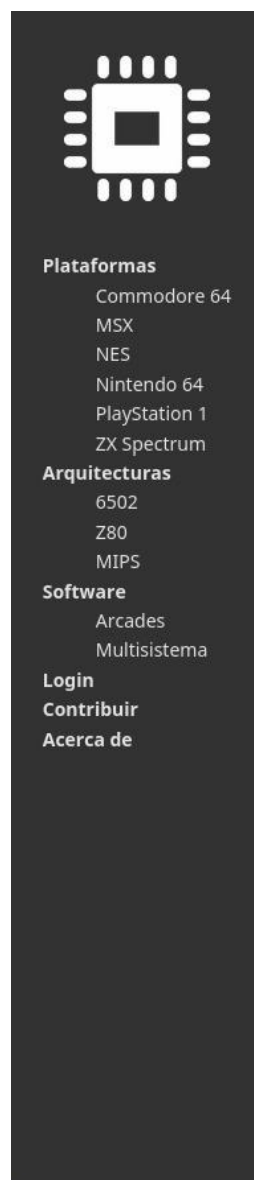
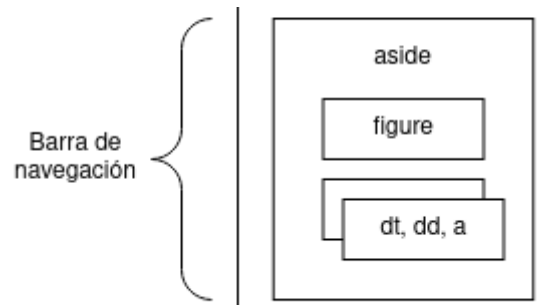
Por otra parte, el muro de sugerencias cuenta con un elemento <div> por cada una de las entradas, que agrupa las partes constituyentes de estas: una imagen () que representa el icono del usuario que haya publicado el mensaje, en caso de estar registrado, el nombre o nick (<h1>) y el contenido del mensaje (<p>).



Mapa de etiquetas de la página de contribución.

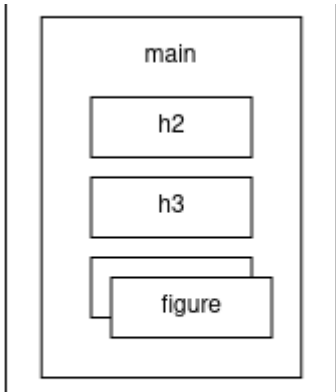
CSS

Página principal



Texto de bienvenida

Imágenes de decoración

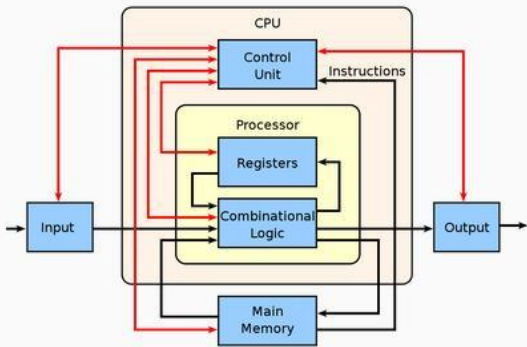


Bienvenido a Hardware Planet

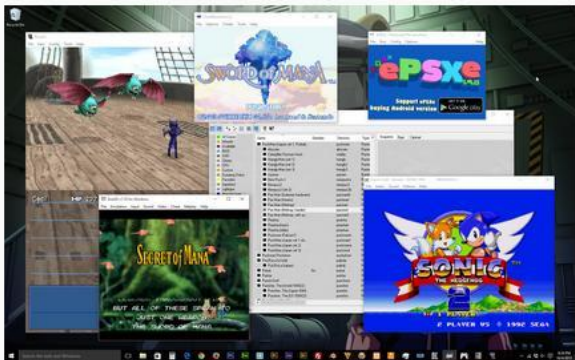
Una web de referencia sobre...



Hardware



Arquitecturas



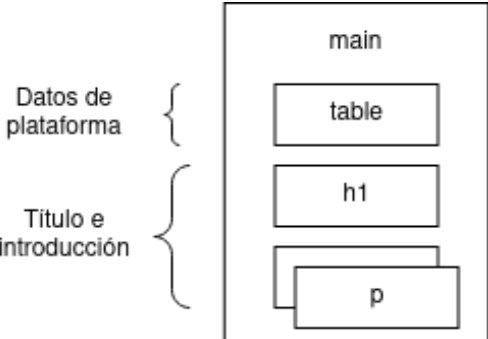
Emulación

Descargo de responsabilidad

footer

Hardware Planet 2023. La finalidad de esta página es exclusivamente informativa.

Plataformas




PlayStation

La PlayStation, a menudo abreviada como PS1, es una consola de quinta generación lanzada por Sony a finales de 1994, contando con un precio de salida de \$299.99 en EEUU. Sus principales competidores fueron la Nintendo 64 y la Sega Saturn.

Sony comenzó a desarrollar la plataforma tras una fallida empresa comercial con Nintendo que tenía por objetivo crear un periférico que permitiese el uso de CDs como medio de almacenamiento para los juegos de la SNES. Tras ese evento, Sony decidió apostar por una consola basada en CDs, con énfasis en gráficos poligonales (3D) y que contase con apoyo masivo por parte de *third parties*, estudios de desarrollo de videojuegos que no estuviesen ligados a la propia compañía.

Precisamente de mano de terceros llegaron algunas de las sagas más populares de esta consola: Crash Bandicoot (Naughty Dog), Spyro (Insomniac), Tomb Raider (Core Design), Metal Gear (Konami) o Final Fantasy (Square). El número de títulos lanzados a lo largo de su vida se estima en 3000, con unas ventas totales de 967 millones de copias.

Gracias a la relativa facilidad de desarrollo que ofrecía, a comparación del resto de consolas del momento, amplio catálogo de juegos, bajo precio en relación a sus especificaciones y agresiva campaña de marketing orientada a adolescentes y jóvenes adultos, la consola se convirtió en un éxito comercial sin precedentes en la industria.



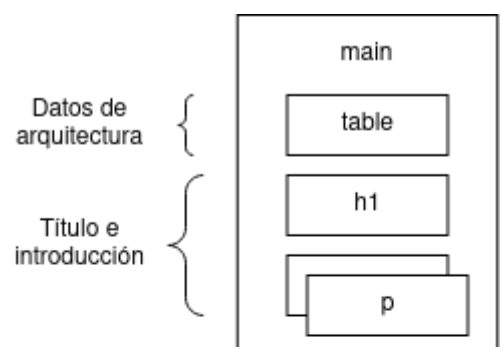
Nombre	PlayStation
Fabricante	Sony
Fecha de lanzamiento	3 de diciembre de 1994
Fecha de discontinuación	23 de marzo de 2006
Unidades vendidas	102,4 millones de consolas
Arquitectura empleada	MIPS



Nombre	Plataformas	Lenguaje de programación	Precisión	Código abierto	En desarrollo	Recomendado
DuckStation		C++, C	Alta	✓	✓	✓
Mednafen		C++	Alta	✓	✓	✓
XEBRA		Desconocido	Alta	✗	✓	✗
ares		C++	Alta	✓	✓	✗
ePSXe		Desconocido	Dependiente de los plugins	✗	✗	✗

Arquitectura

Las páginas de arquitecturas siguen el mismo estilo que las de plataformas pero sin tabla de emuladores.



6502

El MOS Technology 6502 es un procesador de 8 bits diseñado por un reducido equipo de desarrollo para MOS Technology. Dicho equipo había trabajado previamente en Motorola, concretamente en el proyecto del 6800, procesador similar a este, pero menos potente y más caro de producir.

Cuando se introdujo en 1975, el 6502 era el microprocesador menos costoso del mercado por un margen sustancial, vendiéndose por alrededor de un sexto del coste de los diseños que competían con el mismo, como el Motorola 6800 ya mencionado o el Intel 8080. A causa de ello, los precios de los procesadores empezaron a decrementsar progresivamente. Por otra parte, junto al Zilog Z80, el 6502 dio vida a una serie de proyectos que resultarían en la revolución de los ordenadores domésticos a comienzos de los años 80.

Numerosas consolas y ordenadores domésticos populares en los 80 y comienzos de los 90 emplearon su arquitectura, entre los que podemos nombrar la Atari 2600, el Apple II, la Nintendo Entertainment System, el Commodore 64 o el BBC Micro. Sin embargo, poco después de que el 6502 saliese al mercado, MOS Technology fue adquirido por Commodore International, lo que obligó a ciertos fabricantes, que competían con Commodore, a modificar levemente ciertos aspectos de la arquitectura, como eliminar el soporte para BCD, *Binary Coded Decimal*, a fin de evitar conflictos legales.

En 1981, Western Design Center comenzó el desarrollo de una versión CMOS, el 65C02, que continúa siendo usado en sistemas empostrados, con una producción anual de cientos de millones de chips.

Instrucciones

A pesar de la simpleza de su conjunto de instrucciones, la arquitectura se considera CISC debido a que el tamaño de las mismas no es siempre igual. En función del modo de direccionamiento, varían entre 1 y 3 bytes (de 8 a 24 bits).

- Aritmético-lógicas: 15 instrucciones (ADC, AND, ASL, DEC, DEX, DEY, EOR, INC, INX, INY, LSR, ORA, ROL, ROR, SBC) que permiten modificar el contenido de los registros o la memoria empleando la ALU. Todas ellas cambian el estado del procesador de modo que no sea necesario realizar comparaciones a posteriori con los valores obtenidos.
- Salto condicional: 8 instrucciones (BPL, BMI, BVC, BVS, BCC, BCS, BNE, BEQ) que permiten realizar saltos en función del registro de estado del procesador. No actualizan dicho estado y el número de ciclos que requieren varía en función de si el salto se toma y si se cruza a otra página de memoria.
- Salto incondicional: 2 instrucciones (JSR, JMP) que permiten modificar el flujo de ejecución del programa. La diferencia entre ellas es que una almacena el registro contador de programa en la pila (JSR), para

Nombre	MOS 6502
Fabricante	MOS Technology, Rockwell, Synetek...
Fecha de lanzamiento	1975
Frecuencia máxima	Entre 1 y 3 MHz
Tamaño de palabra	8 bits
Tamaño de dirección	16 bits
Número de instrucciones	56
Categoría	CISC

Contribuir

Contribuir

Por favor, antes de contribuir echa un vistazo al muro de sugerencias y comprueba que la tuya no ha sido publicada antes por otra persona, ya que nos facilita mucho la labor de administración. Gracias.

Nombre (o apodo)

Apellidos

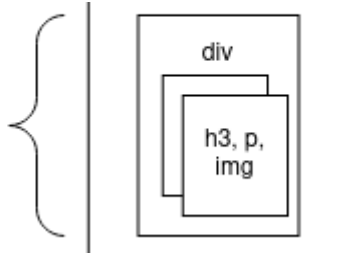
País

Correo electrónico

label, input, textarea

Enviar

Muro de
sugerencias



Muro de sugerencias



Pedro

Pienso que sería buena idea añadir una sección separada para cada emulador en que se explicaran las ideas tras su diseño y las características que ofrecen, así como su compatibilidad con diferentes juegos.



César

Muy buena página en lo relativo al contenido, aunque faltan datos sobre consolas actuales.



Sonia

Me gustaría que quienes tenemos una cuenta en la página pudiésemos marcar artículos a seguir y recibiésemos notificaciones cuando son editados por alguien.

JavaScript

Modo noche



PlayStation

La PlayStation, a menudo abreviada como PS1, es una consola de quinta generación lanzada por Sony a finales de 1994, contando con un precio de salida de \$299.99 en EEUU. Sus principales competidores fueron la Nintendo 64 y la Sega Saturn.

Sony comenzó a desarrollar la plataforma tras una fallida empresa comercial con Nintendo que tenía por objetivo crear un periférico que permitiese el uso de CDs como medio de almacenamiento para los juegos de la SNES. Tras ese evento, Sony decidió apostar por una consola basada en CDs, con énfasis en gráficos poligonales (3D) y que contase con apoyo masivo por parte de *third parties*, estudios de desarrollo de videojuegos que no estuviesen ligados a la propia compañía.

Precisamente de mano de terceros llegaron algunas de las sagas más populares de esta consola: Crash Bandicoot (Naughty Dog), Spyro (Insomniac), Tomb Raider (Core Design), Metal Gear (Konami) o Final Fantasy (Square). El número de títulos lanzados a lo largo de su vida se estima en 3000, con unas ventas totales de 967 millones de copias.

Gracias a la relativa facilidad de desarrollo que ofrecía, a comparación del resto de consolas del momento, amplio catálogo de juegos, bajo precio en relación a sus especificaciones y agresiva campaña de marketing orientada a adolescentes y jóvenes adultos, la consola se convirtió en un éxito comercial sin precedentes en la industria.

Especificaciones

- CPU: R3000 de 32 bits, siguiendo la arquitectura MIPS (RISC).
- Caché L1: 5 KIB, de los cuales 4 constituyen la caché de instrucciones y el resto se emplea como caché de datos SRAM no asociativa.
- Unidades adicionales de la CPU: Geometry Transformation Engine, que proporciona instrucciones vectoriales empleadas para procesamiento de gráficos 3D, Motion Decoder, que permite la reproducción de vídeo y la descompresión de este, junto a imágenes, a la VRAM, y System Control Coprocessor, que gestiona la memoria virtual, las interrupciones de la CPU y las excepciones.
- RAM: 2 MB de DRAM principal, junto a memoria adicional para la GPU (1 MB de framebuffer) y la unidad de audio (512 KIB).
- GPU: Unidad de procesamiento gráfico de 32 bits, diseñada por Toshiba, que soporta renderizado de polígonos, texturización y gráficos en 2D, incluyendo transformaciones de sprites, cuya señal de vídeo alcanza una resolución de 640x480 píxeles, con true color (24 bits, 16 millones de colores).
- SPU: Unidad de procesamiento de sonido de 16 bits, desarrollada por Sony, que soporta muestreo hasta frecuencias de 44,1 KHz, así como efectos digitales y secuenciación MIDI.
- Medios: Discos CD-ROM.

Emulación

Los emuladores de PlayStation han estado disponibles desde finales de los años 90, funcionando generalmente mejor que aquellos de Nintendo 64 (perteneciente a la misma generación de consolas), a pesar de emplear un sistema de plugins.

Tanto plugins como emuladores eran a menudo de código cerrado, se actualizaban poco y su precisión resultaba cuestionable, pero a partir de la década de 2010 comenzó a surgir nuevo software que ofrece elevada precisión e incluso mejoras al software original, permitiendo, por ejemplo, incrementar la resolución interna:

Nombre	Plataformas	Lenguaje de programación	Precisión	Código abierto	En desarrollo	Recomendado
DuckStation	Windows, Linux, macOS	C++, C	Alta	✓	✓	✓
Mednafen	Windows, Linux, macOS, Raspberry Pi	C++	Alta	✓	✓	✓
XEBRA	Windows	Desconocido	Alta	✗	✓	✗



Nombre PlayStation
Fabricante Sony
Fecha de lanzamiento 3 de diciembre de 1994
Fecha de discontinuación 23 de marzo de 2006
Unidades vendidas 102,4 millones de consolas
Arquitectura empleada MIPS

```
2 // Obtenemos los elementos cuyo color queremos modificar
3 const body = document.getElementsByTagName("body")[0];
4 const platform_data = document.getElementsByClassName("platform_data")[0];
5
6 function handleKey(event) {
7     if(event.key === "n" && event.getModifierState("Alt")) {
8         if(body.classList.contains("night_mode")) { // El modo noche ya está activo y queremos eliminarlo
9             body.classList.remove("night_mode");
10            platform_data.classList.remove("night_mode");
11        }
12        else {
13            body.classList.add("night_mode");
14            platform_data.classList.add("night_mode");
15        }
16    }
17 }
18
19
20 body.addEventListener("keydown", handleKey); // Cuando se pulsa alguna tecla, llamamos a la función callback handleKey()
21
```

Uno de los efectos implementados mediante el uso de JavaScript ha sido la adición de un modo noche, de forma que si se pulsa la combinación de teclas Alt + N, se activa, invirtiendo el patrón de colores para posibilitar una mejor experiencia de cara a usuarios que la visiten en condiciones de luz reducida.

Para ello, reaccionamos a un evento de tipo “keydown”, producido cuando se pulsa alguna tecla, comprobamos que se trata de la N y que en efecto el modificador Alt se encuentra activo, y en caso de ser así, activamos o desactivamos el modo mencionado.

Por otra parte, como métodos de acceso al DOM se han utilizado `getElementsByTagName()`, que proporciona los elementos correspondientes a una determinada etiqueta HTML, y `getElementsByClassName`, análogo pero buscando en base al atributo “class”.

La modificación del patrón de colores en sí se lleva a cabo agregando a los elementos afectados una nueva clase, “night_mode”, que se encarga de cambiar los colores tanto del background como del foreground y los bordes:

```
29  .night_mode {  
30      background: #222222;  
31      color: #FAFAFA;  
32      border-color: #FAFAFA;  
33  }
```

Barra de navegación colapsable



Otro efecto añadido a la web ha sido el de colapsar la barra de navegación para aumentar el espacio disponible a la hora de renderizar el contenido principal. Esto se lleva a cabo mediante unos botones agregados en la barra de navegación en sí y en el título de cada página (en este caso, solo visible cuando la barra se encuentra oculta).

Al pulsarlos (evento “click”), se alterna su estado, modificando la propiedad “display” y estableciéndola a “none” o “block” según corresponda. También se alterna la visibilidad del botón situado en el título de la página.

Como métodos de acceso al DOM se han empleado `getElementById()`, que devuelve el primer elemento hallado en el documento HTML que cuente con un cierto identificador, y `querySelectorAll()`, que permite emplear directamente un selector como los de CSS para obtener un conjunto de nodos.

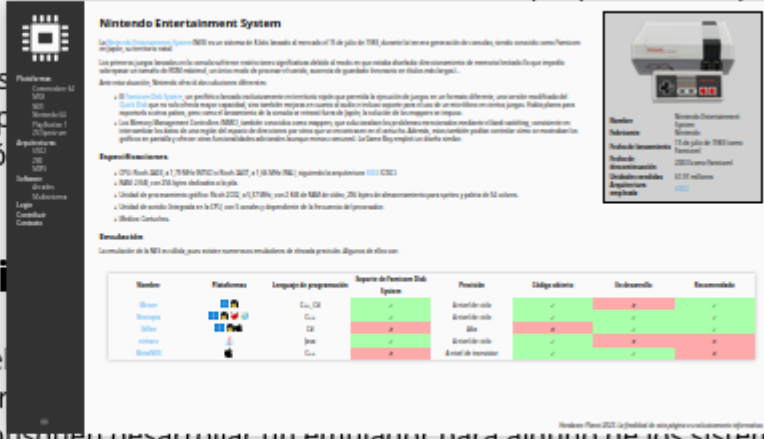
```
1
2  const bar = document.getElementById("navigation_bar");
3  const nav_toggles = document.querySelectorAll(".navigation_toggle");
4
5  function toggleBar() {
6      if(bar.style.display === "none") {
7          bar.style.display = "flex";
8          document.getElementById("header_toggle").style.display = "none";
9      }
10     else {
11         bar.style.display = "none";
12         document.getElementById("header_toggle").style.display = "inline";
13     }
14 }
15
16 nav_toggles.forEach(function(toggle) {
17     toggle.addEventListener("click", toggleBar);
18 })
19
```

Previsualización de enlaces

- Transferencia: 6 instrucciones (TAX, TAY, TSX, TXA, TXS, TYA) que permiten copiar otro.
- Misceláneas: Además como causar interrupción para realizar una operación (BIT).

Referencia adicional

Existen numerosas webs de direccionamiento u otros (memoria), o quienes busquen desarrollar un emulador para alguno de los sistemas: el [Commodore 64](#) o la [Nintendo Entertainment System](#).



Modelo	Plataforma	Lenguaje de programación	Número de chips	Procesador	Código fuente	Disquete	Reconstruido
NES-101	FC	C++	12	6502	✓	✓	✓
NES-101	FC	C++	12	6502	✓	✓	✓
NES-101	FC	C++	12	6502	✓	✓	✓
NES-101	FC	C++	12	6502	✓	✓	✓
NES-101	FC	C++	12	6502	✓	✓	✓

Este es, con diferencia, el efecto más complejo implementado con JavaScript puro. Cuando se pasa el ratón por encima de alguno de los enlaces presentes en la página, se muestra una previsualización de la web a la que lleva ese enlace. Sin embargo, por seguridad, esto está limitado únicamente a páginas asociadas (no externas).

La implementación se lleva a cabo mediante “iframes”. En un primer momento, cuando se abre la página y se ejecuta el script, se añade un iframe a cada uno de los links presentes en esta, dotándolo del estilo deseado (propiedad “style” en JavaScript) para que mantenga unas dimensiones razonables y se pueda diferenciar visualmente del resto de contenido de la web. Además, se define una función callback a ejecutar cuando se pone el ratón encima del enlace (evento “mouseenter”) o, análogamente, sale de la región correspondiente al enlace (evento “mouseleave”).

```

44  for(elem of links) {
45      if(!elem.getAttribute('href').startsWith('http')) {
46          let popup_child = document.createElement('span');
47          popup_child.style.visibility = 'hidden';
48          popup_child.style.position = 'absolute';
49          popup_child.style.zIndex = '100';
50          popup_child.style.boxShadow = '0 0 10px rgba(0, 0, 0, 0.30)';
51
52          let popup_iframe = document.createElement('iframe');
53          popup_iframe.setAttribute('src', elem.getAttribute('href'));
54          popup_iframe.classList.add('preview');
55          popup_child.appendChild(popup_iframe);
56
57          let parent = elem.parentElement;
58          parent.insertBefore(popup_child, elem);
59
60          let frame_dimensions = popup_iframe.getBoundingClientRect();
61          popup_child.style.width = frame_dimensions.width + 'px';
62          popup_child.style.height = frame_dimensions.height + 'px';
63
64          elem.addEventListener('mouseenter', togglePopup);
65          elem.addEventListener('mouseleave', togglePopup);
66      }
67  }

```

Este callback se encarga de comprobar el tipo de evento, y, en caso de ser “mouseenter”, halla las coordenadas en que ubicar el iframe, teniendo en cuenta las dimensiones de la página, y cambia la propiedad “visibility” para mostrar la previsualización. De lo contrario, podemos asumir que el evento es “mouseleave” y solamente debemos hacer invisible el elemento.

Por otra parte, algunos de los métodos utilizados para acceder al DOM han sido `querySelectorAll()`, `getAttribute()`, `setAttribute()` o `appendChild()`.

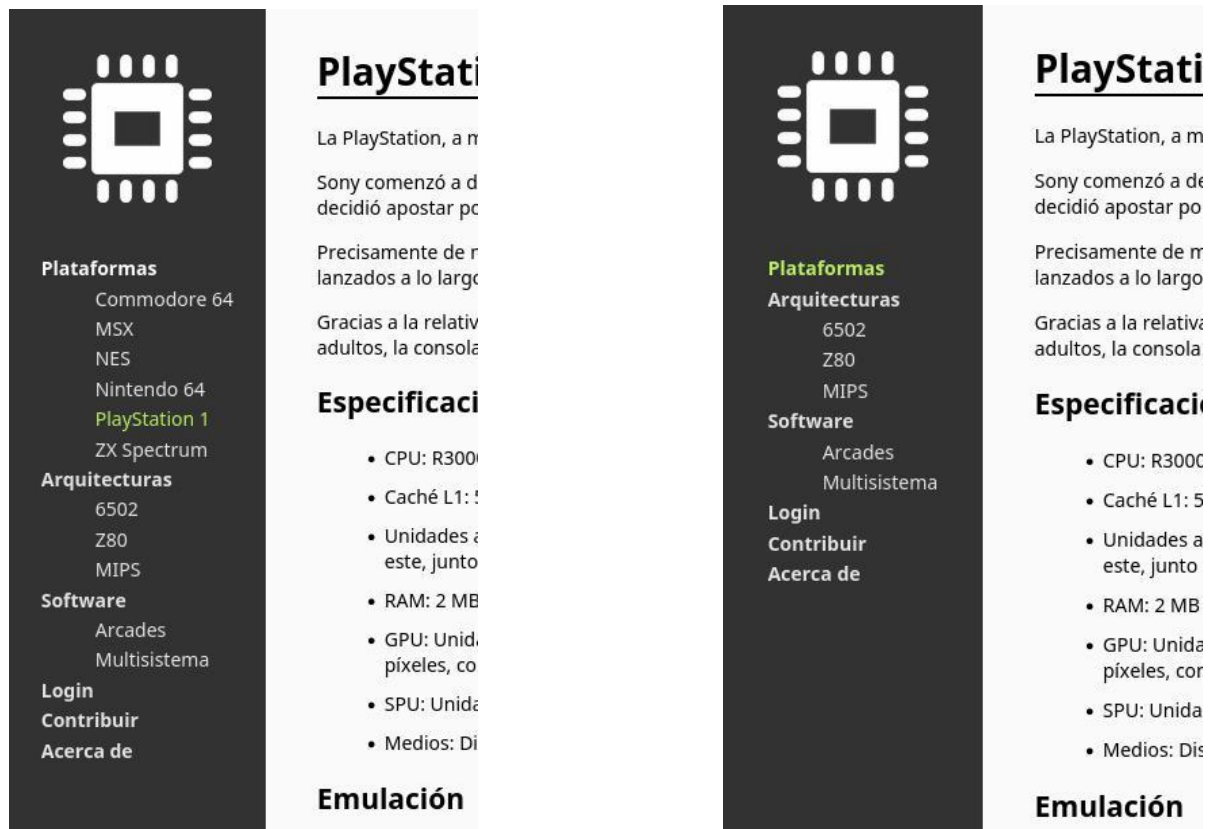
```

4 function togglePopup(event) {
5     if(event.type === 'mouseenter') {
6         // Find vertical coordinate
7         let top_offset = this.getBoundingClientRect().bottom + 10;
8         let elem_height = this.getBoundingClientRect().bottom - this.getBoundingClientRect().top;
9
10        let container_height = parseInt(this.previousSibling.style.height);
11        let container_width = parseInt(this.previousSibling.style.width);
12
13        let window_height = document.documentElement.clientHeight;
14        if(top_offset + container_height > window_height) {
15            top_offset -= elem_height;
16            top_offset -= container_height;
17            top_offset -= 20;
18        }
19
20        // Find horizontal coordinate
21        let left_offset = this.getBoundingClientRect().left;
22        let window_width = document.documentElement.clientWidth;
23
24
25        if(left_offset + container_width > window_width) {
26            let right_offset = window_width - this.getBoundingClientRect().right;
27            this.previousSibling.style.right = right_offset + 'px';
28            this.previousSibling.style.removeProperty('left');
29        }
30        else {
31            this.previousSibling.style.left = left_offset + 'px';
32            this.previousSibling.style.removeProperty('right');
33        }
34
35        this.previousSibling.style.top = top_offset + 'px';
36
37        this.previousSibling.style.visibility = 'visible';
38    }
39    else {
40        this.previousSibling.style.visibility = 'hidden';
41    }
42 }

```


JQuery

Menús laterales ocultables



Otro de los efectos que se ha añadido a la barra lateral es la posibilidad de ocultar parte de la misma haciendo click en alguno de los apartados. Al llevar a cabo esta acción, se produce una animación que hace desaparecer los elementos pertenecientes al mismo y, en caso de encontrarse la página actual en dicho apartado, se colorea su nombre para indicarlo.

La implementación se ha realizado mediante el uso de JQuery. Cuando el documento se carga (función `ready()`), se colorea la sección actual y se añade un manejador (handler) para el evento “click” en cualquiera de las secciones de la barra lateral.

En la función callback, se recorre todos los nodos que representan páginas de la sección en cuestión, animándolos con un efecto “slide” para posteriormente activar o desactivar su visibilidad.

Por último, si alguno de los nodos recorridos corresponde a la página actual, se modifica el color del subapartado para indicar que el usuario se encuentra en el mismo.

```
1
2 $(document).ready(function() {
3
4     $('#active_item').css('color', '#abed40');
5
6     $('.content_menu').on('click', function() {
7
8         let elem = $(this).next();
9
10        let contains_current = false;
11
12        while($(elem).prop('tagName').toLowerCase() == 'dd') {
13            let curr = elem;
14            if($(elem).css('display') != 'none') {
15                $(curr).slideUp(1000, () => $(curr).css('display', 'none'));
16            }
17            else {
18                $(curr).slideDown(1000, () => $(curr).css('display', 'block'));
19            }
20
21            if($(curr).attr('id') == 'active_item') {
22                contains_current = true;
23            }
24
25            elem = $(elem).next();
26        }
27
28        if(contains_current && $(this).css('color') != 'rgb(' + 0xAB + ', ' + 0xED + ', ' + 0x40 + ')') {
29            $(this).animate({color: '#ABED40'}, 1000);
30        }
31        else {
32            $(this).animate({color: '#FAFAFA'}, 1000);
33        }
34    });
35
36
37 });
38
```


ES6

Aviso de cookies

La legislación europea nos obliga a avisarte de que este sitio podría hacer uso de cookies para mejorar tu experiencia.



Al diseñar y desarrollar un sitio web, es importante tomar en consideración la legislación relativa al país donde este se vaya a alojar. En el caso de la Unión Europea, sus leyes obligan a las páginas a avisar de que van a emplear cookies en caso de ser así.

En general, esta clase de avisos suelen implementarse de modo que, cuando un usuario los cierra, no vuelven a aparecer si visita de nuevo la web. Para ello, es necesario emplear algún tipo de almacenamiento persistente que permita señalar si ya se ha notificado al usuario.

Concretamente, en este caso, se ha empleado la Web Storage API de JavaScript, que proporciona un objeto llamado `localStorage` para guardar de manera indefinida información en el navegador de quien accede a la página, junto a una propiedad `"cookies_shown"` que, en caso de estar presente, indica que el mensaje ya ha sido mostrado y descartado por el usuario con anterioridad.

De este modo, cuando el script en cuestión se ejecuta, se comprueba si `"cookies_shown"` se encuentra definida en `localStorage` y, de lo contrario, genera un popup que se muestra una vez se ha terminado de cargar la página. Al pulsar el botón que permite cerrarlo, se establece la propiedad en cuestión y el aviso no se vuelve a mostrar de nuevo.

Por otra parte, han resultado de utilidad algunas características introducidas en JavaScript ES6, como las variables restringidas a ámbitos (palabra reservada `"let"`) y las funciones arrow, que reducen la cantidad de código necesaria para definir una función callback que gestione eventos.

```

1
2 if(!localStorage.getItem('cookies_shown')) {
3
4     let cookie_notice = document.createElement('div');
5
6     cookie_notice.style.height = '6vh';
7     cookie_notice.style.width = '100vw';
8     cookie_notice.style.backgroundColor = '#202020';
9     cookie_notice.style.position = 'fixed';
10    cookie_notice.style.bottom = '0px';
11    cookie_notice.style.left = '0px';
12    cookie_notice.style.zIndex = '2';
13    cookie_notice.style.display = 'flex';
14    cookie_notice.style.flexDirection = 'row';
15    cookie_notice.style.alignItems = 'center';
16    cookie_notice.style.justifyContent = 'center';
17
18    let cookie_text = document.createElement('p');
19    cookie_text.innerText = 'La legislación europea nos obliga a avisarte de que este sitio podría hacer uso de cookies para mejorar tu experiencia.';
20    cookie_text.style.color = '#DDDDDD';
21
22    let cookie_ignore_button = document.createElement('img');
23
24    let current_script = document.currentScript.src;
25
26    let re = new RegExp(/^(.*)\/.*\//);
27    let root_path = re.exec(current_script)[1];
28
29    cookie_ignore_button.src = root_path + '/images/close.png';
30    cookie_ignore_button.style.height = '3vh';
31    cookie_ignore_button.style.width = '3vh';
32    cookie_ignore_button.style.cursor = 'pointer';
33    cookie_ignore_button.style.marginLeft = '1vw';
34
35    cookie_ignore_button.addEventListener('click', ev => {
36        cookie_notice.style.display = 'none';
37        localStorage.setItem('cookies_shown', 'yes');
38    });
39
40    window.addEventListener('load', ev => {
41        cookie_notice.appendChild(cookie_text);
42        cookie_notice.appendChild(cookie_ignore_button);
43        document.getElementsByTagName('body')[0].appendChild(cookie_notice);
44    });
45
46 }
47

```

Expresiones regulares

Validación de elementos en formularios

🌐 file://

El nombre no tiene un formato válido. Ha de estar compuesto por entre 4 y 100 letras, números o guiones bajos.

OK

🌐 file://

El correo electrónico no tiene un formato válido.

OK

Si presentamos al usuario un formulario que pueda cubrir con los datos relativos a su persona, resulta necesario comprobar la información introducida para verificar que se ajusta a lo esperado. Por ejemplo, no resulta razonable permitir un email que no corresponda al formato “nombre@dominio”, y trabajar con datos inválidos podría constituir un problema de seguridad.

Una forma de evitarlo es emplear regex (expresiones regulares) que determinen si los datos se ajustan a un patrón dado. En el contexto de este proyecto, eso se ha llevado a cabo en los formularios de contribución por parte de los usuarios.

Cuando un usuario introduce su nombre o nick, tiene que cumplir una serie de condiciones (indicadas en la imagen) que se le hacen saber en caso de no estar satisfaciéndolas. Lo mismo ocurre con el correo electrónico, aunque en ese caso no se avisa del formato, ya que se supone conocido.

Para la implementación, se ha empleado una combinación entre JQuery y la clase RegExp de JavaScript. Cuando la página se carga (evento “ready”) se selecciona el botón que permite al usuario enviar los datos de formulario y se le añade un callback para el evento “click” que se encarga de construir las expresiones regulares

deseadas y comprobar la adecuación de los datos, avisando al usuario mediante un mensaje en el navegador (función alert()) de haber algún problema con ellos.

Por último, en caso de que los datos sean válidos, se comunica al usuario que la función de contribución se encuentra temporalmente deshabilitada (ya que requeriría enviar datos a un servidor web que no hemos implementado).

```
1
2 $(document).ready(function() {
3
4     $('#form_submit').on('click', function() {
5
6         let name_regex = new RegExp('^[a-zA-Z0-9_]{4,100}$');
7         let name = $('#form #fname').val();
8
9         if(!name_regex.test(name)) {
10             alert("El nombre no tiene un formato válido. Ha de estar compuesto por entre 4 y 100 letras, números o guiones bajos.");
11             return;
12         }
13
14         let email_regex = new RegExp('^[a-zA-Z0-9_\\.-]+@[a-zA-Z0-9_-]+\\.([a-zA-Z0-9]{2,4})$');
15         let email = $('#form #email').val();
16
17         if(!email_regex.test(email)) {
18             alert("El correo electrónico no tiene un formato válido.");
19             return;
20         }
21
22         alert('La función de contribución está temporalmente desactivada. Por favor, inténtalo de nuevo más tarde.')
23     })
24 })
25
26
```

AJAX

Carga de datos de plataformas leídos en XML



Como los datos de cada una de las plataformas hardware mencionadas son similares en cuanto a formato, podemos minimizar la redundancia en el código HTML leyendo de manera dinámica los datos de un fichero XML en lugar de insertarlo directamente en el documento.

En este caso, el proceso se ha llevado a cabo empleando XMLHttpRequest, una clase de JavaScript que nos permite especificar los parámetros de una solicitud de documento XML, enviar dicha solicitud y procesar los datos una vez sea respondida.

Primeramente, especificamos que queremos obtener el contenido del fichero “platforms.xml” almacenado en localhost (127.0.0.1) empleando el método GET y de manera asíncrona.

A continuación, definimos qué hacer con los datos una vez se reciban, a través de la función onload(). En este caso, queremos comprobar el status de la petición, para verificar que se ha respondido correctamente, y después buscar, dentro de los datos de la respuesta, la información correspondiente a la plataforma deseada, e insertarla en la tabla asociada, modificando el código HTML de la misma.

Por último, simplemente enviamos la solicitud.

```
4 let request = new XMLHttpRequest();
5 request.open('GET', 'http://127.0.0.1/platforms.xml', true);
6 request.onload = function() {
7     if(this.status == 200) {
8         let data = this.responseXML.getElementsByTagName('platform');
9
10        let item = null;
11        for(let elem of data) {
12            let name = elem.getElementsByTagName('name')[0];
13            if(name.textContent == table.getAttribute('platform')) {
14                item = elem;
15                break;
16            }
17        }
18
19        if(item != null) {
20
21            let image_path = item.getElementsByTagName('image_path')[0];
22            table.innerHTML += '<tr><td colspan="2"></td></tr>';
23
24            let name = item.getElementsByTagName('name')[0];
25            table.innerHTML += '<tr><th>Nombre</th><td>' + name.textContent + '</td></tr>';
26
27            let manufacturer = item.getElementsByTagName('manufacturer')[0];
28            table.innerHTML += '<tr><th>Fabricante</th><td>' + manufacturer.textContent + '</td></tr>';
29
30            let release_date = item.getElementsByTagName('release_date')[0];
31            table.innerHTML += '<tr><th>Fecha de lanzamiento</th><td>' + release_date.textContent + '</td></tr>';
32
33            let discontinuation_date = item.getElementsByTagName('discontinuation_date')[0];
34            table.innerHTML += '<tr><th>Fecha de discontinuación</th><td>' + discontinuation_date.textContent + '</td></tr>';
35
36            let units_sold = item.getElementsByTagName('units_sold')[0];
37            table.innerHTML += '<tr><th>Unidades vendidas</th><td>' + units_sold.textContent + '</td></tr>';
38
39            let architecture_name = item.getElementsByTagName('architecture')[0].getElementsByTagName('name')[0];
40            let architecture_link = item.getElementsByTagName('architecture')[0].getElementsByTagName('link')[0];
41            table.innerHTML += '<tr><th>Arquitectura empleada</th><td><a href="' + architecture_link.textContent + '">' + architecture_name.textContent + '</a></td></tr>';
42        }
43        else {
44            table.innerHTML += '<tr><td colspan="2">Error cargando los datos.</td></tr>';
45            console.error("No se ha encontrado los datos de la plataforma indicada en el archivo XML.");
46        }
47    }
48    else {
49        table.innerHTML += '<tr><td colspan="2">Error cargando los datos.</td></tr>';
50        console.error("Se ha producido un error en la solicitud de datos sobre plataformas.");
51    }
52 }
53 request.send();
```

Carga de datos de arquitecturas leídos en JSON



De manera análoga al apartado anterior, podemos eliminar código HTML duplicado leyendo la información sobre las arquitecturas hardware de un archivo externo.

En este caso, vamos a emplear un fichero JSON, “architectures.json”, y un método distinto para la carga de datos: Fetch API.

El empleo de la API mencionada simplifica el proceso: para empezar, hemos de llamar a la función asíncrona `fetch()`, indicando el archivo cuyo contenido deseamos leer.

Como toda función `async`, devolverá un objeto `Promise` que representa la ejecución de dicha función, y que nos permitirá especificar a través de `then()` qué hacer con la respuesta cuando se disponga de ella. Llamaremos al método `json()` para parsear el resultado y obtener un objeto JavaScript que represente el contenido del fichero y, a continuación, buscaremos los datos de la arquitectura correspondiente.

Una vez hallados, de forma similar al caso de XML, añadimos dichos datos a la tabla modificando el HTML correspondiente a esta.

```
2
3 const table = document.getElementsByClassName("platform_data")[0];
4
5 fetch('http://127.0.0.1/architectures.json').then(response => response.json()).then(data => {
6
7     let item = null;
8     let name = null;
9     for(elem of Object.keys(data.Architectures)) {
10         if(elem == table.getAttribute('platform')) {
11             item = data.Architectures[elem];
12             name = elem;
13             break;
14         }
15     }
16
17     if(item != null) {
18         table.innerHTML += '<tr><td colspan="2"></td></tr>';
19         table.innerHTML += '<tr><th>Nombre</th><td>' + name + '</td></tr>';
20         table.innerHTML += '<tr><th>Fabricante</th><td>' + item["Manufacturer"] + '</td></tr>';
21         table.innerHTML += '<tr><th>Fecha de lanzamiento</th><td>' + item["Release date"] + '</td></tr>';
22         table.innerHTML += '<tr><th>Frecuencia máxima</th><td>' + item["Frequency"] + '</td></tr>';
23         table.innerHTML += '<tr><th>Tamaño de palabra</th><td>' + item["Word length"] + '</td></tr>';
24         table.innerHTML += '<tr><th>Tamaño de dirección</th><td>' + item["Direction length"] + '</td></tr>';
25         table.innerHTML += '<tr><th>Número de instrucciones</th><td>' + item["Instructions"] + '</td></tr>';
26         table.innerHTML += '<tr><th>Categoría</th><td>' + item["Category"] + '</td></tr>';
27     }
28     else {
29         table.innerHTML += '<tr><td colspan="2">Error cargando los datos.</td></tr>';
30         console.error("No se ha encontrado los datos de la arquitectura indicada en el archivo json.");
31     }
32
33 }).catch(err => {
34     table.innerHTML += '<tr><td colspan="2">Error cargando los datos.</td></tr>';
35     console.error("Se ha producido un error en la solicitud de datos sobre arquitecturas: " + err);
36 });
37
```