

phpGroupware IPC Interface

Version: 2004-05-12

Author: Dirk Schaller <dschaller@probusiness.de>

Inhaltsverzeichnis

1. IPC Interface.....	2
1.1 Concept and Design.....	2
1.2 How to.....	4
2. IPC in the phpGW Applications.....	5
2.1 Addressbook.....	5
2.2 Bookmarks.....	6
2.3 Calendar.....	7
2.4 Notes.....	8
2.5 Todo.....	9
2.6 Infolog.....	10
2.7 Email.....	11
2.8 Projects.....	12
3. IPC from outside phpGW (XML-RPC and IPC Methods).....	13
3.1 system.login.....	13
3.2 system.logout.....	15
3.3 phpgwapi.ipc_manager.execIPC.....	16
3.4 <phpgw_application>.addData.....	17
3.5 <phpgw_application>.getData.....	18
3.6 <phpgw_application>.removeData.....	19
3.7 <phpgw_application>.replaceData.....	20
3.8 <phpgw_application>.existData.....	21
3.9 <phpgw_application>.getIdList.....	22

1. IPC Interface

1.1 Concept and Design

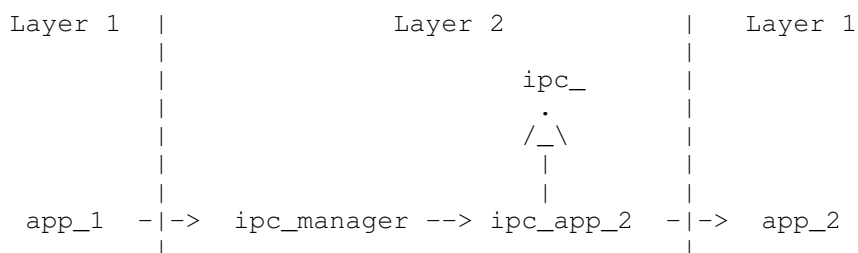
Layer 1 (applications):

```
class ipc_"appName" (implemented for each application)
+ addData(data, type, version) : id
+ getData(id, type, version) : data
+ getIdList(lastmod, restriction) : array
+ removeData(id) : boolean
+ replaceData(id, data, type, version) : boolean
+ existsData(id) : boolean
(methods above are the implementation of the abstract methods)
+ ... (other ipc methods, everything you like)
```

Layer 2 (API):

```
class ipc_manager
+ getIPC(appName) : object
+ execIPC(ipcAppMethod, ipcAppMethodParams) : mixed
+ destroyIPC(appName) : boolean
- _checkIPCApp($appName) : boolean
- _createIPCAppClassName($appName) : string
```

```
class ipc_
<abstract>
+ addData(data, type, version) : id
+ getData(id, type, version) : data
+ getIdList(lastmod, restriction) : array
+ removeData(id) : boolean
+ replaceData(id, data, type, version) : boolean
+ existData(id) : boolean
Methods are abstract and have to be implemented in the ipc_<appName>
class.
```



The ipc_manager handles the application call (check if the called app2 is available and if acl rights are okay) and create the ipc_app_2 object. This object contains the ipc methods for add/get/... data from app_2.

The type parameter of the ipc methods specifies the mime type of the passed data or result data. That's because only the application knows her database structures and data records. So the application transforms the data result to the applied mime type format. For example we want export an calendar entry as iCal. For do this the calendar application has to provide an ipc getData method which "implements" the database queries (use the storage object) and the db result transformation to a iCal. On the other hand there is also the need for import transformations. All the transformations can be done by using available classes of the application and api.

It's possible to transfer any type of data over the ipc layer. The only need is to implement the data conversion between the application data structures and the supported mime type.

Each application implements her own ipc_<appName> class. This class is derived from the abstract ipc_ class of the api. Physically the ipc_app_2 class belongs to the app_2 and is located in the inc-directory of app_2.

An applicaion ipc class should contain a set of methods for import and export data in useful mime type formats. I thinks it is clear that the applications can use the api if there is a solution for converting the data to another mime type. Each application developer can decide which import/export mime types his application supports.

1.2 How to

Now take a look at a pseudo implemation for discussion (no error handling, not optimised):

in app_1:

```
$ipc_manager = CreateObject('phpgwapi.ipc_manager');
$ipc_app_2 = $ipc_manager->getIPC('app_2');
$result = $ipc_app_2->getData(123, 'x-phpgroupware/app_2');
```

in app_2:

```
class ipc_app_2 extends ipc_
function getData($id, $type, $version)
{
    // 1: read record from application bo
    $bo = CreateObject('app_2.app_2_bo');
    $result = $bo->get...($id);

    // 2: convert data to passed mime type
    switch ($type)
    {
        case 'x-phpgroupware/app_2':
            // transfer result to application data structure
            ...
            return $converted_result;
        break;
        case 'text/xml':
            // convert result to xml
            ...
            return $converted_result;
        break;
        case 'text/csv':
            // convert result to csv
            ...
            return $converted_result;
        break;
        case '...':
            // convert result to ...
            ...
            return $converted_result;
        break;
        default:
            return false;
        break;
    }
}
```

The interprocess communication layer provides the oppertunaty to access an application. But the layer doesn't know anything about the data which will be "send" over this layer. Only the application knows her internal data representation and can provide the data in certain mime types. The called application delivers the data in the applied mime type.

2. IPC in the phpGW Applications

2.1 Addressbook

class.ipc_addressbook.inc.php

When using the vcard mime type the version specified the vcard version. This can be '2.1' or '3.0'.

supported mime types:

addData() / replaceData()

text/x-vcard	string	a vcard
text/vcard	string	a vcard
x-phpgroupware/addressbook-ldap	array	a ldap entry

getData()

text/x-vcard	string	a contact as vcard
text/vcard	string	a contact as vcard
text/xml	string	a contact as xml structure
x-phpgroupware/search-index-data-item	object	a dom xml object

Ldap fields (for array structure see php ldap functions):

cn, sn, surname, givenname, initials, title, personaltitle, distinguishedname (ou, o), department, postaladdress, postofficebox, postalcode, l, locality, countryname, textencodedoraddress (s, n, i), facsimiletelephonenumber, homephone, telephonenumber, mail, rfc822mailbox, othermailbox, mobile, internationalisdnumber, pager, homepage, url

used classes:

phpgwapi.contacts	read/save contact data
phpgwapi.vcard	import/export vcard
phpgwapi.validator	check email format
search.index_xml*	create the dom xml structur for search engine

getIdList restrictions:

syncable returns only person contact ids

searchabel returns both person and org contact ids without check owner

2.2 Bookmarks

class.ipc_bookmarks.inc.php

supported mime types:

addData() / replaceData()

x-phpgroupware/bookmarks	array	list of bookmark fields
--------------------------	-------	-------------------------

getData()

x-phpgroupware/bookmarks	array	a bookmark as array
--------------------------	-------	---------------------

x-phpgroupware/search-index-data-item	object	a dom xml object
---------------------------------------	--------	------------------

bookmark fields:

bookmark_id, bookmark_url, bookmark_title, bookmark_description,
bookmark_keywords, bookmark_rating, bookmark_visits, bookmark_access,
bookmark_owner, bookmark_category, bookmark_timestamp_added,
bookmark_timestamp_lastvisit, bookmark_timestamp_lastmod

used classes:

bookmarks.bo	read/save bookmark data
--------------	-------------------------

search.index_xml*	create the dom xml structur for search engine
-------------------	---

2.3 Calendar

class.ipc_calendar.inc.php

supported mime types:

addData() / replaceData()

text/x-ical	string	a iCalendar message
text/calendar	string	a iCalendar message

getData()

text/x-ical	string	a date as iCalendar
text/calendar	string	a data as iCalendar
x-phpgroupware/search-index-data-item	object	a dom xml object

used classes:

calendar.bocalendar	read/save calendar data
calendar.boicalendar	import/export ical
search.index_xml*	create the dom xml structur for search engine

2.4 Notes

class.ipc_notes.inc.php

supported mime types:

addData() / replaceData()

text/x-vnote	string	a vnote
x-phpgroupware/notes	array	a note entry

getData()

text/x-vnote	string	a note as vnote
x-phpgroupware/notes	array	a note as array
x-phpgroupware/search-index-data-item	object	a dom xml object

note fields:

note_id, note_owner, note_access, note_createdate, note_category,
note_description

used classes:

notes.bonotes	read/save note data
search.index_xml*	create the dom xml structur for search engine

2.5 Todo

class.ipc_todo.inc.php

supported mime types:

addData() / replaceData()

x-phpgroupware/todo	array	a todo entry
---------------------	-------	--------------

getData()

x-phpgroupware/todo	array	a todo as array
---------------------	-------	-----------------

x-phpgroupware/search-index-data-item	object	a dom xml object
---------------------------------------	--------	------------------

todo field:

todo_id, todo_id_parent, todo_id_main, todo_level, todo_title,
todo_description, todo_status, todo_priority, todo_category, todo_start_date,
todo_end_date, todo_create_date, todo_access, todo_owner

used classes:

todo.bo	read/save todo data
---------	---------------------

search.index_xml*	create the dom xml structur for search engine
-------------------	---

2.6 Infolog

class.ipc_infolog.inc.php

not implemented yet

supported mime types:

addData() / replaceData()

getData()

used classes:

2.7 Email

class.ipc_email.inc.php

not implemented yet

supported mime types:

addData() / replaceData()

getData()

used classes:

2.8 Projects

class.ipc_projects.inc.php

not implemented yet

supported mime types:

addData() / replaceData()

getData()

used classes:

3. IPC from outside phpGW (XML-RPC and IPC Methods)

3.1 system.login

XML-RPC method

3.1.1 Parameter

in	[USERNAME]	string	phpGroupware user account
in	[PASSWORD]	string	user account password
in	[DOMAIN]	string	domain name (optional)

on success

return	[SESSIONID]	string	username for http authentication
return	[KP3]	string	password for http authentication
return	[DOMAIN]	string	name of the used domain

on fail

return	false	boolean	login fail
--------	-------	---------	------------

3.1.2 Request

```
<?xml version="1.0" encoding="iso-8859-1"?>
<methodCall>
  <methodName>system.login</methodName>
  <params>
    <param>
      <value>
        <struct>
          <member>
            <name>domain</name>
            <value>
              <string>[DOMAIN]</string>
            </value>
          </member>
          <member>
            <name>username</name>
            <value>
              <string>[USERNAME]</string>
            </value>
          </member>
          <member>
            <name>password</name>
            <value>
              <string>[PASSWORD]</string>
            </value>
          </member>
        </struct>
      </value>
    </param>
  </params>
</methodCall>
```

3.1.3 Response

on success

```
<?xml version="1.0" encoding="iso-8859-1"?>
<methodResponse>
<params>
  <param>
    <value>
      <struct>
        <member>
          <name>sessionId</name>
          <value>
            <string>[SESSIONID]</string>
          </value>
        </member>
        <member>
          <name>kp3</name>
          <value>
            <string>[KP3]</string>
          </value>
        </member>
        <member>
          <name>domain</name>
          <value>
            <string>[DOMAIN]</string>
          </value>
        </member>
      </struct>
    </value>
  </param>
</params>
</methodResponse>
```

on fail

```
<?xml version="1.0" encoding="iso-8859-1"?>
<methodResponse>
<params>
  <param>
    <value>
      <boolean>0</boolean>
    </value>
  </param>
</params>
</methodResponse>
```

3.2 system.logout

XML-RPC method

3.2.1 Parameter

return true boolean logout successful

3.2.2 Request

```
<?xml version="1.0" encoding="iso-8859-1"?>
<methodCall>
  <methodName>system.logout</methodName>
  <params></params>
</methodCall>
```

3.2.3 Response

```
<?xml version="1.0" encoding="iso-8859-1"?>
<methodResponse>
  <params>
    <param>
      <value>
        <boolean>1</boolean>
      </value>
    </param>
  </params>
</methodResponse>
```

3.3 phpgwapi.ipc_manager.execIPC

XML-RPC method

3.3.1 Parameter

in [METHOD] string name of called application ipc method
[METHOD] is the string "<phpgw_application>.<ipc_method>" that specifies the name of the called application and her ipc method.
in [PARAM] struct parameter array for the ipc method
return [RETURN] mixed depends on the ipc method return

3.3.2 Request

```
<?xml version="1.0" encoding="iso-8859-1"?>
<methodCall>
  <methodName>phpgwapi.ipc_manager.execIPC</methodName>
  <params>
    <param>
      <value>
        <string>[METHOD]</string>
      </value>
    </param>
    <param>
      <value>[PARAM]</value>
    </param>
  </params>
</methodCall>
```

3.3.3 Response

```
<?xml version="1.0" encoding="iso-8859-1"?>
<methodResponse>
  <params>
    <param>
      <value>[RETURN]</value>
    </param>
  </params>
</methodResponse>
```


3.4 <phpgw_application>.addData

Add data in a certain mime type format to the application.

3.4.1 Parameter

in	[DATA]	mixed	data to add to the application
in	[TYPE]	string	mime type of the passed data
in	[VERSION]	string	mime type version (optional)

on success

return	[ID]	int	id of the added data
--------	------	-----	----------------------

on fail

return	false	boolean	add data fail
--------	-------	---------	---------------

3.4.2 Request

```
[PARAM] :  
<struct>  
  <member>  
    <name>0</name>  
    <value>  
      <...>[DATA]</...>  
    </value>  
  </member>  
  <member>  
    <name>1</name>  
    <value>  
      <string>[TYPE]</string>  
    </value>  
  </member>  
  <member>  
    <name>2</name>  
    <value>  
      <string>[VERSION]</string>  
    </value>  
  </member>  
</struct>
```

3.4.3 Response

on success

```
[RETURN] :  
<int>[ID]</int>
```

on fail

```
[RETURN] :  
<boolean>0</boolean>
```

3.5 <phpgw_application>.getData

Get data from the application in a certain mime type format.

3.5.1 Parameter

in	[ID]	int	id of data to get from the application
in	[TYPE]	string	mime type for the returned data
in	[VERSION]	string	mime type version (optional)

on success

return	[DATA]	mixed	application data in the mime format
--------	--------	-------	-------------------------------------

on fail

return	false	bool	get data fail e.g. data not exists
--------	-------	------	------------------------------------

3.5.2 Request

```
[PARAM] :  
<struct>  
  <member>  
    <name>0</name>  
    <value>  
      <int>[ID]</int>  
    </value>  
  </member>  
  <member>  
    <name>1</name>  
    <value>  
      <string>[TYPE]</string>  
    </value>  
  </member>  
  <member>  
    <name>2</name>  
    <value>  
      <string>[VERSION]</string>  
    </value>  
  </member>  
</struct>
```

3.5.3 Response

on success

```
[RETURN] :  
<...>[DATA]</...>
```

on fail

```
[RETURN] :  
<boolean>0</boolean>
```

3.6 <phpgw_application>.removeData

Remove the application data of the passed data id.

3.6.1 Parameter

in	[ID]	int	id of the data which will be deleted
return	[STATUS]	boolean	true if data deleted / false if failure

3.6.2 Request

```
[PARAM] :  
<struct>  
  <member>  
    <name>0</name>  
    <value>  
      <int>[ID]</int>  
    </value>  
  </member>  
</struct>
```

3.6.3 Response

on success

```
[RETURN] :  
<boolean>1</boolean>
```

on fail

```
[RETURN] :  
<boolean>0</boolean>
```

3.7 <phpgw_application>.replaceData

Replace the existing data of the passed id with the passed data. The passed data is in a certain mime type format.

3.7.1 Parameter

in	[ID]	int	id of the data which will be replaced
in	[DATA]	mixed	data to replace in the application
in	[TYPE]	string	mime type of the passed data
in	[VERSION]	string	mime type version (optional)
return	[STATUS]	boolean	true if data replaced / false if failure

3.7.2 Request

```
[PARAM] :
<struct>
  <member>
    <name>0</name>
    <value>
      <int>[ID]</int>
    </value>
  </member>
  <member>
    <name>1</name>
    <value>
      <...>[DATA]</...>
    </value>
  </member>
  <member>
    <name>2</name>
    <value>
      <string>[TYPE]</string>
    </value>
  </member>
  <member>
    <name>3</name>
    <value>
      <string>[VERSION]</string>
    </value>
  </member>
</struct>
```

3.7.3 Response

on success

```
[RETURN] :
<boolean>1</boolean>
```

on fail

```
[RETURN] :
<boolean>0</boolean>
```

3.8 <phpgw_application>.existData

Checks if the passed id exists in the application.

3.8.1 Parameter

in	[ID]	int	data id to check
return	[STATUS]	boolean	true if data exists else false

3.8.2 Request

```
[PARAM] :  
<struct>  
  <member>  
    <name>0</name>  
    <value>  
      <int>[ID]</int>  
    </value>  
  </member>  
</struct>
```

3.8.3 Response

on success

```
[RETURN] :  
<boolean>1</boolean>
```

on fail

```
[RETURN] :  
<boolean>0</boolean>
```

3.9 <phpgw_application>.getIdList

Return a list with the available id's in the application. The optional lastmod parameter allows a time limitation of the data id list. The list contains all the id's of the modified data since the passed lastmod timestamp. The optional second parameter allows a individual restriction of the id list. The restriction dedends on the module.

3.9.1 Parameter

in	[LASTMOD]	int	last modification time (optional)
in	[RESTRICTION]	string	restriction type (optional)
return	[IDLIST]	array	list of data id's

3.9.2 Request

```
[PARAM] :  
<struct>  
  <member>  
    <name>0</name>  
    <value>  
      <int>[LASTMOD]</int>  
    </value>  
  </member>  
  <member>  
    <name>1</name>  
    <value>  
      <string>[RESTRICTION]</string>  
    </value>  
  </member>  
</struct>
```

3.9.3 Response

```
[RETURN] :  
<struct>[IDLIST]</struct>
```