Hyperwave

Introduction

Hyperwave has been developed at » IICM in Graz. It started with the name Hyper-G and changed to Hyperwave when it was commercialised (in 1996).

Hyperwave is not free software. The current version, 5.5 is available at » http://www.hyperwave.com/. A time limited version can be ordered for free (30 days).

See also the Hyperwave API module.

Hyperwave is an information system similar to a database (HIS, Hyperwave Information Server). Its focus is the storage and management of documents. A document can be any possible piece of data that may as well be stored in file. Each document is accompanied by its object record. The object record contains meta data for the document. The meta data is a list of attributes which can be extended by the user. Certain attributes are always set by the Hyperwave server, other may be modified by the user. An attribute is a name/value pair of the form name=value. The complete object record contains as many of those pairs as the user likes. The name of an attribute does not have to be unique, e.g. a title may appear several times within an object record. This makes sense if you want to specify a title in several languages. In such a case there is a convention, that each title value is preceded by the two letter language abbreviation followed by a colon, e.g. *'en:Title in English'* or *'ge:Titel in deutsch'*. Other attributes like a description or keywords are potential candidates. You may also replace the language abbreviation by any other string as long as it separated by colon from the rest of the attribute value.

Each object record has native a string representation with each name/value pair separated by a newline. The Hyperwave extension also knows a second representation which is an associated array with the attribute name being the key. Multilingual attribute values itself form another associated array with the key being the language abbreviation. Actually any multiple attribute forms an associated array with the string left to the colon in the attribute value being the key. (This is not fully implemented. Only the attributes Title, Description and Keyword are treated properly yet.)

Besides the documents, all hyper links contained in a document are stored as object records as well. Hyper links which are in a document will be removed from it and stored as individual objects, when the document is inserted into the database. The object record of the link contains information about where it starts and where it ends. In order to gain the original document you will have to retrieve the plain document without the links and the list of links and reinsert them. The functions hw_pipedocument() and hw_gettext() do this for you. The advantage of separating links from the document is obvious. Once a document to which a link is pointing to changes its name, the link can easily be modified accordingly. The document containing the link is not affected at all. You may even add a link to a document without modifying the document itself.

Saying that hw_pipedocument() and hw_gettext() do the link insertion automatically is not as simple as it sounds. Inserting links implies a certain hierarchy of the documents. On a web server this is given by the file system, but Hyperwave has its own hierarchy and names do not reflect the position of an object in that hierarchy. Therefore creation of links first of all requires a mapping from the Hyperwave hierarchy and namespace into a web hierarchy respective web namespace. The fundamental difference between Hyperwave

and the web is the clear distinction between names and hierarchy in Hyperwave. The name does not contain any information about the objects position in the hierarchy. In the web the name also contains the information on where the object is located in the hierarchy. This leads to two possibles ways of mapping. Either the Hyperwave hierarchy and name of the Hyperwave object is reflected in the URL or the name only. To make things simple the second approach is used. Hyperwave object with name *my_object* is mapped to *http://host/my_object* disregarding where it resides in the Hyperwave hierarchy. An object with name *parent/my_object* could be the child of *my_object* in the Hyperwave hierarchy, though in a web namespace it appears to be just the opposite and the user might get confused. This can only be prevented by selecting reasonable object names.

Having made this decision a second problem arises. How do you involve PHP? The URL *http://host/my_object* will not call any PHP script unless you tell your web server to rewrite it to e.g. *http://host/php_script/my_object* and the script *php_script* evaluates the *$PATH_INFO* variable and retrieves the object with name *my_object* from the Hyperwave server. Their is just one little drawback which can be fixed easily. Rewriting any URL would not allow any access to other document on the web server. A PHP script for searching in the Hyperwave server would be impossible. Therefore you will need at least a second rewriting rule to exclude certain URLs like all e.g. starting with *http://host/Hyperwave* This is basically sharing of a namespace by the web and Hyperwave server.

Based on the above mechanism links are insert into documents.

It gets more complicated if PHP is not run as a server module or CGI script but as a standalone application e.g. to dump the content of the Hyperwave server on a CD-ROM. In such a case it makes sense to retain the Hyperwave hierarchy and map in onto the file system. This conflicts with the object names if they reflect its own hierarchy (e.g. by choosing names including '/'). Therefore '/' has to be replaced by another character, e.g. '_'.

The network protocol to communicate with the Hyperwave server is called » HG-CSP (Hyper-G Client/Server Protocol). It is based on messages to initiate certain actions, e.g. get object record. In early versions of the Hyperwave Server two native clients (Harmony, Amadeus) were provided for communication with the server. Those two disappeared when Hyperwave was commercialised. As a replacement a so called wavemaster was provided. The wavemaster is like a protocol converter from HTTP to HG-CSP. The idea is to do all the administration of the database and visualisation of documents by a web interface. The wavemaster implements a set of placeholders for certain actions to customise the interface. This set of placeholders is called the PLACE Language. PLACE lacks a lot of features of a real programming language and any extension to it only enlarges the list of placeholders. This has led to the use of JavaScript which IMO does not make life easier.

Adding Hyperwave support to PHP should fill in the gap of a missing programming language for interface customisation. It implements all the messages as defined by the HG-CSP but also provides more powerful commands to e.g. retrieve complete documents.

Hyperwave has its own terminology to name certain pieces of information. This has widely been taken over and extended. Almost all functions operate on one of the following data types.

- object ID: A unique integer value for each object in the Hyperwave server. It is also one of the attributes of the object record (ObjectID). Object ids are often used as an

input parameter to specify an object.

- object record: A string with attribute-value pairs of the form attribute=value. The pairs are separated by a carriage return from each other. An object record can easily be converted into an object array with **hw_object2array()**. Several functions return object records. The names of those functions end with obj.
- object array: An associative array with all attributes of an object. The keys are the attribute names. If an attribute occurs more than once in an object record it will result in another indexed or associative array. Attributes which are language depended (like the title, keyword, description) will form an associative array with the keys set to the language abbreviations. All other multiple attributes will form an indexed array. PHP functions never return object arrays.
- hw_document: This is a complete new data type which holds the actual document, e.g. HTML, PDF etc. It is somewhat optimized for HTML documents but may be used for any format.

Several functions which return an array of object records do also return an associative array with statistical information about them. The array is the last element of the object record array. The statistical array contains the following entries:

Hidden
: Number of object records with attribute PresentationHints set to Hidden.

CollectionHead
: Number of object records with attribute PresentationHints set to CollectionHead.

FullCollectionHead
: Number of object records with attribute PresentationHints set to FullCollectionHead.

CollectionHeadNr
: Index in array of object records with attribute PresentationHints set to CollectionHead.

FullCollectionHeadNr
: Index in array of object records with attribute PresentationHints set to FullCollectionHead.

Total
: Total: Number of object records.

Installing/Configuring

Requirements

This extension needs a Hyperwave server downloadable from » http://www.hyperwave.com/.

Installation

This » PECL extension is not bundled with PHP.

In order to use these functions you must compile PHP with Hyperwave support by using the *--with-hyperwave[=DIR]* configure option.

Windows users will enable *php_hyperwave.dll* inside of *php.ini* in order to use these functions. The DLL for this PECL extension may be downloaded from either the » PHP Downloads page or from » http://pecl4win.php.net/

Runtime Configuration

The behaviour of these functions is affected by settings in *php.ini*.

Hyperwave configuration options

Name	Default	Changeable	Changelog
hyperwave.allow_persistent	"0"	PHP_INI_SYSTEM	Available since PHP 4.3.2. Removed in PHP 5.0.0.
hyperwave.default_port	"418"	PHP_INI_ALL	Removed in PHP 5.0.0.

For further details and definitions of the PHP_INI_* constants, see the php.ini directives.

Resource Types

This extension has no resource types defined.

Predefined Constants

The constants below are defined by this extension, and will only be available when the extension has either been compiled into PHP or dynamically loaded at runtime.

HW_ATTR_LANG (integer)

HW_ATTR_NR (integer)

HW_ATTR_NONE (integer)

Integration with Apache

The Hyperwave extension is best used when PHP is compiled as an Apache module. In such a case the underlying Hyperwave server can be hidden from users almost completely if Apache uses its rewriting engine. The following instructions will explain this.

Since PHP with Hyperwave support built into Apache is intended to replace the native Hyperwave solution based on Wavemaster, we will assume that the Apache server will only serve as a Hyperwave web interface for these examples. This is not necessary but it simplifies the configuration. The concept is quite simple. First of all you need a PHP script which evaluates the *$_ENV['PATH_INFO']* variable and treats its value as the name of a Hyperwave object. Let's call this script *'Hyperwave'*. The URL *http://your.hostname/Hyperwave/name_of_object* would than return the Hyperwave object with the name *'name_of_object'*. Depending on the type of the object the script has to react accordingly. If it is a collection, it will probably return a list of children. If it is a document it will return the mime type and the content. A slight improvement can be achieved if the Apache rewriting engine is used. From the users point of view it would be more straight forward if the URL *http://your.hostname/name_of_object* would return the object. The rewriting rule is quite easy:

```
RewriteRule ^/(.*) /usr/local/apache/htdocs/HyperWave/$1 [L]
```

Now every URL relates to an object in the Hyperwave server. This causes a simple to solve problem. There is no way to execute a different script, e.g. for searching, than the 'Hyperwave' script. This can be fixed with another rewriting rule like the following:

```
RewriteRule ^/hw/(.*) /usr/local/apache/htdocs/hw/$1 [L]
```

This will reserve the directory */usr/local/apache/htdocs/hw* for additional scripts and other files. Just make sure this rule is evaluated before the one above. There is just a little drawback: all Hyperwave objects whose name starts with *'hw/'* will be shadowed. So, make sure you don't use such names. If you need more directories, e.g. for images just add more rules or place them all in one directory. Before you put those instructions, don't forget to turn on the rewriting engine with

```
RewriteEngine on
```

You will need scripts:

- to return the object itself
- to allow searching
- to identify yourself
- to set your profile
- one for each additional function like to show the object attributes, to show information about users, to show the status of the server, etc.

As an alternative to the Rewrite Engine, you can also consider using the Apache

ErrorDocument directive, but be aware, that *ErrorDocument* redirected pages cannot receive POST data.

Hyperwave Functions

Todo

There are still some things to do:

- The hw_InsertDocument has to be split into hw_insertobject() and **hw_putdocument()** .
- The names of several functions are not fixed, yet.
- Most functions require the current connection as its first parameter. This leads to a lot of typing, which is quite often not necessary if there is just one open connection. A default connection will improve this.
- Conversion form object record into object array needs to handle any multiple attribute.

hw_Array2Objrec

hw_Array2Objrec -- Convert attributes from object array to object record

Description

string **hw_array2objrec** (array `$object_array`)

Converts an *`object_array`* into an object record. Multiple attributes like 'Title' in different languages are treated properly.

Parameters

`object_array`
The array.

Return Values

Returns an object record.

See Also

- hw_objrec2array()

hw_changeobject

hw_changeobject -- Changes attributes of an object (obsolete)

Description

bool **hw_changeobject** (int `$link`, int `$objid`, array `$attributes`)

Warning
This function is currently not documented; only its argument list is available.

hw_Children

hw_Children -- Object ids of children

Description

array **hw_children** (int `$connection`, int `$objectID`)

Returns the identifiers of the collection children.

Parameters

`connection`
The connection identifier.

`objectID`
The object identifier.

Return Values

Returns an array of object ids. Each id belongs to a child of the collection with ID *`objectID`*. The array contains all children both documents and collections.

hw_ChildrenObj

hw_ChildrenObj -- Object records of children

Description

array **hw_childrenobj** (int `$connection`, int `$objectID`)

Returns the object records of the collection children.

Parameters

`connection`
The connection identifier.

`objectID`
The object identifier.

Return Values

Returns an array of object records. Each object record belongs to a child of the collection with ID *`objectID`*. The array contains all children both documents and collections.

hw_Close

hw_Close -- Closes the Hyperwave connection

Description

bool **hw_close** (int `$connection`)

Closes down the connection to a Hyperwave server.

Parameters

`connection`
The connection identifier.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

hw_Connect

hw_Connect -- Opens a connection

Description

int **hw_connect** (string `$host`, int `$port` [, string `$username`], string `$password`)

Opens a connection to a Hyperwave server. You can have multiple connections open at once.

Parameters

`host`
: The server host name.

`port`
: The server port number.

`username`
: The Hyperwave user name. If ommited, no identification with the server will be done. It is similar to identify as user anonymous.

`password`
: The password for *`username`*. Keep in mind, that the password is not encrypted.

Return Values

Returns a connection index on success, or **FALSE** if the connection could not be made.

See Also

- hw_pconnect()

hw_connection_info

hw_connection_info -- Prints information about the connection to Hyperwave server

Description

void **hw_connection_info** (int `$link`)

Warning
This function is currently not documented; only its argument list is available.

hw_cp

hw_cp -- Copies objects

Description

int **hw_cp** (int `$connection`, array `$object_id_array`, int `$destination_id`)

Copies the objects with object ids as specified in the second parameter to the collection with the id *`destination id`*.

Parameters

`connection`
: The connection identifier.

`object_id_array`
: An array of object ids.

`destination_id`
: The target collection id.

Return Values

Returns the number of copied objects.

See Also

- hw_mv()

hw_Deleteobject

hw_Deleteobject -- Deletes object

Description

bool **hw_deleteobject** (int `$connection`, int `$object_to_delete`)

Deletes the object with the given object id in the second parameter. It will delete all instances of the object.

Parameters

`connection`
The connection identifier.

`object_to_delete`
The object identifier.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

See Also

- hw_mv()

hw_DocByAnchor

hw_DocByAnchor -- Object id object belonging to anchor

Description

int **hw_docbyanchor** (int `$connection`, int `$anchorID`)

Returns an th object id of the document to which *`anchorID`* belongs.

Parameters

`connection`
The connection identifier.

`anchorID`
The anchor identifier.

Return Values

Returns the document object id.

hw_DocByAnchorObj

hw_DocByAnchorObj -- Object record object belonging to anchor

Description

string **hw_docbyanchorobj** (int `$connection`, int `$anchorID`)

Returns an th object record of the document to which *`anchorID`* belongs.

Parameters

`connection`
The connection identifier.

`anchorID`
The anchor identifier.

Return Values

Returns an object record.

hw_Document_Attributes

hw_Document_Attributes -- Object record of hw_document

Description

string **hw_document_attributes** (int `$hw_document`)

Returns the object record of the document.

For backward compatibility, **hw_documentattributes()** is also accepted. This is deprecated, however.

Parameters

`hw_document`
The document identifier.

Return Values

Returns the object record of the document.

See Also

- hw_document_bodytag()
- hw_document_size()

hw_Document_BodyTag

hw_Document_BodyTag -- Body tag of hw_document

Description

string **hw_document_bodytag** (int `$hw_document` [, string `$prefix`])

Returns the BODY tag of the document. If the document is an HTML document the BODY tag should be printed before the document.

For backward compatibility, **hw_documentbodytag()** is also accepted. This is deprecated, however.

Parameters

`hw_document`
The document identifier.

`prefix`

Return Values

Returns the BODY tag as a string.

See Also

- hw_document_attributes()
- hw_document_size()

hw_Document_Content

hw_Document_Content -- Returns content of hw_document

Description

string **hw_document_content** (int `$hw_document`)

Gets the content of the document.

Parameters

`hw_document`
The document identifier.

Return Values

Returns the content of the document. If the document is an HTML document the content is everything after the BODY tag. Information from the HEAD and BODY tag is in the stored in the object record.

See Also

- hw_document_attributes()
- hw_document_size()
- hw_document_setcontent()

hw_Document_SetContent

hw_Document_SetContent -- Sets/replaces content of hw_document

Description

bool **hw_document_setcontent** (int `$hw_document`, string `$content`)

Sets or replaces the content of the document. If the document is an HTML document the content is everything after the BODY tag. Information from the HEAD and BODY tag is in the stored in the object record. If you provide this information in the content of the document too, the Hyperwave server will change the object record accordingly when the document is inserted. Probably not a very good idea. If this functions fails the document will retain its old content.

Parameters

`hw_document`
The document identifier.

`content`

Return Values

Returns **TRUE** on success or **FALSE** on failure.

See Also

- hw_document_attributes()
- hw_document_size()
- hw_document_content()

hw_Document_Size

hw_Document_Size -- Size of hw_document

Description

int **hw_document_size** (int `$hw_document`)

Gets the size of the document.

For backward compatibility, **hw_documentsize()** is also accepted. This is deprecated, however.

Parameters

`hw_document`
The document identifier.

Return Values

Returns the size in bytes of the document.

See Also

- hw_document_attributes()
- hw_document_bodytag()

hw_dummy

hw_dummy -- Hyperwave dummy function

Description

string **hw_dummy** (int `$link`, int `$id`, int `$msgid`)

Warning
This function is currently not documented; only its argument list is available.

hw_EditText

hw_EditText -- Retrieve text document

Description

bool **hw_edittext** (int `$connection`, int `$hw_document`)

Uploads the text document to the server. The object record of the document may not be modified while the document is edited.

This function will only works for pure text documents. It will not open a special data connection and therefore blocks the control connection during the transfer.

Parameters

`connection`
The connection identifier.

`hw_document`
The document identifier.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

See Also

- hw_pipedocument()
- hw_free_document()
- hw_document_bodytag()
- hw_document_size()
- hw_output_document()
- hw_gettext()

hw_Error

hw_Error -- Error number

Description

int **hw_error** (int `$connection`)

Returns the last error number, for the last command.

Parameters

`connection`
: The connection identifier.

Return Values

Returns the last error number or 0 if no error occured.

See Also

- hw_errormsg()

hw_ErrorMsg

hw_ErrorMsg -- Returns error message

Description

string **hw_errormsg** (int `$connection`)

Returns a string containing the last error message related to the last command.

Parameters

`connection`
The connection identifier.

Return Values

Returns a string containing the last error message or 'No Error'. If **FALSE** is returned, this function failed.

See Also

- hw_error()

hw_Free_Document

hw_Free_Document -- Frees hw_document

Description

bool **hw_free_document** (int `$hw_document`)

Frees the memory occupied by the Hyperwave document.

Parameters

`hw_document`
The document identifier.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

hw_GetAnchors

hw_GetAnchors -- Object ids of anchors of document

Description

array **hw_getanchors** (int `$connection`, int `$objectID`)

Returns an array of object ids with anchors of the specified document.

Parameters

`connection`
The connection identifier.

`objectID`
The document object id.

Return Values

Returns an array of object ids.

hw_GetAnchorsObj

hw_GetAnchorsObj -- Object records of anchors of document

Description

array **hw_getanchorsobj** (int `$connection`, int `$objectID`)

Returns records with anchors of the document with object ID *`objectID`*.

Parameters

`connection`
: The connection identifier.

`objectID`
: The object identifier.

Return Values

Returns an array of object records.

hw_GetAndLock

hw_GetAndLock -- Return object record and lock object

Description

string **hw_getandlock** (int `$connection`, int `$objectID`)

Returns the object record for the object with ID *`objectID`*. It will also lock the object, so other users cannot access it until it is unlocked.

Parameters

`connection`
: The connection identifier.

`objectID`
: The object identifier.

Return Values

Returns the object record for the object with ID *`objectID`*.

See Also

- hw_unlock()
- hw_getobject()

hw_GetChildColl

hw_GetChildColl -- Object ids of child collections

Description

array **hw_getchildcoll** (int `$connection`, int `$objectID`)

Returns object ids. Each object ID belongs to a child collection of the collection with ID *`objectID`*. The function will not return child documents.

Parameters

`connection`
: The connection identifier.

`objectID`
: The object identifier.

Return Values

Returns an array of object ids.

See Also

- hw_children()
- hw_getchilddoccoll()

hw_GetChildCollObj

hw_GetChildCollObj -- Object records of child collections

Description

array **hw_getchildcollobj** (int `$connection`, int `$objectID`)

Returns object records. Each object records belongs to a child collection of the collection with ID *`objectID`*. The function will not return child documents.

Parameters

`connection`
: The connection identifier.

`objectID`
: The object identifier.

Return Values

Returns an array of object records.

See Also

- hw_childrenobj()
- hw_getchilddoccollobj()

hw_GetChildDocColl

hw_GetChildDocColl -- Object ids of child documents of collection

Description

array **hw_getchilddoccoll** (int `$connection`, int `$objectID`)

Returns array of object ids for child documents of a collection.

Parameters

`connection`
The connection identifier.

`objectID`
The object identifier.

Return Values

Returns an array of object ids.

See Also

- hw_children()
- hw_getchildcoll()

hw_GetChildDocCollObj

hw_GetChildDocCollObj -- Object records of child documents of collection

Description

array **hw_getchilddoccollobj** (int `$connection`, int `$objectID`)

Returns an array of object records for child documents of a collection.

Parameters

`connection`
: The connection identifier.

`objectID`
: The object identifier.

Return Values

Returns an array of object records.

See Also

- hw_childrenobj()
- hw_getchildcollobj()

hw_GetObject

hw_GetObject -- Object record

Description

mixed **hw_getobject** (int `$connection`, mixed `$objectID` [, string `$query`])

Gets an object record. If the second parameter is an array of integer the function will

Parameters

`connection`
The connection identifier.

`objectID`
The object identifier, or an array of identifiers.

`query`
The query string has the following syntax:

```
<expr> ::= "(" <expr> ")" |
          "!" &lt;expr> |          /* NOT */
          <expr> "||" <expr> |  /* OR */
          <expr> "&amp;&amp;" <expr> |  /* AND */
          <attribute> <operator> <value>

<attribute> ::= /* any attribute name (Title, Author, DocumentType ...) */

<operator> ::= "=" |    /* equal */
               "<" |    /* less than (string compare) */
               ">" |    /* greater than (string compare) */
               "~"      /* regular expression matching */
```

The query allows to further select certain objects from the list of given objects. Unlike the other query functions, this query may use not indexed attributes. How many object records are returned depends on the query and if access to the object is allowed.

Return Values

Returns the object record for the given object ID if the second parameter is an integer.

If the second parameter is an array of integer the function will return an array of object records. In such a case the last parameter is also evaluated which is a query string.

See Also

- hw_getandlock()

- hw_getobjectbyquery()

hw_GetObjectByQuery

hw_GetObjectByQuery -- Search object

Description

array **hw_getobjectbyquery** (int `$connection`, string `$query`, int `$max_hits`)

Searches for objects on the whole server and returns an array of object ids.

Parameters

`connection`
: The connection identifier.

`query`
: The query will only work with indexed attributes.

`max_hits`
: The maximum number of matches is limited to *`max_hits`*. If *`max_hits`* is set to -1 the maximum number of matches is unlimited.

Return Values

Returns an array of object ids.

See Also

- hw_getobjectbyqueryobj()

hw_GetObjectByQueryColl

hw_GetObjectByQueryColl -- Search object in collection

Description

array **hw_getobjectbyquerycoll** (int `$connection`, int `$objectID`, string `$query`, int `$max_hits`)

Searches for objects in collection with ID *`objectID`* and returns an array of object ids.

Parameters

`connection`
: The connection identifier.

`objectID`
: The collection id.

`query`
: The query will only work with indexed attributes.

`max_hits`
: The maximum number of matches is limited to *`max_hits`*. If *`max_hits`* is set to -1 the maximum number of matches is unlimited.

Return Values

Returns an array of object ids.

See Also

- hw_getobjectbyquerycollobj()

hw_GetObjectByQueryCollObj

hw_GetObjectByQueryCollObj -- Search object in collection

Description

array **hw_getobjectbyquerycollobj** (int `$connection`, int `$objectID`, string `$query`, int `$max_hits`)

Searches for objects in collection with ID *`objectID`* and returns an array of object records.

Parameters

`connection`
: The connection identifier.

`objectID`
: The collection id.

`query`
: The query will only work with indexed attributes.

`max_hits`
: The maximum number of matches is limited to *`max_hits`*. If *`max_hits`* is set to -1 the maximum number of matches is unlimited.

Return Values

Returns an array of object records.

See Also

- hw_getobjectbyquerycoll()

hw_GetObjectByQueryObj

hw_GetObjectByQueryObj -- Search object

Description

array **hw_getobjectbyqueryobj** (int `$connection`, string `$query`, int `$max_hits`)

Searches for objects on the whole server and returns an array of object records.

Parameters

`connection`
: The connection identifier.

`query`
: The query will only work with indexed attributes.

`max_hits`
: The maximum number of matches is limited to *`max_hits`*. If *`max_hits`* is set to -1 the maximum number of matches is unlimited.

Return Values

Returns an array of object records.

See Also

- hw_getobjectbyquery()

hw_GetParents

hw_GetParents -- Object ids of parents

Description

array **hw_getparents** (int `$connection`, int `$objectID`)

Returns the object identifiers of the parents of an object.

Parameters

`connection`
The connection identifier.

`objectID`
The object identifier.

Return Values

Returns an indexed array of object ids. Each object id belongs to a parent of the object with ID *`objectID`*.

hw_GetParentsObj

hw_GetParentsObj -- Object records of parents

Description

array **hw_getparentsobj** (int `$connection`, int `$objectID`)

Returns object records and statistical information about the object records. Each object record belongs to a parent of the object with ID *`objectID`*.

Parameters

`connection`
The connection identifier.

`objectID`
The object identifier.

Return Values

Returns an indexed array of object records plus an associated array with statistical information about the object records. The associated array is the last entry of the returned array.

hw_getrellink

hw_getrellink -- Get link from source to dest relative to rootid

Description

string **hw_getrellink** (int `$link`, int `$rootid`, int `$sourceid`, int `$destid`)

Warning
This function is currently not documented; only its argument list is available.

hw_GetRemote

hw_GetRemote -- Gets a remote document

Description

int **hw_getremote** (int `$connection`, int `$objectID`)

Gets a remote document.

Remote documents in Hyperwave notation are documents retrieved from an external source. Common remote documents are for example external web pages or queries in a database.

In order to be able to access external sources through remote documents Hyperwave introduces the HGI (Hyperwave Gateway Interface) which is similar to the CGI. Currently, only ftp, http-servers and some databases can be accessed by the HGI.

Calling hw_getremote() returns the document from the external source. If you want to use this function you should be very familiar with HGIs. You should also consider to use PHP instead of Hyperwave to access external sources. Adding database support by a Hyperwave gateway should be more difficult than doing it in PHP.

Parameters

`connection`
: The connection identifier.

`objectID`
: The object identifier.

Return Values

Returns a remote document.

See Also

- hw_getremotechildren()

hw_getremotechildren

hw_getremotechildren -- Gets children of remote document

Description

mixed **hw_getremotechildren** (int `$connection`, string `$object_record`)

Returns the children of a remote document. Children of a remote document are remote documents itself. This makes sense if a database query has to be narrowed and is explained in Hyperwave Programmers' Guide.

If you want to use this function you should be very familiar with HGIs. You should also consider to use PHP instead of Hyperwave to access external sources. Adding database support by a Hyperwave gateway should be more difficult than doing it in PHP.

Parameters

`connection`
The connection identifier.

`object_record`
The object record.

Return Values

If the number of children is 1 the function will return the document itself formatted by the Hyperwave Gateway Interface (HGI). If the number of children is greater than 1 it will return an array of object record with each maybe the input value for another call to hw_getremotechildren(). Those object records are virtual and do not exist in the Hyperwave server, therefore they do not have a valid object ID. How exactly such an object record looks like is up to the HGI.

See Also

- hw_getremote()

hw_GetSrcByDestObj

hw_GetSrcByDestObj -- Returns anchors pointing at object

Description

array **hw_getsrcbydestobj** (int `$connection`, int `$objectID`)

Gets the object records of all anchors pointing to the object with ID *`objectID`*.

Parameters

`connection`
: The connection identifier.

`objectID`
: The object identifier. The object can either be a document or an anchor of type destination.

Return Values

Returns an array of object records.

See Also

- hw_getanchors()

hw_GetText

hw_GetText -- Retrieve text document

Description

int **hw_gettext** (int `$connection`, int `$objectID` [, mixed `$rootID/prefix`])

Returns the document with object ID *`objectID`*. If the document has anchors which can be inserted, they will be inserted already.

This function will only work for pure text documents. It will not open a special data connection and therefore blocks the control connection during the transfer.

Parameters

`connection`
The connection identifier.

`objectID`
The object identifier.

`rootID/prefix`
The optional parameter *`rootID/prefix`* can be a string or an integer. If it is an integer it determines how links are inserted into the document. The default is 0 and will result in links that are constructed from the name of the link's destination object. This is useful for web applications. If a link points to an object with name 'internet_movie' the HTML link will be <A href="/internet_movie">. The actual location of the source and destination object in the document hierarchy is disregarded. You will have to set up your web browser, to rewrite that URL to for example '/my_script.php/internet_movie'. 'my_script.php' will have to evaluate $PATH_INFO and retrieve the document. All links will have the prefix '/my_script.php/'. If you do not want this you can set the optional parameter *`rootID/prefix`* to any prefix which is used instead. Is this case it has to be a string. If *`rootID/prefix`* is an integer and unequal to 0 the link is constructed from all the names starting at the object with the id *`rootID/prefix`* separated by a slash relative to the current object. If for example the above document 'internet_movie' is located at 'a-b-c-internet_movie' with '-' being the separator between hierarchy levels on the Hyperwave server and the source document is located at 'a-b-d-source' the resulting HTML link would be: <A HREF="../c/internet_movie">. This is useful if you want to download the whole server content onto disk and map the document hierarchy onto the file system.

Return Values

Returns the text document.

See Also

- hw_pipedocument()
- hw_free_document()
- hw_document_bodytag()
- hw_document_size()
- hw_output_document()

hw_getusername

hw_getusername -- Name of currently logged in user

Description

string **hw_getusername** (int `$connection`)

Returns the username of the connection.

Parameters

`connection`
The connection identifier.

Return Values

Returns the username as a string.

hw_Identify

hw_Identify -- Identifies as user

Description

string **hw_identify** (int `$link`, string `$username`, string `$password`)

Identifies as user with *`username`* and *`password`*. Identification is only valid for the current session. I do not think this function will be needed very often. In most cases it will be easier to identify with the opening of the connection.

Parameters

`link`
The connection identifier.

`username`
The username.

`password`
The password.

See Also

- hw_connect()

hw_InCollections

hw_InCollections -- Check if object ids in collections

Description

array **hw_incollections** (int `$connection`, array `$object_id_array`, array `$collection_id_array`, int `$return_collections`)

Checks whether a set of objects (documents or collections) specified by the *`object_id_array`* is part of the collections listed in *`collection_id_array`*.

Parameters

`connection`
: The connection identifier.

`object_id_array`
: An array of object ids.

`collection_id_array`
: An array of collection ids.

`return_collections`
: When set to 0, the subset of object ids that is part of the collections (i.e., the documents or collections that are children of one or more collections of collection ids or their subcollections, recursively) is returned as an array. When set to 1, the set of collections that have one or more objects of this subset as children are returned as an array. This option allows a client to, e.g., highlight the part of the collection hierarchy that contains the matches of a previous query, in a graphical overview.

Return Values

Returns an array of object ids.

hw_Info

hw_Info -- Info about connection

Description

string **hw_info** (int `$connection`)

Returns information about the current connection.

Parameters

`connection`
The connection identifier.

Return Values

The returned string has the following format: <Serverstring>, <Host>, <Port>, <Username>, <Port of Client>, <Byte swapping>

hw_InsColl

hw_InsColl -- Insert collection

Description

int **hw_inscoll** (int `$connection`, int `$objectID`, array `$object_array`)

Inserts a new collection with attributes as in *`object_array`* into collection with object ID *`objectID`*.

Parameters

`connection`
The connection identifier.

`objectID`

`object_array`

Return Values

hw_InsDoc

hw_InsDoc -- Insert document

Description

int **hw_insdoc** (resource `$connection`, int `$parentID`, string `$object_record` [, string `$text`])

Inserts a new document with attributes as in *`object_record`* into acollection.

If you want to insert a general document of any kind use hw_insertdocument() instead.

Parameters

`connection`
: The connection identifier.

`parentID`
: The collection id.

`object_record`
: Object attributes.

`text`
: If provided, this ascii text will be inserted too.

Return Values

See Also

- hw_insertdocument()
- hw_inscoll()

hw_insertanchors

hw_insertanchors -- Inserts only anchors into text

Description

bool **hw_insertanchors** (int `$hwdoc`, array `$anchorecs`, array `$dest` [, array `$urlprefixes`])

Warning
This function is currently not documented; only its argument list is available.

hw_InsertDocument

hw_InsertDocument -- Upload any document

Description

int **hw_insertdocument** (int `$connection`, int `$parent_id`, int `$hw_document`)

Uploads a document into the given collection.

The document has to be created before with hw_new_document(). Make sure that the object record of the new document contains at least the attributes: Type, DocumentType, Title and Name. Possibly you also want to set the MimeType.

Parameters

`connection`
The connection identifier.

`parent_id`
The collection identifier.

`hw_document`
The document identifier.

Return Values

The functions returns the object id of the new document or **FALSE**.

See Also

- hw_pipedocument()

hw_InsertObject

hw_InsertObject -- Inserts an object record

Description

int **hw_insertobject** (int `$connection`, string `$object_rec`, string `$parameter`)

Inserts an object into the server.

Note: If you want to insert an Anchor, the attribute Position has always been set either to a start/end value or to 'invisible'. Invisible positions are needed if the annotation has no corresponding link in the annotation text.

Parameters

`connection`
: The connection identifier.

`object_rec`
: The object can be any valid hyperwave object.

`parameter`
: See the HG-CSP documentation for a detailed information on how the parameters have to be.

See Also

- hw_pipedocument()
- hw_insertdocument()
- hw_insdoc()
- hw_inscoll()

hw_mapid

hw_mapid -- Maps global id on virtual local id

Description

int **hw_mapid** (int `$connection`, int `$server_id`, int `$object_id`)

Maps a global object id on any hyperwave server, even those you did not connect to with hw_connect(), onto a virtual object id.

This virtual object id can then be used as any other object id, e.g. to obtain the object record with hw_getobject().

Note: In order to use this function you will have to set the F_DISTRIBUTED flag, which can currently only be set at compile time in hg_comm.c. It is not set by default. Read the comment at the beginning of hg_comm.c

Parameters

`connection`
: The connection identifier.

`server_id`
: The server id is the first part of the global object id (GOid) of the object which is actually the IP number as an integer.

`object_id`
: The object identifier.

Return Values

Returns the virtual object id.

hw_Modifyobject

hw_Modifyobject -- Modifies object record

Description

bool **hw_modifyobject** (int `$connection`, int `$object_to_change`, array `$remove`, array `$add` [, int `$mode`])

This command allows to remove, add, or modify individual attributes of an object record. The object is specified by the Object ID *`object_to_change`*. In order to modify an attribute one will have to remove the old one and add a new one. hw_modifyobject() will always remove the attributes before it adds attributes unless the value of the attribute to remove is not a string or array.

The keys of both arrays are the attributes name. The value of each array element can either be an array, a string or anything else. If it is an array each attribute value is constructed by the key of each element plus a colon and the value of each element. If it is a string it is taken as the attribute value. An empty string will result in a complete removal of that attribute. If the value is neither a string nor an array but something else, e.g. an integer, no operation at all will be performed on the attribute. This is necessary if you want to add a completely new attribute not just a new value for an existing attribute. If the remove array contained an empty string for that attribute, the attribute would be tried to be removed which would fail since it doesn't exist. The following addition of a new value for that attribute would also fail. Setting the value for that attribute to e.g. 0 would not even try to remove it and the addition will work.

If you would like to change the attribute 'Name' with the current value 'books' into 'articles' you will have to create two arrays and call hw_modifyobject().

Example #1 - modifying an attribute

```
<?php
      // $connect is an existing connection to the Hyperwave server
      // $objid is the ID of the object to modify
      $remarr = array("Name" => "books");
      $addarr = array("Name" => "articles");
      $hw_modifyobject($connect, $objid, $remarr, $addarr);
?>
```

In order to delete/add a name=value pair from/to the object record just pass the remove/add array and set the last/third parameter to an empty array. If the attribute is the first one with that name to add, set attribute value in the remove array to an integer.

Example #2 - adding a completely new attribute

```
<?php
      // $connect is an existing connection to the Hyperwave server
      // $objid is the ID of the object to modify
      $remarr = array("Name" => 0);
      $addarr = array("Name" => "articles");
```

```
      $hw_modifyobject($connect, $objid, $remarr, $addarr);
?>
```

Note

Multilingual attributes, e.g. 'Title', can be modified in two ways. Either by providing the attributes value in its native form 'language':'title' or by providing an array with elements for each language as described above. The above example would than be:

Example #3 - modifying Title attribute

```
<?php
      $remarr = array("Title" => "en:Books");
      $addarr = array("Title" => "en:Articles");
      $hw_modifyobject($connect, $objid, $remarr, $addarr);
?>
```

or

Example #4 - modifying Title attribute

```
<?php
      $remarr = array("Title" => array("en" => "Books"));
      $addarr = array("Title" => array("en" => "Articles",
"ge"=>"Artikel"));
      $hw_modifyobject($connect, $objid, $remarr, $addarr);
?>
```

This removes the English title 'Books' and adds the English title 'Articles' and the German title 'Artikel'.

Example #5 - removing attribute

```
<?php
      $remarr = array("Title" => "");
      $addarr = array("Title" => "en:Articles");
      $hw_modifyobject($connect, $objid, $remarr, $addarr);
?>
```

Note

This will remove all attributes with the name 'Title' and adds a new 'Title' attribute. This comes in handy if you want to remove attributes recursively.

Note

If you need to delete all attributes with a certain name you will have to pass an empty string as the attribute value.

Note

Only the attributes 'Title', 'Description' and 'Keyword' will properly handle the language prefix. If those attributes don't carry a language prefix, the prefix 'xx' will be assigned.

Note

The 'Name' attribute is somewhat special. In some cases it cannot be complete removed. You will get an error message 'Change of base attribute' (not clear when this happens). Therefore you will always have to add a new Name first and than remove the old one.

Note

You may not surround this function by calls to hw_getandlock() and hw_unlock(). hw_modifyobject() does this internally.

Parameters

connection
: The connection identifier.

object_to_change
: The object to be changed.

remove
: An array of attributes to remove.

add
: An array of attributes to add.

mode
: The last parameter determines if the modification is performed recursively. 1 means recursive modification. If some of the objects cannot be modified they will be skipped without notice. hw_error() may not indicate an error though some of the objects could not be modified.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

hw_mv

hw_mv -- Moves objects

Description

int **hw_mv** (int `$connection`, array `$object_id_array`, int `$source_id`, int `$destination_id`)

Moves the specified objects from a collection to another.

Parameters

`connection`
: The connection identifier.

`object_id_array`
: An array of object ids.

`source_id`
: The source collection id.

`destination_id`
: The target collection id. If set to 0 the objects will be unlinked from the source collection. If this is the last instance of that object it will be deleted. If you want to delete all instances at once, use hw_deleteobject().

Return Values

Returns the number of moved objects.

See Also

- hw_cp()
- hw_deleteobject()

hw_New_Document

hw_New_Document -- Create new document

Description

int **hw_new_document** (string `$object_record`, string `$document_data`, int `$document_size`)

Returns a new Hyperwave document with the given document data and object record.

This function does not insert the document into the Hyperwave server.

Parameters

`object_record`
The object record.

`document_data`
The document data.

`document_size`
The document size. Must be the length of *`document_data`*.

Return Values

Returns the new Hyperwave document.

See Also

- hw_free_document()
- hw_document_size()
- hw_document_bodytag()
- hw_output_document()
- hw_insertdocument()

hw_objrec2array

hw_objrec2array -- Convert attributes from object record to object array

Description

array **hw_objrec2array** (string `$object_record` [, array `$format`])

Converts an *`object_record`* into an object array.

Parameters

`object_record`
: The object record.

`format`
: An associative array with the attribute name as its key and the value being one of *HW_ATTR_LANG* or *HW_ATTR_NONE*.

Return Values

Returns an array. The keys of the resulting array are the attributes names. Multi-value attributes like 'Title' in different languages form its own array. The keys of this array are the left part to the colon of the attribute value. This left part must be two characters long.

Other multi-value attributes without a prefix form an indexed array. If the optional parameter is missing the attributes 'Title', 'Description' and 'Keyword' are treated as language attributes and the attributes 'Group', 'Parent' and 'HtmlAttr' as non-prefixed multi-value attributes. By passing an array holding the type for each attribute you can alter this behaviour.

See Also

- hw_array2objrec()

hw_Output_Document

hw_Output_Document -- Prints hw_document

Description

bool **hw_output_document** (int `$hw_document`)

Prints the document without the BODY tag.

For backward compatibility, **hw_outputdocument()** is also accepted. This is deprecated, however.

Parameters

`hw_document`
The document identifier.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

hw_pConnect

hw_pConnect -- Make a persistent database connection

Description

int **hw_pconnect** (string `$host`, int `$port` [, string `$username`], string `$password`)

Opens a persistent connection to a Hyperwave server. You can have multiple persistent connections open at once.

Parameters

`host`
The server host name.

`port`
The server port number.

`username`
The Hyperwave user name. If ommited, no identification with the server will be done. It is similar to identify as user anonymous.

`password`
The password for *`username`*. Keep in mind, that the password is not encrypted.

Return Values

Returns a connection index on success, or **FALSE** if the connection could not be made.

See Also

- hw_connect()

hw_PipeDocument

hw_PipeDocument -- Retrieve any document

Description

int **hw_pipedocument** (int `$connection`, int `$objectID` [, array `$url_prefixes`])

Gets the Hyperwave document with the given object ID. If the document has anchors which can be inserted, they will have been inserted already.

The document will be transferred via a special data connection which does not block the control connection.

Parameters

`connection`
The connection identifier.

`objectID`
The object identifier.

`url_prefixes`

Return Values

Returns the Hyperwave document.

See Also

- hw_gettext()
- hw_free_document()
- hw_document_size()
- hw_document_bodytag()
- hw_output_document()

hw_Root

hw_Root -- Root object id

Description

int **hw_root** (void)

Returns the object ID of the hyperroot collection. Currently this is always 0. The child collection of the hyperroot is the root collection of the connected server.

Return Values

Returns 0.

hw_setlinkroot

hw_setlinkroot -- Set the id to which links are calculated

Description

int **hw_setlinkroot** (int `$link`, int `$rootid`)

Warning
This function is currently not documented; only its argument list is available.

hw_stat

hw_stat -- Returns status string

Description

string **hw_stat** (int `$link`)

Warning
This function is currently not documented; only its argument list is available.

hw_Unlock

hw_Unlock -- Unlock object

Description

bool **hw_unlock** (int `$connection`, int `$objectID`)

Unlocks a document, so other users regain access.

Parameters

`connection`
: The connection identifier.

`objectID`
: The document object identifier.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

See Also

- hw_getandlock()

hw_Who

hw_Who -- List of currently logged in users

Description

array **hw_who** (int `$connection`)

Gets the list of currently logged in users.

Parameters

`connection`
The connection identifier.

Return Values

Returns an array of users currently logged into the Hyperwave server. Each entry in this array is an array itself containing the elements id, name, system, onSinceDate, onSinceTime, TotalTime and self. 'self' is 1 if this entry belongs to the user who initiated the request.