# Adding Users to Projects

The REST interface enforces a simple security model. In this model, a single user may be associated with multiple projects, and each project will have multiple users associated with it. In every case, these associations will have one of two access levels:

1. Read-only access: The user can see the list of objects in the project, but cannot create, modify or delete objects. The user will also be able to load the views in the project and see the visualizations. Once the user has loaded a view, the user will be allowed to modify the view locally (for instance, changing filters, color mappings, adding and removing objects) but the user will not be allowed to save any changes made to the view. The user will not be allowed to invite users to the project, nor will be allowed to modify access levels for existing users.
2. Full-access: The user will be allowed to perform all operations in the project, including inviting other users and modifying access levels of existing users.

## PROJECT_REF and USER_REF entities

A single user-project association requires two different entities in the system:

1. An entity of type "PROJECT_REF", which links the project to the user
2. An entity of type "USER_REF", which links the user to the project

Both entities have a property named "access_level", which can have one of three values:

1. "none" -- Used to block access, it is the default value in case the property or object does not exit
2. "read" -- Used for read-only access
3. "full" -- Used for full access

It is up to the application to ensure the "access_level" property has the same value for any pair of PROJECT_REF and USER_REF entities. If the application fails to enforce this symmetry, this could lead to unpredictable behavior as API calls will be checking different values depending on the context of these calls.

The following diagram illustrates a scenario in which one user is assigned to two different projects:

## Understanding collections

The REST entity model supports the concept of a collection. A single entity may contain multiple collections. Like single-value properties in the entity, collections have a unique name in the entity. For instance, the "users" collection in a "PROJECT" entity will list all the "USER_REF" entities that belong to that particular project entity. There is also one default collection for every entity, which contains at least the entity. The default collection is identified by an empty string.

In fact, the call described in the Getting all Project Entities section, requests the default collection for a given project entity. In the case of projects, the default collection contains all spatial entities contained by the project. The call to retrieve the default collection for the project with "myproject" ID would be:

http://localhost:58697/entity.ashx?project=myproject

This will return all entities contained by the default collection, including the project entity itself.

If instead, we would like to access the user collection for the project, the REST call would look like this:

http://localhost:58697/entity.ashx?project=users:myproject

Here we have added the "users:" term to the ID of the project. Instead of requesting the default collection, we are requesting the entities in the "users" collection.

Note that the "project" parameter is still used in this call, but this parameter in fact denotes a collection name, not a project ID anymore. The "project" parameter would also be used if the entity holding the collection was not a project, in the context of this API, the property represents a collection. The name "project" was used for the sake of simplicity, since for most use cases, projects are the only collections the API user needs to use.

The same applies to the "project" property in entities. This property must contain the name of the collection that owns the entity. In most cases, this will be an actual project entity.

Users are collections too in this model. The default collection for a user contains all PROJECT_REF entities.

## Adding USER_REF entities

Entities of USER_REF type bind users to projects. These entities are contained in the "users" collection for the project. The following table lists the expected properties for this entity, and the rules for their values:

| | |
|---|---|
| ID | USER_REF entities will have their id composed from three parts:<br><br>users:<ProjectID>:<UserID><br><br>Here "ProjectID" should be replaced by the ID of the project. "UserID" should be replaced by the ID of the user, which is the User Id used to login to the application converted to a Base64 string. |
| type | Set to "USER_REF" |
| access_level | Set to "read" to provide read-only access, set to "full" to provide full access. |
| project | Contains the identifier for the collection that owns this entity. The collection identifier is composed in the following way:<br><br>users:<ProjectID><br><br>Here "ProjectID" should be replaced by the ID of the project. |
| user_ref | Contains the user ID as a Base64 value. |

## Adding PROJECT_REF entities

Entities of PROJECT_REF type bind projects to users. These entities are contained in the default collection for the user entity. The following table lists the expected properties for PROJECT_REF entities, and the rules for their values:

| | |
|---|---|
| ID | PROJECT_REF entities will have their id composed from three parts:<br><br>users:<ProjectID>:<UserID><br><br>Here "ProjectID" should be replaced by the ID of the project. "UserID" should be replaced by the ID of the user, which is the User Id used to login to the application converted to a Base64 string. |
| type | Set to "PROJECT_REF" |
| access_level | Set to "read" to provide read-only access, set to "full" to provide full access. |
| project | Contains the identifier for the collection that owns this entity, which in this case is the user's default collection (not a project entity). The collection identifier in this case is equal to the user ID converted to Base64. |
| project_ref | Contains the identifier of the project entity that will be bound to the user. |

Revision #2
Created 17 March 2025 16:24:08 by admin
Updated 17 March 2025 16:27:02 by admin