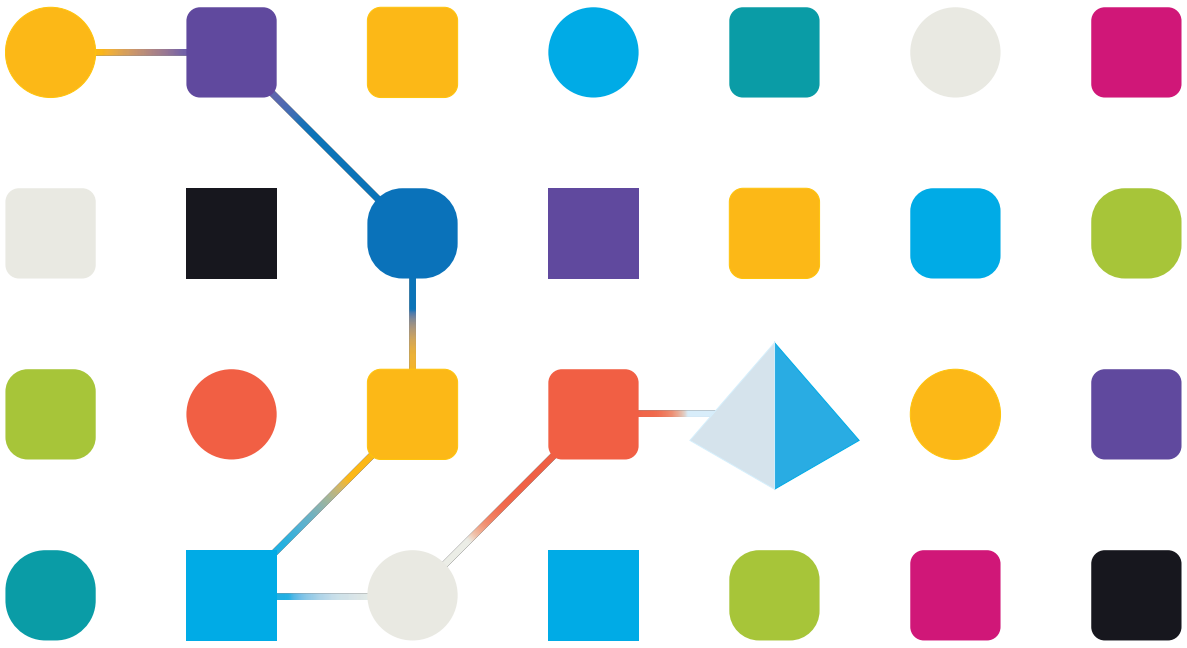
# Blue Prism 7.1

## Multi-team Environments

Document Revision: 1.0



# Trademarks and Copyright



Blue Prism Limited, 2 Cinnamon Park, Crab Lane, Warrington, WA2 0XP, United Kingdom.
Registered in England: Reg. No. 4260035. Tel: +44 370 879 3000. Web: www.blueprism.com

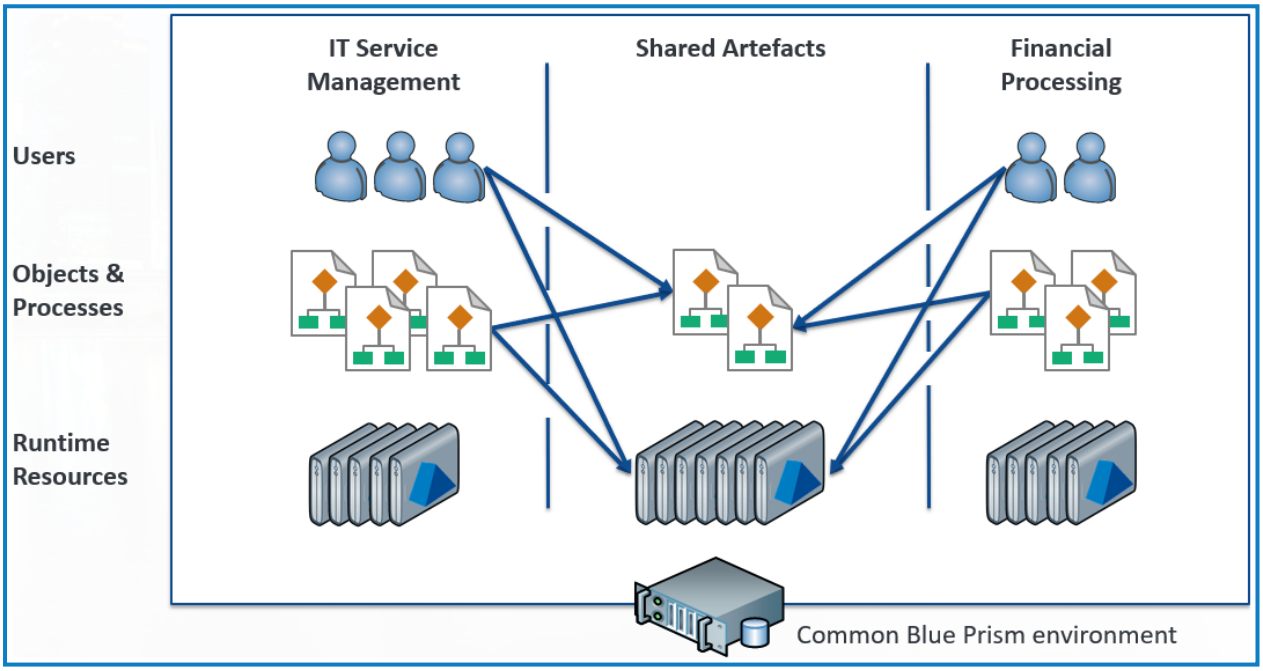
# Contents

# Multi-team environments

Multi-team Environments provide a way for multiple lines of business, operating within a single deployment of Blue Prism, to share artifacts. This is achieved by extending role-based access controls to enable more granular configurations.

These capabilities better enable organizations to share Blue Prism assets, such as business objects and runtime resources, with multiple teams within a given Blue Prism environment by allowing permissions to be assigned, not only based on the type of asset, but also based on the hierarchical structure of those assets.

For example, users who are members of a team may have full access to some business objects but may only have the ability to view or execute others, as may be appropriate for assets that are shared by multiple teams.




The list below shows where multi-team environments are effective in the Blue Prism interactive client:

- **Home** – The data displayed on the Home page is not filtered by the access rights applied to groups.
- **Studio** – This is where users with the appropriate permissions manage access rights for process groups and object groups.
- **Control** – Users are given a custom view of the Control Room so they only see and interact with the processes and resources appropriate to their role. This does not include the Queue Management area – users with access to this area can view information related to any conventional queue within the environment.

  > For active queues, users only see a queue if they have execute permission on the process that works the queue and control permission on all target resources for that queue.

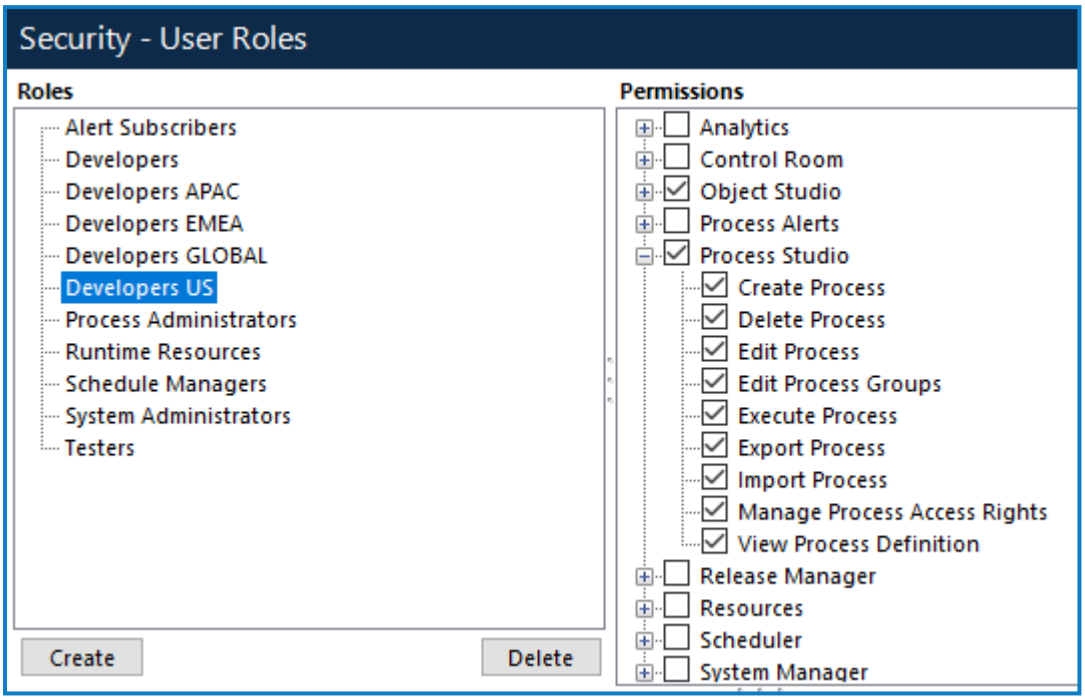
- **Analytics** – The data displayed in dashboard tiles is not filtered by the access rights applied to groups.
- **Releases** – Only items to which a user's permissions allow access, can be included in a release or package.

- **System** – The following areas of System Manager have been updated for multi-team environments:
  - **Resources** – This is where users with the appropriate permissions manage access rights for runtime resource groups.
  - **Audit (Session Logs)** – Process logs and object logs are subject to the logged in user's permissions – they can only see the logs for the items that their role allows. Audit logs are not filtered by the access rights applied to groups.

The Find References and View and Compare operations are also updated so they only return objects that the user has access to.

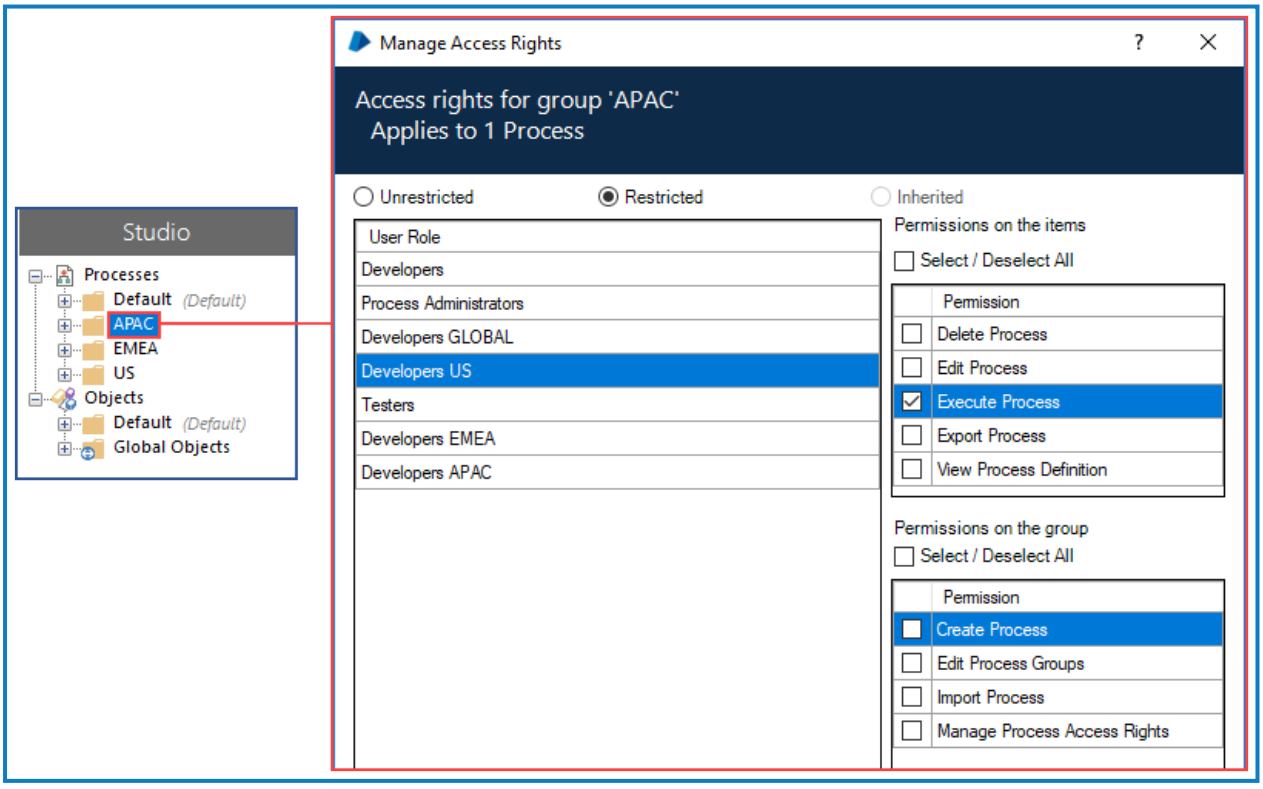
## Access based on hierarchical group membership

If group level permissions are not defined, users granted permissions for a particular item type, such as processes, have the same permissions for every item of that type in the system.




By restricting permissions to a group of processes, objects, or resources, the permissions defined by user roles can be configured to give the required access rights to the items in the group.

This example shows the permissions applied to the APAC process group for users assigned the Developers US role. The access rights for this group have been restricted to only allow users with the Developers US role to execute processes.



Permissions applied at group level cannot grant users access rights beyond the maximum permitted by their role. For example, if a role allows a user to edit processes, that permission can be removed for a group. If the role does not allow a user to delete processes, that permission cannot be granted for that role at group level.

# Configure access rights

System Administrators can configure access rights for all processes, objects, and resources in an environment. This ability can be delegated to other users by applying the *Manage Access Rights* permission to their role for specific groups.

Changes to access rights are applied at login so it is recommended that users are logged out when permissions are updated.

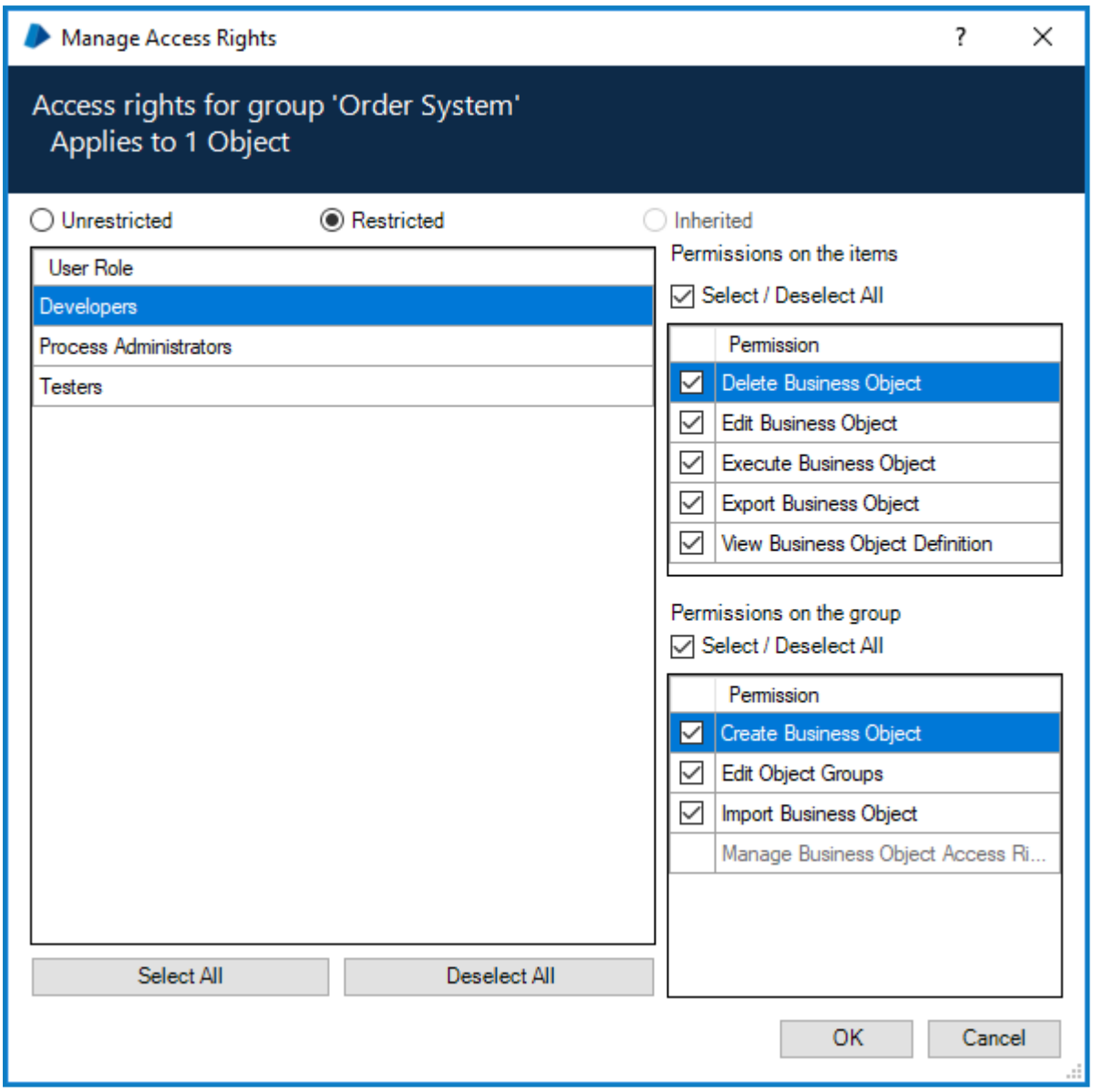
When configuring multi-team environments, it is important to first define the maximum capabilities for each role using the Security role permissions.

Click **System** > **Security** > **User Roles** and select a role to configure its permissions.

Once the maximum capabilities are set for each role, configure access rights for particular process, object, and resource groups to limit access based on the requirements of the user:

- To configure access rights for processes and objects, click **Studio**, right-click on a process group or object group and select **Access Rights**.
- To configure access rights for resources, click **System** > **Resources** > **Management**, right-click on a resource group and select **Access Rights**.

The Manage Access Rights dialog lists all user roles that have Security permissions to access that particular area of the product.




The built-in roles, System Administrators and runtime resources, are able to access all restricted and unrestricted items secured under multi-team environments. Users with these roles have access to all items regardless of their assigned permissions. As a result, these roles are not available as an option in the Manage Access Rights dialog.

A group can be in one of the following three states:

- **Unrestricted** – As access rights have not been specifically determined for the group they are determined by the permissions set in the roles assigned to the user. This is the default setting for all new and pre-existing groups. Unrestricted groups are identified by an icon overlay .
- **Restricted** – Access rights have been configured for the group, refining the user role permissions for all items in the group, including any child groups and their contents. The access rights defined for restricted groups are impacted by subsequent changes to role permissions – remove a permission for a role and it is removed for that role on all groups.

- **Inherited** – The item is already in a restricted group and its access rights are set by a parent group – when permissions are set for a group, they are automatically applied to all child groups. This could be the direct parent, or any higher-level folder as access rights are taken from the highest level restricted parent group. The name of the group from which the access rights are inherited from is shown alongside the *Inherited* radio button.

If there is a requirement to prevent users from seeing processes, objects, or resources in a particular group, remove all permissions on all groups for the required role. The items in all such restricted groups will not be visible to users assigned that role anywhere in Blue Prism and will cause processing to fail if they try to run those items or ones that reference them.

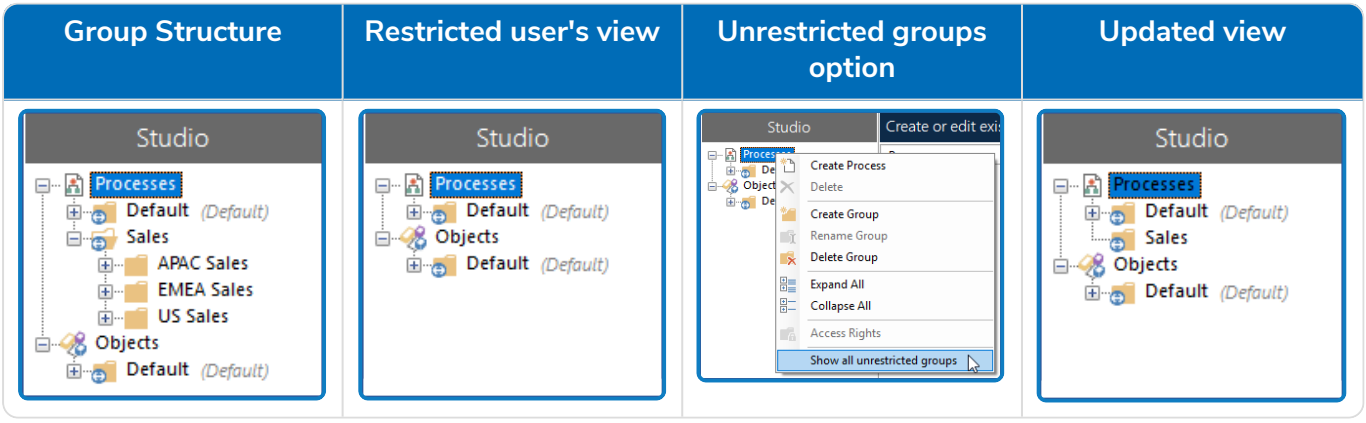
For an example of this scenario, see Dedicated process access.

## Show all unrestricted groups

If an unrestricted object or process group contains only restricted child groups (i.e. does not directly contain any objects, processes, or any unrestricted groups), users that do not have permissions to view any of the child groups will not see the unrestricted parent folder within the hierarchy. This streamlines the user's view by omitting the groups that they are unlikely to need to access from the navigation tree.

The display can be toggled to show unrestricted parent groups by selecting **Show all unrestricted groups** from the context menu in Studio. Once selected, unrestricted parent groups are visible but any restricted child groups remain hidden. This option applies to all groups in Studio for the duration of the user's session – if users log out and back in again, the option is reset to the default setting and unrestricted parent groups with restricted child groups are not displayed.

In the example below, the unrestricted *Sales* group contains three restricted groups that can be viewed only by the users responsible for the processes in their geographical location. When a user who is not part of the Sales team and therefore does not have the required permissions on the restricted groups logs in, the *Sales* group is not visible. When that user selects **Show all unrestricted groups** the *Sales* group displays but does not show the restricted child groups.



# Access rights for processes and objects

Access rights for processes and objects are split into two types – item and group.

## Item permissions

These permissions apply to items in a group and determine what users can do with those items.

| Permission | Description |
|---|---|
| Delete | Users can delete items in the group. |
| Edit | Users can edit items in the group. *Execute* and *View* permissions are also assumed. |
| Execute | Users can execute items but cannot open or edit them. Execute does not automatically grant permission to view definitions, this would require Edit permissions. Users with the Execute permission who do not have View Definition or Edit permissions cannot step into the item while debugging an object or process. |
| Export | Users can export processes, objects and releases if they have the access rights to do so. |
| View Definition | Users can view an object or process but cannot run or edit it. This also prevents users from successfully running a process that references an object for which they only have the right to view the definition. Providing the user has the permissions to view and run the process, it will stop running only when it reaches a prohibited object. |
| Execute as Web Service | Users can call an object or process that is exposed as a web service |

## Group permissions

These permissions determine what actions users can perform in relation to a group.

| Permission | Description |
|---|---|
| Create | Users can create items in the group, including using Save As in the editor. Edit, Execute, and View permissions are also assumed. |
| Edit Groups | Users can create, rename, move and delete groups within the object and process Studio trees. Users can also create and edit groups at the root level and in other unrestricted groups. This permission also allows users to retire and unretire processes if they also have the Manage Access Rights permission.<br>Both Manage Access Rights and Edit Groups permissions are required on any Restricted Group involved in a move. For more information, see Moving Groups. |
| Manage Access Rights | Users can edit the access rights for the roles that have access to the group, to refine the permissions that the roles allow. A user with this permission cannot grant access rights that are denied by the permissions set for a user role. Users cannot update this permission for their own user role. This prevents a user giving themselves access rights management to a group which they have been previously denied those rights. |

# Access rights for resources

The following permissions are available for resources and can be configured for user roles in System Manager and using Multi-Team Environments.

| Permission | Description |
|---|---|
| Authenticate as Resource<br><br>(Only available as a permission for user roles) | Allows the user account to be used for runtime authentication when starting a resource. User accounts that do not have this permission cannot be used to start a resource that explicitly authenticates against the environment.<br><br>When upgrading from versions prior to 6.3, all existing roles are granted this permission to ensure that they can continue to start existing resources. New roles and all roles in new installations are not granted this permission automatically, it must be actively applied.<br><br>Users can debug and start the resource that can optionally be automatically started locally when they log into the Blue Prism client. |
| Configure Resource | Users can edit a resource in System Manager to reset a resource's FQDN and change a resource's logging options. Users can also retire and unretire resources providing they also have the *Control Resource* permission. The *View Resource* and *Manage Resource Access Rights* permissions are implicitly granted. For appropriate configurations, this permission is required to reset a resource's FQDN.<br><br>To add a resource to an active work queue, users require a minimum of the access rights provided by the *Control Resource* permission on the related resource group – *Configure Resource* does not allow users to add a resource to an active queue |
| Control Resource | Users can create, start, stop, and delete sessions in Control Room. Once a session is running, users with this permission can fully interact with it in the Control Room to perform tasks such as starting the process or sending terminate or stop requests.<br><br>Users with this permission are implicitly granted the *View Resource* permission. |
| Edit Resource Group | Users can create, edit, delete, and rename groups in System management. They can also configure the resources hierarchy and move resources between groups.<br><br>Both *Manage Access Rights* and *Edit Groups* permissions are required on any restricted group involved in a move. For more information, see Moving groups. |
| Manage Resource Access Rights | Users can manage the access rights for the group providing they have *Manage Access Rights* enabled in their role. Users with this permission cannot update it for their own user role. This prevents a user from giving themselves access rights management to a group they had previously been denied. The *View Resource* permission is implicitly granted. |
| View Resource | Users can see resources in Control Room and System Management and can view session management data relating to those resources. This permission also allows some queries to be made via the telnet/HTTP interface. |
| View Resource Screen Capture | Users can view exception screen captures, assuming they have access to control room. The *View Resource* permission is implicitly granted. |

In order to retire a resource, *Configure Resource*, *Edit Resource Group*, and *Manage Resource Access Rights* permissions are required on the group containing the resource.

## Runtime resources role

When instructed to carry out a session management action, the runtime resource inherits the permissions of the controlling user by default. The runtime resources role provides the necessary and additional permissions required by the runtime resource to perform as required.

The runtime resources role must only be granted to user accounts that will be used exclusively to authenticate runtime resources against the Blue Prism environment. Users that interact with the platform for other purposes (i.e. via the interactive client or via scripting) must not be granted this role. Where users need the ability to start a runtime resource using their own credentials, they will require the Authenticate as Resource permission.

Resources that were started without explicitly authenticating against the environment (not recommended) assume the runtime resource role automatically.

# Administrative permissions

There are a number of administrative actions that are secured with specific roles. Users with these roles require access to certain record types across the system, irrespective of their group membership. Therefore, users with these administrative permissions might have access beyond what is configured within the group-level access rights.

For example, if a user has the *System* > *Archiving* permission, they can perform archiving activities for all sessions irrespective of the user's rights to the processes or resources to which the session data relates.

The following administrative permissions are not affected by those set at group level:

- **Importing Business Object** – Users can import objects and groups anywhere in the business object tree structure, from the File menu or the Releases tab. The location of each incoming items is determined by its position in the structure being imported.
- **Importing Process** – Users can import processes and groups anywhere in the process tree structure from the File menu or the Releases tab. The location of each incoming items is determined by its position in the structure being imported.
- **Releases** – When viewing packages, users can see all objects and processes regardless of their group permissions. However, restrictions do apply when configuring packages and generating release packages:
  - **Creating packages** – Users cannot include items they do not have permissions to access.
  - **Modifying packages** – Users can modify a package that contains items that they do not have the permissions to access. However, those items will be removed and the user cannot add those items again whilst their permissions prevent access.
  - **Creating releases** – Users cannot export a release that includes packages containing items that they do not have the permissions to access.
- **System** – **Archiving**– Archiving is not affected by permissions applied to groups.
- **System** – **Reporting**– Reporting is not affected by permissions applied to groups.
- **Analytics** – The data displayed in dashboard tiles is not filtered by the access rights applied to groups.

# Importing releases, objects, and processes

When importing a release, process or object, the following rules apply:

- *Import Release*, *Import Process*, and *Import Business Object* are administrative permissions that allow items to be imported anywhere in the target structure.
- The permissions applied to restricted groups in the target structure are inherited by any child group and item created in a restricted group during an import.
- New root-level groups and those added to unrestricted groups during an import, are created unrestricted.
- Where items with the same name or internal ID are present in both source and target structures, these are highlighted during an import and options are provided for their resolution.
- See Conflict resolution for further details.

## Import permissions

The following role and group permissions required to import releases, processes, and objects.

| Action | Required role or group permissions |
|---|---|
| Import a release | Release Manager > Import Release |
| Import objects or processes individually or as part of a release. | Object Studio > Import Business Object<br>Process Studio > Import Process |

The permissions are cumulative so, to import a release, users would need to be assigned to a role that granted the following permissions:

- Release Manager > Import Release
- Process Studio > Import Process

### Permissions required to overwrite objects and processes on import

Depending on the version of Blue Prism, *Edit Business Object* and/or *Edit Process* permissions may additionally be required to import objects and processes that overwrite items in the target structure.

The table below shows which Blue Prism releases require edit permissions to import objects or processes that overwrite items.

| Permission | 6.3 | 6.4 | 6.5 | 6.6 | 6.7 | 6.8 | 6.9 | 6.10 |
|---|---|---|---|---|---|---|---|---|
| Edit Business Object | Yes | No | No | Yes | No | No | No | No |
| Edit Process | Yes | No | No | Yes | No | No | No | No |

The correct behavior is displayed in versions 6.4 and 6.5 – edit permissions are not required to overwrite items on import – and this will be reflected in future releases of Blue Prism.

# Import scenarios

The examples below show the permissions required to import releases in a number of different scenarios. The examples show the import of processes but the behaviors described are the same for business objects.

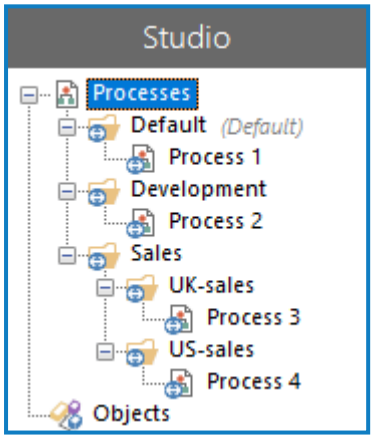
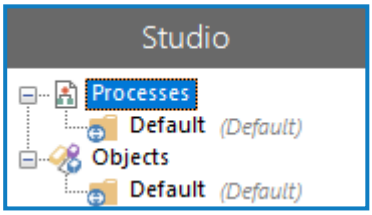
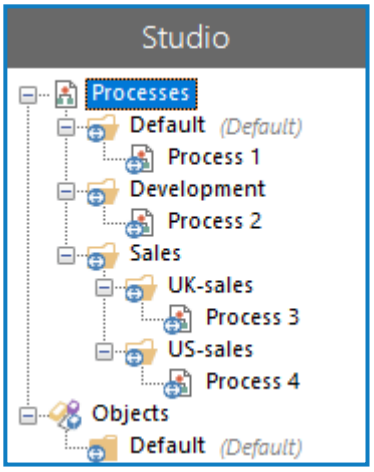
## Empty target group structure

To import a release into a structure that is not currently populated with any groups, objects or processes, the following role permissions are required:

- Release Manager > Import Release
- Process Studio > Import Process

### Result

The full group structure and associated processes are imported.

| Source | Target | Result |
|---|---|---|
|  |  |  |

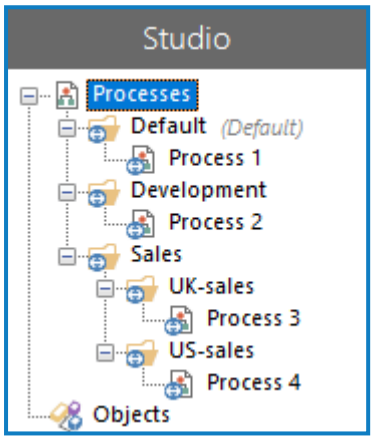
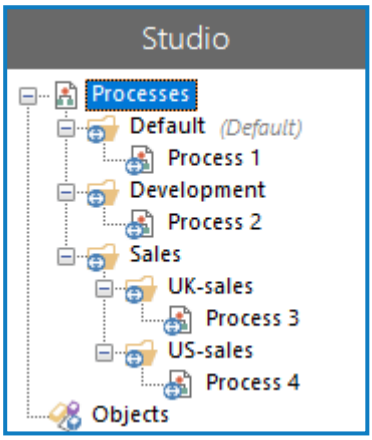
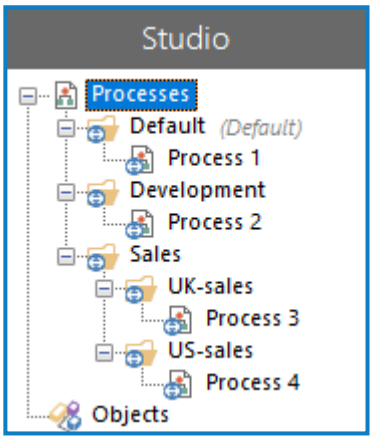
## Identical group structure

To import a release that has identical groups and items, the following role permissions are required:

- Release Manager > Import Release
- Process Studio > Import Process

### Result

The target structure is maintained following the import as the source is identical.

| Source | Target | Result |
|---|---|---|
|  |  |  |

## New subgroups created in restricted and unrestricted groups in the target structure

To import a release into a structure where the sub-groups are not present and will be created during the import, the following role permissions are required:

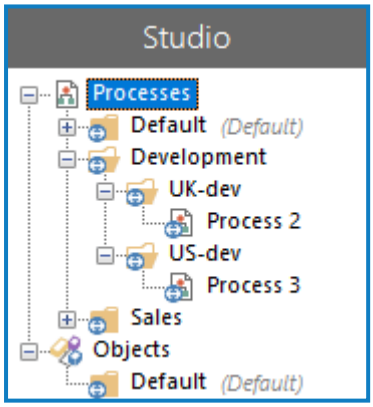
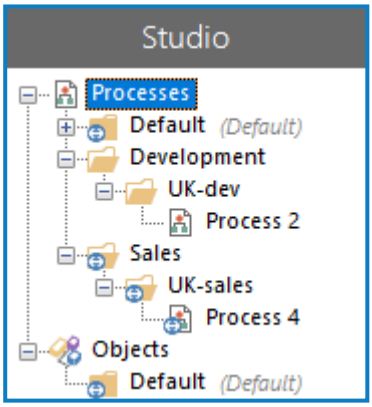
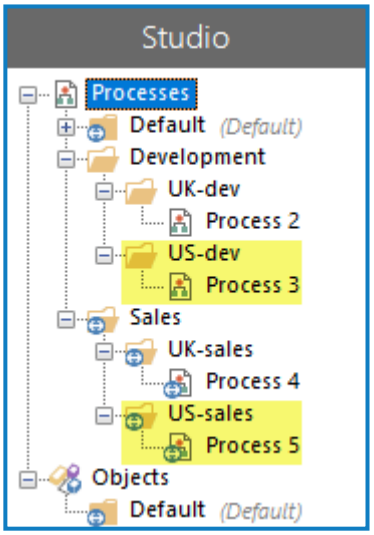
- Release Manager > Import Release
- Process Studio > Import Process

In this example:

- The US-dev group is not present in the target group – the parent Development group is restricted in the target
- The US-sales group is not present in the target group – the parent Sales group is unrestricted in the target

### Result

- The US-dev group is created during the import, inheriting permissions from the restricted Development parent group
- The US-sales group is created unrestricted during the import as the parent Sales group is unrestricted

| Source | Target | Result |
|---|---|---|
|  |  |  |

## A parent group does not exist in the target structure

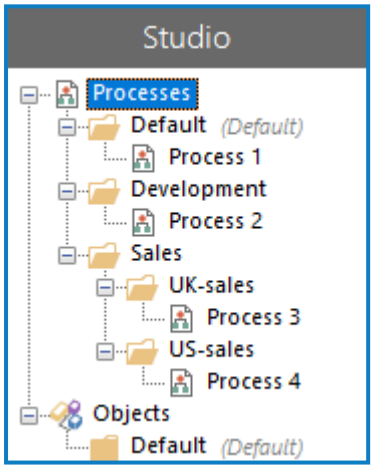
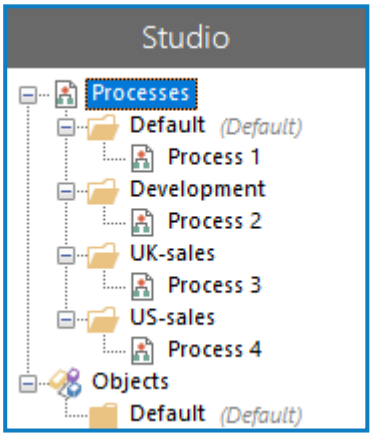
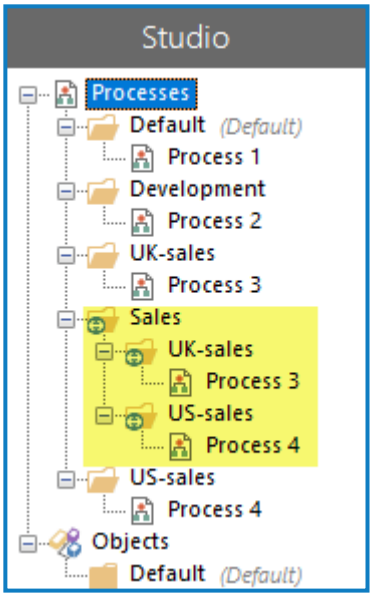
To import a release into a structure where a parent or ancestor group is not present, the following role permissions are required:

- Release Manager > Import Release
- Process Studio > Import Process

In this example, the Sales parent group is not present in the target – the UK-sales and US sales groups are at the top level in the processes structure.

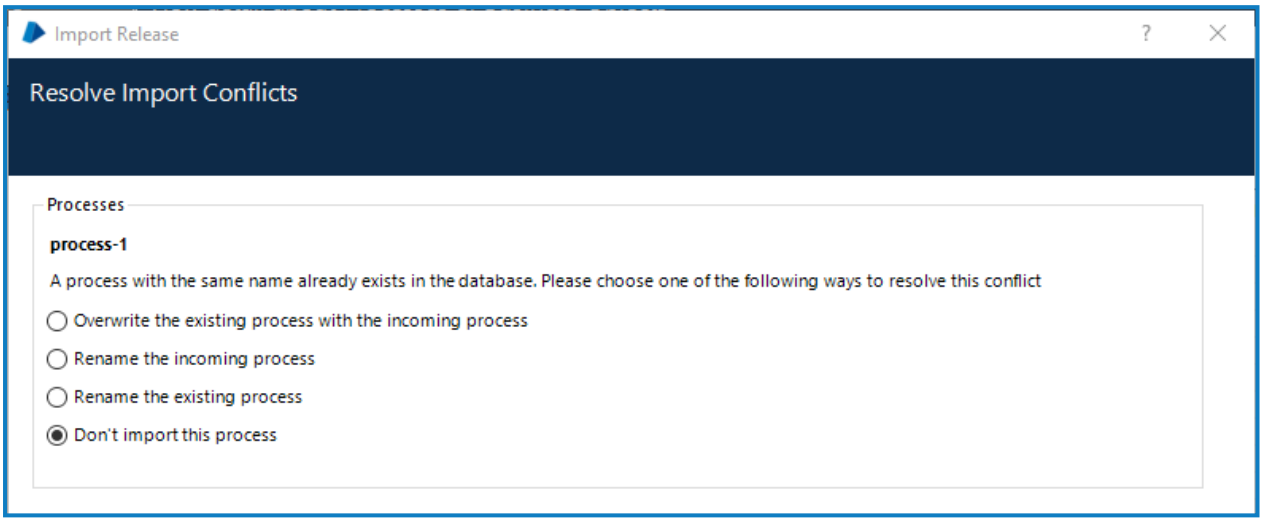
### Result

- The Sales group is created during import, containing the UK-sales and US-sales groups and their associated processes.
- The UK-sales and US-sales groups are stilpresent at the top level in the processes structure.

| Source | Target | Result |
|---|---|---|
|  |  |  |

## Conflict resolution

Objects and processes cannot be automatically imported if an item with the same name or internal ID already exists in the database. During an import conflicts are highlighted, prompting users to select an action for each duplicate item.



The following table details the options available when conflicts arise in different import scenarios.

| Incoming item location | Existing item location | Available Options |
|---|---|---|
| Root level | Default group | • Overwrite – The incoming item replaces the existing one in the default group.<br>• Rename existing – The existing item is renamed and the incoming item is moved to the default group.<br>• Rename incoming – The incoming item is renamed and imported into the default group.<br>• Do not import – No change |
| Default group | Default group | • Overwrite – The incoming item replaces the existing one in the default group.<br>• Rename existing – The existing item is renamed and the incoming item is imported into the default group.<br>• Rename incoming – The incoming item is renamed and imported into the default group.<br>• Do not import – no change. |
| Other group | Default group | • Overwrite – The incoming item replaces the existing one in the default group. A reference to the item is also added to match the location of the item in the release package. If the imported group is already present, the contents of the existing and imported groups are merged.<br>• Rename existing – The existing item is renamed and the incoming one is imported in its release package group structure. If a group of the same name is already present, the contents of the existing and incoming groups are merged.<br>• Rename incoming – The incoming item is renamed and imported in its release package group structure. If a group of the same name is already present, the contents of the existing and incoming groups are merged.<br>• Do not import – no change. |

| Incoming item location | Existing item location | Available Options |
|---|---|---|
| Other group | Other group | • Overwrite – The incoming item replaces the item in the existing group. If the incoming item is in a different location in the release package, a reference to the object is also added to that location. If an imported the group is already present, the contents of the existing and imported groups are merged.<br>• Rename existing – The existing item is renamed and the incoming one is imported in it its release package group structure. If a group of the same name is already present, the contents of the existing and incoming groups are merged.<br>• Rename incoming – The incoming item is renamed and imported in the release package group structure. If a group of the same name is already present, the contents of the existing and incoming groups are merged.<br>• Do not import – no change. |

This behavior is similar for items that have the same internal ID. However, for such items, there is no option to rename the existing item.

# Inherited permissions

When access rights are set for a group, they apply to every process, object, or resource in that group, including all child groups and their contents. This also applies when moving groups and items to another group – all items inherit the permissions of the group they are moved to, overwriting any permissions already applied. Each time a group or item is moved and the permissions are changed, a message box displays informing that the move will have an impact.

If a group is unrestricted, the Access Rights for child groups can be set as required, for restricted groups, the Access Rights for child groups can be viewed but not edited.

## Moving groups

To move groups the following permissions are required and must be enabled in the user's security role permissions so they can be applied at group level as required:

- Edit group permissions are required to move any group.
- Manage Access Rights and Edit group permissions are required to move a restricted group to a different parent or ancestor group. To move a restricted group within the same restricted ancestor, no additional permissions are required as the groups already share the same inherited permissions.

Groups can be directly and indirectly restricted:

- **Directly** – Permissions are determined by the access rights applied specifically to that group.
- **Indirectly** – Permissions are determined by the access rights applied to a parent group.

The impact of moving to and from groups with different access levels is explained in the following tables.

## Moving unrestricted groups

The group being moved does not have any permissions applied.

| Move to... | Required group permissions | | Impact on source group |
|---|---|---|---|
| | Source | Target | |
| **Root** | N/A | N/A | **Permissions are unaffected** – The source group is unrestricted and has moved to the root so group permissions do not apply. |
| **Unrestricted** | N/A | N/A | **Permissions are unaffected** – The source and target groups are unrestricted so group permissions do not apply. |
| **Restricted** | N/A | Edit Group<br>Manage Access Rights | **Increased restrictions** – The source group inherits permissions of the new restricted ancestor. |

## Moving directly restricted groups

The group being moved has permissions applied directly.

| Move to... | Required group permissions | | Impact on source group |
|---|---|---|---|
| | Source | Target | |
| **Root** | Edit Group<br>Manage Access Rights | N/A | **Permissions are unaffected** – The source group maintains its permissions. |
| **Unrestricted** | Edit Group<br>Manage Access Rights | N/A | **Decreased restrictions** – The source group is now unrestricted as it is inheriting permissions for an unrestricted ancestor. |
| **Restricted** | Edit Group<br>Manage Access Rights | Edit Group<br>Manage Access Rights | **Modified restrictions** – The source group inherits the permissions of the new ancestor group – restrictions could be increased, decreased, or unchanged, depending on the permissions of the target group. |

## Moving indirectly restricted groups

The group being moved does not have any permissions directly applied but inherits them from an ancestor group.

| Move to... | Required group permissions | | Impact on source group |
|---|---|---|---|
| | Source | Target | |
| **Root** | Edit Group<br>Manage Access Rights | N/A | **Decreased restrictions** – The source group is now unrestricted as it is no longer inheriting permissions. |
| **Unrestricted** | Edit Group<br>Manage Access Rights | N/A | **Decreased restrictions** – The source group is now unrestricted as it is inheriting permissions from an unrestricted ancestor. |
| **Restricted** | Edit Group<br>Manage Access Rights | Edit Group<br>Manage Access Rights | **Modified restrictions** – The source group inherits the permissions of the new ancestor group – restrictions could be increased, decreased, or unchanged, depending on the permissions of the target group. |

# Security role permissions

As part of the Multi-Team Environments feature, a number of changes have been made to existing security role permissions. Some permissions have been added whilst others have been updated or reassigned. Following an upgrade from versions prior to 6.3, all existing users maintain the access rights that their role currently provides.

Due to the increased granularity introduced by Multi-Team Environments, it is recommended that Security role permissions are reviewed to ensure that all users have only the permissions they require.

## New permissions

The following security role permissions were introduced in Blue Prism 6.3 to support Multi-Team Environments.

| Area | Permission | Description |
|---|---|---|
| **Object Studio** | Execute Business Object | Users can execute objects but cannot open or edit them. |
| | Manage Business Object Access Rights | Users can control access rights for object groups. |
| | Execute Business Object as Web Service | Allows the user account to be used to call associated business objects that are exposed as a web services. |
| **Process Studio** | Execute Process | Users can execute processes but cannot open or edit them. |
| | Execute Process as Web Service | Allows the user account to be used to call associated processes that are exposed as a web services. |

## Removed and updated permissions

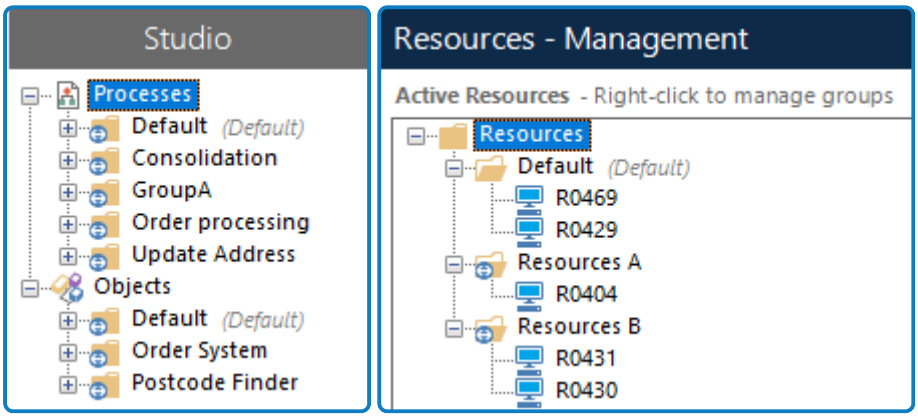
The following security role permissions were updated or removed in Blue Prism 6.3 to support Multi-Team Environments.

| Area | Permission | Details |
|---|---|---|
| **Control Room** | Full Access to Session Management | The same access rights are granted by the *Control Resource* permission. |
| | Read Access to Session Management | The same access rights are granted by the *View Resource* permission. |
| **Object Studio** | Compare Business Objects | Users can compare business objects with the *View Object Definition*permission. |
| | Create Business Objects | Renamed *Create/Clone Business Object* with added permission to clone objects. |
| | Test Business Object | The same access rights are granted by the *Execute Business Object* permission. |
| | View Business Object | Renamed *View Business Object Definition*. |
| **Process Studio** | Compare Processes | Users can compare processes with the *View Process Definition* permission. |
| | Create Process | Renamed *Create/Clone Process* with added permission to clone processes. |
| | Test Process | The same access rights are granted by the *Execute Process* permission. |
| | View Process | Renamed *View Process Definition*. |
| **Scheduler** | System – Scheduler | Users with this role, can now only modify the schedule if they also have the Edit Schedule permission. |

New permissions have also been implemented for resources. For further information, see Access rights for resources.

# Default groups

All processes, objects, and resources must be in a group and to help facilitate this, the Process Studio, Object Studio, and Resource Management hierarchies contain a Default group. Default groups are a container for items that are not in any other group. With appropriate permissions applied, this ensures processes, objects, and resources can be automatically secured upon creation.



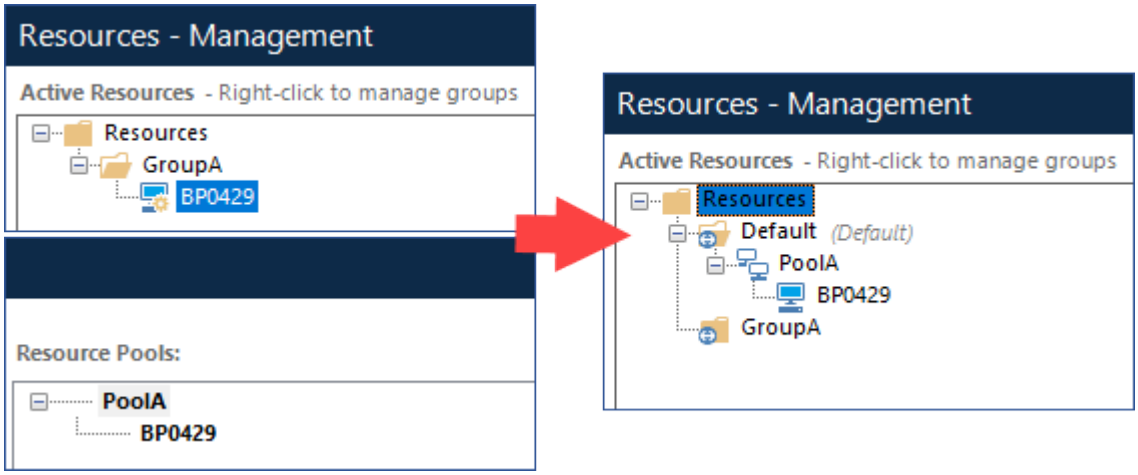

Default group behavior:

- Objects, processes, and resources cannot be stored in the root of a navigation tree – any item, not in another group appears under the default group.
- Default groups can be renamed but cannot be deleted or moved – the *(Default)* suffix is permanently displayed.
- Child groups can be created in default groups.
- All new processes and objects created at the root of a navigation tree are automatically created in the appropriate default groups.
- New objects and processes created by the Save as option in Process Studio and Object Studio are added to their default groups.
- Newly registered resources are added to the resource default group.
- On import, processes and objects at the root level are moved to the appropriate default group. If the default group is restricted, the imported item inherits the its access rights. For more information about importing and default groups, see Default group import options.

Default group upgrade behavior:

- All processes, objects, and resources that are not already in a group, are added to their respective Default groups.

- Resource pools and their associated resources are added to the default resource group. Where a resource was in a group and a pool prior to upgrade, the resource is removed from the group and added to the default group under the resource.



## Securing default groups

Default groups can be secured in the same way as any other group – access rights are applied on the group for each user role.

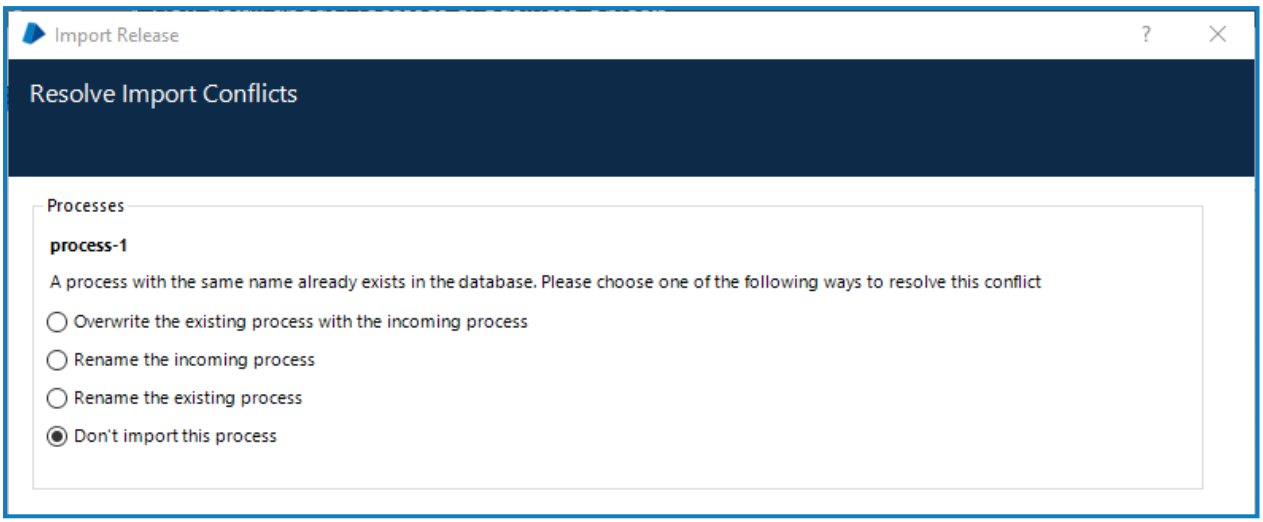
When securing default groups, consideration must be given to the impact a restriction will have on a users' capabilities within the system. Some Blue Prism actions automatically add items to default groups – if a user does not have the appropriate permissions on the default group they are prevented from taking the action.

The table below details the actions that require access rights on the default group. These actions should be considered before restricting the default group.

| Action | Impact |
|---|---|
| Create a new process or object | New items created from the File menu, context menu on the root level, or the menu button, are added to the default group.<br>Users require Create permission on the default group to carry out these actions.<br>Users are still able to create objects and processes from the context menu for groups that they have appropriate permissions on. |
| Use the Save as option in Process or Object Studio | Selecting Save as for a process or object adds a copy of the item to the default group.<br>Users require the Create Process or Create Object permission on the default group to carry out this action. |
| Remove an item from a process, object, or resource group | The *Remove from group* option adds processes, objects, and resources to the default group.<br>Users require Create and Edit Groups Permissions on the default group to carry out this action. |
| Import a release or package | Any process or object in a release or package that is at the root level is added to a default group.<br>Users require Create permissions on the default group to carry out this action. |

## Default group import options

Objects and processes cannot be automatically imported if an item with the same name or internal ID already exists in the database. During an import conflicts are highlighted, prompting users to select an action for each duplicate item.


Import Release
Resolve Import Conflicts
Processes
process-1
A process with the same name already exists in the database. Please choose one of the following ways to resolve this conflict
Overwrite the existing process with the incoming process
Rename the incoming process
Rename the existing process
Don't import this process

The following table details the options available when conflicts arise in different import scenarios.

| Incoming item location | Existing item location | Available Options |
|---|---|---|
| Root level | Default group | • Overwrite – The incoming item replaces the existing one in the default group.<br>• Rename existing – The existing item is renamed and the incoming item is moved to the default group.<br>• Rename incoming – The incoming item is renamed and imported into the default group.<br>• Do not import – No change |
| Default group | Default group | • Overwrite – The incoming item replaces the existing one in the default group.<br>• Rename existing – The existing item is renamed and the incoming item is imported into the default group.<br>• Rename incoming – The incoming item is renamed and imported into the default group.<br>• Do not import – no change. |
| Other group | Default group | • Overwrite – The incoming item replaces the existing one in the default group. A reference to the item is also added to match the location of the item in the release package. If the imported group is already present, the contents of the existing and imported groups are merged.<br>• Rename existing – The existing item is renamed and the incoming one is imported in its release package group structure. If a group of the same name is already present, the contents of the existing and incoming groups are merged.<br>• Rename incoming – The incoming item is renamed and imported in its release package group structure. If a group of the same name is already present, the contents of the existing and incoming groups are merged.<br>• Do not import – no change. |

| Incoming item location | Existing item location | Available Options |
|---|---|---|
| Other group | Other group | • Overwrite – The incoming item replaces the item in the existing group. If the incoming item is in a different location in the release package, a reference to the object is also added to that location. If an imported the group is already present, the contents of the existing and imported groups are merged.<br>• Rename existing – The existing item is renamed and the incoming one is imported in it its release package group structure. If a group of the same name is already present, the contents of the existing and incoming groups are merged.<br>• Rename incoming – The incoming item is renamed and imported in the release package group structure. If a group of the same name is already present, the contents of the existing and incoming groups are merged.<br>• Do not import – no change. |

This behavior is similar for items that have the same internal ID. However, for such items, there is no option to rename the existing item.

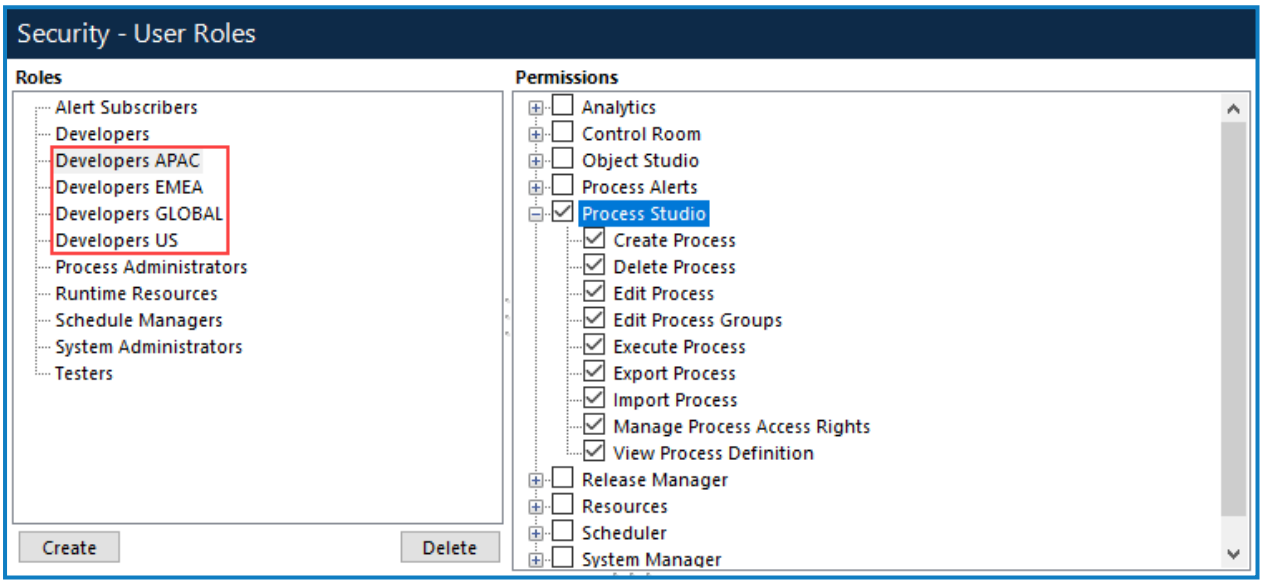
# Multi-Team Environments use cases

The following examples demonstrate how access rights for process, object, and resource groups can be used in different scenarios.

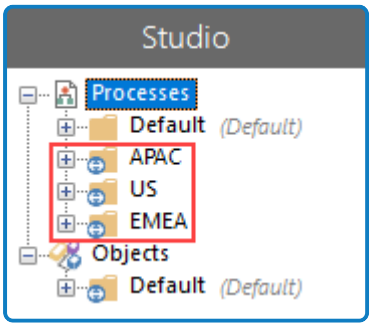
## Dedicated process access

**Restrict access to processes so they are only available to specific users**

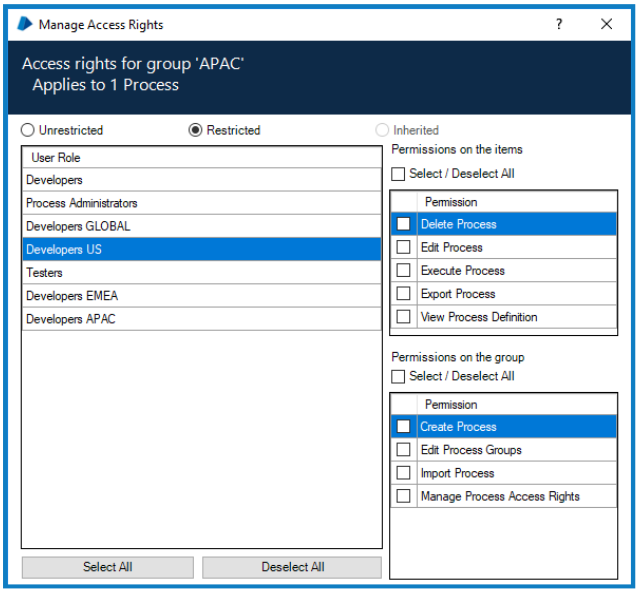
There are three process teams: Developers APAC, Developers EMEA, and Developers US, each with identical Security role permission sets. Each team is responsible for the processes for their graphical location. There is also a Developers GLOBAL team which has responsibilities in each location.



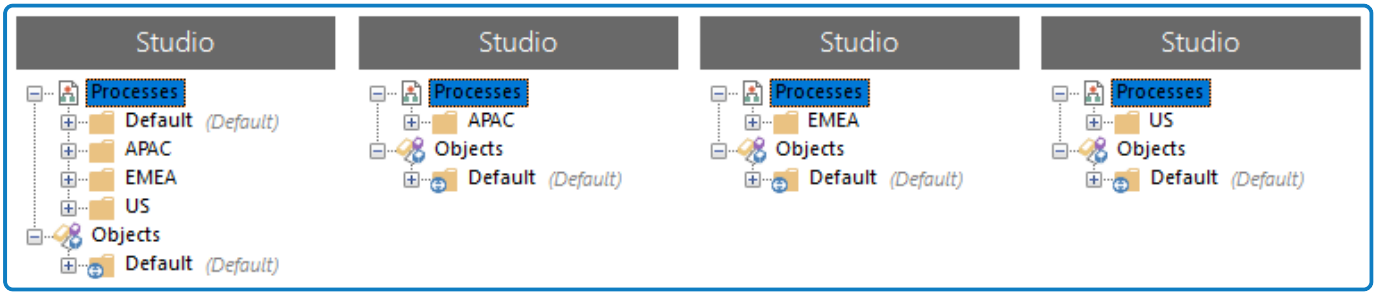

In Studio, there is a group for each geographical location containing all the processes required by the developers in that location. Currently each group has unrestricted access rights, as denoted by the icon overlay. This means that access is determined by Security role permissions and access rights have not been set at process group level.



Users only require access to the processes in the group related to their location. By updating the access rights for each group, permissions can be applied so users cannot access the processes for another location. Set the group to restricted and remove all permissions for the user roles that do not require access. The example below shows the access rights for the APAC process group with the permissions for the Developers US role removed.



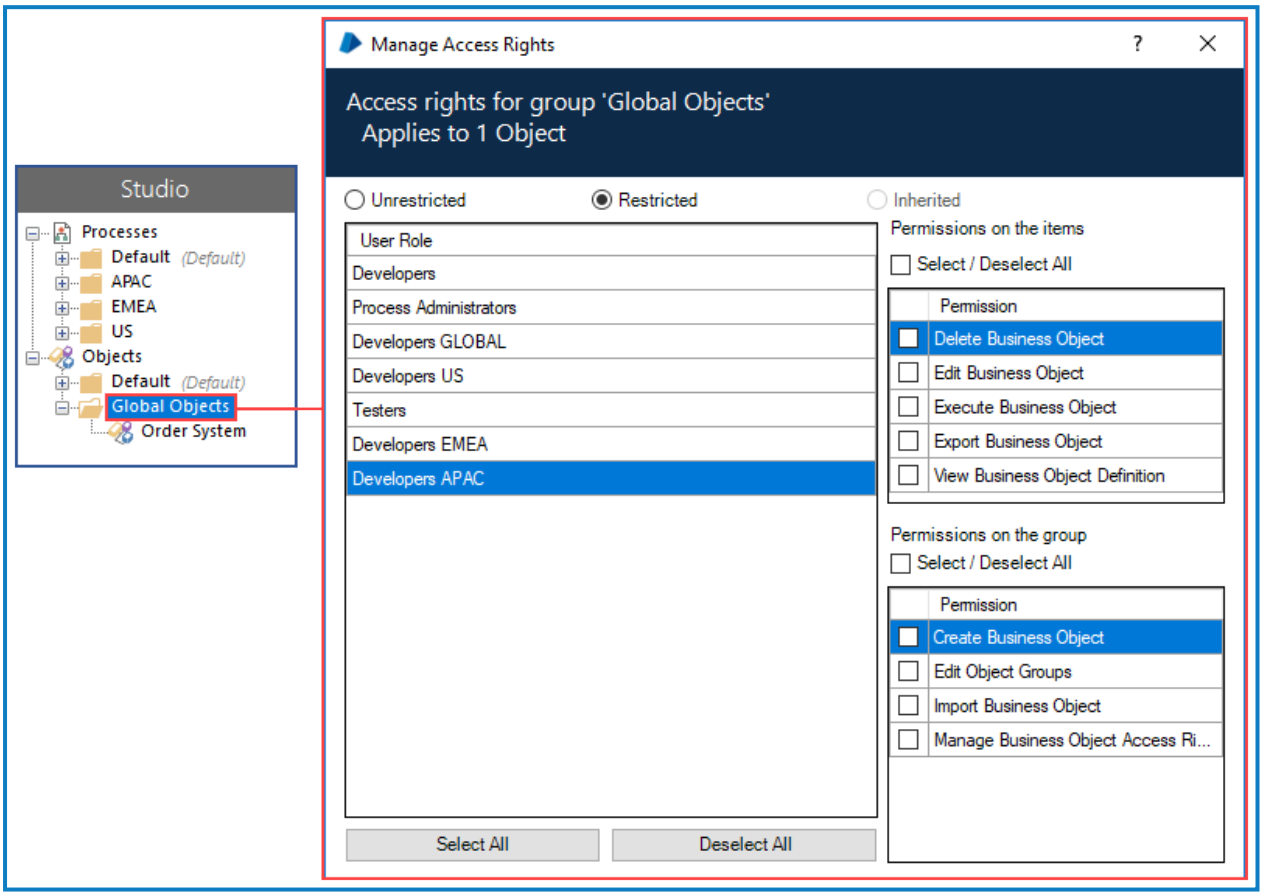

When this is repeated for each group, only members of the Developers GLOBAL team can still see all groups, however the icon overlays are no longer displayed as the groups are restricted. When a user in one of the geographic specific roles logs in, they only see the group for their location.



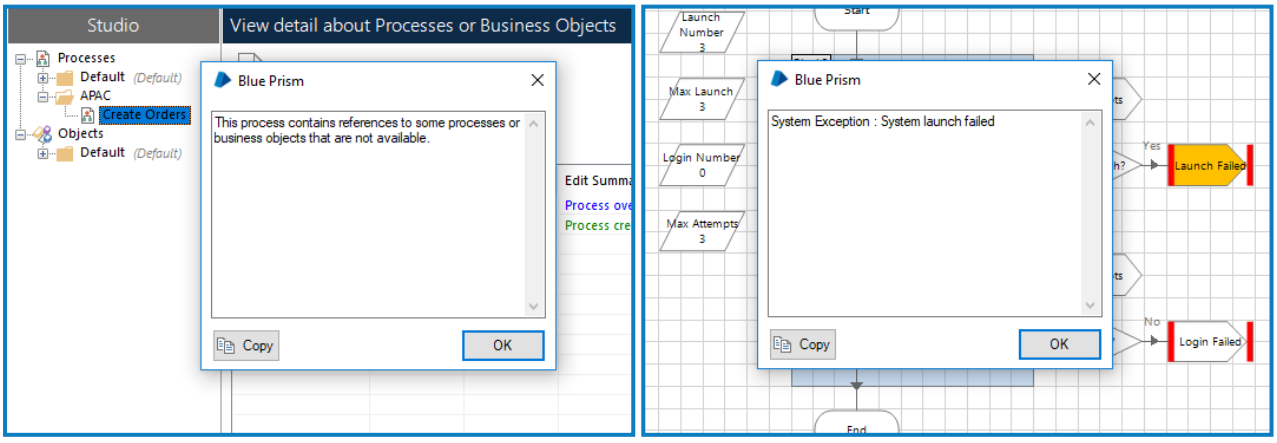
# Run a process that references a restricted object

**A user attempts to run a process that references an object for which they have no permissions.**

The Developers APAC role has no access rights on the folder containing the Order System object which automates an end-user application.




A process in the APAC group references the Order System object. When a user with the Developers APAC role opens the process, a message box displays informing them that the process contains references to items that are not available. This does not prevent them from opening and running the process, but when it attempts to launch the prohibited application, the process fails.

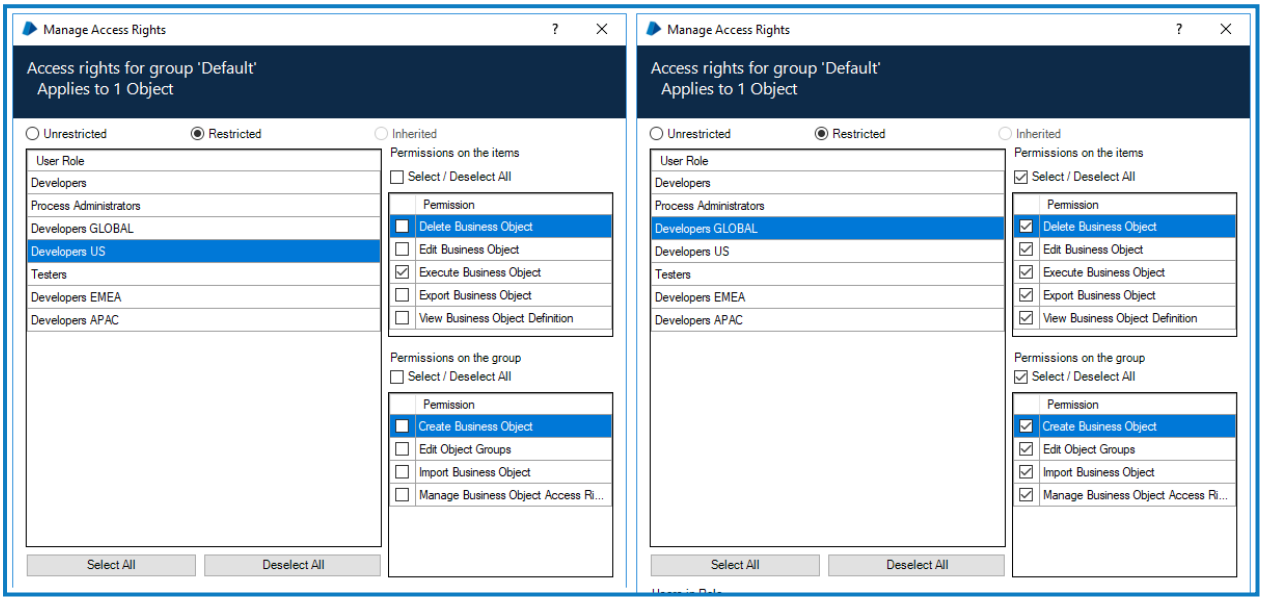


# Single business object shared by multiple teams

**Allow users to in different geographical locations to execute the same business object**

The Developers APAC, Developers EMEA, and Developers US teams all need to use the same business object in the processes they maintain. The Developers GLOBAL team requires access to update and maintain the object. All the geographical developer roles must have at least Execute Business Object and Execute Process permissions enabled for their roles. The Developers GLOBAL role requires full permissions for Process Studio and Object Studio.

Set the access rights on the folder containing the object to give Execute Business Object permissions for the three geographical teams and full access to the Developers GLOBAL team.



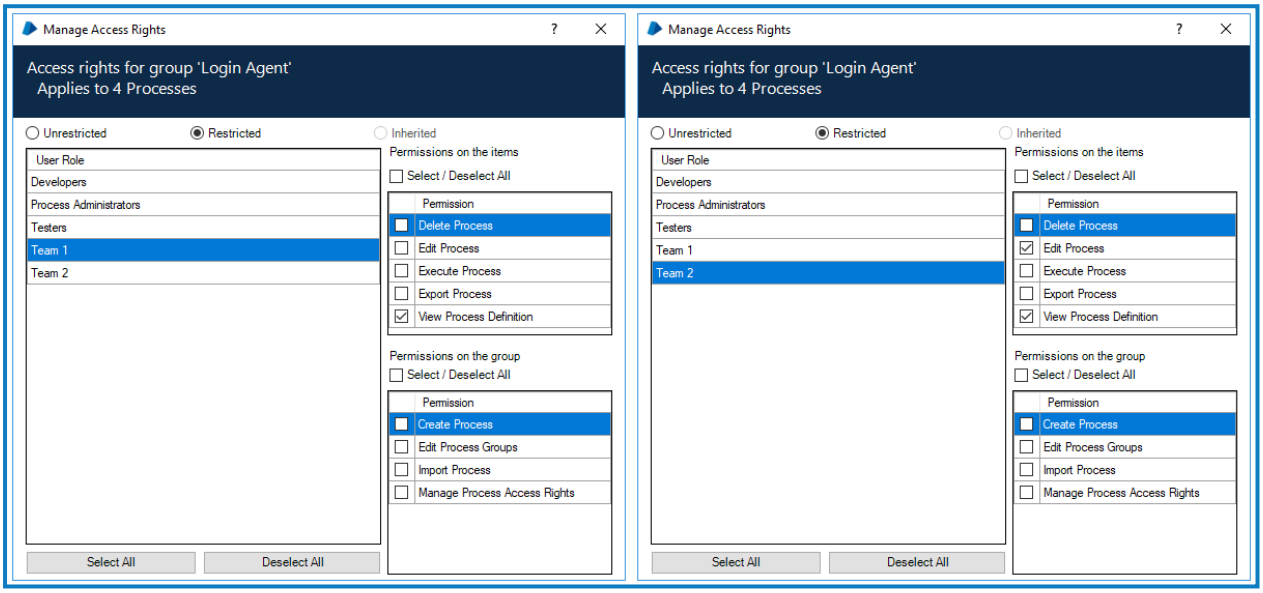

When the Developers APAC, Developers EMEA, and Developers US teams run their processes, the shared business object referenced by each team runs successfully. However, they cannot view or edit the object – only users assigned to the Developers GLOBAL role have the permissions to configure it.

## Multiple user roles

**Where there are permission conflicts due to a user being assigned multiple user roles, the most generous permission wins**

A user is in two teams:

- Team 1, which has *View Process Definition* permission on Process A
- Team 2, which has *Edit* permission on Process A




The user has full access to open and edit Process A as this is the most generous level. Colleagues who are only in Team 1 can open and view Process A but cannot edit it.

## Multiple groups

**Where there are permission conflicts due to a process being in multiple groups, the least restrictive permission wins**

The Order System object is in the restricted Default group and it is subsequently added to the unrestricted Global objects group. The Order System object inherits the permissions of the unrestricted Global objects as the permissions are the least restrictive.

Although the Default group is restricted, the object displays the unrestricted icon overlay as the permissions are inherited from the unrestricted Global objects group.