

## Plexcel Developer's Guide

### Table of Contents

<b>Overview</b>	<b>3</b>
Authentication.....	3
Kerberos SPNEGO Single Sign-On (SSO).....	3
Explicit Logon with a Username and Password.....	3
Plexcel Authentication Functions.....	3
Delegation and Authentication with Other Servers.....	4
Advanced LDAP API.....	5
Accessing Accounts.....	5
Binding and Searching.....	5
Automatic Attribute Conversion.....	7
Security Features.....	7
Fast Access Control.....	7
Setting Passwords and Generating Keytabs.....	7
LDAP Signing and Encryption.....	8
Other Features.....	8
DNS Lookup Services.....	8
Easy Installation.....	8
Internationalization (i18n).....	8
<b>API Reference</b>	<b>9</b>
Common Functions.....	9
plexcel_new.....	9
Plexcel LDAP URL Examples.....	11
plexcel_status.....	12
plexcel_find_authorities_by_domain.....	13
The DNS SRV Lookup Logic.....	14
The PLEXCEL_AUTHORITY_CHKHOST Flag.....	14
plexcel_get_authority.....	15
plexcel_get_domain.....	16
plexcel_log.....	16
Security Functions.....	18
plexcel_preamble.....	18
plexcel_authenticate.....	20
Explicit Logon by Form.....	21
SSO Authentication.....	21
plexcel_sso.....	23
The plexcel_sso function calls die.....	23
plexcel_logon.....	24
Cached Credentials.....	24
plexcel_logoff.....	25
plexcel_accept_token.....	26
plexcel_is_member_of.....	27
Access Control Lists.....	28

plexcel_set_password.....	29
plexcel_change_password.....	29
plexcel_gen_service_keytab.....	30
Directory Functions.....	32
plexcel_search_objects.....	32
User Input in Filters.....	33
Use the Same Directory Server for Data Consistency.....	33
Specifying Times in Filters.....	33
plexcel_get_account.....	34
The PLEXCEL_SUPPLEMENTAL Constant.....	34
plexcel_add_object.....	36
Creating Accounts.....	36
plexcel_modify_object.....	37
plexcel_delete_object.....	39
plexcel_rename_object.....	40
plexcel_set_attrdefs.....	41
The type element.....	41
The flags element.....	41
The conv element.....	42
plexcel_get_attrdefs.....	43
plexcel_set_conv_attrdefs.....	45

# Overview

Plexcel is a PHP extension that provides a number of features ideal for integrating Linux and FreeBSD systems into Windows centric networks. No changes on the Windows side are necessary. Plexcel understands all of the necessary protocols used by Windows.

Plexcel provides location services, authentication, access control, password setting / changing, directory functions and much more. This section provides an overview of these features.

## Authentication

Plexcel supports both Kerberos Single Sign-On (SSO) and explicit username/password logon. Typically these features are used together to allow users to enter alternative credentials or to fail-over to a logon form if the client cannot perform SSO.

### **Kerberos SPNEGO Single Sign-On (SSO)**

Single Sign-On allows users to access resources such as websites or file servers without repeatedly entering their password. Users enter their credentials only once when they log onto their workstation. This feature is not just more convenient, it is more secure because it minimizes exposure of user credentials.

Kerberos is a network service used as a trusted third party to authenticate clients with servers. Clients obtain "tickets" from this Kerberos service which are used like passwords to access resources such as websites or file servers. One of the central functions of Microsoft Active Directory (herein abbreviated AD) is to provide this Kerberos service.

Plexcel fully supports Windows clients and servers using Kerberos. It can authenticate web clients and it can use the delegated credential supplied by the client, the HTTP service credential or credentials acquired with an explicit logon to initiate authentication with other Kerberos services.

Kerberos tickets issued by AD also encode within them the fully expanded list of groups the user is a member of. Windows servers extract this list and use it to perform access control checks. Just like Windows, Plexcel also extracts this list and uses it to perform access checks. This form of access checking is extremely efficient.

### **Explicit Logon with a Username and Password**

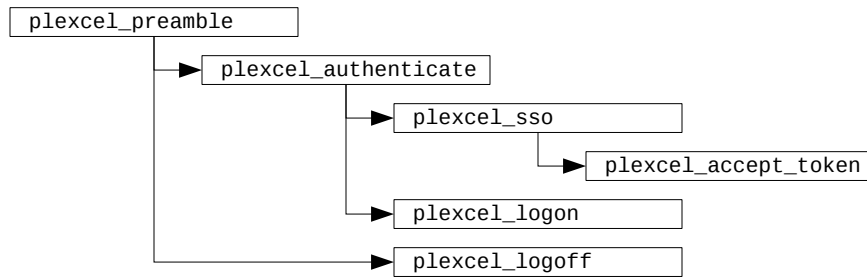
Single Sign-On cannot work if the user is not logged in using AD credentials. It may also be desirable to allow certain users to enter alternative credentials. For these scenarios, Plexcel can perform authentication with a username and password (e.g. via HTML form). All Plexcel functions work regardless of how the client authenticated<sup>1</sup>.

### **Plexcel Authentication Functions**

There are several Plexcel authentication functions to choose from. The below diagram shows that the `plexcel_preamble` function provides the most feature full, abstract interface whereas the others provide more direct control of authentication.

---

<sup>1</sup> Although the behavior of `plexcel_is_member_of` is different if BUILTIN or some Domain Local Groups are involved. See the `plexcel_is_member_of` documentation for details.



The *plexcel\_preamble* function combines all of the usual authentication and delegation related work (the “preamble”) into one simple easy to use function however it requires specific parameters be used and as such is the least flexible.

For applications that need slightly finer control, the *plexcel\_authenticate* function also abstracts SSO and explicit logon but unlike *plexcel\_preamble* it does not make any assumptions about HTTP request parameters or error handling.

The *plexcel\_sso* function abstracts the finer details of SSO authentication.

The *plexcel\_logon* and performs an explicit logon.

The *plexcel\_logoff* function is used to destroy the cached credential created by *plexcel\_logon*.

The *plexcel\_accept\_token* function is the base function used to consume and emit the authentication tokens used by the SPNEGO protocol.

If you are interfacing Plexcel with an existing application, it may be necessary to use *plexcel\_accept\_token* and *plexcel\_logon* / *plexcel\_logoff* directly. See the various Plexcel plugins available on the IOPLEX Software website for examples.

Note: For explicit control over all aspects of authentication in a PHP application using Plexcel, such as gracefully switching between SSO and a login form, it is strongly recommend that *plexcel\_accept\_token* and *plexcel\_logon* / *plexcel\_logoff* be used directly. The Joomla! and MediaWiki plugins use these functions directly. Feel free to copy and modify the plugin code to suit your needs.

## **Delegation and Authentication with Other Servers**

Web applications frequently need to authenticate with databases, AD servers, the Global Catalog, other LDAP servers, other web servers, etc. This is sometimes referred to as the “two hop problem”. Plexcel can use a client's delegated credential, the HTTP service account credential or credentials acquired by explicit logon to initiate authentication with other Kerberos services.

Consider the following minimal “whoami” script (error handling has been omitted for brevity). This script simply prints account information about the authenticated user:

```

<?php
session_start();
require_once('plexcel.php');
$px = plexcel_new(NULL, NULL);
plexcel_sso($px);
$acct = plexcel_get_account($px, NULL, NULL);
print_r($acct);
?>
  
```

Listing 1: A simple “whoami” script

When the *plexcel\_get\_account* function (described below) tries to authenticate with AD, it will locate and use the client's delegated credential. However, if the *plexcel\_sso* call is commented out, the HTTP service credential will

be used instead. Alternatively, *plexcel\_logon* may be used to acquire a credential for a specific account<sup>2</sup>. These are the three possible scenarios for acquiring a credential necessary to initiate authentication with other Kerberos services.

There is an additional requirement if the client authenticated using SSO and the function initiating Kerberos authentication is not a Plexcel function. When using functions such as *pgsql\_connect*, *curl\_open* or *ldap\_sasl\_bind*, it is necessary to set the `putenv_krb5ccname` option on the Plexcel context resource like:

```
$px = plexcel_new(NULL, array('putenv_krb5ccname' => TRUE));
```

*Listing 2: Setting the putenv\_krb5ccname option*

This instructs Plexcel to save the client's delegated credential and export the KRB5CCNAME environment variable thereby allowing non-Plexcel client functions to locate the appropriate credential and transparently authenticate with the target service. When the script exits, the credential is destroyed.

Note: If *plexcel\_logon* is used instead of SSO, it is not necessary to set the `putenv_krb5ccname` option. The *plexcel\_logon* function always uses a cached credential file and it always exports the KRB5CCNAME environment variable (but it unsets it when the script completes).

Note: Currently there is no way to use the HTTP service credential with non-Plexcel functions like *pgsql\_connect*, *curl\_open* or *ldap\_sasl\_bind*. In theory it should be possible to set the KRB5\_KTNAME environment variable to the `plexcel.keytab` path, but for a variety of reasons this does not work. The *plexcel\_logon* function must be used instead.

## Advanced LDAP API

---

### Accessing Accounts

One of Active Directory's primary functions is to manage accounts of all types. Plexcel has a special *plexcel\_get\_account* function which is optimized for accessing account objects using a wide variety of names. The following *plexcel\_get\_account* calls are all equivalent:

```
// Get account by CN
$acct = plexcel_get_account($px, 'Hans Müller', NULL);
// or by SAMAccountName
$acct = plexcel_get_account($px, 'hmuller', NULL);
// or by NetBIOSName\SAMAccountName
$acct = plexcel_get_account($px, 'EXAMPLE\hmuller', NULL);
// or by userPrincipalName (except for groups)
$acct = plexcel_get_account($px, 'hmuller@example.com', NULL);
// or by distinguishedName
$acct = plexcel_get_account($px, 'CN=Hans Müller,CN=Users,DC=example,DC=com', NULL);
```

*Listing 3: Account name forms accepted by plexcel\_get\_account.*

### Binding and Searching

Plexcel's LDAP API makes binding, searching, modifying and deleting objects in the directory easy. Consider the following Plexcel code fragment (error handling has been omitted for brevity):

```
$px = plexcel_new('ldap:///DefaultNamingContext??sub?(cn=Hans Müller)', NULL);
plexcel_set_conv_attrdefs($px); /* for automatic attribute conversions */
$objs = plexcel_search_objects($px, NULL);
echo '<pre>';
print_r($objs[0]);
```

---

<sup>2</sup> This is particularly useful if your directory servers do not permit simple binds.

```
echo '</pre>';
```

Listing 4: Searching the directory for an account with the help of an advanced binding string

This script searches for accounts with a cn attribute of 'Hans Müller' and prints the first (and presumably only) element with *print\_r*. Notice that the LDAP URL binding string eliminates a number of otherwise tedious tasks:

- The directory server hostname is not required. A server is located automatically using DNS SRV queries.
- A specific base DN was not required. There are several special base DN names like 'DefaultNamingContext' that may be used in LDAP URLs.
- Default search parameters can be supplied in the LDAP URL.

See the *plexcel\_new* documentation for LDAP URL details and example binding strings.

Now consider the following sample output of the above script:

```
Array
(
    [objectClass] => Array
        (
            [0] => user
            [1] => organizationalPerson
            [2] => person
            [3] => top
        )
    [cn] => Hans Müller
    [sn] => Müller
    [c] => DE
    [description] => Array
        (
            [0] => This is Hans Müller's description.
        )
    [telephoneNumber] => 0611/11111 0
    [givenName] => Hans
    [distinguishedName] => CN=Hans Müller,CN=Users,DC=example,DC=com
    [whenCreated] => 1158682853000
    [whenChanged] => 1172889174000
    [displayName] => Hans Müller
    [info] => This is the Notes field in the Telephones tab for Hans Müller.
    [memberOf] => Array
        (
            [0] => CN=Managers,CN=Users,DC=example,DC=com
            [1] => CN=Publishing,CN=Users,DC=example,DC=com
            [2] => CN=Private Client,CN=Users,DC=example,DC=com
        )
    [userAccountControl] => 512
    [homeDirectory] => \\ad1\hmuller
    [homeDrive] => H:
    [lastLogon] => 1173222197833
    [objectSid] => S-1-5-21-4133388617-793952518-2001621813-1396
    [sAMAccountName] => hmuller
    [userPrincipalName] => hmuller@example.com
    [mail] => hans@example.com
    [homePhone] => 0611/22 33 44
    [mobile] => 0611/33 44 5-55
    ...
)
```

The above output illustrates several noteworthy features of Plexcel.

- Single valued elements are directly indexed by name. PHP's raw LDAP API treats all attributes as multivalued. For example, the sn element would have to be addressed using the expression `$acct['sn'][0]` with an extra array that does nothing but waste memory. Every Plexcel context maintains a table of attribute definitions. It knows which attributes are single valued.
- Strings are automatically converted to the locale encoding. With PHP's raw LDAP API, the umlaut in "Müller" would have to be converted from the directory encoding of UTF-8 to the locale encoding (e.g. `de_DE.ISO-8859-1@euro`). This is not necessary with Plexcel's API. Plexcel automatically converts strings to the default locale encoding Apache runs under.
- Values can be automatically converted to more useful forms. For example, all time values can be normalized to milliseconds since 1970. More on this below.

## Automatic Attribute Conversion

Plexcel allows conversion flags to be set with attribute definitions that will cause attribute values to be automatically converted to more useful values. For example, Active Directory represents time attributes as either UTC time strings or as nanoseconds since 1601 values. Considerable work is required to manually convert these values into human readable dates or values that can be used in date logic. With automatic attribute conversions turned on, time values will be automatically normalized into milliseconds since 1970 values which are ideal for use within PHP. These conversions are also applied when setting values as well. See the `plexcel_set_attrdefs` API documentation for a complete list of possible conversions.

## Security Features

---

### Fast Access Control

Fine grained security for advanced content management systems is easy with Plexcel. Plexcel provides script level access control with a `plexcel_is_member_of` function. It accepts a variety of group name forms and returns TRUE or FALSE to indicate that the user is or is not in the specified group.

```
<?php
    if (plexcel_is_member_of($px, 'Managers')) {
        // content for the Managers group here ...
    }
?>
```

*Listing 5: A simple `plexcel_is_member_of` example*

The above code fragment returns TRUE only if the currently authenticated user is a member of the 'Managers' group in AD. The group name may be a CN or SAM account name, a domain qualified SAM account name (e.g. 'EXAMPLE\Managers') or a full DN. The name may also refer to a specific user.

These `plexcel_is_member_of` function calls are very fast. The list of groups a user is in is extracted from their Kerberos ticket. There is no network communication necessary to get the group list. This has significant performance implications because after the group names used throughout your scripts have been resolved, no additional network communication is required to perform access checks. The result is that, after the first few visitors, the `plexcel_is_member_of` calls become effectively instantaneous. This permits chaining and looping to create Access Control Lists (ACLs).

### Setting Passwords and Generating Keytabs

Plexcel provides a `plexcel_set_password` function that can be used to easily set account passwords. Plexcel can also generate keytab files. See the `plexcel_set_password` and `plexcel_gen_service_keytab` API documentation for details.

## LDAP Signing and Encryption

Plexcel supports signing and encrypting all LDAP communication. LDAP signing and encryption is enabled by setting `plexcel.ldap.signing = 0n` and `plexcel.ldap.encryption = 0n` respectively. Signing and encryption may also be specified on a per-context basis using the `signing` and `encryption` options of the `plexcel_new` function. See the `plexcel_new` function API documentation for details.

## Other Features

---

### DNS Lookup Services

Plexcel provides a `plexcel_find_authorities_by_domain` function that may be used to list domain controllers. This is required to lookup the Global Catalog. It is also required when querying non-replicated attributes. For example, because the `lastLogon` attribute is not replicated between domain controllers, to determine when a user last logged onto their account, a script must iterate through all AD servers and find the greatest `lastLogon` value. See the `plexcel_find_authorities_by_domain` API documentation for an example of this.

### Easy Installation

Because Plexcel can natively create accounts and set passwords, a Plexcel Setup script is provided that allows the operator to logon to AD and can create HTTP service accounts. This eliminates steps that would otherwise have to be performed using Windows utilities. In particular, running `ktpass.exe` is not necessary. An experienced operator with the necessary privileges can install Plexcel and create the required HTTP service account in only a few minutes.

### Internationalization (i18n)

All text accepted or returned by Plexcel functions will be encoded using the default language encoding that the web server runs under. The programmer does not need to be concerned with character conversions. See the Plexcel Operator's Manual for details.

# API Reference

## Common Functions

---

### plexcel\_new

#### Synopsis

```
resource plexcel_new(string $bindstr, array $options)
```

#### Description

The *plexcel\_new* function creates a new Plexcel context for use with other Plexcel functions. A Plexcel context resource tracks binding state and context specific options with respect to authentication, LDAP routines and other Plexcel features.

The *\$bindstr* parameter specifies a target domain controller and optionally default search parameters. The value may be one of the following.

<b>\$bindstr Value</b>	<b>Description</b>
NULL or empty string	A value of NULL or an empty string indicates that the Plexcel context should be bound to a default server located using DNS SRV queries.
domain name	A domain name indicates that the Plexcel context should be bound to a default server in the specified domain located using DNS SRV queries.
hostname	A hostname indicates that the Plexcel context should be bound to the specific server provided that it is validated using DNS SRV queries.
LDAP URL	An LDAP URL can specify the domain name or specific hostname and default search parameters. See the Plexcel LDAP URLs section below.

Note: The domain name or domain of a hostname or server component of a Plexcel LDAP URL is resolved by querying DNS using SRV queries for `_ldap._tcp.dc._msdcs.<domain name>`, `_kerberos._tcp.dc._msdcs.<domain name>` and `_kerberos._udp.dc._msdcs.<domain name>` where `<domain name>` is either the name supplied, successively shorter domain name suffixes of the name supplied or, if no name was supplied, the realm of the HTTP service principal in the local credential file (the realm of the first entry in the `plexcel.keytab` file).

The \$options parameter of the plexcel\_new function is an array that may be NULL or an empty array to indicate that no options are desired or any of the following options.

Array Key	Value Description
putenv_krb5ccname	If set to TRUE, the Plexcel authentication functions will save the users delegated credential (if supplied) into a file in the Plexcel temporary directory and set the KRB5CCNAME environment variable to the said file's path. This allows non-Plexcel clients that support Kerberos such as <i>ldap_sasl_bind</i> , <i>pgsql_connect</i> and <i>curl_open</i> to initiate authentication with other servers using the client's delegated credential.
signing	If set to TRUE, all LDAP communication will use signing.  LDAP signing may also be turned on globally using the <code>plexcel.ldap.signing</code> INI option.  If LDAP signing is already turned on using the <code>plexcel.ldap.signing</code> INI option, setting this option has no effect (e.g. setting it to FALSE will not turn off signing).  If your directory server is set to <i>Require signing</i> under <i>Domain Controller Security Policy &gt; Local Policies &gt; Security Options &gt; Domain controller: LDAP server signing requirements</i> , signing must be used or an <code>ldap_sasl_bind_s: Strong(er) authentication required</code> error will occur.
encryption	If set to true, all LDAP communication will use encryption.  LDAP encryption may also be turned on globally using the <code>plexcel.ldap.encryption</code> INI option.  If LDAP encryption is already turned on using the <code>plexcel.ldap.encryption</code> INI option, setting this option has no effect (e.g. setting it to FALSE will not turn off encryption).

## Plexcel LDAP URLs

The `plexcel_new $bindstr` parameter may be in the form of a Plexcel LDAP URL. The syntax of these RFC 2255 style URLs is roughly as follows:

```
ldap://[<server>[:port]]/[<base>][?<attrs>[?<scope>[?<filter>]]]
```

The server component of a Plexcel LDAP URL can be an empty string, a domain name or a hostname and is resolved using DNS SRV queries.

The base component of a Plexcel LDAP URL can be an empty string to indicate the RootDSE, a full DN or one of the following alternative base component strings.

Alt. Base Component String	Description
RootDSE	Bind the RootDSE. Same as an empty string.
DefaultNamingContext	Bind the default naming context as defined in the RootDSE.
Users	Use a WKGUID bind on the Users GUID A9D1CA15768811D1AED00C04FD8D5CD.
Computers	Use a WKGUID bind on the Computers GUID AA312825768811D1AED00C04FD8D5CD.
System	Use a WKGUID bind on the System GUID AB1D30F3768811D1AED00C04FD8D5CD.
Domain Controllers	Use a WKGUID bind on the Domain Controllers GUID

A361B2FFFFD211D1AA4B00C04FD7D83A.

Infrastructure	Use a WKGUID bind on the Infrastructure GUID 2FBAC1870ADE11D297C400C04FD8D5CD.
Deleted Objects	Use a WKGUID bind on the Deleted Objects GUID 18E2EA80684F11D2B9AA00C04F79F805.
Lost and Found	Use a WKGUID bind on the Lost and Found GUID AB8153B7768811D1AED00C04FD8D5CD.

## Plexcel LDAP URL Examples

`ldap:///` - The default directory server will be located using DNS SRV queries.

`ldap:///RootDSE` - Same as '`ldap:///`'

`ldap://example.com/` - The default directory server in the example.com domain will be located using DNS SRV queries.

`ldap://ad1.example.com/` - The specified server is used provided that DNS SRV queries can verify that it is listed in the example.com domain.

`ldap:///CN=Hans Müller,DC=Users,DC=example,DC=com` - This binds a specific user object on the local directory server.

`ldap:///CN=Users,DC=example,DC=com` - This specifies the Users container as the base DN.

`ldap:///Users` - This also specifies the Users container but it uses WKGUID binding with the Users GUID as the base DN.

`ldap:///CN=Hans Müller,DC=Users,DC=example,DC=com?cn,telephoneNumber,homePhone` - A search with no explicit parameters using this binding would select the cn and some phone oriented information for the specified user from the local directory server.

`ldap:///Users??sub?(objectClass=user)` - A search with no explicit parameters using this binding would select all attributes of all users (but not groups) in the default Users container on the local directory server.

`ldap://ad1.example.com/Users??sub?(lastLogon>=128175527431758394)` - This would return all users that have logged onto the specified AD server after March 5th, 2007.

Note: SID binding does not work with Windows 2000.

## Returns

The `plexcel_new` function returns the new Plexcel context resource or FALSE to indicate an error has occurred in which case `plexcel_status` should be queried (with a NULL context).

## Example

The following PHP fragment illustrates how to use a Plexcel LDAP URL with `plexcel_new` to simplify searching the directory.

```
<?php
// Bind the defaultNamingContext on the local directory server
$px = plexcel_new('ldap:///DefaultNamingContext', NULL);
if ($px == FALSE)
    die('<pre>' . plexcel_status(NULL) . '</pre>');
} else {
```

```

$params = array( // no need to specify base
    'scope' => 'sub',
    'filter' => '(&(objectClass=user)(logonCount=0))'
);
$objs = plexcel_search_objects($px, $params);
...

```

Listing 6: Using DefaultNamingContext in an LDAP URL with *plexcel\_new*

## See also

*plexcel\_preamble*

## plexcel\_status

### Synopsis

```
string plexcel_status(resource $px, [string $status])
```

### Description

The *plexcel\_status* function returns a string indicating the current status of the specified Plexcel context resource.

The *\$px* parameter is the Plexcel context resource to be queried or NULL if no such context is appropriate (e.g. the *plexcel\_new* function returned FALSE).

If the optional *\$status* parameter is supplied, the status string for *\$px* is set to *\$status*.

The strings returned by *plexcel\_status* are, under normal circumstances, predefined constants with values equal to that of the symbol name (e.g. the constant PLEXCEL\_NO\_CREDS is defined as a string 'PLEXCEL\_NO\_CREDS'). These constants may be logically tested in a conditional expression. The following is a list of only some of these constants (the full list includes many more PLEXCEL\_LDAP\_\* status constants).

```

PLEXCEL_SUCCESS
PLEXCEL_FAILURE
PLEXCEL_CONTINUE_NEEDED
PLEXCEL_NO_CREDS
PLEXCEL_LOGON_FAILED
PLEXCEL_PRINCIPAL_UNKNOWN
PLEXCEL_LDAP_SUCCESS
PLEXCEL_LDAP_OPERATIONS_ERROR
PLEXCEL_LDAP_PROTOCOL_ERROR
PLEXCEL_LDAP_TIMELIMIT_EXCEEDED
...

```

Listing 7: A small subset of *plexcel\_status* codes.

If a *plexcel* function is being use inappropriately or if an exceptional error occurs, the *plexcel\_status* function may also return detailed failure information. Below is an example of this type of status string.

```

src/creds.c:322:psec_logon_with_password: KRB5: Clients credentials have been revoked
src/set_password.c:204:psec_gen_service_keytab:
src/plexcel.c:1416:plexcel_gen_service_keytab:

```

Listing 8: A sample exception returned by *plexcel\_status*

## Returns

The `plexcel_status` function returns a string constant or diagnostic string regarding the status of the specified Plexcel context resource.

## Example

The following PHP fragment illustrates how one might handle a failure of the `plexcel_authenticate` function.

```
if (plexcel_authenticate($px, session_id(), $options) == FALSE) {
    if (plexcel_status($px) == PLEXCEL_NO_CREDS) {
        // creds gone, fallback to logon form
    } else if (plexcel_status($px) == PLEXCEL_PRINCIPAL_UNKNOWN) {
        $username = plexcel_get_param('p_username');
        $err = "<p/>Principal unknown: $username";
    } else if (plexcel_status($px) == PLEXCEL_LOGON_FAILED) {
        $err = '<p/>Logon failed (e.g. bad password)';
    } else {
        $err = '<p/>Plexcel error: <pre>' . plexcel_status($px) . '</pre>';
    }
    ...
}
```

Listing 9: A `plexcel_authenticate` example with error handling

## plexcel\_find\_authorities\_by\_domain

### Synopsis

```
array plexcel_find_authorities_by_domain(string $name, int $nn, int $flags)
```

### Description

The `plexcel_find_authorities_by_domain` function performs DNS SRV queries for specific services and returns an array of server hostnames.

The `$name` parameter is the name to be queried. The DNS SRV lookup logic tries successively shorter name suffixes until the name is resolved. See *The DNS SRV Lookup Logic* section below for details.

If the `$name` parameter is NULL, the default domain will be queried. The default domain is the domain of the HTTP service account.

The `$nn` parameter currently is not used and should be 0.

The `$flags` parameter specifies the service of interest using the below constants. If multiple flags are bitwise OR'd together, servers providing **any** of the specified services will be returned.

Flag	DNS SRV Lookup Names
PLEXCEL_AUTHORITY_KERBEROS	<code>_kerberos._tcp.dc._msdcs.&lt;name&gt;</code> <code>_kerberos._udp.dc._msdcs.&lt;name&gt;</code>
PLEXCEL_AUTHORITY_KPASSWD	<code>kpasswd._tcp.&lt;name&gt;</code> <code>_kpasswd._udp.&lt;name&gt;</code>
PLEXCEL_AUTHORITY_LDAP	<code>_ldap._tcp.dc._msdcs.&lt;name&gt;</code>
PLEXCEL_AUTHORITY_GC	<code>_gc._tcp.&lt;name&gt;</code>
PLEXCEL_AUTHORITY_CHKHOST	This flag is special, see below.

## The DNS SRV Lookup Logic

The DNS SRV lookup logic tries successively shorter name suffixes until the name is successfully resolved. For example, if there is one Global Catalog server `gc1.example.com` and the below code is used, the algorithm will try four separate DNS queries:

```
// if the GC for h1.nyc.us.example.com is gc1.example.com,
// the algorithm tries the following DNS SRV lookups
// _gc._tcp.h1.nyc.us.example.com (not found)
//   _gc._tcp.nyc.us.example.com (not found)
//     _gc._tcp.us.example.com (not found)
//       _gc._tcp.example.com (success -> gc1.example.com)
$name = 'h1.nyc.us.example.com';
$gcs = plexcel_find_authorities_by_domain($name, PLEXCEL_AUTHORITY_GC);
```

*Listing 10: A `plexcel_find_authorities_by_domain` example with lookup algorithm comments*

## The PLEXCEL\_AUTHORITY\_CHKHOST Flag

This `PLEXCEL_AUTHORITY_CHKHOST` flag indicates that `$name` is a hostname that should be validated by performing the normal DNS SRV lookup logic but, at each step, checking to see if the `$name` is in the list of query results. This algorithm verifies that `$name` does or does not provide the desired service. If `$name` does provide the service, only `$name` is returned. If `$name` was not in a list or no query was successful, `FALSE` is returned.

This flag is used internally by the LDAP URL binding code to permit it to accept both a hostname and domain name. Use this flag to give your applications the same behavior.

## Returns

The `plexcel_find_authorities_by_domain` function returns an array of server hostnames offering the specified service or `FALSE` to indicate that no servers were found.

## Example

The following PHP script determined when the user `bcarter` last logged on by queries each domain controller for the user's `lastLogon` attribute (the `lastLogon` attribute is not replicated and therefore all domain controllers must be queried separately).

```
<?php
$name = 'bcarter';
$host = '';
$lastLogon = 0;

$dcs = plexcel_find_authorities_by_domain(NULL,
    0,
    PLEXCEL_AUTHORITY_LDAP);
$attrdefs = array(
    'lastLogon' => array(
        'type' => PLEXCEL_TYPE_INT64,
        'flags' => PLEXCEL_SINGLE_VALUED,
        'conv' => PLEXCEL_CONV_TIME1970M_X_TIME1601));
foreach ($dcs as $dc) {
    $px = plexcel_new($dc, NULL);
    // normalize lastLogon to milliseconds since 1970 for use with date()
```

```

    plexcel_set_attrdefs($px, $attrdefs);
    $acct = plexcel_get_account($px, $name, array('lastLogon'));
    if (isset($acct['lastLogon']) && $acct['lastLogon'] > $lastLogon) {
        $host = $dc;
        $lastLogon = $acct['lastLogon'];
    }
}
echo "$name last logged onto $host on " . date('M j, Y g:i:s A', $lastLogon / 1000.0);
?>

```

Listing 11: Properly determining the lastLogon time with `plexcel_find_authorities_by_domain`

## plexcel\_get\_authority

### Synopsis

```
string plexcel_get_authority(resource $px, bool $user = FALSE)
```

### Description

The `plexcel_get_authority` function returns the hostname of the server with which the supplied `$px` resource is bound. If the `$user` parameter is `FALSE`, the hostname of the directory binding is returned. If the `$user` value is `TRUE`, the hostname of the server that is an authority for the user is returned.

For example, if `plexcel_new(NULL, NULL)` is used, `plexcel_get_authority` will return the hostname of the specific domain controller chosen by Plexcel. HTTP scripts can, and should, use this function to retrieve the directory binding and either store it in the user's session or passed to the client so that it may be used with `plexcel_new` in subsequent request to ensure that the same server is used. This behavior is sometimes referred to as "server stickyness". Otherwise, setting data on one server may not be visible in a subsequent request because a different authority was chosen and the data has not replicated between them.

### Returns

The `plexcel_get_authority` function a hostname string or `FALSE` to indicate that an error has occurred in which case `plexcel_status` should be consulted.

### Example

The following PHP script prints the hostname of the `$px` resource's domain controller.

```

<?php
$px = plexcel_new(NULL, NULL);
if (!$px)
    die('<pre>' . plexcel_status(NULL) . '</pre>');

$authority = plexcel_get_authority($px, FALSE);
if ($authority === FALSE)
    die('<pre>' . plexcel_status($px) . '</pre>');

echo $authority;

```

Listing 12: Retrieving the plexcel resource's domain controller hostname with `plexcel_get_authority`.

## plexcel\_get\_domain

### Synopsis

```
array plexcel_get_domain(resource $px, string $dname)
```

### Description

The *plexcel\_get\_domain* function retrieves information about the named domain including its `dnsRoot`, `nETBIOSName` and `objectSid`. The `$dname` parameter may be `NULL` to indicate that the default domain of the supplied `$px` context should be returned.

This function is particularly useful for canonicalizing usernames before calling *plexcel\_logon* (e.g. convert `EXAMPLE\username` to `username@example.com` as required by *plexcel\_logon*).

Note: Currently this function cannot be used with NetBIOS domain names unless the corresponding DNS name is resolved first. Plexcel currently does not support NetBIOS name lookups. See Issue 11 in the Plexcel Operator's Manual.

### Returns

The *plexcel\_get\_domain* function returns an array of attributes or `FALSE` to indicate that an error has occurred in which case *plexcel\_status* should be consulted.

### Example

The following PHP script prints information about the specified domain.

```
<?php
$dname = 'example.com';

$px = plexcel_new(null, null);
if (!$px)
    die('<pre>' . plexcel_status(NULL) . '</pre>');

$domain = plexcel_get_domain($px, $dname);
if ($domain === FALSE)
    die('<pre>' . plexcel_status($px) . '</pre>');

echo '<pre>';
print_r($domain);
echo '</pre>';
```

Listing 13: Retrieving the `NetBIOSName`, `dnsRoot` and `objectSid` of a domain with *plexcel\_get\_domain*.

## plexcel\_log

### Synopsis

```
bool plexcel_log(int $log_level, string $msg)
```

### Description

The *plexcel\_log* function writes a message to the Plexcel log file provided that the supplied log level and internal

log level match sufficiently.

The `$log_level` parameter is an integer representing the level and category of the message. This value is compared to the corresponding `plexcel.log.level` PHP INI property.

A `$log_level` of 1 is the simplest way to write a message to the Plexcel log file. For more advanced logging, the category may also be specified as described below.

The lower 8 bits of the `$log_level` is the level. The upper 24 bits is the category. If the category of either the `$log_level` value or the internal `plexcel.log.level` value is 0, then only the priority is considered.

The following table lists category values and their meaning.

Category	Value	Description
SIG	0x00001000	Signals
IO	0x00002000	Socket I/O
NET	0x00004000	Networking
MEM	0x00008000	Memory Management
SEC	0x00010000	Security (not from routines internal to Kerberos libraries)
LOCK	0x00020000	Locking
SEGV	0x00040000	Segfault handler (requires gdb)
DNS	0x00080000	DNS querying (not from routines internal to Kerberos or LDAP libraries)
HTTP	0x00100000	Currently not used by Plexcel functions
SMB	0x00200000	SMB/CIFS communication
DCE	0x00400000	DCE/RPC communication
LDAP	0x00800000	LDAP communication (not from routines internal to LDAP library)

The `$msg` parameter is the message to be written to the log.

## Returns

The `plexcel_log` function returns FALSE if an error occurs. Otherwise, TRUE is returned.

## Example

The following PHP fragment writes the `QUERY_STRING` of a request to the Plexcel log file provided that the `plexcel.log.level` is less than or equal to 1.

```
plexcel_log(1, 'test.php request params: ' . $_SERVER['QUERY_STRING']);
```

*Listing 14: Logging request parameters to the Plexcel log file*

## Security Functions

---

### plexcel\_preamble

#### Synopsis

```
session_start();
require_once('plexcel.php');

array plexcel_preamble(array $options=NULL)
```

#### Description

The *plexcel\_preamble* function abstracts all of the normal authentication related work (the “preamble”) into a single easy-to-use function. It allows for a specific domain controller to be selected and abstracts error handling, parameter handling and client logoff.

It is recommended that scripts requiring general purpose authentication use this function or create their own customized version of it.

Note: Because this function may call *plexcel\_sso* and because *plexcel\_sso* needs to be called twice for each authentication, this function should be invoked before any code that is not absolutely necessary.

Note: It is strongly recommended that <https://> be used when submitting client credentials to the server in plain text.

The optional `$options` parameter specifies the following possible options.

Option	Description
base	The LDAP URL base DN used as a the default base with <i>plexcel_search_objects</i> (e.g. <code>DefaultNamingContext, OU=Asia, DC=example, DC=com</code> , etc).
authority	A domain or specific domain controller (e.g. <code>example.com, ad1.example.com</code> , etc).
auth_type	One of <code>PLEXCEL_AUTH_SSO</code> , <code>PLEXCEL_AUTH_LOGON</code> or <code>PLEXCEL_AUTH_SSO_LOGON</code> as described in the <i>plexcel_authenticate</i> documentation.
logonurl	The URL of the logon form if it is different from the current script. See the <i>plexcel_authenticate</i> documentation.
putenv_krb5ccname	If set to <code>TRUE</code> , this allows non-Plexcel clients that support Kerberos such as <i>ldap_sasl_bind</i> , <i>pgsql_connect</i> and <i>curl_open</i> to initiate authentication with other servers using the client's delegated credential. See the <i>plexcel_new</i> documentation for more information.

This function requires that specific HTTP request parameters are used. The following table describes these parameters in detail.

Request Parameter	Description
p_action	This parameter controls the action of the <i>plexcel_preamble</i> function. There are currently three possible actions – <code>default</code> , <code>logoff</code> , and <code>change_authority</code> . If no <code>p_action</code> parameter is sent, the action will be <code>default</code> . The <code>default</code> action is to authenticate the client with <i>plexcel_authenticate</i> . The <code>logoff</code> action logs off the client with

*plexcel\_logoff*. The *change\_authority* action causes *plexcel\_preamble* to do nothing but return NULL for the authority element (see the *Returns* section below).

---

<i>p_authority</i>	This parameter specifies a target domain in which to find a domain controller or the FQDN of a specific domain controller to which the Plexcel context resource should be bound. The script must ensure that this parameter is resent with each request (e.g. with a hidden form element). If it is not, and a different domain controller is selected, querying objects in the directory may not yield consistent results if attributes are not been replicated or are in the process of being replicated.  If no <i>p_authority</i> parameter is sent, an authority will be located using <i>plexcel_find_authorities_by_domain</i> and returned in the authority element (see the <i>Returns</i> section below).
<i>p_username</i>	If this parameter is sent, <i>plexcel_preamble</i> will attempt an explicit logon ( <i>plexcel_preamble</i> calls <i>plexcel_authenticate</i> calls <i>plexcel_logon</i> ).  If no <i>p_username</i> parameter is sent, <i>plexcel_preamble</i> will attempt SSO authentication ( <i>plexcel_preamble</i> calls <i>plexcel_authenticate</i> calls <i>plexcel_sso</i> ).
<i>p_password</i>	This parameter specifies the password to be used when an explicit logon is performed.

---

## Returns

The *plexcel\_preamble* function returns an array containing several elements. These elements, which may be accessed by index and by name, are described further in the following table.

By Index	By Name	Description
0	<i>authority</i>	The domain controller to which the Plexcel context resource ( <i>px</i> ) is bound.  If the <i>p_action</i> request parameter was <i>change_authority</i> , this element will be NULL. This allows the script to allow the user to specify an explicit authority (see the <i>plexcel/examples/preamble.php</i> example).
1	<i>bindstr</i>	The binding string used to construct the Plexcel context resource ( <i>px</i> ).
2	<i>px</i>	The Plexcel context resource. The <i>authority</i> element is the domain controller to which the context is bound. The <i>bindstr</i> element is the binding string used to construct the context.
3	<i>err</i>	The error string if an error occurred. If no error has occurred, this value is NULL.
4	<i>action</i>	The action performed (see description of <i>p_action</i> ) unless the action was <i>logoff</i> in which case the action is changed to <i>default</i> .
5	<i>is_authenticated</i>	Set to TRUE if the client was successfully authenticated. Otherwise, this element is set to FALSE.

---

Tip: Use PHP's *list* construct to dereference this array directly into variables for use within scripts.

```
$preamble = plexcel_preamble();  
list($authority, $bindstr, $px, $err, $action, $is_authenticated) = $preamble;
```

Listing 15: Using PHP's *list* construct to dereference the *\$preamble* array into variables

## Example

The following is a highly simplified version of the `plexcel/examples/preamble.php` script. Notice that this example is significantly simpler than the `plexcel_authenticate` example and yet it provides the same features.

```
<?php
session_start();
require_once('plexcel.php');

$ preamble = plexcel_preamble();
list($authority, $bindstr, $px, $err, $action, $is_authenticated) = $preamble;

echo "<form name='f' method='POST'>";
echo "<input type='hidden' name='p_action' value='$action'/>";
echo "<input type='hidden' name='p_authority' value='$authority'/>";

$username = plexcel_get_param('p_username');
if ($err)
    echo $err;

if ($is_authenticated == FALSE) {
    echo "<p>Username: <input type='text' name='p_username' value='" .
        htmlspecialchars($username) . "'/> must be UPN<br/>";
    echo "Password: <input type='password' name='p_password'/>";
} else {
    echo "<input type='hidden' name='p_username' value='" .
        htmlspecialchars($username) . "'/>";
    echo "<a
href=\"javascript:document.f.p_action.value='logoff';document.f.submit();\">Logoff</a>
";
    echo '<p>You are successfully logged on.';
}
echo "<p/><input type='submit'/>";
echo "</form>";
?>
```

Listing 16: A complete `plexcel_preamble` example with logon form

## See also

`plexcel_authenticate` | `plexcel_sso` | `plexcel_logon` | `plexcel_logoff`

## plexcel\_authenticate

### Synopsis

```
session_start();
require_once('plexcel.php');

bool plexcel_authenticate(resource $px,
    string $ssn_id,
    array $options=NULL)
```

### Description

The `plexcel_authenticate` function authenticates a web client using Kerberos Single Sign-On (SSO) but it will also explicitly logon the client if adequate credentials are sent. If the client does not support SSO, the script can

also fall-back to an explicit logon.

Note: Because this function may call *plexcel\_sso* and because *plexcel\_sso* needs to be called twice for each authentication, this function should be invoked before any code that is not absolutely necessary.

Note: If the same domain controller is not used, querying objects in the directory may not yield consistent results if attributes are not or have not yet been replicated. Use *plexcel\_preamble* instead or copy it's use of *plexcel\_find\_authorities\_by\_domain* to locate and reuse the same domain controller.

## Explicit Logon by Form

If a *p\_username* HTTP request parameter is sent, the *plexcel\_logon* function will be used to explicitly logon the client with the supplied *\$ssn\_id* and the *p\_username* and *p\_password* request parameters as described in *plexcel\_logon* documentation.

Note: It is strongly recommended that `https://` be used when submitting client credentials to the server in plain text.

## SSO Authentication

If a *p\_username* HTTP request parameter is not sent, SSO is attempted. If SSO authentication fails, *plexcel\_authenticate* will return FALSE and set *plexcel\_status* to PLEXCEL\_NO\_CREDS. This indicates that the script should fall-back to a logon form through which the *p\_username* and *p\_password* request parameters can be submitted.

The *\$px* parameter is the Plexcel context resource being authenticated.

The *\$ssn\_id* parameter is the session id used with the *plexcel\_logon* function. The string returned by PHP's builtin *session\_id* function is ideal for this purpose.

The *\$options* parameter is an array of options. The following table lists the permitted options.

Option	Description
<i>auth_type</i>	This option can limit the permitted types of authentication to SSO only or logon only. The default behavior is to allow both types – see table below.
<i>logonurl</i>	This option may be set to a URL or relative URL specifying the location of the application's logon form. This is only used by the JavaScript redirect invoked by clients that do not support SSO authentication.

The following table lists the possible values for the *auth\_type* option.

Value for <i>auth_type</i> option	Description
PLEXCEL_AUTH_SSO	If set, only SSO authentication will be performed. The <i>plexcel_logon</i> function will not be called.
PLEXCEL_AUTH_LOGON	If set, only explicit logon with <i>p_username</i> and <i>p_password</i> parameters will be performed. The <i>plexcel_sso</i> function will not be called.
PLEXCEL_AUTH_SSO_LOGON	If set, both SSO and explicit logon will be performed. This is the default value.

## Returns

The *plexcel\_authenticate* function returns TRUE if the client was successfully authenticated. Otherwise, FALSE is returned in which case *plexcel\_status* should be consulted.

## Example

The following PHP script illustrates how to properly authenticate a web client using the `plexcel_authenticate` function. This example has been carefully crafted to exhibit the following properties.

- Every request is authenticated. The `plexcel_authenticate` function handles credential caching.
- If SSO authentication is successful, no logon form is presented.
- Even if SSO is successful, the logoff link allows the user to get to the logon form and enter alternative credentials.
- The `PLEXCEL_NO_CREDS` error indicates that the script should present the user with the logon form.
- After the client has successfully logged in with the form, resubmitting the page does not require logging on. The session id and hidden `p_username` parameter are sufficient to utilize any cached credential.
- If the client is logging off, they should still be authenticated so as to prevent any possibility of unauthenticated clients from maliciously logging off other users. Logging off is a privileged operation.
- Not calling `plexcel_logoff` will not affect the performance of the application however, it is a security risk to abandon credentials on the server. Always call `plexcel_logoff` if the client will no longer need their cached credential.

```
<?php
session_start();
require_once('../plexcel.php');

$username = plexcel_get_param('p_username');
$logoff = plexcel_get_param('logoff');
$is_authenticated = FALSE;
$error = NULL;

$px = plexcel_new(NULL, NULL);
$is_authenticated = plexcel_authenticate($px, session_id());
if ($is_authenticated == FALSE) {
    if (plexcel_status($px) == PLEXCEL_NO_CREDS) {
        // fall through to logon form
    } else if (plexcel_status($px) == PLEXCEL_PRINCIPAL_UNKNOWN) {
        $error = 'Bad username';
    } else if (plexcel_status($px) == PLEXCEL_LOGON_FAILED) {
        $error = 'Logon failed (e.g. bad password)';
    } else {
        $error = 'Error: <pre>' . plexcel_status($px) . '</pre>';
    }
}

if ($logoff == '1') {
    if ($username)
        plexcel_logoff($px, session_id(), $username);
    $is_authenticated = FALSE;
    $error = 'You are logged off.';
}

if ($error)
    echo $error;

echo "<form name='f' method='POST'>";
if ($is_authenticated == FALSE) {
    echo "<p/>Username: <input name='p_username' type='text' /> (must be UPN)<br/>";
    echo "Password: <input name='p_password' type='password' />";
} else {
    echo "<input name='p_username' type='hidden' value='" .
```

```

        htmlspecialchars($username) . "'/>";
    echo "<input name='logoff' type='hidden' value='0' />";
    echo "<p/>You have been successfully authenticated.";
    echo "<p/><a
href='javascript:document.f.logoff.value=1;document.f.submit();'>logoff</a>";
}
echo "<p/><input type='submit' />";
echo "</form>\n";
?>

```

Listing 17: A complete `plexcel_authenticate` example with error handling and logon form

## See also

`plexcel_preamble` | `plexcel_logon` | `plexcel_logoff` | `plexcel_sso`

## plexcel\_sso

### Synopsis

```

session_start();
require_once('plexcel.php');

bool plexcel_sso(resource $px, array $options=NULL)

```

### Description

The `plexcel_sso` function authenticates a web client using Kerberos Single Sign-On (SSO).

Note: It is recommended that general purpose web applications try to use the `plexcel_preamble` or possibly the `plexcel_authenticate` function instead. They provide higher-level, more feature-full interfaces.

Note: This function is defined in `plexcel.php` which requires that sessions be initialized. Therefore, this function must be preceded by a call to `session_start()` and `require_once('plexcel.php')`.

Note: If you find that IE is not submitting POST parameters when using this function, please note that you must call this function with *every single request*. If IE negotiates HTTP SSO authentication, it will pro-actively attempt to re-authenticate SSO authentication before it will submit POST parameters.

### The `plexcel_sso` function calls die

Initially, a client will not supply the required Authorization header token. In this case, this function will set a response header, set the status to 401 Unauthorized and call `die`. If the client supports SSO, it should then re-submit it's request with the required token and be authenticated. This means that *this function will be called twice for each request*. For performance reasons, it is strongly recommended that this function be invoked before any code that is not absolutely necessary.

The `$px` parameter is the Plexcel context resource being authenticated.

The optional `$options` parameter it is an array of options used by the higher-level `plexcel_authenticate` and `plexcel_preamble` functions to set a JavaScript redirect for clients that do not support SSO.

### Returns

The `plexcel_sso` function returns TRUE if the client was successfully authenticated. Otherwise, FALSE is returned in which case `plexcel_status` should be consulted.

## Example

The following PHP script illustrates how to authenticate a client using the *plexcel\_sso* function.

```
<?php
session_start();
require_once('../plexcel.php');

$px = plexcel_new(NULL, NULL);
if (plexcel_sso($px) == FALSE)
    die('<pre>' . plexcel_status($px) . '</pre>');

echo 'You have been successfully authenticated.';
?>
```

Listing 18: A simple *plexcel\_sso* example

## See also

[plexcel\\_preamble](#) | [plexcel\\_authenticate](#)

## plexcel\_logon

### Synopsis

```
bool plexcel_logon(resource $px,
                  string $ssn_id,
                  string $acctname,
                  string $password)
```

### Description

The *plexcel\_logon* function performs a manual logon using a session id, account name and password. This function exhibits significantly different behavior if the password supplied is NULL – see the *Cached Credentials* section below.

Note: It is recommended that general purpose web applications use the *plexcel\_preamble* or possibly the *plexcel\_authenticate* function instead. They provide higher-level, more feature-full interfaces.

The *\$px* parameter is the Plexcel context resource being authenticated.

The *\$ssn\_id* is a string identifier that will be used as a logon session identifier. The string returned by the *session\_id* function is a good candidate for this parameter.

The *\$acctname* parameter is the name of the account being authenticated. Currently this parameter must be in the user principal name form (e.g. *bcarter@example.com*).

The *\$password* parameter is the password for the named account. This parameter may also be NULL – see the *Cached Credentials* section below.

Note: It is strongly recommended that <https://> be used when submitting client credentials to the server in plain text.

### Cached Credentials

If the *\$password* parameter is NULL, the *plexcel\_logon* function will attempt to locate a cached credential (a TGT) using the supplied *\$ssn\_id* as a key. If one is found, the Plexcel context resource is authenticated and TRUE is returned. If a cached credential is not found, *plexcel\_logon* sets the *plexcel\_status* to

PLEXCEL\_NO\_CREDS and returns FALSE.

If the `$password` parameter is not NULL, the `plexcel_logon` function will attempt to perform a Kerberos AS-REQ authentication. If the authentication is successful, the resulting credential (the TGT) is cached using the supplied `$ssn_id` parameter as a key (for subsequent retrieval when `$password` is NULL as described above). If the authentication is unsuccessful, `plexcel_logon` returns FALSE in which case `plexcel_status` should be consulted.

The `plexcel_logoff` function should be called to destroy the user's cached credential.

## Returns

The `plexcel_logon` function returns TRUE if the logon was successful and FALSE if it was not in which case `plexcel_status` should be consulted.

## Example

This example uses `plexcel_logon` to get a credential and use it to access a Kerberos protected web page.

```
<?php
session_start();
require_once('../plexcel.php');

$url = 'http://www2.example.com/private.html';
$username = 'user@example.com';
$password = 'pass';

$px = plexcel_new(NULL, array('putenv_krb5ccname' => TRUE));
if ($px == NULL)
    die('<pre>' . plexcel_status(NULL) . '</pre>');
if (plexcel_logon($px, session_id(), $username, $password) == FALSE)
    die('<pre>' . plexcel_status($px) . '</pre>');

$ch = curl_init($url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_GSSNEGOTIATE);
curl_setopt($ch, CURLOPT_USERPWD, NULL);
$res = curl_exec($ch);
curl_close($ch);

echo '<small><pre>' . htmlspecialchars($res) . '</pre></small>';
?>
```

Listing 19: Using `plexcel_logon` to get a credential and use it to access a Kerberos protected web page.

## See also

`plexcel_logoff` | `plexcel_preamble` | `plexcel_authenticate` | `plexcel_sso`

## plexcel\_logoff

### Synopsis

```
bool plexcel_logoff(resource $px, string $ssn_id, string $username)
```

## Description

The *plexcel\_logoff* function destroys a cached credential created with the *plexcel\_logon* function.

The *\$px* parameter is the Plexcel context resource that was used with *plexcel\_logon* to create the credential being destroyed.

The *\$ssn\_id* parameter is the session identifier key that was used with *plexcel\_logon* to create the credential being destroyed.

The *\$username* parameter is the username that was used to logon with *plexcel\_logon*.

## Returns

The *plexcel\_logoff* function returns TRUE to indicate that the operation was successful. A value of FALSE is returned to indicate that the operation failed in which case the *plexcel\_status* function should be consulted.

## See also

*plexcel\_logon*

## plexcel\_accept\_token

### Synopsis

```
string plexcel_accept_token(resource $px, string $token)
```

## Description

The *plexcel\_accept\_token* function accepts and returns base 64 encoded authentication tokens and authenticates the Plexcel context resource in the process. It is used almost exclusively by the *plexcel\_sso* function, in conjunction with *plexcel\_status*, to implement the 'Negotiate' form of HTTP authentication supported by modern browsers.

Note: If you find that IE is not submitting POST parameters when using this function, please note that you must call this function with *every single request*. If IE negotiates HTTP SSO authentication, it will pro-actively attempt to re-authenticate SSO authentication before submitting POST parameters.

## Returns

The *plexcel\_accept\_token* function returns a base 64 encoded token that should be returned to the initiator (e.g. the web client) or FALSE if no reply token should be sent.

Notice that, unlike most other Plexcel functions, a return value of FALSE does not indicate that an error has occurred. The *plexcel\_status* function must be used to check the result of the call.

## Example

The following highly simplified version of the *plexcel\_sso* function illustrates how to authenticate web clients with the *plexcel\_accept\_token* function. Note that the calling convention of this function is somewhat awkward in that it calls *die* if there is no Authorization header.

For more complete examples, see the various Plexcel plugins for popular PHP applications.

```
function plexcel_sso_simple($px, $options=NULL) {
```

```

$token = '';
$headers = apache_request_headers();
if (isset($headers["Authorization"])) {
    $token = $headers["Authorization"];
    $token = plexcel_accept_token($px, $token);
    if (plexcel_status($px) != PLEXCEL_CONTINUE_NEEDED) {
        if (plexcel_status($px) == PLEXCEL_SUCCESS) {
            if ($token) /* mutual auth requires reply token */
                header('WWW-Authenticate: Negotiate ' . $token, TRUE, 200);
            return TRUE; /* authentication success */
        }
        /* authentication failed or something unexpected happend */
        return FALSE;
    }
    $token = ' ' . $token;
}
header('WWW-Authenticate: Negotiate' . $token);
header('HTTP/1.1 401 Unauthorized');
die('Kerberos Authentication Required');
}

```

Listing 20: A simple version of `plexcel_sso` that uses `plexcel_accept_token` directly

## See also

`plexcel_sso`, `plexcel_preamble`, `plexcel_authenticate`, `plexcel_logon`

## plexcel\_is\_member\_of

### Synopsis

```
bool plexcel_is_member_of(resource $px, string $name)
```

### Description

The `plexcel_is_member_of` function determines if the calling user is a member of the named group or if the name refers to the user's own account. The calling user is the last user to successfully authenticate using the supplied Plexcel context (or the HTTP service account if no such authentication has occurred).

The `$px` parameter is the Plexcel context resource representing the directory binding and context specific options.

The `$name` parameter specifies the account that should be compared to the calling user's list of groups. Just like the `plexcel_get_account` function, the `$name` parameter may be a CN, a traditional SAM account name, a SAM account name with a NetBIOS domain name prefix, a user principal name<sup>3</sup> or a full DN.

Note: It is recommended that a qualified name form be used. Specifying only a CN or a SAM account name without a domain may not uniquely identify the correct account.

### Behavior Differences Between SSO and Explicit Logon

Due to how group information is retrieved, BUILTIN groups (e.g. Account Operators) and Domain Local Groups may not be included in the `plexcel_is_member_of` check depending on how the user is authenticated. If SSO is used, BUILTIN groups will not be included in the `plexcel_is_member_of` check and only Domain Local Groups in the same domain as the web server service account will be in scope. If explicit logon is used, only Domain Local Groups in the same domain as the user will be in scope.

<sup>3</sup> Groups do not have UPNs and thus only a user account may be specified by UPN.

Therefore, for consistent results, do not use BUILTIN groups and avoid Domain Local Groups unless you are certain they will always be in scope (e.g. the HTTP service account and users are all in the same domain).

## Group Changes

If groups membership is changed this function may not recognize the change until the client logs off and back on to refresh their list of security groups.

## Access Control Lists

The `plexcel_is_member_of` function is ideal for implementing ACLs in applications. After the `objectSid` for the named account has been cached, no additional network communication is necessary making these calls very efficient.

Note: If no PAC field is present in Kerberos tickets, the `plexcel_is_member_of` function will be significantly slower. This is because Plexcel must fall-back to querying the directory for authorization data whereas with the PAC it would not need to contact the directory at all. See Issue 16 in the Plexcel Operator's Manual for details.

Plexcel does not currently provide higher level ACL functions but they are easy to implement. Consider the following simple implementation that uses an array of names as an ACL.

```
function plexcel_access_check($px, $acl) {
    foreach ($acl as $ace) {
        if (plexcel_is_member_of($px, $ace) == TRUE) {
            return TRUE;
        }
    }
    return FALSE;
}

// example usage
$acl = array('bcarter@example.com',
            'Managers',
            'EXAMPLE\QA Group 1',
            'example.com\CMS Beta Testers',
            'CN=Sales,OU=Europe,DC=example,DC=com');
if (plexcel_access_check($px, $acl)) {
    echo 'Access granted';
} else {
    echo 'Access denied';
}
```

Listing 21: An ACL checking function that calls `plexcel_is_member_of` in a loop

## Returns

The `plexcel_is_member_of` function returns TRUE if the calling user is determined to be a member of the named group or if the name refers to the user's own account. Otherwise FALSE is returned in which case `plexcel_status` should be tested to ensure that an error has not occurred – see example below.

## Example

The following PHP fragment illustrates how to properly use the `plexcel_is_member_of` function.

```
if (plexcel_is_member_of($px, 'Managers') == FALSE) {
    if (plexcel_status($px) != PLEXCEL_SUCCESS)
        die('<pre>' . plexcel_status($px) . '</pre>');
    echo 'You are not authorized to view this content.';
} else {
```

```
    echo 'You are in the Manager\'s group.';
}
```

Listing 22: A `plexcel_is_member_of` example with proper error handling

## plexcel\_set\_password

### Synopsis

```
bool plexcel_set_password(resource $px,  
    string $acctname,  
    string $password)
```

### Description

The `plexcel_set_password` function sets the password on the named account.

Note: Only Administrators and Account Operators can set account passwords. Regular users can change their own password but they must use the `plexcel_change_password` function.

The `$px` parameter is the Plexcel context resource representing the binding to the server on which the password will be set.

The `$acctname` parameter is the name of the account for which the password is being set. Currently this name must be in the user principal name (UPN) form (e.g. `bcarter@example.com`).

The `$password` parameter is the password that should be set on the named account.

### Returns

The `plexcel_set_password` function returns TRUE if the password was successfully set. Otherwise, FALSE is returned in which case `plexcel_status` should be consulted.

### See also

`plexcel_change_password` | `plexcel_get_service_keytab`

## plexcel\_change\_password

### Synopsis

```
bool plexcel_change_password(resource $px,  
    string $acctname,  
    string $password,  
    string $new_password)
```

### Description

The `plexcel_change_password` function sets the password on the named account. Users who are not in the Administrators or Account Operators groups must use this function to set their own password.

The `$px` parameter is the Plexcel context resource representing the binding to the server on which the password will be set.

The `$acctname` parameter is the name of the account for which the password is being set. Currently this name must be in the user principal name (UPN) form (e.g. `bcarter@example.com`).

The `$password` parameter is the current password for the named account.

The `$new_password` parameter is the new password to be set.

## Returns

The `plexcel_change_password` function returns TRUE if the password was successfully changed. Otherwise, FALSE is returned in which case `plexcel_status` should be consulted.

## See also

`plexcel_set_password` | `plexcel_get_service_keytab`

## plexcel\_gen\_service\_keytab

### Synopsis

```
bool plexcel_gen_service_keytab(resource $px,  
    string $acctname,  
    string $password,  
    string $keytab_path)
```

### Description

The `plexcel_gen_service_keytab` function creates a keytab file from the supplied account name and password. The first entry of the keytab file will have a principal matching the UPN of the named account. Additional entries will be created for each `servicePrincipalName` set on the named account. All cryptographic keys will be the same. The key is generated from the supplied password. The current enctype and kvno are extracted from a TGT for the account.

The following output of Heimdal's `ktutil` command shows the contents of a keytab file created using the `plexcel_gen_service_keytab` function.

```
/var/lib/plexcel/plexcel.keytab:  
  
Vno  Type                Principal  
  5  arcfour-hmac-md5    http_sso_www1@EXAMPLE.COM  
  5  arcfour-hmac-md5    HTTP/ww1.example.com@EXAMPLE.COM  
  5  arcfour-hmac-md5    HTTP/as1.example.com@EXAMPLE.COM
```

*Listing 23: Sample contents of a keytab file created with `plexcel_gen_service_keytab`*

The `$px` parameter is the Plexcel context resource representing the directory binding and context specific state.

The `$acctname` parameter is the account name from which the keytab principals, enctype and kvno will be determined. Currently this parameter must be in user principal name form (e.g. `server5@EXAMPLE.COM`).

The `$password` parameter is the password with which the key for all keytab entries will be computed.

The `$keytab_path` is the full path name of the keytab file to be generated. An existing file will be overwritten.

## Returns

The `plexcel_gen_service_keytab` function returns TRUE if the keytab file was successfully created. Otherwise, FALSE is returned in which case `plexcel_status` should be consulted.

## Example

The following PHP fragment demonstrates how to properly use the `plexcel_gen_service_keytab` function. Note that there is no way to retrieve the password for an account. This example simply sets the password to a known value in advance.

```
if (plexcel_set_password($px, $userPrincipalName, $password) == FALSE) {
    die('<pre>' . plexcel_status($px) . '</pre>');
} else {
    if (plexcel_gen_service_keytab($px,
        $userPrincipalName,
        $password,
        $keytab_path) == FALSE) {
        die('<pre>' . plexcel_status($px) . '</pre>');
    } else {
        echo 'The keytab file was successfully created.';
    }
}
```

Listing 24: A `plexcel_gen_service_keytab` example

## See also

`plexcel_set_password` | `plexcel_change_password`

## Directory Functions

---

### plexcel\_search\_objects

#### Synopsis

```
array plexcel_search_objects(resource $px, array $params)
```

#### Description

The *plexcel\_search\_objects* function accepts an array of search parameters *\$params* with which it searches the directory bound through *\$px* and returns an array of results.

The *\$px* parameter must be a Plexcel context created with *plexcel\_new*. The *\$params* parameter may be NULL or an array containing zero or more of the following elements.

Name	Type	Description	Default Value	Example Value
base	string	The base is the base DN of entries to which the search applies.	An empty string meaning the RootDSE	OU=Asia,DC=example,DC=com
scope	string	The scope, which is either base, one or sub, indicates that the search scope is limited to only the specified base DN, the one level below the base DN or the base DN and all of its sub-entries respectively.	base	
filter	string	The filter is a standard LDAP filter expression.	(objectClass=*)	(&(objectClass=user)(logonCount=0))
attrs	array	The <i>attrs</i> array is the list of LDAP attribute names that should be retrieved in the search.	NULL	array('userPrincipalName', 'displayName')
attrsonly	boolean	If TRUE is specified, this parameter indicates that only the attribute types should be retrieved.	FALSE	
timeout	integer	The <i>timeout</i> parameter specifies the maximum number of seconds that the server should allow for the search.	60	

Table 1: Possible elements of the *\$params* array parameter

All *\$params* parameters are optional. If the *\$params* parameter is NULL or if an element was not specified, default values will be used. Default values may be overridden using the LDAP URL supplied to *plexcel\_new*.

## User Input in Filters

When using user supplied data in a filter the data must be properly escaped to prevent the user from accidentally or maliciously injecting their own filter expressions. The following PHP fragment illustrates how to properly escape externally supplied filter data (for an example that allows the user to supply a wildcard, see the `plexcel/setup.php` script).

```
function filterEscape($str)
{
    $ret = '';
    $len = strlen($str);
    for ($si = 0; $si < $len; $si++) {
        $ch = $str[$si];
        $ord = ord($ch);
        if ($ord < 0x20 || $ord > 0x7e || strpos('()*\/', $ch)) {
            $ch = '\\' . dechex($ord);
        }
        $ret .= $ch;
    }
    return $ret;
}

$search_expr = filterEscape($search_expr);

$params = array('scope' => 'sub',
                'filter' => "(&(objectClass=user)(cn=$search_expr))");

$saccts = plexcel_search_objects($px, $params);
if (is_array($saccts) == FALSE) {
    ...
}
```

## Use the Same Directory Server for Data Consistency

When searching for data that may have been just modified, it is important that the same directory server be queried. Otherwise, your application may not display consistent results because the directory servers will not have had time to replicate your changes.

The `plexcel_preamble` function is designed to remedy this problem. Provided that your HTML form correctly propagates the `p_authority` request parameter the returned `$px` and `$bindstr` parameters will refer to the same directory server across requests. Please review the `plexcel_preamble` documentation for details.

Alternatively an explicit directory server may be specified with `plexcel_new`.

## Specifying Times in Filters

If you wish to specify a time in a search filter it will be necessary to represent the time value in it's native form and not as the value returned by Plexcel functions that have used automatic attribute conversions. In a future release functions for converting conventional dates to AD nanoseconds since 1601 values may be provided.

## Returns

The `plexcel_search_objects` function returns an array of arrays. Each array represents an object that matches the search criteria. An empty array indicates that no objects were found. A return value of FALSE indicates that an error occurred and that the `plexcel_status` function should be consulted.

Note that because an empty array in PHP will evaluate to FALSE if tested with the `==` operator, the return value **must** be tested with `is_array` or the `===` operator to distinguish between an empty array which indicates that no objects were found and FALSE which indicates that an error has occurred.

## Example

The following PHP fragment illustrates how to use the *plexcel\_search\_objects* function.

```
// find accounts with a logonCount of 0
$params = array(
    'base' => 'DC=example,DC=com',
    'scope' => 'sub',
    'filter' => '(&(objectClass=user)(logonCount=0))'
);
$objs = plexcel_search_objects($px, $params);
if (is_array($objs) == FALSE) {
    die('<pre>' . plexcel_status($px) . '</pre>');
} else {
    $count = 0;
    foreach ($objs as $obj) {
        echo $obj['distinguishedName'] . "\n";
        $count++;
    }
    echo "$count objects found\n";
    ...
}
```

Listing 25: A *plexcel\_search\_objects* example that searches for accounts with a *logonCount* of 0

## See also

[plexcel\\_new](#) | [plexcel\\_get\\_account](#)

## plexcel\_get\_account

### Synopsis

```
array plexcel_get_account(resource $px, string $name, array $attrs)
```

### Description

The *plexcel\_get\_account* function accepts a wide variety of account name forms and returns an array of attributes representing the named account.

The *\$name* parameter may be NULL, a CN, a traditional SAM account name, a SAM account name with a NetBIOS domain name prefix, a user principal name or a full DN.

If the *\$name* parameter is NULL, the account of the current user will be retrieved. The current user is the authenticated user or, if no authentication has successfully occurred, the HTTP service account.

Note: It is recommended that a qualified name form be used. Specifying only a CN or a SAM account name without a domain may not uniquely address the object.

The *\$attrs* parameter may be NULL, the constant `PLEXCEL_SUPPLEMENTAL` or an array of attribute names that should be retrieved. If *\$attrs* is NULL, all attributes of the account will be retrieved.

### The PLEXCEL\_SUPPLEMENTAL Constant

If the *\$attrs* parameter is the special constant `PLEXCEL_SUPPLEMENTAL` only the following attributes will be retrieved:

```
userPrincipalName
objectSid
sAMAccountName
displayName
homeDirectory
homeDrive
```

Listing 26: The “supplemental” attributes retrieved with the PLEXCEL\_SUPPLEMENTAL constant

Calling `plexcel_get_account` with PLEXCEL\_SUPPLEMENTAL is usually very fast<sup>4</sup> and should be favored when only the above attributes are required (e.g. only the `displayName` is needed).

Note: Currently attribute definitions set with the `plexcel_set_attrdefs` function have no effect on account data retrieved with the PLEXCEL\_SUPPLEMENTAL flag.

## Returns

The `plexcel_get_account` function returns an array of attributes representing the desired account or FALSE to indicate that an error has occurred in which case `plexcel_status` should be consulted.

## Example

The following are examples are all equivalent (assuming the CN and sAMAccountName are unique).

```
// Get account by CN
$acct = plexcel_get_account($px, 'Hans Müller', NULL);
// or by sAMAccountName
$acct = plexcel_get_account($px, 'hmuller', NULL);
// or by NetBIOSName\sAMAccountName
$acct = plexcel_get_account($px, 'EXAMPLE\hmuller', NULL);
// or by userPrincipalName (except for groups)
$acct = plexcel_get_account($px, 'hmuller@example.com', NULL);
// or by distinguishedName
$acct = plexcel_get_account($px, 'CN=Hans Müller,CN=Users,DC=example,DC=com', NULL);
```

Listing 27: Account name forms used with `plexcel_get_account` (again).

The following PHP fragment illustrates how to retrieve the account information encoded in the Kerberos ticket of the authenticated user.

```
...
if (plexcel_sso($px) == FALSE) {
    die('<pre>' . plexcel_status($px) . '</pre>');
} else {
    $acct = plexcel_get_account($px, NULL, PLEXCEL_SUPPLEMENTAL);
    if (is_array($acct) == FALSE)
        die('<pre>' . plexcel_status($px) . '</pre>');
    } else {
        echo '<pre>';
        print_r($acct);
        echo '</pre>';
        ...
    }
```

Listing 28: A `plexcel_get_account` example that uses PLEXCEL\_SUPPLEMENTAL

<sup>4</sup> If the `$name` parameter is NULL and the user has been authenticated using Kerberos, no communication with the directory is necessary. The PLEXCEL\_SUPPLEMENTAL attributes are retrieved directly from the PAC of the Kerberos ticket supplied by the client during authentication.

## See also

plexcel\_search\_objects

## plexcel\_add\_object

### Synopsis

```
bool plexcel_add_object(resource $px, array $obj, array $attrs)
```

### Description

The *plexcel\_add\_object* function adds an object to the directory.

The *\$px* parameter is the Plexcel context resource representing the directory binding and context specific options.

The *\$obj* parameter is an array of attributes representing the object being added to the directory. The *distinguishedName* attribute in this parameter is used to address the object being created and must be present.

The *\$attrs* parameter is an array of attribute names indicating the attributes in the *\$obj* array that should be created. If the *\$attrs* parameter is NULL, all attributes in *\$obj* will be created when the object is added to the directory. Using this parameter can make script logic simpler.

### Creating Accounts

When creating accounts beware that the *userAccountControl* flag *PLEXCEL\_PASSWD\_NOTREQD* is required. Otherwise a *PLEXCEL\_LDAP\_UNWILLING\_TO\_PERFORM* error may occur. The error is caused by a password policy violation. Specifying the *PLEXCEL\_PASSWD\_NOTREQD* flag allows creating an account without the password and thus no password policy violation will occur.

### Returns

The *plexcel\_add\_object* function returns TRUE if the object was successfully added to the directory. Otherwise, FALSE is returned in which case the *plexcel\_status* function should be consulted.

### Example

The following PHP script will create a user account using the *plexcel\_add\_object* function.

```
<?php
session_start();

$base           = 'OU=Europe,DC=example,DC=com';
$domain         = 'example.com';
$username       = "user@$domain";
$password       = 'pass';

$givenName      = 'Hans';
$sn             = 'Müller';
$sAMAccountName = 'hmuller';
// always use PLEXCEL_PASSWD_NOTREQD to prevent password policy violation
```

```

$userAccountControl = PLEXCEL_NORMAL_ACCOUNT | PLEXCEL_PASSWD_NOTREQD;

$cn = "$givenName $sn";
$distinguishedName = "CN=$cn,$base";
$userPrincipalName = "$sAMAccountName@$domain";
$description = "This is $cn's description.";
$telephoneNumber = '0611/11111 0';

$px = plexcel_new(NULL, NULL);

if (plexcel_logon($px, session_id(), $username, $password) == FALSE) {
    die('<pre>' . plexcel_status($px) . '</pre>');
} else {
    $distinguishedName = "CN=$cn,$base";

    $acct = plexcel_get_account($px, $sAMAccountName, array('sAMAccountName'));
    if (is_array($acct)) {
        die("An account with the name $sAMAccountName already exists: " .
            $acct['distinguishedName']);
    } else {
        $acct = array('objectClass' => array('user'),
            'distinguishedName' => $distinguishedName,
            'sAMAccountName' => $sAMAccountName,
            'givenName' => $givenName,
            'sn' => $sn,
            'userPrincipalName' => $userPrincipalName,
            'description' => array($description),
            'telephoneNumber' => $telephoneNumber,
            'userAccountControl' => $userAccountControl);
        if (plexcel_add_object($px, $acct, NULL) == FALSE) {
            die('<pre>' . plexcel_status($px) . '</pre>');
        } else {
            echo 'The account was created successfully.';
        }
    }
}
}
?>

```

Listing 29: Adding a user to the directory using `plexcel_add_object`

## See also

`plexcel_modify_object` | `plexcel_delete_object` | `plexcel_search_objects` | `plexcel_get_account`

## plexcel\_modify\_object

### Synopsis

```
bool plexcel_modify_object(resource $px, array $obj, array $attrs)
```

### Description

The `plexcel_modify_object` function can add, delete and replace attributes of an object in the directory. Multiple attribute modifications of different types can be performed with a single call.

The `$px` parameter is the Plexcel context resource representing the directory binding and context specific options.

The `$obj` parameter is an array of attribute names and values representing the attributes being added, replaced or deleted. All modifications are performed on the object identified by the `distinguishedName` attribute in the `$obj` parameter and therefore it must be present.

The `$attrs` parameter is an array of attribute names and corresponding modification types. The modification types are described in the following table.

Flag	Description
PLEXCEL_MOD_ADD	<p>This indicates that the value of the named attribute in the <code>\$obj</code> array is to be added to the target object.</p> <p>If the attribute is multivalued, the value in <code>\$obj</code> must be an array in which case all values will be added to the target object.</p>
PLEXCEL_MOD_DELETE	<p>This indicates that the value of the named attribute in the <code>\$obj</code> array is to be deleted from the target object.</p> <p>If the attribute is multivalued, the value in <code>\$obj</code> must be an array in which case all values will be deleted from the target object.</p> <p>If no value for the named attribute is supplied in the <code>\$obj</code> array, all attributes with that attribute name will be deleted in the target object.</p>
PLEXCEL_MOD_REPLACE	<p>This indicates that the named attribute is to be replaced with the corresponding value in the <code>\$obj</code> parameter. If the attribute does not exist, it will be created.</p> <p>IMPORANT: If the attribute is multivalued, all existing values are <b>deleted</b> and replaced with the values in the <code>\$obj</code> parameter. For example, when setting a member attribute in a group object, if PLEXCEL_MOD_REPLACE is used, all existing members of the group will be deleted and completely replaced with only those members supplied through <code>\$obj</code> which is probably not the desired behavior. Use PLEXCEL_MOD_ADD to add members to a group – see example below.</p>

An `$attrs` element may also be only an attribute name with no modification type. In this case, the modification type will default to PLEXCEL\_MOD\_REPLACE. Meaning `array('sn' => PLEXCEL_MOD_REPLACE)` and `array('sn')` are equivalent.

## Returns

The `plexcel_modify_object` function returns TRUE if the directory successfully processed all modifications. Otherwise, FALSE is returned in which case the `plexcel_status` function should be consulted.

## Example

The following PHP fragment illustrates how to add a member to a group using `plexcel_modify_object`.

```
$gacct = array('distinguishedName' => 'CN=Managers,OU=Europe,DC=example,DC=com',
              'member' => array('CN=Hans Müller,CN=Users,DC=example,DC=com'));
$attrs = array('member' => PLEXCEL_MOD_ADD);
if (plexcel_modify_object($px, $gacct, $attrs) == FALSE) {
    die('<pre>' . plexcel_status($px) . '</pre>');
}
```

Listing 30: Adding a user to a group with `plexcel_modify_object`

The following somewhat unrealistic PHP fragment demonstrates adding, replacing and deleting multiple attributes with one call to *plexcel\_modify\_object*.

```
$acct = array('distinguishedName' => 'CN=fs1,OU=Linux Servers,DC=example,DC=com');
$acct['servicePrincipalName'] = array(
    'host/fs1.example.com',
    'FTP/fs1.example.com',
    'cifs/fs1.example.com');
$acct['info'] = 'Storage for infrastructure and developers';
//
// add SPNs, replace info field and delete description field
//
$attrs = array('servicePrincipalName' => PLEXCEL_MOD_ADD,
    'info' => PLEXCEL_MOD_REPLACE,
    'description' => PLEXCEL_MOD_DELETE);
if (plexcel_modify_object($px, $acct, $attrs) == FALSE) {
    $err = die('<pre>' . plexcel_status($px) . '</pre>');
}
```

Listing 31: Adding, replacing and deleting multiple attributes with one call to *plexcel\_modify\_object*

## See also

[plexcel\\_add\\_object](#) | [plexcel\\_delete\\_object](#) | [plexcel\\_rename\\_object](#)

## plexcel\_delete\_object

### Synopsis

```
bool plexcel_delete_object(resource $px, string $dn)
```

### Description

The *plexcel\_delete\_object* function deletes the object identified by the *\$dn* parameter from the directory.

The *\$px* parameter is the Plexcel context resource representing the directory binding and context specific options.

The *\$dn* parameter is the full distinguished name of the object to be deleted.

### Returns

The *plexcel\_delete\_object* function returns TRUE if the object was successfully deleted. Otherwise, FALSE is returned in which case *plexcel\_status* should be consulted.

## See also

[plexcel\\_add\\_object](#) | [plexcel\\_modify\\_object](#) | [plexcel\\_rename\\_object](#)

## plexcel\_rename\_object

### Synopsis

```
bool plexcel_rename_object(resource $px,  
    string $dn,  
    string $newrdn,  
    string $newparent)
```

### Description

The *plexcel\_rename\_object* function renames and / or moves the object identified by the \$dn parameter in the directory.

The \$px parameter is the Plexcel context resource representing the directory binding and context specific options.

The \$dn parameter is the full distinguished name of the object to be renamed.

The \$newrdn is the new Relative Distinguished Name (RDN) of the object (e.g. "CN=Alice Baker").

The \$newparent is the new base DN of the object after it is renamed. This parameter is optional and defaults to NULL. Meaning, if a \$newparent parameter is supplied and is not NULL, the object will be effectively moved between containers.

Note: This function supports an optional 5th boolean parameter that indicates as to whether or not the old entry should be deleted. However, currently Active Directory does not support any value other than the default of TRUE.

### Example

The following simple fragment of PHP illustrates how to rename an entry in the directory. It also uses the optional \$newparent parameter to move the object to a new container.

```
$dn = 'CN=TS110,OU=Tést,DC=example,DC=com';  
$newrdn = 'CN=PS110';  
$newparent = 'OU=Prod,DC=example,DC=com';  
if (plexcel_rename_object($px, $dn, $newrdn, $newparent) == FALSE) {  
    die('<pre>' . plexcel_status($px) . '</pre>');  
}  
echo 'The object was renamed successfully.';
```

### Returns

The *plexcel\_rename\_object* function returns TRUE if the object was successfully renamed. Otherwise, FALSE is returned in which case *plexcel\_status* should be consulted.

### See also

[plexcel\\_add\\_object](#) | [plexcel\\_modify\\_object](#) | [plexcel\\_delete\\_object](#)

## plexcel\_set\_attrdefs

### Synopsis

```
bool plexcel_set_attrdefs(resource $px, array $attrdefs)
```

### Description

The *plexcel\_set\_attrdefs* function sets attribute definitions that allow Plexcel functions to accept and return attribute values that are more convenient for the programmer.

Note: Currently attribute definitions do not work with the PLEXCEL\_SUPPLEMENTAL flag used with *plexcel\_get\_account*.

Attribute definitions for more than 200 attributes commonly found in AD are preloaded which means that it may not be necessary to use this function at all. However, all other attributes, such as those from less commonly used objects or from schema customizations, will be represented within scripts as multivalued and binary (the default behavior is just like the raw LDAP API). In this case, it may be convenient for the programmer to add their own attribute definitions with this function.

The *\$px* parameter is the Plexcel context resource for which attribute definitions will be set. Attribute definitions are limited in scope to the specified Plexcel context and will not affect attribute definitions of other contexts.

The *\$attrdefs* parameter is an array of attribute definitions. An attribute definition is an array containing the elements *type*, *flags* and *conv* described below.

### The type element

The type element instructs Plexcel as to how to represent the attribute at the script level. The following table describes the possible type values.

Value	Description
PLEXCEL_TYPE_STRING	An attribute of this type will be represented within scripts as a string. The value will automatically be converted from UTF-8 to the default locale encoding Apache runs under (e.g. ISO-8859-1@euro).
PLEXCEL_TYPE_BOOLEAN	An attribute of this type will be represented within scripts as a boolean.  For PHP this means 1 or 0 or the identifiers TRUE or FALSE. For example, setting a boolean attribute to the string 'FALSE' would probably not produce the desired behavior.
PLEXCEL_TYPE_BINARY	An attribute of this type is not modified from it's directory representation.
PLEXCEL_TYPE_TIME	An attribute of this type will be represented as a standard LDAP UTC time string like 20070307033755.0Z.
PLEXCEL_TYPE_INT32	An attribute of this type will be represented as a simple integer.
PLEXCEL_TYPE_INT64	An attribute of this type will be represented within PHP as a double value.

### The flags element

The flags element should be set to PLEXCEL\_SINGLE\_VALUED if the attribute is single valued or 0 if it is multivalued. If the PLEXCEL\_SINGLE\_VALUED flag is set, Plexcel functions that return objects will represent the attribute as a simple array value indexed by name (e.g. *\$obj['name']*). Otherwise, like the raw LDAP API,

single valued attributes will be represented as a 0 indexed array value within an array indexed by name (e.g. \$obj[ ' name ' ][0]). When manipulating large numbers of attributes, this small change can make your significantly simpler and more efficient.

## The conv element

The conv element indicates to Plexcel that certain attributes should be automatically converted between a script-level representation and the directory representation. The conv element values are always in the form PLEXCEL\_CONV\_<script representation>\_X\_<directory representation> as shown in the below table of possible conv values.

Value	Description
PLEXCEL_CONV_BASE64_X_BINARY	This conv value automatically converts between a base 64 encoding within scripts and binary within the directory. For example, an objectGUID attribute would be represented within scripts as the string t7xt74WBqUW6iS1gKsr11A==. This conversion helps prevent scripts from emitting garbage data at the user or into data files.
PLEXCEL_CONV_SIDSTR_X_BINARY	This conv value automatically converts between a SID string like S-1-5-21-4133389617-793951518-2001521813-1341 and it's binary representation in the directory. This conversion is useful for SID binding and advanced utilities that work with SIDs.
PLEXCEL_CONV_TIME1970M_X_TIMEUTC	This conv value automatically converts between milliseconds since 1970 and a UTC time string like 20060719203335.0Z. This conversion is ideal for producing human readable dates and performing date logic. See the example below.
PLEXCEL_CONV_TIME1970M_X_TIME1601	This conv value automatically converts between milliseconds since 1970 and nanoseconds since 1601 dates commonly used by AD. This conversion is also ideal for producing human readable dates and performing date logic.
PLEXCEL_CONV_INT32_X_STRING	This conv value automatically converts between an integer and a string.
PLEXCEL_CONV_STRING_X_BINARY	This conv value automatically converts between strings and binary.

## Returns

The *plexcel\_set\_attrdefs* function returns TRUE if the attribute definitions were successfully added to the Plexcel context resource. Otherwise, FALSE is returned in which case *plexcel\_status* should be consulted.

## Example

The following PHP fragment illustrates how to set an attribute definition with the PLEXCEL\_CONV\_TIME1970M\_X\_TIMEUTC automatic type conversion and then print a human readable time using PHP's *date* function.

```
$attrdefs = array(
    'currentTime' => array(
        'type' => PLEXCEL_TYPE_TIME,
```

```

        'flags' => PLEXCEL_SINGLE_VALUED,
        'conv' => PLEXCEL_CONV_TIME1970M_X_TIMEUTC));
if (plexcel_set_attrdefs($px, $attrdefs) == FALSE)
    die('<pre>' . plexcel_status($px) . '</pre>');

// call plexcel_search_objects to get currentTime from RootDSE

// This prints a date string like Mar 16, 2007 5:58:51 PM

echo 'currentTime:' . date('M j, Y g:i:s A', $rootdse['currentTime'] / 1000.0);

```

Listing 32: Pretty-printing the currentTime with help from a plexcel\_set\_attrdefs conversion.

## See also

plexcel\_set\_conv\_attrdefs | plexcel\_get\_attrdefs

## plexcel\_get\_attrdefs

### Synopsis

```
array plexcel_get_attrdefs(resource $px, array $attrs)
```

### Description

The *plexcel\_get\_attrdefs* function will retrieve a list of attribute definitions for the specified attribute. This function may be used by applications that require attribute intelligence such as a directory editor or to validate user input.

The *\$px* parameter is the Plexcel context resource from which attribute definitions are being queried.

The *\$attrs* parameter is an array of attribute names for which attribute definitions should be returned.

A Plexcel attribute definition is an array with elements *type*, *flags* and *conv*. The *type* and *conv* values are defined by the constants listed in for *plexcel\_set\_attrdefs*. When querying attribute definitions however, a complete list of values for the flags element is necessary. These are shown below.

Flag	Description
PLEXCEL_TYPE_UNDEFINED	This flag indicates that no attribute definition is set and that the default definition was returned (multivalued binary).
PLEXCEL_SINGLE_VALUED	This flag indicates that the attribute is single valued.
PLEXCEL_CASE_EXACT	This flag indicates that the attribute is a string that should use case sensitive behavior when compared to other strings. Currently Plexcel ignores this flag.
PLEXCEL_DN	This flag indicates that the attribute is a distinguished name. Default attribute definitions set this flag for DNs but otherwise ignore it.
PLEXCEL_PROTECTED	This flag indicates that the attribute is protected. Currently only the distinguishedName attribute has this flag set and is otherwise currently ignored by Plexcel.

### Returns

The *plexcel\_get\_attrdefs* returns an array of zero or more attribute definitions. Otherwise, FALSE is returned in which case *plexcel\_status* should be consulted.

Note: Because an empty array will evaluate to FALSE when tested with the == operator, it is recommended that the `is_array` function be used to test the return value to distinguish between an empty array and FALSE.

## Example

The following PHP script prints the default attribute definitions for the `displayName`, `servicePrincipalName` and `bogus` attributes.

```
<?php
$px = plexcel_new(NULL, NULL);
$attrs = array('displayName', 'servicePrincipalName', 'bogus');
$attrdefs = plexcel_get_attrdefs($px, $attrs);
if (is_array($attrdefs) == FALSE)
    die('<pre>' . plexcel_status(NULL) . '</pre>');
echo '<pre>';
print_r($attrdefs);
echo '</pre>';
?>
```

Listing 33: Examining default attribute definitions with `plexcel_get_attrdefs`

The above script prints the following output (minus comments added afterward).

```
Array
(
    [lastLogon] => Array
        (
            [type] => 6      // PLEXCEL_TYPE_INT64
            [flags] => 4    // PLEXCEL_SINGLE_VALUED
            [conv] => 0     //
        )

    [servicePrincipalName] => Array
        (
            [type] => 1     // PLEXCEL_TYPE_STRING
            [flags] => 0    //
            [conv] => 0     //
        )

    [bogus] => Array
        (
            [type] => 3     // PLEXCEL_TYPE_BINARY
            [flags] => 2    // PLEXCEL_TYPE_UNDEFINED
            [conv] => 0     //
        )
)
```

Listing 34: The `plexcel_get_attrdefs` example `print_r` output

## See also

`plexcel_set_attrdefs`

## plexcel\_set\_conv\_attrdefs

### Synopsis

```
session_start();
require_once('plexcel.php');

bool plexcel_set_conv_attrdefs($px)
```

### Description

The `plexcel_set_conv_attrdefs` function sets attribute definitions with standard conversions for many commonly used attributes.

Below is a partial list of the conversions performed.

Attribute	Conversion
whenCreated	PLEXCEL_CONV_TIME1970M_X_TIMEUTC
whenChanged	PLEXCEL_CONV_TIME1970M_X_TIMEUTC
objectId	PLEXCEL_CONV_SIDSTR_X_BINARY
objectGUID	PLEXCEL_CONV_BASE64_X_BINARY
badPasswordTime	PLEXCEL_CONV_TIME1970M_X_TIME1601
lastLogoff	PLEXCEL_CONV_TIME1970M_X_TIME1601
lastLogon	PLEXCEL_CONV_TIME1970M_X_TIME1601
pwdLastSet	PLEXCEL_CONV_TIME1970M_X_TIME1601
accountExpires	PLEXCEL_CONV_TIME1970M_X_TIME1601 Note: This value is usually set to (double)0x7fffffffffffffffffff. Plexcel has a constant for this value: PLEXCEL_EXPIRES_NEVER.
userParameters	PLEXCEL_CONV_BASE64_X_BINARY

### Returns

The `plexcel_set_conv_attrdefs` function returns TRUE if the the attribute definitions were successfully set. Otherwise, FALSE is returned in which case `plexcel_status` should be consulted.

### See also

`plexcel_set_attrdefs` | `plexcel_get_attrdefs`

## Index of Listings

Listing 1: A simple “whoami” script.....	4
Listing 2: Setting the putenv_krb5ccname option.....	5
Listing 3: Account name forms accepted by plexcel_get_account.....	5
Listing 4: Searching the directory for an account with the help of an advanced binding string.....	6
Listing 5: A simple plexcel_is_member_of example.....	7
Listing 6: Using DefaultNamingContext in an LDAP URL with plexcel_new.....	12
Listing 7: A small subset of plexcel_status codes.....	12
Listing 8: A sample exception returned by plexcel_status.....	12
Listing 9: A plexcel_authenticate example with error handling.....	13
Listing 10: A plexcel_find_authorities_by_domain example with lookup algorithm comments.....	14
Listing 11: Properly determining the lastLogon time with plexcel_find_authorities_by_domain.....	15
Listing 12: Retrieving the plexcel resource’s domain controller hostname with plexcel_get_authority.....	15
Listing 13: Retrieving the NetBIOSName, dnsRoot and objectSid of a domain with plexcel_get_domain.....	16
Listing 14: Logging request parameters to the Plexcel log file.....	17
Listing 15: Using PHP’s list construct to dereference the \$preamble array into variables.....	19
Listing 16: A complete plexcel_preamble example with logon form.....	20
Listing 17: A complete plexcel_authenticate example with error handling and logon form.....	23
Listing 18: A simple plexcel_sso example.....	24
Listing 19: Using plexcel_logon to get a credential and use it to access a Kerberos protected web page.....	25
Listing 20: A simple version of plexcel_sso that uses plexcel_accept_token directly.....	27
Listing 21: An ACL checking function that calls plexcel_is_member_of in a loop.....	28
Listing 22: A plexcel_is_member_of example with proper error handling.....	29
Listing 23: Sample contents of a keytab file created with plexcel_gen_service_keytab.....	30
Listing 24: A plexcel_gen_service_keytab example.....	31
Listing 25: A plexcel_search_objects example that searches for accounts with a logonCount of 0.....	34
Listing 26: The “supplemental” attributes retrieved with the PLEXCEL_SUPPLEMENTAL constant.....	35
Listing 27: Account name forms used with plexcel_get_account (again).....	35
Listing 28: A plexcel_get_account example that uses PLEXCEL_SUPPLEMENTAL.....	35
Listing 29: Adding a user to the directory using plexcel_add_object.....	37
Listing 30: Adding a user to a group with plexcel_modify_object.....	38
Listing 31: Adding, replacing and deleting multiple attributes with one call to plexcel_modify_object.....	39
Listing 32: Pretty-printing the currentTime with help from a plexcel_set_attrdefs conversion.....	43
Listing 33: Examining default attribute definitions with plexcel_get_attrdefs.....	44
Listing 34: The plexcel_get_attrdefs example print_r output.....	44