Users & Profiles

Joomla User Object

Current User

Accessing the Joomla! User object for the current user can be done as follows:

```
$juser =& JFactory::getUser();
```

Other Users

To access user info for anyone not the current user (accepts user ID number or username):

```
$otheruser =& JUser::getInstance($id);
```

Any field from the user database table may then be accessed through the get('fieldname') method:

```
$id = $juser->get('id');
$name = $juser->get('name');
```

Object Member Variables and Parameters

These are the relevant member variables automatically generated on a call to getUser():

- id - The unique, numerical user id. Use this when referencing the user record in other database tables.
- name - The name of the user. (e.g. Vint Cerf)
- username - The login/screen name of the user. (e.g. shmuffin1979)
- email - The email address of the user. (e.g. crashoverride@hackers.com)
- password - The encrypted version of the user's password
- password_clear - Set to the user's password only when it is being changed. Otherwise, remains blank.
- usertype - The role of the user within Joomla!. (Super Administrator, Editor, etc...)
- gid - Set to the user's group id, which corresponds to the usertype.
- block - Set to '1' when the user is set to 'blocked' in Joomla!.
- registerDate - Set to the date when the user was first registered.

- lastvisitDate - Set to the date the user last visited the site.
- guest - If the user is not logged in, this variable will be set to '1'. The other variables will be unset or default values.

In addition to the member variables (which are stored in the database in columns), there are parameters for the user that hold preferences. To get one of these parameters, call the getParam() member function of the user object, passing in the name of the parameter you want along with a default value in case it is blank.

```
$user =& JFactory::getUser();
$language = $user->getParam('language', 'the default');

echo "<p>Your language is set to {$language}.</p>";
```

HUBzero Extended Profile

HUBzero comes with extended user profiles that allow for considerably more information than the standard Joomla! User. Extended fields include information about disability, gender, race, bios, picture, etc. To access an extended profile, use the XProfile object and load() method (accepts user ID number or username).

```
// Import the needed library
ximport('Hubzero_User_Profile');

// Instantiate a new profile object
$profile = new Hubzero_User_Profile();

// Load the profile
$profile->load( $id );
```

Any field from the user database table may then be accessed through the get('fieldname') method:

```
$email = $profile->get('email');
$name = $profile->get('name');
```

Multi-option fields such as disability will return arrays.

Checking if a User is logged in

Checking if a user is currently logged in can be done as follows:

```
// Call the user object
$juser =& JFactory::getUser();

// If 'guest' is true, they are logged OUT
// If 'guest' is false, they are logged IN
if ($juser->get('guest')) {
 return false;
}
```

Privileges

Not all authenticated users are given equal rights. For instance, a Super Administrator may be able to edit anyone's content, while a Publisher may only be able to edit their own. The authorize() member function can be used to determine if the current user has permission to do a certain task. The first parameter is used to identify which component or function we wish to authenticate against. The second represents the task. The third and fourth are optional; they further break the permissions down into record types and ownership respectively.

In Joomla! 1.5, the rights for all of the core components are stored in libraries/joomla/user/authorization.php. These are available to all extensions wherever authentication is required. If the permission scheme of the Content component suits your extension's needs, you can use code similar to the following to determine what functions to give to a specific user.

```
$user =& JFactory::getUser();

if ($user->authorize('com_content', 'edit', 'content', 'all')) {
 echo "<p>You may edit all content.</p>";
} else {
 echo "<p>You may not edit all content.</p>";
}

if ($user->authorize('com_content', 'publish', 'content', 'own')) {
 echo "<p>You may publish your own content.</p>";
} else {
 echo "<p>You may not publish your own content.</p>";
}
```

The permissions for core functions may not be suitable for your extension. If this is the case, you can create your own permissions. You will probably want to add this code in a place where it will always be executed, such as the beginning of the component you are building or in a systemwide plugin. First, you need to get an authorization object using the getACL() member function of JFactory. This works like getUser() in that it only creates one authorization object during any particular Joomla! request. Once you have this object, call the addACL() member function to add permissions. Pass in the name of your component or function, the task name, the string 'users', and the user type (in lowercase) respectively. If you want to also define record sets and ownership, pass those in as an additional two parameters.

Note that in Joomla! 1.5, permissions are not inherited. For example, if you give an Administrator the right to edit content, Super Administrators do not automatically get this right; you must grant it separately.

```
$auth =& JFactory::getACL();

$auth->addACL('com_userinfo15', 'persuade', 'users', 'super administra
tor');
$auth->addACL('com_userinfo15', 'persuade', 'users', 'administrator');
$auth->addACL('com_userinfo15', 'persuade', 'users', 'manager');

$user =& JFactory::getUser();

if ($user->authorize('com_userinfo15', 'persuade')) {
 echo "<p>You may persuade the system to do what you wish.</p>";
} else {
 echo "<p>You are not very persuasive.</p>";
}
```

Group Memberships

Sometimes you may have a component or plugin that is meant to be accessed by members of a certain group or displays specific data based on membership in certain groups.

```
// Get the user object
$juser =& JFactory::getUser();

// Include a needed HUBzero library
ximport('Hubzero_User_Helper');

// Get the groups of the current logged-in user
$user_groups = Hubzero_User_Helper::getGroups( $juser->get('id') );
```

The getGroups() method is passed a user ID and returns an array of objects if any group memberships are found. It will return false if no group memberships are found. Each object contains data specifying the user's status within the group, among other things.

```
Array (
    [0] => stdClass Object (
  [published] => 1
  [cn] => greatgroup
  [description] => A Great Group
  [registered] => 1
  [regconfirmed] => 1
  [manager] => 0
 )
    [1] => stdClass Object (
  [published] => 1
  [cn] => mygroup
  [description] => My Group
  [registered] => 1
  [regconfirmed] => 1
  [manager] => 1
 )
)
```

- published - 0 or 1, the published state of the group
- cn - string, the group alias
- description - string, the group title
- registered - 0 or 1, if the user applied for membership to this group (only 0 if the user was invited)
- regconfirmed - 0 or 1, if the user's membership application has been accepted (automatically 1 for invitees)
- manager - 0 or 1, if the user is a manager of this group