eGroupWare Application Development

Dan Kuykendall <dan@kuykendall.org>

v0.9 29 September 2000

This document explains eGroupWare's infrastructure and API, along with what is required to integrate applications into it.

Contents

1 Introduction

eGroupWare is a web based groupware application framework (API), for writing applications. Integrated applications such as email, calendar, todo list, address book, and file manager are included. eGroupWare is a fork of phpGroupWare, for which the original version of this document was written.

1.1 Overview of application writing

We have attempted to make writing applications for eGroupWare as painless as possible. We hope any pain and suffering is caused by making your application work, but not dealing with eGroupWare itself.

1.2 What does the eGroupWare API provide?

The eGroupWare API handles session management, user/group management, has support for multiple databases, using either PHPLIB or ADODB database abstraction methods, we support templates using the PHPLIB Templates class, a file system interface, and even a network i/o interface.

On top of these standard functions, eGroupWare provides several functions to give you the information you need about the users environment, and to properly plug into eGroupWare.

2 Guidelines

2.1 Requirements

These guidelines must be followed for any application that wants considered for inclusion into eGroupWare:

- It must run on PHP4 and PHP5.
- SQL statements must be compatible with both MySQL and PostgreSQL. When in doubt it is best to stick with SQL92.
- It must use our default header.inc.php include.
- It must use our $GLOBALS['phpgw']->link($url) for all links (this is for session support).
- It must use "post" for forms.
- It must respect phpGW group rights and phpGW user permissions.
- It must use our directory structure, template support and lang (multi-language) support.
- Where possible it should run on both Unix and NT platforms.
- For applications that do not meet these requirements, they can be made available to users however you decide. If you need help converting your application to templates and our lang support, we will try to connect you with someone to help.

2.2 Writing/porting your application

Include files

Each PHP page you write will need to include the header.inc.php along with a few variables.
This is done by putting this at the top of each PHP page.

```
  <?php

$GLOBALS['phpgw_info']['flags']['currentapp'] = 'appname';

include('../header.inc.php');
  ?>
```

Of course change application name to fit.
This include will provide the following things:

- The phpgwAPI - The eGroupWare API will be loaded.
- The phpGW navbar will be loaded (by default, but can be disabled until a later point.
- appname/inc/functions.inc.php - This file is loaded just after the phpgwAPI and before any HTML code is generated. This file should include all your application specific functions.. You are welcome to include any additional files you need from within this file.
- appname/inc/header.inc.php - This file is loaded just after the system header/navbar, and allows developers to use it for whatever they need to load.

- appname/inc/footer.inc.php - This file is loaded just before the system footer, allowing developers to close connections and whatever else they need.
- The phpGW footer will be loaded, which closes several connections.

3 Installing your application

3.1 Overview

It is fairly simple to add and delete applications to/from eGroupWare.

3.2 Automatic features

To make things easy for developers we go ahead and load the following files.

- appname/inc/functions.inc.php - This file should include all your application specific functions.
- appname/inc/header.inc.php - This file is loaded by $GLOBALS['phpgw']->common->header just after the system header/navbar, and allows developers to use it for whatever they need to load.
- appname/inc/footer.inc.php - This file is loaded by $GLOBALS['phpgw']->common->footer just before the system footer, allowing developers to close connections and whatever else they need.

3.3 Adding files, directories and icons.

You will need to create the following directories for your code
(replace 'appname' with your application name)

```
`--appname
  |--inc
  |   |--functions.inc.php
  |   |--header.inc.php
  |   |--hook_preferences.inc.php
  |   |--hook_admin.inc.php
  |   `--footer.inc.php
  `--templates
  |   `--default
```

3.4 Making eGroupWare aware of your application

Please see the documentation in the setup/doc directory for information on integrating into eGroupWare. This is very important since the steps for database table setup and modification discussed there must be followed.

3.5 Hooking into Administration page

When a user goes to the Administration page, it starts appname/inc/hook_admin.inc.php for each application that is enabled, in alphabetical order of application title. If the file exists, it is include()d in the hopes it will display a selection of links to configure that application.

Simple Example:

```
<?php

$img = '/' . $appname . '/images/navbar.gif';
  section_start('My Application',$img);
  echo '<a HREF="' . $GLOBALS['phpgw']->link('myAdminPage.php') . '">';
  echo lang('Change myApp settings') . '</a>';
  section_end();
?>
```

Look at headlines/inc/hook_admin.inc.php and admin/inc/hook_admin.inc.php for more examples.

Things to note:

- Links are relative to the admin/index.php file, not your application's base directory. (so use $appname in your link() calls)
- The file is brought in with include() so be careful to not pollute the name-space too much

The standard $GLOBALS['phpgw'] and $GLOBALS['phpgw_info'] variables are in-scope, as is $appname which corresponds to the application name in the path.

There are 2 functions to coordinate the display of each application's links, section_start() and section_end()

section_start

section_start($title,$icon_url) starts the section for your application. $title is passed through lang() for you. $icon_url should be page-relative to admin/index.php or an absolute URL.

section_end

section_end() closes the section that was started with section_start().

3.6 Hooking into Preferences page

The mechanism to hook into the preferences page is identical to the one used to hook into the administration page, however it looks for appname/inc/hook_preferences.inc.php instead of appname/inc/hook_admin.inc.php. The same functions and variables are defined.

4 Infrastructure

4.1 Overview

eGroupWare attempts to provide developers with a sound directory structure to work from.
The directory layout may seem complex at first, but after some use, you will see that it is designed to

accommodate a large number of applications and functions.

4.2 Directory tree

```
.--appname
|   |--inc
|   |   |--functions.inc.php
|   |   |--header.inc.php
|   |   |--hook_preferences.ini.php
|   |   |--hook_home.inc.php
|   |   `--footer.inc.php
|   |--manual
|   |--setup
|   |   |--tables_baseline.inc.php
|   |   |--tables_current.inc.php
|   |   |--tables_update.inc.php
|   |   |--setup.inc.php
|   `--templates
|   |   `--default
|   |      `--images
|   |        `--navbar.png
|   |--preferences.php
|--docs (installation docs)
|--files
|   |--groups
|   `--users
`--phpgwapi
   |--cron (egroupware's optional daemons)
   |--doc (developer docs)
   |--inc
   |   |--class.phpgw.inc.php
   |   |--class.common.inc.php
   |   `--etc..
   |--manual
   |--setup
   |   |--tables_baseline.inc.php
   |   |--tables_current.inc.php
   |   |--tables_update.inc.php
   |   |--setup.inc.php
   |--templates
```

```
|   |--default
|   |   `--images
|   |   |--home.gif
|   |   `--preferences.gif
|   `--verilak
|       `--images
        |--home.gif
        `--preferences.gif
`--themes
   `--default.theme
```

4.3 Translations

The translations are now being done thru the database, and will be configurable to use other mechanisms.

The application, developer_tools, provides developers/translators a nice GUI for building and updating translations.

5 The API

5.1 Introduction

eGroupWare attempts to provide developers with a useful API to handle common tasks.

To do this we have created a multi-dimensional class $GLOBALS['phpgw'].

This allows for terrific code organization, and help developers easily identify the file that the function is in. All the files that are part of this class are in the inc/core directory and are named to match the sub-class.

Example: $GLOBALS['phpgw']->send->msg() is in the inc/phpgwapi/phpgw_send.inc.php file.

5.2 Basic functions

$GLOBALS['phpgw']->link

$GLOBALS['phpgw']->link($url)
Add support for session management. ALL links must use this, that includes href's form actions and header location's.

If you are just doing a form action back to the same page, you can use it without any parameters.

This function is right at the core of the class because it is used so often, we wanted to save developers a few keystrokes. Example:

```
<form name="copy" method="post" action="<?php echo $GLOBALS['phpgw']->link();?>">
/* If session management is done via passing url parameters */
/* The the result would be */
/* <form name="copy" method="post" action="somepage.php?sessionid=87687693276?kp3=kjh98u80"> */
```

5.3 Application Functions

$GLOBALS['phpgw']->common->phpgw_header();

$GLOBALS['phpgw']->phpgw_header()
Print out the start of the HTML page, including the navigation bar and includes appname/inc/header.php

$GLOBALS['phpgw']->common->phpgw_footer();

$GLOBALS['phpgw']->phpgw_footer()
Prints the system footer, and includes appname/inc/footer.php

$GLOBALS['phpgw']->common->appsession();

$GLOBALS['phpgw']->common->appsession($data)
Store important information session information that your application needs.
$GLOBALS['phpgw']->appsession will return the value of your session data is you leave the parameter empty [i.e. $GLOBALS['phpgw']->appsession("")], otherwise it will store whatever data you send to it.
You can also store a comma delimited string and use explode() to turn it back into an array when you receive the value back.

Example:

```
  $GLOBALS['phpgw']->common->appsession('/path/to/something');
  echo "Dir: " . $GLOBALS['phpgw']->common->appsession();
```

5.4 File functions

Please also see the phpgwapi/doc/vfs directory for additional VFS class documentation

$GLOBALS['phpgw']->vfs->read_file

$GLOBALS['phpgw']->vfs->read_file($file)
Returns the data from $file.
You must send the complete path to the file.
Example:

```
$data = $GLOBALS['phpgw']->vfs->read_file('/some/dir/to/file.txt');
```

$GLOBALS['phpgw']->vfs->write_file

$GLOBALS['phpgw']->vfs->write_file($file, $contents)
Write data to $file.
You must send the complete path to the file.
Example:

```
$data = $GLOBALS['phpgw']->vfs->write_file("/some/dir/to/file.txt");
```

$GLOBALS['phpgw']->vfs->read_userfile

$GLOBALS['phpgw']->vfs->read_userfile($file)
Returns the data from $file, which resides in the users private dir.
Example:

```
$data = $GLOBALS['phpgw']->vfs->read_userfile("file.txt");
```

$GLOBALS['phpgw']->vfs->write_userfile

$GLOBALS['phpgw']->write_userfile($file, $contents)
Writes data to $file, which resides in the users private dir.
Example:

```
$data = $GLOBALS['phpgw']->vfs->write_userfile("file.txt");
```

$GLOBALS['phpgw']->vfs->list_userfiles

$GLOBALS['phpgw']->vfs->list_userfiles()
Returns an array which has the list of files in the users private dir.
Example:

```
$filelist = array();
$filelist = $GLOBALS['phpgw']->vfs->list_userfiles();
```

5.5 Email/NNTP Functions

$GLOBALS['phpgw']->send->msg

$GLOBALS['phpgw']->send->msg($service, $to, $subject, $body, $msgtype, $cc, $bcc)
Send a message via email or NNTP and returns any error codes.
Example:

```
$to = "someuser@domain.com";
$subject = "Hello buddy";
$body = "Give me a call\n Been wondering what your up to.";
$errors = $GLOBALS['phpgw']->send->msg("email", $to, $subject, $body);
```

6 Configuration Variables

6.1 Introduction

eGroupWare attempts to provide developers with as much information about the user, group, server, and application configuration as possible.

To do this we provide a multi-dimensional array called '$GLOBALS['phpgw_info'][]', which includes all the information about your environment.

Due to the multi-dimensional array approach. getting these values is easy.

Here are some examples:

```
<?php
// To do a hello username
echo "Hello " . $GLOBALS['phpgw_info']['user']['fullname'];
//If username first name is John and last name is Doe, prints: 'Hello John Doe'
?>
<?php
// To find out the location of the imap server
echo "IMAP Server is named: " . $GLOBALS['phpgw_info']['server']['imap_server'];
//If imap is running on localhost, prints: 'IMAP Server is named: localhost'
?>
```

6.2 User information

$GLOBALS['phpgw_info']["user"]["userid"] = The user ID.

$GLOBALS['phpgw_info']["user"]["sessionid"] = The session ID

$GLOBALS['phpgw_info']["user"]["theme"] = Selected theme

$GLOBALS['phpgw_info']["user"]["private_dir"] = Users private dir. Use eGroupWare core functions for access to the files.

$GLOBALS['phpgw_info']["user"]["firstname"] = Users first name

$GLOBALS['phpgw_info']["user"]["lastname"] = Users last name

$GLOBALS['phpgw_info']["user"]["fullname"] = Users Full Name

$GLOBALS['phpgw_info']["user"]["groups"] = Groups the user is a member of

$GLOBALS['phpgw_info']["user"]["app_perms"] = If the user has access to the current application

$GLOBALS['phpgw_info']["user"]["lastlogin"] = Last time the user logged in.

$GLOBALS['phpgw_info']["user"]["lastloginfrom"] = Where they logged in from the last time.

$GLOBALS['phpgw_info']["user"]["lastpasswd_change"] = Last time they changed their password.

$GLOBALS['phpgw_info']["user"]["passwd"] = Hashed password.

$GLOBALS['phpgw_info']["user"]["status"] = If the user is enabled.

$GLOBALS['phpgw_info']["user"]["logintime"] = Time they logged into their current session.

$GLOBALS['phpgw_info']["user"]["session_dla"] = Last time they did anything in their current session

$GLOBALS['phpgw_info']["user"]["session_ip"] = Current IP address

6.3 Group information

$GLOBALS['phpgw_info']["group"]["group_names"] = List of groups.

6.4 Server information

$GLOBALS['phpgw_info']["server"]["server_root"] = Main installation directory

$GLOBALS['phpgw_info']["server"]["include_root"] = Location of the 'inc' directory.

$GLOBALS['phpgw_info']["server"]["temp_dir"] = Directory that can be used for temporarily storing files

$GLOBALS['phpgw_info']["server"]["files_dir"] = Directory er and group files are stored

$GLOBALS['phpgw_info']["server"]["common_include_dir"] = Location of the core/shared include files.

$GLOBALS['phpgw_info']["server"]["template_dir"] = Active template files directory. This is defaulted by the server, and changeable by the user.

$GLOBALS['phpgw_info']["server"]["dir_separator"] = Allows compatibility with WindowsNT directory format,

$GLOBALS['phpgw_info']["server"]["encrpytkey"] = Key used for encryption functions

$GLOBALS['phpgw_info']["server"]["site_title"] = Site Title will show in the title bar of each webpage.

$GLOBALS['phpgw_info']["server"]["webserver_url"] = URL to eGroupWare installation.

$GLOBALS['phpgw_info']["server"]["hostname"] = Name of the server eGroupWare is installed upon.

$GLOBALS['phpgw_info']["server"]["charset"] = default charset, default:iso-8859-1

$GLOBALS['phpgw_info']["server"]["version"] = eGroupWare version.

6.5 Database information

It is unlikely you will need these, because $GLOBALS['phpgw_info']_db will already be loaded as a database for you to use.

$GLOBALS['phpgw_info']["server"]["db_host"] = Address of the database server. Usually this is set to localhost.

$GLOBALS['phpgw_info']["server"]["db_name"] = Database name.

$GLOBALS['phpgw_info']["server"]["db_user"] = User name.

$GLOBALS['phpgw_info']["server"]["db_pass"] = Password

$GLOBALS['phpgw_info']["server"]["db_type"] = Type of database. Currently MySQL and PostgreSQL are supported.

6.6 Mail information

It is unlikely you will need these, because most email needs are services thru core eGroupWare functions.

$GLOBALS['phpgw_info']["server"]["mail_server"] = Address of the IMAP server. Usually this is set to localhost.

$GLOBALS['phpgw_info']["server"]["mail_server_type"] = IMAP or POP3

$GLOBALS['phpgw_info']["server"]["imap_server_type"] = Cyrus or Uwash

$GLOBALS['phpgw_info']["server"]["imap_port"] = This is usually 143, and should only be changed if there is a good reason.

$GLOBALS['phpgw_info']["server"]["mail_suffix] = This is the domain name, used to add to email address

$GLOBALS['phpgw_info']["server"]["mail_login_type"] = This adds support for VMailMgr. Generally this should be set to 'standard'.

$GLOBALS['phpgw_info']["server"]["smtp_server"] = Address of the SMTP server. Usually this is set to localhost.

$GLOBALS['phpgw_info']["server"]["smtp_port"] = This is usually 25, and should only be changed if there is a good reason

6.7 NNTP information

$GLOBALS['phpgw_info']["server"]["nntp_server"] = Address of the NNTP server.

$GLOBALS['phpgw_info']["server"]["nntp_port"] = This is usually XX, and should only be changed if there is a good reason.

$GLOBALS['phpgw_info']["server"]["nntp_sender"] = Unknown

$GLOBALS['phpgw_info']["server"]["nntp_organization"] = Unknown

$GLOBALS['phpgw_info']["server"]["nntp_admin"] = Unknown

6.8 Application information

Each application has the following information available.

$GLOBALS['phpgw_info']["apps"]["appname"]["title"] = The title of the application.

$GLOBALS['phpgw_info']["apps"]["appname"]["enabled"] = If the application is enabled. True or False.

$GLOBALS['phpgw_info']["server"]["app_include_dir"] = Location of the current application include files.

$GLOBALS['phpgw_info']["server"]["app_template_dir"] = Location of the current application tpl files.

$GLOBALS['phpgw_info']["server"]["app_lang_dir"] = Location of the current lang directory.

$GLOBALS['phpgw_info']["server"]["app_auth"] = If the server and current user have access to current application

$GLOBALS['phpgw_info']["server"]["app_current"] = name of the current application.

7 Using Language Support

7.1 Overview

eGroupWare is built using a multi-language support scheme. This means the pages can be translated to other languages very easily. Translations of text strings are stored in the phpGroupWare database, and can be modified by the eGroupWare administrator.

Please see the setup/doc directory for a document which contains more complete documentation of the language system.

7.2 How to use lang support

The lang() function is your application's interface to eGroupWare's internationalization support.

While developing your application, just wrap all your text output with calls to lang(), as in the following code:

```
$x = 42;
echo lang("The counter is %1",$x)."<br>";
```

This will attempt to translate "The counter is %1", and return a translated version based on the current application and language in use. Note how the position that $x will end up is controlled by the format string, **not** by building up the string in your code. This allows your application to be translated to languages where the actual number is not placed at the end of the string.

When a translation is not found, the original text will be returned with a * after the string. This makes it easy to develop your application, then go back and add missing translations (identified by the *) later.

Without a specific translation in the lang table, the above code will print:

```
The counter is 42*<br>
```

If the current user speaks Italian, they string returned may instead be:

```
il contatore è 42<br>
```

The lang function

```
lang($key, $m1="", $m2="", $m3="", $m4="", $m5="",
          $m6="", $m7="", $m8="", $m9="", $m10="")
```

$key
: is the string to translate and may contain replacement directives of the form %n.

$m1
: is the first replacement value or may be an array of replacement values (in which case $m2 and above are ignored).

$m2 - $m10
: the 2nd through 10th replacement values if $m1 is not an array.

The database is searched for rows with a lang.message_id that matches $key. If a translation is not found, the original $key is used. The translation engine then replaces all tokens of the form %N with the Nth parameter (either $m1[N] or $mN).

Adding translation data

An application called **Transy** is being developed to make this easier, until then you can create the translation data manually.

The lang table

The translation class uses the lang table for all translations. We are concerned with 4 of the columns to create a translation:

message_id
: The key to identify the message (the $key passed to the lang() function). This is written in English.

app_name
: The application the translation applies to, or common if it is common across multiple applications.

lang
: The code for the language the translation is in.

content
: The translated string.

7.3 Common return codes

If you browse through the eGroupWare sources, you may notice a pattern to the return codes used in the higher-level functions. The codes used are partially documented in the doc/developers/CODES file.

Codes are used as a simple way to communicate common error and progress conditions back to the user. They are mapped to a text string through the check_code() function, which passes the strings through lang() before returning.

For example, calling

```
echo check_code(13);
```

Would print

```
Your message has been sent
```

translated into the current language.

8 Using Templates

8.1 Overview

eGroupWare is built using a templates-based design. This means the display pages, stored in tpl files, can be translated to other languages, made to look completely different.

8.2 How to use templates

Some instructions on using templates:

For Further info read the PHPLIBs documentation for their template class. `http://phplib.netuse.de`

9 About this document

9.1 New versions

The newest version of this document can be found on our website `http://www.egroupware.org` as lyx source, HTML, and text.

9.2 Comments

Comments on this HOWTO should be directed to the eGroupWare developers mailing list egroupware-developers@lists.sourceforge.net

To subscribe, go to `http://lists.sourceforge.net/lists/egroupware-developers`

9.3 History

This document was written by Dan Kuykendall.

2000-09-25 documentation on lang(), codes, administration and preferences extension added by Steve Brown.

2001-01-08 fixed directory structure, minor layout changes, imported to lyx source - Darryl VanDorp

2003-12-29 adapted for eGroupWare and updated for setup and use of GLOBALS - Miles Lott

9.4 Copyrights and Trademarks

9.5 Acknowledgments and Thanks

Thanks to Joesph Engo for starting phpGroupWare (at the time called webdistro). Thanks to all the developers and users who contribute to making eGroupWare and phpGroupWare such a success.