Users & Profiles

User Object

Current User

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

```
$user = User::getInstance();
```

Other Users

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

```
$otheruser = User::getInstance($id);
```

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

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

Data for the **current logged-in user only** may also be retrieved through the User facade with a slightly terser syntax. Usage of all facades is global and can be called in the following manner from anywhere within the CMS:

```
$id = User::get('id');
$name = User::get('name');
```

Again, the above technique only applies to the current logged-in user. Any other user's information must use the User::getInstance($id); method.

Object Member Variables and Parameters

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

- **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. Jane Doe)
- **username** - The login/screen name of the user. (e.g. janedoe2015)
- **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.
- **block** - Set to '1' when the user is set to 'blocked'.
- **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.

```
$language = User::getParam('language', 'the default');

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

User Profile

HUBzero comes with extended user profiles that allow for considerably more information than the standard User. Extended fields include information about disability, gender, race, bios, picture, etc. To access an extended profile, use the User object. Any field from the user database table may then be accessed through the get('fieldname') method:

```
$profile = User::getInstance($id);
$bio = $profile->get('bio');
$gender = $profile->get('gender');
```

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 by calling the isGuest() method on the global User facade:

```
// If true, they are logged OUT
// If false, they are logged IN
if (User::isGuest())
{
 return false;
}
```

Alternatively, one may need to work with a user object more directly:

```
// Get the root object behind the facade
$user = User::getInstance();

// ... Do some processing on the $user

if ($user->isGuest())
{
 return false;
}
```

The isGuest() method checks the guest property on the user object. This property can be directly accessed, if desired:

```
// If true, they are logged OUT
// If false, they are logged IN
if (User::get('guest'))
{
 return false;
}
```

Access Groups

There are cases where you may need to retrieve the specific access groups a user is assigned.

These access groups determine what permissions the user has throughout the CMS

```
// Get the groups of the current logged-in user
$accessgroups = User::accessgroups();
```

This returns a HubzeroDatabaseRows object that can be iterated or counted.

```
// Get the groups of the current logged-in user
$accessgroups = User::getInstance(1000)->accessgroups();

foreach ($accessgroups as $accessgroup)
{
    // Do something
}
```

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 groups of the current logged-in user
$user_groups = HubzeroUserHelper::getGroups(User::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