Security Service

A BPMN process can be complex and elaborated, and may also contain lane and role definitions to control which user is allowed to execute which process step. All these permissions are managed by the security service.

Each request to the service is handled in a dedicated Runtime thread. Request can be start events, submission of forms, and more. At the begin of each thread, an instance of the security service is created to check if the requester is allowed to perform this request. Next, if permission is granted, the generated code is executed.

Access the Security Service

You can access the security service from within your service implementation via the **Security** class that is part of the **Base Types.PAS_Platform** package.

> This is only possible within a **Get Data** execution. Other executions do not provide the necessary context.

The security service provides the following information:

Information	Operation	Remark
Get the current security service instance	getSecurityService()	This static operation returns an instance of the security service related to the current user. Depending on the context where this operation has been called this may be an actual person or a service user.
UUID of current user	getCurrentUserUUID()	This operation returns the UUID of the current user.
Roles of current user	getCurrentUserRoles()	This operation returns a list of all roles that have been assigned to the current user. You need a current security service object to call this operation (which you can get using getSecurityService()).
User is authorized	isAuthorized()	This operation returns if the current user or specified role is allowed to perform the provided action(s) on the provided resource.
User has a role	hasRole()	This operation returns if the current user has the specified role assigned.

Custom Security Service

> Deprecated The other operations of the **Security** class handle custom instances of the security service and should not be used by modelers.

- **SecurityService()**
  This operation implements the constructor for the security service to create a new instance of the security service.
- **setSecurityService()**
  This operation sets the instance of the security service to be the current security service to be used.
- **setCurrentUserUUID()**
  This operation sets the UUID of the current user within the given instance of the security service.
- **setCurrentUserRoles()**
  This operation sets the roles of the current user within the given instance of the security service.

Related Pages:

- getCurrentUserRoles()
- getCurrentUserUUID()
- getSecurityService()
- hasRole()
- isAuthorized()