Manual de Programación

API del **MerkaWeb®**

ÍNDICE

Autor Koldo G. Castillo
Modificado el 19 de enero de 2007

1. Introducción.

La API es un conjunto de clases, funciones, variables y constantes, destinadas a:

1. Poder acceder a la información de la base de datos del gestor, sin necesidad de conocer su distribución (tablas, campos y relaciones.
2. La información obtenida está debidamente formateada.
3. Funciones principales para la obtención de la Web.
4. Utilidades que faciliten el trabajo al usuario.

Esta compuesto por una clase principal que instancia el resto de clases e incluye los archivos que contienen funciones y constantes globales.

Clases:

MKWConn	api.inc	gestiona la petición de las webs.
MKWIdiomas	apiidiomas.inc	gestión de los idiomas.
MKWFind	apifind.inc	búsqueda de elementos en la base de datos.
MKWUsuario	apiusuario.inc	gestión del registro de usuarios en la web.
MKWMail	apimail.inc	utilidad para enviar correos a varias personas.
MKWTemplate	apitemplate.inc	gestión de las plantillas.

Funciones y constantes:

Funciones	apiformat.inc	funciones de formateo de los datos.
Constantes	apidefines.inc	constantes globales.

2. Clase MKWConn.

2.1. Variables:

Clases instanciadas:

$Idiomas	gestión de idiomas.
$Find	gestión de busquedas.
$Template	gestión de plantillas.
$Usuario	gestión del usuario registrado.

Configuración de la Website:

$name_plantilla	nombre de la Website.
$ctrlseccion	filtro de la Website para buscar información.
$SecDef	sección inicial de la Website.
$grupo_boletin	grupo de boletín de la Website.
$boletin_email	email de envío del boletín de la Website.
$boletin_entidad	nombre del email de envío del boletín de la Website.
$idioma_defecto	idioma por defecto de la Website.
$googlecode	código de la Website para unir a google-analitics.
$id_plantilla	identificador de la Website.
$url_web	dirección URL de la Website.
$con_sinpermiso	contenido que mostrar si no tiene acceso a la Website.
$perfilusuarios;	perfil de permisos por defecto para los usuarios al registrarse.

Variables de carácter general:

$Conn	Conexión directa con la base de datos.
$Idioma	Idioma con el que se está trabajando (abreviatura).
$Migas	Datos que procesar (idioma, parámetros, sección y contenido).

2.2. Funciones Internas:

Son funciones de uso interno, tanto desde la creación o la llamada al API, hasta la carga inicial de datos y su procesamiento.

No es necesario que el usuario que programe las funciones de "*template_preprocess*" y "*template_posprocess*" recurra a estas funciones.

function MKWConn ();	
Entrada	--ninguna--
Salida	Instancia de la clase
Funcionamiento	1. Inicia las variables de la clase. 2. Crea las instancias de las clases que necesita y las asigna a sus variables.

function Conectar ($urlIndex);	
Entrada	$urlIndex puede ser la Url de la web o la Firma.
Salida	true/false en función a si ha podido conectar o no.
Funcionamiento	1. Conectar con la base de datos (si no puede =>false) 2. Cargar los datos de la plantilla en las variables(si no puede => false) 3. Si hay archivo de procesos en la carpeta de la plantilla, carga sus funciones. 4. Iniciar la sesión. 5. Cargar los datos de petición de la web en la variable MIGAS (código

	de sección, código de contenido, tipo de contenido, acción, $_GET y $_POST), y en las clases: Idiomas, Find, Template y Usuario. 6. Si existe la función de pre-proceso (debería estar en el archivo de procesos de la plantilla), la ejecuta. 7. En función al tipo de contenido que se haya pedido, lo busca y lo carga en los datos del contenido. a. Comprobando que si es privado, el usuario registrado tenga acceso, sino, carga el contenido que indica que no tiene permiso (AnularCarga()). b. En caso de que no haya código de contenido que cargar, entonces entiende que quiere el listado completo y lo carga. (si no hay listado posible => false). 8. Comprueba el resultado: a. Si el contenido devuelto no tiene el mismo código que el pedido, algún error ha habido y lo vacía. b. Si no había ninguna sección, pero el contenido tenia una predefinida, carga la sección del contenido. c. Si la Web que pide la sección no tiene acceso a ella, la vacía. d. Si no hay contenido pero la sección si tiene alguno asociado, carga en el contenido el primer contenido de la sección. e. Si hay un contenido cargado y tiene Meta-Tags propios, hay que cambiar los de la sección por los suyos. f. Si hay un contenido cargado y tiene menús, los agrega a los de la sección. 9. Si existe la función de post-proceso (debería estar en el archivo de procesos de la plantilla), la ejecuta. 10. Devolverá true.

function Desconectar ();	
Entrada	--ninguna--
Salida	--ninguna--
Funcionamiento	1. Realiza el registro estadístico de la Web. 2. Elimina la conexión con la base de datos.

function CargarMigas ();	
Entrada	--ninguna--
Salida	--ninguna-- carga los datos en la variable “Migas”.
Funcionamiento	1. Recoge los datos de la petición de la Web. 2. Introduce $_GET en su campo dentro de las “Migas”. 3. Introduce $_POST en su campo dentro de las “Migas”. 4. Si no hay idioma especificado, coge el idioma por defecto de la Web. 5. Si el idioma no es el mismo que en la petición anterior, y no hay más datos en la petición de la Web, carga los datos de la copia anterior para poder enseñarlos en el nuevo idioma seleccionado. 6. Sino, carga el código de la sección y del contenido en sus campos dentro de las “Migas”.

function AnularCarga();	
Entrada	--ninguna--
Salida	Los datos del contenido definido para notificar que no se tiene permiso.
Funcionamiento	1. Cargar los datos del contenido. 2. Anula la carga de datos que se haya hecho en las “Migas”: a. Tipo de contenido = CONTENIDO. b. Accion = “”. c. Sección = La sección predeterminada del contenido.

2.3. Funciones para el Usuario:

Son funciones preparadas para facilitar la programación de las funciones de procesado de las plantillas.

Además de estas funciones, existen las del resto de clases instanciadas desde el principio en la API. Mencionadas en el apartado previo de variables.

function GetData...();Idioma, Seccion, Contenido, Tipo, Accion, Orden, Get, Post	
Entrada	--ninguna--
Salida	El dato que quieren consultar de las "Migas".
Funcionamiento	El proceso inverso que la función siguiente. 1. Coge de la variable "Migas", el dato que quiera consultar. a. Las primeras son las de la petición de la Web. b. Get y Post son las de $_GET y $_POST respectivamente.

function SetData...($dato);Idioma, Seccion, Contenido, Tipo, Accion, Orden, Get, Post	
Entrada	$dato es el dato que introducir en la variable "Migas".
Salida	--ninguna--
Funcionamiento	El proceso inverso que la función anterior. 1. Introduce en la variable "Migas", el dato que quiera cambiar.

function GetIdioma ();	
Entrada	--ninguna--
Salida	Idioma en el que se está viendo la Web. Formato: nombre y abreviatura.
Funcionamiento	--

function GetIdiomas ();	
Entrada	--ninguna--
Salida	Listado de idiomas disponibles para ver la Web. Formato: (número, 1=>(nombre y abreviatura), 2=>…, …)
Funcionamiento	--

function SetIdioma ($Patron);	
Entrada	$Patron es el nombre o abreviatura del idioma que queremos cargar.
Salida	true/false si lo ha encontrado o no.
Funcionamiento	--

function Traducir ($Palabra);	
Entrada	$Palabra es el termino a buscar en el diccionario.
Salida	El equivalente del término en el idioma en que se está viendo la Web.
Funcionamiento	--

function Migas();	
Entrada	--
Salida	La variable de las "Migas".
Funcionamiento	--

function GetDoc ($Patron);	
Entrada	$Patron es el código del documento o nombre del mismo.
Salida	Documento debidamente formateado.
Funcionamiento	Manda la petición a la clase de búsqueda "Find".

function GetAlbum ($Patron);	
Entrada	$Patron es el código del album o nombre del mismo.
Salida	Album debidamente formateado.
Funcionamiento	Manda la petición a la clase de búsqueda "Find".

function GetMenu ($Patron, $Hijos=true);	
Entrada	$Patron es el código del menú o nombre del mismo. $Hijos indica si queremos que cargue también los enlaces, o no.
Salida	Menú debidamente formateado.
Funcionamiento	Manda la petición a la clase de búsqueda "Find".

function GetContenido ($Patron, $Hijos=true);	
Entrada	$Patron es el código del contenido o nombre del mismo. $Hijos indica si queremos que cargue también todos los datos relacionados con el contenido (otros contenidos, documentos y menús).
Salida	Contenido debidamente formateado.
Funcionamiento	Manda la petición a la clase de búsqueda "Find".

function GetContenidosCriterio($Patron, $Orden, $Pagina, $RPP);	
Entrada	$Patron es el texto por el que queremos buscar los contenidos. Usado para generar la consulta que lo compara con el nombre y el resto de campos por idioma. $Orden es el campo por el que queremos ordenar el resultado. $Pagina es la página que queremos mostrar. $RPP son los registros por página que queremos mostrar.
Salida	Un listado de los contenidos encontrados y debidamente formateados. Formato: (número, 1=>contenido(sin contenidos relacionados), 2=>…, …)
Funcionamiento	1. Construye el filtro de búsqueda para aplicar el patrón al nombre y al título, resumen y texto por idioma. 2. Manda la petición a la clase de búsqueda "Find".

function GetSeccion ($Patron, $Hijos=true);	
Entrada	$Patron es el código de la sección o nombre del mismo. $Hijos indica si queremos que cargue también todos los datos relacionados con la sección (menús, contenidos y metas).
Salida	Sección debidamente formateada.
Funcionamiento	1. Manda la petición a la clase de búsqueda "Find". 2. Le añade el listado de Canales RSS que tenga la Web.

function GetBoletin ($Patron, $Hijos=true, $Publico=-1);	
Entrada	$Patron es el código del boletín o el nombre del mismo. $Hijos indica si queremos que cargue también todos los datos relacionados con el boletín (menús y contenidos). $Publico indica si queremos buscar entre todos los boletines (-1), solo los públicos (1) o solo los privados (0).
Salida	Boletín debidamente formateado.
Funcionamiento	Manda la petición a la clase de búsqueda "Find".

function GetBoletinesCriterio ($Patron, $Orden, $Publico=-1);	
Entrada	$Patron es texto por el que queremos buscar los boletines. Usado para generar la consulta al agregarlo como sentencia "WHERE". $Orden es el campo por el que queremos ordenarlo. $Publico indica si queremos buscar entre todos los boletines (-1), solo los públicos (1) o solo los privados (0).

Salida	Listado de los boletines encontrados y debidamente formateados. Formato: (número, 1=>boletín (sin contenidos relacionados), 2=>…, …)
Funcionamiento	Manda la petición a la clase de búsqueda "Find".

function RegistrarBoletin ($email, $nombre, $idioma="", $grupo="") ;	
Entrada	$email y $nombre son los datos que quiere el visitante registrar en el boletín de la Web. $idioma, si no se indica lo contrario, cogerá en el que se está viendo la Web. $grupo, además del definido en la Web, puede apuntarse a alguno más.
Salida	true/false si ha conseguido registrarlo, o no por que no ha pasado suficientes datos, por que ya estaba registrado o se ha producido un error al realizar el registro en la base de datos
Funcionamiento	Manda la petición a la clase de usuario "usuario".

function SetUsuario($email, $clave);	
Entrada	$email y $clave son los datos para identificar al usuario.
Salida	true/false si es un usuario registrado y puede identificarse en la Web.
Funcionamiento	Manda la petición a la clase de usuarios "Usuario".

function GetUsuario();	
Entrada	--ninguna--
Salida	Datos del usuario debidamente formateados.
Funcionamiento	Manda la petición a la clase de usuarios "Usuario".

function ActualizarUsuario ($datos) ;	
Entrada	$datos es un vector con los datos del usuario formateado como el resultado de la función "FormatUsuario".
Salida	true/false si ha conseguido actualizarlo, o no por que no hay usuario identificado o se ha producido un error al realizar el registro en la base de datos
Funcionamiento	Manda la petición a la clase de usuario "usuario".

function RegistrarUsuario ($datos) ;	
Entrada	$datos es un vector con los datos del usuario formateado como el resultado de la función "FormatUsuario".
Salida	true/false si ha conseguido actualizarlo, o no por que ya hay un usuario registrado con esa identificación, o se ha producido un error al realizar el registro en la base de datos
Funcionamiento	Manda la petición a la clase de usuario "usuario".

function GetCanal ($Patron, $Hijos=true);	
Entrada	$Patron es el código del canal o el nombre del mismo. $Hijos indica si queremos que cargue todos los datos relacionados con el canal (contenidos del canal).
Salida	Canal debidamente formateado.
Funcionamiento	Manda la petición a la clase de búsqueda "Find".

function GetCanales();	
Entrada	--ninguna--
Salida	Listado de los canales encontrados y debidamente formateados. Formato: (número, 1=>canal (sin contenidos relacionados), 2=>…, …)
Funcionamiento	Manda la petición a la clase de búsqueda "Find".

function RegistroEstadistico($campos=array());	
Entrada	$campos que se quieran añadir al registro estadístico de la Web. Formato: Array ("campo1" => Array("campo"=>"estIdioma","dato"=>$this->Idiomas->Idioma[CAMPO_NOMBRE],"type"=>"char") "campo2" => …)
Salida	--ninguno--
Funcionamiento	1. Fuerza el envío de código que haya pendiente para el navegador. 2. Prepara la sentencia SQL para registrar todos los datos de la página que se está mostrando. 3. Ejecuta la sentencia en la base de datos.

function GoogleAnalitics();	
Entrada	--ninguna-- (usa el código para google analictics definido en la Website)
Salida	--ninguna-- (hace una salida por pantalla)
Funcionamiento	1. Fuerza el envío de código que haya pendiente para el navegador. 2. Envía a pantalla el código para unir la página Web a Google-Analitics.

function EnviarEmail ($from, $fromname, $to, $toname, $contenido, $datos);	
Entrada	$from y $fromname son los datos del que quiere enviar el email. $to y $toname son los datos del que va ha recibir el email. $contenido es un contenido del que se coge el texto como plantilla a usar. $datos son los datos que emplear con la plantilla.
Salida	true/false si ha conseguido enviar el email, o no.
Funcionamiento	1. Abre la plantilla pasándole el texto del contenido, e indicando que es un email, y pasándole los datos recibidos. 2. Procesa la plantilla y obtiene el resultado. 3. Cambia las direcciones de los archivos para que tengan el camino completo. 4. Instancia la clase MKWEmail con los datos que tenemos. 5. Envía el correo. 6. Devuelve si ha podido enviarlo.

function Template($conSeccion=true);	
Entrada	$conSeccion indica que evaluamos la plantilla incluyendo la sección, o solo el contenido.
Salida	Resultado de la plantilla (código HTML).
Funcionamiento	1. Si es un documento o elemento multimedia el que tiene que cargar: a. Genera los encabezados adecuados (HEADER). b. Si está en el disco carga su contenido en una variable. c. Sino, lo coge de la base de datos. d. Devuelve como resultado el contenido del documento. 2. Sino, a. Abre la plantilla con los datos de la API (sección, contenido y tipo). b. Procesa la plantilla y coge el resultado. c. Cambia las direcciones de los archivos para que tengan el camino completo. 3. Devuelve el resultado.

function HtmlBoletin($boletin);	
Entrada	$boletin puede ser el boltín ya formateado, o el código o nombre para buscar sus datos (con contenidos relacionados).
Salida	Resultado de la plantilla (código HTML).
Funcionamiento	1. Carga los datos del boletín si no estaban ya cargados.

	2. Los datos del boletín son el contenido, y los datos de la sección se buscan con el código de la sección para boletines definida en la Website. 3. Abre la plantilla con estos datos y los procesa. 4. Cambia las direcciones de los archivos para que tengan el camino completo. 5. Devuelve el resultado.

function ProcessCode($datos, $code, $archivo="");	
Entrada	$datos que se quieren pasar a la plantilla para procesarla. $code es el código que se quiere usar como plantilla. $archivo es el archivo que contiene la plantilla que se quiere usar como plantilla.
Salida	Resultado de la plantilla (código HTML).
Funcionamiento	1. Instancia una variable de la clase "*MKWTemaplate*". 2. Carga los datos de la Website y parámetros generales. 3. Abre la plantilla pasándole como parámetros: a. Datos de la sección nula (array()). b. Contenido de tipo CONTENIDO_CONTENIDO_CONTENIDO. c. Datos ($datos) del contenido. d. Sin HTML para la sección. e. Con código Html ($code) que procesar para el contenido. f. Sin nombre de plantilla específica para la sección. g. Con el nombre de la plantilla ($archivo) para el contenido proporcionado por parámetro. 4. Procesa la plantilla (sin sección) y coge el resultado. 5. Libera la variable instanciada. 6. Devuelve el resultado.

3. Clase MKWIdiomas.

Es una clase de uso interno para la clase "*MKWConn*" que la instancia y que ya proporciona funciones de acceso para el Usuario.

No es necesario que el usuario que programe las funciones de "*template_preprocess*" y "*template_posprocess*" recurra a esta clase.

3.1. Variables:

$Idiomas	listado de los idiomas.
$Idioma	idioma en el que se está viendo la Web.
$Conn	conexión directa con la base de datos.

3.2. Funcione:

function MKWIdiomas ();	
Entrada	--ninguna--
Salida	Instancia de la clase
Funcionamiento	1. Inicializa las variables a sus valores por defecto (es-español)

function Cargar ($Conn);	
Entrada	$Conn es el enlace a la base de datos
Salida	true/false si ha conseguido cargar los idiomas, o no.
Funcionamiento	1. Busca en la base de datos los idiomas. 2. Los carga en su variable local "Idiomas" con el formato: Idiomas[abreviatura] = nombre_idioma

function GetIdioma ();	
Entrada	--ninguna--
Salida	Idioma en curso debidamente formateado (abreviatura, nombre)
Funcionamiento	--

function GetIdiomas ();	
Entrada	--ninguna--
Salida	Listado de idiomas "Idiomas".
Funcionamiento	--

function SetIdioma($Patron);	
Entrada	$Patron es el idioma por nombre o por abreviatura.
Salida	true/false si ha encontrado el idioma, o no.
Funcionamiento	1. Busca el idioma en su lista. 2. Si lo encuentra lo carga en la variable "Idioma".

function GetTraduccion($Palabra, $Idioma="");	
Entrada	$Palabra es el término que hay que buscar en la tabla del diccionario. $Idioma es para poder especificar un idioma distinto del que se está usando en la Web.
Salida	El término en el idioma seleccionado, o una cadena vacía si no se ha encontrado.
Funcionamiento	--

4. Clase MKWFind.

Es una clase de uso interno para la clase "*MKWConn*" que la instancia y que ya proporciona funciones de acceso para el Usuario.

No es necesario que el usuario que programe las funciones de "*template_preprocess*" y "*template_posprocess*" recurra a esta clase.

4.1. Variables:

$Idioma	idioma en el que se está viendo la Web.
$Conn	conexión directa con la base de datos.

4.2. Funciones:

function MKWFind ();	
Entrada	--ninguna--
Salida	Instancia de la clase
Funcionamiento	1. Inicializa las variables a cadena vacía.

function Cargar ($Conn, $Idioma);	
Entrada	$Conn es el enlace con la base de datos. $Idioma es el idioma en el que se está mostrando la Web.
Salida	--ninguna--
Funcionamiento	1. Asigna los valores de los parámetros a sus variables.

function FindDoc($Patron);	
Entrada	$Patron puede ser el código o el nombre del documento.
Salida	Datos del documento debidamente formateado.
Funcionamiento	1. Busca el registro y pide a la función "FormatDoc" que lo formatee.

function FindAlbum ($Patron);	
Entrada	$Patron puede ser el código o el nombre del álbum.
Salida	Listado de documentos debidamente formateados. Formato: (número, 1=>(documento), 2=>…, …)
Funcionamiento	1. Busca los documentos asociados al álbum. 2. Los introduce en el listado que será devuelto.

function EscogerBanner($enlaces);	
Entrada	$enlace es un listado de enlaces, entre los que habrá que escoger un enlace de forma aleatoria pero teniendo en cuenta sus pesos.
Salida	Enlace escogido.
Funcionamiento	

function FindMenu($Patron, $Hijos=true);	
Entrada	$Patron es el código o el nombre del menú. $Hijos indica si queremos que cargue todos los datos relacionados con el menú (submenús).
Salida	Menú debidamente formateado.
Funcionamiento	1. Busca el registro y pide a la función "FormatMenu" que lo formatee. 2. Si tiene que cargar también los datos relacionados: a. Llama a "FindSubMenu" indicando que él es el patrón y que el

	padre es 0, para obtener los enlaces. b. Pero si es un menú de banners, llama a "EscogerBanner" para que seleccione 1 de entre sus enlaces. c. El o Los enlaces, se agregan al menú en el campo "CAMPO_ENLACES".

function FindSubMenu ($Patron, $Padre);	
Entrada	$Patron es el código o nombre del menú. $Padre es el código del submenú del que cuelga en la estructura de árbol. 0 para los que cuelgan directamente del menú.
Salida	Listado de enlaces debidamente formateados: Formato: (número, 1=>(datos del enlace+CAMPO_ENLACES=>(número, …)), 2=>…)
Funcionamiento	1. Busca los enlaces según el patrón para el menú y el padre para ver el nivel desde el que hay que buscar. 2. Los introduce en una lista cada enlace debidamente formateado por "FormatSubMenu". 3. Comprueba si a su vez puede tener enlaces que penden de él y los mete en el campo CAMPO_ENLACES.

function FindContenido… ($Patron); Contenidos, Documentos y Menus	
Entrada	$Patron es el código del contenido del que hay que buscar los contenidos, documentos o menús relacionados.
Salida	Listado de contenidos, documentos o menús, debidamente formateados.
Funcionamiento	1. Busca los elementos relacionados con el contenido. 2. Los introduce en una lista. 3. Devuelve la lista.

function FindContenido($Patron, $Hijos=true);	
Entrada	$Patron es el código o el nombre del contenido. $Hijos indica si queremos que cargue todos los datos relacionados con el contenido (contenidos, documentos y menús).
Salida	Contenido debidamente formateado.
Funcionamiento	1. Busca los datos del contenido y los formatea con "FormatContenido". 2. En caso de que tenga que cargar los datos relacionados, llamará a las funciones anteriormente descritas y agregará el resultado en los campos: a. CAMPO_CONTENIDOS. b. CAMPO_DOCS. c. CAMPO_MENUS.

function FindContenidosCriterio($Patron, $Orden, $Pagina, $RPP);	
Entrada	$Patron es la sentencia para agregar al "Where". $Orden es el campo por el que queremos ordenar el resultado. $Pagina es la página que queremos mostrar. $RPP son los registros por página que queremos mostrar.
Salida	Un listado de los contenidos encontrados y debidamente formateados. Formato: (número, 1=>contenido(sin contenidos relacionados), 2=>…, …)
Funcionamiento	1. Realiza la consulta según el patrón y genera un listado con los resultados. 2. Además guarda con el listado otros valores en los campos: a. CAMPO_TOTAL, con el número de registros de la consulta. b. CAMPO_PAGINAS, número de páginas del resultado. c. CAMPO_PAGINA, página que se está devolviendo.

function FindSeccion... ($Patron); Contenidos y Menus	
Entrada	$Patron es el código de la sección del que hay que buscar los contenidos o menús relacionados.
Salida	Listado de contenidos o menús, debidamente formateados.
Funcionamiento	1. Busca los elementos relacionados con el contenido. 2. Los introduce en una lista. 3. Devuelve la lista.

function FindSeccion($Patron, $Hijos=true);	
Entrada	$Patron es el código o el nombre del menú. $Hijos indica si queremos que cargue todos los datos relacionados con la sección (contenidos, menús y meta-tags).
Salida	Sección debidamente formateada.
Funcionamiento	1. Busca los datos del contenido y los formatea con "FormatSeccion". 2. En caso de que tenga que cargar los datos relacionados, llamará a las funciones anteriormente descritas y agregará el resultado en los campos: a. CAMPO_CONTENIDOS. b. CAMPO_MENUS. c. CAMPO_METAS.

function FindBoletin... ($Patron); Contenidos y Menus	
Entrada	$Patron es el código del boletín del que hay que buscar los contenidos o menús relacionados.
Salida	Listado de contenidos o menús, debidamente formateados.
Funcionamiento	1. Busca los elementos relacionados con el contenido. 2. Los introduce en una lista. 3. Devuelve la lista.

function FindBoletin($Patron, $Hijos=true, $Publico=-1);	
Entrada	$Patron es el código o el nombre del boletín. $Hijos indica si queremos que cargue todos los datos relacionados con el contenido (contenidos y menús). $Publico indica si queremos limitarnos a los públicos (1), a los privados (0), o a todos (-1).
Salida	Boletín debidamente formateado.
Funcionamiento	1. Busca los datos del contenido y los formatea con "FormatBoletin". 2. En caso de que tenga que cargar los datos relacionados, llamará a las funciones anteriormente descritas y agregará el resultado en los campos: a. CAMPO_CONTENIDOS. b. CAMPO_MENUS.

function FindBoletinesCriterio($Patron, $Orden, $Publico=-1	
Entrada	$Patron es la sentencia para agregar al "Where". $Orden es el campo por el que queremos ordenar el resultado. $Publico indica si queremos limitarnos a los públicos (1), a los privados (0), o a todos (-1).
Salida	Un listado de los boletines encontrados y debidamente formateados. Formato: (número, 1=>boletín(sin contenidos relacionados), 2=>…, …)
Funcionamiento	1. Realiza la consulta según el patrón y genera un listado.

function FindMetas($Patron);	
Entrada	$Patron es el código del grupo de meta-tags que se quiere buscar.
Salida	Un listado de los met-tags encontrados y debidamente formateados.

	Formato: (número, 1=>meta-tag(debidamente formateado), 2=>…, …)
Funcionamiento	1. Busca los datos del los meta-tags del grupo y los formatea con "FormatMeta".

function FindCanalContenidos ($Patron);	
Entrada	$Patron es el código del canal del que se quieren buscar contenidos relacionados a él.
Salida	Un listado de los contenidos encontrados y debidamente formateados. Formato: (número, 1=>contenido(sin contenidos relacionados), 2=>…, …)
Funcionamiento	1. Busca los datos del los contenidos asociados al canal y los formatea con "FormatContenido" y sin contenidos relacionados.

function FindCanal($Patron, $Hijos=false);	
Entrada	$Patron es el código o el nombre del canal. $Hijos indica si queremos que cargue todos los datos relacionados con el contenido (contenidos).
Salida	Canal debidamente formateado
Funcionamiento	1. Busca los datos del canal y los formatea con "FormatCanal". 2. En caso de que tenga que cargar los datos relacionados, llamará a la funciones anteriormente descrita y agregará el resultado en el campo: a. CAMPO_ENLACES.

function FindCanales ($filtro);	
Entrada	$filtro es el patrón por el que buscar los canales por nombre que empieze por él.
Salida	Un listado de los canales encontrados y debidamente formateados. Formato: (número, 1=>canal(sin contenidos relacionados), 2=>…, …)
Funcionamiento	1. Busca los datos del los canales cuyo nombre empieze por el filtro, y acoge los datos que le de la función anteriormente descrita, pero sin contenidos relacionados.

5. Clase MKWUsuario.

Es una clase de uso interno para la clase "*MKWConn*" que la instancia y que ya proporciona funciones de acceso para el Usuario.

No es necesario que el usuario que programe las funciones de "*template_preprocess*" y "*template_posprocess*" recurra a esta clase.

3.1. Variables:

$Conn	conexión directa con la base de datos.
$ctrlseccion	firma de la Website que guardar con el usuario al registrarlo.
$perfil	perfil de acceso configurado en la Website para los usuarios.
$Usuario	datos debidamente formateados del usuario identificado.

3.2. Funciones:

function MKWUsuario ();	
Entrada	--ninguna--
Salida	Instancia de la clase
Funcionamiento	1. Inicia las variables a sus valores nulos.

function Cargar ($Conn, $ctrlseccion, $perfil);	
Entrada	$Conn es el enlace con la base de datos. $ctrlseccion es la firma del Website a registrar en los usuarios. $perfil es el código del perfil que asignar a los usuarios registrados.
Salida	--ninguna--
Funcionamiento	1. Asigna los parámetros a sus variables. 2. Llama a "SetCliente", pasandole los datos guardados en la sesión. Por si se ha identificado ya un cliente.

function SetCliente ($email="", $pass="") ;	
Entrada	$email y $pass son los datos de los usuarios para identificarse.
Salida	true/false si ha encontrado el usuario y se ha identificado, o no.
Funcionamiento	1. Si no hay datos suficientes, devuelve false. 2. Si no encuentra en la base de datos el usuario, devuelve false. 3. Sino. a. Guarda los datos del usuario formateados por "FormatUsuario" en su variable. b. Guarda sus datos en la sesión para que en la siguiente carga pueda volver a identificarse automáticamente. c. Carga sus permisos sobre los contenidos y documentos privados.

function ActualizarUsuario ($datos);	
Entrada	$datos son los datos de un usuario, formateados igual que por "FormatUsuario".
Salida	true/false si ha podido guardar sus datos, o no.
Funcionamiento	1. Genera la sentencia SQL para actualizar los datos del usuario cargado. 2. Ejecuta la sentencia SQL. 3. Llama a "SetCliente", para ver si ha guardado el registro y puede identificarse.

function RegistrarUsuario ($datos);	
Entrada	$datos son los datos de un usuario, formateados igual que por "FormatUsuario".
Salida	true/false si ha podido guardar sus datos, o no.
Funcionamiento	1. Genera la sentencia SQL para insertar un nuevo usuario con los datos. 2. Ejecuta la sentencia SQL. 3. Llama a "SetCliente", para ver si ha guardado el registro y puede identificarse.

function Permiso ($Tipo, $Id);	
Entrada	$Tipo es el tipo de contenido privado que quiere comprobar (contenido o documento). $Id es el código de dicho contenido.
Salida	true/false si tiene permiso, o no.
Funcionamiento	1. Comprueba si tiene permiso en los accesos que tiene por perfil. 2. Si no los tiene, lo comprueba en los que tiene personalizados. 3. Devuelve si lo ha encontrado o no.

function RegistrarBoletin ($email, $nombre, $idioma, $grupo, $ctrlseccion);	
Entrada	$email del usuario a registrar. $nombre con el que quiere identificarse. $idioma en el que quiere recibir el boletín. $grupo de usuarios al que apuntarle. $ctrlsección es la website a la que hay que asociarlo.
Salida	true/false si ha podido guardar sus datos, o no por que ya existe o por que se ha producido un error al intentar registrarlo en la base de datos.
Funcionamiento	1. Busca ya un usuario con ese email en esa Web. 2. Si existe sale devolviendo false. 3. Si no se define un idioma, coge el de la Web. 4. Registra el usuario en la Web. 5. Si hay un grupo especificado, lo asocia a ese grupo de usuarios de boletín.

6. Clase MKWMail.

Es una clase de uso interno para la clase "*MKWConn*" que la instancia y que ya proporciona funciones de acceso para el Usuario.

Aún así, si el usuario que programe las funciones de "*template_preprocess*" y "*template_posprocess*" quiere preparar una aplicación de envío de emails, puede instancias en una variable local a su función esta clase, y emplear sus funciones como se describen en este apartado.

6.1. Variables:

$from	email del que envía.
$fromname	nombre o identificación del que envía.
$tolst	listado de destinatarios del email.
$contenido	contenido que enviar por el email.

6.2. Funciones:

function MKWMail ($from="",$fromname="",$to="",$toname="",$contenido=array());	
Entrada	Los datos de las variables de la clase
Salida	Instancia de la clase.
Funcionamiento	1. Guarda los parámetros en las variables. 2. Agrega $to y $toname a la lista de destinatarios, si es que se han introducido como parámetros.

function Agregar($to, $toname);	
Entrada	$to y $toname son el email y el nombre del destinatario que agregar a la lista.
Salida	--ninguna--
Funcionamiento	--

function Enviar ($to="", $toname="", $contenido=array());	
Entrada	$to y $toname son el email y el nombre del destinatario que agregar a la lista. $contenido es el contenido que enviar por el email.
Salida	true/false si ha podido enviar el email a todos, o no.
Funcionamiento	1. Por cada uno de los destinatarios del email, lo envia y comprueba el error. 2. Devuelve si ha podido enviar todos.

7. Clase MKWTemplate.

Es una clase de uso interno para la clase "*MKWConn*" que la instancia y que ya proporciona funciones de acceso para el Usuario.

No es necesario que el usuario que programe las funciones de "*template_preprocess*" y "*template_posprocess*" recurra a esta clase. Pero puede consultarla para entender mejor el funcionamiento de la programación de las plantillas y su resultado en la Web.

7.1. Variables:

Variables configuradas por parámetros a través de alguna función:

$MKW	enlace a la clase principal de la API (MKWConn).
$idiomas	clase con la gestión de idiomas (MKWIdiomas).
$idioma	abreviatura del idioma en el que se quiere mostrar la Web.
$ctrlseccion	firma de la Website.
$path	dirección donde se encuentran las plantillas de la Website.
$datos	datos iniciales para la plantilla (sección + contenido).
$ctipo	tipo de contenido.
$htmlIniSeccion	html para la sección, en lugar de que coja de una plantilla.
$htmlIniContenido	html para el contenido, en lugar de que coja de una plantilla.
$temSecc	plantilla para la sección, en lugar de que coja de la lista.
$temCon	plantilla para el contenido, en lugar de que coja de la lista.

Variables usadas a nivel interno:

$is_listado	comprobación de si el contenido es un listado de contenidos.
$plantillas	nombres de las plantillas para los contenidos o listados.
$s_estilo	estilo de la sección.
$c_estilo	estilo del contenido.
$estilo	estilo de lo que se está procesando (sección o contenido).
$archivo	archivo que se ha cargado para ser procesado.
$strResultado	variable donde se guarda el resultado del proceso.
$indBloqueId	en un bloque a repetir por elemento, es el número que lleva.
$indBloqueLim	en un bloque a repetir por elemento, es el límite.
$variables	es el vector donde se guardan las variables globales.

7.2. Funciones:

function MKWTemplate ($MKW);	
Entrada	$MKW es el enlace a la clase primaria de la API "MKWConn".
Salida	Instancia de la clase.
Funcionamiento	1. Inicia las variables a sus valores nulos. 2. Menos la del parámetro, que la asigna a su variable local.

function Datos ($path, $ctrlseccion, $idiomas);	
Entrada	$path es la dirección configurada en el Website, donde buscar las plantillas. $ctrlseccion es la firma del Website. $idiomas es la clase de gestión de idiomas "MKWIdiomas".
Salida	-ninguna--
Funcionamiento	1. Carga los parámetros en las variables locales.

	2. Coge de ellas el idioma en el que se está mostrando la Web.

function Open ($seccion, $ctipo, $contenido, $htmlIniSeccion="", $htmlIniContenido="", $temSecc="", $temCon="");	
Entrada	$sección y $contenido son los datos que usar en la plantilla $ctipo indica el tipo de contenido que es. $htmlIniSeccion y $htmlIniContenido son el código html para la sección y el contenido si no queremos que lo cargue desde un archivo. $temSecc y $temCon son el nombre de las plantillas que queremos que use en lugar del correspondiente de la lista de plantillas.
Salida	--ninguna-- (inicializa variables de la clase)
Funcionamiento	1. Carga los estilo de la sección y del contenido. 2. Carga los datos de la sección en la variable de datos. $datos = $seccion; $datos[CAMPO_CONTENIDO_HTML] = (se guardaré el resultado del procesado del contenido); 3. Si el contenido es un listado: a. Se cargan las plantillas para listados. b. Y el contenido como un conjunto de enlaces de la sección: $datos[CAMPO_CONTENIDO][CAMPO_ENLACES] = $contenido; c. E indica en la variable local que es un listado. 4. En caso de que no lo sea: a. Se cargan las plantillas para contenidos. b. Y el contenido como un contenido normarl: $datos[CAMPO_CONTENIDO] = $contenido; 5. Carga el resto de parámetros en sus variables. 6. Si el tipo de contenido era una acción o no está definido, lo redefine a tipo CONTENIDO.

function Process($conSeccion);	
Entrada	$conSeccion indica si queremos que procese también la sección o solo el contenido.
Salida	true/false si ha podido realizar el procesado, o no.
Funcionamiento	1. Si no hay un código que procesar para la sección ($htmlIniSeccion), y no hay plantilla definida ($temSec), escoge una de la lista. 2. Igualmente hace con el contenido. 3. Si es un boletín a. Procesa el contenido de los datos como un boletín, y los guarda en la variable de resultado ($strResultado). 4. Sino a. Si hay contenido, lo procesa y el resultado lo guarda para que pueda ser usado en la sección en el campo: $datos[CAMPO_CONTENIDO_HTML] = $contenido; b. Si no ha habido ningún error i. Si hay que procesar la sección, la procesa y guarda en la variable de resultado ($strResultado) el resultado. ii. Sino, guarda en la variable de resultado el resultado del contenido. 5. Devuelve si ha no habido ningún error.

function HayError();	
Entrada	--
Salida	true/false si se ha dado un error, o no.
Funcionamiento	--

function ErrorMsg();	
Entrada	--ninguna--
Salida	Texto registrado del error
Funcionamiento	--

function ErrorTxt();	
Entrada	--ninguna--
Salida	Texto ya formateado del error: Formato: [MKWTemplate] [f: archivo] [msg: texto]
Funcionamiento	--

function ErrorFile();	
Entrada	--ninguna--
Salida	El archivo que se está examinando cuando se ha producido el error.
Funcionamiento	--

8. Referencias.

8.1. Clase MKWConn:

```
var
        $Conn, $Idiomas, $Idioma,
        $Migas, $Find, $Template,
        $Usuario,
        $name_plantilla, $ctrlseccion, $SecDef,
        $grupo_boletin, $sec_boletin, $boletin_email, $boletin_entidad, $idioma_defecto,
        $googlecode, $id_plantilla, $url_web, $con_sinpermiso, $perfilusuarios;

function MKWConn ();

function Conectar ($urlIndex) ;

function Desconectar();

function CargarMigas();

function AnularCarga();

function GetDataIdioma() ;
function GetDataAccion();
function GetDataSeccion();
function GetDataContenido();
function GetDataTipo();
function GetDataOrden();
function GetDataGet();
function GetDataPost();

function SetDataIdioma($dato);
function SetDataAccion($dato);
function SetDataSeccion($dato);
function SetDataContenido($dato);
function SetDataTipo($dato);
function SetDataOrden($dato);
function SetDataGet($dato);
function SetDataPost($dato);

function GetIdioma ();
function GetIdiomas ();
function SetIdioma ($Patron);
function Traducir ($Palabra);

function Migas() ;

function GetDoc ($Patron);
function GetAlbum ($Patron);
function GetMenu ($Patron, $Hijos=true);
function GetContenido ($Patron, $Hijos=true);
function GetContenidosCriterio($Patron, $Orden, $Pagina, $RPP);
function GetSeccion ($Patron, $Hijos=true);
function GetBoletin ($Patron, $Hijos=true, $Publico=-1);
function GetBoletinesCriterio ($Patron, $Orden, $Publico=-1);
```

```
function RegistrarBoletin ($email, $nombre, $idioma="", $grupo="") ;
function SetUsuario($email, $clave);
function GetUsuario();
function ActualizarUsuario ($datos) ;
function RegistrarUsuario ($datos) ;

function GetCanal ($Patron, $Hijos=true);
function GetCanales();

function RegistroEstadistico($campos=array());

function GoogleAnalitics();

function EnviarEmail ($from, $fromname, $to, $toname, $contenido, $datos);

function Template($conSeccion=true);

function HtmlBoletin($boletin);

function ProcessCode($datos, $code, $archivo="");
```

8.2. Clase MKWIdiomas:

```
var
        $Conn, $Idioma, $Idiomas;

function MKWIdiomas ();
function Cargar($Conn);
function GetIdioma ();
function GetIdiomas ();
function SetIdioma($Patron);
function GetTraduccion($Palabra, $Idioma="");
```

8.3. Clase MKWFind:

```
var
        $Conn, $Idioma;

function MKWFind ();

function Cargar ($Conn, $Idioma);

function FindDoc($Patron);
function FindAlbum ($Patron);

function EscogerBanner($enlaces);
function FindMenu($Patron, $Hijos=true);
function FindSubMenu ($Patron, $Padre);

function FindContenidoContenidos ($Patron);
function FindContenidoDocumentos ($Patron);
function FindContenidoMenus ($Patron);
function FindContenido($Patron, $Hijos=true);
function FindContenidosCriterio($Patron, $Orden, $Pagina, $RPP);
```

```
function FindSeccionContenidos ($Patron);
function FindSeccionMenus ($Patron);
function FindSeccion($Patron, $Hijos=true);

function FindBoletinContenidos ($Patron);
function FindBoletinMenus ($Patron);
function FindBoletin($Patron, $Hijos=true, $Publico=-1);
function FindBoletinesCriterio($Patron, $Orden, $Publico=-1);

function FindMetas($Patron);

function FindCanalContenidos ($Patron);
function FindCanal($Patron, $Hijos=false);
function FindCanales ($filtro);
```

8.4. Clase MKWUsuario:

```
var
        $Conn, $Usuario, $Permisos,
        $ctrlseccion, $perfil;

function MKWUsuario ();

function Cargar ($Conn, $ctrlseccion, $perfil);

function SetCliente ($email="", $pass="");

function ActualizarUsuario ($datos) ;

function RegistrarUsuario ($datos) ;

function Permiso($Tipo, $Id);

function RegistrarBoletin ($email, $nombre, $idioma, $grupo, $ctrlseccion);
```

8.5. Clase MKWMail:

```
var
        $from, $fromname,
        $tolst, $contenido;

function MKWMail ($from="", $fromname="", $to="", $toname="", $contenido=array());

function Agregar($to, $toname);

function Enviar ($to="", $toname="", $contenido=array());
```

8.6. Clase MKWTemplate:

```
var
        $MKW, $idiomas, $idioma, $ctrlseccion, $is_listado, $plantillas, $s_estilo, $c_estilo, $estilo;
        $datos, $path, $ctipo, $archivo, $Error, $strResultado;
        $indBloqueId, $indBloqueLim, $variables;
```

```
        $htmlIniSeccion, $htmlIniContenido, $temCon, $temSecc;


function MKWTemplate ($MKW);

function Datos ($path, $ctrlseccion, $idiomas);

function Open ($seccion, $ctipo, $contenido, $htmlIniSeccion="", $htmlIniContenido="",
                $temSecc="", $temCon="");

function Process($conSeccion);

function HayError();
function ErrorMsg();
function ErrorTxt();
function ErrorFile();
```

8.7. APIFormat.inc:

```
function QuitarComodines($texto, $idioma);
function PonerComodines($texto, $idioma);

function FormatHref ($idioma, $seccion, $tipo=CONTENIDO_CONTENIDO_CONTENIDO,
                      $contenido=0, $href_accion="", $transformar=true, $ssl=false);

function FormatDoc($Campos, $Idioma);
function FormatMenu($Campos, $Grafico, $Idioma);
function FormatSubMenu($Campos, $Grafico, $Idioma);
function FormatContenido($Campos, $Grafico, $Mini, $Idioma);
function FormatSeccion($Campos, $Grafico, $Idioma);
function FormatBoletin($Campos, $Grafico, $Idioma);
function FormatMeta ($Campos, $Idioma);
function FormatCanal($Campos, $Grafico, $Idioma);
function FormatUsuario($Campos);
```

8.8. APIDefines.inc:

```
// ************************************
// ** CONFIGURARCIÓN DE UN LLAMADA
// ************************************

define(PARAMETRO_IDIOMA,"idioma");
define(PARAMETRO_SECCION,"seccion");
define(PARAMETRO_CONTENIDO_TIPO,"ctipo");
define(PARAMETRO_CONTENIDO,"contenido");
define(PARAMETRO_ACCION,"accion");

// ************************************
// ** CAMPOS EN LOS ARRAYS DE DATOS
// ************************************

define(CAMPO_ABREV, "abrev");
define(CAMPO_ACTIVO, "activo");
define(CAMPO_ALBUMES,"albumes");
define(CAMPO_ALTO, "alto");
```

```
define(CAMPO_ANCHO, "ancho");
define(CAMPO_APP, "app");
define(CAMPO_ARCHIVO, "archivo");
define(CAMPO_ASUNTO, "asunto");
define(CAMPO_CANAL, "canal");
define(CAMPO_CANALES, "canales");
define(CAMPO_CONTENIDO, "contenido");
define(CAMPO_CONTENIDO_HTML, "contenido_html");
define(CAMPO_CONTENIDO_TIPO, "contenidotipo");
define(CAMPO_CONTENIDOS, "contenidos");
define(CAMPO_DATOS, "datos");
define(CAMPO_DISCO, "disco");
define(CAMPO_DOC_CONTENIDO, "contenido");
define(CAMPO_DOC_HEIGHT, "height");
define(CAMPO_DOC_MIME, "mime");
define(CAMPO_DOC_SIZE, "size");
define(CAMPO_DOC_TYPE, "type");
define(CAMPO_DOC_WIDTH, "width");
define(CAMPO_DOC_NOMBRE, "nombre");
define(CAMPO_DOC, "doc");
define(CAMPO_DOCS, "docs");
define(CAMPO_EMAIL_ASUNTO, "email_asunto");
define(CAMPO_EMAIL_FROM, "email_from");
define(CAMPO_EMAIL_NAME, "email_nombre");
define(CAMPO_EMAIL_TEXTO, "email_texto");
define(CAMPO_EMAIL_HTML, "email_html");
define(CAMPO_ENLACE, "enlace");
define(CAMPO_ENLACES, "enlaces");
define(CAMPO_ESTILO, "estilo");
define(CAMPO_ESTADO, "estado");
define(CAMPO_GRAFICO, "grafico");
define(CAMPO_GRUPOS, "grupos");
define(CAMPO_HTML, "html");
define(CAMPO_HTML_ENLACE, "htmlenlace");
define(CAMPO_HREF, "href");
define(CAMPO_ID, "id");
define(CAMPO_IDIOMA, "idioma");
define(CAMPO_MENUS, "menus");
define(CAMPO_METAS, "metas");
define(CAMPO_MIGAS_CONTENIDO, "miga_contenido");
define(CAMPO_MIGAS_COPIA, "miga_copia");
define(CAMPO_MIGAS_GET, "miga_get");
define(CAMPO_MIGAS_IDIOMA, "miga_idioma");
define(CAMPO_MIGAS_PARAMETROS, "miga_parametros");
define(CAMPO_MIGAS_POST, "miga_post");
define(CAMPO_MIGAS_SECCION, "miga_seccion");
define(CAMPO_MIGAS_USUARIO, "miga_usuario");
define(CAMPO_MINI, "mini");
define(CAMPO_NIVEL, "nivel");
define(CAMPO_NOMBRE, "nombre");
define(CAMPO_NUMERO, "num");
define(CAMPO_ORIGEN, "origen");
define(CAMPO_PADRE, "padre");
define(CAMPO_PAGINA, "pagina");
define(CAMPO_PAGINAS, "paginas");
define(CAMPO_PAGO, "pago");
define(CAMPO_PESO, "peso");
```

```
define(CAMPO_PRIVADO, "privado");
define(CAMPO_POPUP, "popup");
define(CAMPO_RESUMEN, "resumen");
define(CAMPO_SECCION, "seccion");
define(CAMPO_SIZE, "size");
define(CAMPO_SSL, "ssl");
define(CAMPO_SUBTITULO, "subtitulo");
define(CAMPO_TEXTO, "texto");
define(CAMPO_TITULO, "titulo");
define(CAMPO_TOTAL, "total");
define(CAMPO_USUARIO, "usuario");
define(CAMPO_USUARIO_CONTACTO,"usu_contaco");
define(CAMPO_USUARIO_CLAVE,"usu_pass");
define(CAMPO_USUARIO_CP,"usu_cp");
define(CAMPO_USUARIO_DIRECCION,"usu_direccion");
define(CAMPO_USUARIO_DNI,"usu_dni");
define(CAMPO_USUARIO_EMAIL,"usu_email");
define(CAMPO_USUARIO_EMPRESA,"usu_empresa");
define(CAMPO_USUARIO_IDIOMA,"usu_idioma");
define(CAMPO_USUARIO_PAIS,"usu_pais");
define(CAMPO_USUARIO_POBLACION,"usu_poblacion");
define(CAMPO_USUARIO_PROVINCIA,"usu_provincia");
define(CAMPO_USUARIO_TELEFONO,"usu_telefono");
define(CAMPO_USUARIO_MOVIL,"usu_movil");
define(CAMPO_USUARIO_PERFIL,"usu_perfil");


// ************************************
// ** DEFINICIÓN DE LOS CONTENIDOS
// ************************************

$i=-1;
define(CONTENIDO_NINGUNO, ++$i);
define(CONTENIDO_MAQUETACION_MENU, ++$i);
define(CONTENIDO_CONTENIDO_CONTENIDO, ++$i);
define(CONTENIDO_CONTENIDO_DOCUMENTO, ++$i);
define(CONTENIDO_CONTENIDO_IMAGEN, ++$i);
define(CONTENIDO_CONTENIDO_MULTIMEDIA, ++$i);
define(CONTENIDO_CONTENIDO_ALBUM, ++$i);
define(CONTENIDO_BOLETIN_BOLETIN, ++$i);
define(CONTENIDO_CANAL_CANAL, ++$i);
define(CONTENIDO_ACCION, ++$i);
define(CONTENIDO_EMAIL, ++$i);
unset($i);


// ************************************
// ** COMODINES
// ************************************

define(COMODIN_IDIOMA, "%idioma%");
define(COMODIN_SERVIDOR, "%servidor%");
define(COMODIN_SERVIDOR_FIJO, "%servidor_fijo%");
define(COMODIN_CAMPOS, "%");
```

```
// ************************************
// ** PLANTILLAS
// ************************************

define(TEMPLATE_PLANTILLAS_SEPARADOR, "|");
define(TEMPLATE_PLANTILLAS_EXTENSION, ".tpl");
define(TEMPLATE_PLANTILLAS_PROCESOS, "process.inc");
define(TEMPLATE_PLANTILLAS_LISTADO_CONTENIDO,
        "main".TEMPLATE_PLANTILLAS_SEPARADOR.
        "menu".TEMPLATE_PLANTILLAS_SEPARADOR.
        "contenido".TEMPLATE_PLANTILLAS_SEPARADOR.
        "documento".TEMPLATE_PLANTILLAS_SEPARADOR.
        "imagen".TEMPLATE_PLANTILLAS_SEPARADOR.
        "album".TEMPLATE_PLANTILLAS_SEPARADOR.
        "boletin".TEMPLATE_PLANTILLAS_SEPARADOR.
        "canal");
define(TEMPLATE_PLANTILLAS_LISTADO_LISTADOS,
        "main".TEMPLATE_PLANTILLAS_SEPARADOR.
        "lstmenus".TEMPLATE_PLANTILLAS_SEPARADOR.
        "lstcontenidos".TEMPLATE_PLANTILLAS_SEPARADOR.
        "lstdocumentos".TEMPLATE_PLANTILLAS_SEPARADOR.
        "lstimagenes".TEMPLATE_PLANTILLAS_SEPARADOR.
        "lstalbunes".TEMPLATE_PLANTILLAS_SEPARADOR.
        "lstboletines".TEMPLATE_PLANTILLAS_SEPARADOR.
        "lstcanales");

define(TEMPLATE_DELIMIT_IN_INI, "{");
define(TEMPLATE_DELIMIT_IN_FIN, "}");
define(TEMPLATE_DELIMIT_OUT_INI, "{/");
define(TEMPLATE_DELIMIT_OUT_FIN, "}");
define(TEMPLATE_DELIMIT_SEPARATOR, " ");
define(TEMPLATE_DELIMIT_ASIGN, "=");

define(TEMPLATE_DEFINE_CAMPO_INICIO, "//");
define(TEMPLATE_DEFINE_CAMPO_LOCAL, "/");
define(TEMPLATE_DEFINE_CAMPO_BUSCAR, "#");
define(TEMPLATE_DEFINE_CAMPOS_CAMPOS, "/");
define(TEMPLATE_DEFINE_CAMPOS_ID, ":");

define(TEMPLATE_DEFINE_BLOQUE, "block");
define(TEMPLATE_DEFINE_ITEM, "item");
define(TEMPLATE_DEFINE_ENLACE, "enlace");
define(TEMPLATE_DEFINE_PHP, "php");
define(TEMPLATE_DEFINE_IF, "if");
define(TEMPLATE_DEFINE_DICC, "dicc");
define(TEMPLATE_DEFINE_ASSIGN, "assign");
define(TEMPLATE_DEFINE_VALUE, "value");

define(TEMPLATE_CAMPO_NOMBRE, "name");
define(TEMPLATE_CAMPO_CAMPO, "field");
define(TEMPLATE_CAMPO_PADRE, "block");
define(TEMPLATE_CAMPO_NUMERO, "id");
define(TEMPLATE_CAMPO_SRC, "src");
define(TEMPLATE_CAMPO_SENTIDO, "sen");
define(TEMPLATE_CAMPO_FORMAT, "format");
define(TEMPLATE_CAMPO_SIZE, "size");
define(TEMPLATE_CAMPO_CONDICION, "con");
```

```
define(TEMPLATE_CAMPO_VALOR, "value");
define(TEMPLATE_CAMPO_TAGS, "tags");

define(TEMPLATE_CAMPO_SENTIDO_DEBAJO, -1);
define(TEMPLATE_CAMPO_SENTIDO_NULO, "0");
define(TEMPLATE_CAMPO_SENTIDO_ENCIMA, 1);

define(TEMPLATE_CAMPO_CONDICION_DES, "0");
define(TEMPLATE_CAMPO_CONDICION_IGU, "1");
define(TEMPLATE_CAMPO_CONDICION_MAY, "2");
define(TEMPLATE_CAMPO_CONDICION_MEN, "3");
define(TEMPLATE_CAMPO_CONDICION_MAI, "4");
define(TEMPLATE_CAMPO_CONDICION_MEI, "5");


define(TEMPLATE_BLOQUE_MAIN, "main");
define(TEMPLATE_ARCHIVO_MAIN, "main");
define(TEMPLATE_BLOQUE_CONTENIDO, "contenido");
```

9. Datos generados.

Al procesar la información en la API y buscarla en la base de datos con alguna de sus clases, los resultado vuelven siempre debidamente formateados por las funciones en el archivo "*APIFormat.inc*".

A cualquier estructura generada por la API, se puede añadir más datos en los campos que el programador defina, y que luego puedan ser accedidos desde las plantillas. Como cuando queremos ampliar datos de algún contenido o sección.

Para describir los vectores de datos que se pueden encontrar, empezaremos por la que tiene inicialmente una plantilla que va ha procesar una página Web e iremos bajando de nivel.

9.1. Datos iniciales:

$datos	datos de una sección.
$datos[CAMPO_CONTENIDO]	datos de un contenido.
$datos[CAMPO_CONTENIDO_HTML]	resultado del procesado del contenido.

Por ejemplo, desde una de las plantillas de sección (main.tpl) podría accederse a estos datos de la siguiente manera:

```
{item name=titulo_seccion field=/CAMPO_TITULO}
{item name=titulo_contenido field=/CAMPO_CONTENIDO/CAMPO_TITULO}

{item name=resultado_del_contenido field=/CAMPO_CONTENIDO_HTML}
```

La última línea es ejemplo de cómo enlazar en la plantilla de la sección lo que se ha procesado en el contenido.

9.2. Datos de una sección:

$seccion[CAMPO_ID]	identificador de la sección.
$seccion[CAMPO_NOMBRE]	nombre de la sección.
$seccion[CAMPO_TITULO]	título de la sección.
$seccion[CAMPO_GRAFICO]	datos del documento (gráfico).
$seccion[CAMPO_HTML]	código html equivalente (gráfico o título).
$seccion[CAMPO_ESTILO]	estilo de la sección.

Si ha cargado los datos relacionados con la sección:

$seccion[CAMPO_MENUS]	listado de menús (sin datos relacionados).
$seccion[CAMPO_CONTENIDOS]	listado de contenidos (sin datos relacionados).
$seccion[CAMPO_METAS]	listado de meta-tags.

9.3. Datos de un contenido:

$contenido[CAMPO_ID]	identificador del contenido.
$contenido[CAMPO_NOMBRE]	nombre del contenido.
$contenido[CAMPO_PRIVADO]	identifica como un contenido privado.
$contenido[CAMPO_TITULO]	título del contenido.
$contenido[CAMPO_SUBTITULO]	subtítulo del contenido.
$contenido[CAMPO_RESUMEN]	resumen del contenido.

$contenido[CAMPO_TEXTO]	texto del contenido.
$contenido[CAMPO_MINI]	datos del documento (gráfico pequeño).
$contenido[CAMPO_GRAFICO]	datos del documento (gráfico grande).
$contenido[CAMPO_CANAL]	identificador del canal RSS al que está registrado.
$contenido[CAMPO_SECCION]	identificador de la sección en la que debería estar.
$contenido[CAMPO_ESTILO]	estilo del contenido.
$contenido[CAMPO_METAS]	listado de meta-tags.

Si ha cargado los datos relacionados con el contenido:

$contenido[CAMPO_MENUS]	listado de menús (sin datos relacionados).
$contenido[CAMPO_CONTENIDOS]	listado de contenidos (sin datos relacionados).
$contenido[CAMPO_DOCS]	listado de documentos.

Por ejemplo, desde una de las plantillas de contenidos (contenido.tpl):

```
{item name=identificador_seccion field=//CAMPO_ID}
{item name=titulo_contenido field=/ CAMPO_TITULO}
```

9.4. Datos de un documento:

$documento[CAMPO_ID]	identificador del documento.
$documento[CAMPO_NOMBRE]	nombre del documento.
$documento[CAMPO_PRIVADO]	identifica como un documento privado.
$documento[CAMPO_DISCO]	identifica como un documento en disco.
$documento[CAMPO_TITULO]	título del documento.
$documento[CAMPO_ARCHIVO]	enlace para ver el documento.
$documento[CAMPO_APP]	tipo de documento.
$documento[CAMPO_SIZE]	tamaño del documento.
$documento[CAMPO_ALTO]	alto del documento.
$documento[CAMPO_ANCHO]	ancho del documento.
$documento[CAMPO_GRAFICO]	identifica como un gráfico.
$documento[CAMPO_CONTENIDO]	contenido del documento si no está en disco.
$documento[CAMPO_ORIGEN]	nombre del archivo de origen del documento.
$documento[CAMPO_TITULO]	título del documento.
$documento[CAMPO_HTML]	código html si es un gráfico.

9.5. Datos de un menú:

$menu[CAMPO_ID]	identificador del menú.
$menu[CAMPO_NOMBRE]	nombre del menú.
$menu[CAMPO_TITULO]	título del menú.
$menu[CAMPO_GRAFICO]	datos del documento (gráfico).
$menu[CAMPO_HTML]	código html (gráfico o título).

Si ha cargado los datos relacionados con el menú:

$menú[CAMPO_ENLACES]	listado de enlaces.

9.6. Datos de un enlace:

$enlace[CAMPO_ID]	identificador del enlace.
$enlace[CAMPO_PESO]	peso del enlace.
$enlace[CAMPO_TITULO]	título del menú.

$enlace[CAMPO_GRAFICO]	datos del documento (gráfico).
$enlace[CAMPO_SECCION]	identificador de la sección del enlace.
$enlace[CAMPO_CONTENIDO]	identificador del contenido del enlace.
$enlace[CAMPO_CONTENIDO_TIPO]	identificador del tipo de contenido del enlace.
$enlace[CAMPO_HREF]	acción o dirección web del enlace.
$enlace[CAMPO_POPUP]	indica si debe abrirse en otra ventana.
$enlace[CAMPO_SSL]	indica si debe usarse servidor seguro.
$enlace[CAMPO_ENLACE]	dirección del enlace con los datos anteriores.
$enlace[CAMPO_HTML_ENLACE]	la dirección en html (.
$enlace[CAMPO_HTML]	código html del enlace con el título o el gráfico: (<a ..>título o gráfico).

9.7. Datos de los metas:

$metatag[CAMPO_NOMBRE]	nombre del met-tag.
$metatag[CAMPO_HTML]	código html (<meta> o <title>).

9.8. Datos de los listados:

$listado[CAMPO_NUMERO]	número de elementos.
$listado[1]	datos del elemento número 1.
$listado[…]	datos del elemento número ….

9.9. Datos de un boletín:

$boletin[CAMPO_ID]	identificador del boletín.
$boletin[CAMPO_NOMBRE]	nombre del boletín.
$boletin[CAMPO_ASUNTO]	asunto del boletín.
$boletin[CAMPO_TITULO]	título del boletín.
$boletin[CAMPO_SUBTITULO]	subtitulo del boletín.
$boletin[CAMPO_ESTILO]	estilo del boletín.
$boletin[CAMPO_GRAFICO]	datos del documento (gráfico).
$boletin[CAMPO_HTML]	código html del boletín (gráfico o título).

Si ha cargado los datos relacionados con el boletín:

$boletin[CAMPO_MENUS]	listado de menús.
$boletin[CAMPO_CONTENIDOS]	listado de contenidos.

9.10. Datos de un canal:

$canal[CAMPO_ID]	identificador del boletín.
$canal[CAMPO_NOMBRE]	nombre del boletín.
$canal[CAMPO_TITULO]	título del boletín.
$canal[CAMPO_RESUMEN]	resumen del boletín.
$canal[CAMPO_GRAFICO]	datos del documento (gráfico).

Si ha cargado los datos relacionados con el menú:

$menú[CAMPO_ENLACES]	listado de contenidos (sin datos relacionados).

9.11. Datos de un usuario:

$usuario[CAMPO_ID]	identificador del usuario.
$usuario[CAMPO_USUARIO_CONTACTO]	nombre de contacto del usuario.
$usuario[CAMPO_USUARIO_EMPRESA]	empresa del usuario.
$usuario[CAMPO_USUARIO_DNI]	DNI o CIF del usuario.
$usuario[CAMPO_USUARIO_DIRECCION]	dirección del usuario.
$usuario[CAMPO_USUARIO_POBLACION]	población del usuario.
$usuario[CAMPO_USUARIO_CP]	C.P. del usuario.
$usuario[CAMPO_USUARIO_PROVINCIA]	provincia del usuario.
$usuario[CAMPO_USUARIO_PAIS]	país del usuario.
$usuario[CAMPO_USUARIO_IDIOMA]	idioma del usuario.
$usuario[CAMPO_USUARIO_TELEFONO]	teléfono del usuario.
$usuario[CAMPO_USUARIO_MOVIL]	móvil del usuario.
$usuario[CAMPO_USUARIO_EMAIL]	email del usuario.
$usuario[CAMPO_USUARIO_CLAVE]	clave de identificación del usuario.
$usuario[CAMPO_USUARIO_PERFIL]	perfil de acceso del usuario.