Server-side function

HasObjectPermissionTo

This function returns whether or not the given object has access to perform the given action.

> **Note:** Only certain action names work. This function seems to return *nil* and output a bad argument error when checking if an object has rights for an action which doesn't start with *function.*, *command.* or *resource.* keywords.

Scripts frequently wish to limit access to features to particular users. The naïve way to do this would be to check if the player who is attempting to perform an action is in a particular group (usually the Admin group). The main issue with doing this is that the Admin group is not guaranteed to exist. It also doesn't give the server admin any flexibility. He might want to allow his 'moderators' access to the function you're limiting access to, or he may want it disabled entirely.

This is where using the ACL properly comes in, and luckily this is very easy. It all comes down to using this function. This, somewhat confusingly named function lets you check if an ACL object (a player or a resource) has a particular ACL right. In this case, we just care about players.

So, first of all, think of a name for your 'right'. Let's say we want a private area only certain people can go in, we'll call our right accessPrivateArea. Then, all you need to do is add one 'if' statement to your code:

```
if hasObjectPermissionTo ( player, "resource.YourResourceName.accessPrivateArea", false ) then
-- Whatever you want to happen if they're allowed in
else
-- Whatever you want to happen if they aren't
end
```

Notice that we've named the *right* using *resource.YourResourceName.accessPrivateArea* - this is just for neatness, so that the admin knows what resource the right belongs to. It's strongly advised you follow this convention. The *false* argument specifies the 'defaultPermission', false indicating that if the user hasn't had the right allowed or dissallowed (i.e. the admin hasn't added it to the config), that it should default to being not allowed.

The only downside of using this method is that the admin has to modify his config. The upsides are that the admin has much more control and your script will work for any server, however the admin has configured it.

Syntax

```
bool hasObjectPermissionTo ( string / element theObject, string theAction [, bool defaultPermission = true ] )
```

OOP Syntax Help! I don't understand this!

> **Note**: *This function is also a static function underneath the ACL class.*
> **Method**: *ACL.hasObjectPermissionTo(...)*

Required Arguments

- **theObject:** The object to test if has permission to. This can be a client element (ie. a player), a resource or a string in the form "user.<name>" or "resource.<name>".
- **theAction:** The action to test if the given object has access to. Ie. "function.kickPlayer".

Optional Arguments

NOTE: When using optional arguments, you might need to supply all arguments before the one you wish to use. For more information on optional arguments, see optional arguments.

- **defaultPermission:** The default permission if none is specified in either of the groups the given object is a member of. If this is left to true, the given object will have permissions to perform the action unless the opposite is explicitly specified in the ACL. If false, the action will be denied by default unless explicitly approved by the Access Control List.

Returns

Returns *true* if the given object has permission to perform the given action, *false* otherwise. Returns *nil* if the function failed because of bad arguments.