

# ÍNDICE

<b>I.</b>	<b>FASE DE ESTUDIO</b>	<b>3</b>
	<b>1. Gestión de alcance</b>	<b>3</b>
	1.1. Base de partida.... Estudio de comunidades virtuales	3
	1.2. Objetivo	10
	1.3. Público objetivo. Perfiles de usuario	10
	1.4. Contenidos y servicios	10
	1.5. Aspectos legales	12
	<b>2. Tecnologías involucradas</b>	<b>15</b>
<b>II.</b>	<b>GUIÓN MULTIMEDIA</b>	<b>16</b>
	<b>1. Estructura de navegación e interacción</b>	<b>16</b>
	1.1. Zona de usuarios	16
	1.2. Zona de administración	17
	<b>2. Descripción de pantallas</b>	<b>18</b>
	2.1. Descripción de pantallas de la zona de usuarios	18
	2.2. Descripción de pantallas de la zona de administración	28
	2.3. Elementos comunes	43
	<b>3. Lenguaje de color</b>	<b>43</b>
	3.1. Descripción de estilos estandar de la zona de usuarios	43
	3.2. Descripción de estilos estandar de la zona de administración	44
	3.3. Descripción de colores	44
	<b>4. Imágenes de ejemplo</b>	<b>45</b>
	4.1. Imágenes de ejemplo de la zona de usuarios	45
	4.2. Imágenes de ejemplo de la zona de administración	47
<b>III.</b>	<b>IMPLEMENTACIÓN DE LA PLATAFORMA</b>	<b>55</b>
	<b>1. Definición de la base de datos</b>	<b>55</b>
	1.1. Presentación de la base de datos Oceanográfico	55
	1.2. Descripción de los atributos de cada relación	56
	1.3. Esquema relacional de la base de datos	57
	<b>2. Implementación de la zona de usuarios</b>	<b>59</b>
	2.1. index.php	59
	2.2. login.php	62
	2.3. crear_conexion_bd.php	63
	2.4. registro.php	63
	2.5. enviando_datos.php	67
	2.6. acreditado.php	69
	2.7. mensajes.php	71
	2.8. mensajes2.php	77
	2.9. comentario_escribir.php	78
	2.10. comentario_escribir_envio.php	80
	2.11. comentario_leer.php	80
	2.12. acreditado2.php	81
	2.13. usuario_modificar_usuario_2.php	88
	<b>3. Implementación de la zona de administración</b>	<b>91</b>
	3.1. admin.php	91
	3.2. administrador2.php	92

3.3.	administrador3.php	93
3.4.	administrador_opcion_crear_administrador.php	95
3.5.	administrador_opcion_listar_administradores.php	97
3.6.	administrador_opcion_borrar_administrador.php	99
3.7.	administrador_creacion_cuenta.php	101
3.8.	administrador_opcion_listar_usuarios.php	101
3.9.	administrador_opcion_listar_usuarios_impresion.php	106
3.10.	administrador_opcion_listar_puntuaciones.php	108
3.11.	administrador_opcion_listar_puntuaciones_impresion.php	112
3.12.	administrador_opcion_modificar_usuario.php	115
3.13.	administrador_modificar_usuario.php	117
3.14.	administrador_modificar_usuario_2.php	123
3.15.	administrador_opcion_borrar_usuario.php	124
3.16.	administrador_borrar_usuario.php	126
3.17.	administrador_opcion_crear_bases_de_datos.php	126
3.18.	creacion_base_de_datos.php	128
3.19.	administrador_opcion_destruir_bases_de_datos.php	130
3.20.	destruir_base_de_datos.php	132
<b>4.</b>	<b>Implementación de los mapas</b>	<b>134</b>
4.1.	mapa_sin_usuarios.swf	134
4.2.	mapa_usuario_activo.swf	135
4.3.	juego1.php	135
4.4.	juego2.php	136
4.5.	juego3.php	136
4.6.	ranking1.php	136
4.7.	ranking2.php	138
4.8.	ranking3.php	140
<b>IV.</b>	<b>DESARROLLO E IMPLEMENTACIÓN DE LOS JUEGOS</b>	<b>142</b>
<b>1.</b>	<b>Scott, el pájaro que no sabe volar</b>	<b>142</b>
1.1.	Fase de estudio	142
1.2.	Fase de implementación	146
<b>2.</b>	<b>Igor, el pingüino defensor</b>	<b>152</b>
2.1.	Fase de estudio	152
2.2.	Fase de implementación	153
<b>3.</b>	<b>Bill, el cazador de estrellas</b>	<b>160</b>
3.1.	Fase de estudio	160
3.2.	Fase de implementación	162
<b>V.</b>	<b>BIBLIOGRAFÍA</b>	<b>181</b>
	<b>ANEXO I. FICHAS DE LOS JUEGOS ANALIZADOS DURANTE EL ESTUDIO DE LAS COMUNIDADES VIRTUALES</b>	<b>182</b>
	<b>ANEXO II. GUÍA DE INSTALACIÓN</b>	<b>191</b>

# I. FASE DE ESTUDIO

## 1. Gestión de alcance

### 1.1 Base de partida... Estudio de comunidades virtuales

Nuestro proyecto consiste en el desarrollo de videojuegos orientados a la comunicación publicitaria en la web. Para ello hemos desarrollado una plataforma de juegos, creados en Flash, basada en el Oceanográfico de la Ciudad de las Artes y las Ciencias, desarrollando un juego para cada una de las zonas que forman el parque. Esta plataforma irá orientada hacia un público infantil, tal y como explicaremos detenidamente más adelante.

Para ello, previamente hemos realizado un estudio sobre las comunidades virtuales de juegos existentes en la red.

Existen multitud de comunidades virtuales de juegos, por lo que podemos observar existe una multitud de tendencias dentro de este amplio abanico. Queda claro que uno de los logros de la red ha sido crear un punto de encuentro en el que no importa el lugar desde el que se conecta el internauta. Este es uno de los motivos que ha propiciado el éxito de los juegos on-line. Se trata de una diversión que consiste en jugar simultáneamente, desde cualquier lugar del planeta, a un juego concreto. Al final, pueden consultarse unas extensas estadísticas que recogen todo tipo de datos de cada participante. Estas comunidades virtuales son numerosas y para suscribirse, normalmente de manera gratuita, tan sólo hay que rellenar un formulario que, habitualmente, podemos encontrarlo en la propia web. Después, lo lógico es tenerse que bajar un programa determinado para hacer posible la conexión en la que se realizará el juego.

Por otro lado, muchas comunidades virtuales de juegos se han creado con la intención de publicitar unos productos comerciales que la misma marca dueña de la comunidad ofrece en las tiendas. A este tipo de juegos, encargados de publicitar un producto se les conoce por el nombre de advergames. El término “advergame” deriva de la fusión de las palabras “advertising” (publicidad) y “videogame” (videojuego). Según cuenta la historia oficial, sus inventores son Dan Ferguson y Mike Bielinski, dos jóvenes norteamericanos que, en 1998, para demostrar su capacidad como diseñadores web, crearon un videojuego que tenía como protagonista al ex presidente norteamericano Bill Clinton y lo hicieron circular por correo electrónico. Fue un éxito: los usuarios de Internet se mandaban el juego por e-mail, en un ejemplo perfecto de marketing contagioso. Hoy, Ferguson y Bielinski se dedican a hacer “advergimes” para el gigante de las telecomunicaciones Nokia.

Según la consultora “Forrester Research”, de aquí al 2005, sólo en Europa, los negocios vinculados a los “advergimes” generarán unos 5.000 millones de euros. Un boom que se explica así: el “advergame” hipnotiza al consumidor y lo vuelve fiel a una marca. Un estudio de la Universidad de Michigan reveló que la memorización de información cuando se juega a un videojuego es 10 veces mayor que cuando se mira televisión. Una marca presente en un videojuego, entonces, se graba en la memoria con mayor velocidad y más a fondo que si aparece en un anuncio de televisión. Muchas veces, para jugar, es necesario registrarse. Así, las compañías adquieren una base de datos de clientes a los que después les pueden mandar mensajes mucho más

acordes a sus gustos y necesidades. Los “advergames”, además, prenden entre los consumidores particularmente apetecibles, como son los jóvenes y los profesionales. De hecho, dicen los expertos, los picos de acceso se registran a la hora del almuerzo. Lo cual significa que los “advergames” se juegan en la oficina, en un momento de descanso. Por ese motivo es que atrajeron la atención de los grandes inversores. Estos empiezan a prestarle atención a un medio que, hasta ahora, estaba reservado a los banners de los casinos online y los sitios para adultos. Lorenzo Montagna, experto en marketing, explica: “Un buen ejemplo es la firma Ferrero. Desde hace dos años viene insertando en sus clásicos bombones de chocolate un código que les permite a los chicos jugar online. Hoy ese sitio está disponible en 22 idiomas.

Así pues, hemos realizado el análisis de diferentes comunidades virtuales que presentaban juegos publicitarios, y a partir de ahí hemos obtenido las siguientes conclusiones:

Una de las páginas webs que más nos ha llamado la atención ha sido la de la firma deportiva Nike <http://www.nike.com> , por el hecho de que no solo era una única comunidad de juegos, sino que englobaba varias comunidades diferentes, agrupadas por deportes. Nos hemos centrado en la sección de fútbol, que contenía seis juegos. Uno de los que más llaman la atención es “Houseball”, el cual combinaba elementos de balompié con el clásico juego del pinball (los tacos eran zapatillas, la bola un balón, y jugadores de fútbol aparecían por el escenario en el centro del cual aparecía el símbolo de la marca). Por otro lado, este juego era el único de la comunidad que no exigía al usuario registrarse.



De manera negativa, encontrábamos el juego “V.I.P.”, el cual presentaba un proceso de registro excesivamente largo y una alta tasa de injugabilidad.

La firma Nabisco (conocida marca de galletas), ofrece una gran cantidad de juegos en su página web, divididos en tres categorías (arcade, deporte, puzzle). Destaca el hecho de que los juegos más populares aparecen en un top games, en el cual aparecen los seis juegos más populares independientemente de la categoría a la que pertenecen. Unos de los juegos más destacables son el “Mini Mini Golf” y el “Mah Jongg”. El primero de ellos es un atractivo y adictivo juego de mini golf, en el cual jugamos en reducidos escenarios que se pueden encontrar en cualquier hogar (por ejemplo, una mesa de escritorio) y sobre los cuales aparecen cajas y galletas de los productos ofertados por esta marca. Es un claro ejemplo de advergame, ya que combina a la

perfección elementos propios de un juego clásico con la marca que patrocina.



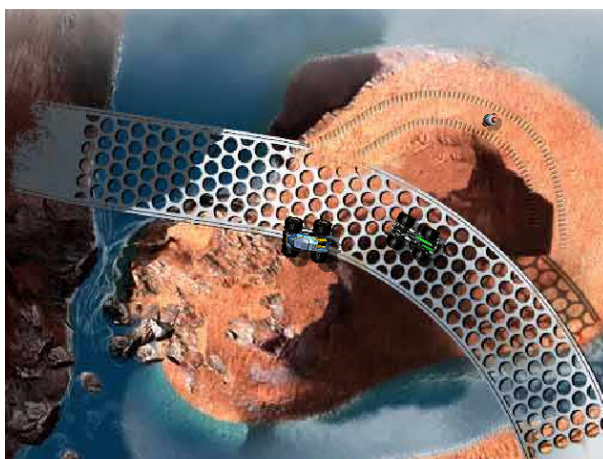
El “Mah Jongg” es mucho más simple. Es la sencilla recreación del clásico juego de mesa japonés, donde los símbolos que aparecen sobre las fichas han sido sustituidos por imágenes de galletas y logotipos de la marca, introduciendo de esta manera tan hábil, el componente publicitario dentro del videojuego.



También encontramos juegos de mayor sencillez. Uno de ellos era “Dunking Game” el cual consistía en ir cazando galletas con un vaso de leche antes de que cayeran al suelo.



Otra de las comunidades que nos llamaron la atención fue la de la conocida marca de juguetes Lego <http://www.lego.com> . En esta comunidad encontramos tres juegos en la pantalla principal, destacando uno de ellos por encima de los demás, y a través de un enlace podemos llegar a una gran cantidad de juegos. El juego al que le daban mayor importancia era el llamado “Lego Racer”, un juego de carreras de coches a la antigua usanza, pero que no tenía mucho que ver con la marca, ya que no encontrábamos ninguna referencia a los juguetes que vende la empresa.



Este sería un ejemplo de un juego advergame que no cumple su objetivo primordial, ya que no aparece ninguna referencia al producto y podría encajar perfectamente en cualquier portal en el que aparezcan juegos en flash sin ningún tipo de afiliación comercial.

También encontramos juegos como el “World Builder”, consistente en construir escenarios con productos Lego. Este juego cumple claramente el objetivo propuesto de introducir los productos al cliente en el juego, pero su excesiva complejidad y su falta de un objetivo claro lo convierten en un juego aburrido.



La empresa Nestle nos ofrece en su web <http://www.nestle.es> una serie de juegos. Lo primero que observamos fue la asociación existente entre Nestle y Dreamworks, estando presente en todo momento los personajes de Shrek en la web de Nestle, a modo de publicidad de la película. Por otro lado, analizando detenidamente los juegos encontramos juegos como “Fotoquick de Quicky”, presentado por el

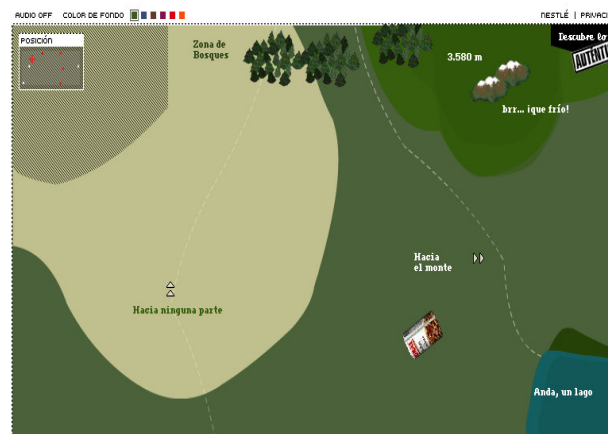


conocido personaje Quicky, mascota de Nesquick, en un juego falto de diversión pero aun así con cierta originalidad. El juego consistía simplemente en ir haciendo fotos con un objetivo a un personaje evitando fotografiar focos y otros elementos no acordes con el escenario.



Los elementos de la marca que aparecían en el juego eran la presentación realizada por la mascota de la firma y los colores utilizados, los cuales son los que componen el envase del producto publicitado.

A través de Nestle también podemos acceder a otro de sus productos como es la Fabada Litoral, la cual presenta una web muy original, con una navegabilidad compleja pero divertida. En esta web el usuario es representado en forma de lata de conservas, la cual se desplazará a lo largo de un mapa que representa Asturias y encontrará juegos sencillos a lo largo de este. Destaca por encima de ellos el juego de preguntas y respuestas, todas ellas relativas a la historia del producto, donde se nos ofrece para cada pregunta dos opciones de respuesta, una de ellas la correcta y la otra absolutamente incoherente, con lo cual todo aquel que juegue al juego aprenderá más sobre la marca. Es un claro ejemplo de juego advergame ya que hace total hincapié sobre el producto, pero que puede llegar a agotar al usuario debido precisamente al hecho de ser tan monotemático y la poca adicción que presentan sus pruebas.



Para finalizar estuvimos observado una comunidad virtual de jugadores de juegos en Flash sin ningún tipo de afiliación comercial clara. Es decir, los juegos simplemente se desarrollaron con el único objetivo de entretener al usuario, sin ningún

tipo de intención de publicitar un producto. Una de las webs que más nos gustó fue <http://www.minijuegos.com>, en la cual aparecían juegos de todo tipo, ordenados por categorías (acción, deportes, clásicos, coches, estrategia, etc.) así como un listado de los últimos juegos incorporados a la web, otro listado con los juegos de mayor popularidad y otro listado con los mejores juegos, los cuales han sido puntuados por los usuarios registrados. También se nos ofrece la posibilidad de registrarnos en el portal para poder puntuar estos juegos.



Tras realizar el análisis de las comunidades virtuales, hemos observado la gran importancia que tienen los juegos a la hora de comercializar un producto en su web, ya que es una buena manera de mantener entretenido al usuario mientras de manera indirecta se le intenta despertar el interés por el producto. Una buena forma para que este objetivo se cumpla del modo más exitoso posible es la representación de elementos del entorno del juego con elementos característicos de la marca y el producto, así como la importancia de que el juego presente un amplio grado de atracción y adicción hacia el usuario. No es recomendable la realización de juegos con un argumento muy complicado y de un manejo poco intuitivo así como excesivamente difícil.

Tampoco es recomendable el hecho de sobresaturar el juego con elementos de la marca comercial de modo que el juego se acerque más a ser un anuncio interactivo que a ser un entretenimiento.

En cuanto a la duración de las partidas de los juegos, cabría comentar que en la gran mayoría son partidas relativamente cortas, que rondan la duración entre uno y tres minutos a lo sumo, con lo cual, se intenta despertar el interés del jugador hacia el juego durante los primeros instantes de este. Un juego, con un largo proceso de aprendizaje y con una línea argumental excesivamente elaborada puede provocar la pérdida de interés de la mayor parte de los usuarios, por lo que el objetivo principal del juego (atraer la atención del usuario) quedaría incumplido.

Los tiempos de carga de estos juegos deben ser muy pequeños para que posible usuario no cierre el juego durante su carga. Por ello, es más importante optimizar el juego para una carga rápida que para una espectacularidad gráfica y sonora.

Los juegos suelen disponer de efectos de sonido pero no de música ambiental, ya que la inserción de banda sonora aumenta considerablemente el tamaño del juego en



bytes, y en consecuencia el tiempo de carga.

El algunos juegos sería interesante el hecho de que apareciera una opción de modo de dificultad. Aquí podríamos definir tres tendencias. La primera de ellas sería la más simple y utilizada. Es el hecho de poder escoger la dificultad antes de comenzar a jugar (dificultad fácil, mediana o difícil...). El otro de modo, también muy utilizado, y en ocasiones en combinación con el anterior, se da en los juegos cuya dificultad incrementa conforme va incrementando el tiempo o a medida que el jugador va completando niveles o pantallas. Por último, sería interesante que en las comunidades virtuales en las que se permita el registro de usuarios, la dificultad venga dada por el historial del jugador. Esto quiere decir que si se observa que un jugador domina los niveles básicos y más sencillos, se le haga jugar inicialmente desde un nivel de dificultad más complicado.

En algunas ocasiones se permite a dos usuarios jugar al mismo tiempo desde un mismo equipo. Esto es conocido coloquialmente como “jugar a dobles”. Esto no debería ser permitido ya que rompe con el concepto de un jugador que accede a una comunidad virtual y juega a los juegos, ya que únicamente uno de los dos jugadores podría estar conectado como usuario acreditado en ese momento en ese equipo, y la puntuación obtenida no vendría dada exclusivamente por el jugador registrado sino que intervendría un tercer elemento o elemento extraño en forma de usuario invitado. Sin embargo, si que es interesante la posibilidad de jugar de manera remota contra otros usuarios conectados al mismo tiempo al servidor de la comunidad virtual.

Otro de los aspectos que cabe destacar es la existencia de un ranking con las mejores puntuaciones para cada uno de los juegos, pero en ningún caso los usuarios se benefician de ninguna ventaja (a excepción del orgullo personal). En las webs que no exigen registro se solicita el nombre únicamente al lograr una puntuación mayor que cualquiera de las existentes en el actual ranking. En las que requieren registro el nombre del usuario viene dado por el usuario activo. Consideramos que se debería dar mayor importancia a las puntuaciones, proporcionando ventajas a los usuarios con mayores puntuaciones y partidas jugadas, ofreciéndoles el acceso a juegos exclusivos, nuevos niveles y regalos de la marca a los mejores jugadores. Esta opción es un aliciente mayor para que el usuario juegue a los juegos, ya que a parte de entretenimiento puede obtener un beneficio material.

Centrándonos detenidamente en lo que son las comunidades virtuales en si, cabría destacar el hecho de la cantidad de juegos que ofrece la comunidad. Existen comunidades virtuales que se centran en ofrecer pocos juegos pero de muy alta calidad, mientras que por otro lado existe la tendencia de ofrecer la mayor cantidad de juegos posibles, cuando la mayor parte de estos no llegan al mínimo de calidad exigido. Por lo tanto concluimos en que es más correcto centrarse en la calidad antes que en la cantidad.

Es importante también mostrar las nuevas incorporaciones de juegos a la comunidad al usuario. Esto se puede hacer de muchas maneras. Una de ellas puede ser mostrar un enlace al juego desde la página principal de la comunidad o aumentando el tamaño o resaltando el enlace del juego nuevo dentro del listado de juegos disponibles. También es posible mantener una lista de correo desde la que se informe a los usuarios registrados de la incorporación de nuevos juegos a la comunidad.

En muchas comunidades hemos observado que introducen un listado con los juegos más populares. Esto es un arma de doble filo, ya que por un lado el usuario puede acceder de manera directa a los que, en teoría, son considerados mejores juegos por los usuarios de la comunidad, pero por otro lado puede provocar la omisión de los juegos no incluidos en este top o ranking, ya que todo nuevo usuario accederá de manera directa a estos juegos dándoles más votos que a los otros y además ignorando al resto y prejuzgando la calidad de los otros juegos en base a los juegos contenidos en el ranking.

## **1.2 Objetivo**

El objetivo que pretendemos cumplir es la creación de una comunidad virtual de jugadores donde estos jugadores aprendan sobre el oceanográfico y los animales que allí se encuentran, todo ello en un entorno visualmente atractivo para los niños y que a la vez sea adictivo, y al mismo tiempo conforme vayan jugando sus conocimientos también avancen. Por otro lado, ha de conseguirse que los niños que no hayan ido al Oceanográfico despierten su interés en ir. Con esto lograremos que los usuarios acudan al parque con los beneficios económicos que esto supone.

## **1.3 Público objetivo. Perfiles de usuario**

El público objetivo de nuestra aplicación serán los niños con una edad comprendida entre seis y doce años, que presenten interés por los animales expuestos en el oceanográfico. Estos niños estarán tutorizados por un adulto, el cual les acompañará durante el proceso de registro, ya que es posible que el niño tenga problemas a la hora de realizarlo.

Los niños que jueguen a los juegos de nuestra plataforma podrán hacerlo a través de la página web de la plataforma creada expresamente para su uso en internet.

Dentro de nuestra aplicación encontraremos tres tipos de usuario diferentes. El primero de ellos y el más básico será el usuario no registrado o invitado. Podrá leer las descripciones de los juegos contenidos en la plataforma y será invitado a registrarse. También tendrá la oportunidad de leer los mensajes editados por los usuarios registrados que aparecen en el tagboard situado en la parte izquierda de la pantalla, así como tendrá oportunidad de ver las mejores puntuaciones de los distintos juegos, y un mensaje aleatorio sobre la plataforma. También tendrá oportunidad de escribir un e-mail al administrador de la plataforma. El segundo tipo de usuario será el usuario registrado, el cual dispondrá de acceso a todos los juegos y tendrá opción de aparecer en las tablas de puntuaciones más altas de dichos juegos. También podrá modificar sus datos de registro pinchando en la opción pertinente, así como podrá escribir mensajes en el tagboard. Por último, tendrá la oportunidad de cerrar su sesión. En último lugar encontramos el administrador, también conocido por superusuario, que tendrá plenos poderes sobre la gestión de la plataforma.

## **1.4 Contenidos y servicios**

Nuestra plataforma estará compuesta por tres zonas claramente diferenciadas. La primera de ellas será el encabezado que únicamente contendrá una imagen con el logotipo del Oceanográfico. La segunda de ellas será el menú de opciones y se situará justo debajo del encabezado. Dependiendo del usuario activo este menú mostrará

diferentes opciones. En el caso de ser un usuario invitado el que accede a la plataforma, nos encontraremos con una opción de acceso a una cuenta registrada, y en caso de no tenerla, poder crear una cuenta de usuario nueva. La otra opción que aparecerá en este menú será la opción de “Contacta con nosotros”, en la cual se podrá mandar un e-mail al administrador. También tendremos acceso a un mapa del Oceanográfico en el cual aparecerán los juegos disponibles en la web y al pinchar sobre cada uno de ellos poder leer la descripción del juego, una invitación a registrarse y el ranking de mejores puntuaciones obtenidas por los usuarios registrados.

En la parte izquierda de la pantalla aparecerá un tagboard que mostrará los últimos 10 comentarios realizados por usuarios registrados. Un usuario invitado únicamente podrá leer estos mensajes. A la derecha del mapa encontraremos las tablas con las mejores puntuaciones de cada uno de los juegos.

Si se trata de un usuario registrado, tendremos el mismo menú de juegos pero con la diferencia de que al pinchar sobre cada uno de los juegos, obtendremos su descripción, el ranking y en lugar de ser invitados a registrarnos, se nos ofrecerá la posibilidad de jugar a ellos. Estos juegos se cargarán en un navegador nuevo. También habrá una sección de administración de la cuenta del usuario, donde este podrá modificar sus datos personales. En cuanto al tagboard, se le permitirá al usuario activo publicar un mensaje en él simplemente pinchando sobre la opción “escribir comentario”. Por último, el usuario tendrá la posibilidad de cerrar su cuenta accediendo después al área de invitados.

Finalmente, encontramos al administrador. Este no accederá la web desde la página principal, sino que accederá desde una dirección php especial que únicamente él conoce y que una vez introducido su login y password, se le permitirá acceder a las siguientes opciones: Crear administrador, Listar Administradores, Borrar Administrador, Listar usuarios, Listar puntuaciones, modificar usuario, borrar usuario, crear bases de datos y eliminar bases de datos.

La opción Crear administrador permite crear un nuevo usuario administrador de la plataforma el cual tendrá plenos privilegios sobre las opciones de esta zona de administración.

Listar administradores nos mostrará el listado de los usuarios acreditados como usuario administrador de la plataforma.

Borrar administrador nos permitirá eliminar de manera permanente a un usuario registrado como usuario administrador.

En la opción listar usuarios podremos acceder a los datos de cada usuario y modificarlos. También dispondremos de la opción de eliminar a cualquier usuario. Dentro de esta opción también se nos permite la posibilidad de imprimir en papel dicho listado.

Listar puntuaciones nos permite ver las mejores puntuaciones obtenidas en cada uno de los juegos, así como el número total de accesos a dichos juegos. Con esto podremos comprobar el grado de atracción que presenta el juego una vez leída su descripción. Al igual que en la opción listar usuarios, se nos permitirá imprimir estas puntuaciones. La opción Modificar usuario nos permitirá modificar cualquiera de los campos que componen los datos personales del usuario.

Borrar usuario nos permitirá eliminar por completo a un usuario registrado en la plataforma.

Y por último las opciones crear bases de datos y destruir bases de datos nos permitirán por un lado generar las bases de datos necesarias por el sistema (estas bases de datos se

generarán totalmente vacías), mientras que la opción nos eliminará todas las bases de datos de la plataforma.

## 1.5 Aspectos legales

En el desarrollo de la plataforma hay una serie de aspectos legales que hemos de tener en cuenta y que no podemos obviar, principalmente estas conciernen a la protección de datos de los usuarios, la polémica ley de Ley de Servicios de la Sociedad de la Información y Comercio Electrónico y lo referente a los videojuegos ya que hay una serie de normas que debemos cumplir y unas informaciones que debemos ofrecer. A través de la plataforma el usuario, durante el registro, nos facilita unos datos personales, sobre los cuales tenemos unas obligaciones y una serie de normas que cumplir, todas ellas vienen regidas por la Ley de Protección de Datos.

En primer lugar en lo referente a la Ley de Protección de Datos, más conocida como LOPD, nos hemos regido por :

- Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal (Título VI con rango de ley ordinaria).
- Real Decreto 428/1993, de 26 de marzo, por el que se aprueba el Estatuto de la Agencia Española de Protección de Datos.

La función de LOPD es velar por el cumplimiento de la legislación sobre protección de datos y controlar su aplicación, en especial en lo relativo a los derechos de información, acceso, rectificación, oposición y cancelación de datos.

Una vez introducida muy brevemente la LOPD vamos a centrarnos en lo que nos afecta a la creación de nuestra plataforma de videojuegos ,el uso que podemos dar de los datos de nuestros clientes ,las precauciones que debemos tomar y una serie de obligaciones legales.

Estas obligaciones legales serian referentes a la Calidad de datos, Deber de información, Tratamiento y cesión de los datos, Deber de colaboración con la agencia, Transferencias internacionales, Deber de guardar secreto, Atención de los derechos de los ciudadano, Inscripción de ficheros y Medidas de seguridad.

Nosotros como creadores de la plataforma y ser nosotros quienes tratamos los datos tenemos una serie de funciones que cumplir tales como

- Emitir autorizaciones previstas en la Ley.
- Requerir medidas de corrección.
- Ordenar, en caso de ilegalidad, el cese en el tratamiento y la cancelación de los datos.
- Ejercer la potestad sancionadora.
- Recabar ayuda e información que precise.
- Autorizar las transferencias internacionales de datos

Al estar centrados en materia de telecomunicaciones tendríamos la función de tutelar los derechos y garantías de los abonados y usuarios en el ámbito de las comunicaciones electrónicas, incluyendo el envío de comunicaciones comerciales no

solicitudes realizadas a través de correo electrónico o medios de comunicación electrónica equivalente.

Brevemente esto sería una pequeña explicación de los aspectos legales que tendríamos que tener en cuenta en torno a la protección de datos.

Respecto a la normativa sobre el videojuego en sí, hay que tener en cuenta las normativas existentes sobre videojuegos.

Desde Junio del 2003 existe una nueva normativa reguladora con el objetivo que los consumidores cuenten con toda la información disponible sobre estos productos para poder elegir los más adecuados a la edad del usuario. A partir de ahora se establecen más edades intermedias y se introducen unos iconos descriptores de contenidos.

El nuevo código de autorregulación denominado PEGI (Pan European Game Information) es válido en 16 países europeos: los Quince (a excepción de Alemania) más Noruega y Suiza. El objetivo que se persigue es que los consumidores, en especial los padres y los educadores, cuenten con información suficiente sobre los videojuegos que están a la venta en España, o en cualquier otro país europeo adscrito al código, para poder elegir los productos más adecuados a la edad del usuario, limitando de esta manera la exposición de los niños a contenidos inadecuados.

Para los juegos on-line como el que vamos a realizar no hay una normativa clara, ya que esta normativa afecta a los juegos para Pc, videoconsolas o para dispositivos móviles.

No vamos a encontrar ningún problema con nuestros juegos que al tratarse de juegos educativos para niños, ni emplearemos lenguaje soez, ni violencia, ni contenidos sexuales que pudieran ocasionar alguna queja.

Otro de los aspectos legales a tener en cuenta es la polémica Ley de Servicios de la Sociedad de la Información y Comercio Electrónico, más conocida como LSSI. Esta ley desde su aparición en el año 2001 ha causado gran controversia ya que afecta a la gran mayoría de usuarios de Internet.

Centrándonos en nuestra plataforma y en la información que vamos a manejar con los datos de los usuarios así como la creación de la página web, estamos sujetos a la LSSI ya que realizamos actividades económicas a través de medios telemáticos como puede ser el correo electrónico y la dirección y gestión de sus negocios esté centralizada en España.

Nuestra plataforma dispondrá de información sobre actividades, productos y servicios que ofrece el Oceanográfico, y aunque no puedan controlarse a través de la página web deberemos facilitar, a través de nuestra plataforma, un conjunto de información mínima sobre su denominación, domicilio y actividad, y asegurarse de que la publicidad de otras empresas que, en su caso, figure en la plataforma pueda distinguirse claramente del contenido propio de la página y esté identificado el anunciante.

No sería necesario necesaria ninguna autorización y podríamos poner a disposición de nuestros usuarios, los servicios, en nuestro caso juegos, para que puedan utilizarlos sin ningún tipo de problema.

Nuestra plataforma dispondrá de banners acerca de la Cac por tanto, conforme a la normativa de la LSSI tendremos que tener una información en la pagina web:

La información que debe facilitarse es la siguiente:

- a. Su nombre, NIF, domicilio (indicando, al menos, la localidad y provincia de residencia) y dirección de correo electrónico.
- b. Los códigos de conducta a los que, en su caso, esté adherido y la manera de consultarlos electrónicamente.

La publicidad que se muestre en la página web deberá ajustarse a lo establecido en la Ley, la cual obliga a identificar al anunciante y a presentarla de manera claramente distinguible de los contenidos no publicitarios de la página. Así mismo, deberán respetarse las restantes normas sobre publicidad, recogidas en otras leyes.

El artículo 10 de la Ley indica que la información sobre el prestador de servicios y su actividad ha de ponerse a disposición de los usuarios por medios electrónicos, de forma permanente, fácil, directa y gratuita. Cuando los servicios se prestan a través de una página en Internet, bastará con incluir en ella esa información de manera que ésta sea accesible en la forma indicada. Esta información estara disponible en la pagina principal.

El tema referente al envío de correos electrónicos informando acerca del Oceanográfico, ya sean promociones, ofertas, novedades esta tipificado como actividades comerciales a través del correo electrónico y tendremos una serie de obligaciones.

La Ley permite la realización de comunicaciones comerciales mediante el uso de Internet u otros medios electrónicos, siempre que puedan identificarse como tales y a la persona o empresa en nombre del cual se realizan o anunciante.

Se permite el envío de mensajes publicitarios o comerciales por correo electrónico a aquellos usuarios que previamente lo hubieran autorizado o lo hubieran solicitado de forma expresa. No obstante, se permite el envío de comunicaciones comerciales a aquellos usuarios con los que exista una relación contractual previa, en cuyo caso el proveedor podrá enviar publicidad sobre productos o servicios similares a los contratados por el cliente. Estas reglas son también aplicables al envío de mensajes publicitarios por otros medios de comunicación electrónica individual equivalente, como el servicio de mensajería de la telefonía móvil.

En todo caso, el prestador deberá ofrecer al destinatario la posibilidad de oponerse al tratamiento de sus datos con fines promocionales, tanto en el momento de recogida de los datos como en cada una de las comunicaciones comerciales que le dirija.

La Ley obliga, además, a los prestadores de servicios a habilitar procedimientos sencillos y gratuitos para que los destinatarios puedan revocar el



consentimiento que hubieran prestado, así como a facilitar información accesible por vía telemática sobre dichos procedimientos.

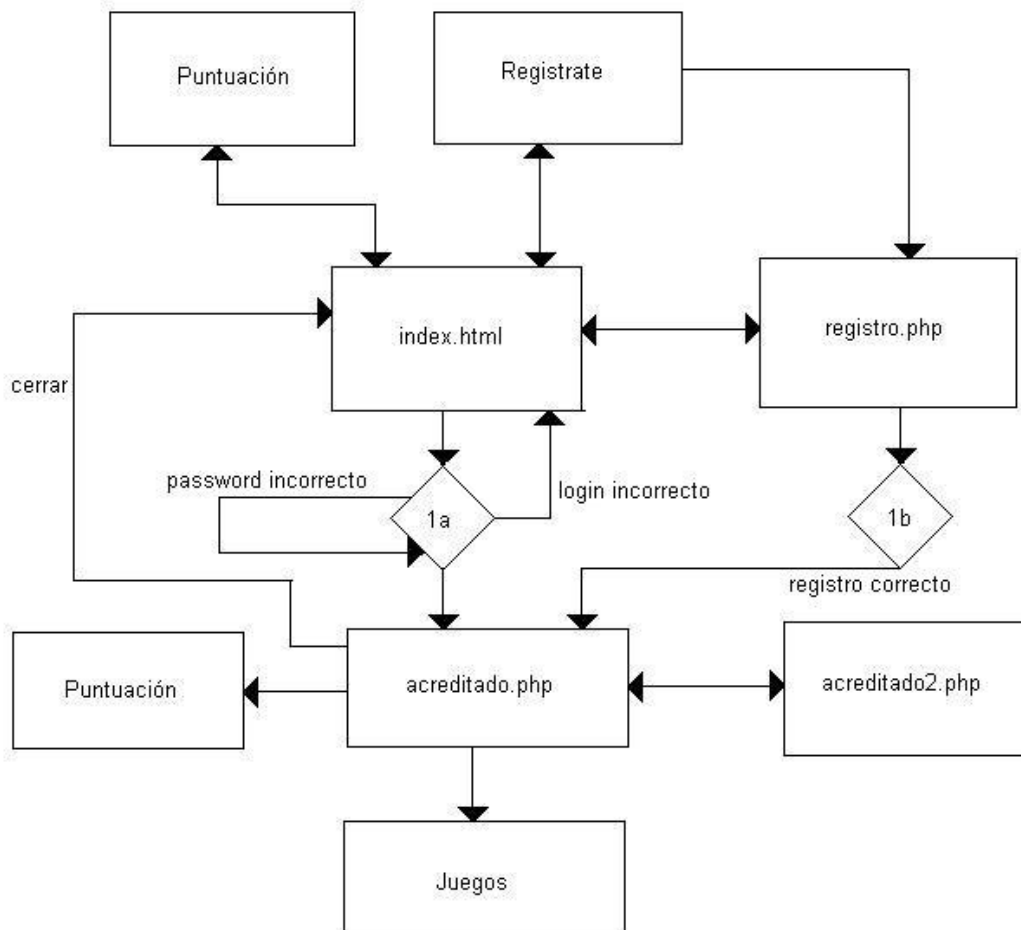
## **2. Tecnologías involucradas**

Para la realización de la plataforma necesitaremos conocer HTML, PHP y SQL. Para el desarrollo de los juegos utilizaremos la suite Macromedia, concretamente la herramienta Flash MX 2004 y su lenguaje de programación ActionScript. También requeriremos de conocimiento de algún programa de edición de imágenes como por ejemplo Adobe Photoshop y de edición de sonido como Cool Edit Pro 2.

## II. GUIÓN MULTIMEDIA

### 1. Estructura de navegación e interacción

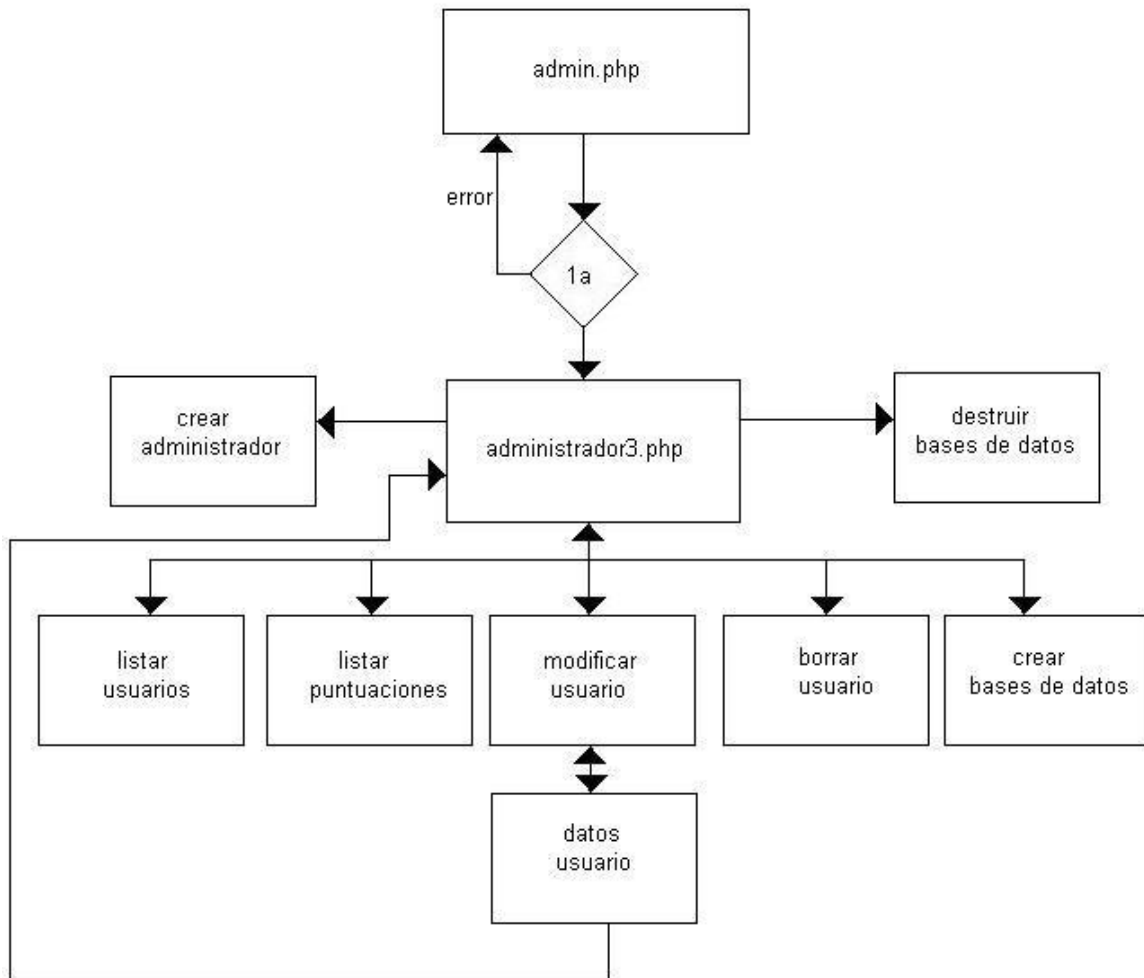
#### 1.1. Zona de usuarios



**1a:** comprueba si el el usuario y la contraseña son correctos

**1b:** comprueba que todos los datos de registro son correctos

## 1.2. Zona de administración

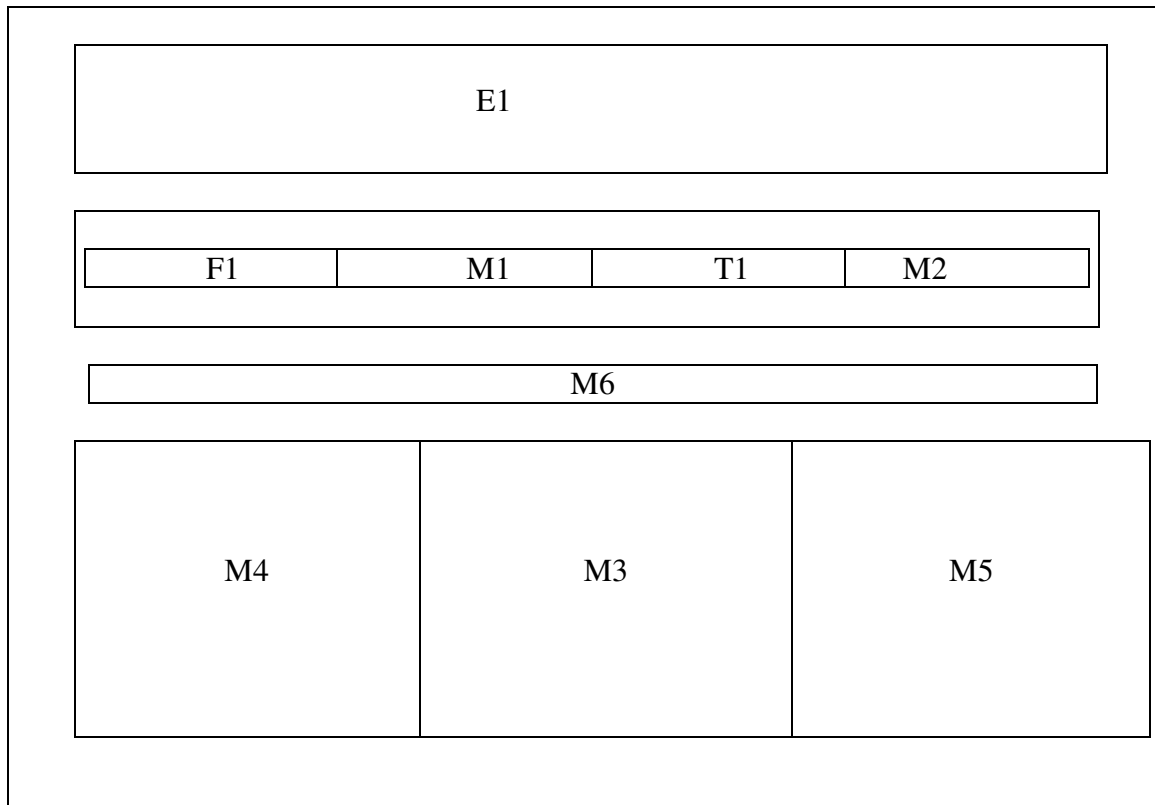


**R1:** En todo momento desde cualquier sección se puede saltar al resto  
**1a:** comprueba si el el usuario y la contraseña son correctos

## 2. Descripción de pantallas

### 2.1. Descripción de pantallas de la zona de usuarios

index.html	Pantalla de inicio de la plataforma.
------------	--------------------------------------



<b>F1</b>	
Nombre	F1login
Descripción	Formulario de entrada a la plataforma. Se solicitará el nombre de usuario y su contraseña.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

<b>M1</b>	
Nombre	M1CrearUsuario
Descripción	Enlace a la página que permite que un usuario se de de alta en la plataforma.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>T1</b>	
Nombre	T1TextoError
Descripción	Mensaje de texto que únicamente aparecerá en caso de la existencia de algún error durante el uso de la plataforma.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Heredado.

<b>M2</b>	
Nombre	M2EnviarCorreo
Descripción	Enlace que permite enviar un e-mail al administrador.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

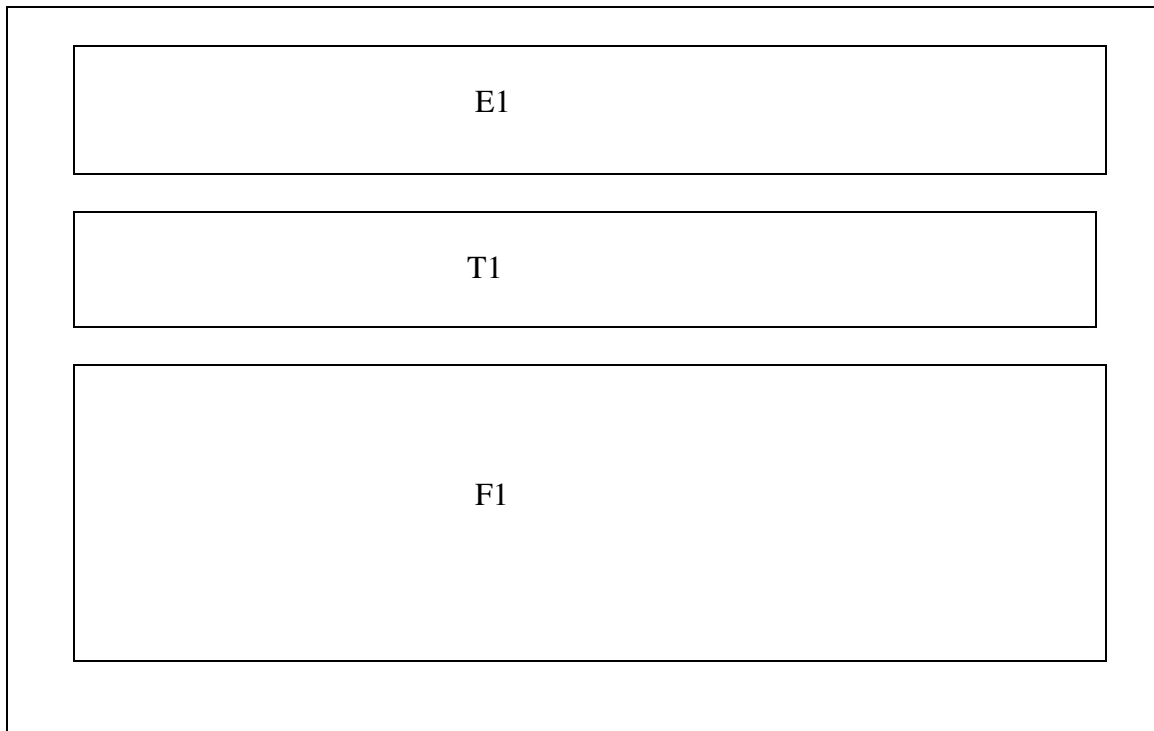
<b>M3</b>	
Nombre	M3MapaNoAcreditado
Descripción	Mapa del oceanográfico que posibilitará ver las ventajas de registrarse en la plataforma.
Apariencia	Específico.
Comportamiento	Específico.
Origen de datos	URL.

<b>M4</b>	
Nombre	M4TagBoard
Descripción	Listado de los últimos mensajes escritos por los usuarios registrados de la plataforma.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

<b>M5</b>	
Nombre	M5Rankings
Descripción	Listado con las mejores puntuaciones de los tres juegos.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

<b>M6</b>	
Nombre	M6TextoAleatorio
Descripción	Mensaje aleatorio sobre el estado de los usuarios de la plataforma.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

registro.php	Página de registro de nuevos usuarios de la plataforma
--------------	--------------------------------------------------------

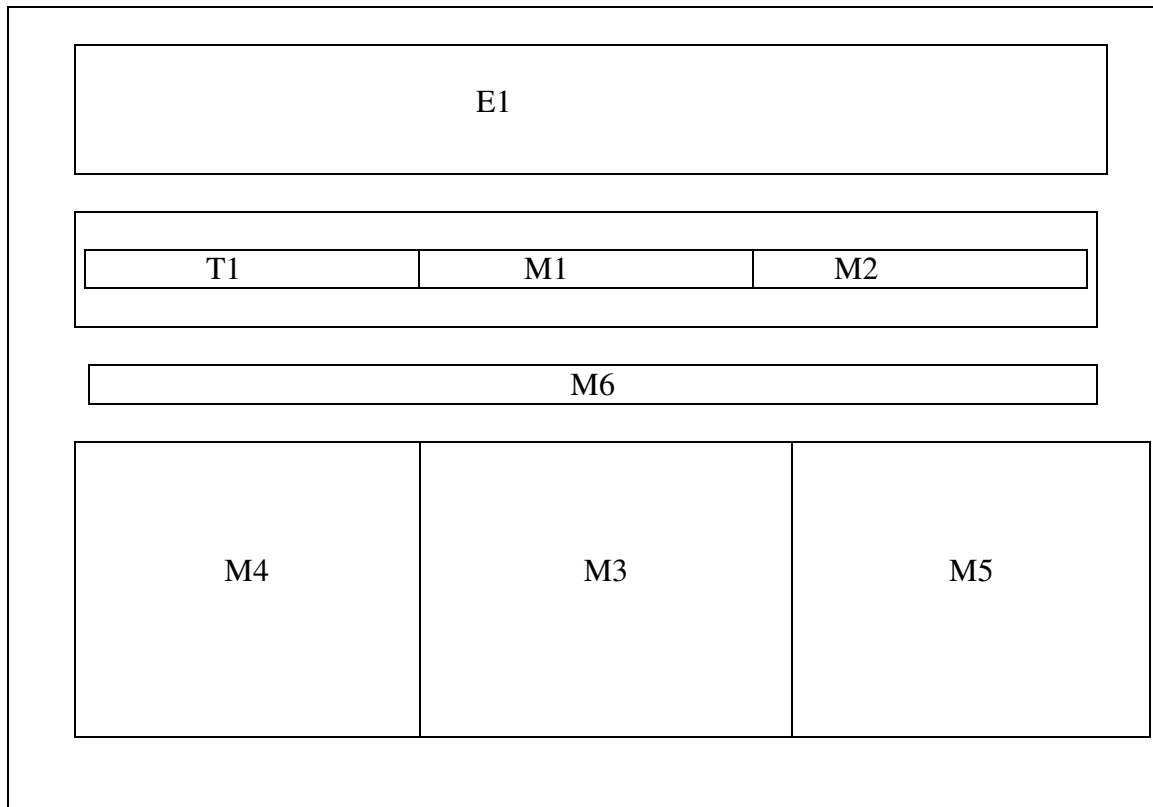


<b>T1</b>	
Nombre	T1Registro
Descripción	Mensaje de texto que da la bienvenida a un nuevo usuario a la sección de registro de la plataforma.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>F1</b>	
Nombre	F1Registro
Descripción	Formulario para el registro de nuevos usuarios: nombre de usuario, contraseña, nombre, primer apellido, segundo apellido, edad, sexo, provincia y colegio.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.



acreditado.php	Página principal para los usuarios acreditados en la plataforma.
----------------	------------------------------------------------------------------



<b>T1</b>	
Nombre	T1Bienvenida
Descripción	Mensaje de texto que da la bienvenida a un usuario que se acaba de acreditar en la plataforma.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

<b>M1</b>	
Nombre	M1ModificarDatos
Descripción	Botón que permite a un usuario acreditado acceder al área de modificación de datos personales.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>M2</b>	
Nombre	M2CerrarSesión
Descripción	Botón que permite a un usuario acreditado cerrar su sesión.
Apariencia	Heredado.

Comportamiento	Heredado.
Origen de datos	Texto estático.

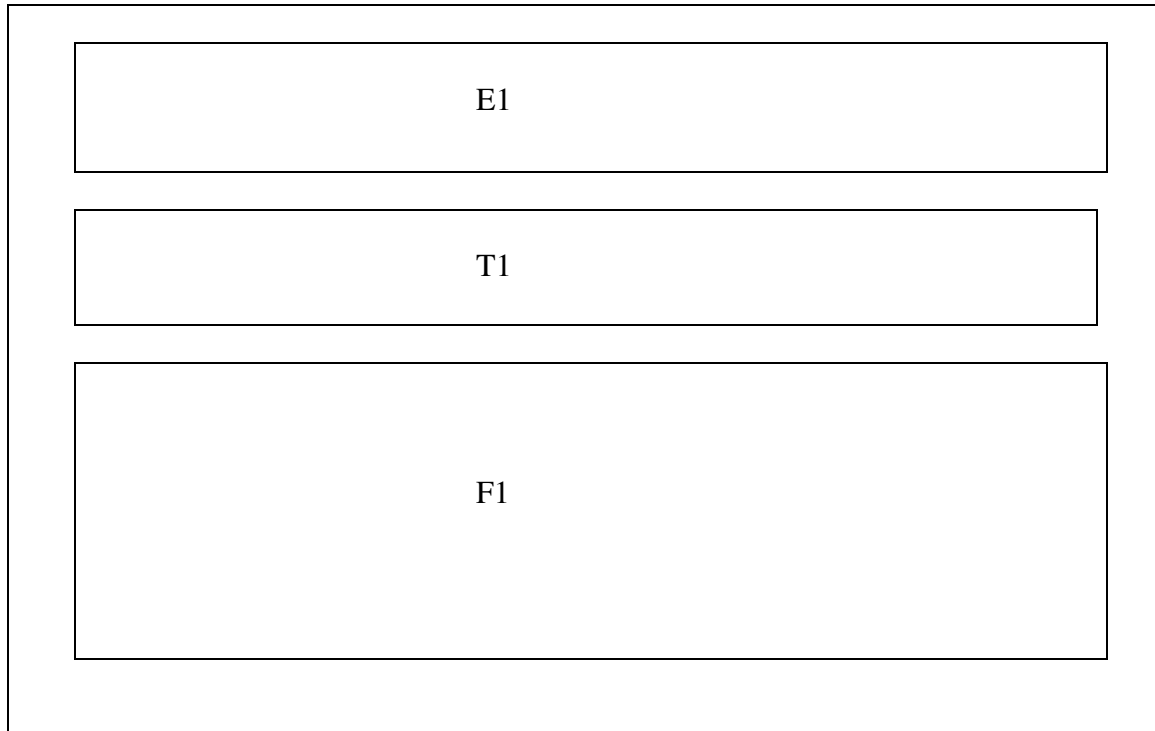
<b>M3</b>	
Nombre	M3MapaNoAcreditado
Descripción	Mapa del oceanográfico que posibilitará ver las ventajas de registrarse en la plataforma.
Apariencia	Específico.
Comportamiento	Específico.
Origen de datos	URL.

<b>M4</b>	
Nombre	M4TagBoard
Descripción	Listado de los últimos mensajes escritos por los usuarios registrados de la plataforma.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

<b>M5</b>	
Nombre	M5Rankings
Descripción	Listado con las mejores puntuaciones de los tres juegos.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

<b>M6</b>	
Nombre	M6TextoAleatorio
Descripción	Mensaje aleatorio sobre el estado de los usuarios de la plataforma.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

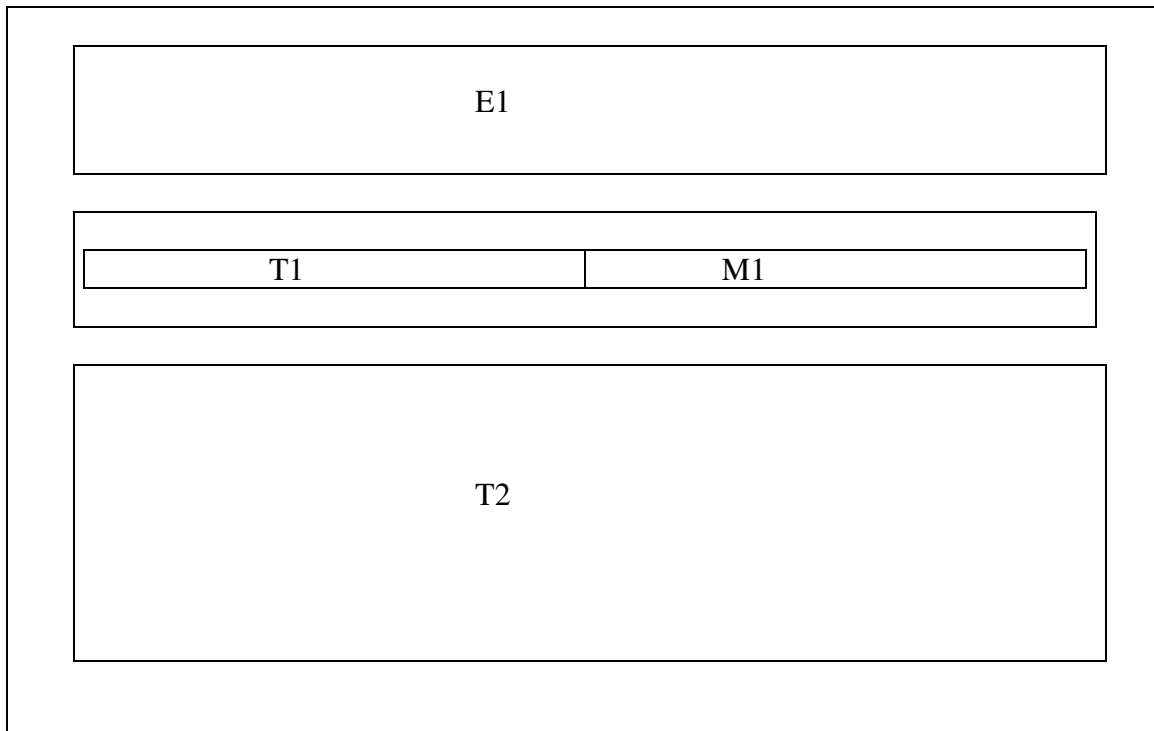
acreditado2.php	Página que permite a un usuario acreditado modificar sus datos de registro.
-----------------	-----------------------------------------------------------------------------



<b>T1</b>	
Nombre	T1ModificarDatos
Descripción	Mensaje de texto que da la bienvenida a un usuario registrado que desea modificar sus datos.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

<b>F1</b>	
Nombre	F1ModificarDatos
Descripción	Formulario que permite a un usuario acreditado modificar sus datos de registro: nombre de usuario, contraseña, nombre, primer apellido, segundo apellido, edad, sexo, provincia y colegio.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

ranking1.php	Página que muestra las máximas puntuaciones del juego “Scott, el pájaro que no sabe volar”
--------------	--------------------------------------------------------------------------------------------

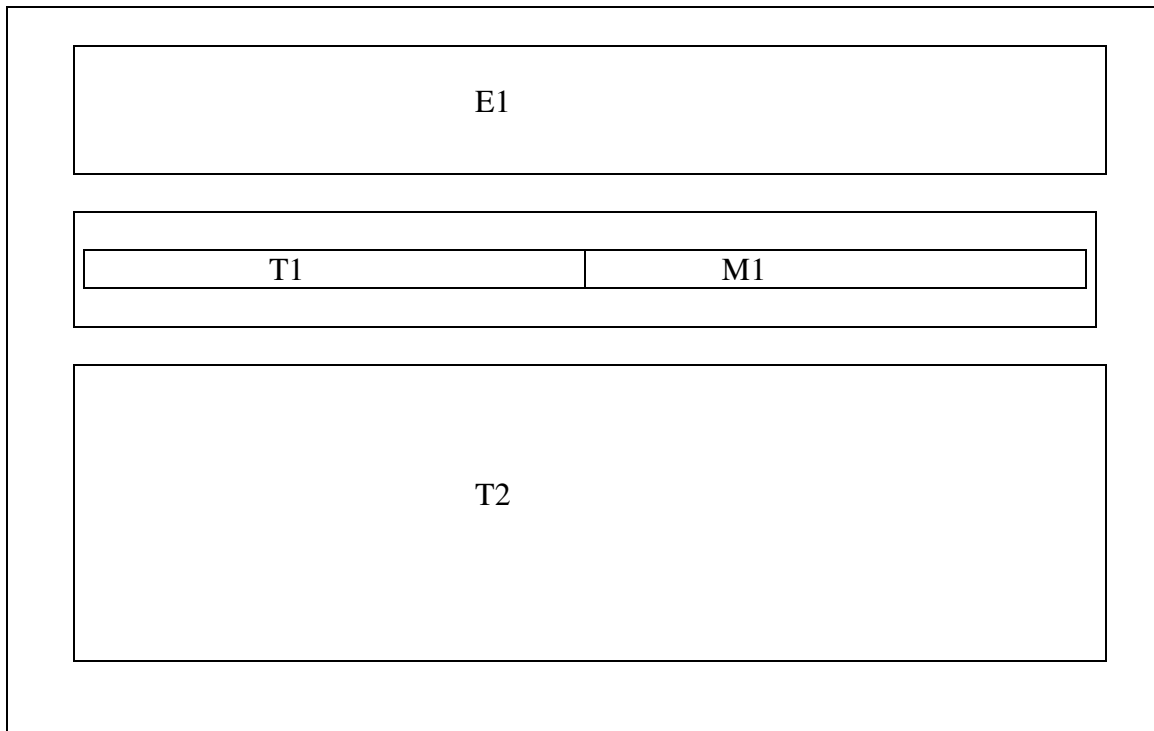


<b>T1</b>	
Nombre	T1Juego1
Descripción	Mensaje de texto que da la bienvenida a la pantalla de puntuaciones del juego “Scott, el pájaro que no sabe volar”.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>M1</b>	
Nombre	M1CerrarVentana
Descripción	Botón que permite cerrar la ventana que nos muestra las puntuaciones.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>T2</b>	
Nombre	T2Juego1
Descripción	Máximas puntuaciones del juego “Scott, el pájaro que no sabe volar”.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

ranking2.php	Página que muestra las máximas puntuaciones del juego “Igor, el pingüino defensor”
--------------	------------------------------------------------------------------------------------

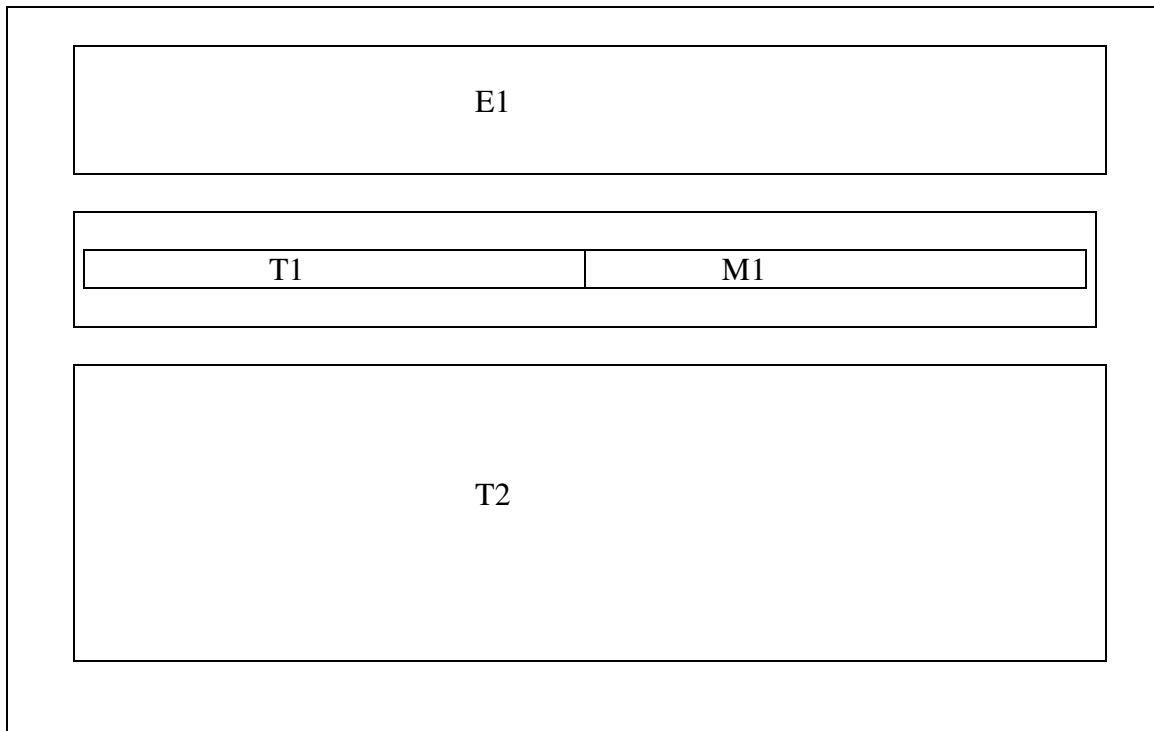


<b>T1</b>	
Nombre	T1Juego2
Descripción	Mensaje de texto que da la bienvenida a la pantalla de puntuaciones del juego “Igor, el pingüino defensor”.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>M1</b>	
Nombre	M1CerrarVentana
Descripción	Botón que permite cerrar la ventana que nos muestra las puntuaciones.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>T2</b>	
Nombre	T2Juego2
Descripción	Máximas puntuaciones del juego “Igor, el pingüino defensor”.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

ranking3.php	Página que muestra las máximas puntuaciones del juego “Bill, el cazador de estrellas”
--------------	---------------------------------------------------------------------------------------



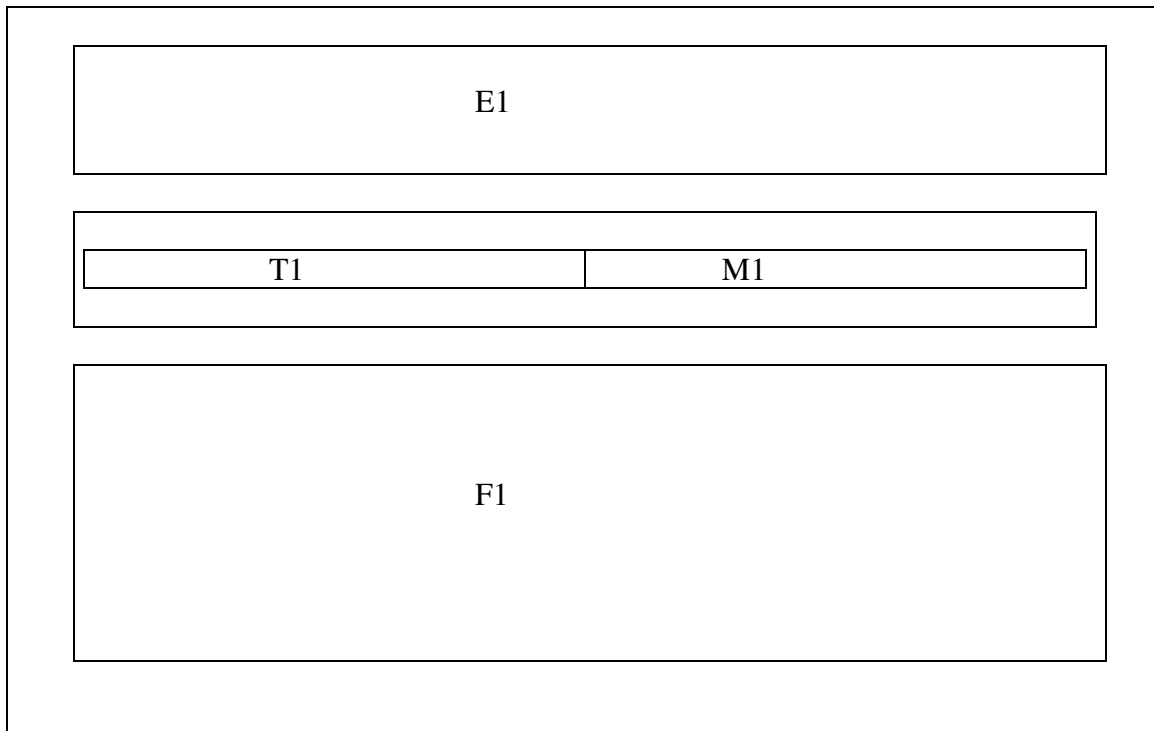
<b>T1</b>	
Nombre	T1Juego3
Descripción	Mensaje de texto que da la bienvenida a la pantalla de puntuaciones del juego “Bill, el cazador de estrellas”.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>M1</b>	
Nombre	M1CerrarVentana
Descripción	Botón que permite cerrar la ventana que nos muestra las puntuaciones.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>T2</b>	
Nombre	T2Juego3
Descripción	Máximas puntuaciones del juego “Bill, el cazador de estrellas”.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.



comentario_escribir.php	Página que posibilita a un usuario escribir un comentario para insertarlo en el tagboard.
-------------------------	-------------------------------------------------------------------------------------------



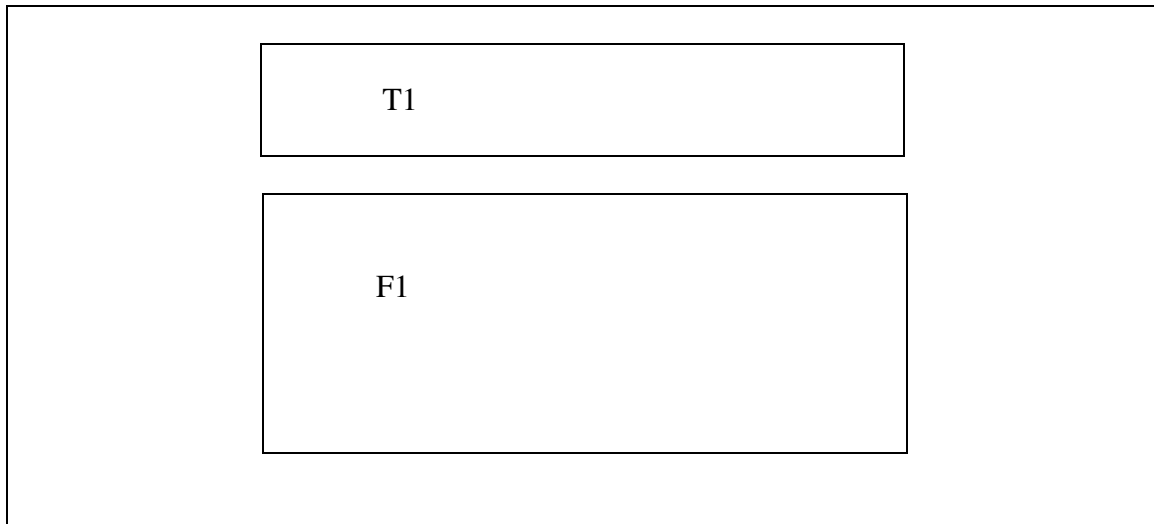
<b>T1</b>	
Nombre	T1TextoBienvenida
Descripción	Mensaje de texto que da la bienvenida a la pantalla de inserción de mensajes en el tagboard.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>M1</b>	
Nombre	M1Volver
Descripción	Botón que permite volver a la ventana anterior.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>F1</b>	
Nombre	F1EnvioMensaje
Descripción	Formulario que permite el envío de un texto que se añadirá al tagboard.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

## 2.2. Descripción de pantallas de la zona de administración

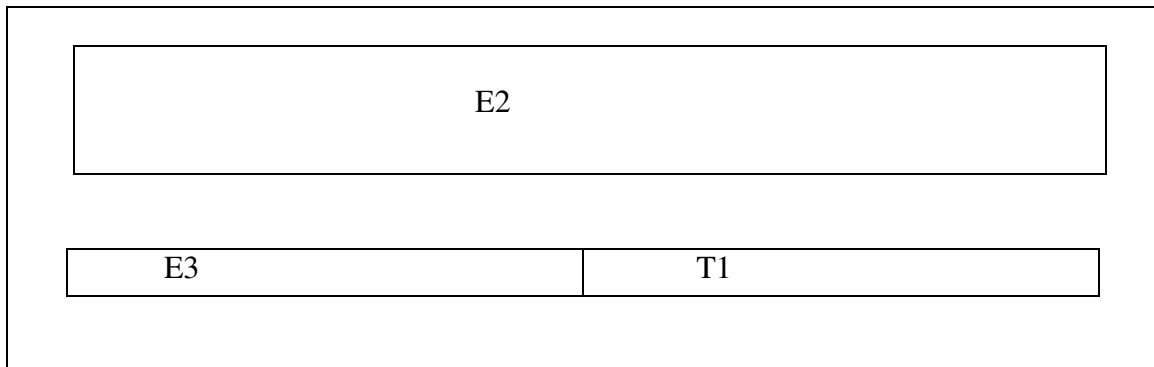
admin.php	Pantalla de acreditación de los administradores.
-----------	--------------------------------------------------



<b>T1</b>	
Nombre	T1Bienvenida
Descripción	Mensaje de bienvenida a la zona de administración de la plataforma.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

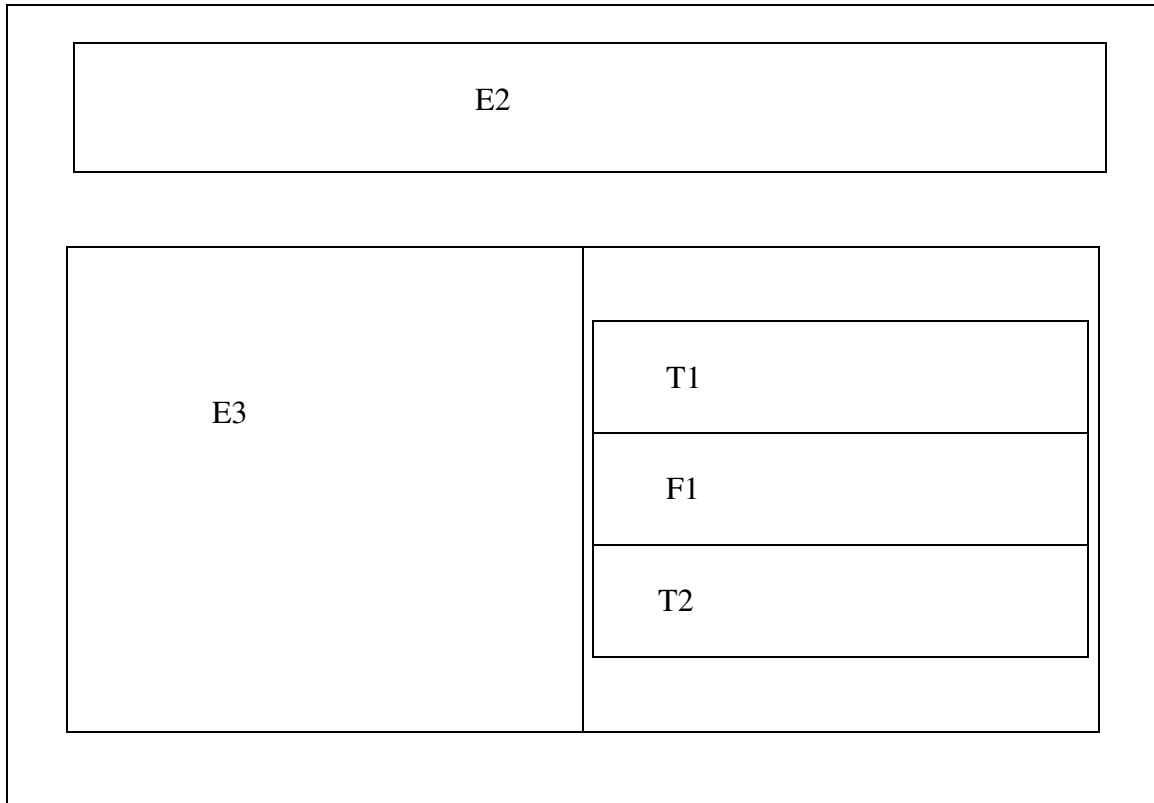
<b>F1</b>	
Nombre	F1LoginAdministrador
Descripción	Formulario que permite a un administrador acreditarse en la plataforma: nombre de administrador y contraseña.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

administrador3.php	Zona de trabajo para los administradores acreditados.
--------------------	-------------------------------------------------------



<b>T1</b>	
Nombre	T1ResultadoOperación
Descripción	Mensaje obtenido al realizar alguna operación sobre la plataforma.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	PHP.

administrador_opcion_crear_administrador.php	Opción para registrar nuevos administradores de la plataforma.
----------------------------------------------	----------------------------------------------------------------

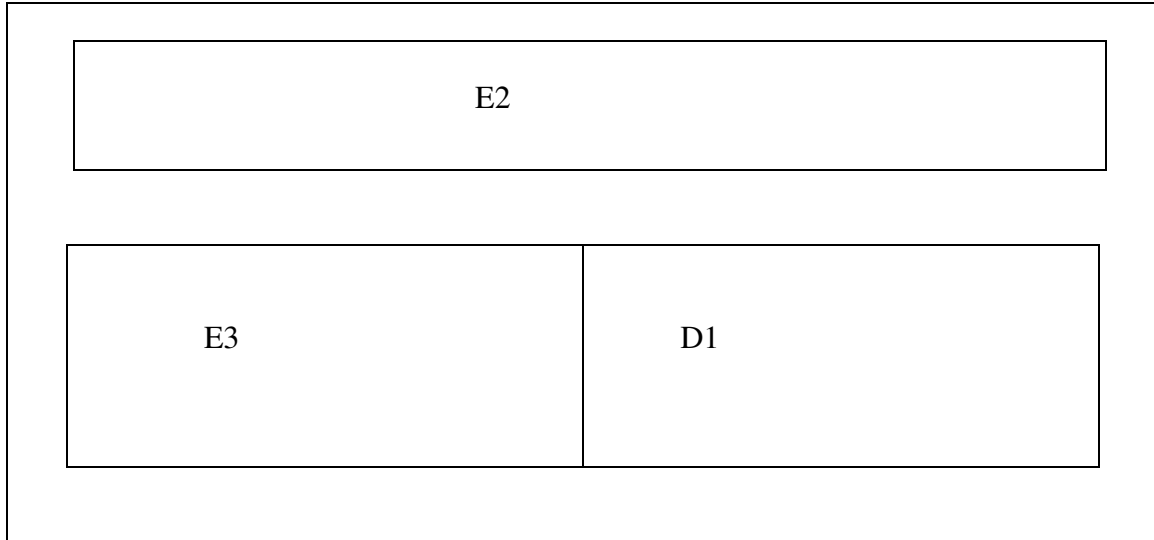


<b>T1</b>	
Nombre	T1CrearAdministrador
Descripción	Mensaje de explicación de esta opción.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>F1</b>	
Nombre	T1CrearAdministrador
Descripción	Formulario que permite la creación de un nuevo administrador: nombre de administrador y contraseña.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

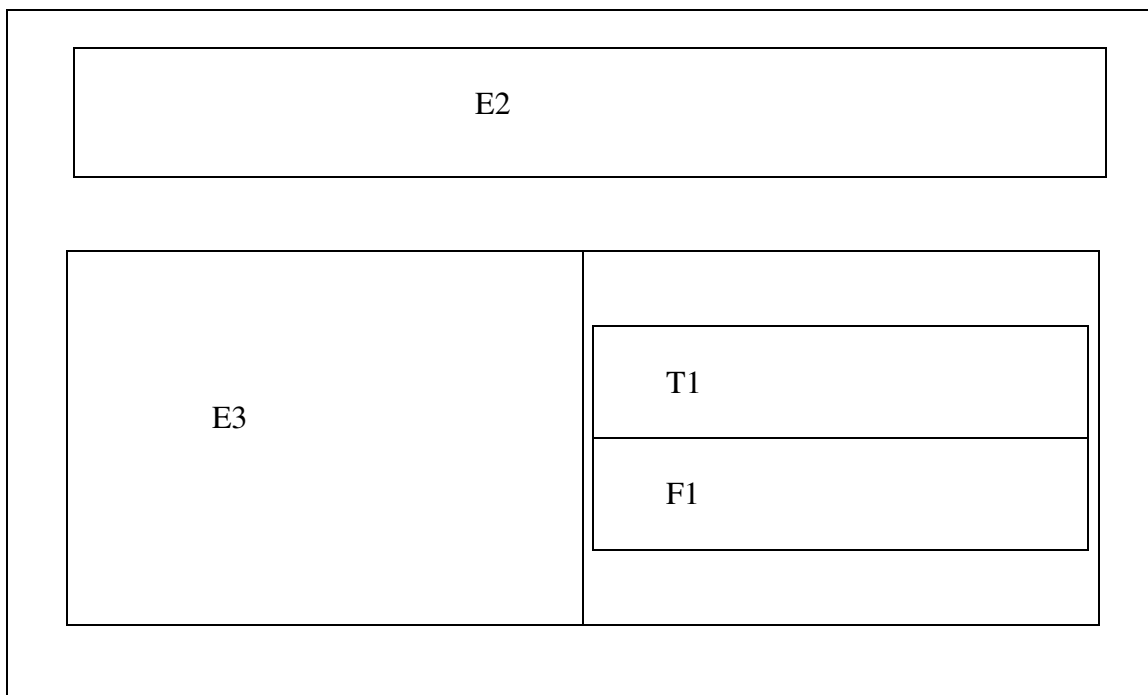
<b>T2</b>	
Nombre	T2ErrorCrearAdministrador
Descripción	Mensaje de error generado al crear un administrador ya existente. opción.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Heredado.

administrador_opcion_listar_administradores.php	Opción para listar los administradores registrados en la plataforma.
-------------------------------------------------	----------------------------------------------------------------------



<b>D1</b>	
Nombre	D1ListadoAdministradores
Descripción	Listado de todos los administradores de la plataforma.
Apariencia	Heredado.
Comportamiento	Unicamente se muestra el campo nick del administrador.
Origen de datos	Base de datos.

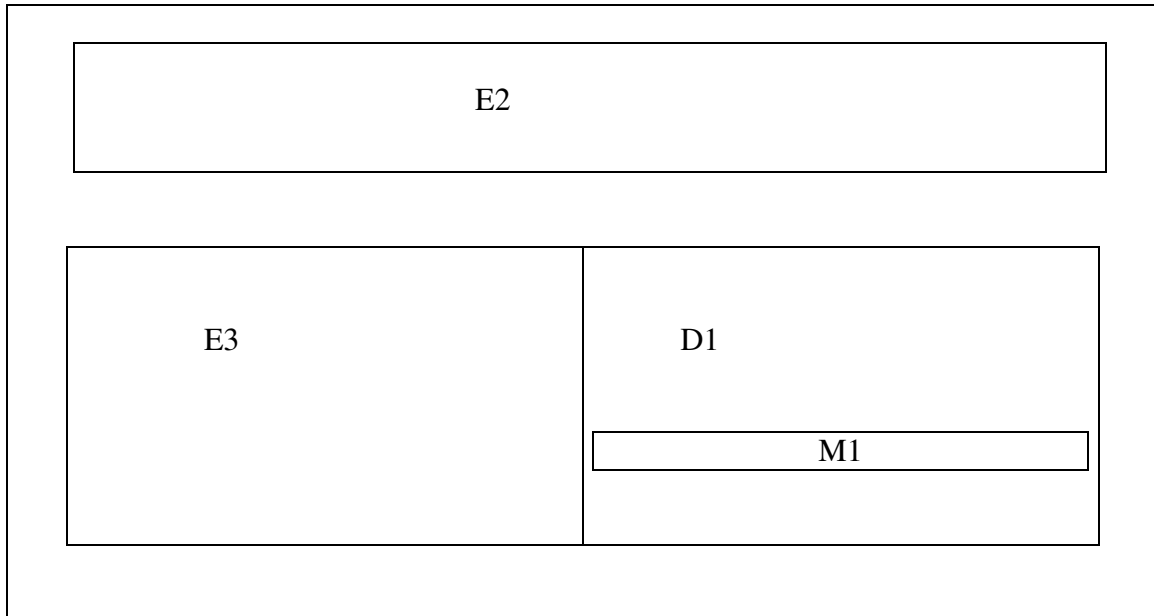
administrador_opcion_borrar_administrador.php	Opción para eliminar administradores de la plataforma.
-----------------------------------------------	--------------------------------------------------------



<b>T1</b>	
Nombre	T1BorrarAdministrador
Descripción	Mensaje de explicación de esta opción.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>F1</b>	
Nombre	F1BorrarAdministrador
Descripción	Formulario en el que introducir el nombre del administrador que se desea eliminar.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

administrador_opcion_listar_usuarios.php	Opción para listar los usuarios que están registrados en la plataforma, con todos sus datos personales.
------------------------------------------	---------------------------------------------------------------------------------------------------------

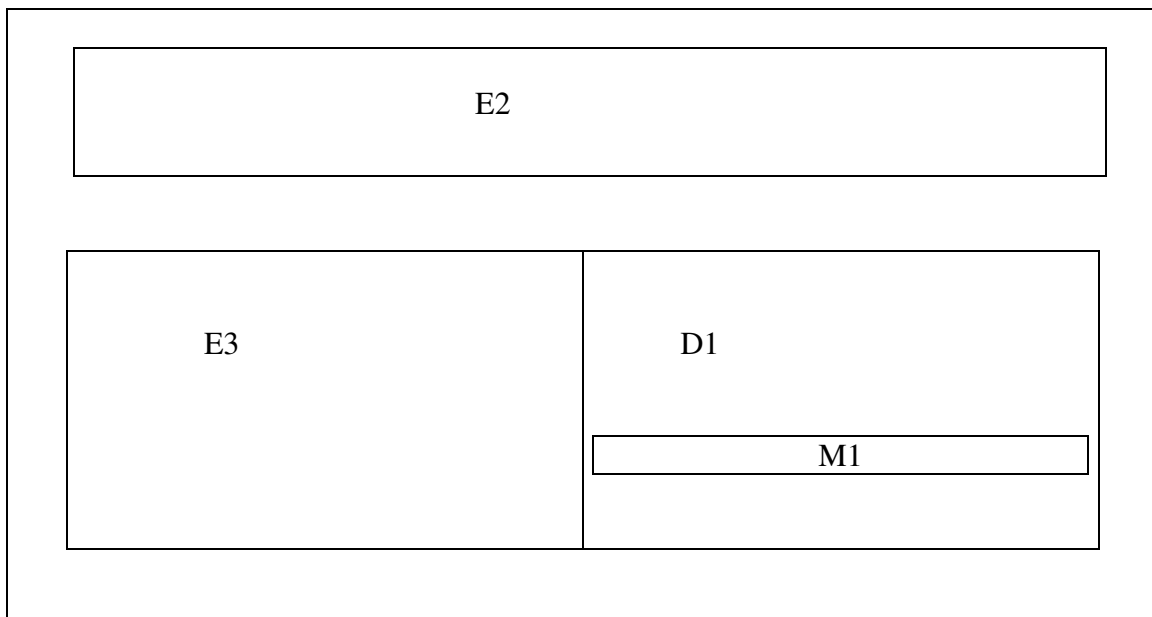


<b>D1</b>	
Nombre	D1ListadoUsuarios
Descripción	Listado de todos los usuarios registrados en la plataforma.
Apariencia	Heredado.
Comportamiento	Se muestran todos los campos de todos los usuarios en forma de tabla: nick, contraseña, nombre, primer apellido, segundo apellido, edad, sexo, ciudad y colegio.
Origen de datos	Base de datos.

<b>M1</b>	
Nombre	M1ImprimirListado
Descripción	Enlaza a un listado de todos los usuarios registrados en la plataforma en su versión para imprimir.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.



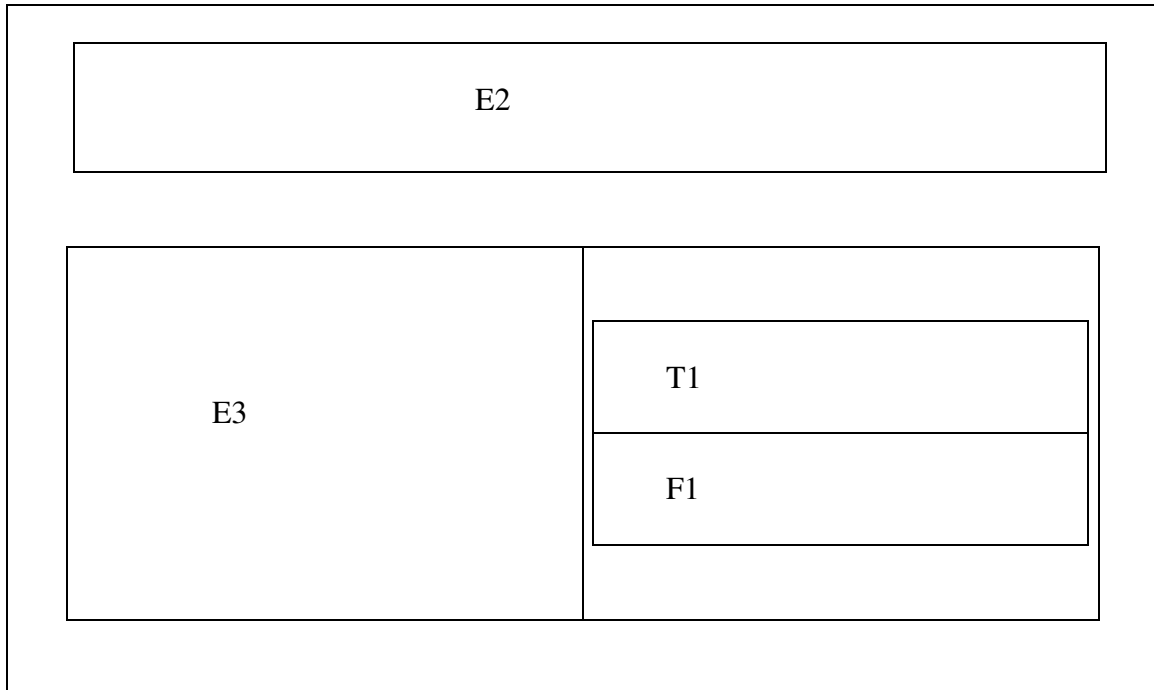
administrador_opcion_listar_puntuaciones.php	Opción para listar las puntuaciones obtenidas en los diferentes juegos.
----------------------------------------------	-------------------------------------------------------------------------



<b>D1</b>	
Nombre	D1ListadoPuntuaciones
Descripción	Listado de todos los usuarios registrados en la plataforma.
Apariencia	Heredado.
Comportamiento	Se muestran el campo nick y el campo puntuación de las 20 máximas puntuaciones de cada uno de los juegos. Además también se muestra el número total de partidas jugadas a cada uno de los juegos.
Origen de datos	Base de datos.

<b>M1</b>	
Nombre	M1ImprimirListado
Descripción	Enlaza a un listado de todas las puntuaciones en su versión para imprimir.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

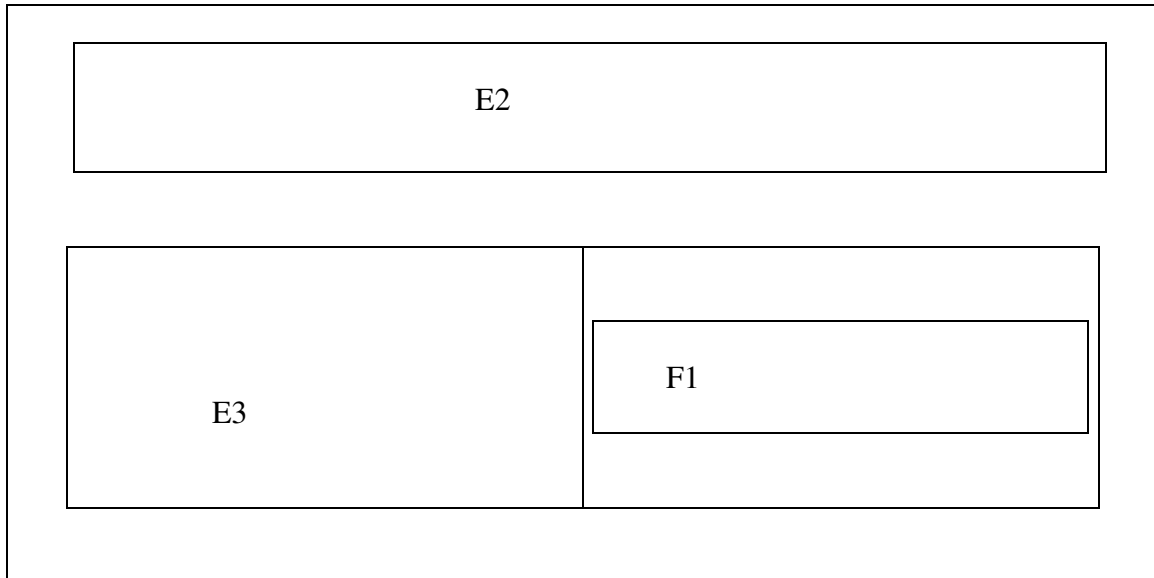
administrador_opcion_modificar_usuario.php	Opción para modificar datos de los usuarios de la plataforma.
--------------------------------------------	---------------------------------------------------------------



<b>T1</b>	
Nombre	T1ModificarUsuario.
Descripción	Mensaje de explicación de esta opción.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

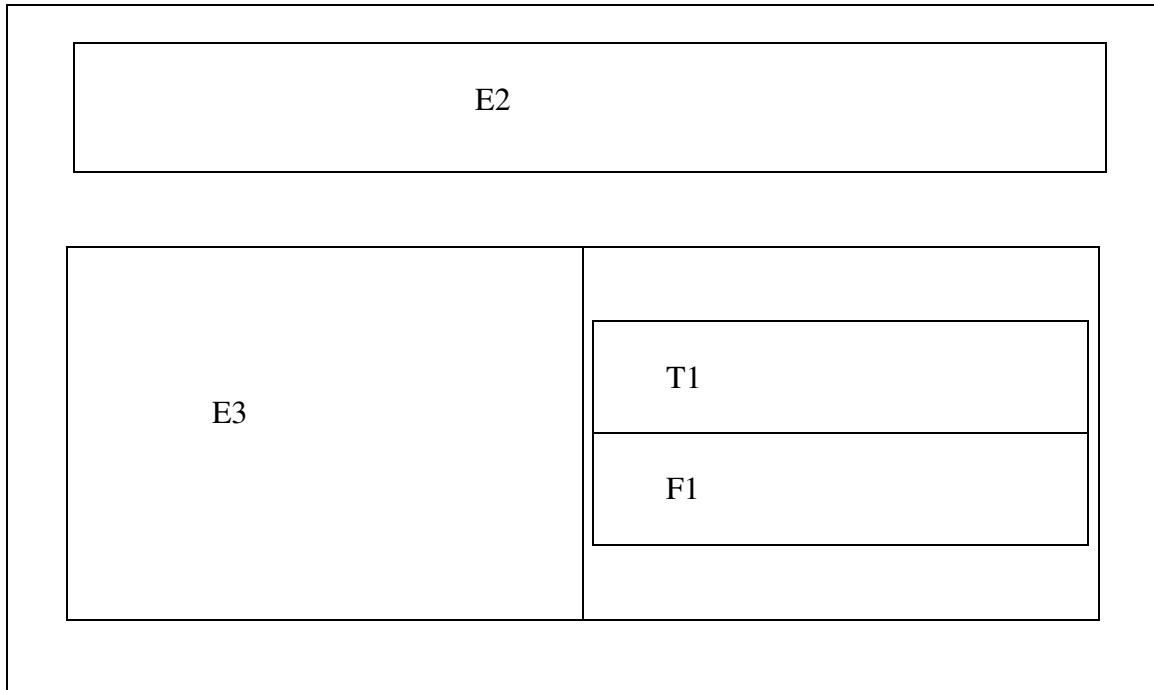
<b>F1</b>	
Nombre	F1ModificarUsuario
Descripción	Formulario en el que introducir el nombre del usuario del que se desea modificar los datos..
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

administrador_modificar_usuario_2.php	Opción para modificar datos de los usuarios de la plataforma.
---------------------------------------	---------------------------------------------------------------



<b>F1</b>	
Nombre	F1ModificarUsuario2.
Descripción	Formulario que permite cambiar los datos del usuario y almacenar estos cambios en la base de datos.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

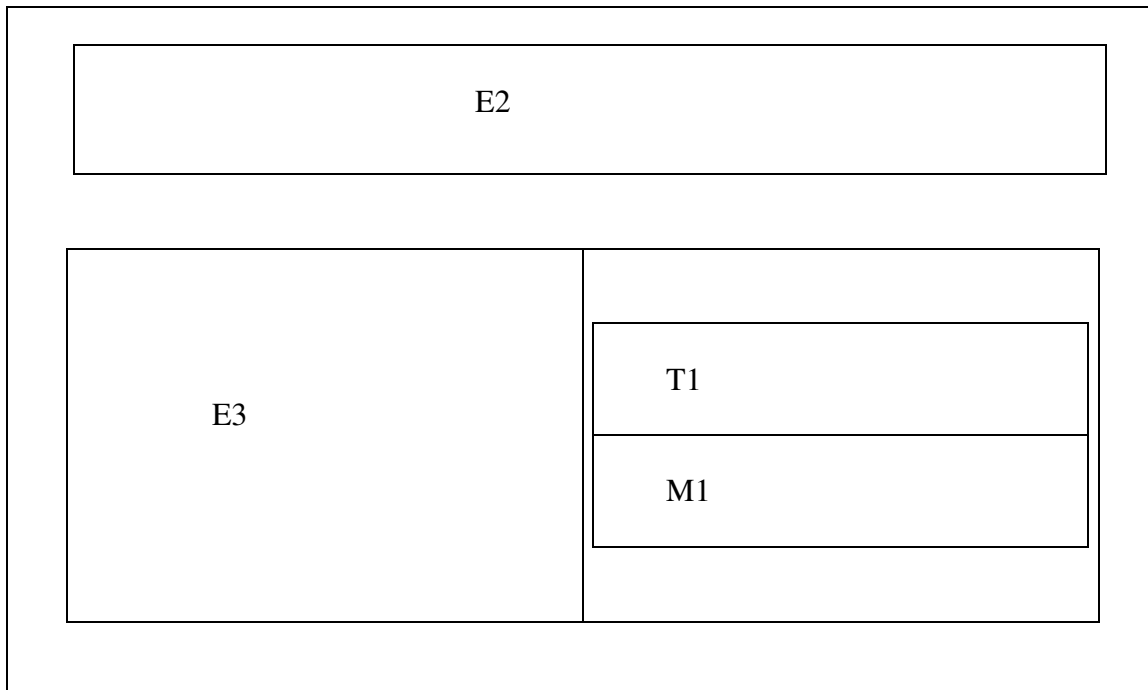
administrador_opcion_borrar_usuario.php	Opción para eliminar usuarios de la plataforma.
-----------------------------------------	-------------------------------------------------



<b>T1</b>	
Nombre	T1BorrarUsuario
Descripción	Mensaje de explicación de esta opción.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>F1</b>	
Nombre	F1BorrarUsuario
Descripción	Formulario en el que introducir el nombre del usuario que se desea eliminar.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

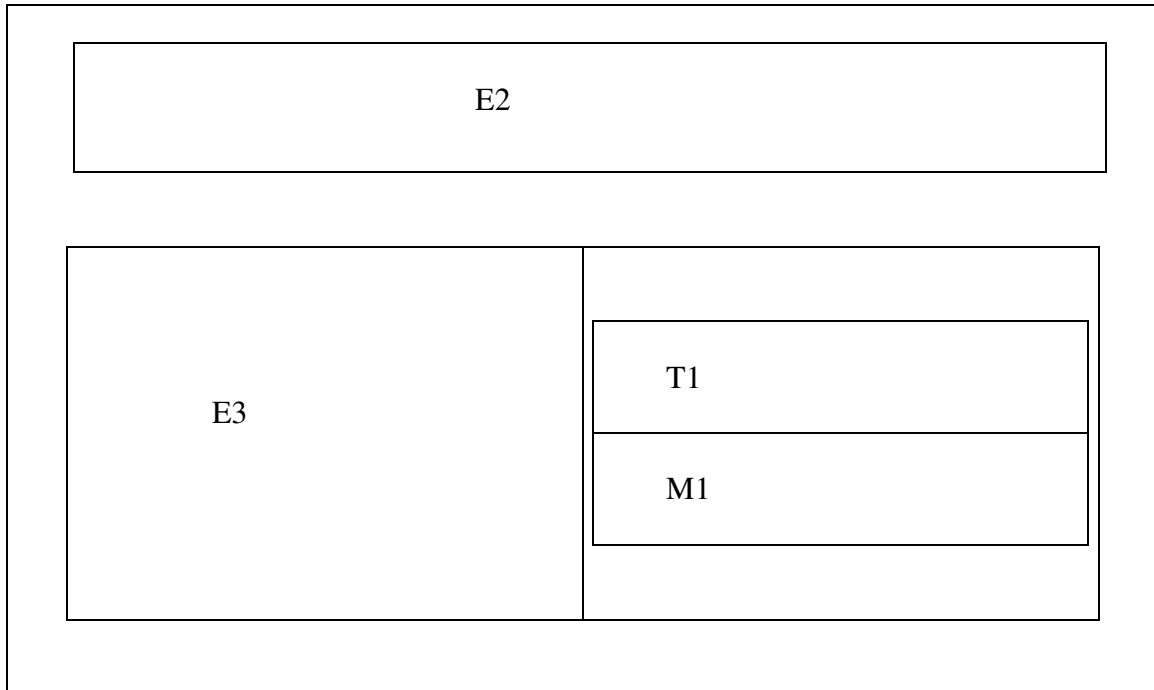
administrador_opcion_crear_bases_de_datos.php	Opción para generar las bases de datos necesarias en el sistema.
-----------------------------------------------	------------------------------------------------------------------



<b>T1</b>	
Nombre	T1CrearBasesDeDatos
Descripción	Mensaje de explicación de esta opción.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>M1</b>	
Nombre	M1CrearBasesDeDatos
Descripción	Botón que procede a generar las bases de datos del sistema.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

administrador_opcion_eliminar_bases_de_datos.php	Opción para generar las bases de datos necesarias en el sistema.
--------------------------------------------------	------------------------------------------------------------------



<b>T1</b>	
Nombre	T1EliminarBasesDeDatos
Descripción	Mensaje de explicación de esta opción.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

<b>M1</b>	
Nombre	M1EliminarBasesDeDatos
Descripción	Botón que procede a eliminar las bases de datos del sistema.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

administrador_opcion_listar_usuarios_impresion.php	Listado preparado para la impresión de los usuarios registrados en la plataforma
----------------------------------------------------	----------------------------------------------------------------------------------

T1
M1

<b>T1</b>	
Nombre	T1ListadoUsuarios
Descripción	Listado de los usuarios registrados en la plataforma con todos sus datos personales.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

<b>M1</b>	
Nombre	M1BotonImprimir
Descripción	Botón que permite imprimir la página actual y posteriormente cerrarla.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.

administrador_opcion_listar_puntuaciones_impresion.php	Listado preparado para la impresión de las puntuaciones de los juegos
--------------------------------------------------------	-----------------------------------------------------------------------

T1
M1

<b>T1</b>	
Nombre	T1ListadoPuntuaciones
Descripción	Listado de las puntuaciones obtenidas en los juegos de la plataforma.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Base de datos.

<b>M1</b>	
Nombre	M1BotonImprimir
Descripción	Botón que permite imprimir la página actual y posteriormente cerrarla.
Apariencia	Heredado.
Comportamiento	Heredado.
Origen de datos	Texto estático.



## 2.3. Elementos comunes

Elemento E1:



Elemento E2:



Elemento E3:



## 3. Lenguaje de color

### 3.1. Descripción de estilos estandar de la zona de usuarios

Texto estandar	
Apariencia	Texto: Verdana, Arial, Helvetica, sans-serif Color: #CCCCCC Tamaño: 9 px
Comportamiento	Ninguno

Texto formulario	
Apariencia	Texto: Verdana, Arial, Helvetica, sans-serif Color: #025576 Tamaño: 9 px
Comportamiento	Ninguno

Texto error	
Apariencia	Texto: Verdana, Arial, Helvetica, sans-serif Color: #FF0000 Tamaño: 12 px
Comportamiento	Ninguno

Texto enlace	
Apariencia	Igual a texto estandar; en roll over adopta el color #FFFFFF y subrayado
Comportamiento	Abre el vínculo en la misma ventana

### 3.2. Descripción de estilos estandar de la zona de administración

Texto estandar	
Apariencia	Texto: Verdana, Arial, Helvetica, sans-serif Color: #000000 Tamaño: 9 px
Comportamiento	Ninguno

Texto enlace	
Apariencia	Texto: Verdana, Arial, Helvetica, sans-serif Color: #FFFFFF, en roll over se subraya Tamaño: 9 px
Comportamiento	Abre el vínculo en la misma ventana

Texto mensaje	
Apariencia	Texto: Verdana, Arial, Helvetica, sans-serif Color: #FFFFFF Tamaño: 9 px
Comportamiento	Ninguno

Texto error	
Apariencia	Texto: Verdana, Arial, Helvetica, sans-serif Color: #FF0000 Tamaño: 12 px
Comportamiento	Ninguno

### 3.3. Descripción de colores

Descripción	Valor hexadecimal	Ejemplo
Gris	#CCCCCC	Color gris
Gris azulado	#025576	Color gris azulado
Rojo	#FF0000	Color rojo
Negro	#000000	Color negro
Blanco	#FFFFFF	Color blanco

## 4. Imágenes de ejemplo

### 4.1. Imágenes de ejemplo de la zona de usuarios

UN CHIAPUZON DE DIVERSION

Usuario  Contraseña  Enviar Registrarse Contacta con nosotros

gonlo ha sido el mejor ayudando a Igor defendiendo su poblado

**Últimos comentarios**

**juan:** esq en el juego de Bill no solo te pueden matar si te toca un bicho, tambien te matan una vida si una estrella de mar se te escapa por el borde izquierdo d la pantalla...

**kinken:** Bill, el cazador de estrellas me mata sin sentido a veces..... sin tokar ningun tiburon me mara !:(

**kinken:** Bill, el cazador de estrellas me mata sin sentido a veces..... sin tokar ningun tiburon me mara !:(

**kinken:** pedazo de web! mola mazo

**kinken:** pedazo de web! mola mazo

**kinken:** pedazo de web! mola mazo

**carmarsa:** Sanoriento.mira Cinema

Política de privacidad / Condiciones de uso

**Scott, el pájaro que no sabe volar**

luis	84
carmarsa	73
kinken	73
gonlo	41
kinken	40
kinken	34
kinken	32
luis	31

**Igor, el pingüino defensor**

gonlo	2540
juan	2480
kinken	1040
damon	820
Claudia	750
kinken	650
Claudia	560
carlos	550

**Bill, el cazador de estrellas**

*index.html*

UN CHIAPUZON DE DIVERSION

¡Bienvenido a la página de registro !

Escoge tu nombre de usuario:

Escoge tu contraseña:

Nombre:

Primer apellido:

Segundo apellido:

Edad: 3 años

Sexo: Chico

Provincia: Alava

Colegio:

Volver

*registro.php*



Bienvenido, Juan Modificar datos usuario [Cerrar sesión](#)

¿Serás capaz de superar los 2540 puntos que ha conseguido gonlo ayudando a Igor?

**Últimos comentarios**

**juan:** esq en el juego de Bill no solo te pueden matar si te toca un bicho, tambien te matan una vida si una estrella de mar se te escapa por el borde izquierdo d la pantalla...

**kinken:** Bill, el cazador de estrellas me mata sin sentido a veces..... sin tokar ningun tiburón me mara !:(

**kinken:** Bill, el cazador de estrellas me mata sin sentido a veces..... sin tokar ningun tiburón me mara !:(

**kinken:** pedazo de web! mola mazo

**kinken:** pedazo de web! mola mazo

**kinken:** pedazo de web! mola mazo



**Scott, el pájaro que no sabe volar**


luis	84
carmarsa	73
kinken	73
gonlo	41
kinken	40
kinken	34
kinken	32
luis	31

**Igor, el pingüino defensor**

gonlo	2540
juan	2480
kinken	1040
damon	820
Claudia	750
kinken	650
Claudia	560
carlos	550

**Bill, el cazador de estrellas**

acreditado.php



¡Hola Juan! En esta sección de la web podrás modificar tus datos personales

Nombre de usuario: Juan

Contraseña: ●●●●

Nombre: Juan

Primer apellido: Pampló

Segundo apellido: Moreno


Sexo: Chico

Edad: 14

Localidad: Alava

Colegio: EUI

acreditado2.php



Desde aquí podras escribir algún comentario para que todo aquel que entre en la web lo lea. [Volver](#)

Hola, |

(Puedes escribir 200 caracteres)

Enviar

*comentario\_escribir.php*



## 4.2. Imágenes de ejemplo de la zona de administración

Bienvenido a la zona para administradores de la plataforma:  
¡Un chapuzón de diversión!

Usuario

Contraseña

*admin.php*

 **UN CHAPUZON DE DIVERSION**   
Zona de administración

Cerrar sesión

- Crear administrador
- Listar administradores
- Borrar administrador
- Listar usuarios
- Listar puntuaciones
- Modificar usuario
- Borrar usuario
- Crear bases de datos
- Destruir bases de datos

*administrador3.php*



*administrador\_opcion\_crear\_administrador.php*



*administrador\_opcion\_listar\_administradores.php*



*administrador\_opcion\_borrar\_administrador.php*



*administrador\_opcion\_listar\_usuarios.php*



Nick	Password	Nombre	Apellido1	Apellido2	Edad	Sexo	Ciudad	Colegio
luis	fffi	luis	merle	farinos	3	Chico	Alava	eui
Juan	12345	Juan	Pampló	Moreno	14	Chico	Valencia	EUI
Toni	789456123	Toni	Moreno	Beleña	5	Chico	Alava	conselleria
Carlos	asdf	Carlos	Martínez	Sañudo	7	Chico	Valencia	eui
Victor	qwert	Víctor	Luján	Gómez	9	Chico	Alava	eui
carmarsa	bilbo10	carlos	martinez	saqudo	3	Chico	La Gomera	yeah
Zenit	Zenit	Daniel	Martinez	Izquierdo	14	Chico	Valencia	Torres
guybrush	lechuck	guy	brush	threewood	7	Chico	Lugo	sw
Damon	atletico	Ruf	hola	hola	9	Chico	Valencia	
diego	diego	diego	diego	diego	3	Chico	Valencia	
kinken	jairostar	kike	.	.	10	Chico	Alava	
hulk	4444	jorge	perez	lopez	9	Chico	Alava	jesuitas
prof	prof	prof	prof	prof	14	Chico	Valencia	UPV
gonlo	delaalmendra	Sergio	Gonzalez	Lozano	3	Chico	Alava	legolianos
Claudia	nene	Claudia	Climent	Sánchez	14	Chica	Valencia	

Imprimir

*administrador\_opcion\_listar\_usuarios\_impresion.php*

## UN CHIAPUZON DE DIVERSION

Zona de administración

[Cerrar sesión](#)

	Scott, el pájaro que no sabe volar		Igor, el pingüino defensor		Bill, el cazador de estrellas	
	Nick	Puntuación	Nick	Puntuación	Nick	Puntuación
Crear administrador	Número de partidas jugadas: 87		Número de partidas jugadas: 35		Número de partidas jugadas: 59	
Listar administradores	luis	84	gonlo	2540	juan	192
Borrar administrador	carmarsa	73	juan	2480	juan	182
Listar usuarios	kinken	73	kinken	1040	luis	174
Listar puntuaciones	gonlo	41	damon	820	toni	173
Modificar usuario	kinken	40	Claudia	750	juan	125
Borrar usuario	kinken	34	kinken	650	juan	120
Crear bases de datos	kinken	32	Claudia	560	juan	87
Destruir bases de datos	luis	31	carlos	550	luis	61
	kinken	31	luis	550	luis	60
	kinken	29	carlos	450	gonlo	53
	kinken	25	toni	400	kinken	47
	gonlo	23	kinken	360	juan	46
	kinken	23	victor	350	toni	43
	gonlo	22	carlos	350	juan	41
	juan	22	juan	300	kinken	38
	juan	21	diego	300	juan	37
	gonlo	20	carmarsa	280	gonlo	37
	juan	20	carmarsa	280	kinken	37
	kinken	20	gonlo	280	juan	36
	undefined	19	carmarsa	280	kinken	36

*administrador\_opcion\_listar\_puntuaciones.php*

Scott, el pájaro que no sabe volar		Igor, el pingüino defensor		Bill, el cazador de estrellas	
Número de partidas jugadas: 87		Número de partidas jugadas: 35		Número de partidas jugadas: 59	
Nick	Puntuación	Nick	Puntuación	Nick	Puntuación
luis	84	gonlo	2540	juan	192
carmarsa	73	juan	2480	juan	182
kinken	73	kinken	1040	luis	174
gonlo	41	damon	820	toni	173
kinken	40	Claudia	750	juan	125
kinken	34	kinken	650	juan	120
kinken	32	Claudia	560	juan	87
luis	31	carlos	550	luis	61
kinken	31	luis	550	luis	60
kinken	29	carlos	450	gonlo	53
kinken	25	toni	400	kinken	47
gonlo	23	kinken	360	juan	46
kinken	23	victor	350	toni	43
gonlo	22	carlos	350	juan	41
juan	22	juan	300	kinken	38
juan	21	diego	300	juan	37
gonlo	20	carmarsa	280	gonlo	37
juan	20	carmarsa	280	kinken	37
kinken	20	gonlo	280	juan	36
undefined	19	carmarsa	280	kinken	36

Imprimir

*administrador\_opcion\_listar\_puntuaciones\_impresion.php*

## UN CHIAPUZON DE DIVERSION

Zona de administración

Cerrar sesión

- Crear administrador
- Listar administradores
- Borrar administrador
- Listar usuarios
- Listar puntuaciones
- Modificar usuario
- Borrar usuario
- Crear bases de datos
- Destruir bases de datos

Inserta el nombre del usuario del que deseas modificar sus datos:

Usuario:

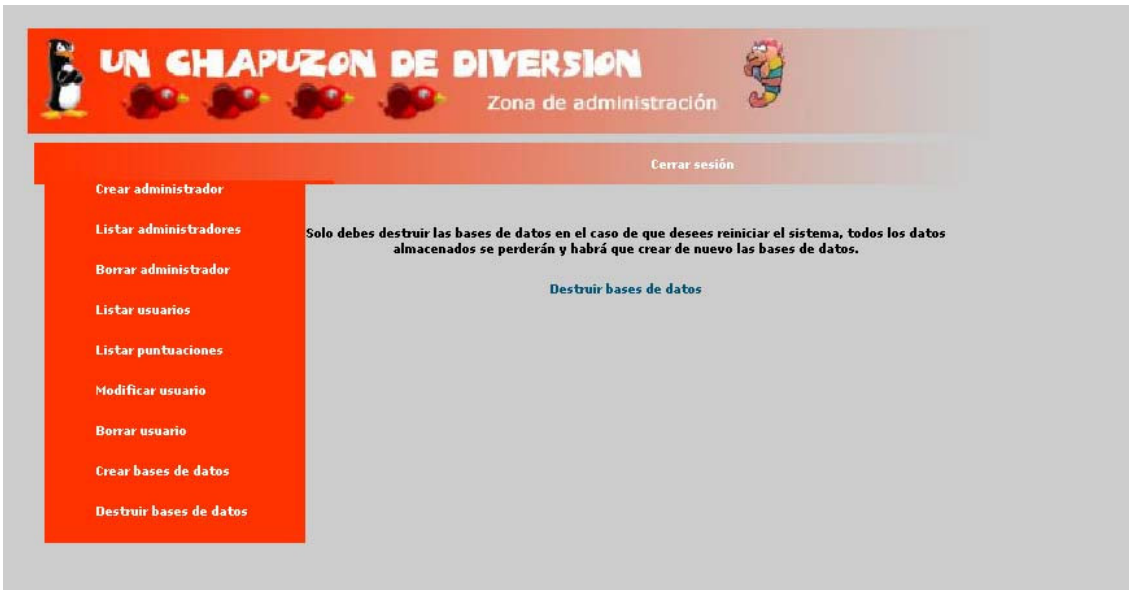
*administrador\_opcion\_modificar\_usuario.php*



*administrador\_opcion\_borrar\_usuario.php*



*administrador\_opcion\_crear\_bases\_de\_datos.php*



*administrador\_opcion\_destruir\_bases\_de\_datos.php*

# III. IMPLEMENTACIÓN DE LA PLATAFORMA

## 1. Definición de la base de datos

### 1.1. Presentación de la base de datos Oceanográfico

Se desea mantener información de la plataforma que estamos desarrollando; para ello se ha definido una base de datos cuyo esquema se muestra a continuación.

**USUARIOS** (nick: d\_nck, nombre: d\_nom, apellido1: d\_ap1, apellido2: d\_ap2, sexo: d\_sex, edad: d\_edad, provincia: d\_prov, colegio: d\_col, password: d\_pswd)

CP: {nick}  
VNN: {nombre}  
VNN: {apellido1}  
VNN: {apellido2}  
VNN: {sexo}  
VNN: {edad}  
VNN: {provincia}  
VNN: {password}

**ADMINISTRADOR** (nick: d\_nck, password: d\_pswd)

CP: {nick}  
VNN: {password}

**MENSAJES** (id: d\_id, nick: d\_nck, mensaje: d\_msj)

CP: {id}

**JUEGO1** (id: d\_id, nick: d\_nck, puntuacion: d\_punt)

CP: {id}  
VNN: {nick}  
VNN: {puntuacion}

**JUEGO2** (id: d\_id, nick: d\_nck, puntuacion: d\_punt)

CP: {id}  
VNN: {nick}  
VNN: {puntuacion}

**JUEGO3** (id: d\_id, nick: d\_nck, puntuacion: d\_punt)

CP: {id}  
VNN: {nick}  
VNN: {puntuacion}

## Definición de los dominios:

<u>Nombre</u>	<u>Tipo de datos</u>
d_nck	varchar(30)
d_msj	varchar(200)
d_nom	varchar(30)
d_ap1	varchar(30)
d_ap2	varchar(30)
d_sex	entero(1)
d_edad	entero(2)
d_prov	varchar(30)
d_col	varchar(30)
d_pswd	varchar(30)
d_id	entero(4)

## 1.2. Descripción de los atributos de cada relación

### Usuarios

**nick:** nombre ficticio que adoptará el jugador al registrarse como usuario en nuestra plataforma, y a través del cual será identificado.

**nombre:** nombre real del usuario que se registra en la plataforma.

**apellido1:** primer apellido del usuario registrado.

**apellido2:** segundo apellido del usuario registrado.

**sexo:** determina si el usuario es niño o niña.

**edad:** cuántos años tiene.

**provincia:** nombre de la provincia en la que vive el usuario.

**colegio:** nombre del colegio en el que estudia.

**password:** contraseña que utiliza el usuario para poder acceder como usuario registrado a la plataforma.

### Administrador

**nick:** nombre ficticio que adoptará cualquier usuario que haga las funciones de administrador y que le servirá para acceder a la zona restringida de administradores de la plataforma.

**password:** contraseña que utiliza el usuario para poder acceder como administrador a la zona de administración de la plataforma.

### Mensajes

**id:** número de indentificación del mensaje.

**nick:** identificador del usuario que ha escrito el mensaje.

**mensaje:** texto que se mostrará en el tagboard de la plataforma.

### Juego1

**id:** número de identificación que permite que un usuario se pueda inscribir diferentes veces en la tabla de resultados o ranking.

**nick:** identificador del usuario que ha puntuado en el juego.

**puntuación:** número de puntos obtenidos por un usuario en el juego.

### Juego2

**id:** número de identificación que permite que un usuario se pueda inscribir

diferentes veces en la tabla de resultados o ranking.

*nick*: identificador del usuario que ha puntuado en el juego.

*puntuación*: número de puntos obtenidos por un usuario en el juego.

### **Juego3**

*id*: número de identificación que permite que un usuario se pueda inscribir diferentes veces en la tabla de resultados o ranking.

*nick*: identificador del usuario que ha puntuado en el juego.

*puntuación*: número de puntos obtenidos por un usuario en el juego.

## **1.3. Esquema relacional de la base de datos**

El esquema relacional anterior se ha definido en MySQL de la manera siguiente:

```
CREATE TABLE usuarios (  
    nick varchar(30) NOT NULL PRIMARY KEY,  
    nombre varchar(30) NOT NULL,  
    apellido1 varchar(30) NOT NULL,  
    apellido2 varchar(30) NOT NULL,  
    sexo INT(1) NOT NULL,  
    edad INT(2) NOT NULL,  
    provincia varchar(30) NOT NULL,  
    colegio varchar(30),  
    puntos int(6),  
    password varchar(30) NOT NULL  
);  
  
CREATE TABLE administrador (  
    nick varchar(30) NOT NULL PRIMARY KEY,  
    password varchar(30) NOT NULL  
);  
  
CREATE TABLE mensajes (  
    id int(4) NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    nick varchar(30),  
    mensaje varchar(200)  
);  
  
CREATE TABLE juego1 (  
    id int(4) NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    nick varchar(30) NOT NULL,  
    puntuacion int(6) NOT NULL  
);  
  
CREATE TABLE juego2 (  
    id int(4) NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    nick varchar(30) NOT NULL,  
    puntuacion int(6) NOT NULL  
);
```

```
CREATE TABLE juego3 (  
    id int(4) NOT NULL AUTO_INCREMENT PRIMARY KEY,  
    nick varchar(30) NOT NULL,  
    puntuacion int(6) NOT NULL  
);
```



## 2. Implementación de la zona de usuarios

### 2.1. index.php

El fichero index.php es el fichero a través del cual se accederá a la web, y consiste en una página html a través de la cual podremos acceder a la zona para usuarios registrados con nuestro nombre de usuario y contraseña o acceder a la zona de registro de nuevos usuarios. También se incluye un mapa del oceanográfico.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html> // Comienza la descripción de la página html

<head> // Comienza la cabecera del documento, a partir de aquí el programa utilizado
para su edición (Macromedia Dreamweaver MX 2004) inserta código autogenerado
que servirá para mantener la compatibilidad con diferentes navegadores web.

<title>&iexcl;Un chapuz&oacute;n de diversi&oacute;n!</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);
//-->
</script>

<link rel="stylesheet" href="oceanografico.css" type="text/css" /> //Cargamos una
hoja de estilo (oceanográfico.css), que será común para todas las páginas de la
plataforma.

</head>

<body bgcolor="#0099CC"> // Comienza el cuerpo de la página. La página ha sido
dividida en capas para una mayor facilidad a la hora de distribuir sus contenidos en
el espacio.

<p class="Estilo10">&nbsp;</p>
<p class="Estilo10">&nbsp;</p>
<div id="Layer3" style="position:absolute; width:864px; height:76px; z-index:3; left:
9px; top: 23px; font-size: 9px;"> // Esta capa se encarga de dibujar la cabecera de la
web, será la misma en todas las páginas que componen la plataforma.
  <p> </p>
</div>
```

```

<p class="Estilo10">&nbsp;&nbsp;&nbsp;</p>
<div id="Capajuego" style="position:absolute; width:561px; height:295px; z-index:2;
left: 165px; top: 251px; font-size: 9px;"> // Esta capa se encarga de dibujar en
pantalla el mapa a traves del cual se podrán acceder a descripciones de cada uno de
los juegos disponibles.
<p>
  <object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#ve
rsion=6,0,29,0" width="585" height="328">
  <param name="movie" value="imagenes/mapa_sin_usuarios.swf">
  <param name="quality" value="high">
  <embed src="imagenes/mapa_sin_usuarios.swf" quality="high"
pluginspage="http://www.macromedia.com/go/getflashplayer" type="application/x-
shockwave-flash" width="585" height="328"></embed>
  </object>
  <font color="#FFFFFF"> </font></p>
</div>
<div id="Layer1" style="position:absolute; width:849px; height:45px; z-index:5; left:
30px; top: 165px; background-color: #0033CC; layer-background-color: #0033CC;
border: 1px none #000000; background-image: url(imagenes/water-texture-blue.jpg);
layer-background-image: url(imagenes/water-texture-blue.jpg);">
  <span style="font-size: 9px"></span></div>
<div id="Layer2" style="position:absolute; width:834px; height:31px; z-index:4; left:
36px; top: 133px; background-color: #0000CC; layer-background-color: #0000CC;
border: 1px none #000000;"> // Capa correspondiente a la entrada de usuarios y
posible registro de los mismos en la plataforma.
  <form name="form1" method="post" action="login.php"> //Utilizaremos un
formulario para la entrada de los usuarios. Para el acceso a la base de datos de los
usuarios este formulario hará una llamada a la página login.php mandandole los
datos del formulario mediante el método post. Los datos del formulario serán escritos
en recuadros de texto cuyos identificadores serán nick y password. Estos recuadros y
el texto descriptor que los acompaña forman parte de una tabla que contribuye a
mantener un orden en la página.
  <table width="809" border="0">
    <tr>
      <td width="37"><span
class="EstiloFormulario"><strong>Usuario</strong></span></td>
      <td width="144"><span class="Estilo9">
        <input name="nick" type="text" id="nick2">
      </span></td>
      <td width="56"><span
class="EstiloFormulario"><strong>Contrase&ntilde;a</strong></span></td>
      <td width="144"><input name="password" type="password"
id="password3"></td>
      <td width="51"><input name="enviar" type="submit" id="enviar3"
value="Enviar"></td>
      <td width="53"><span class="Estilo9"><a href="registro.php"
class="EstiloFormulario">Reg&iacute;strate</a></span></td>
      <td width="294"> // Para los usuarios no registrados, se incluye un enlace a
registro.php, donde podrán darse de alta.

```

```

<?php // En caso de que el usuario que trata de acceder a la página ponga la
contraseña mal, incluimos un script php que se encarga de escribir un mensaje de
error recibido de login.php. Este mensaje será escrito siempre, pero en el caso de
acceder a la página de la forma normal será un mensaje vacío y por tanto no se
imprimirá nada por pantalla.
    $mensaje_de_error_2 = '<p class="EstiloError">';
    $mensaje_de_error_2 .= $mensaje_de_error;
    $mensaje_de_error_2 .= '</p>';
    echo $mensaje_de_error_2;
    ?>
</td>
</tr>
</table>
</form>
<span style="font-size: 9px">
</span></div>
<p>&nbsp;</p>
<p><span style="font-size: 9px">
</span></p><div id="Layer4" style="position:absolute; width:203px; height:304px; z-
index:6; left: 762px; top: 242px;">
    <?php
        include('mensajes2.php'); // Se incluye un fichero que muestra las mejores
puntuaciones.
    ?>
</div>
<div id="Layer4" style="position:absolute; width:413px; height:29px; z-index:16; left:
44px; top: 209px;">
    <?php
        include('mensajes.php'); // Se incluye un fichero que muestra un mensaje
aleatorio con datos sobre los usuarios de la plataforma.
    ?>
</div>
<div id="Layer5" style="position:absolute; width:604px; height:35px; z-index:7; left:
146px; top: 601px;">
    <div align="center"><a href="privacidad.php" class="Estilo9">Política
de privacidad / Condiciones de uso</a></div> // Enlace a un texto con la política de
privacidad de la plataforma.
</div>
<div id="Layer6" style="position:absolute; width:168px; height:26px; z-index:8; left:
14px; top: 59px;">
    <?php
        include('comentario_ver.php'); // Se incluye un fichero que muestra el tagboard
    ?>
</div>
</body>
</html>

```

## 2.2. login.php

El fichero login.php se encarga de realizar una conexión a la base de datos para buscar el usuario y el password solicitados por index.html y, en base a los resultados obtenidos, llevar al usuario al lugar correspondiente

```
<?php

    include('crear_conexion_bd.php'); // Con esta instrucción incluimos el fichero
    encargado de realizar la conexión a la base de datos.

    $nick = $_POST["nick"]; // Esta instrucción y la siguiente recogen los datos
    enviados por index.html y los almacenan para su uso posterior
    $password = $_POST["password"];

    $sql = "SELECT count(*) FROM usuarios WHERE nick = '$nick' AND
    password = '$password'"; // La variable $sql será utilizada para almacenar el texto de
    una consulta sql. En este caso buscamos el número de usuarios que existen en la base
    de datos cuyos valores nick y password sean los recibidos a través del formulario.

    $salida_sql = @mysql_query($sql); // Ejecutamos la consulta y almacenamos
    el resultado en $salida_sql
    $salida_sql_vector = mysql_fetch_array($salida_sql); // Gracias a
    mysql_fetch_array almacenamos el resultado de la consulta en un vector.
    $temp = $salida_sql_vector[0]; // Debido a que la consulta únicamente
    devuelve un valor, este corresponde a la primera posición del vector, y asignamos este
    valor a la variable temporal $temp, en este caso será 1 si el usuario existe en la base
    de datos con la contraseña que ha indicado.

    if ($temp == '1') {
        $nombre_variable = $nick;
        include('acreditado.php'); // El fichero acreditado.php es la página en la que se
        moverán los usuarios registrados. Al llegar a este punto sabemos que el usuario ha
        puesto un nick y una contraseña correctos, y por tanto incluimos esta página para
        que la visualicen.
    } else { // En este punto sabemos que el usuario y la contraseña no estan
    asociados, debemos averiguar el motivo, primero vemos si nick forma parte de la base
    de datos, de forma análoga a lo realizado anteriormente.
        $sql = "SELECT count(*) FROM usuarios WHERE nick = '$nick'";
        $salida_sql = @mysql_query($sql);
        $salida_sql_vector = mysql_fetch_array($salida_sql);
        $temp = $salida_sql_vector[0];
        if ($temp == '1') { // En este punto hemos averiguado que el usuario
        forma parte de la base de datos, por tanto la contraseña que ha puesto es incorrecta.
            $mensaje_de_error = ('La contraseña es incorrecta');
            include('index.html'); // Enviamos al usuario a la página
            principal, indicando que $mensaje_de_error tiene el valor asignado, y en este caso al
            cargar la página se cargará mostrando ese error.
        } else { // Como última posibilidad, el usuario puede no existir, lo que
        nos hace llegar a este punto.
```

```

        $mensaje_de_error_2 = '<p class="EstiloError">Usuario no registrado,
regístrate</p>';
        include ('registro.php'); // La página registro.php nos permite dar de
alta a un usuario de la base de datos, a traves de aquí accedemos a esta página
indicando que ha habido un error (usuario no registrado).
    }
}
?>

```

### 2.3. crear\_conexion\_bd.php

Este fichero se encarga de crear una conexión con la base de datos. En caso de realizar un cambio en la configuración de la base de datos estos cambios se tendrán que indicar únicamente aquí.

```

<?php
$conexion = mysql_connect('localhost', 'root', ""); // Creamos la conexión. El primer
valor de la función indica el servidor, el segundo el usuario y el tercero su
contraseña. En nuestro caso la conexión se realiza en el servidor local a traves del
usuario root, que no tiene contraseña.
    if (!$conexion) {
        die( '<p>Imposible conectar a la base de datos</p>' ); // En este caso no se ha
establecido la conexión, por tanto abortamos la aplicación.
    }
    if ( ! @mysql_select_db('oceanografico') ) {
        die( '<p>Imposible seleccionar la base de datos</p>' ); // En caso de conectar a la
base de datos seleccionaremos el conjunto correcto de datos, en nuestro caso
oceanografico, si llegamos a este punto nuestro conjunto de datos no estará
disponible y abortaremos la aplicación.
    }
?> // En caso de no obtener ningun error, la conexión con la base de datos estará
activa durante la duración de este script, al ser un script que siempre se utilizará
mediante la instrucción include, se podrá utilizar en otros scripts de forma que
simulen que este fichero forma parte de ellos.

```

### 2.4. registro.php

El fichero registro.php será utilizado para la creación de nuevos usuarios. Se accederá a esta página en caso de acceder explícitamente desde el enlace en index.html o en caso de tratar de entrar en la zona de usuarios registrados y poner un usuario incorrecto. Volvemos a encontrar las mismas capas que en el fichero index.html, pero en lugar del mapa tenemos la posibilidad de poner los datos que se nos pidan en un formulario. El otro cambio respecto a la página principal es la supresión de la posibilidad de poner nuestros datos, sustituidos por un mensaje general y otro que aparecerá en función de la forma de acceder a la web.

```

<html>
<head>
<link rel="stylesheet" href="oceanografico.css" type="text/css" />
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);
//-->
</script>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
1"><title>&iexcl;Reg&iacute;strate!</title></head>

<body bgcolor="#0099FF" text="#000000">
<p class="Estilo9">&nbsp;</p>
<p class="Estilo9">
</p>
<div id="Layer3" style="position:absolute; width:864px; height:76px; z-index:3; left:
9px; top: 23px; font-size: 9px;">
  <p> </p>
</div>
<p>&nbsp;</p>
<p>
</p>

<p>&nbsp;</p>
<div id="Layer1" style="position:absolute; width:849px; height:34px; z-index:5; left:
30px; top: 157px; background-color: #0033CC; layer-background-color: #0033CC;
border: 1px none #000000; background-image: url(imagenes/water-texture-blue.jpg);
layer-background-image: url(imagenes/water-texture-blue.jpg);"> <span style="font-
size: 9px"></span></div>
<div id="Layer2" style="position:absolute; width:834px; height:31px; z-index:4; left:
36px; top: 133px; background-color: #0000CC; layer-background-color: #0000CC;
border: 1px none #000000; font-size: 9px; font-weight: bold;"
class="EstiloFormulario"> &iexcl;Bienvenido a la p&aacute;gina de registro ! </div> //
Esta capa únicamente nos muestra el mensaje de bienvenida.
<p>&nbsp;</p>
<p>&nbsp;</p>
<div id="Layer4" style="position:absolute; width:200px; height:115px; z-index:6; left:
78px; top: 198px;"> // Aquí comenzamos una capa que dibujará un formulario en el
cual solicitaremos al usuario los datos que necesitamos para su registro. <p><?php
echo $mensaje_de_error_2 ?></p> // Lo primero de todo es indicar si el usuario ha
accedido a través de login.php, el cual se habrá encargado de darle un valor a la
variable $mensaje_de_error_2.

```

```

<form action="enviando_datos.php" method="post" enctype="multipart/form-data"
name="form1"> // El formulario enviará los datos al servidor a través de
enviando_datos.php mediante el método post.
  <table width="893" border="0">
    <tr>
      <td width="157" valign="top"><span class="Estilo9">Escoge tu nombre de
usuario:</span></td>
      <td width="158" valign="top"><span class="Estilo9">
        <input name="nick" type="text" id="nick2">
      </span></td>
      <td width="564" valign="top">
<?php
$mensaje_de_error_2 = '<p class="EstiloError">';
$mensaje_de_error_2 .= $nick_error;
$mensaje_de_error_2 .= '</p>';
echo $mensaje_de_error_2;
?> // Para cada uno de los posibles errores en el registro (Campos vacios) indica la
existencia de un error en forma de mensaje. En caso de que un error no se haya dado
su variable estará vacia y no escribirá nada. Estos mensajes de error los enviará la
página enviando_datos.php.
</td>
    </tr>
    <tr>
      <td><span class="Estilo9">Escoge tu contrase&ntilde;a:</span></td>
      <td><span class="Estilo9">
        <input name="password" type="password" id="password2">
      </span></td>
      <td><?php
$mensaje_de_error_2 = '<p class="EstiloError">';
$mensaje_de_error_2 .= $password_error;
$mensaje_de_error_2 .= '</p>';
echo $mensaje_de_error_2;
?></td>
    </tr>
    <tr>
      <td><span class="Estilo9">Nombre:</span></td>
      <td><span class="Estilo9">
        <input name="nombre" type="text" id="nombre2">
      </span></td>
      <td><?php
$mensaje_de_error_2 = '<p class="EstiloError">';
$mensaje_de_error_2 .= $nombre_error;
$mensaje_de_error_2 .= '</p>';
echo $mensaje_de_error_2;
?></td>
    </tr>
    <tr>
      <td><span class="Estilo9">Primer apellido:</span></td>
      <td><span class="Estilo9">
        <input name="apellido1" type="text" id="apellido2">

```

```

        </span></td>
        <td><?php
$mensaje_de_error_2 = '<p class="EstiloError">';
$mensaje_de_error_2 .= $apellido1_error;
$mensaje_de_error_2 .= '</p>';
echo $mensaje_de_error_2;
?></td>
    </tr>
    <tr>
        <td><span class="Estilo9">Segundo apellido:</span></td>
        <td><span class="Estilo9">
            <input name="apellido2" type="text" id="apellido22">
        </span></td>
        <td><?php
$mensaje_de_error_2 = '<p class="EstiloError">';
$mensaje_de_error_2 .= $apellido2_error;
$mensaje_de_error_2 .= '</p>';
echo $mensaje_de_error_2;
?></td>
    </tr>
    <tr>
        <td><span class="Estilo9">Edad:</span></td>
        <td><span class="Estilo9">
            <select name="edad" id="select">
                <option value="3">3 a&ntilde;os</option>
                <option value="4">4 a&ntilde;os</option>
                <option value="5">5 a&ntilde;os</option>
                <option value="6">6 a&ntilde;os</option>
                <option value="7">7 a&ntilde;os</option>
                <option value="8">8 a&ntilde;os</option>
                <option value="9">9 a&ntilde;os</option>
                <option value="10">10 a&ntilde;os</option>
                <option value="11">11 a&ntilde;os</option>
                <option value="12">12 a&ntilde;os</option>
                <option value="13">13 a&ntilde;os</option>
                <option value="14">14 a&ntilde;os</option>
            </select>
        </span></td>
        <td>&nbsp;</td>
    </tr>
    <tr>
        <td class="Estilo9">Sexo:</td>
        <td><span class="Estilo9">
            <select name="sexo" id="select2">
                <option value="1">Chico</option>
                <option value="0">Chica </option>
            </select>
        </span></td>
        <td>&nbsp;</td>
    </tr>

```



```

<tr>
  <td class="Estilo9">Provincia:</td>
  <td><span class="Estilo9"><a href="index.html"> </a>
    <select name="provincia" id="select3">
      </select>
    </span></td>
  <td>&nbsp;</td>
</tr>
<tr>
  <td class="Estilo9">Colegio</td>
  <td><input name="colegio" type="text" id="colegio2"></td>
  <td>&nbsp;</td>
</tr>
<tr>
  <td class="Estilo9"><a href="index.html">Volver</a> </td> // Incluimos la
  posibilidad de volver a la página principal, para aquellos usuarios que realmente no
  quiera registrarse (por ejemplo si han escrito mal el nombre de usuario
  accidentalmente)
  <td><span class="Estilo9"><a href="index.html">
    <input type="submit" name="Submit" value="Enviar">
  </a></span></td>
  <td>&nbsp;</td>
</tr>
</table>
<p class="Estilo9"></p>
</form>
</div>
<p>&nbsp;</p>
</body>
</html>

```

## 2.5. enviando\_datos.php

Este fichero sirve para dar de alta a usuarios en la base de datos. Recibe los datos de registro.php y se encarga de conectar con la base de datos y efectuar el registro. En caso de encontrar algún error en los datos recibidos desde registro.php volverá a esta página indicando estos errores.

```

<?php

include('crear_conexion_bd.php');

// Tomamos los datos obtenidos a traves de registro.php y los almacenamos
para su posterior uso
$nick = $_POST["nick"];
$password = $_POST["password"];
$nombre = $_POST["nombre"];
$apellido1 = $_POST["apellido1"];

```

```
$apellido2 = $_POST["apellido2"];
$edad = $_POST["edad"];
$sexo = $_POST["sexo"];
$colegio = $_POST["colegio"];
$localidad = $_POST["localidad"];
```

*\$aceptar\_envio\_datos = '1'; // \$aceptar\_envio\_datos valdrá 1 siempre que no se encuentre ningun error en los datos recibidos anteriormente. Presuponemos que los datos son correctos, y despues verificamos si algún campo estaba en blanco. En caso de que se localice algun campo en blanco, ponemos \$aceptar\_envio\_datos a 0.*

```
if ($nick == "") { $aceptar_envio_datos = '0'; $nick_error = 'El campo NICK no puede estar en blanco'; } // Para cada campo que se encuentre en blanco le definimos un mensaje de error, cada uno en una variable diferente.
```

```
if ($password == "") { $aceptar_envio_datos = '0'; $password_error = 'El campo PASSWORD no puede estar en blanco'; }
```

```
if ($nombre == "") { $aceptar_envio_datos = '0'; $nombre_error = 'El campo NOMBRE no puede estar en blanco'; }
```

```
if ($apellido1 == "") { $aceptar_envio_datos = '0'; $apellido1_error = 'El campo APELLIDO1 no puede estar en blanco'; }
```

```
if ($apellido2 == "") { $aceptar_envio_datos = '0'; $apellido2_error = 'El campo APELLIDO2 no puede estar en blanco'; }
```

```
if ($aceptar_envio_datos == '0') { include ('registro.php'); } // En este caso no podemos aceptar el registro, y volvemos a llevar al usuario a la página de registro, que se cargará poniendo los mensajes de error encontrados en el intento de registro anterior.
```

```
if ($aceptar_envio_datos == '1') { // En caso de que no haya problemas en los datos, se añade el usuario a la base de datos.
```

```
$sql = "INSERT INTO usuarios SET
    nick='$nick',
    nombre='$nombre',
    apellido1='$apellido1',
    apellido2='$apellido2',
    sexo='$sexo',
    edad='$edad',
    colegio='$colegio',
    password='$password'";
```

```
if (@mysql_query($sql)) {
```

```
    $nombre_variable = $nick;
```

```
    include('acreditado.php'); // Al realizar la consulta correctamente se envia al usuario a acreditado.php, y ya estará dado de alta en la base de datos y dentro de la zona para los usuarios.
```

```
    } else {
```

```
        $nick_error = '<p class="EstiloError">';
```

```
        $nick_error .= 'El nombre de usuario ya existe, elige otro';
```

```
        $nick_error .= '</p>';
```

```
        include('registro.php'); // Puede darse el caso de que el nick con el que el usuario se quiere registrar ya exista. Al ser el campo nick la clave primaria el acceso a la base de datos dará un error y el código nos llevará a este punto, donde lo
```

*devolveremos a la página de registro con el mensaje de error del nick indicando este hecho.*

```
}  
  }  
?>
```

## 2.6. acreditado.php

Cuando un usuario tiene cuenta de usuario y accede a la zona privada para los usuarios registrados es aquí donde viene.

```
<body bgcolor="#0099CC">  
<div id="Layer3" style="position:absolute; width:864px; height:76px; z-index:3; left:  
9px; top: 23px; font-size: 9px;">  
  <p> </p>  
</div>  
<p>&nbsp;</p>  
<p>&nbsp;</p>  
<p>&nbsp;</p>  
<div id="Capajuego" style="position:absolute; width:561px; height:295px; z-index:2;  
left: 165px; top: 251px; font-size: 9px;">  
  <p>  
    <object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"  
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#ve  
rsion=6,0,29,0" width="585" height="328">  
  <?php $temp = '<param name="movie"  
value="imagenes/mapa_usuario_activo.swf?nick=';  
    $temp .= $nombre_variable;  
    $temp .= "'>';  
    echo $temp;  
    ?> // El mapa en flash se cargará pasandole la variable $nombre_variable,  
para que sepa cual es el usuario que hay activo en el sistema.  
    <param name="quality" value="high">  
    <?php $temp = '<embed src="imagenes/mapa_usuario_activo.swf?nick=';  
    $temp .= $nombre_variable;  
    $temp .= ' quality="high"  
pluginspage="http://www.macromedia.com/go/getflashplayer" type="application/x-  
shockwave-flash" width="322" height="269"></embed>';  
    echo $temp;  
    ?>  
  </object>  
  <font color="#FFFFFF"> </font></p>  
</div>  
<div id="Layer1" style="position:absolute; width:849px; height:45px; z-index:5; left:  
30px; top: 161px; background-color: #0033CC; layer-background-color: #0033CC;  
border: 1px none #000000; background-image: url(imagenes/water-texture-blue.jpg);  
layer-background-image: url(imagenes/water-texture-blue.jpg);">  
  <span style="font-size: 9px"></span></div>
```

```

<div id="Layer2" style="position:absolute; width:834px; height:31px; z-index:4; left:
36px; top: 133px; background-color: #0000CC; layer-background-color: #0000CC;
border: 1px none #000000;">
<table width="830" border="0">
<tr>
<td width="577" class="EstiloFormulario"><?php
    if ($usuario_activo != NULL){
        $nombre_variable = $usuario_activo;
    }
    if ($nombre_variable != NULL){
        $texto = '<p class="EstiloFormulario">Bienvenido, ';
        $texto .= $nombre_variable;
        $texto .= '</p>';
        echo $texto; // Este script saluda al usuario y le sirve para saber que ha
accedido correctamente al sistema.

        } else { echo ('<p class="EstiloError">ERROR</p>'); } // Al acceder a la
página directamente indicando su dirección no hay usuario activo, y nos indica un
error.
?>
    <span class="Estilo9">    </span></td>
<td width="145"><span class="Estilo9">
    <?php
    $texto2 = '<a href="acreditado2.php?usuario_activo="';
    $texto2 .= $nombre_variable;
    $texto2 .= '" class="EstiloFormulario">Modificar datos usuario</a> </p>';
    echo $texto2;
    ?> // Hay un enlace a acreditado2.php, al que se le pasa el nombre del usuario de
forma explícita. Desde esta página podrá modificar sus datos.
    </span></td>
<td width="94"><span class="Estilo9">
    <?php
    $texto3 = '<a href="index.html" class="EstiloFormulario">Cerrar sesión</a></p>'; //
Cerrar sesión únicamente será ir a la página principal, con lo que se perderan todas
las variables activas y dejará de haber un usuario activo en el sistema.
    echo $texto3;
    ?>
    </span></td>
</tr>
</table>
</div>
<?php
    include('mensajes2.php'); // Se incluye un fichero que muestra las mejores
puntuaciones.
?>
</div>
<div id="Layer4" style="position:absolute; width:413px; height:29px; z-index:16; left:
44px; top: 209px;">
<?php

```

```

        include('mensajes.php'); // Se incluye un fichero que muestra un mensaje
aleatorio con datos sobre los usuarios de la plataforma.
?>
</div>
<div id="Layer5" style="position:absolute; width:604px; height:35px; z-index:7; left:
146px; top: 601px;">
    <div align="center"><a href="privacidad.php" class="Estilo9">Política
de privacidad / Condiciones de uso</a></div> // Enlace a un texto con la política de
privacidad de la plataforma.
</div>
<div id="Layer6" style="position:absolute; width:168px; height:26px; z-index:8; left:
14px; top: 59px;">
    <?php
        include('comentario_ver.php'); // Se incluye un fichero que muestra el tagboard
        $texto2 = '<a href="comentario_escribir.php?usuario_activo=';
        $texto2 .= $nombre_variable;
        $texto2 .= '" class="Estilo9">Escribir comentario</a></p>';
        echo $texto2; // Se enlaza a un fichero que permite escribir un nuevo mensaje
al usuario activo.
    ?>
</div>
</body>

```

## 2.7. mensajes.php

Este fichero muestra por pantalla un mensaje seleccionado al azar con datos sobre los usuarios de la plataforma.

```

<?php
include('crear_conexion_bd.php');

$provincias[0] = 'Alava'; // Se crea un vector con el nombre de las provincias de
españa para su uso a la hora de mostrar datos por pantalla, ya que en la base de datos
estas provincias estan asociadas a un número.
$provincias[1] = 'Albacete';
$provincias[2] = 'Alicante';
$provincias[3] = 'Almeria';
$provincias[4] = 'Asturias';
$provincias[5] = 'Avila';
$provincias[6] = 'A Coruña';
$provincias[7] = 'Badajoz';
$provincias[8] = 'Barcelona';
$provincias[9] = 'Burgos';
$provincias[10] = 'Caceres';
$provincias[11] = 'Cadiz';
$provincias[12] = 'Cantabria';
$provincias[13] = 'Castellon';
$provincias[14] = 'Ceuta';
$provincias[15] = 'Ciudad Real';
$provincias[16] = 'Cordoba';

```

```
$provincias[17] = 'Cuenca';
$provincias[18] = 'Girona';
$provincias[19] = 'Granada';
$provincias[20] = 'Guadalajara';
$provincias[21] = 'Guipuzkoa';
$provincias[22] = 'Huelva';
$provincias[23] = 'Huesca';
$provincias[24] = 'Jaen';
$provincias[25] = 'La Rioja';
$provincias[26] = 'Leon';
$provincias[27] = 'Lleida';
$provincias[28] = 'Lugo';
$provincias[29] = 'Madrid';
$provincias[30] = 'Malaga';
$provincias[31] = 'Melilla';
$provincias[32] = 'Murcia';
$provincias[33] = 'Navarra';
$provincias[34] = 'Ourense';
$provincias[35] = 'Palencia';
$provincias[36] = 'Pontevedra';
$provincias[37] = 'Salamanca';
$provincias[38] = 'Segovia';
$provincias[39] = 'Sevilla';
$provincias[40] = 'Soria';
$provincias[41] = 'Tarragona';
$provincias[42] = 'Teruel';
$provincias[43] = 'Toledo';
$provincias[44] = 'Valencia';
$provincias[45] = 'Valladolid';
$provincias[46] = 'Vizcaya';
$provincias[47] = 'Zamora';
$provincias[48] = 'Zaragoza';
$provincias[49] = 'Mallorca';
$provincias[50] = 'Ibiza';
$provincias[51] = 'Menorca';
$provincias[52] = 'Formentera';
$provincias[53] = 'Las Palmas';
$provincias[54] = 'Gran Canaria';
$provincias[56] = 'Lanzarote';
$provincias[57] = 'Tenerife';
$provincias[58] = 'La Gomera';
$provincias[59] = 'La Palma';
$provincias[60] = 'El Hierro';
```

```
rand ((float) microtime() * 10000000); // Se inicializa el generador de números aleatorios utilizando el tiempo del microprocesador.
```

```
$frase = rand(1,16); // Se escoge un número al azar entre 1 y 16, que será el número del mensaje a representar.
```

```
if ($frase == '1'){
```

```

$ssql = "SELECT provincia, count(provincia) as cuenta FROM usuarios GROUP BY
provincia"; // Consulta que selecciona de que provincia hay más usuarios.
$salida_sql = @mysql_query($ssql);
$row = mysql_fetch_array($salida_sql);
$temp = $row['cuenta'];
$temp2 = $row['provincia'];

while($row = mysql_fetch_array($salida_sql)) {
    if($temp <= $row['cuenta']){
        $temp = $row['cuenta'];
        $temp2 = $row['provincia'];
    }
}
$message = "Hay más usuarios de $provincias[$temp2] que de ningún otro lado";
} else if ($frase == '2'){
    $ssql = "SELECT count(id) FROM juego1"; // Selecciona el número de partidas
jugadas en el juego 1
    $salida_sql = @mysql_query($ssql);
    $salida_sql_vector = mysql_fetch_array($salida_sql);
    $juego1 = $salida_sql_vector[0];
    $ssql = "SELECT count(id) FROM juego2";
    $salida_sql = @mysql_query($ssql);
    $salida_sql_vector = mysql_fetch_array($salida_sql);
    $juego2 = $salida_sql_vector[0];
    $ssql = "SELECT count(id) FROM juego3";
    $salida_sql = @mysql_query($ssql);
    $salida_sql_vector = mysql_fetch_array($salida_sql);
    $juego3 = $salida_sql_vector[0];

    if($juego1 >= $juego2){
        if(juego1 >= $juego3){
            $mensaje = "El juego más jugado es Scott, el pájaro que no sabe
volar";
        } else if ($juego2 >= $juego3){
            $mensaje = "El juego más jugado es Igor, el pingüino defensor";
        } else { $mensaje = "El juego más jugado es Bill, el cazador de estrellas";
        }
    }
} // Muestra el juego del que se ha jugado mayor número de partidas.

} else if ($frase == '3'){
    $ssql = "SELECT * FROM juego1 ORDER BY puntuacion DESC LIMIT 1";
// Se selecciona al usuario con mayor puntuación del juego de Scott
    $salida_sql = @mysql_query($ssql);
    $row = mysql_fetch_array($salida_sql);
    $nick = $row[nick];
    $mensaje = "El que mejor ha enseñado a volar a Scott es $nick";
} else if ($frase == '4'){
    $ssql = "SELECT * FROM juego2 ORDER BY puntuacion DESC LIMIT 1";
// Se selecciona al usuario con mayor puntuación del juego de Igor
    $salida_sql = @mysql_query($ssql);

```

```

    $row = mysql_fetch_array($salida_sql);
    $nick = $row[nick];
    $mensaje = "$nick ha sido el mejor ayudando a Igor defendiendo su poblado";
} else if ($frase == '5'){
    $sql = "SELECT * FROM juego3 ORDER BY puntuacion DESC LIMIT 1";
// Se selecciona al usuario con mayor puntuación del juego de Bill
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $nick = $row[nick];
    $mensaje = "Nadie como $nick para ayudar a Bill a cazar estrellas";
} else if ($frase == '6'){
    $sql = "SELECT * FROM juego1 ORDER BY puntuacion DESC LIMIT 1";
// Se selecciona al mejor jugador del juego de Scott
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $nick = $row[nick];
    $sql = "SELECT * FROM usuarios WHERE nick='$nick'";
// Se obtiene el sexo y la provincia del mejor jugador del juego de Scott
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $provincia = $row[provincia];
    $sexo = $row[sexo];
    if ($sexo == '0'){
        $mensaje = "Una chica de $provincias[$provincia] ha mostrado ser una
experta enseñando a volar a Scott";
    } else {
        $mensaje = "Un chico de $provincias[$provincia] sabe bien como hacer
que Scott vuele lo mejor que pueda";
    } // Se muestra un mensaje diferente sobre el mejor jugador del juego de Scott
en base a su sexo.
} else if ($frase == '7'){
    $sql = "SELECT * FROM juego2 ORDER BY puntuacion DESC LIMIT 1";
// Se selecciona al mejor jugador del juego de Igor
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $nick = $row[nick];
    $sql = "SELECT * FROM usuarios WHERE nick='$nick'";
// Se obtiene el sexo y la provincia del mejor jugador del juego de Igor
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $provincia = $row[provincia];
    $sexo = $row[sexo];
    if ($sexo == '0'){
        $mensaje = "La que mejor ha ayudado a Igor a defender su poblado vive
en $provincias[$provincia] ";
    } else {
        $mensaje = "Hay un chico en $provincias[$provincia] que es todo un
profesional defendiendo el poblado con Igor";
    } // Se muestra un mensaje diferente sobre el mejor jugador del juego de Igor
en base a su sexo.

```



```

} else if ($frase == '8'){
    $sql = "SELECT * FROM juego3 ORDER BY puntuacion DESC LIMIT 1";
// Se selecciona al mejor jugador del juego de Bill
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $nick = $row[nick];
    $sql = "SELECT * FROM usuarios WHERE nick='$nick'";
// Se obtiene el sexo y la provincia del mejor jugador del juego de Bill
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $provincia = $row[provincia];
    $sexo = $row[sexo];
    if ($sexo == '0'){
        $mensaje = "En $provincias[$provincia] hay una gran cazadora de
estrellas llamada $nick";
    } else {
        $mensaje = "Se rumorea que el que ha logrado que Bill coja más estrellas
vive en $provincias[$provincia]";
        // Se muestra un mensaje diferente sobre el mejor jugador del juego de Scott
en base a su sexo.
    }
} else if ($frase == '9'){
    $sql = "SELECT nick, count(*) as cuenta FROM juego1 GROUP BY nick order
by cuenta desc"; // Se selecciona al jugador que más partidas ha jugado al juego de
Scott
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $nick = $row[nick];
    $mensaje = "$nick lleva jugadas $row[cuenta] partidas para enseñar a jugar a
Scott a volar";
} else if ($frase == '10'){
    $sql = "SELECT nick, count(*) as cuenta FROM juego2 GROUP BY nick order
by cuenta desc"; // Se selecciona al jugador que más partidas ha jugado al juego de
Igor
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $nick = $row[nick];
    $mensaje = "$nick es el que más veces ha intentado salvar el poblado de Igor";
} else if ($frase == '11'){
    $sql = "SELECT nick, count(*) as cuenta FROM juego3 GROUP BY nick order
by cuenta desc"; // Se selecciona al jugador que más partidas ha jugado al juego de
Bill
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $nick = $row[nick];
    $mensaje = "Bill tiene un gran amigo llamado $nick que no para de jugar con
el";
} else if ($frase == '11'){
    $sql = "SELECT * FROM juego1 ORDER BY puntuacion DESC LIMIT 1";
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);

```

```

$nick = $row[nick];
$sql = "SELECT * FROM usuarios WHERE nick='$nick'";
$salida_sql = @mysql_query($sql);
$row = mysql_fetch_array($salida_sql);
$provincia = $row[provincia];
$colegio = $row[colegio];
$mensaje = "El gran campeón del juego de Scott va al colegio $colegio en
$provincias[$provincia], ¿Irás a tu clase?"; // Se muestra el colegio del mejor jugador
del juego de Scott
} else if ($frase == '12'){
    $sql = "SELECT * FROM juego2 ORDER BY puntuacion DESC LIMIT 1";
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $nick = $row[nick];
    $sql = "SELECT * FROM usuarios WHERE nick='$nick'";
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $provincia = $row[provincia];
    $colegio = $row[colegio];
    $mensaje = "Ademas de haber logrado de ayudar a Igor mejor que nadie, $nick
estudia en el colegio $colegio en $provincias[$provincia]"; // Se muestra el colegio del
mejor jugador del juego de Igor
} else if ($frase == '13'){
    $sql = "SELECT * FROM juego3 ORDER BY puntuacion DESC LIMIT 1";
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $nick = $row[nick];
    $sql = "SELECT * FROM usuarios WHERE nick='$nick'";
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $provincia = $row[provincia];
    $colegio = $row[colegio];
    $mensaje = "El mejor cazador de estrellas va al colegio en
$provincias[$provincia]";
    if (!$colegio == ""){
        $mensaje .= " y su colegio es $colegio";
    } // Se muestra el colegio del mejor jugador del juego de Bill
} else if ($frase == '14'){
    $sql = "SELECT * FROM juego1 ORDER BY puntuacion DESC LIMIT 1";
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $nick = $row[nick];
    $puntuacion = $row[puntuacion];
    $mensaje = "$nick es el mejor jugador de Scott, el pájaro que no sabe volar, con
$puntuacion puntos"; // Se muestra el mejor jugador del juego de Scott con su
puntuación
} else if ($frase == '15'){
    $sql = "SELECT * FROM juego2 ORDER BY puntuacion DESC LIMIT 1";
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);

```

```

    $nick = $row[nick];
    $puntuacion = $row[puntuacion];
    $mensaje = "¿Serás capaz de superar los $puntuacion puntos que ha conseguido
$nick ayudando a Igor?"; // Se muestra el mejor jugador del juego de Igor con su
puntuación
} else if ($frase == '16'){
    $sql = "SELECT * FROM juego3 ORDER BY puntuacion DESC LIMIT 1";
    $salida_sql = @mysql_query($sql);
    $row = mysql_fetch_array($salida_sql);
    $nick = $row[nick];
    $puntuacion = $row[puntuacion];
    $mensaje = "¿Intenta superar los $puntuacion puntos que ha hecho $nick
jugando con Bill!";
} // Se muestra el mejor jugador del juego de Bill con su puntuación

    $msj = '<table border="0"><tr><td>';
    $msj .= '<span class="Estilo9">';
    $msj .= $mensaje;
    $msj .= '</span></td></tr></table>';
    echo $msj;

?>

```

## 2.8. mensajes2.php

Se muestra por pantalla la tabla con las mejores puntuaciones de los juegos.

```

<?php
include('crear_conexion_bd.php');
srand ((float) microtime() * 10000000);
$tabla = rand(1,3);

    $sql = "SELECT * FROM juego1 ORDER BY puntuacion DESC LIMIT 8";
    $salida_sql = @mysql_query($sql);

    echo '<table width="134" border="0"><tr bgcolor="#0000CC"><td
class="EstiloFormulario">Scott, el pájaro que no sabe volar</td></tr></table>';
    echo '<table width="134" border="0" bgcolor="#B4D5E9">';
    while($row = mysql_fetch_array($salida_sql)){
        $texto = '<tr><td class="Estilo9">';
        $texto .= $row["nick"];
        $texto .= '</td><td class="Estilo9bis">';
        $texto .= $row["puntuacion"];
        $texto .= '</td></tr>';
        echo $texto; // Se muestra la tabla de puntuaciones del juego 1 con los
campos usuario y puntuación.
    }

    echo '</table>';
    echo '<br>';
    $sql = "SELECT * FROM juego2 ORDER BY puntuacion DESC LIMIT 8";

```

```

$salida_sql = @mysql_query($sql);

echo '<table width="134" border="0"><tr bgcolor="#0000CC"><td
class="EstiloFormulario">Igor, el pingüino defensor</td></tr></table>';
echo '<table width="134" border="0" bgcolor="#B4D5E9">';
while($row = mysql_fetch_array($salida_sql)){
$texto = '<tr><td class="Estilo9">';
$texto .= $row["nick"];
$texto .= '</td><td class="Estilo9bis">';
$texto .= $row["puntuacion"];
$texto .= '</td></tr>';
echo $texto;
}
echo '</table>';
echo '<br>';
$sql = "SELECT * FROM juego3 ORDER BY puntuacion DESC LIMIT 8";
$salida_sql = @mysql_query($sql);

echo '<table width="134" border="0"><tr bgcolor="#0000CC"><td
class="EstiloFormulario">Bill, el cazador de estrellas</td></tr></table>';
echo '<table width="134" border="0" bgcolor="#B4D5E9">';
while($row = mysql_fetch_array($salida_sql)){
$texto = '<tr><td class="Estilo9">';
$texto .= $row["nick"];
$texto .= '</td><td class="Estilo9bis">';
$texto .= $row["puntuacion"];
$texto .= '</td></tr>';
echo $texto;
}
echo '</table>';
?>

```

## 2.9. comentario\_escribir.php

Este fichero muestra por pantalla un formulario en el que el usuario podrá escribir un mensaje de un máximo de 200 letras que se mostrará a los demás usuarios una vez almacenado en la base de datos.

```

<script language="javascript" type="text/javascript">
function limitText(limitField, limitCount, limitNum) {
    if (limitField.value.length > limitNum) {
        limitField.value = limitField.value.substring(0, limitNum);
    } else {
        limitCount.value = limitNum - limitField.value.length;
    }
}
</script> // Función que limita el número de caracteres que podrán escribirse en la
caja de texto.

```

```

<body bgcolor="#0099CC">
<p class="Estilo10">&nbsp;</p>
<p class="Estilo10">&nbsp;</p>
<div id="Layer3" style="position:absolute; width:864px; height:76px; z-index:3; left:
9px; top: 23px; font-size: 9px;">
  <p> </p>
</div>
<p class="Estilo10">&nbsp;</p>
<div id="Capajuego" style="position:absolute; width:561px; height:295px; z-index:2;
left: 165px; top: 251px; font-size: 9px;">
  <p>&nbsp;</p>
  <form method="post" action="comentario_escribir_envio.php">
    <div align="center"> </div>
    <div align="justify"></div>
    <div align="center"><br>
      <span class="Estilo9"> </span>
<textarea name="limitedtextarea"
onKeyDown="limitText(this.form.limitedtextarea,this.form.countdown,200);"
  onKeyUp="limitText(this.form.limitedtextarea,this.form.countdown,200);"
cols="30" rows="5"> // En el momento que se escriba una letra se verificará que el
mensaje no supere los 200 caracteres utilizando la función en javascript anterior.
  </textarea>
  <br>
<font size="1">(Puedes escribir 200 caracteres)<br>
</font>
  <?php
    $texto = '<input type="hidden" name="nick" value="';
    $texto .= $usuario_activo;
    $texto .= "'>';
    echo $texto; // Junto al formulario se enviará el nombre del usuario en forma
de variable oculta.
  ?>
  <input name="submit" type="submit" value="  Enviar  ">
</div>
</form>
<p><font color="#FFFFFF"> </font></p>
</div>
<div id="Layer1" style="position:absolute; width:849px; height:45px; z-index:5; left:
30px; top: 165px; background-color: #0033CC; layer-background-color: #0033CC;
border: 1px none #000000; background-image: url(imagenes/water-texture-blue.jpg);
layer-background-image: url(imagenes/water-texture-blue.jpg);">
  <span style="font-size: 9px"></span></div>
<div id="Layer2" style="position:absolute; width:834px; height:31px; z-index:4; left:
37px; top: 135px; background-color: #0000CC; layer-background-color: #0000CC;
border: 1px none #000000;">
  <form name="form1" method="post" action="login.php">
    <table width="809" border="0">
      <tr>

```

```

        <td width="715"><span class="EstiloFormulario">Desde aqu&iacute; podras
        escribir alg&uacute;n comentario para que todo aquel que entre en la web lo lea.
        </span></td>
        <td width="84"><span class="Estilo9"><a href="acreditado.php"
        class="EstiloFormulario">Volver</a></span></td>
    </tr>
</table>
</form>
<span style="font-size: 9px">
</span></div>
<p>&nbsp;</p>
<p><span style="font-size: 9px"> </span></p>
</body>

```

## 2.10. comentario\_escribir\_envio.php

El fichero comentario\_escribir\_envio.php se encarga de almacenar en la base de datos el mensaje recibido por comentario\_escribir.php

```

<?php
include('crear_conexion_bd.php');

$nick = $_POST["nick"]; // Se recibe el nick del usuario, enviado de forma oculta
$mensaje = $_POST["limitedtextarea"]; // Se recibe el mensaje del usuario.

$sql = "INSERT INTO mensajes SET
        nick='$nick',
        mensaje='$mensaje';
if (@mysql_query($sql)) {
    $nombre_variable = $nick;
} else {
    echo "error: ";
    echo mysql_error();
} // Se inserta el mensaje en la base de datos.
include('acreditado.php');
?>

```

## 2.11 comentario\_leer.php

Mostramos por pantalla una tabla en la que se verán los 10 últimos mensajes escritos en el tagboard.

```

<body>
<table width="134" border="0">
  <tr bgcolor="#0000CC">
    <td class="EstiloFormulario">&Uacute;ltimos comentarios</td> // Nombre de la
    tabla.
  </tr>
</tr>

```

```

<td>
  <?php
    include('crear_conexion_bd.php');

    $sql = "SELECT * FROM mensajes ORDER BY id DESC LIMIT 10";
    $salida_sql = @mysql_query($sql);
    while($row = mysql_fetch_array($salida_sql)){
      $texto = '<tr><td class="EstiloMensaje" bgcolor="#B4D5E9"><strong
class="Estilo9">';
        $texto .= $row["nick"];
        $texto .= ': </strong><span class="Estilo9bis">';
        $texto .= $row["mensaje"];
        $texto .= '</span></td></tr><br>';
        echo $texto;
      }
    ?>
  </td>
</tr>
</table>
</body>

```

## 2.12. acreditado2.php

Desde acreditado2.php un usuario podrá modificar sus datos personales.

```

<body bgcolor="#0099CC">

<p>&nbsp;</p>

<p>&nbsp;</p>

<div id="Layer3" style="position:absolute; width:864px; height:76px; z-index:3; left:
9px; top: 23px; font-size: 9px;">

  <p> </p>

</div>

<div id="Layer2" style="position:absolute; width:834px; height:31px; z-index:4; left:
36px; top: 133px; background-color: #0000CC; layer-background-color: #0000CC;
border: 1px none #000000;">

<?php

```

```

    $nombre_variable = $_GET["usuario_activo"];

    if ($nombre_variable != NULL){

        $texto = '<p class="EstiloFormulario">¡Hola ';

        $texto .= $nombre_variable;

        $texto .= '! En esta sección de la web podrás modificar tus datos personales';

        $texto .= '</p>';

        echo $texto;

        } else { echo ("ERROR"); }

?>
</div>

<div id="Layer1" style="position:absolute; width:849px; height:45px; z-index:5; left:
30px; top: 165px; background-color: #0033CC; layer-background-color: #0033CC;
border: 1px none #000000; background-image: url(imagenes/water-texture-blue.jpg);
layer-background-image: url(imagenes/water-texture-blue.jpg);"> <span style="font-
size: 9px"></span></div>

<p>&nbsp;</p>

<div id="Capajuego" style="position:absolute; width:561px; height:295px; z-index:2;
left: 165px; top: 226px; font-size: 9px;">

<?php

    include('crear_conexion_bd.php');

    $nick = $_GET["usuario_activo"];

    $sql = "SELECT * FROM usuarios WHERE nick = '$nick'";

    $salida_sql = @mysql_query($sql);

    while($row = mysql_fetch_array($salida_sql)) {

        $password = $row["password"];

        $nombre = $row["nombre"];

```



```
$apellido1 = $row["apellido1"];
```

```
$apellido2 = $row["apellido2"];
```

```
$edad = $row["edad"];
```

```
$sexo = $row["sexo"];
```

```
$colegio = $row["colegio"];
```

```
$localidad = $row["localidad"];
```

**} // Almacenamos en tantas variables como datos almacenamos del usuario los datos actuales del usuario. A partir de aquí creamos un formulario completamente en php (imprimiendo las etiquetas html mediante echo) para mostrar estos datos y permitir al usuario que cambie los datos que desee cambiar. También impedimos que el usuario pueda cambiar el campo nick.**

```
$texto = '<form name="form1" id="form1" method="post"
action="usuario_modificar_usuario_2.php?usuario_activo='; // El formulario recién
creado llamará al script usuario_modificar_usuario_2.php, que se encargará de la
actualización de la base de datos.
```

```
$texto .= $nick;
```

```
$texto .= ">";
```

```
echo $texto;
```

```
echo ('<table width="200" border="0">');
```

```
echo ('<tr>');
```

```
echo ('<td>');
```

```
echo '<span class="Estilo9">Nombre de usuario</span>';
```

```
echo ('</td>');
```

```
echo ('<td>');
```

```
echo ('<span class="Estilo9">');
```

```
echo $nick;
```

```
$nick2 = $nick;
```

```
echo ('</span>');
```

```

echo ('</td>');

echo ('<tr>');

echo ('<td>');

echo '<br>';

echo '<span class="Estilo9">Contrase&ntilde;a:</span>';

echo ('</td>');

echo ('<td>');

echo ' <input name="password2" type="password" id="password2" value="";

echo $password;

echo "/>';

echo ('</td>');

echo ('<tr>');

echo ('<td>');

echo '<br>';

echo '<span class="Estilo9">Nombre:</span>';

echo ('</td>');

echo ('<td>');

echo ' <input name="nombre2" type="text" id="nombre2" value="";

echo $nombre;

echo "/>';

echo ('</td>');

echo ('<tr>');

echo ('<td>');

echo '<br>';

echo '<span class="Estilo9">Primer apellido:</span>';

```

```
echo ('</td>');

echo ('<td>');

echo ' <input name="apellido12" type="text" id="apellido12" value="";

echo $apellido1;

echo "/>';

echo ('</td>');

echo ('<tr>');

echo ('<td>');

echo '<br>';

echo '<span class="Estilo9">Segundo apellido:</span>';

echo ('</td>');

echo ('<td>');

echo ' <input name="apellido22" type="text" id="apellido22" value="";

echo $apellido2;

echo "/>';

echo ('</td>');

echo ('<tr>');

echo ('<td>');

echo '<br>';

echo '<span class="Estilo9">Sexo:</span>';

echo ('</td>');

echo ('<td>');

echo ' <select name="sexo2" id="sexo2" value="";

echo $sexo;
```

```

echo '>';

echo '<option value="1">Chico</option>';

echo '<option value="0">Chica </option>';

echo '</select>';

echo ('</td>');

echo ('<tr>');

echo ('<td>');

echo '<br>';

echo '<span class="Estilo9">Edad:</span>';

echo ('</td>');

echo ('<td>');

echo '<input name="edad2" type="text" id="edad2" value="";

echo $edad;

echo "/>';

echo ('</td>');

    echo ('<tr>');

echo ('<td>');

echo '<br>';

echo '<span class="Estilo9">Localidad:</span>';

echo ('</td>');

echo ('<td>');

echo '<select name="localidad2" id="localidad2" value="";

echo $localidad;

echo '>';

echo '<option value="1">Valencia</option>';

```

```

echo '<option value="0">Resto del mundo</option>';

echo '</select>';

    echo ('</td>');

    echo ('<tr>');

    echo ('<td>');

    echo '<br>';

    echo '<span class="Estilo9">Colegio</span>';

    echo ('</td>');

    echo ('<td>');

    echo ' <input name="colegio2" type="text" id="colegio2" value="";

    echo $colegio;

    echo ";</>';

    echo ('</td>');

    echo ('<tr>');

    echo ('<td>');

    echo '<input name="modificar" type="submit" id="modificar" value="Enviar"
/> ';

    echo ('</td>');

    echo ('<td>');

    $texto2 = '<a href="acreditado.php?nombre_variable=';

    $texto2 .= $nick2;

    $texto2 .= '" class="Estilo9">Volver</a> </p>';

    echo $texto2; // Añadimos un botón volver, para el usuario que haya entrado
aquí por error.

    echo ('</td>');

```

```

        echo ('<tr>');

        echo ('<td>');

        echo ('</table>');

        echo '</form>';

?>

<p><font color="#FFFFFF"> </font></p>

</div>

<p>&nbsp;</p>

<p>&nbsp;</p>

<p>&nbsp;</p>

</body>

```

### 2.13. usuario\_modificar\_usuario\_2.php

El último fichero que forma parte de la zona de los usuarios, encargado de efectuar la conexión a la base de datos a la hora de modificar los datos de un usuario.

```

<?php

include ('crear_conexion_bd.php');

$nick = $usuario_activo;

$password = $_POST["password2"];

$nombre = $_POST["nombre2"];

$apellido1 = $_POST["apellido12"];

$apellido2 = $_POST["apellido22"];

$edad = $_POST["edad2"];

$sexo = $_POST["sexo2"];

$colegio = $_POST["colegio2"];

$localidad = $_POST["localidad2"];

```

**// En \$sql creamos una consulta a la base de datos en la que se actualizan sus datos con los nuevos valores que hayamos puesto.**

```
$sql = "UPDATE usuarios SET password = "";
```

```
$sql .= $password;
```

```
$sql .= ", nombre = "";
```

```
$sql .= $nombre;
```

```
$sql .= ", apellido1 = "";
```

```
$sql .= $apellido1;
```

```
$sql .= ", apellido2 = "";
```

```
$sql .= $apellido2;
```

```
$sql .= ", colegio = "";
```

```
$sql .= $colegio;
```

```
$sql .= ", provincia = "";
```

```
$sql .= $localidad;
```

```
$sql .= ", sexo = "";
```

```
$sql .= $sexo;
```

```
$sql .= ", edad = "";
```

```
$sql .= $edad;
```

```
$sql .= "";
```

```
$sql .= " WHERE nick = "";
```

```
$sql .= $nick;
```

```
$sql .= "";
```

```
if (@mysql_query($sql)) {
```

```
  $nombre_variable = $nick;
```

```
include('acreditado.php'); // Si la actualización se hace de forma correcta, se devuelve al usuario a la zona de entrada de los usuarios registrados.
```

```
} else {
```

```
    echo('<p>Error al modificar los datos: ' .
```

```
        mysql_error() . '</p>');
```

```
}
```

```
?>
```



### 3. Implementación de la zona de administración

#### 3.1. admin.php

Únicamente tenemos una capa donde el usuario introducirá sus datos y serán enviados a administrador2.php para que verifique si son correctos.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link rel="stylesheet" href="oceanografico.css" type="text/css" />
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);
//-->
</script>
<style type="text/css">
<!--
.Estilo13 {color: #FFFFFF}
-->
</style>
</head>

<body>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>&nbsp;</p>
<div id="Layer3" style="position:absolute; width:496px; height:215px; z-index:3; left:
126px; top: 128px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;">
  <form name="form1" method="post" action="administrador2.php">
    <p align="center" class="EstiloMensaje">
      <?php $imprimir_mensaje = '<p align="center" class="Estilo9 Estilo13">';
      $imprimir_mensaje .= $error;
      $imprimir_mensaje .= '</p>';
```

```

        echo $imprimir_mensaje;
    ?>
</p>

<p align="center" class="EstiloMensaje">Bienvenido a la zona para administradores
de la plataforma:<br>
&iexcl;Un chapuz&oacute;n de diversi&oacute;n! </p>
<table width="200" border="0" align="center">
    <tr>
        <td class="EstiloMensaje">Usuario</td>
        <td><input name="nick" type="text" id="nick"></td>
    </tr>
    <tr>
        <td class="EstiloMensaje">Contrase&ntilde;a</td>
        <td><input name="password" type="password" id="password"></td>
    </tr>
</table>

<p align="center">
    <input type="submit" name="Submit" value="Enviar">
</p>
</form></div>
</body>
</html>

```

### 3.2. administrador2.php

Este script lo utilizaremos para verificar que el administrador que ha introducido su contraseña realmente existe. En caso de exista, se le permitirá el acceso a las diferentes opciones de la zona de administración.

```

<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title><?php

    include('crear_conexion_bd.php');

    $nick = $_POST["nick"];
    $password = $_POST["password"];

    $sql = "SELECT count(*) FROM administrador WHERE nick = '$nick' AND
password = '$password'";

    $salida_sql = @mysql_query($sql);
    $salida_sql_vector = mysql_fetch_array($salida_sql);
    $temp = $salida_sql_vector[0];

    if ($temp == '1') {
        $nombre_variable = $nick;
        include('administrador3.php');
    }
}

```

```

    } else {
        $error = "Error en el acceso, vuelve a intentarlo";
        include('admin.php');
    }
?>

```

### 3.3. administrador3.php

Aquí encontramos cuatro capas, una de ellas la cabecera, otra el menú, otra vacía, que será usada para poner las opciones del menú y una cuarta conteniendo una imagen por razones puramente estéticas.

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link rel="stylesheet" href="oceanografico.css" type="text/css" />
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
    if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
        document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
    else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);
//-->
</script>
<style type="text/css">
<!--
.Estilo13 {color: #FFFFFF}
-->
</style>
</head>

<body>
<p>&nbsp;</p>
<div id="Layer1" style="position:absolute; width:740px; height:81px; z-index:5;
background-image: url(cabecera%20administracion.jpg); layer-background-image:
url(cabecera%20administracion.jpg); border: 1px none #000000; left: 25px; top:
25px;"></div>
<div id="Layer2" style="position:absolute; width:223px; height:191px; z-index:2; left:
39px; top: 142px; background-color: #FF3300; layer-background-color: #FF3300;"
class="Estilo9">

```

```

<blockquote>
  <p align="left" class="Estilo13"><a
href="administrador_opcion_crear_administrador.php" class="Estilo9">
  Crear administrador </a></p>
  <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_administradores.php" class="Estilo9">Listar
administradores</a></p>
  <p align="left" class="Estilo13"><a
href="administrador_opcion_borrar_administrador.php" class="Estilo9">Borrar
administrador </a></p>
  <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_usuarios.php" class="Estilo9"> Listar
usuarios</a></p>
  <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_puntuaciones.php" class="Estilo9">Listar
puntuaciones</a> </p>
  <p align="left" class="Estilo13"><a
href="administrador_opcion_modificar_usuario.php" class="Estilo9"> Modificar
usuario</a> </p>
  <p align="left" class="Estilo13"> <a
href="administrador_opcion_borrar_usuario.php" class="Estilo9">Borrar usuario</a>
</p>
  <p class="Estilo13"><a href="administrador_opcion_crear_bases_de_datos.php"
class="Estilo9"> Crear bases de datos</a> </p>
  <p><span class="Estilo13"><a
href="administrador_opcion_destruir_bases_de_datos.php" class="Estilo9">
  Destruir bases de datos</a></span> </p>
</blockquote>
</div>
<div id="Layer3" style="position:absolute; width:496px; height:215px; z-index:3; left:
239px; top: 145px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;">
<?php
  $imprimir_mensaje = '<p align="center" class="Estilo9 Estilo13">';
  $imprimir_mensaje .= $mensaje;
  $imprimir_mensaje .= '</p>';
  echo $imprimir_mensaje;
  ?> // Algunas opciones del menú envían un mensaje a la página principal con
los resultados de sus acciones, que serán impresos por este bloque de código.
</div>
<div id="Layer4" style="position:absolute; width:731px; height:26px; z-index:1; left:
30px; top: 113px; background-color: #FF3300; layer-background-color: #FF3300;
border: 1px none #000000; background-image: url(tira.jpg); layer-background-image:
url(tira.jpg);">
  <table width="100%" height="32" border="0">
    <tr>
      <td width="65%">&nbsp;   </td>
      <td width="35%"><a href="admin.php" class="Estilo9">Cerrar
sesi&oacute;n</a></td> // Botón cerrar sesión.
    </tr>
  </table>

```

```
</table>
</div>
</body>
</html>
```

### 3.4. administrador\_opcion\_crear\_administrador.php

Nuevamente existen las cuatro capas anteriores, pero en este caso la capa anteriormente vacía ha sido rellena con un formulario que nos da la posibilidad de crear nuevos administradores.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link rel="stylesheet" href="oceanografico.css" type="text/css" />
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);
//-->
</script>
<style type="text/css">
<!--
.Estilo13 {color: #FFFFFF}
.Estilo14 {color: #000000}
-->
</style>
</head>

<body>
<div id="Layer2" style="position:absolute; width:223px; height:191px; z-index:2; left:
38px; top: 142px; background-color: #FF3300; layer-background-color: #FF3300;"
class="Estilo9">
  <blockquote>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_crear_administrador.php" class="Estilo9">
      Crear administrador </a></p>
    <p align="left" class="Estilo13"><a
```

```

href="administrador_opcion_listar_administradores.php" class="Estilo9">Listar
administradores</a></p>
<p align="left" class="Estilo13"><a
href="administrador_opcion_borrar_administrador.php" class="Estilo9">Borrar
administrador </a></p>
<p align="left" class="Estilo13"><a
href="administrador_opcion_listar_usuarios.php" class="Estilo9">
Listar usuarios</a></p>
<p align="left" class="Estilo13"><a
href="administrador_opcion_listar_puntuaciones.php" class="Estilo9">Listar
puntuaciones</a> </p>
<p align="left" class="Estilo13"><a
href="administrador_opcion_modificar_usuario.php" class="Estilo9">
Modificar usuario</a> </p>
<p align="left" class="Estilo13"> <a
href="administrador_opcion_borrar_usuario.php" class="Estilo9">Borrar
usuario</a> </p>
<p class="Estilo13"><a href="administrador_opcion_crear_bases_de_datos.php"
class="Estilo9">
Crear bases de datos</a> </p>
<p><span class="Estilo13"><a
href="administrador_opcion_destruir_bases_de_datos.php" class="Estilo9">
Destruir bases de datos</a></span> </p>
</blockquote>
</div>
<div id="Layer3" style="position:absolute; width:455px; height:215px; z-index:3; left:
239px; top: 145px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;">
<form name="form1" method="post" action="administrador_creacion_cuenta.php">
// Se crea un formulario en el que pondremos el nick y la contraseña de un nuevo
administrador.
<p align="center" class="Estilo9">&nbsp;</p>
<p align="center" class="Estilo9 Estilo14">Rellena los siguientes campos para crear
un nuevo administrador:</p>
<table width="200" border="0" align="center">
<tr>
<td><div align="center"><span class="Estilo9
Estilo14">Nombre:</span></div></td>
<td><span class="Estilo9">
<input name="nick" type="text" id="nick">
</span></td>
</tr>
<tr>
<td><div align="center"><span class="Estilo9
Estilo14">Contrase&ntilde;a:</span></div></td>
<td><span class="Estilo9">
<input name="password" type="password" id="password2">
</span></td>
</tr>
<tr>

```

```

<td><div align="center"></div></td>
<td><span class="Estilo9">
  <input name="enviar" type="submit" id="enviar2" value="Enviar">
</span></td>
</tr>
</table>
<?php
  $imprimir_mensaje = '<p align="center" class="EstiloError">';
  $imprimir_mensaje .= $mensaje;
  $imprimir_mensaje .= '</p>';
  echo $imprimir_mensaje;
?>

<p align="center" class="Estilo9">&nbsp;</p>
<p class="Estilo9">&nbsp;</p>
</form>
</div>
<div id="Layer4" style="position:absolute; width:732px; height:26px; z-index:4; left:
29px; top: 119px; background-color: #FF3300; layer-background-color: #FF3300;
border: 1px none #000000; background-image: url(tira.jpg); layer-background-image:
url(tira.jpg);"></div>
<div id="Layer1" style="position:absolute; width:740px; height:81px; z-index:5;
background-image: url(cabecera%20administracion.jpg); layer-background-image:
url(cabecera%20administracion.jpg); border: 1px none #000000; left: 25px; top:
25px;"></div>
</body>
</html>

```

### 3.5. administrador\_opcion\_listar\_administradores.php

Opción que permite listar los usuarios existentes en el sistema con todos sus datos personales, y la opción para imprimir estos resultados.

```

<body>
<div id="Layer2" style="position:absolute; width:223px; height:191px; z-index:2; left:
38px; top: 142px; background-color: #FF3300; layer-background-color: #FF3300;"
class="Estilo9">
  <blockquote>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_crear_administrador.php" class="Estilo9">
      Crear administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_administradores.php" class="Estilo9">Listar
      administradores</a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_borrar_administrador.php" class="Estilo9">Borrar
      administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_usuarios.php" class="Estilo9">
      Listar usuarios</a></p>

```

```

<p align="left" class="Estilo13"><a
href="administrador_opcion_listar_puntuaciones.php" class="Estilo9">Listar
puntuaciones</a> </p>
<p align="left" class="Estilo13"><a
href="administrador_opcion_modificar_usuario.php" class="Estilo9">
Modificar usuario</a> </p>
<p align="left" class="Estilo13"> <a
href="administrador_opcion_borrar_usuario.php" class="Estilo9">Borrar
usuario</a> </p>
<p class="Estilo13"><a href="administrador_opcion_crear_bases_de_datos.php"
class="Estilo9">
Crear bases de datos</a> </p>
<p><span class="Estilo13"><a
href="administrador_opcion_destruir_bases_de_datos.php" class="Estilo9">
Destruir bases de datos</a></span> </p>
</blockquote>
</div>
<div id="Layer3" style="position:absolute; width:704px; height:350px; z-index:3; left:
239px; top: 145px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;">
<div id="Layer5" style="position:absolute; width:200px; height:115px; z-index:6; left:
173px; top: 34px;">
<?php

include('crear_conexion_bd.php');

$sql = "SELECT count(nick) FROM administrador";
$salida_sql = @mysql_query($sql);
$salida_sql_vector = mysql_fetch_array($salida_sql);
$numero_usuarios = $salida_sql_vector[0];

$sql = "SELECT * FROM administrador";

$salida_sql = @mysql_query($sql);

echo '<table width="500" border="0">';
echo '<tr>';
echo '<td class="EstiloMensaje">Administrador</td>';
echo '<tr>';
echo '<td class="EstiloMensaje">-----</td></tr>';

while($row = mysql_fetch_array($salida_sql)) {
echo '<tr>';
echo '<td class="EstiloMensaje2">';
echo $row["nick"];
echo '</td>';
echo '<td>';
echo '<form name="form4" method="post"
action="administrador_borrar_administrador.php">';
$temp = '<input name="nick" type="hidden" value="';

```



```

                $temp .= $row["nick"];
                $temp .= "'>';
                echo $temp;
                echo '<input name="enviar" type="submit" id="enviar"
value="Borrar"></form>';
                echo '</td>';
                echo '</tr>';
            }

                echo '<tr>';
                echo '</table>';

?>
</div>
</div>
<div id="Layer4" style="position:absolute; width:731px; height:26px; z-index:1; left:
30px; top: 113px; background-color: #FF3300; layer-background-color: #FF3300;
border: 1px none #000000; background-image: url(tira.jpg); layer-background-image:
url(tira.jpg);">
    <table width="100%" height="32" border="0">
        <tr>
            <td width="65%">&nbsp;</td>
            <td width="35%"><a href="admin.php" class="Estilo9">Cerrar
sesi&oacute;n</a></td>
        </tr>
    </table>
</div>
<div id="Layer1" style="position:absolute; width:740px; height:81px; z-index:5;
background-image: url(cabecera%20administracion.jpg); layer-background-image:
url(cabecera%20administracion.jpg); border: 1px none #000000; left: 25px; top:
25px;"></div>
</body>

```

### 3.6. administrador\_opcion\_borrar\_administrador.php

Opción que nos permite borrar un administrador.

```

<body>
<div id="Layer2" style="position:absolute; width:223px; height:191px; z-index:2; left:
38px; top: 142px; background-color: #FF3300; layer-background-color: #FF3300;"
class="Estilo9">
    <blockquote>
        <p align="left" class="Estilo13"><a
href="administrador_opcion_crear_administrador.php" class="Estilo9">
            Crear administrador </a></p>
        <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_administradores.php" class="Estilo9">Listar
            administradores</a></p>
        <p align="left" class="Estilo13"><a
href="administrador_opcion_borrar_administrador.php" class="Estilo9">Borrar

```

```

    administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_usuarios.php" class="Estilo9">
    Listar usuarios</a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_puntuaciones.php" class="Estilo9">Listar
    puntuaciones</a> </p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_modificar_usuario.php" class="Estilo9">
    Modificar usuario</a> </p>
    <p align="left" class="Estilo13"> <a
href="administrador_opcion_borrar_usuario.php" class="Estilo9">Borrar
    usuario</a> </p>
    <p class="Estilo13"><a href="administrador_opcion_crear_bases_de_datos.php"
class="Estilo9">
    Crear bases de datos</a> </p>
    <p><span class="Estilo13"><a
href="administrador_opcion_destruir_bases_de_datos.php" class="Estilo9">
    Destruir bases de datos</a></span> </p>
</blockquote>
</div>
<div id="Layer3" style="position:absolute; width:495px; height:322px; z-index:3; left:
239px; top: 145px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;">
    <form name="form4" method="post"
action="administrador_borrar_administrador.php">
    <p align="center" class="Estilo9 Estilo14">&nbsp; </p>
    <p align="center" class="Estilo9 Estilo14">Introduce el nombre del administrador
    que deseas eliminar:</p>
    <p align="center" class="Estilo9"> <span class="Estilo14">Administrador:</span>
    <input name="nick" type="text" id="nick">
    </p>
    <p align="center" class="Estilo9">
    <input name="enviar" type="submit" id="enviar" value="Enviar">
    </p>
</form>
</div>
<div id="Layer4" style="position:absolute; width:731px; height:26px; z-index:1; left:
30px; top: 113px; background-color: #FF3300; layer-background-color: #FF3300;
border: 1px none #000000; background-image: url(tira.jpg); layer-background-image:
url(tira.jpg);">
    <table width="100%" height="32" border="0">
    <tr>
    <td width="65%">&nbsp;</td>
    <td width="35%"><a href="admin.php" class="Estilo9">Cerrar
sesi&oacute;n</a></td>
    </tr>
    </table>
</div>
<div id="Layer1" style="position:absolute; width:740px; height:81px; z-index:5;

```

```
background-image: url(cabecera%20administracion.jpg); layer-background-image:
url(cabecera%20administracion.jpg); border: 1px none #000000; left: 25px; top:
25px;"></div>
</body>
```

### 3.7. administrador\_creacion\_cuenta.php

Este fichero realiza una conexión con la base de datos proporcionándole un nuevo administrador y su contraseña.

```
<?php
include('crear_conexion_bd.php');

$nick = $_POST["nick"];
$password = $_POST["password"];

$sql = "INSERT INTO administrador SET
nick='$nick',
password='$password'";
if (@mysql_query($sql)) {
    $mensaje = "Se ha creado un nuevo administrador";
    include('administrador3.php'); // Cuando creas un nuevo administrador,
te lleva a la página principal con el mensaje pertinente.
} else {
    $mensaje = "El administrador ";
    $mensaje .= $nick;
    $mensaje .= " ya existe, elige otro nombre";
    include('administrador_opcion_crear_administrador.php'); // En caso de
ya existir el administrador que se pretendía crear, se obtiene un error y se da la
posibilidad de crear uno nuevo.
}
?>
```

### 3.8. administrador\_opcion\_listar\_usuarios.php

Este fichero nos muestra por pantalla a todos los usuarios registrados en la plataforma, junto con todos los datos solicitados en la fase de registro.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link rel="stylesheet" href="oceanografico.css" type="text/css" />
<script language="JavaScript" type="text/JavaScript">
<!--
```

```

function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);
//-->
</script>
<style type="text/css">
<!--
.Estilo13 {color: #FFFFFF}
-->
</style>
</head>

<body>
<div id="Layer2" style="position:absolute; width:223px; height:191px; z-index:2; left:
38px; top: 142px; background-color: #FF3300; layer-background-color: #FF3300;
border: 1px none #000000; background-image: url(haciaabajo.jpg); layer-background-
image: url(haciaabajo.jpg);" class="Estilo9">
  <blockquote>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_crear_administrador.php" class="Estilo9"> Crear
administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_usuarios.php" class="Estilo9"> Listar usuarios</a>
</p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_modificar_usuario.php" class="Estilo9"> Modificar
usuario</a> </p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_borrar_usuario.php" class="Estilo9"> Borrar usuario</a>
</p>
    <p class="Estilo13"><a href="administrador_opcion_crear_bases_de_datos.php"
class="Estilo9"> Crear bases de datos</a> </p>
    <p><span class="Estilo13"><a
href="administrador_opcion_destruir_bases_de_datos.php" class="Estilo9"> Destruir
bases de datos</a></span> </p>
  </blockquote>
</div>
<div id="Layer3" style="position:absolute; width:704px; height:215px; z-index:3; left:
239px; top: 145px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;">
<?php

    include('crear_conexion_bd.php');

```

```

$sql = "SELECT count(nick) FROM usuarios";
$salida_sql = @mysql_query($sql);
$salida_sql_vector = mysql_fetch_array($salida_sql);
$numeros_usuarios = $salida_sql_vector[0];

$sql = "SELECT * FROM usuarios";

$salida_sql = @mysql_query($sql); // En la variable $salida_sql almacenamos todos los datos de todos los usuarios de forma vectorial, de modo que utilizaremos mysql_tetch_array para leer estos datos en orden.

echo '<table width="500" border="0">';
echo '<tr>';
echo '<td class="EstiloMensaje">Nick</td><td class="EstiloMensaje">Password</td><td class="EstiloMensaje">Nombre</td><td class="EstiloMensaje">Apellido 1</td><td class="EstiloMensaje">Apellido 2</td><td class="EstiloMensaje">Edad</td><td class="EstiloMensaje">Sexo</td><td class="EstiloMensaje">Ciudad</td><td class="EstiloMensaje">Colegio</td>'; // Mostramos en la primera linea de una tabla que valor va a contener cada columna.
echo '<tr><td class="EstiloMensaje">-----</td><td class="EstiloMensaje">-----</td><td class="EstiloMensaje">-----</td><td class="EstiloMensaje">-----</td><td class="EstiloMensaje">-----</td><td class="EstiloMensaje">-----</td><td class="EstiloMensaje">-----</td><td class="EstiloMensaje">-----</td><td class="EstiloMensaje">-----</td><td class="EstiloMensaje">-----</td></tr>'; // Añadimos un separador en forma de guiones.

while($row = mysql_fetch_array($salida_sql)) { // Recorremos el vector de datos hasta que no queden usuarios
echo '<tr>';
echo '<td class="EstiloMensaje">';
echo $row["nick"]; // Para cada usuario, imprimimos el valor correspondiente a cada columna de la tabla
echo '</td>';
echo '<td class="EstiloMensaje">';
echo $row["password"];
echo '</td>';
echo '<td class="EstiloMensaje">';
echo $row["nombre"];
echo '</td>';
echo '<td class="EstiloMensaje">';
echo $row["apellido1"];
echo '</td>';
echo '<td class="EstiloMensaje">';
echo $row["apellido2"];
echo '</td>';
echo '<td class="EstiloMensaje">';
echo $row["edad"];
echo '</td>';
echo '<td class="EstiloMensaje">';
if ($row["sexo"] == '1') {echo 'Chico;'} else {echo 'Chica;'} // En el caso

```

*del sexo y de la provincia el valor estará almacenado de forma numérica, y tendremos que averiguar cual es el nombre correspondiente a ese número, ya que cada posible nombre tiene asignado un valor exclusivo. Por ejemplo la provincia Albacete tiene asociado el valor 1.*

```
echo '</td>';
echo '<td class="EstiloMensaje">';
if ($row["provincia"] == '0') {echo 'Alava';} else
if ($row["provincia"] == '1') {echo 'Albacete';} else
if ($row["provincia"] == '2') {echo 'Alicante';} else
if ($row["provincia"] == '3') {echo 'Almeria';} else
if ($row["provincia"] == '4') {echo 'Asturias';} else
if ($row["provincia"] == '5') {echo 'Avila';} else
if ($row["provincia"] == '6') {echo 'A Coruña';} else
if ($row["provincia"] == '7') {echo 'Badajoz';} else
if ($row["provincia"] == '8') {echo 'Barcelona';} else
if ($row["provincia"] == '9') {echo 'Burgos';} else
if ($row["provincia"] == '10') {echo 'Caceres';} else
if ($row["provincia"] == '11') {echo 'Cadiz';} else
if ($row["provincia"] == '12') {echo 'Cantabria';} else
if ($row["provincia"] == '13') {echo 'Castellon';} else
if ($row["provincia"] == '14') {echo 'Ceuta';} else
if ($row["provincia"] == '15') {echo 'Ciudad Real';} else
if ($row["provincia"] == '16') {echo 'Cordoba';} else
if ($row["provincia"] == '17') {echo 'Cuenca';} else
if ($row["provincia"] == '18') {echo 'Girona';} else
if ($row["provincia"] == '19') {echo 'Granada';} else
if ($row["provincia"] == '20') {echo 'Guadalajara';} else
if ($row["provincia"] == '21') {echo 'Guipuzkoa';} else
if ($row["provincia"] == '22') {echo 'Huelva';} else
if ($row["provincia"] == '23') {echo 'Huesca';} else
if ($row["provincia"] == '24') {echo 'Jaen';} else
if ($row["provincia"] == '25') {echo 'La Rioja';} else
if ($row["provincia"] == '26') {echo 'Leon';} else
if ($row["provincia"] == '27') {echo 'Lleida';} else
if ($row["provincia"] == '28') {echo 'Lugo';} else
if ($row["provincia"] == '29') {echo 'Madrid';} else
if ($row["provincia"] == '30') {echo 'Malaga';} else
if ($row["provincia"] == '31') {echo 'Melilla';} else
if ($row["provincia"] == '32') {echo 'Murcia';} else
if ($row["provincia"] == '33') {echo 'Navarra';} else
if ($row["provincia"] == '34') {echo 'Ourense';} else
if ($row["provincia"] == '35') {echo 'Palencia';} else
if ($row["provincia"] == '36') {echo 'Pontevedra';} else
if ($row["provincia"] == '37') {echo 'Salamanca';} else
if ($row["provincia"] == '38') {echo 'Segovia';} else
if ($row["provincia"] == '39') {echo 'Sevilla';} else
if ($row["provincia"] == '40') {echo 'Soria';} else
if ($row["provincia"] == '41') {echo 'Tarragona';} else
if ($row["provincia"] == '42') {echo 'Teruel';} else
if ($row["provincia"] == '43') {echo 'Toledo';} else
```

```

        if ($row["provincia"] == '44') {echo 'Valencia';} else
        if ($row["provincia"] == '45') {echo 'Valladolid';} else
        if ($row["provincia"] == '46') {echo 'Vizcaya';} else
        if ($row["provincia"] == '47') {echo 'Zamora';} else
        if ($row["provincia"] == '48') {echo 'Zaragoza';} else
        if ($row["provincia"] == '49') {echo 'Mallorca';} else
        if ($row["provincia"] == '50') {echo 'Ibiza';} else
        if ($row["provincia"] == '51') {echo 'Menorca';} else
        if ($row["provincia"] == '52') {echo 'Formentera';} else
        if ($row["provincia"] == '53') {echo 'Las Palmas';} else
        if ($row["provincia"] == '54') {echo 'Gran Canaria';} else
        if ($row["provincia"] == '55') {echo 'Lanzarote';} else
        if ($row["provincia"] == '57') {echo 'Tenerife';} else
        if ($row["provincia"] == '58') {echo 'La Gomera';} else
        if ($row["provincia"] == '59') {echo 'La Palma';} else
        if ($row["provincia"] == '60') {echo 'El Hierro';}

        echo '</td>';
        echo '<td class="EstiloMensaje">';
        echo $row["colegio"];
        echo '</td>';

        echo '</tr>';
    }

        echo '<tr>';
        echo '</table>';
?>

<p><a TARGET = "_blank"
href="administrador_opcion_listar_usuarios_impresion.php" class="Estilo9">Ver
versi&oacute;n
para imprimir</a></p> // Se permite el acceso a una versi&oacute;n de esta p&agrave;gina para su
impresi&oacute;n.

</div>
<div id="Layer4" style="position:absolute; width:732px; height:26px; z-index:4; left:
29px; top: 119px; background-color: #FF3300; layer-background-color: #FF3300;
border: 1px none #000000; background-image: url(tira.jpg); layer-background-image:
url(tira.jpg);"></div>
<div id="Layer1" style="position:absolute; width:740px; height:81px; z-index:5;
background-image: url(cabecera%20administracion.jpg); layer-background-image:
url(cabecera%20administracion.jpg); border: 1px none #000000; left: 25px; top:
25px;"></div>
</body>
</html>

```

### 3.9. administrador\_opcion\_listar\_usuarios\_impresion.php

Versión para imprimir de la página anterior.

```
<?php

include('crear_conexion_bd.php');

$sql = "SELECT count(nick) FROM usuarios";
$salida_sql = @mysql_query($sql);
$salida_sql_vector = mysql_fetch_array($salida_sql);
$numero_usuarios = $salida_sql_vector[0];

$sql = "SELECT * FROM usuarios";

$salida_sql = @mysql_query($sql);

echo '<table width="600" border="0">';
echo '<tr>';
echo '<td class="EstiloMensaje">Nick</td><td
class="EstiloMensaje">Password</td><td class="EstiloMensaje">Nombre</td><td
class="EstiloMensaje">Apellido1</td><td class="EstiloMensaje">Apellido2</td><td
class="EstiloMensaje">Edad</td><td class="EstiloMensaje">Sexo</td><td
class="EstiloMensaje">Ciudad</td><td
class="EstiloMensaje">Colegio</td><td></td><td></td>';
echo '<tr><td class="EstiloMensaje">-----</td><td class="EstiloMensaje">-----
-----</td><td class="EstiloMensaje">-----</td><td class="EstiloMensaje">-----
</td><td class="EstiloMensaje">-----</td><td class="EstiloMensaje">-----
</td><td class="EstiloMensaje">-----</td><td class="EstiloMensaje">-----</td><td
class="EstiloMensaje">-----</td><td></td><td></td></tr>';

while($row = mysql_fetch_array($salida_sql)) {
echo '<tr>';
echo '<td class="EstiloMensaje2">';
echo $row["nick"];
echo '</td>';
echo '<td class="EstiloMensaje2">';
echo $row["password"];
echo '</td>';
echo '<td class="EstiloMensaje2">';
echo $row["nombre"];
echo '</td>';
echo '<td class="EstiloMensaje2">';
echo $row["apellido1"];
echo '</td>';
echo '<td class="EstiloMensaje2">';
echo $row["apellido2"];
echo '</td>';
echo '<td class="EstiloMensaje2">';
echo $row["edad"];

```



```

echo '</td>';
echo '<td class="EstiloMensaje2">';
    if ($row["sexo"] == '1') {echo 'Chico;'} else {echo 'Chica;'}
echo '</td>';
    echo '<td class="EstiloMensaje2">';
    if ($row["provincia"] == '0') {echo 'Alava;'} else
    if ($row["provincia"] == '1') {echo 'Albacete;'} else
    if ($row["provincia"] == '2') {echo 'Alicante;'} else
    if ($row["provincia"] == '3') {echo 'Almeria;'} else
    if ($row["provincia"] == '4') {echo 'Asturias;'} else
    if ($row["provincia"] == '5') {echo 'Avila;'} else
    if ($row["provincia"] == '6') {echo 'A Coruña;'} else
    if ($row["provincia"] == '7') {echo 'Badajoz;'} else
    if ($row["provincia"] == '8') {echo 'Barcelona;'} else
    if ($row["provincia"] == '9') {echo 'Burgos;'} else
    if ($row["provincia"] == '10') {echo 'Caceres;'} else
    if ($row["provincia"] == '11') {echo 'Cadiz;'} else
    if ($row["provincia"] == '12') {echo 'Cantabria;'} else
    if ($row["provincia"] == '13') {echo 'Castellon;'} else
    if ($row["provincia"] == '14') {echo 'Ceuta;'} else
    if ($row["provincia"] == '15') {echo 'Ciudad Real;'} else
    if ($row["provincia"] == '16') {echo 'Cordoba;'} else
    if ($row["provincia"] == '17') {echo 'Cuenca;'} else
    if ($row["provincia"] == '18') {echo 'Girona;'} else
    if ($row["provincia"] == '19') {echo 'Granada;'} else
    if ($row["provincia"] == '20') {echo 'Guadalajara;'} else
    if ($row["provincia"] == '21') {echo 'Guipuzkoa;'} else
    if ($row["provincia"] == '22') {echo 'Huelva;'} else
    if ($row["provincia"] == '23') {echo 'Huesca;'} else
    if ($row["provincia"] == '24') {echo 'Jaen;'} else
    if ($row["provincia"] == '25') {echo 'La Rioja;'} else
    if ($row["provincia"] == '26') {echo 'Leon;'} else
    if ($row["provincia"] == '27') {echo 'Lleida;'} else
    if ($row["provincia"] == '28') {echo 'Lugo;'} else
    if ($row["provincia"] == '29') {echo 'Madrid;'} else
    if ($row["provincia"] == '30') {echo 'Malaga;'} else
    if ($row["provincia"] == '31') {echo 'Melilla;'} else
    if ($row["provincia"] == '32') {echo 'Murcia;'} else
    if ($row["provincia"] == '33') {echo 'Navarra;'} else
    if ($row["provincia"] == '34') {echo 'Ourense;'} else
    if ($row["provincia"] == '35') {echo 'Palencia;'} else
    if ($row["provincia"] == '36') {echo 'Pontevedra;'} else
    if ($row["provincia"] == '37') {echo 'Salamanca;'} else
    if ($row["provincia"] == '38') {echo 'Segovia;'} else
    if ($row["provincia"] == '39') {echo 'Sevilla;'} else
    if ($row["provincia"] == '40') {echo 'Soria;'} else
    if ($row["provincia"] == '41') {echo 'Tarragona;'} else
    if ($row["provincia"] == '42') {echo 'Teruel;'} else
    if ($row["provincia"] == '43') {echo 'Toledo;'} else
    if ($row["provincia"] == '44') {echo 'Valencia;'} else

```

```

        if ($row["provincia"] == '45') {echo 'Valladolid';} else
        if ($row["provincia"] == '46') {echo 'Vizcaya';} else
        if ($row["provincia"] == '47') {echo 'Zamora';} else
        if ($row["provincia"] == '48') {echo 'Zaragoza';} else
        if ($row["provincia"] == '49') {echo 'Mallorca';} else
        if ($row["provincia"] == '50') {echo 'Ibiza';} else
        if ($row["provincia"] == '51') {echo 'Menorca';} else
        if ($row["provincia"] == '52') {echo 'Formentera';} else
        if ($row["provincia"] == '53') {echo 'Las Palmas';} else
        if ($row["provincia"] == '54') {echo 'Gran Canaria';} else
        if ($row["provincia"] == '55') {echo 'Lanzarote';} else
        if ($row["provincia"] == '57') {echo 'Tenerife';} else
        if ($row["provincia"] == '58') {echo 'La Gomera';} else
        if ($row["provincia"] == '59') {echo 'La Palma';} else
        if ($row["provincia"] == '60') {echo 'El Hierro';}

        echo '</td>';
        echo '<td class="EstiloMensaje2">';
        echo $row["colegio"];
        echo '</td>';
        echo '</tr>';
    }
    echo '<tr>';
    echo '</table>';

?>
<SCRIPT LANGUAGE="JavaScript">

if (window.print) {
document.write('<form>'
+ '<input type=button name=print value="Imprimir" '
+ 'onClick="javascript:window.print(); javascript:window.close()"></form>');
} // Se manda a imprimir la pantalla y posteriormente se cierra.
</script>
</body>

```

### 3.10. administrador\_opcion\_listar\_puntuaciones.php

A través de esta pantalla se nos mostrarán las máximas puntuaciones en cada uno de los juegos, y el número de partidas jugadas en cada uno de estos.

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link rel="stylesheet" href="oceanografico.css" type="text/css" />

```

```

<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);
//-->
</script>
<style type="text/css">
<!--
.Estilo13 { color: #FFFFFF}
-->
</style>
</head>

<body>
<div id="Layer2" style="position:absolute; width:223px; height:191px; z-index:2; left:
38px; top: 142px; background-color: #FF3300; layer-background-color: #FF3300;"
class="Estilo9">
  <blockquote>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_crear_administrador.php" class="Estilo9">
      Crear administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_administradores.php" class="Estilo9">Listar
administradores</a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_borrar_administrador.php" class="Estilo9">Borrar
administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_usuarios.php" class="Estilo9">
      Listar usuarios</a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_puntuaciones.php" class="Estilo9">Listar
puntuaciones</a> </p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_modificar_usuario.php" class="Estilo9">
      Modificar usuario</a> </p>
    <p align="left" class="Estilo13"> <a
href="administrador_opcion_borrar_usuario.php" class="Estilo9">Borrar
usuario</a> </p>
    <p class="Estilo13"><a href="administrador_opcion_crear_bases_de_datos.php"
class="Estilo9">
      Crear bases de datos</a> </p>
    <p><span class="Estilo13"><a

```

```

href="administrador_opcion_destruir_bases_de_datos.php" class="Estilo9">
  Destruir bases de datos</a></span> </p>
</blockquote>
</div>
<div id="Layer3" style="position:absolute; width:185px; height:308px; z-index:3; left:
239px; top: 145px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;">
<?php

    include('crear_conexion_bd.php');

    $sql = "SELECT max(id) FROM juego1";
    $salida_sql = @mysql_query($sql);
    $salida_sql_vector = mysql_fetch_array($salida_sql);
    $numero_partidas = $salida_sql_vector[0];

    $sql = "SELECT * FROM juego1 ORDER BY puntuacion DESC LIMIT 20";
    $salida_sql = @mysql_query($sql);

    echo '<table width="200" border="0"><tr><td class="EstiloMensaje">Scott, el
pájaro que no sabe volar</td></tr><tr><td class="EstiloMensaje">-----
-----</td></tr><tr>';
    $texto = '<td class="EstiloMensaje2">Número de partidas jugadas: ';
    $texto .= $numero_partidas;
    $texto .= '</td>';
    echo $texto;
    echo '</tr></table>';
    echo '<table width="200" border="0"><tr><td
class="EstiloMensaje">Nick</td><td
class="EstiloMensaje">Puntuación</td></tr><tr><td class="EstiloMensaje">-----
</td><td class="EstiloMensaje">-----</td></tr>';
    while($row = mysql_fetch_array($salida_sql)){
    $texto = '<tr><td class="EstiloMensaje2">';
    $texto .= $row["nick"];
    $texto .= '</td><td class="EstiloMensaje2">';
    $texto .= $row["puntuacion"];
    $texto .= '</td></tr>';
    echo $texto;
    }
    echo '</table>';

    echo '<p></p><p></p>';

?>
</div>
<div id="Layer3" style="position:absolute; width:185px; height:308px; z-index:3; left:
239px; top: 145px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;">
<?php

```

```

include('crear_conexion_bd.php');

$sql = "SELECT max(id) FROM juego2";
$salida_sql = @mysql_query($sql);
$salida_sql_vector = mysql_fetch_array($salida_sql);
$numero_partidas = $salida_sql_vector[0];

$sql = "SELECT * FROM juego2 ORDER BY puntuacion DESC LIMIT 20";
$salida_sql = @mysql_query($sql);

echo '<table width="200" border="0"><tr><td class="EstiloMensaje">Igor, el
pingüino defensor</td></tr><tr><td class="EstiloMensaje">-----
-----</td></tr><tr>';
$texto = '<td class="EstiloMensaje2">Número de partidas jugadas: ';
$texto .= $numero_partidas;
$texto .= '</td>';
echo $texto;
echo '</tr></table>';
echo '<table width="200" border="0"><tr><td
class="EstiloMensaje">Nick</td><td
class="EstiloMensaje">Puntuación</td></tr><tr><td class="EstiloMensaje">-----
</td><td class="EstiloMensaje">-----</td></tr>';
while($row = mysql_fetch_array($salida_sql)){
$texto = '<tr><td class="EstiloMensaje2">';
$texto .= $row["nick"];
$texto .= '</td><td class="EstiloMensaje2">';
$texto .= $row["puntuacion"];
$texto .= '</td></tr>';
echo $texto;
}
echo '</table>';

echo '<p></p><p></p>';

?>
</div>
<div id="Layer3" style="position:absolute; width:185px; height:308px; z-index:3; left:
239px; top: 145px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;">
<?php

include('crear_conexion_bd.php');

$sql = "SELECT max(id) FROM juego3";
$salida_sql = @mysql_query($sql);
$salida_sql_vector = mysql_fetch_array($salida_sql);
$numero_partidas = $salida_sql_vector[0];

```

```

$sql = "SELECT * FROM juego3 ORDER BY puntuacion DESC LIMIT 20";
$salida_sql = @mysql_query($sql);

echo '<table width="200" border="0"><tr><td class="EstiloMensaje">Bill, el
cazador de estrellas</td></tr><tr><td class="EstiloMensaje">-----
-----</td></tr><tr>';
$texto = '<td class="EstiloMensaje2">Número de partidas jugadas: ';
$texto .= $numero_partidas;
$texto .= '</td>';
echo $texto;
echo '</tr></table>';
echo '<table width="200" border="0"><tr><td
class="EstiloMensaje">Nick</td><td
class="EstiloMensaje">Puntuación</td></tr><tr><td class="EstiloMensaje">-----
</td><td class="EstiloMensaje">-----</td></tr>';
while($row = mysql_fetch_array($salida_sql)){
$texto = '<tr><td class="EstiloMensaje2">';
$texto .= $row["nick"];
$texto .= '</td><td class="EstiloMensaje2">';
$texto .= $row["puntuacion"];
$texto .= '</td></tr>';
echo $texto;
}
echo '</table>';

echo '<p></p><p></p>';

?>
</div>
<div id="Layer4" style="position:absolute; width:732px; height:26px; z-index:4; left:
29px; top: 119px; background-color: #FF3300; layer-background-color: #FF3300;
border: 1px none #000000; background-image: url(tira.jpg); layer-background-image:
url(tira.jpg);"></div>
<div id="Layer1" style="position:absolute; width:740px; height:81px; z-index:5;
background-image: url(cabecera%20administracion.jpg); layer-background-image:
url(cabecera%20administracion.jpg); border: 1px none #000000; left: 25px; top:
25px;"></div>
</body>
</html>

```

### 3.11. administrador\_opcion\_listar\_puntuaciones\_impresion.php

Versión para imprimir de la página anterior.

```

<body>
<table width="75%" border="0">
<tr>
<td width="6%">
<?php

```

```

include('crear_conexion_bd.php');

$sql = "SELECT max(id) FROM juego1";
$salida_sql = @mysql_query($sql);
$salida_sql_vector = mysql_fetch_array($salida_sql);
$numero_partidas = $salida_sql_vector[0];

$sql = "SELECT * FROM juego1 ORDER BY puntuacion DESC LIMIT 20";
$salida_sql = @mysql_query($sql);

echo '<table width="200" border="0"><tr><td class="EstiloMensaje">Scott, el
pájaro que no sabe volar</td></tr><tr><td class="EstiloMensaje">-----
-----</td></tr><tr>';
$texto = '<td class="EstiloMensaje2">Número de partidas jugadas: ';
$texto .= $numero_partidas;
$texto .= '</td>';
echo $texto;
echo '</tr></table>';
echo '<table width="200" border="0"><tr><td
class="EstiloMensaje">Nick</td><td
class="EstiloMensaje">Puntuación</td></tr><tr><td class="EstiloMensaje">-----
</td><td class="EstiloMensaje">-----</td></tr>';
while($row = mysql_fetch_array($salida_sql)){
$texto = '<tr><td class="EstiloMensaje2">';
$texto .= $row["nick"];
$texto .= '</td><td class="EstiloMensaje2">';
$texto .= $row["puntuacion"];
$texto .= '</td></tr>';
echo $texto;
}
echo '</table>';

echo '<p></p><p></p>';

?>
</td>
<td width="6%">
<?php

include('crear_conexion_bd.php');

$sql = "SELECT max(id) FROM juego2";
$salida_sql = @mysql_query($sql);
$salida_sql_vector = mysql_fetch_array($salida_sql);
$numero_partidas = $salida_sql_vector[0];

$sql = "SELECT * FROM juego2 ORDER BY puntuacion DESC LIMIT 20";
$salida_sql = @mysql_query($sql);

```

```

        echo '<table width="200" border="0"><tr><td class="EstiloMensaje">Igor, el
pingüino defensor</td></tr><tr><td class="EstiloMensaje">-----
</td></tr><tr>';
        $texto = '<td class="EstiloMensaje2">Número de partidas jugadas: ';
        $texto .= $numero_partidas;
        $texto .= '</td>';
        echo $texto;
        echo '</tr></table>';
        echo '<table width="200" border="0"><tr><td
class="EstiloMensaje">Nick</td><td
class="EstiloMensaje">Puntuación</td></tr><tr><td class="EstiloMensaje">-----
</td><td class="EstiloMensaje">-----</td></tr>';
        while($row = mysql_fetch_array($salida_sql)){
        $texto = '<tr><td class="EstiloMensaje2">';
        $texto .= $row["nick"];
        $texto .= '</td><td class="EstiloMensaje2">';
        $texto .= $row["puntuacion"];
        $texto .= '</td></tr>';
        echo $texto;
        }
        echo '</table>';

        echo '<p></p><p></p>';

?>
</td>
<td width="88%">
<?php

        include('crear_conexion_bd.php');

        $sql = "SELECT max(id) FROM juego3";
        $salida_sql = @mysql_query($sql);
        $salida_sql_vector = mysql_fetch_array($salida_sql);
        $numero_partidas = $salida_sql_vector[0];

        $sql = "SELECT * FROM juego3 ORDER BY puntuacion DESC LIMIT 20";
        $salida_sql = @mysql_query($sql);

        echo '<table width="200" border="0"><tr><td class="EstiloMensaje">Bill, el
cazador de estrellas</td></tr><tr><td class="EstiloMensaje">-----
--</td></tr><tr>';
        $texto = '<td class="EstiloMensaje2">Número de partidas jugadas: ';
        $texto .= $numero_partidas;
        $texto .= '</td>';
        echo $texto;
        echo '</tr></table>';
        echo '<table width="200" border="0"><tr><td

```



```

class="EstiloMensaje">Nick</td><td
class="EstiloMensaje">Puntuación</td></tr><tr><td class="EstiloMensaje">-----
</td><td class="EstiloMensaje">-----</td></tr>';
    while($row = mysql_fetch_array($salida_sql)){
        $texto = '<tr><td class="EstiloMensaje2">';
        $texto .= $row["nick"];
        $texto .= '</td><td class="EstiloMensaje2">';
        $texto .= $row["puntuacion"];
        $texto .= '</td></tr>';
        echo $texto;
    }
    echo '</table>';

?>
    </td>
    </tr>
</table>
<SCRIPT LANGUAGE="JavaScript">
if (window.print) {
document.write('<form>'
+ '<input type=button name=print value="Imprimir" '
+ 'onClick="javascript:window.print(); javascript:window.close()"></form>');
}
</script>
</body>

```

### 3.12. administrador\_opcion\_modificar\_usuario.php

El administrador podrá cambiar los datos de cualquier usuario a través de esta opción. En este fichero únicamente realizamos una llamada a administrador\_modificar\_usuario.php indicando el nick del usuario mediante un sencillo formulario.

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link rel="stylesheet" href="oceanografico.css" type="text/css" />
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
    if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
        document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
    else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)

```

```

location.reload();
}
MM_reloadPage(true);
//-->
</script>
<style type="text/css">
<!--
.Estilo13 {color: #FFFFFF}
-->
</style>
</head>

<body>
<div id="Layer2" style="position:absolute; width:223px; height:191px; z-index:2; left:
38px; top: 142px; background-color: #FF3300; layer-background-color: #FF3300;"
class="Estilo9">
  <blockquote>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_crear_administrador.php" class="Estilo9">
      Crear administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_administradores.php" class="Estilo9">Listar
      administradores</a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_borrar_administrador.php" class="Estilo9">Borrar
      administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_usuarios.php" class="Estilo9">
      Listar usuarios</a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_puntuaciones.php" class="Estilo9">Listar
      puntuaciones</a> </p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_modificar_usuario.php" class="Estilo9">
      Modificar usuario</a> </p>
    <p align="left" class="Estilo13"> <a
href="administrador_opcion_borrar_usuario.php" class="Estilo9">Borrar
      usuario</a> </p>
    <p class="Estilo13"><a href="administrador_opcion_crear_bases_de_datos.php"
class="Estilo9">
      Crear bases de datos</a> </p>
    <p><span class="Estilo13"><a
href="administrador_opcion_destruir_bases_de_datos.php" class="Estilo9">
      Destruir bases de datos</a></span> </p>
  </blockquote>
</div>
<div id="Layer3" style="position:absolute; width:496px; height:215px; z-index:3; left:
239px; top: 145px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;">
<form name="form3" method="post" action="administrador_modificar_usuario.php">

```

```

<p align="center" class="Estilo9 Estilo14">&nbsp; </p>
<p align="center" class="Estilo9 Estilo14">Inserta el nombre del usuario del que
deseas modificar sus datos:</p>
<p align="center" class="Estilo9"> <span class="Estilo14">Usuario:</span>
  <input name="nick" type="text" id="nick">
</p>
<p align="center" class="Estilo9">    <input name="enviar" type="submit"
id="enviar" value="Enviar">

</p>
</form>
</div>
<div id="Layer4" style="position:absolute; width:732px; height:26px; z-index:4; left:
29px; top: 119px; background-color: #FF3300; layer-background-color: #FF3300;
border: 1px none #000000; background-image: url(tira.jpg); layer-background-image:
url(tira.jpg);"></div>
<div id="Layer1" style="position:absolute; width:740px; height:81px; z-index:5;
background-image: url(cabecera%20administracion.jpg); layer-background-image:
url(cabecera%20administracion.jpg); border: 1px none #000000; left: 25px; top:
25px;"></div>
</body>
</html>

```

### 3.13. administrador\_modificar\_usuario.php

Llegados a este punto tenemos un nombre de usuario a modificar, y se generará una tabla con todos sus datos para que los modifiquemos libremente, y después mediante una llamada a administrador\_modificar\_usuario\_2.php se guardarán estos resultados en la base de datos.

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link rel="stylesheet" href="oceanografico.css" type="text/css" />
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);
//-->

```

```

</script>
<style type="text/css">
<!--
.Estilo13 {color: #FFFFFF}
-->
</style>
</head>

<body>
<div id="Layer2" style="position:absolute; width:223px; height:191px; z-index:2; left:
38px; top: 142px; background-color: #FF3300; layer-background-color: #FF3300;"
class="Estilo9">
  <blockquote>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_crear_administrador.php" class="Estilo9">
      Crear administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_administradores.php" class="Estilo9">Listar
administradores</a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_borrar_administrador.php" class="Estilo9">Borrar
administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_usuarios.php" class="Estilo9">
      Listar usuarios</a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_puntuaciones.php" class="Estilo9">Listar
puntuaciones</a> </p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_modificar_usuario.php" class="Estilo9">
      Modificar usuario</a> </p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_borrar_usuario.php" class="Estilo9">Borrar
usuario</a> </p>
    <p class="Estilo13"><a href="administrador_opcion_crear_bases_de_datos.php"
class="Estilo9">
      Crear bases de datos</a> </p>
    <p><span class="Estilo13"><a
href="administrador_opcion_destruir_bases_de_datos.php" class="Estilo9">
      Destruir bases de datos</a></span> </p>
  </blockquote>
</div>
<div id="Layer3" style="position:absolute; width:496px; height:215px; z-index:3; left:
239px; top: 145px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;">

<?php
  include('crear_conexion_bd.php');
  $nick = $_POST["nick"];
  $sql = "SELECT * FROM usuarios WHERE nick = '$nick'";

```

```

$salida_sql = @mysql_query($sql);

while($row = mysql_fetch_array($salida_sql)) {
    $password = $row["password"];
    $nombre = $row["nombre"];
    $apellido1 = $row["apellido1"];
    $apellido2 = $row["apellido2"];
    $edad = $row["edad"];
    $sexo = $row["sexo"];
    $colegio = $row["colegio"];
    $localidad = $row["localidad"];
}

```

*// Creamos el formulario y lo imprimimos en pantalla con los resultados recién obtenidos de la consulta a la base de datos.*

```

$texto = '<form name="form1" id="form1" method="post"
action="administrador_modificar_usuario_2.php?usuario_activo=';
$texto .= $nick;
$texto .= "'>';
echo $texto;
    echo ('<table width="200" border="0">');
    echo ('<tr>');
    echo ('<td>');
    echo '<span class="Estilo9">Nombre de usuario</span>';
    echo ('</td>');
    echo ('<td>');
    echo ('<span class="Estilo9">');
    echo $nick;
    $nick2 = $nick;
    echo ('</span>');
    echo ('</td>');
    echo ('<tr>');
    echo ('<td>');
    echo '<br>';
    echo '<span class="Estilo9">Contrase&ntilde;a:</span>';
    echo ('</td>');
    echo ('<td>');
    echo '<input name="password2" type="password" id="password2" value=""';
    echo $password;
    echo '</td>';
    echo ('</td>');
    echo ('<tr>');
    echo ('<td>');
    echo '<br>';
    echo '<span class="Estilo9">Nombre:</span>';
    echo ('</td>');
    echo ('<td>');
    echo '<input name="nombre2" type="text" id="nombre2" value=""';
    echo $nombre;

```

```

echo ";>";
echo ('</td>');
echo ('<tr>');
echo ('<td>');
echo '<br>';
echo '<span class="Estilo9">Primer apellido:</span>';

echo ('</td>');
echo ('<td>');
echo ' <input name="apellido12" type="text" id="apellido12" value="";
echo $apellido1;
echo " />";
echo ('</td>');
echo ('<tr>');
echo ('<td>');
echo '<br>';
echo '<span class="Estilo9">Segundo apellido:</span>';
echo ('</td>');
echo ('<td>');
echo ' <input name="apellido22" type="text" id="apellido22" value="";
echo $apellido2;
echo " />";
echo ('</td>');
echo ('<tr>');
echo ('<td>');
echo '<br>';
echo '<span class="Estilo9">Sexo:</span>';
echo ('</td>');
echo ('<td>');
echo ' <select name="sexo2" id="sexo2" value="";
echo $sexo;
echo ">";
echo '<option value="1">Chico</option>';
echo '<option value="0">Chica </option>';
echo '</select>';
echo ('</td>');
echo ('<tr>');
echo ('<td>');
echo '<br>';
echo '<span class="Estilo9">Edad:</span>';
echo ('</td>');
echo ('<td>');
echo ' <input name="edad2" type="text" id="edad2" value="";
echo $edad;
echo " />";
echo ('</td>');
echo ('<tr>');
echo ('<td>');
echo '<br>';
echo '<span class="Estilo9">Localidad:</span>';

```

```

echo ('</td>');
echo ('<td>');
echo ' <select name="localidad2" id="localidad2" value="";
echo $localidad;
echo ">';
echo '<option value="0">Alava </option>';
    echo '<option value="1">Albacete </option>';
    echo '<option value="2">Alicante </option>';
    echo '<option value="3">Almeria </option>';
    echo '<option value="4">Asturias </option>';
    echo '<option value="5">Avila </option>';
    echo '<option value="6">A Coruña </option>';
echo '<option value="7">Badajoz </option>';
    echo '<option value="8">Barcelona </option>';
    echo '<option value="9">Burgos </option>';
    echo '<option value="10">Caceres </option>';
    echo '<option value="11">Cadiz </option>';
    echo '<option value="12">Cantabria </option>';
    echo '<option value="13">Castellon </option>';
    echo '<option value="14">Ceuta </option>';
    echo '<option value="15">Ciudad Real </option>';
    echo '<option value="16">Cordoba </option>';
    echo '<option value="17">Cuenca </option>';
    echo '<option value="18">Girona </option>';
    echo '<option value="19">Granada </option>';
    echo '<option value="20">Guadalajara </option>';
echo '<option value="21">Guipuzkoa </option>';
    echo '<option value="22">Huelva </option>';
    echo '<option value="23">Huesca </option>';
    echo '<option value="24">Jaen </option>';
    echo '<option value="25">La Rioja </option>';
    echo '<option value="26">Leon </option>';
    echo '<option value="27">Lleida </option>';
echo '<option value="28">Lugo </option>';
    echo '<option value="29">Madrid </option>';
    echo '<option value="30">Malaga </option>';
    echo '<option value="31">Melilla </option>';
    echo '<option value="32">Murcia </option>';
    echo '<option value="33">Navarra </option>';
    echo '<option value="34">Ourenese </option>';
echo '<option value="35">Palencia </option>';
    echo '<option value="36">Pontevedra </option>';
    echo '<option value="37">Salamanca </option>';
    echo '<option value="38">Segovia </option>';
    echo '<option value="39">Sevilla </option>';
    echo '<option value="40">Soria </option>';
    echo '<option value="41">Tarragona </option>';
echo '<option value="42">Teruel </option>';
    echo '<option value="43">Toledo </option>';
    echo '<option value="44">Valencia </option>';

```

```

        echo '<option value="45">Valladolid </option>';
        echo '<option value="46">Vizcaya </option>';
        echo '<option value="47">Zamora </option>';
        echo '<option value="48">Zaragoza </option>';
    echo '<option value="49">Mallorca </option>';
        echo '<option value="50">Ibiza </option>';
        echo '<option value="51">Menorca </option>';
        echo '<option value="52">Formentera </option>';
        echo '<option value="53">Las Palmas </option>';
        echo '<option value="54">Gran Canaria </option>';
        echo '<option value="55">Lanzarote </option>';
    echo '<option value="56">Formentera </option>';
        echo '<option value="57">Tenerife </option>';
        echo '<option value="58">La Gomera </option>';
        echo '<option value="59">La Palma </option>';
        echo '<option value="60">El Hierro </option>';
echo '</select>';
    echo ('</td>');
    echo ('<tr>');
    echo ('<td>');
    echo '<br>';
    echo '<span class="Estilo9">Colegio</span>';
    echo ('</td>');
    echo ('<td>');
    echo '<input name="colegio2" type="text" id="colegio2" value=""';
    echo $colegio;
    echo ";</>';
    echo ('</td>');

    echo ('<tr>');
    echo ('<td>');
    echo '<input name="modificar" type="submit" id="modificar" value="Enviar"
/> ';
    echo ('</td>');
    echo ('<td>');
    echo ('</td>');
    echo ('<tr>');
    echo ('<td>');
    echo ('</table>');
    echo '</form>';
?>

</div>
<div id="Layer4" style="position:absolute; width:732px; height:26px; z-index:4; left:
29px; top: 119px; background-color: #FF3300; layer-background-color: #FF3300;
border: 1px none #000000; background-image: url(tira.jpg); layer-background-image:
url(tira.jpg);"></div>
<div id="Layer1" style="position:absolute; width:740px; height:81px; z-index:5;
background-image: url(cabecera%20administracion.jpg); layer-background-image:
url(cabecera%20administracion.jpg); border: 1px none #000000; left: 25px; top:

```



```
25px;"></div>
</body>
</html>
```

### 3.14. administrador\_modificar\_usuario\_2.php

Los datos modificados deben ser almacenados en la base de datos, y este fichero contiene la orden a la base de datos para realizar dicha operación.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
</head>

<body>

<?php
include ('crear_conexion_bd.php');

$nick = $usuario_activo; // Desde el script anterior nos llega el nick que deseamos
modificar con el nombre de $usuario_activo, para una mayor claridad asignamos a la
variable $nick este valor.

$password = $_POST["password2"];
$nombre = $_POST["nombre2"];
$apellido1 = $_POST["apellido12"];
$apellido2 = $_POST["apellido22"];
$edad = $_POST["edad2"];
$sexo = $_POST["sexo2"];
$colegio = $_POST["colegio2"];
$provincia = $_POST["provincia2"];

    $sql = "UPDATE usuarios SET password = ";
    $sql .= $password;
    $sql .= ", nombre = ";
    $sql .= $nombre;
    $sql .= ", apellido1 = ";
    $sql .= $apellido1;
    $sql .= ", apellido2 = ";
    $sql .= $apellido2;
    $sql .= ", sexo = ";
    $sql .= $sexo;
    $sql .= ", provincia = ";
    $sql .= $provincia;
```

```

    $sql .= ", colegio = ";
    $sql .= $colegio;
    $sql .= ", edad = ";
    $sql .= $edad;
    $sql .= """;
    $sql .= " WHERE nick = ";
    $sql .= $nick;
    $sql .= """;

    if (@mysql_query($sql)) {
    $mensaje = "Los datos del usuario ";
        $mensaje .= $nick;
        $mensaje .= " han sido modificados correctamente";
        include('administrador3.php');
    } else {
    echo('<p>Error al modificar los datos: ' .
        mysql_error() . '</p>');
    }
?>

</body>
</html>

```

### 3.15. administrador\_opcion\_borrar\_usuario.php

A la hora de borrar un usuario debemos conocer el nick de este (podemos consultarlo con la opción listar usuarios) y entonces introduciendolo en el formulario que genera este script será eliminado en una llamada a administrador\_borrar\_usuario.php.

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link rel="stylesheet" href="oceanografico.css" type="text/css" />
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);
//-->

```

```

</script>
<style type="text/css">
<!--
.Estilo13 {color: #FFFFFF}
-->
</style>
</head>

<body>
<div id="Layer2" style="position:absolute; width:223px; height:191px; z-index:2; left:
38px; top: 142px; background-color: #FF3300; layer-background-color: #FF3300;"
class="Estilo9">
  <blockquote>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_crear_administrador.php" class="Estilo9">
      Crear administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_administradores.php" class="Estilo9">Listar
administradores</a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_borrar_administrador.php" class="Estilo9">Borrar
administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_usuarios.php" class="Estilo9">
      Listar usuarios</a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_puntuaciones.php" class="Estilo9">Listar
puntuaciones</a> </p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_modificar_usuario.php" class="Estilo9">
      Modificar usuario</a> </p>
    <p align="left" class="Estilo13"> <a
href="administrador_opcion_borrar_usuario.php" class="Estilo9">Borrar
usuario</a> </p>
    <p class="Estilo13"><a href="administrador_opcion_crear_bases_de_datos.php"
class="Estilo9">
      Crear bases de datos</a> </p>
    <p><span class="Estilo13"><a
href="administrador_opcion_destruir_bases_de_datos.php" class="Estilo9">
      Destruir bases de datos</a></span> </p>
  </blockquote>
</div>
<div id="Layer3" style="position:absolute; width:495px; height:215px; z-index:3; left:
240px; top: 145px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;">
<form name="form4" method="post" action="administrador_borrar_usuario.php">
  <p align="center" class="Estilo9 Estilo14">&nbsp; </p>
  <p align="center" class="Estilo9 Estilo14">Introduce el nombre del usuario que
deseas eliminar:</p>
  <p align="center" class="Estilo9"> <span class="Estilo14">Usuario:</span>

```

```

<input name="nick" type="text" id="nick">
</p>
<p align="center" class="Estilo9">
  <input name="enviar" type="submit" id="enviar" value="Enviar">
</p>
</form>
</div>
<div id="Layer4" style="position:absolute; width:732px; height:26px; z-index:4; left:
29px; top: 119px; background-color: #FF3300; layer-background-color: #FF3300;
border: 1px none #000000; background-image: url(tira.jpg); layer-background-image:
url(tira.jpg);"></div>
<div id="Layer1" style="position:absolute; width:740px; height:81px; z-index:5;
background-image: url(cabecera%20administracion.jpg); layer-background-image:
url(cabecera%20administracion.jpg); border: 1px none #000000; left: 25px; top:
25px;"></div>
</body>
</html>

```

### 3.16. administrador\_borrar\_usuario.php

Este script realiza una sencilla llamada a mysql indicandole que borre el nick recibido por el script anterior. Una vez realizado nos carga la página principal con el mensaje correspondiente.

```

<?php
  include('crear_conexion_bd.php');
  $nick = $_POST["nick"];

  $sql = "DELETE FROM usuarios WHERE nick='$nick'";
  if (@mysql_query($sql)) {
    $mensaje = "El usuario ";
      $mensaje .= $nick;
      $mensaje .= "ha sido eliminado de la base de datos";
      include('administrador3.php');
  } else {
    $mensaje = "No se ha podido eliminar al usuario ";
      $mensaje .= $nick;
      $mensaje .= " de la base de datos";
      include('administrador3.php');
  }
?>

```

### 3.17. administrador\_opcion\_crear\_bases\_de\_datos.php

En caso de que el sistema se acabe de instalar o se haya eliminado deberá recurrirse a esta opción que se encargará de crear todas las bases de datos que nos hagan falta.

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link rel="stylesheet" href="oceanografico.css" type="text/css" />
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);
//-->
</script>
<style type="text/css">
<!--
.Estilo13 {color: #FFFFFF}
-->
</style>
</head>

<body>
<div id="Layer2" style="position:absolute; width:223px; height:191px; z-index:2; left:
38px; top: 142px; background-color: #FF3300; layer-background-color: #FF3300;"
class="Estilo9">
  <blockquote>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_crear_administrador.php" class="Estilo9">
      Crear administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_administradores.php" class="Estilo9">Listar
      administradores</a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_borrar_administrador.php" class="Estilo9">Borrar
      administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_usuarios.php" class="Estilo9">
      Listar usuarios</a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_puntuaciones.php" class="Estilo9">Listar
      puntuaciones</a> </p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_modificar_usuario.php" class="Estilo9">

```

```

    Modificar usuario</a> </p>
    <p align="left" class="Estilo13"> <a
href="administrador_opcion_borrar_usuario.php" class="Estilo9">Borrar
    usuario</a> </p>
    <p class="Estilo13"><a href="administrador_opcion_crear_bases_de_datos.php"
class="Estilo9">
    Crear bases de datos</a> </p>
    <p><span class="Estilo13"><a
href="administrador_opcion_destruir_bases_de_datos.php" class="Estilo9">
    Destruir bases de datos</a></span> </p>
</blockquote>
</div>
<div id="Layer3" style="position:absolute; width:496px; height:215px; z-index:3; left:
239px; top: 145px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;">
    <p align="center" class="Estilo9 Estilo14">&nbsp;</p>
    <p align="center" class="Estilo9 Estilo14">Desde aqu&iacute; podr&aacute;s
regenerar las bases de datos, para que esto suceda no deben existir en el sistema, utiliza
esta opci&oacute;n solo si acabas de instalar el sistema.</p>
    <p align="center" class="Estilo9 Estilo14"><a
href="creacion_bases_de_datos.php">Crear bases de datos</a> </p>
</div>
<div id="Layer4" style="position:absolute; width:732px; height:26px; z-index:4; left:
29px; top: 119px; background-color: #FF3300; layer-background-color: #FF3300;
border: 1px none #000000; background-image: url(tira.jpg); layer-background-image:
url(tira.jpg);"></div>
<div id="Layer1" style="position:absolute; width:740px; height:81px; z-index:5;
background-image: url(cabecera%20administracion.jpg); layer-background-image:
url(cabecera%20administracion.jpg); border: 1px none #000000; left: 25px; top:
25px;"></div>
</body>
</html>

```

### 3.18. creacion\_base\_de\_datos.php

Despues de decidir las tablas que se requerirían para almacenar los datos de los usuarios y de los juegos, para crearlas se optó por la creación de un script php que las creará una tras otra sin necesidad de la interacción de ningún usuario, de forma que para cada ordenador en el que se instale el sistema no haya posibilidad de error humano al crear estas tablas.

```

<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title><?php
$conexion = mysql_connect('localhost', 'root', '');
if (!$conexion) {
    echo( '<p>Imposible conectar a la base de datos</p>' );
    exit();
}

```

```

$sql = 'create database oceanografico';
@mysql_query($sql); // Debemos crear una base de datos llamada oceanográfico,
donde almacenaremos las tablas que la compondrán, y esta es la instrucción
encargada de ello.
@mysql_select_db('oceanografico'); // Seleccionamos la base de dato recién creada
para proceder a añadirle las tablas que la componen.

$mensaje = "Tablas creadas: ";

$sql = 'CREATE TABLE usuarios (
    nick varchar(30) NOT NULL PRIMARY KEY,
    nombre varchar(30) NOT NULL,
    apellido1 varchar(30) NOT NULL,
    apellido2 varchar(30) NOT NULL,
    sexo INT(1) NOT NULL,
    edad INT(2) NOT NULL,
    provincia varchar(30) NOT NULL,
    colegio varchar(30),
    password varchar(30) NOT NULL
)'; // Creamos la tabla usuarios con los campos necesarios.
if ( @mysql_query($sql) ) {
    $mensaje .= "Usuarios ";
} else {
    echo('<p>Error creando la tabla de usuarios: ' .
        mysql_error() . '</p>'); // En caso de algún error en la creación de la tabla se nos
indicará por pantalla.
}

$sql = 'CREATE TABLE administrador (
    nick varchar(30) NOT NULL PRIMARY KEY,
    password varchar(30) NOT NULL
)'; // Creamos la tabla administrador con los campos necesarios.

if ( @mysql_query($sql) ) {
    $mensaje .= "Administradores ";
} else {
    echo('<p>Error creando la tabla de administrador: ' .
        mysql_error() . '</p>');
}

$sql = "INSERT INTO administrador SET
    nick='root',
    password='root'";
if ( @mysql_query($sql) ) { // Al crear la tabla administrador crearemos
automáticamente un usuario con nick root y contraseña root, el cual será el primer
administrador de nuestro sistema.
    $mensaje .= "(root/root) ";
} else {
    echo('<p>Error creando al adminbistrador: ' .
        mysql_error() . '</p>');
}

```

```

}

$sql = 'CREATE TABLE juego1 (
    id int(4) NOT NULL AUTO_INCREMENT PRIMARY KEY,
    nick varchar(30) NOT NULL,
    puntuacion int(6) NOT NULL
); // Creamos la tabla del primer juego con los campos necesarios.

if ( @mysql_query($sql) ) {
    $mensaje .= "Juego1 ";
} else {
    echo('<p>Error creando la tabla de puntuaciones del juego 1: ' .
        mysql_error() . '</p>');
}

$sql = 'CREATE TABLE juego2 (
    id int(4) NOT NULL AUTO_INCREMENT PRIMARY KEY,
    nick varchar(30) NOT NULL,
    puntuacion int(6) NOT NULL
); // Creamos la tabla del segundo juego con los campos necesarios.

if ( @mysql_query($sql) ) {
    $mensaje .= "Juego2 ";
} else {
    echo('<p>Error creando la tabla de puntuaciones del juego 2: ' .
        mysql_error() . '</p>');
}

$sql = 'CREATE TABLE juego3 (
    id int(4) NOT NULL AUTO_INCREMENT PRIMARY KEY,
    nick varchar(30) NOT NULL,
    puntuacion int(6) NOT NULL
); // Creamos la tabla del tercer juego con los campos necesarios.

if ( @mysql_query($sql) ) {
    $mensaje .= "Juego3 ";
} else {
    echo('<p>Error creando la tabla de puntuaciones del juego 3: ' .
        mysql_error() . '</p>');
}

include ('administrador3.php');

?>

```

### 3.19. administrador\_opcion\_destruir\_bases\_de\_datos.php

En caso de que el sistema haya sufrido un contratiempo (corrupción de la base



de datos, entrada de usuarios no deseados o simplemente si se desea purgar los datos, será conveniente destruir las tablas para volver a generarlas posteriormente. A través de la opción Destruir bases de datos podremos realizar esta tarea.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link rel="stylesheet" href="oceanografico.css" type="text/css" />
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);
//-->
</script>
<style type="text/css">
<!--
.Estilo13 {color: #FFFFFF}
-->
</style>
</head>

<body>
<div id="Layer2" style="position:absolute; width:223px; height:191px; z-index:2; left:
38px; top: 142px; background-color: #FF3300; layer-background-color: #FF3300;"
class="Estilo9">
  <blockquote>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_crear_administrador.php" class="Estilo9">
      Crear administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_administradores.php" class="Estilo9">Listar
      administradores</a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_borrar_administrador.php" class="Estilo9">Borrar
      administrador </a></p>
    <p align="left" class="Estilo13"><a
href="administrador_opcion_listar_usuarios.php" class="Estilo9">
      Listar usuarios</a></p>
    <p align="left" class="Estilo13"><a
```

```

href="administrador_opcion_listar_puntuaciones.php" class="Estilo9">Listar
  puntuaciones</a> </p>
  <p align="left" class="Estilo13"><a
href="administrador_opcion_modificar_usuario.php" class="Estilo9">
  Modificar usuario</a> </p>
  <p align="left" class="Estilo13"> <a
href="administrador_opcion_borrar_usuario.php" class="Estilo9">Borrar
  usuario</a> </p>
  <p class="Estilo13"><a href="administrador_opcion_crear_bases_de_datos.php"
class="Estilo9">
  Crear bases de datos</a> </p>
  <p><span class="Estilo13"><a
href="administrador_opcion_destruir_bases_de_datos.php" class="Estilo9">
  Destruir bases de datos</a></span> </p>
</blockquote>
</div>
<div id="Layer3" style="position:absolute; width:496px; height:215px; z-index:3; left:
239px; top: 145px; background-color: #CCCCCC; layer-background-color: #CCCCCC;
border: 1px none #000000;" class="EstiloMensaje">
  <div align="center">
  <p>&nbsp;</p>
  <p>Solo debes destruir las bases de datos en el caso de que desees reiniciar el
sistema, todos los datos almacenados se perder&acute;n y habr&acute; que crear de
nuevo las bases de datos.</p>
  <p><a href="destruir_base_de_datos.php">Destruir bases de datos</a> </p>
  </div>
</div>
<div id="Layer4" style="position:absolute; width:732px; height:26px; z-index:4; left:
29px; top: 119px; background-color: #FF3300; layer-background-color: #FF3300;
border: 1px none #000000; background-image: url(tira.jpg); layer-background-image:
url(tira.jpg);"></div>
<div id="Layer1" style="position:absolute; width:740px; height:81px; z-index:5;
background-image: url(cabecera%20administracion.jpg); layer-background-image:
url(cabecera%20administracion.jpg); border: 1px none #000000; left: 25px; top:
25px;"></div>
</body>
</html>

```

### 3.20. destruir\_base\_de\_datos.php

El script destruir\_base\_de\_datos.php se encargará de eliminar las bases de datos, dejando el sistema inutil con la necesidad de una regeneración de estas bases de datos mediante la opción crear bases de datos o siguiendo el último punto de la instalación del sistema en un servidor.

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>

```

```
<title>Zona del administrador de &iexcl;Un chapuz&oacute;n de
diversi&oacute;n!</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body>
<?php
    include('crear_conexion_bd.php');
    $sql = ('DROP DATABASE oceanografico');
    mysql_query($sql);
    $mensaje = "Las bases de datos han sido destruidas, a continuación debes ir a la
opción Crear bases de datos y regenerarlas, o el sistema permanecerá inservible";
    include('administrador3.php');
?>
</body>
</html>
```

## 4. Implementación de los mapas

### 4.1. mapa\_sin\_usuarios.swf

Mapa\_sin\_usuarios.swf nos ofrece la posibilidad de ver en que consiste cada uno de los juegos de la plataforma, ver sus máximas puntuaciones y registrarnos en la plataforma. Está estructurado de forma que el frame 1 corresponde a la carga inicial y los frames posteriores a los diferentes juegos de los que componen nuestra plataforma, siendo el número de juego correspondiente a cada frame el número de frame en el que está situado menos dos. Así, juego 1 irá al frame 3, juego 2 al frame 4 y juego 3 al frame 5. El frame 2 estará reservado a los botones que carezcan de juego, a pesar de que en un futuro es posible que se cree un juego para ellos.

Encontramos un stop(); como código único de \_root en cada frame, sin embargo cada botón tiene un código asociado de gran sencillez que se utilizará para navegar dentro del mapa. Este código es el siguiente:

```
on (release) {  
gotoAndStop(2);  
}
```

Dentro de cada frame de los juegos ya existentes existen dos nuevos elementos, que son la mascota animada que corresponde al botón sobre el que el usuario ha pinchado y un clip animado de gran atractivo visual con 4 capas.

La capa inferior muestra un bocadillo al estilo de los comics que se hace grande hasta alcanzar un tamaño adecuado y en el cual se va a mostrar toda la información y nuevos botones. Las otras 3 capas pertenecen cada una a un botón, el primero de ellos es CERRAR, y consiste en una X en la esquina superior derecha del bocadillo cuyo único código es:

```
_root.gotoAndPlay(1);
```

y simula efectivamente al clásico botón cerrar de las aplicaciones windows.

El botón REGISTRATE cumple el propósito de permitir al usuario darse de alta en la plataforma llevandolo a la página de registro. Al igual que sucedía con el botón anterior, su código es de gran sencillez y facil entendimiento.

```
on (release) {  
url = "registro.php";  
getURL(url);  
}
```

## 4.2. mapa\_usuario\_activo.swf

La estructura de este fichero es exactamente igual a la del fichero anterior, con una única diferencia, el botón REGISTRATE es sustituido por el botón JUGAR, y su código es ligeramente más complejo, pero igualmente de fácil comprensión.

```
on (release) {  
url = "./juegos/juego1.php?usuario_activo="+_root.nick;  
getURL(url, "_blank");  
}
```

En este caso concatenamos la url que solicitamos tiene un argumento llamado usuario\_activo, que será utilizado para enviarle al juego que estamos cargando el nick del usuario activo en el sistema, para que se almacene su resultado en la base de datos, y la instrucción getURL incluye el argumento opcional “\_BLANK” gracias al cual la página es cargada en un nuevo explorador, para así no cerrar la ventana donde el usuario se encuentra.

Vemos que estamos llamando al fichero juego1.php. Cada uno de los juegos tiene su propio fichero de juego, consistente en una variante modificada del fichero autogenerado por el flash al exportar el proyecto.

## 4.3. juego1.php

Las modificaciones que se han realizado sobre el fichero exportado por el flash (además de cambiarle la extensión para mantener coherencia) son las siguientes:

```
<?php $temp = '<param name="movie" value="juego1.swf?nick=';  
        $temp .= $usuario_activo;  
        $temp .= "'>';  
        echo $temp;  
?>
```

```
<?php $temp = '<embed src="juego1.swf?nick=';  
        $temp .= $usuario_activo;  
        $temp .= ' quality="high"  
pluginpage="http://www.macromedia.com/go/getflashplayer" type="application/x-  
shockwave-flash" width="322" height="269"></embed>';  
        echo $temp;  
?>
```

Estos dos fragmentos de código sustituyen a los inclusores de la película flash en el fichero por unos nuevos inclusores donde además de pasar el nombre de la película, añadimos la variable \$usuario\_activo, que todos los ficheros de juego aceptarán y utilizarán para almacenar la puntuación obtenida en la base de datos.

#### 4.4. juego2.php

Realiza exactamente la misma función que el fichero juego1.php, pero actuando sobre el juego2 en lugar del juego1. El código será el siguiente.

```
<?php $temp = '<param name="movie" value="juego2.swf?nick=';
        $temp .= $usuario_activo;
        $temp .= "'>';
        echo $temp;
?>
```

```
<?php $temp = '<embed src="juego2.swf?nick=';
        $temp .= $usuario_activo;
        $temp .= ' quality="high"
pluginspage="http://www.macromedia.com/go/getflashplayer" type="application/x-
shockwave-flash" width="322" height="269"></embed>';
        echo $temp;
?>
```

#### 4.5. juego3.php

Realiza exactamente la misma función que los ficheros juego1.php y juego2.php, pero actuando sobre el juego3 en lugar del juego1 o juego2. El código será el siguiente.

```
<?php $temp = '<param name="movie" value="juego3.swf?nick=';
        $temp .= $usuario_activo;
        $temp .= "'>';
        echo $temp;
?>
```

```
<?php $temp = '<embed src="juego3.swf?nick=';
        $temp .= $usuario_activo;
        $temp .= ' quality="high"
pluginspage="http://www.macromedia.com/go/getflashplayer" type="application/x-
shockwave-flash" width="322" height="269"></embed>';
        echo $temp;
?>
```

#### 4.6. ranking1.php

Fichero encargado de mostrar en una ventana nueva las diez mejores puntuaciones del juego 1, con un pequeño código en Javascript para salir de ella.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>Un chapuz&oacute;n de diversi&oacute;n</title>
```

```

<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);
//-->
</script>
<link rel="stylesheet" href="oceanografico.css" type="text/css" />
</head>

<body bgcolor="#0099CC">
<div id="Layer3" style="position:absolute; width:864px; height:76px; z-index:3; left:
9px; top: 23px; font-size: 9px;">
  <p> </p>
</div>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>&nbsp;</p>
<div id="Capapuntuaciones" style="position:absolute; width:561px; height:295px; z-
index:2; left: 165px; top: 251px; font-size: 9px;">
  <?php

      include('crear_conexion_bd.php');
      $sql = "SELECT * FROM juego1 ORDER BY puntuacion DESC LIMIT 10";
      $salida_sql = @mysql_query($sql); // Obtenemos las diez mejores
puntuaciones del juego correspondiente.

      echo '<table width="200" border="0" align="center"><tr><td
class="Estilo9">M&aacute;ximas puntuaciones</td><td class="Estilo9"></td></tr>';
      while($row = mysql_fetch_array($salida_sql)){
        $texto = '<tr><td class="Estilo9">';
        $texto .= $row["nick"];
        $texto .= '</td><td class="Estilo9">';
        $texto .= $row["puntuacion"];
        $texto .= '</td></tr>';
        echo $texto;
      }
      echo '</table>'; // Se creará una tabla donde se mostrarán los usuarios que han
realizado las mejores puntuaciones con estas puntuaciones.

  ?>

  <p> <font color="#FFFFFF"> </font></p>
</div>

```

```

<div id="Layer1" style="position:absolute; width:849px; height:45px; z-index:5; left:
30px; top: 161px; background-color: #0033CC; layer-background-color: #0033CC;
border: 1px none #000000; background-image: url(imagenes/water-texture-blue.jpg);
layer-background-image: url(imagenes/water-texture-blue.jpg);">
  <span style="font-size: 9px"></span></div>
<div id="Layer2" style="position:absolute; width:834px; height:31px; z-index:4; left:
36px; top: 133px; background-color: #0000CC; layer-background-color: #0000CC;
border: 1px none #000000;">
  <table width="830" border="0">
    <tr>
      <td width="577" class="EstiloFormulario">Bienvenido a las m&aacute;ximas
puntuaciones de Scott, el p&aacute;jaro que no sabe volar </td>
      <td width="145"><span class="Estilo9">
        </span></td>
      <td width="94"><span class="Estilo9">
        <?php
        $texto3 = '<a href="javascript:close()" class="EstiloFormulario">Cerrar
ventana</a></p>'; // Realizamos una llamada a la funci3n close de javascript, que nos
permite cerrar la ventana activa, y la situamos en un enlace llamado cerrar ventana.
        echo $texto3;
        ?>
        </span></td>
      </tr>
    </table>

</div>
</body>
</html>

```

#### 4.7. ranking2.php

Fichero encargado de mostrar en una ventana nueva las diez mejores puntuaciones del juego 2, con un pequeño código en Javascript para salir de ella.

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>Un chapuz&oacute;n de diversi&oacute;n</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; } }
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}

```



```

MM_reloadPage(true);
//-->
</script>
<link rel="stylesheet" href="oceanografico.css" type="text/css" />
</head>

<body bgcolor="#0099CC">
<div id="Layer3" style="position:absolute; width:864px; height:76px; z-index:3; left:
9px; top: 23px; font-size: 9px;">
  <p> </p>
</div>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>&nbsp;</p>
<div id="Capapuntuaciones" style="position:absolute; width:561px; height:295px; z-
index:2; left: 165px; top: 251px; font-size: 9px;">
  <?php

      include('crear_conexion_bd.php');
      $sql = "SELECT * FROM juego2 ORDER BY puntuacion DESC LIMIT 10";
      $salida_sql = @mysql_query($sql);

      echo '<table width="200" border="0" align="center"><tr><td
class="Estilo9">M&acute;ximas puntuaciones</td><td class="Estilo9"></td></tr>';
      while($row = mysql_fetch_array($salida_sql)){
        $texto = '<tr><td class="Estilo9">';
        $texto .= $row["nick"];
        $texto .= '</td><td class="Estilo9">';
        $texto .= $row["puntuacion"];
        $texto .= '</td></tr>';
        echo $texto;
      }
      echo '</table>';
  ?>

  <p> <font color="#FFFFFF"> </font></p>
</div>
<div id="Layer1" style="position:absolute; width:849px; height:45px; z-index:5; left:
30px; top: 161px; background-color: #0033CC; layer-background-color: #0033CC;
border: 1px none #000000; background-image: url(imagenes/water-texture-blue.jpg);
layer-background-image: url(imagenes/water-texture-blue.jpg);">
  <span style="font-size: 9px"></span></div>
<div id="Layer2" style="position:absolute; width:834px; height:31px; z-index:4; left:
36px; top: 133px; background-color: #0000CC; layer-background-color: #0000CC;
border: 1px none #000000;">
  <table width="830" border="0">
    <tr>
      <td width="577" class="EstiloFormulario">Bienvenido a las m&acute;ximas
puntuaciones de Igor, el ping&uuml;ino defensor </td>
      <td width="145"><span class="Estilo9">

```

```

        </span></td>
        <td width="94"><span class="Estilo9">
        <?php
        $texto3 = '<a href="javascript:close()" class="EstiloFormulario">Cerrar
ventana</a></p>';
        echo $texto3;
        ?>
        </span></td>
        </tr>
        </table>

</div>
</body>
</html>

```

#### 4.8. ranking3.php

Fichero encargado de mostrar en una ventana nueva las diez mejores puntuaciones del juego 3, con un pequeño código en Javascript para salir de ella.

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>Un chapuz&oacute;n de diversi&oacute;n</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) { //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
MM_reloadPage(true);
//-->
</script>
<link rel="stylesheet" href="oceanografico.css" type="text/css" />
</head>
<body bgcolor="#0099CC">
<div id="Layer3" style="position:absolute; width:864px; height:76px; z-index:3; left:
9px; top: 23px; font-size: 9px;">
  <p> </p>
</div>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>&nbsp;</p>

```

```

<div id="Capapuntuaciones" style="position:absolute; width:561px; height:295px; z-
index:2; left: 165px; top: 251px; font-size: 9px;">
  <?php

    include('crear_conexion_bd.php');
    $sql = "SELECT * FROM juego3 ORDER BY puntuacion DESC LIMIT 10";
    $salida_sql = @mysql_query($sql);

    echo '<table width="200" border="0" align="center"><tr><td
class="Estilo9">M&acuteximas puntuaciones</td><td class="Estilo9"></td></tr>';
    while($row = mysql_fetch_array($salida_sql)){
      $texto = '<tr><td class="Estilo9">';
      $texto .= $row["nick"];
      $texto .= '</td><td class="Estilo9">';
      $texto .= $row["puntuacion"];
      $texto .= '</td></tr>';
      echo $texto;
    }
    echo '</table>';
  ?>
  <p> <font color="#FFFFFF"> </font></p>
</div>
<div id="Layer1" style="position:absolute; width:849px; height:45px; z-index:5; left:
30px; top: 161px; background-color: #0033CC; layer-background-color: #0033CC;
border: 1px none #000000; background-image: url(imagenes/water-texture-blue.jpg);
layer-background-image: url(imagenes/water-texture-blue.jpg);">
  <span style="font-size: 9px"></span></div>
<div id="Layer2" style="position:absolute; width:834px; height:31px; z-index:4; left:
36px; top: 133px; background-color: #0000CC; layer-background-color: #0000CC;
border: 1px none #000000;">
  <table width="830" border="0">
    <tr>
      <td width="577" class="EstiloFormulario">Bienvenido a las m&acuteximas
puntuaciones de Bill, el cazador de estrellas</td>
      <td width="145"><span class="Estilo9">
        </span></td>
      <td width="94"><span class="Estilo9">
        <?php
          $texto3 = '<a href="javascript:close()" class="EstiloFormulario">Cerrar
ventana</a></p>';
          echo $texto3;
        ?>
        </span></td>
    </tr>
  </table>

</div>
</body>
</html>

```

## IV. DESARROLLO E IMPLEMENTACIÓN DE LOS JUEGOS

### 1. Scott, el pájaro que no sabe volar

#### 1.1. Fase de estudio

Debemos realizar un juego para niños sobre una temática concreta, como es el oceanográfico en este caso, y este juego formará parte de una plataforma de usuarios donde podremos ver las mejores puntuaciones de este juego. Observando la mayor parte de juegos para niños existentes en el mercado podemos apreciar que hacen hincapie en la enseñanza de conocimientos, pero este tipo de juegos suelen acabar resultando cansinos para sus usuarios, por eso preferimos ofrecer un enfoque puramente lúdico, donde prime la habilidad a los conocimientos, ya que los conocimientos los adquirirán en su visita al oceanográfico y en un parque de estas características puede resultar cansino el ya conocer cosas que vas a aprender en el, causando el aburrimiento en el visitante.

Dentro de los diferentes géneros de juegos existentes nos centraremos en los juegos puramente 2D, al requerir menos recursos del computador y menor tiempo de carga. Los tipos de juegos clásicos son los de plataformas (Super Mario Bros, Sonic), los de ladrillos (Arkanoid), los de naves (Raptor) y los de puzzle (Tetris, Mahong). En los juegos de plataformas nuestro objetivo será llevar a un personaje de un lado a otro de la pantalla sin que ninguno de los múltiples obstáculos y enemigos que encontremos acaben con nosotros, y cada cierto número de niveles deberemos acabar con la vida de un enemigo de mayor dureza que los normales. Estos juegos se componen de enormes mapeados basados en tiles y en ellos es muy importante el diseño de los niveles, ya que lo es todo, un juego con malos niveles no resultará atractivo. El concepto de estos juegos suele ser muy retorcido para que llame la atención, como por ejemplo un fontanero que salva a una princesa en un mundo lleno de tortugas malignas y donde las setas te ayudan a conseguir poderes especiales. El mayor defecto de estos juegos es que su diseño no siempre resulta del agrado de todos los jugadores, con lo que su éxito no siempre está asegurado y en todos los casos de éxito masivo de un juego de estas características siempre ha habido una gran campaña de marketing detrás de ellos. Los juegos de ladrillos son conceptualmente mucho más sencillo y el diseño de sus niveles es igualmente más sencillo. También están basados en tiles, pero normalmente todo lo que hay en el nivel lo ves en todo momento en pantalla, con lo que el conocimiento de lo que te vas a encontrar es mayor (al fin y al cabo lo ves en todo momento). En estos juegos la interacción con el nivel se realiza de forma indirecta, ya que la raqueta que el usuario maneja se limita a darle un comportamiento al elemento (pelota) que interactuará directamente sobre los objetos del nivel (ladrillos). La raqueta también tendrá la función de recoger objetos que generen los ladrillos y que le ayudarán a conseguir nuevas habilidades (mayor tamaño o mayor número de pelotas en pantalla). Estos juegos presentan un nivel de adicción muy elevado pero no suelen presentar nuevos desafíos con el tiempo, lo que hace que los usuarios se cansen de el. Los juegos de naves son muy similares a los de plataformas, se podrían considerar una simplificación de estos, donde suprimimos todo lo que hace que el juego sea lento (las plataformas y el tener que llegar de un lado a otro de la pantalla) y nos centramos en lo que más gusta a la gente, la destrucción de sus enemigos. Estos juegos gozan de una

gran simpleza en sus niveles (Llegando en algunos casos a ser generados de forma aleatoria, como en el Penetrator) y pecan de que la mayoría de sus enemigos se comportan de forma previsible, lo que por otra parte reduce la dificultad que de por sí suelen tener y los hace más asequibles en sus primeras pantallas. Estos juegos suelen ser excesivamente complicados de finalizar, ya que exigen un elevado nivel de atención y de conocimiento de cómo funciona la nave y como se van a comportar todos los enemigos, y incluso de que va a aparecer futuramente en el nivel. Finalmente los juegos de puzzle, los más sencillos en cuanto a concepto (prácticamente inexistente en la mayor parte de ellos, aunque en muchos casos intenten introducir una historia que entra con calzador, ya que no solo es innecesaria, si no que distrae de lo importante) y también los que mayor grado de adicción generan, suponiendo que el usuario se sienta atraído por el juego (La mayor parte de los niños prefieren matar a inocentes seres de otro planeta que ponerse a pensar como encajar dos piezas). En algunos casos ha habido variantes que combinaban este género con el tiempo real, como el caso del conocidísimo Tetris, ejemplo de juego sin ningún tipo de concepto y de gran sencillez, pero brillante en cuanto a jugabilidad y adicción.

Para la realización del juego utilizaremos la herramienta de desarrollo Macromedia Flash MX 2004, última versión de esta popular herramienta que permite magníficos resultados y una alta capacidad de innovación, al ser de gran sencillez de utilización. El gran problema de esta herramienta es que para poder acceder a sus contenidos debe estar preinstalado un plugin en el explorador, pero afortunadamente el peso de este plugin es muy pequeño (al contrario que su herramienta hermana, el Macromedia Director, de mayor potencia y enorme complejidad de uso). La mayor ventaja de esta aplicación es que los productos creados con ella podrán ser accedidos directamente via web, aunque contamos con la posibilidad de guardar el resultado final en un fichero .exe, eliminando la necesidad del plugin en el explorador y obligando a su descarga y almacenamiento en el pc donde se vaya a ejecutar (pero permitiendo su distribución en otros medios como un CDROM). Otro defecto de esta herramienta es que para poder ejecutar un programa, debe estar totalmente descargado en el ordenador que lo ejecuta, pero con el tiempo se ha creado lo que se llama un preloader, que muestra como va la descarga del fichero .swf (esta es la extensión de un fichero compilado de Flash) para tener la idea de cuanto va a tardar en acabar de cargar. Flash ha sido utilizado en internet principalmente con fines no comerciales para la realización de animaciones no interactivas (o con un muy bajo grado de interacción) mostrando un alto grado de creatividad por parte de sus creadores. También podemos encontrar portales de juegos publicitarios enteramente realizados con juegos en Flash ([www.nikefootball.com](http://www.nikefootball.com)) y sitios web donde simplemente demuestran que todavía queda gente con la capacidad de innovar ([www.orisinal.com](http://www.orisinal.com)) y experimentar en el campo del ocio. Flash funciona mediante gráficos vectoriales (De menor consumo de disco pero de mayor simpleza, y con capacidad de ser escalados sin pérdida de calidad) pero goza de la capacidad de importar imágenes no vectoriales (jpg, gif,...) para ser usadas, y utiliza un lenguaje de programación llamado ActionScript, actualmente en su versión 2.0, similar al Java pero de mayor sencillez, para interactuar con las imágenes que tengamos en pantalla y darles un comportamiento. Este lenguaje es un completo lenguaje orientado a eventos (como en el caso del Visual Basic) y en la web podemos encontrar unas cuantas webs de recursos sobre este lenguaje. Por desgracia todas estas webs hacen referencia a los mismos problemas y suelen ser demasiado básicas, teniendo que recurrir a libros especializados, que pecan de ser demasiado avanzados. Al igual que en muchas áreas de conocimiento encontramos el eterno problema del autodidacta: Conocer todo lo

básico pero no ser capaz de entender lo avanzado y no encontrar otra forma de entenderlo que pasarse horas intentando entenderlo sin que nadie te pueda prestar ayuda. También destacar la poca cantidad de foros sobre Flash y la poca calidad de estos.

Dentro de las características del Flash ya hemos destacado su gran facilidad de difusión de contenidos (Al ser vía web, que la dirección sea publicada en alguna web con muchas visitas contribuirá a la rápida difusión de el juego o la animación), pero encontramos un defecto que afecta a los creadores de juegos con esta aplicación, como es no poder excederse demasiado en la complejidad del juego, ya que no es precisamente la herramienta que mejor administra los recursos de los que dispone (Por otra parte lógico, al ser genérica para todo tipo de procesadores). Le sucede lo mismo que al Java, al ser un lenguaje interpretado requiere de lo que llamamos una máquina virtual, es decir, el código no lo ejecutará directamente el computador, primero deberá ser convertido a código que el computador entienda, y esta es la función del plugin que instalamos. Hacer que el ordenador sepa que es un fichero de Flash y que lo adapte al ordenador que lo está ejecutando, con el consecuente gasto de recursos únicamente en la conversión. Sin embargo con el nivel tecnológico actual esto está empezando a dejar de ser un problema, ya que nos podemos permitir malgastar toda la memoria RAM que deseemos, pero debemos tener en cuenta que no todos los usuarios que accedan a los contenidos que creemos van a tener el mismo ordenador que nosotros e igual en algunos casos el juego les funciona mal, por lo que deberíamos testear el juego con diferentes ordenadores y siendo descargado con diferentes velocidades de conexión. Flash incluye un emulador de descarga, para que podamos ver cuanto tarda nuestra aplicación en ser descargada, pero este emulador peca de no ser realista, ya que desde un entorno real puede haber congestiones, y un exceso de uso de la línea, problemas en el servidor y una lista de problemas, pero para hacernos una idea si que nos sirve.

Volviendo al tema principal, para la realización de un juego sobre el oceanográfico deberemos hacer un juego que atraiga a los niños, y no sea excesivamente complejo. Lo adecuado para esto es una simplificación de uno de los géneros existentes (en este caso de los juegos de ladrillos) para crear un juego que nada tiene que ver con este género (ni con ninguno) donde puedan mostrar su habilidad y en caso de derrota deseen volver a jugar para demostrarse a si mismos que eran capaces de más. Observando el mundo real vemos que a los niños les gusta el futbol, y siempre se desafían a ver quien es capaz de hacer más toques con el balón sin que este caiga al suelo, quizá este enfoque pueda parecer exclusivo del género masculino, pero en la actualidad las niñas disfrutan del futbol tanto como los niños. En base a esta idea vamos a hacer un juego que sea una especie de adaptación de esta idea a un mundo digital, donde tengamos una esfera que pueda caer a un lugar donde sea destruida y perdamos el juego, y tengamos varias plataformas donde esta esfera rebote y así no caiga al suelo. Para adaptar esta idea al oceanográfico la basaremos en la sección de los pájaros, consistente en una enorme jaula con diferentes aves y en la que en la parte inferior hay dos pequeños lagos (y un camino para que pasen los visitantes y las observen). A la hora de adoptar este concepto al juego, realizaremos una asociación directa de los elementos del juego con los elementos de la jaula, siendo el escenario en el que jugamos la propia jaula, delimitando por donde la pelota puede estar o no estar, la pelota será un pájaro que nos encontremos en la jaula y los elementos que controlaremos directamente serán unas ramas sobre las que el pájaro saltará y que impedirán que caiga al agua.

## **Objetivo**

Con este juego pretendemos que el usuario pase un rato divertido en el y por tanto asocie diversión con el oceanográfico, y el alto grado de adicción de un juego de estas características contribuye a que el usuario quiera volver a jugar, que es otro objetivo del juego.

## **Público objetivo**

El público objetivo de nuestro juego serán los niños que se hayan registrado en la plataforma y busquen un juego donde puedan probar su habilidad y capacidad de concentración

## **Contenidos y servicios**

El juego gozará de gran sencillez y toda la acción se desarrollará en una única pantalla con fondo selvático. Los elementos que podremos encontrar en el juego serán un pájaro llamado Scott, consistente en 4 fotogramas que unidos formarán una animación del pájaro volando y encontraremos una imagen de un nido, que será situada 3 veces a modo de plataformas sobre las que el pájaro saltará.

El juego se controlará con los cursores, siendo la tecla izquierda utilizada para activar el nido izquierdo, la tecla abajo utilizada para el nido central y la tecla derecha para el nido derecho. El pájaro se moverá como una pelota siguiendo una física no excesivamente realista pero que le da mayor grado de diversión que el uso de una física realista.

En la zona no visible del fichero swf se encuentra la zona en la que acaba el juego, si en algún momento el pájaro toca esa zona, la partida se acaba. No habrá posibilidad de continuar ni se darán vidas extra.

La pantalla de entrada nos ofrecerá una imagen de la mascota del juego sobre un escenario similar (pero no igual) que el que encontraremos durante el juego. Es posible que se efectue algún cameo de los protagonistas de los otros juegos en esta pantalla. Encontraremos también un botón para comenzar a jugar y otro para ver las instrucciones y el nombre del juego: "Scott, el pájaro que no sabía volar".

Las instrucciones del juego estarán redactadas de forma muy directa explicando el problema de Scott (que no sabe volar) y la forma de ayudarlo en sus intentos de aprender a volar, y existirá un único botón para comenzar a jugar. Se considera innecesaria la existencia de un segundo botón que lleve a la pantalla anterior.

## 1.2. Fase de implementación

El fichero Flash generado para la realización del juego de Scott contiene 5 frames, cada uno de los cuales contiene una capa llamada Actionscript en la que se ha escrito todo el código correspondiente a ese frame. Así logramos un mayor orden a la hora de acceder al código fuente, pero por el contrario obtenemos una mayor cantidad de código con referencias a objetos que están en ese momento en pantalla en lugar de asignarle el código a los objetos de forma directa. Considero una forma más ordenada y sencilla de entender y actualizar la forma utilizada.

A continuación encontramos el código fuente de todo el proyecto y los comentarios y explicaciones de las partes que puedan causar algún problema de comprensión.

### FRAME 1 (Pantalla de bienvenida)

```
stop(); // Al inicio de cada frame nos detendremos en el, con lo que obtendremos un
efectivo sistema para movernos a traves del fichero mediante la instrucción
gotoAndPlay(frame); sin perder nunca el control de donde estamos. Las dos
funciones siguientes son un ejemplo de desplazamiento dentro del fichero mediante
botones.

boton_jugar.onPress = function(){
    gotoAndPlay(2);
};
boton_instrucciones.onPress = function(){
    gotoAndPlay(5);
};

sonido_caida = new Sound (); // Definimos una nueva variable de sonido llamada
sonido_caida
sonido_caida.attachSound("caida.wav"); //Al sonido sonido_caida, inicialmente vacio,
le insertamos el sonido "caida.wav", el cual se encuentra en nuestra biblioteca del
Flash.

sonido_pop = new Sound (); // Definimos una nueva variable de sonido llamada
sonido_pop
sonido_pop.attachSound("pop.wav"); //Al sonido sonido_pop, inicialmente vacio, le
insertamos el sonido "pop.wav", el cual se encuentra en nuestra biblioteca del Flash.
```

### FRAME 2 (Preliminares del juego)

```
stop();
puntuacion = ""; // Damos a la puntuación el valor nulo, para que no se muestre por
pantalla.
cuenta = 0; // Esta variable nos servirá para sincronizar el fichero flash en el tiempo
con la animación que muestra la cuenta atrás para comenzar el juego, se basa en
saltar al frame 3 (juego) 3 segundos despues de entrar en este frame. Al estar
```



*configurado el fichero para ejecutarse a 25 frames por segundo, esto nos da un total de 75 frames, que serán los que permaneceremos en el frame 2 antes de saltar al frame 3, como se aprecia en la función siguiente. La animación aún permanece un segundo más en pantalla, pero únicamente por cuestiones estéticas, sin afectar a la marcha del juego.*

```
this.onEnterFrame = function(){
    cuenta++;
    if (cuenta == 75){
        gotoAndPlay(3);
    }
};
```

### FRAME 3 (Juego)

```
stop();
puntuacion = 0; // Iniciamos la cuenta de la puntuación.
vpelotita = Math.floor(Math.random()*10); // Indicamos la velocidad vertical inicial de
la pelota (variable vpelotita), es escogida de forma aleatoria.
vacel = 15; // Especificamos la aceleración inicial de la pelota (variable vacel).
vpelotitax = Math.floor(Math.random()*10-5); //Indicamos la velocidad horizontal
inicial de la pelota (variable vpelotitax);
rebote = 2; // Variable que será utilizada durante el juego para darle variedad
imagen_pelotita=0; // Por razones de lograr una perfecta animación del pájaro en
todo momento, hemos creado una función que reproduce el frame correcto en cada
momento y hace uso de la variable imagen_pelotita,inicializada a 0.

pelotita.onEnterFrame = function(){ // Este código se repite en cada frame de la
pelotita.
    imagen_pelotita++; // Aumentanos la variable imagen_pelotita en 1
    if (imagen_pelotita >=7) { imagen_pelotita = 0}; // En caso de exceder el
número total de frames partido dos (16/2 = 8) de la pelotita, volvemos al cero.
    if (vpelotitax >=0){ // Vemos si la velocidad de la pelota es mayor que 0 (se
mueve a la derecha) o no (se mueve a la izquierda)
        pelotita.gotoAndPlay(1+imagen_pelotita);} // Si es mayor que 0
reproducimos los frames entro 1 y 8 del pájaro, correspondientes al movimiento hacia
la derecha.
    else { pelotita.gotoAndPlay(9+imagen_pelotita);} // Caso opuesto al
anterior, reproducimos los frames entre 9 y 16, correspondientes al movimiento hacia
la izquierda.

//Con las dos siguientes instrucciones movemos la pelotita en el espacio con la
velocidad actual.
    this._y = this._y + vpelotita;
    this._x = this._x + vpelotitax;

    if(vpelotita <= vacel){vpelotita++;}; // Incrementamos la velocidad vertical de
la pelotita en base a la aceleración actual. Realmente podemos apreciar que la
variable vacel no es aceleración, es un limitador de velocidad máxima que será
utilizado para simular aceleración.
```

```

// El siguiente bloque if sirve para alterar las propiedades de la pelotita en caso de
// contacto con algún nido (Botón), en los 3 botones se comporta de forma similar.
if (this.hitTest(boton1)){
    vpelotita = -1*vpelotita; // Al multiplicar por -1 vpelotita, logramos el
// cambio de dirección de la pelotita en el eje vertical.
    puntuacion++; // A cada rebote incrementamos la puntuación en 1

//Cada nido está dividido en 3 secciones, izquierda, centro y derecha, en base a donde
//rebote la pelotita el rebote será de una forma o otra, incrementando la velocidad de la
//pelotita mediante la variable rebote.

    if (pelotita._x <= boton1._x+20){
        _root.sonido_pop.start(); // Al contacto de la pelota con un nido se
//reproducirá el sonido sonido_pop
        if (vpelotitax >= 0){
            vpelotitax = vpelotitax - rebote;
        } else { vpelotitax = vpelotitax + rebote;}
    } else if(pelotita._x >= boton1._x + 50){
        if (vpelotitax >= 0){
            vpelotitax = vpelotitax + rebote;
        } else { vpelotitax = vpelotitax - rebote;}
    }
}
else if(this.hitTest(boton2)){
    _root.sonido_pop.start();
    vpelotita = -1*vpelotita;
    puntuacion++;
    if (pelotita._x <= boton2._x+20){
        if (vpelotitax >= 0){
            vpelotitax = vpelotitax + rebote;
        } else { vpelotitax = vpelotitax - rebote;}
    } else if(pelotita._x >= boton2._x + 50){
        if (vpelotitax >= 0){
            vpelotitax = vpelotitax + rebote;
        } else { vpelotitax = vpelotitax - rebote;}
    }
}
else if (this.hitTest(boton3)){
    _root.sonido_pop.start();
    vpelotita = -1*vpelotita;
    puntuacion++;
    if (pelotita._x <= boton3._x+20){
        if (vpelotitax >= 0){
            vpelotitax = vpelotitax - rebote;
        } else { vpelotitax = vpelotitax + rebote;}
    } else if(pelotita._x >= boton3._x + 50){
        if (vpelotitax >= 0){
            vpelotitax = vpelotitax + rebote;
        } else { vpelotitax = vpelotitax - rebote;}
    }
}
}
}

```

*// Los rebotes contra las paredes son mucho más sencillos, consistiendo simplemente en un cambio de dirección en el eje horizontal, y en un aumento de la velocidad en caso de que la pelotita viaje excesivamente despacio.*

```
if (this.hitTest(pared_izquierda)){
    vpelotitax = -1*vpelotitax;
    if(vpelotitax<=5){
        vpelotitax= 5
    };
    this._x = this._x + vpelotitax;
}
if (this.hitTest(pared_derecha)){
    vpelotitax = -1*vpelotitax;
    if(vpelotitax>=-5){
        vpelotitax= -5
    };
    this._x = this._x + vpelotitax;
}
```

*// La última colisión a verificar es el rebote de la pelotita contra el suelo, elemento no visible, que consistirá en saltar al frame 4 (game over).*

```
if (this.hitTest(suelo)){
    _root.sonido_caída.start(); // En caso de que la pelota caiga al suelo se
    reproducirá el sonido sonido_caída.
    gotoAndPlay("Game_over");
}
};
```

*this.onEnterFrame = function(){ // Al comienzo de cada frame hay que realizar una serie de tareas referentes a los nidos. Por razones de testeo se optó por hacer invisibles las plataformas no seleccionadas, y a pesar de que actualmente no es necesaria esta característica, se ha decidido mantener activa debido a su escaso uso de recursos y altas posibilidades en una actualización futura.*

```
    boton1.gotoAndPlay(1); // Hacemos desaparecer el nido 1 (en su frame 1 es
    invisible, en el 2 es visible)
    boton2.gotoAndPlay(1);
    boton3.gotoAndPlay(1);
    boton1._y = 352.0; // Desplazamos el nido 1 a una zona fuera de la zona
    visible.
    boton2._y = 352.0;
    boton3._y = 352.0;
```

*// Para cada tecla pulsada, situamos el nido asociado a la zona visible de la pantalla y mostramos su frame visible (frame 2).*

```
if(Key.isDown(37)){ // Tecla izquierda
    boton1.gotoAndPlay(2);
```

```

        boton1._y = 252.0;
        boton2._y = 352.0;
        boton3._y = 352.0;
    }

    else if(Key.isDown(40)){ // Tecla abajo
        boton2.gotoAndPlay(2);
        boton1._y = 352.0;
        boton2._y = 252.0;
        boton3._y = 352.0;    }

    else if(Key.isDown(39)){ // Tecla derecha
        boton3.gotoAndPlay(2);
        boton1._y = 352.0;
        boton2._y = 352.0;
        boton3._y = 252.0;    }

```

*// El siguiente conjunto de eventos tienen como única función el hacer el juego más variado, modificando la velocidad y aceleración de la pelotita en base al tiempo (de forma aleatoria) y a la puntuación actual, posibilitando el obtener alturas más altas a mayor puntuación y dándole un grado de aleatoriedad al juego que hace que ninguna partida pueda ser exactamente igual a otra.*

```

        velocidadx = velocidadx + Math.floor(Math.random()*8-4);
        yacel = yacel + Math.floor(Math.random()*8-4);
        if (yacel<= Math.floor(puntuacion/5)){
            yacel = Math.floor(puntuacion/5)
        }
        if (velocidadx <= 2){ velocidadx = 6;};
        if (velocidadx >= -2){ velocidadx = -6;};
        if (yacel >= 35){ yacel = 35;};
    };

```

#### FRAME 4 (Game over)

```

stop();
loadVariablesNum
("http://localhost/juegos/juego_guardar_datos.php?nick="+nick+"&puntuacion="+puntuacion+"&juego=juego1", 0); // Esta larga llamada a una función nos permite
comunicar nuestra puntuación y el usuario activo en el sistema al servidor para su
almacenamiento y posterior muestra en el caso de que el resultado sea uno de los 5
mejores resultados.
boton_jugar2.onPress = function(){
    gotoAndPlay("Juego");
};

```

#### FRAME 5 (Instrucciones)

```
stop();
boton_jugar.onPress = function(){
    gotoAndPlay("Juego");
};
```

## 2. Igor, el pingüino defensor

### 2.1. Fase de estudio

#### Introducción

Hemos decidido que el personaje principal de este juego sea un pingüino, ya que dentro de los animales pertenecientes a la zona del Océano Ártico, quizá sea el animal que puedan gustar más a los niños debido a las características que poseen.

El pingüino protagonista hemos decidido llamarlo Igor.

Esta es la imagen de Igor dentro del juego:



*Igor*

El tipo de publico objetivo al que va dirigido este juego son niños de hasta unos 12 años aproximadamente.

Por tanto para que los niños mas pequeños puedan comprender el juego y su funcionamiento no podemos crear un juego con una mecánica de funcionamiento muy complicada o con unos controles demasiado complicados para niños de esta edad.

Igor únicamente puede desplazarse de izquierda a derecha y disparar.

Los movimientos de Igor se realizaran a través del teclado con las flechas de dirección y el disparo con la barra espaciadora.

Los enemigos del juego son una especie de figuras mutantes, que desde una posición superior nos disparan.

Deberemos esquivar sus disparos y a su vez dispararles para destruirlos.

Los enemigos del juego son figuras ficticias, ya que hemos considerado lo mas correcto, evitando así utilizar otras figuras como animales.

La imagen de fondo del juego es un paisaje con nieve para crear una atmósfera mas real. En esta imagen hemos puesto a modo de guiño o referencia una imagen de Bill, otro de los personajes que integra esta plataforma, con esto lo que se quiere obtener es una integración de distintos juegos dentro de una plataforma común.



*Fondo del juego*

## **Objetivo**

El objetivo del juego es obtener el máximo número de puntos. Para ello deberás destruir todos los mutantes que hay volando por la pantalla, cada uno tiene una puntuación distinta y existe un mutante que aparece unos segundos en pantalla, siendo la puntuación de este la mayor de todas.

## **2.2. Fase de implementación**

Una vez que hayamos accedido al juego, podremos ver en pantalla una precarga para hacer la espera más amena y que el usuario no se desespere y piense que el juego falla.

```
// Precarga
loaded = getBytesLoaded();
total = getBytesTotal();

if (loaded == total) gotoAndStop("Loaded");
pc = int(100 * loaded / total) + "%";
```

*código frame 1*

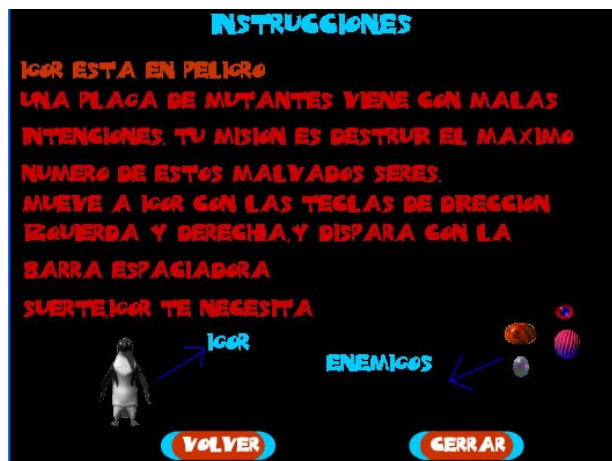
```
gotoAndPlay(1);
```

*código frame 3*

Una vez cargada la precarga, se mostrara por pantalla el menú inicial. Ahí podremos ver la tabla de puntuaciones de los enemigos del juego. Podremos acceder al juego o a las instrucciones.



Si pulsamos el botón de instrucciones, accederemos al menú de instrucciones donde podremos ver unas instrucciones básicas del juego



También hemos incluido un botón cerrar, el cual cerrará la ventana actual.

Una vez visualizada las instrucciones donde vemos los controles del juego, ya estamos en plena disposición de poder jugar. Para ello bastara con pulsar el botón de Jugar.

En el primer frame de la capa de juego tenemos el siguiente código necesario para inicializar las variables.

```
//  
// Igor,el pinguino defensor  
//  
  
setProperty ("LaserFire", _visible, false);
```



```

setProperty ("AlienFire1", _visible, false);
setProperty ("AlienFire2", _visible, false);
setProperty ("AlienFire3", _visible, false);
setProperty ("AlienFire4", _visible, false);
setProperty ("Spaceship", _visible, false);
setProperty ("Alien1", _visible, false);
setProperty ("Alien2", _visible, false);
setProperty ("Alien3", _visible, false);

```

```

for (var i=4; i<9; i++) {
    setProperty ("Life"+i, _visible, false);
}

```

```

lives = 3;
score = 0;
shipScore = 50;
level = 0;

```

A este código le añadimos en el frame 2 el siguiente código que se encarga del movimiento de los enemigos

```

if (level < 84) level += 10;

```

```

shifts = 1;
xMin = 400;
xMax = 0;
dir = 4;
alienHeight = 35 + level;
aliensDead = 0;
newLife = 0;
tonePos = 1;

```

```

for (i=1; i<=lives; i++) {
    setProperty("Life" + i, _visible, true);
}

```

```

for (i=0; i<11; i++) { // POSICION DE CADA FILA DE ENEMIGOS
    duplicateMovieClip("Alien1", "Row1" + i, i);
    duplicateMovieClip("Alien2", "Row2" + i, i + 11);
    duplicateMovieClip("Alien2", "Row3" + i, i + 22);
    duplicateMovieClip("Alien3", "Row4" + i, i + 33);
    duplicateMovieClip("Alien3", "Row5" + i, i + 44);
    for (j=1; j<6; j++) {
        setProperty("Row" + j + i, _x, 102 + 36 * i); // NIVEL PARA BAJAR
        CUANDO LLEGAN AL FINAL
        setProperty("Row" + j + i, _y, alienHeight + 30 * j);
    }
}

```

Con el código de estos dos primeros frames ya tenemos los enemigos en pantalla con su movimiento en horizontal y en vertical cuando llega a un extremo de la pantalla. La variables store almacenara las puntuaciones y la variable lives el número de vidas que disponemos.

Durante el juego aparecerá en pantalla un enemigo fugaz que atravesara la pantalla de forma horizontal y a gran velocidad. Este enemigo en función de donde lo destruyas obtendrás una puntuación u otra. El código del enemigo es el siguiente.

```
tellTarget ("Spaceship") { //CODIGO DEL MALO FUGAZ
if (_visible) {
if (_root.SsSound._currentframe == 1 && _currentframe == 1) _root.SsSound.play();
if (_root.shipScore == 0 || _x<0) {
_root.shipScore = 50;
_visible = false;
_x = 550;
gotoAndStop(1);
}
if (_root.shipScore>=50) _x -= 10;
}
}
```



*Enemigo rapido*

A continuación mostramos el código cuando destruimos al mutante fugaz.

```
// El laser golpea al malo fugaz
if (LaserFire.hitTest(Spaceship) && Spaceship._currentframe == 1) {
stopAllSounds();
Sound.gotoAndPlay("KillSS");
setProperty ("LaserFire", _y, -20);
shipScore = random(6)+1;
score += 50*shipScore;
```

```

Spaceship.gotoAndStop(shipScore+1);
SsSound.gotoAndStop(1);
shipScore = 40;
}

```

Con esto ya tenemos el código del enemigo fugaz. A continuación veremos una muestra del código del resto de enemigos.

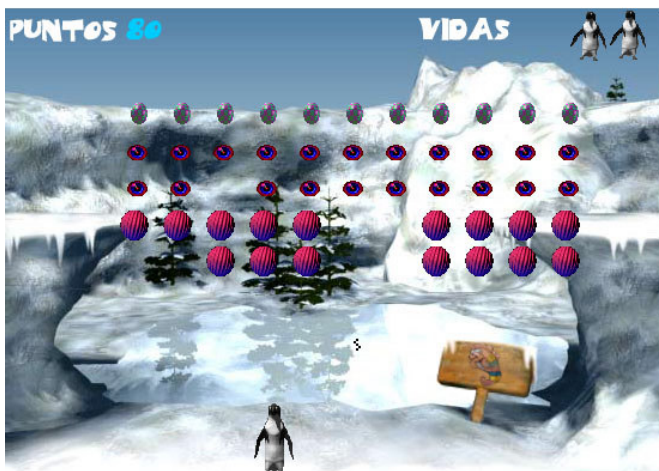
Todo este código afecta al comportamiento de los enemigos principales, en el que indica su movimiento a derecha e izquierda, la carga de los ficheros de sonido y un comportamiento normal en el caso de que no reciba ningún disparo nuestro. Cuando destruimos a un enemigo este desaparecerá. El código asociado es el siguiente:

```

// El disparo del pinguino golpea a un malote
for (var i=0; i<11; i++) {
  for (var j=1; j<6; j++) {
    thisAlien = eval("Row"+j+i);
    if (LaserFire.hitTest(thisAlien)) {
      Sound.gotoAndPlay("Kill");
      aliensDead++;
      LaserFire._y = -25;
      Explode._x = thisAlien._X;
      Explode._y = thisAlien._y;
      Explode.gotoAndPlay(2);
      removeMovieClip (thisAlien);
      if (j==5 || j==4) score += 10;
      else if (j==3 || j==2) score += 20;
      else score += 40;
    }
  }
}
}

```

Los enemigos disparan de manera aleatoria.



El código de estos disparos es el siguiente.

```
// Disparos de los malotes
for (var fire=4; fire>0; fire--) {
    if (!AlienFire4._visible && random(20)>1 && fire == 4) fire = 3;
    alienFire = eval("AlienFire"+fire);
    tellTarget (alienFire) {
        if (_visible) {
            if (this.fire == 4) _y += 12;
            else _y += 8;
            if (_y>412) _visible = false;
        } else if (!_root.newLife && random(170-_root.alienHeight)<1) {
            var i = random(11);
            for(var j=5; j>0; j--) {
                firePos = eval("_root.Row"+j+i);
                if (firePos._visible) {
                    _x = firePos._x;
                    _y = firePos._y+36;
                    _visible = true;
                    j = 0;
                }
            }
        }
    }
}
```

Cuando destruyo todos los enemigos de la pantalla, recibiré 1000 puntos y una vida extra en el caso de que mi numero de vidas sea menor que 2. El código es el siguiente.

```
// SI MATAS A TODOS PASAS DE NIVEL
    if (allGone) {
        score += 1000;
        if (lives<2) lives++;
        gotoAndPlay(2);
    }
}
```

Cuando recibo un disparo de los enemigos, cargo el fichero de audio correspondiente y continuo la partida.

```
// Los malos me dan
    if (!newLife && alienFire._visible && alienFire.hitTest(LaserBase)) {
        Sound.gotoAndPlay("Boom");
        alienFire._visible = false;
        setProperty ("Life"+lives, _visible, false);
        LaserBase.gotoAndStop(2);
        newLife = 3;
        lives--;
    }
}
```

Una vez que mi numero de vidas es igual a 0, la partida habra finalizado  
El código es el siguiente.

```
// Los malos me han vencido
if (lives == 0) {
    LaserBase.gotoAndStop(2);
    gotoAndPlay ("Game Over");
}
```

Una vez que la partida haya finalizado, se mostrara una ventana con la puntuación obtenida. Esto lo obtenemos mostrando por pantalla la variable score.



Una vez mostrada la puntuación podremos volver a jugar o volver a visitar las instrucciones de nuevo.

Para almacenar la puntuación en el registro de puntuaciones de la plataforma, añadimos el siguiente código

```
loadVariablesNum
("http://81.202.10.177/juegos/juego_guardar_datos.php?nick="+nick+"&puntuacion="+
score+"&juego=juego2", 0);
for (i=0; i<11; i++) {
    for (j=1; j<6; j++) {
        removeMovieClip("Row" + j + i);
    }
}
```

### 3. Bill, el cazador de estrellas

### 3.1. Fase de estudio

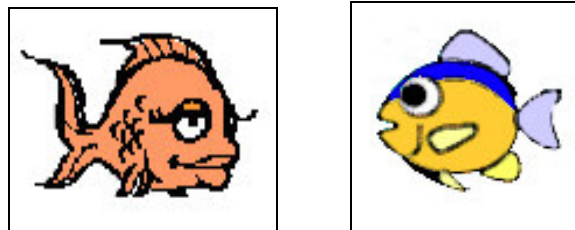
Debemos desarrollar un juego para que pase a formar parte de esta plataforma. El juego deberá reunir los requisitos para que encaje perfectamente dentro del contexto de nuestra plataforma y a su vez sea un juego en el que el jugador despierte su interés en los elementos que aparecen en el juego, que en nuestro caso son animales que se encuentran en el Oceanográfico.

Pues bien, el juego que hemos desarrollado se titula **“Bill, el cazador de estrellas”**. Se trata de un juego de habilidad, en el cual el jugador maneja al personaje Bill, un caballito de mar, con la misión de ir recogiendo estrellas de mar mientras esquiva a los diferentes enemigos que van apareciendo en pantalla.

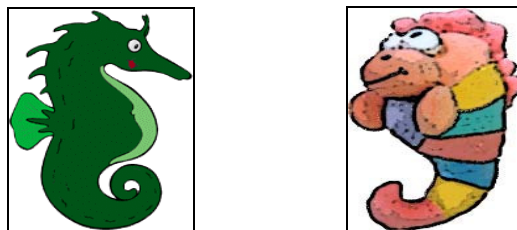
Proceso de creación de personajes.

El proceso de la creación del personaje principal ha sido un proceso más largo del esperado inicialmente. En primer lugar se decidió que nuestro personaje protagonista fuera un pez, el cual pasó por diferentes versiones durante la creación de este personaje.

Pero más tarde la idea de que el protagonista fuera un pez se deshechó pasando a convertirse en un caballito de mar, el cual también sufrió bastantes cambios desde la versión inicial del personaje hasta la versión definitiva que aparece en nuestro juego.



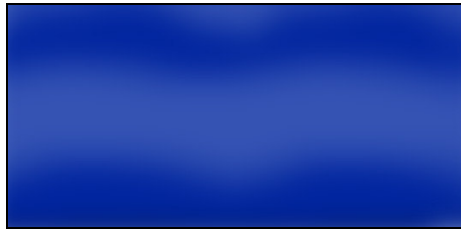
*al principio Bill era un pez*



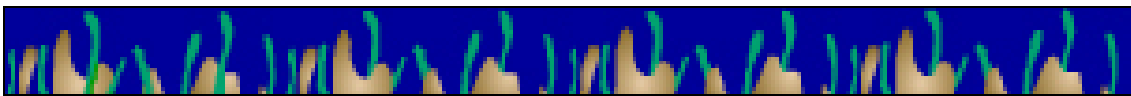
*primera versión de Bill*

*Bill en su versión final*

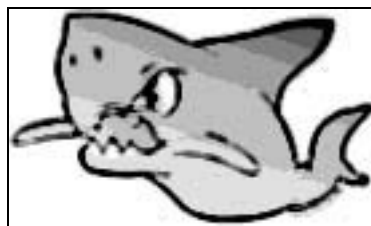
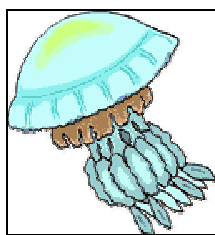
Una vez creado el personaje pensamos en la creación del escenario, el cual no nos creó muchos problemas, ya que era bastante obvio que se iba a tratar del fondo del mar. Decidimos insertar dos elementos básicos en el escenario. El primero de ellos se trataba del fondo del mar, el cual se desarrolló fácilmente utilizando la herramienta Adobe Photoshop. Dentro de este programa utilizamos varios degradados con diferentes tonos de color azul y varios filtros con los cuales conseguimos realizar un fondo adecuado para nuestro juego.



El segundo elemento de nuestro escenario es la parte inferior del fondo. Decidimos que se tratase de un fondo rocoso en el cual aparecieran también algas marinas. Con esto hemos pretendido que el fondo de nuestro juego no diera la impresión de ser un fondo muy monótono en el cual solo apareciera un fondo acuático, sino que con esta parte inferior del fondo aparecieran más elementos en pantalla y mayor colorido, ofreciendo así un mayor atractivo visual al jugador.

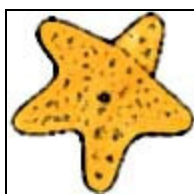


El siguiente paso era determinar los enemigos que nos iríamos encontrando durante el desarrollo del juego. El primer paso que dimos para la selección de los enemigos fue el confeccionar un listado en el que aparecieran animales que podrían encajar en el grupo de animales potencialmente peligrosos que podemos encontrar en el fondo del mar. Los animales más destacados que encontramos en esta lista eran los siguientes: pulpos, anguilas eléctricas, cangrejos, tiburones, pirañas, orcas, peces espada, peces martillo, medusas... El primer animal que descartamos de la lista fue la orca, ya que las dimensiones de este animal eran demasiado grandes para poder incluirlo en nuestro juego. Otro de los animales que descartamos también al principio fue el cangrejo, debido a que es un animal que no puede nadar, por lo que en nuestro juego no podría hacer gran cosa. Después de realizar los diferentes descartes nos quedamos con las pirañas, medusas, pulpos y tiburones, pero finalmente nos vimos obligados a descartar la idea de incluir pulpos debido a que no se nos ocurría ningún comportamiento adecuado dentro del juego para este animal. Así pues, los tres diferentes tipos de enemigo que podemos encontrar en nuestro juego son las pirañas, las medusas y los tiburones.



El siguiente paso era la creación del elemento que tenía que ser nuestro objetivo durante todo el juego. En un principio pensamos en la idea de que Bill fuera recogiendo huevos de caballito de mar, pero más adelante surgió la idea de que el objetivo de Bill fuera recoger estrellas de mar. Muchos fueron las ventajas de incluir estrellas de mar en nuestro juego. La primera y más importante de todas era el hecho de poder incluir de manera indirecta un nuevo animal dentro del juego. Así pues decidimos que el color de las estrellas de mar fuera dorado. Con esto pretendíamos ofrecer la metáfora de que el hecho de recoger estrellas de mar doradas sea comparable al hecho de recoger monedas

de oro, el cual es un objetivo bastante habitual en cualquier juego arcade o de habilidad.



Una vez desarrollados los elementos que van a actuar dentro de nuestro juego, pasamos a explicar los niveles que compondrán el mismo. El juego estará formado por tres niveles o pantallas, los cuales presentarán una dificultad que irá incrementándose conforme el jugador pase de nivel. En el primer nivel el jugador deberá atrapar un número de 50 estrellas de mar, mientras que los enemigos que aparecerán en pantalla serán las pirañas. Durante la segunda pantalla la dificultad aumentará en dos aspectos. El primero de ellos será que el número de estrellas que se deberá recoger será 75. Por otro lado los enemigos con los que nos encontraremos serán otra vez las pirañas y un nuevo tipo de enemigos que aparecerán en este nivel que serán las medusas. Y por último está el tercer nivel, en el cual el número de estrellas que habrá que atrapar volverá a aumentar, en esta ocasión la cifra de estrellas que se deberá conseguir será la de 100 estrellas. Y como no, los enemigos serán todavía más peligrosos. En esta ocasión a las pirañas y medusas que aparecen en el nivel 2, se añadirá un tercer enemigo que entrará en juego. Estos nuevos enemigos serán los tiburones, los cuales dotarán de mayor complejidad a este nivel.

Pues bien, con el desarrollo y creación de todos estos elementos ya estábamos en disposición de comenzar a realizar la implementación y el desarrollo del propio juego.

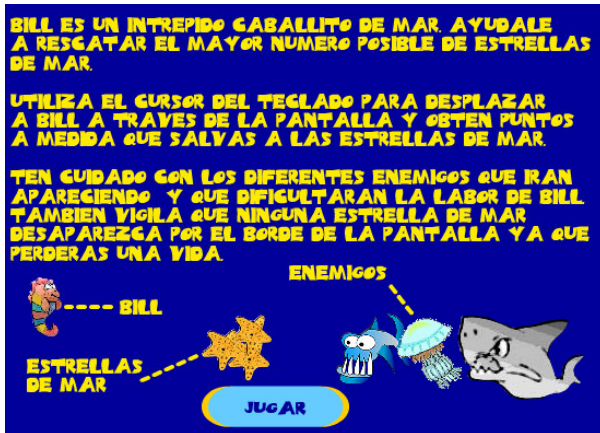
### 3.2. Fase de implementación

El juego comienza con la ventana de inicio, en la cual aparecerá el título del juego, un fondo de pantalla a modo de cartel o presentación de este y tres opciones en forma de botón: “jugar”, “instrucciones” y “cerrar”.



En el caso de escoger la opción “jugar” se saltará directamente al primer frame del juego, mientras que si se decide escoger la opción “instrucciones” se saltará al frame de las instrucciones del juego. Una vez allí el jugador tendrá ocasión de poder leer las instrucciones del juego y tendrá el botón “jugar” como única opción.





### Código asociado al botón “jugar”:

```
on (release) {
    gotoAndPlay(15);
    _root.sonido_water.start();
}
```

Al pinchar sobre el botón provoca el salto al frame 15, el cual es el frame del primer nivel del juego. Por otro lado se reproduce el sonido “sonido\_water”.

### Código asociado al botón “instrucciones”:

```
on (release) {
    gotoAndPlay(2);
    _root.sonido_water.start();
}
```

Al pinchar sobre el botón provoca el salto al frame 2, el cual es el frame de la pantalla de instrucciones del juego. Al igual que en botón “jugar”, se reproduce el sonido “sonido\_water”.

### Código asociado al botón “cerrar”:

```
on (press){
    getURL("../cerrar.php");
}
```

Al pinchar sobre el botón se ejecuta el fichero cerrar.php, el cual incluye una función JavaScript que se encarga de cerrar la ventana actual

Una vez situados dentro del juego hay que tener en cuenta ciertos aspectos antes de comenzar la explicación del comportamiento de los elementos que aparecen en pantalla. Cabe comentar, en primer lugar, la inicialización de ciertas variables que van a

estar activas mientras el juego se esté ejecutando.

En primer lugar está la variable “vidas”. Esta variable contendrá el número de vidas que le quedan a nuestro personaje. Por lo que inicialmente la debemos inicializar al valor que le queramos dar al número de vidas de nuestro personaje. El valor que se ha decidido dar es de tres vidas. Esta variable ha sido inicializada en el primer frame de la capa asociada a nuestro personaje, como ya veremos más adelante, de la siguiente manera: **\_root.vidas=3;** . De esta manera conseguimos que la variable se inicie al principio de cada partida y seguidamente se irá decrementando conforme el jugador vaya perdiendo vidas.

```
_root.vidas;
```



en la parte superior derecha siempre aparecerá el número de vidas que posee en personaje.

Otra de las variables que se tiene en cuenta es la variable “score”. Esta variable determina el número de estrellas que deben ser recogidas. Esta variable se inicializará siempre al principio de cada nivel. Debido a que el juego está compuesto por tres niveles, la variable se inicializará en el primer frame de cada nivel y luego se mantendrá activa durante el resto del nivel, decrementándose en una unidad cada vez que el personaje coja una estrella. En caso de que la variable “score” tome el valor 0 se saltará al frame de “Nivel superado”.

```
_root.score=50;
```

En el nivel 1 serán 50 las estrellas que se tendrán que recoger.

```
_root.score=75;
```

En el nivel 2 habrá que recoger 75 estrellas.

```
_root.score=100;
```

Finalmente, en el nivel 3 se deberá recoger 100 estrellas.

Finalmente, la tercera variable activa durante todo el juego es la variable “puntos”. Esta variable, como su propio nombre indica, determina el número de puntos que lleva acumulados el jugador durante la partida. Al igual que la variable “vidas”, la variable “puntos” se inicializará en el primer frame del juego (**\_root.puntos=0;** evidentemente se inicializará a 0), y se irá incrementando en 1 unidad cada vez que el personaje se apodere de una estrella. En el caso de que se complete un nivel con éxito, este número de puntos se multiplicará por el número de vidas que le queden al jugador, recompensando así el hecho de completar un nivel con el máximo número de vidas posible.

```
_root.puntos=(_root.puntos)*(_root.vidas);
```

Este código irá asociado a los frames de “Nivel1finished”, “Nivel2finished” y “Nivel3finished” como ya veremos más adelante.

Por otro lado tanto el número de vidas como el número de estrellas restantes y el número de puntos siempre serán visibles a lo largo de la partida. Para ello se ha creado un cuadro de texto al que se le asocia siempre el valor de la variable y que aparecerán siempre en la parte superior de la pantalla.

Una vez comentado el comportamiento de estas variables ya estamos en condiciones de comenzar a analizar el comportamiento del juego.

## Personaje principal

En primer lugar veremos el comportamiento de nuestro personaje. Para ello comentaremos directamente su código asociado:

### Código asociado al objeto “Bill”:

```
onClipEvent (load) {  
    velocidad = 20;  
    function reset() {  
        this._x = 130;  
        this._y = 130;  
    }  
    setProperty(this, _x, 130);  
    setProperty(this, _y, 130);  
}  
onClipEvent (enterFrame) {  
    if ((Key.isDown(Key.RIGHT)) && (this._x<=520)) {  
        this._x += velocidad;  
    } else if ((Key.isDown(Key.LEFT)) && (this._x>=20)) {
```

```

        this._x -= velocidad;
    }
    if ((Key.isDown(Key.DOWN)) && (this._y<=380)) {
        this._y += velocidad;
    } else if ((Key.isDown(Key.UP)) && (this._y>=20)) {
        this._y -= velocidad;
    }
}

```

La variable “velocidad” determina la rapidez con la que se mueve nuestro personaje por pantalla. El valor que se le ha decidido dar es el de 20.

A continuación se ha creado una función reset. Esta función servirá para reinicializar la posición de nuestro personaje en pantalla cada vez que haga una llamada a esta función (esto se hará cada vez que nuestro personaje pierda una vida, o bien impactando con un enemigo o bien dejando que una estrella se pierda por el fondo de la pantalla). Esto lo veremos más adelante, tanto en el código de los enemigos como en el código de las estrellas.

En cuanto al código asociado al onClipEvent(enterFrame) cabe comentar que determina el movimiento de nuestro personaje a través de la pantalla. Hemos introducido ciertas restricciones cada vez que se aprieta una tecla del cursor. Por ejemplo en la primera condición **if** vemos que aparece la clausura (this.\_x <=520). Esto quiere decir que siempre que se aprete la tecla derecha del cursor y que las coordenadas de nuestro personaje en el eje x sean menor o igual que 520, este se moverá hacia la derecha en este eje. Mientras que en el caso de que la clausura (this.\_x <=520) no se cumpla, el hecho de apretar la tecla derecha del cursor no producirá ningún efecto en el movimiento del personaje.

Pues bien, como se puede ver este código se ha aplicado para cada una de las teclas de movimiento de Bill y para cada uno de los ejes, dado que se moverá de izquierda a derecha y de arriba a abajo.

## Estrellas de mar

El siguiente paso va a ser el análisis del código asociado a las estrellas que nuestro personaje irá recogiendo durante todo el juego.

### Código asociado a los objetos “Estrella”:

```

onClipEvent (load) {
    function reset() {
        this._x = 650;
        this._y = random(300);
        enemySpeed = random(4)+2;
    }
    reset();
}
onClipEvent (enterFrame) {

```

```
this._x -= enemySpeed;
if (this._x<20) {
    _root.sonido_ow.start();
    _root.vidas -= 1;
    _root.bill2a.reset();
    _root.estrella1.reset();
    _root.estrella2.reset();
    _root.estrella3.reset();
    _root.estrella4.reset();
    _root.estrella5.reset();
    _root.estrella6.reset();
    _root.estrella7.reset();
    _root.estrella8.reset();
    _root.piranya1.reset();
    _root.piranya2.reset();
    _root.piranya3.reset();
    _root.piranya4.reset();
    _root.piranya5.reset();
    _root.piranya6.reset();
    _root.piranya7.reset();
    _root.piranya8.reset();
    _root.medusa1.reset();
    _root.medusa2.reset();
    _root.medusa3.reset();
    _root.medusa4.reset();
    _root.medusa5.reset();
    _root.medusa6.reset();
}
```

```

        _root.medusa7.reset();
        _root.medusa8.reset();
        _root.medusa9.reset();
        _root.medusa10.reset();
        _root.tiburon1.reset();
        _root.tiburon2.reset();
        if (_root.vidas == 0) {
            _root.gotoAndStop("GameOver");
        }
        if (_root.vidas != 0) {
            _root.nextFrame();
        }
    }
}

if (this.hitTest(_root.bill2a)) {
    _root.sonido_pop.start();
    _root.score -= 1;
    _root.puntos += 1;
    reset();
}
if (_root.score == 0) {
    reset();
    _root.gotoAndStop("Nivel1finish");
}
}
}

```

Analizando más detenidamente este código podemos ver que al igual que hemos hecho con el objeto Bill, hemos creado una función reset para reinicializar en pantalla los objetos estrella. Podemos ver que siempre que se llame a esta función, las estrellas aparecerán en el punto 650 del eje x y en un punto aleatorio del eje y situado dentro del intervalo 0...300. Con esto conseguimos que las estrellas no aparezcan siempre por el mismo punto de la pantalla. También aparece una variable enemySpeed que determinará la velocidad del objeto. Como podemos ver la velocidad también será aleatoria. Para ello obtendremos un número aleatorio entre el 1 y el 4, y a este número le sumaremos 2, asegurándonos así una velocidad mínima de 3 y una velocidad máxima de 6.

Seguidamente analizaremos el código asociado al onClipEvent(enterFrame). El primer caso que hemos tratado es el caso en que una estrella atravesase el borde izquierdo de la pantalla, en cuyo caso Bill tiene que perder una vida. Esto se puede ver en la condición if (this.\_x<20). Si se produce la condición entonces ocurrirán una serie de eventos.

\_root.sonido\_ow.start(); :se reproducirá el sonido “sonido\_ow”, el cual se reproduce cada vez que Bill pierde una vida.

\_root.bill2a.reset(); \_root.piranya1.reset(); ... ; \_root.piranya8.reset();  
\_root.medusa1.reset(); ... ; \_root.medusa10.reset(); \_root.tiburon1.reset();

\_root.tiburon2.reset(); \_root.estrella1.reset(); ... ; \_root.estrella8.reset(); :se resetearán todos los objetos que se encuentren activos en pantalla en ese momento.

\_root.vidas-=1; : nuestro personaje perderá una vida.

Cada vez que Bill pierda una vida también se hará la siguiente comprobación. En el caso de que tras perder una vida el número de vidas tome el valor de 0, se saltará al frame “GameOver”, el cual indicará que el juego ha terminado.

Por otro lado hay que realizar el proceso de recogida de estrellas. Ese proceso lo hemos realizado en la siguiente condición: `if (this.hitTest(_root.bill2a))`

Si la condición se cumple es debido a que el objeto estrella ha chocado con el objeto `bill2a` (el cual representa a nuestro personaje). A partir de ahí se producen las siguientes operaciones:

`_root.sonido_pop.start();` : se reproduce el sonido “sonido\_pop”, el cual sonará cada vez que Bill se apodere de una estrella.

`_root.score -=1;` : se resta una estrella al contador de estrellas que hay que obtener.

`_root.puntos +=1;` : se añade un punto a la puntuación total.

`reset();` : se resetea el objeto estrella, de forma que una nueva estrella aparece en pantalla.

Finalmente, se realiza una última comprobación. Esta es que el número de estrellas por recoger sea superior a 0. En caso de que el número de estrellas tome el valor de 0, el programa saltará al frame con etiqueta “Nivel1finish”, “Nivel2finish” o “Nivel3finish”, dependiendo claro está del nivel en el que nos encontremos.

## Enemigos

Seguidamente pasaremos a ver el código asociado a los enemigos que aparecen en el juego. Antes de nada haremos un breve resumen de cada uno de los enemigos y del comportamiento que tienen estos en pantalla.

El primer enemigo que nos encontramos es la **Piraña**. Las pirañas se comportarán de la siguiente manera: aparecerán por el borde derecho de la pantalla y avanzarán a una velocidad aleatoria hasta el borde izquierdo hasta que desaparezcan, en cuyo caso una nueva piraña aparecerá por el borde derecho. Dependiendo del nivel en el que nos encontremos aparecerá un número distinto de pirañas por pantalla.

El segundo tipo de enemigo en la lista es la **Medusa**. El comportamiento de la medusa es distinto a de las pirañas. Estas aparecerán desde el borde inferior de la pantalla y ascenderán en diagonal, también a una velocidad aleatoria, hasta el borde superior donde finalmente desaparecerán. Al igual que las pirañas, cada vez que una medusa desaparezca por el borde superior una nueva medusa aparecerá por el borde inferior.

Finalmente encontramos al tercer y último enemigo que aparecerá en el juego. Se trata del **Tiburón**. El tiburón tiene un comportamiento parecido al de la piraña. Este aparecerá desde el borde derecho de la pantalla y avanzará a una velocidad aleatoria hasta el borde izquierdo, pero al contrario de lo que ocurre con la piraña, cuando el tiburón llegue hasta el borde izquierdo, en lugar de desaparecer invertirá su rumbo, avanzando así hasta la borde derecho, donde volverá a invertir su rumbo en el momento en que llegue hasta allí. Otro de los aspectos que hacen más peligroso al tiburón es el hecho de su tamaño, ya que al ser más grande hace que sea más peligroso impactar contra él.

Pues bien, al tratarse el nivel 1 el teóricamente más fácil, solo aparecerán pirañas en pantalla como enemigos. El número de pirañas que nos encontraremos será de 8. En el nivel 2 a parte de pirañas también encontraremos medusas, por lo que el número de pirañas será menor. En este caso encontraremos 6 pirañas y 10 medusas continuamente en pantalla. Finalmente en el nivel 3 también encontraremos 6 pirañas y 10 medusas simultáneamente en pantalla, a lo que se añadirán 2 tiburones que se estarán moviendo constantemente por el escenario.

Una vez realizada esta breve explicación del comportamiento de los objetos ya estamos en condiciones de analizar el código asociado a los mismos:

### **Código asociado a los objetos “Piraña”:**

```
onClipEvent (load) {
function reset(){
this._x=550;
this._y=random(400);
enemySpeed=random(24)+5;
}
reset();
}

onClipEvent (enterFrame) {

this._x-=enemySpeed;
if (this._x<-240) {
reset();
}
if (this.hitTest(_root.bill2a)){
    _root.sonido_ow.start();

    _root.bill2a.reset();
    _root.piranya1.reset();
    _root.piranya2.reset();
    _root.piranya3.reset();
    _root.piranya4.reset();
    _root.piranya5.reset();
    _root.piranya6.reset();
    _root.piranya7.reset();
    _root.piranya8.reset();
    _root.medusa1.reset();
    _root.medusa2.reset();
    _root.medusa3.reset();
    _root.medusa4.reset();
    _root.medusa5.reset();
    _root.medusa6.reset();
    _root.medusa7.reset();
    _root.medusa8.reset();
    _root.medusa9.reset();
}
```



```

    _root.medusa10.reset();
    _root.tiburon1.reset();
    _root.tiburon2.reset();
    _root.estrella1.reset();
    _root.estrella2.reset();
    _root.estrella3.reset();
    _root.estrella4.reset();
    _root.estrella5.reset();
    _root.estrella6.reset();
    _root.estrella7.reset();
    _root.estrella8.reset();

    _root.vidas-=1;
    if(_root.vidas==0){
        _root.gotoAndStop("GameOver");}
    if(_root.vidas!=0){
        _root.nextFrame()};
}

```

Pasando a analizar el código vemos que al igual que hemos hecho con los objetos Bill y estrella, hemos creado una función reset para reinicializar en pantalla los objetos piraña. Siempre que se llame a esta función, las pirañas aparecerán en el punto 550 del eje x y en un punto aleatorio del eje y situado dentro del intervalo 0...400. También aparece una variable enemySpeed que determinará la velocidad del objeto. La velocidad también será aleatoria. Para ello obtendremos un número aleatorio entre el 1 y el 24, y a este número le sumaremos 5, asegurándonos así una velocidad mínima de 6 y una velocidad máxima de 29.

Analizando el contenido de onClipEvent (enterFrame), la primera línea con la que nos encontramos determina el movimiento de la piraña por el escenario.

A continuación comprobamos que las coordenadas del objeto en el eje x sean iguales o superiores a -240. En caso contrario se hará una llamada a la función reset() del objeto.

Seguidamente nos encontramos con la condición **if (this.hitTest(\_root.bill2a))**. Esta condición determina, por medio de la función hitTest, si el objeto piraña impacta con el objeto bill2a (objeto que representa a nuestro personaje Bill). En caso de que la condición sea afirmativa se efectuarán las siguientes operaciones:

```

_root.sonido_ow.start(); :se reproducirá el sonido "sonido_ow", el cual se reproduce
cada vez que Bill pierde una vida.
_root.bill2a.reset(); _root.piranya1.reset(); ... ; _root.piranya8.reset();
_root.medusa1.reset(); ... ; _root.medusa10.reset(); _root.tiburon1.reset();
_root.tiburon2.reset(); _root.estrella1.reset(); ... ; _root.estrella8.reset(); :se resetearán
todos los objetos que se encuentren activos en pantalla en ese momento.
_root.vidas-=1; : nuestro personaje perderá una vida.

```

Cada vez que Bill pierda una vida también se hará la siguiente comprobación. En el caso de que tras perder una vida el número de vidas tome el valor de 0, se saltará al frame "GameOver", el cual indicará que el juego ha terminado.

### Código asociado a los objetos “Medusa”:

```
onClipEvent (load) {
    function reset() {
        this._x = random(1000);
        this._y = 550;
        enemySpeed = random(8)+2;
    }
    reset();
}
onClipEvent (enterFrame) {
    this._y -= enemySpeed;
    this._x -= enemySpeed;
    if (this._y<-220) {
        reset();
    }
}
if (this.hitTest(_root.bill2a)) {
    _root.sonido_ow.start();
    _root.bill2a.reset();
    _root.piranya1.reset();
    _root.piranya2.reset();
    _root.piranya3.reset();
    _root.piranya4.reset();
    _root.piranya5.reset();
    _root.piranya6.reset();
    _root.piranya7.reset();
    _root.piranya8.reset();
    _root.medusa1.reset();
    _root.medusa2.reset();
    _root.medusa3.reset();
    _root.medusa4.reset();
    _root.medusa5.reset();
    _root.medusa6.reset();
    _root.medusa7.reset();
    _root.medusa8.reset();
    _root.medusa9.reset();
    _root.medusa10.reset();
    _root.estrella1.reset();
    _root.estrella2.reset();
    _root.estrella3.reset();
    _root.estrella4.reset();
    _root.estrella5.reset();
    _root.estrella6.reset();
    _root.estrella7.reset();
    _root.estrella8.reset();
    _root.vidas -= 1;
    if (_root.vidas == 0) {
        _root.gotoAndStop("GameOver");
    }
}
```

```

    } else {
        _root.nextFrame();
    }
}0

```

Podemos ver que el comportamiento del objeto Medusa es bastante parecido en comportamiento al objeto Piraña. Aun así hay algunas pequeñas diferencias. Estas diferencias residen en el movimiento de los objetos. Como hemos dicho antes las medusas se moverían de abajo a arriba en diagonal hacia la izquierda. Eso lo hemos realizado en las líneas `this._y -= enemySpeed; this._x -= enemySpeed;` También cabe comentar la comprobación de que las medusas han cruzado el borde superior de la pantalla, en cuyo caso se realiza una llamada a la función `reset()` reiniciando así la medusa su posición en pantalla y también su valor de velocidad.

### Código asociado a los objetos “Tiburón”:

```

onClipEvent (load) {
    izq = 1;
    function reset() {
        this._x = 600;
        this._y = random(400);
        enemySpeed = random(8)+5;
    }
    reset();
}
onClipEvent (enterFrame) {
    if ((this._x>=20) && (izq == 1)) {
        izq = 1;
        this._x -= enemySpeed;
    }
    if ((this._x<20) && (izq == 1)) {
        izq = 0;
        this._x+enemySpeed;
    }
    if ((this._x<=520) && (izq == 0)) {
        izq = 0;
        this._x += enemySpeed;
    }
    if ((this._x>520) && (izq == 0)) {
        izq = 1;
        this._x -= enemySpeed;
    }
}
onClipEvent (enterFrame) {
    if (this.hitTest(_root.bill2a)) {
        _root.sonido_ow.start();
        _root.bill2a.reset();
        _root.vidas -= 1;
        _root.piranya1.reset();
        _root.piranya2.reset();
    }
}

```

```

        _root.piranya3.reset();
        _root.piranya4.reset();
        _root.piranya5.reset();
        _root.piranya6.reset();
        _root.piranya7.reset();
        _root.piranya8.reset();
        _root.medusa1.reset();
        _root.medusa2.reset();
        _root.medusa3.reset();
        _root.medusa4.reset();
        _root.medusa5.reset();
        _root.medusa6.reset();
        _root.medusa7.reset();
        _root.medusa8.reset();
        _root.medusa9.reset();
        _root.medusa10.reset();
        _root.tiburon1.reset();
        _root.tiburon2.reset();
        _root.estrella1.reset();
        _root.estrella2.reset();
        _root.estrella3.reset();
        _root.estrella4.reset();
        _root.estrella5.reset();
        _root.estrella6.reset();
        _root.estrella7.reset();
        _root.estrella8.reset();
        if (_root.vidas == 0) {
            _root.gotoAndStop("GameOver");
        }
        if (_root.vidas != 0) {
            _root.nextFrame();
        }
        reset();
    }
}

```

Como se puede comprobar, el comportamiento del objeto “Tiburón” no difiere mucho con el de los objetos Medusa y Piraña, tan solo lo hace en su movimiento. En la carga del clip se define una variable *izq*, a la cual le asignamos el valor 1. Mientras esta variable tome este valor significará que el tiburón se moverá hacia la izquierda. En caso de tomar el valor 0, su movimiento será el opuesto. El mecanismo de movimiento del objeto tiburón será el siguiente. Mientras *izq* tome el valor 1 y la posición en el eje x sea superior a 20, el tiburón se moverá sin ningún problema hacia la izquierda. Cuando este llegue a la coordenada 20 del eje x, entonces la variable *izq* tomará el valor 0 y el tiburón se moverá a partir de ese instante hacia la derecha. Esto seguirá así mientras la coordenada en el eje x del tiburón sea inferior a 520. Cuando esta coordenada tome el valor 520 en el eje x, entonces *izq* volverá a tomar el valor de 1 y se moverá de nuevo en la dirección izquierda.

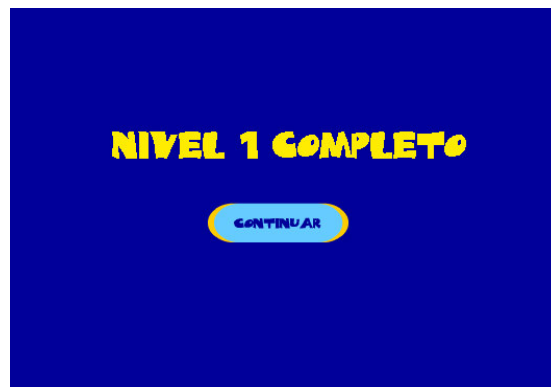
Una vez desarrollado y explicado el comportamiento de todos los elementos activos del juego pasaremos a explicar como funciona el avance de niveles de este.

## Niveles

Cuando al principio de cada partida pinchamos sobre el botón “jugar”, este nos lleva al frame del nivel 1. Si este nivel finalmente se completa saltaremos al frame etiquetado como “Nivel1finish” en el cual se recalcularán los puntos obtenidos por el jugador dependiendo del número de vidas que a este le queden:

```
stop();  
_root.puntos=(_root.puntos)*(_root.vidas);
```

En dicho frame encontraremos también un botón “continuar”. En el momento en que pinchemos sobre el botón este nos mandará al frame del nivel 2.



El código del botón continuar es el siguiente:

```
on (release) {  
    _root.gotoAndStop(20);  
    _root.sonido_water.start();}
```

El frame 20 es el frame del nivel 2. Además se reproducirá el sonido “sonido\_water”.

De la misma manera funcionarán los niveles 2 y 3. Una vez finalizado el nivel 2 el juego saltará al frame “Nivel2finished” donde se volverá a recalcular el número de puntos en función también del número de vidas restantes. Encontraremos también un botón continuar que nos llevará al frame del nivel 3 (frame 25).

Finalmente, finalizado el nivel 3 el juego habrá concluído. Una vez terminado el nivel 3 el juego saltará al frame “Nivel3finished” donde se recalculará una vez más el número de puntos que se ha obtenido en la partida de la misma manera que en los frames “Nivel1finished” y “Nivel2finished”. También encontraremos un botón “Volver a jugar” que nos llevará al frame de inicio del juego, ofreciéndonos así la opción de jugar una nueva partida:

```
on (release) {  
    _root.sonido_water.start();
```

```
} gotoAndPlay(1); // frame del inicio del juego
```

Por otro lado, en cualquier instante del juego en el que el número de vidas del jugador tome el valor de 0, el juego saltará al frame “GameOver”. Este frame indicará que la partida ha finalizado y devolverá el número de puntos que ha obtenido el jugador en la partida. También aparecerá el botón “Volver a jugar” que ofrecerá al jugador la oportunidad de volver a poner a prueba su habilidad con el juego.



## Sonidos

Otro de los aspectos que hay que analizar es la sonorización del juego. Para ello primero de todo se ha exportado a la librería del Macromedia Flash los archivos en formato mp3 que se van a reproducir en el juego. Dichos archivos se enlazarán después con el actionscript mediante la instrucción attachSound. El código lo podemos ver a continuación.

```
sonido_ow = new Sound ();  
sonido_ow.attachSound("ow01");  
  
sonido_pop = new Sound ();  
sonido_pop.attachSound("pop01");  
  
sonido_water = new Sound ();  
sonido_water.attachSound("underwater01");
```

El proceso de definición de sonidos es bastante simple. Analizaremos por ejemplo el primer sonido, sonido\_ow, ya que todos los sonidos se definen de la misma forma.

En la primera línea declaramos la variable sonido\_ow como una variable de tipo sonido, y a continuación en la siguiente línea le añadimos a ese sonido el sonido ow01, el cual se encuentra definido en nuestra librería de objetos.

Pues bien, como se puede ver arriba, tan solo hemos incluido tres sonidos en el juego. Explicaremos brevemente cuando utilizamos cada uno de ellos: En primer lugar nos encontramos con sonido\_ow, el cual es un sonido que va asociado al personaje Bill y que se reproducirá cada vez que este pierda una vida.

Después tenemos el sonido `sonido_pop`, el cual se reproducirá cada vez que Bill atrape una estrella.

Finalmente aparece el sonido `sonido_water`, el cual se reproducirá cada vez que se apriete un botón de acción dentro del juego.

### **Creación del escenario**

El último punto que hay que analizar es la creación del escenario. Básicamente nuestro escenario podríamos decir que está formado por dos elementos, a los que hemos llamado suelo y fondo. El primero de los elementos es el fondo, el cual representa, como su propio nombre indica, el fondo marino, tratándose de una imagen de tonos azules degradados creando así el efecto del agua marina. El segundo elemento, es el suelo, el cual representa el suelo rocoso del fondo del mar. Pues bien, a la hora de realizar el tratamiento de estos dos objetos les hemos proporcionado un comportamiento similar, unicamente variando la velocidad del movimiento de sendos objetos creando así un efecto se conoce como scroll parallax, o lo que es lo mismo una parte del escenario se mueve más rápido que la otra, con lo que se consigue dar un enfoque en el que el usuario pueda distinguir las dimensiones en un escenario con objetos bidimensionales.

Pasaremos a ver el código asociado al objeto suelo:

#### **Código asociado al objeto suelo1:**

```
onClipEvent (load) {
  this._x=280;
  this._y=617;
  function reset(){
    this._x=280;
    this._y=617;
  }
  sueloSpeed=5;
}

onClipEvent (enterFrame) {
  this._x-=sueloSpeed;
  if (this._x<=-280) {
    this.reset();
  }
}
```

#### **Código asociado al objeto suelo2:**

```
onClipEvent (load) {
  this._x=0;
  this._y=617;
  function reset(){
    this._x=280;
    this._y=617;
  }
  sueloSpeed=5;
}
```

```

}

onClipEvent (enterFrame) {
    this._x-=sueloSpeed;
    if (this._x<=-280) {
        this.reset()
    }
}

```

Como se puede apreciar hemos creado dos instancias del objeto suelo. Esto es debido a que ningún objeto es infinito, y por muy largo que lo creemos tarde o temprano se termina, por lo que es necesario superponer cada cierto tiempo un objeto suelo sobre otro, del modo que en el momento en que una instancia del objeto suelo vaya a desaparecer de la pantalla (debido a que ya ha recorrido la pantalla en su totalidad), el otro objeto suelo llamara a su función reset, con lo cual sus coordenadas en pantalla se reiniciarán a su valor por defecto volviendo a aparecer en su posición inicial. De este modo conseguimos crear el efecto de un suelo infinito.

Con el código del objeto fondo hemos realizado algo parecido que con el objeto suelo debido a que su funcionamiento es similar. Simplemente hemos variado la velocidad de movimiento de este objeto haciéndola más lenta que la del suelo para así crear un efecto llamado scroll parallax, el cual ya hemos explicado anteriormente:

#### **Código asociado al objeto fondomar11:**

```

onClipEvent (load) {
    function reset(){
        this._x=0;
        this._y=350;
    }
    sueloSpeed=10;
    this.reset();
}

onClipEvent (enterFrame) {
    this._x-=sueloSpeed;
    if (this._x<=-569) {
        _root.fondomar11.reset()
        _root.fondomar12.reset()}
}

```

#### **Código asociado al objeto fondomar12:**

```

onClipEvent (load) {
    function reset(){
        this._x=569;
        this._y=350;
    }
}

```



```

}
sueloSpeed=10;
this.reset();
}

onClipEvent (enterFrame) {
    if (_root.avance==1){
        this._x-=sueloSpeed;}
        this._x-=sueloSpeed;
        if (this._x<=0) {
            _root.fondomar11.reset()
            _root.fondomar12.reset()
        }
    }
}

```

## Exportación de puntuación a la base de datos

Una vez finalizada una partida resulta necesario exportar la puntuación obtenida a las base de datos que contiene todas las puntuaciones obtenidas por los usuarios registrados que han jugado alguna partida al juego. En primer lugar hay que recordar que todo jugador puede obtener la puntuación que ha obtenido en el juego por medio de dos frames diferentes. El primero de ellos es el frame “Game Over”, al cual se llegará si el jugador no ha sido capaz de superar el juego. El otro es el frame “Nivel3finish”, al cual se llegará si el jugador ha sido capaz de ganar la partida, llegando así al final del juego.

Pues bien, tanto en un frame como en el otro deberemos introducir la siguiente línea de código, la cual será la encargada de pasarle la variable puntos junto con el nick del usuario de la plataforma que está activo en ese momento y el nombre del juego (en nuestro caso juego3) al fichero “juego\_guardar\_datos.php”, el cual se encarga de enviar a la base de datos la puntuación obtenida durante la partida:

```
loadVariablesNum("http://localhost/juegos/juego_guardar_datos.php?nick="+nick+"&puntuacion="+_root.puntos+"&juego=juego3",0);
```

## Precarga

Debido a la lentitud de carga en internet de algunos archivos, hemos decidido incluir una precarga para el juego. La precarga consiste en una barra amarilla que se irá llenando hasta que el juego haya cargado por completo en memoria. El código es el siguiente:

Frame1:

```
loaded = getBytesLoaded();
total = getBytesTotal();
loadpercent = int(100*loaded / total);
if(loadpercent != 100){
    setProperty(barra1, _xscale, loadpercent);
} else {
    gotoAndStop("Loaded");}
```

Frame2:

```
gotoAndPlay(1);
```

Con esto conseguimos que hasta que no haya cargado por completo el juego se mantenga en el frame1 de manera indefinida.



## V. BIBLIOGRAFIA

Bhangal, Sham / Renow-Clarke, Ben, (2003), Action Script para Flash MX. Ediciones Anaya Multimedia.

Curtis, Hillman, (2000), Flash Web Design The Art Of Motion Graphics. New Riders Publishing

Eaton, Eric, (2003), Diseño Web. Elementos de interfaz. Ediciones Anaya Multimedia (Grupo Anaya S.A.)

Frankin, Derek (2003), Macromedia Flash MX. Pearson Education

Franklin, Derek / Makar, Joe (2004), Flash MX ActionScript. Ediciones Anaya Multimedia

Green, Tom, (2004), Studio MX. Creación de sitios web. Anaya Multimedia.

Greenspan, Jay / Bulger, Brad, (2001), MySQL/PHP Database Applications. M&T Books.

Kabir, Mohammed J., (2003), La biblia del servidor Apache. Ediciones Anaya Multimedia-Anaya Interactiva

Makar, Jobe, (2002), Macromedia Flash MX Game Design Demystified – The Official Guide to Creating Games with Flash. Peapchip Press

Medinets, David, (2000), PHP3 Programming Browser-Based Applications. Editorial McGraw-Hill

Musciano, Chuck / Kennedy, Bill, (2002), HTML & XHTML, The definitive guide. O'Reilly & Associates

Paniagua Navarro, Antonio, (2004), Macromedia Flash MX 2004. Ediciones Anaya Multimedia (Grupo Anaya S.A.)

Sin autor (2002), Flash MX Proyectos Profesionales. Ediciones Anaya Multimedia.

Zeldman, Jeffrey, (2003), Diseño con estándares Web. Ediciones Anaya Multimedia (Grupo Anaya S.A.)

## **ANEXO I. FICHAS DE LOS JUEGOS ANALIZADOS DURANTE EL ESTUDIO DE LAS COMUNIDADES VIRTUALES**

**Nombre de la marca:** Nike

**Producto:** Ropa deportiva

**Público objetivo:** jóvenes interesados en los deportes

**Edad:** 10 – 30 años

**Nivel económico:** todos

**Modelo de negocio:**

**B2B (Business To Business):**

**B2C (Business To Consumer):**

**Nombre del proyecto:** Houseball

**URL del proyecto:** <http://www.nikefootball.com>

**Software utilizado:** Flash 7

**Tipo de videojuego:** Pinball

**Tipo de presentación del juego:** integrado en la página web

**Empresa:** Nike

**Síntesis:** Juego consistente en la combinación de elementos de fútbol con el clásico juego de pinball, en el cual los tacos han sido sustituidos por zapatillas de la marca Nike y la pelota por un balón de fútbol.

**Nombre de la marca:** Galletas Oreo

**Producto:** galletas

**Público objetivo:**

**Edad:**

**Nivel económico:** todos

**Modelo de negocio:**

**B2B (Business To Business):**

**B2C (Business To Consumer):**

**Nombre del proyecto:** Mah Jongg

**URL del proyecto:** <http://www.nabisco.com>

**Software utilizado:** Shockwave

**Tipo de videojuego:** Puzzle

**Tipo de presentación del juego:** integrado en la página web

**Empresa:** Nabisco

**Sinopsis:** Sencilla recreación del clásico juego de mesa japonés, donde los símbolos que aparecen sobre las fichas han sido sustituidos por imágenes de galletas y logotipos de la marca, introduciendo de esta manera tan hábil, el componente publicitario dentro del videojuego.

**Nombre de la marca:** Galletas Oreo

**Producto:** galletas

**Público objetivo:**

**Edad:**

**Nivel económico:** todos

**Modelo de negocio:**

**B2B (Business To Business):**

**B2C (Business To Consumer):**

**Nombre del proyecto:** Mini Mini Golf

**URL del proyecto:** <http://www.nabisco.com>

**Software utilizado:** Shockwave

**Tipo de videojuego:** Deportes

**Tipo de presentación del juego:** integrado en la página web

**Empresa:** Nabisco

**Sinopsis:** Atractivo y adictivo juego de mini golf, en el cual jugamos en reducidos escenarios que se pueden encontrar en cualquier hogar (por ejemplo, una mesa de escritorio) y sobre los cuales aparecen cajas y galletas de los productos ofertados por esta marca.

**Nombre de la marca:** Galletas Oreo

**Producto:** galletas

**Público objetivo:**

**Edad:**

**Nivel económico:** todos

**Modelo de negocio:**

**B2B (Business To Business):**

**B2C (Business To Consumer):**

**Nombre del proyecto:** Dunking Game

**URL del proyecto:** <http://www.nabisco.com>

**Software utilizado:** Shockwave

**Tipo de videojuego:** Habilidad

**Tipo de presentación del juego:** integrado en la página web

**Empresa:** Nabisco

**Sinopsis:** Sencillo juego consistente en ir atrapando con un vaso de leche las galletas que caen de la parte superior de la pantalla.

**Nombre de la marca:** Lego

**Producto:** juguetes

**Público objetivo:** niños

**Edad:** 5 – 12 años

**Nivel económico:** todos

**Modelo de negocio:**

**B2B (Business To Business):**

**B2C (Business To Consumer):**

**Nombre del proyecto:** Lego Racer

**URL del proyecto:** <http://www.lego.com>

**Software utilizado:** Flash 7

**Tipo de videojuego:** Juego de carreras

**Tipo de presentación del juego:** integrado en la página web

**Empresa:** Lego

**Sinopsis:** Juego de carreras de coches, en el cual se ven los vehículos desde una perspectiva aérea, consistente en competir contra tres oponentes computerizados dentro de un circuito cerrado.



**Nombre de la marca:** Lego

**Producto:** juguetes

**Público objetivo:** niños

**Edad:** 5 – 12 años

**Nivel económico:** todos

**Modelo de negocio:**

**B2B (Business To Business):**

**B2C (Business To Consumer):**

**Nombre del proyecto:** World Builder

**URL del proyecto:** <http://www.lego.com>

**Software utilizado:** Flash 7

**Tipo de videojuego:** Construcción

**Tipo de presentación del juego:** integrado en la página web

**Empresa:** Lego

**Sinopsis:** Juego consistente en construir figuras con piezas de la marca Lego. No existe ningún objetivo que cumplir pero permite comprobar las posibilidades de las piezas de construcción Lego.

**Nombre de la marca:** Lego

**Producto:** juguetes

**Público objetivo:** niños

**Edad:** 5 – 12 años

**Nivel económico:** todos

**Modelo de negocio:**

**B2B (Business To Business):**

**B2C (Business To Consumer):**

**Nombre del proyecto:** World Builder

**URL del proyecto:** <http://www.lego.com>

**Software utilizado:** Flash 7

**Tipo de videojuego:** Construcción

**Tipo de presentación del juego:** integrado en la página web

**Empresa:** Lego

**Sinopsis:** Juego consistente en construir figuras con piezas de la marca Lego. No existe ningún objetivo que cumplir pero permite comprobar las posibilidades de las piezas de construcción Lego.

**Nombre de la marca:** Nesquick

**Producto:** producto alimenticio para desayunos y meriendas

**Público objetivo:** niños

**Edad:** 5 – 12 años

**Nivel económico:** todos

**Modelo de negocio:**

**B2B (Business To Business):**

**B2C (Business To Consumer):**

**Nombre del proyecto:** Fotoquick de Quicky

**URL del proyecto:** <http://www.nestle.es>

**Software utilizado:** Flash 7

**Tipo de videojuego:** Habilidad

**Tipo de presentación del juego:** integrado en la página web

**Empresa:** Nestle

**Sinopsis:** Juego consistente en ir haciendo fotos con un objetivo a un personaje evitando fotografiar focos y otros elementos no acordes con el escenario.

**Nombre de la marca:** Litoral

**Producto:** fabada asturiana

**Público objetivo:** personas interesadas en el consumo de alimentos tradicionales

**Edad:** 21 – 65 años

**Nivel económico:** todos

**Modelo de negocio:**

**B2B (Business To Business):**

**B2C (Business To Consumer):**

**Nombre del proyecto:** ¡Esta abuela es una lata!

**URL del proyecto:** <http://www.nestle.es>

**Software utilizado:** Flash 7

**Tipo de videojuego:** varios

**Tipo de presentación del juego:** integrado en la página web

**Empresa:** Nestle

**Sinopsis:** El usuario es representado por una lata de fabada que se mueve a través del mapa de Asturias y por donde va encontrando mini-juegos de diversos géneros, como por ejemplo, juegos de preguntas y respuestas.

## **ANEXO II. GUÍA DE INSTALACIÓN**

### **1. Instalación de un servidor web con php**

Para la instalación de la plataforma se debe poseer un equipo que actúe como servidor web. La forma de lograr esto es instalando en este equipo el software adecuado. Existen gran cantidad de programas en el mercado que realizan esta función, algunos de ellos son Windows Server 2003 o Apache. Una vez instalado el programa siguiendo las instrucciones correspondientes, deberá instalarse la extensión php para que el servidor pueda ejecutar scripts en este lenguaje. Esta extensión se puede descargar de manera gratuita de la página web <http://www.php.net> y ser instalada posteriormente siguiendo las instrucciones contenidas en el fichero INSTALL incluida en la distribución.

A la hora de instalar un servidor Apache en un equipo existen formas de realizar esta tarea de forma automatizada, obteniendo ya la extensión php correctamente instalada, a la vez que un módulo mysql activo y correctamente configurado. Para realizar este proceso deberemos descargar e instalar el programa llamado AppServ, que se puede encontrar en <http://www.appservnetwork.com>

### **2. Instalación de MySQL**

Para la instalación de la plataforma necesitaremos una base de datos que contendrá la diferente información requerida por el sistema. El sistema de gestión de bases de datos seleccionado ha sido MySQL, debido a su gran potencia, facilidad de uso, compatibilidad con php y la posibilidad de ser obtenido de manera gratuita a través de su página web <http://www.mysql.com>

En caso de haber instalado AppServ, mencionado anteriormente, no será necesaria la instalación y configuración de un servidor MySQL, debido a que el propio AppServ lo incluye.

### **3. Copia de los ficheros que componen la web de la plataforma en el directorio correspondiente**

Durante la instalación de nuestro servidor, tendremos un directorio que actuará como raíz del servidor web. En el caso de haber utilizado AppServ para la instalación del servidor web y la base de datos, este directorio será por defecto c:\AppServ\www.

Para la instalación de la plataforma en este servidor se deberá descomprimir el fichero "oceanografico.zip" en el directorio correspondiente al servidor instalado, c:\AppServ\www

### **4. Creación de las bases de datos necesarias**

Una vez guardados los ficheros de la plataforma en la ruta de directorios correspondiente, no tendremos acceso a la mayor parte de las funciones de la plataforma debido a la inexistencia de las bases de datos necesarias, por lo que deberemos crearlas. Para la creación de estas bases de datos deberemos ejecutar el script creacion\_bases\_de\_datos.php, que se podrá ejecutar en el equipo servidor a través de un

explorador web introduciendo la dirección siguiente:  
[http://localhost/administrador/creacion\\_bases\\_de\\_datos.php](http://localhost/administrador/creacion_bases_de_datos.php)