KYPO2 User and Group - API Reference

Table of Contents

1. Overview

Kypo User and Group Reference Description.

1.1. Version information

Version : Version: 1.0.0

1.2. Contact information

Contact : Pavel Seda
Contact Email : 441048@mail.muni.cz

1.3. URI scheme

Host : localhost:8080
BasePath : /kypo2-rest-user-and-group/api/v1
Schemes : HTTP, HTTPS

1.4. Tags

- Endpoint for Groups
- Endpoint for Users
- Endpoint for roles

2. Resources

2.1. Endpoint For Groups

2.1.1. Create new group.

```
POST /kypo2-rest-user-and-group/api/v1/groups
```

Parameters

Type	Name	Description	Schema
Body	**body** *required*	Group to be created.	NewGroupDTO

Responses

HTTP Code	Description	Schema
200	Given group created.	GroupDTO

Consumes

- `application/json`

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/groups
```

Request body

```
{
  "name" : "string",
  "description" : "string",
  "users" : [ {
    "id" : 0,
    "fullName" : "string",
    "login" : "string",
    "mail" : "string"
  } ],
  "groupIdsOfImportedUsers" : [ 0 ]
}
```

Example HTTP response

Response 200

```
{
  "id" : 0,
  "name" : "string",
  "description" : "string",
  "roles" : [ {
    "id" : 0,
    "roleType" : "string",
    "nameOfMicroservice" : "string"
  } ],
  "users" : [ {
    "id" : 0,
    "fullName" : "string",
    "login" : "string",
    "mail" : "string"
  } ],
  "source" : "string",
  "canBeDeleted" : true
}
```

2.1.2. Get groups.

```
GET /kypo2-rest-user-and-group/api/v1/groups
```

Parameters

Type	Name	Description	Schema
Query	**fields** *optional*	Fields which should be returned in REST API response	string
Query	**page** *optional*	Results page you want to retrieve (0..N)	integer
Query	**size** *optional*	Number of records per page.	integer
Query	**sort** *optional*	Sorting criteria in the format: property(,asc|desc). Default sort order is ascending. Multiple sort criteria are supported.	< string > array(multi)
Body	**body** *optional*	Parameters for filtering the objects.	< string, < string > array > map

Responses

HTTP Code	Description	Schema
200	successful operation	object

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/groups
```

Request query

```
{
  "fields" : "string",
  "page" : 0,
  "size" : 0,
  "sort" : "asc"
}
```

Request body

```
{ }
```

Example HTTP response

Response 200

```
"object"
```

2.1.3. Updates input group.

```
PUT /kypo2-rest-user-and-group/api/v1/groups
```

Parameters

Type	Name	Description	Schema
Body	**body** *required*	Group to be updated.	UpdateGroupDTO

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

Consumes

- `application/json`

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/groups
```

Request body

```
{
  "id" : 0,
  "name" : "string",
  "description" : "string"
}
```

2.1.4. Tries to delete groups with given ids and returns groups and statuses of their deletion. Statuses: 1) SUCCESS - group was deleted 2) HAS_ROLE - group has at least one role 3) EXTERNAL_VALID - group is from external source and was not marked as deleted 4) MICROSERVICE_ERROR - some error occurred during deleting group in some microservice 5) ERROR_MAIN_GROUP - group cannot be deleted due to it is one of the main group for roles (ADMINISTRATOR, USER, GUEST) 6) ERROR - group could not be deleted, try it later 7) NOT_FOUND - group could not be found

```
DELETE /kypo2-rest-user-and-group/api/v1/groups
```

Parameters

Type	Name	Description	Schema
Body	**body** *required*	Ids of groups to be deleted.	< integer(int64) > array

Responses

HTTP Code	Description	Schema
200	successful operation	< GroupDeletionResponseDTO > array

Consumes

- `application/json`

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/groups
```

Request body

```
[ 0 ]
```

Example HTTP response

Response 200

```
"array"
```

2.1.5. Add users to group.

```
PUT /kypo2-rest-user-and-group/api/v1/groups/users
```

Parameters

Type	Name	Description	Schema
Body	**body** *required*	Ids of members to be added and ids of groups of imported members to group.	AddUsersToGroupDTO

Responses

HTTP Code	Description	Schema
200	successful operation	GroupDTO

Consumes

- `application/json`

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/groups/users
```

Request body

```
{
  "groupId" : 0,
  "idsOfUsersToBeAdd" : [ 0 ],
  "idsOfGroupsOfImportedUsers" : [ 0 ]
}
```

Example HTTP response

Response 200

```
{
  "id" : 0,
  "name" : "string",
  "description" : "string",
  "roles" : [ {
    "id" : 0,
    "roleType" : "string",
    "nameOfMicroservice" : "string"
  } ],
  "users" : [ {
    "id" : 0,
    "fullName" : "string",
    "login" : "string",
    "mail" : "string"
  } ],
  "source" : "string",
  "canBeDeleted" : true
}
```

2.1.6. Assign role with given role ID to group with given ID in chosen microservice

```
PUT /kypo2-rest-user-and-group/api/v1/groups/{groupId}/assign/{roleId}/in-
microservices/{microserviceId}
```

Parameters

Type	Name	Description	Schema
Path	**groupId** *required*	groupId	integer(int64)
Path	**microserviceId** *required*	microserviceId	integer(int64)
Path	**roleId** *required*	roleId	integer(int64)

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/groups/0/assign/0/in-microservices/0
```

2.1.7. Cancel role with given role ID to group with given ID in chosen microservice

```
PUT /kypo2-rest-user-and-group/api/v1/groups/{groupId}/remove/{roleId}/in-
microservices/{microserviceId}
```

Parameters

Type	Name	Description	Schema
Path	**groupId** *required*	groupId	integer(int64)
Path	**microserviceId** *required*	microserviceId	integer(int64)
Path	**roleId** *required*	roleId	integer(int64)

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/groups/0/remove/0/in-microservices/0
```

2.1.8. Get group with given id

```
GET /kypo2-rest-user-and-group/api/v1/groups/{id}
```

Parameters

Type	Name	Description	Schema
Path	**id** *required*	Id of group to be returned.	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	GroupDTO

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/groups/0
```

Example HTTP response

Response 200

```
{
  "id" : 0,
  "name" : "string",
  "description" : "string",
  "roles" : [ {
    "id" : 0,
    "roleType" : "string",
    "nameOfMicroservice" : "string"
  } ],
  "users" : [ {
    "id" : 0,
    "fullName" : "string",
    "login" : "string",
    "mail" : "string"
  } ],
  "source" : "string",
  "canBeDeleted" : true
}
```

2.1.9. Tries to delete group with given id and returns if it was successful. Statuses: 1) SUCCESS - group was deleted 2) HAS_ROLE - group has at least one role 3) EXTERNAL_VALID - group is from external source and was not marked as deleted 4) MICROSERVICE_ERROR - some error occurred during deleting group in some microservice 5) ERROR_MAIN_GROUP - group cannot be deleted due to it is one of the main group for roles (ADMINISTRATOR, USER, GUEST)

```
DELETE /kypo2-rest-user-and-group/api/v1/groups/{id}
```

Parameters

Type	Name	Description	Schema
Path	**id** *required*	Id of group to be deleted.	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	GroupDeletionResponseDTO

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/groups/0
```

Example HTTP response

Response 200

```
{
  "id" : 0,
  "microserviceForGroupDeletionDTOs" : [ {
    "id" : 0,
    "name" : "string",
    "httpStatus" : "string",
    "responseMessage" : "string"
  } ],
  "status" : "string"
}
```

2.1.10. Returns all roles of group with given id.

```
GET /kypo2-rest-user-and-group/api/v1/groups/{id}/roles
```

Parameters

Type	Name	Description	Schema
Path	**id** *required*	id	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	< RoleDTO > array

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/groups/0/roles
```

Example HTTP response

Response 200

```
"array"
```

2.1.11. Remove users from input group.

```
PUT /kypo2-rest-user-and-group/api/v1/groups/{id}/users
```

Parameters

Type	Name	Description	Schema
Path	**id** *required*	Id of group to remove users.	integer(int64)
Body	**body** *required*	Ids of members to be removed from group.	< integer(int64) > array

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

Consumes

- `application/json`

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/groups/0/users
```

Request body

```
[ 0 ]
```

2.2. Endpoint For Users

2.2.1. Gets all users.

```
GET /kypo2-rest-user-and-group/api/v1/users
```

Parameters

Type	Name	Description	Schema
Query	**fields** *optional*	Fields which should be returned in REST API response	string

Type	Name	Description	Schema
Query	**page** *optional*	Results page you want to retrieve (0..N)	integer
Query	**size** *optional*	Number of records per page.	integer
Query	**sort** *optional*	Sorting criteria in the format: property(,asc|desc). Default sort order is ascending. Multiple sort criteria are supported.	< string > array(multi)
Body	**body** *optional*	Parameters for filtering the objects.	< string, < string > array > map

Responses

HTTP Code	Description	Schema
200	successful operation	object

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/users
```

Request query

```
{
  "fields" : "string",
  "page" : 0,
  "size" : 0,
  "sort" : "asc"
}
```

Request body

```
{ }
```

Example HTTP response

Response 200

```
"object"
```

2.2.2. Tries to delete users with given ids and returns users and statuses of their deletion. Statuses: 1) SUCCESS - user was deleted 2) EXTERNAL_VALID - user is from external source and was not marked as deleted 3) ERROR - user could not be deleted, try it later 4) NOT_FOUND - user could not be found

```
DELETE /kypo2-rest-user-and-group/api/v1/users
```

Parameters

Type	Name	Description	Schema
Body	**body** *required*	Ids of users to be deleted.	< integer(int64) > array

Responses

HTTP Code	Description	Schema
200	successful operation	< UserDeletionResponseDTO > array

Consumes

- `application/json`

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/users
```

Request body

```
[ 0 ]
```

Example HTTP response

Response 200

```
"array"
```

2.2.3. Returns details of user who is logged in

```
GET /kypo2-rest-user-and-group/api/v1/users/info
```

Responses

HTTP Code	Description	Schema
200	successful operation	UserInfoDTO

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/users/info
```

Example HTTP response

Response 200

```
{
  "id" : 0,
  "fullName" : "string",
  "login" : "string",
  "mail" : "string",
  "roles" : [ {
    "id" : 0,
    "roleType" : "string",
    "nameOfMicroservice" : "string"
  } ]
}
```

2.2.4. Gets all users except users in given group.

```
GET /kypo2-rest-user-and-group/api/v1/users/not-in-groups/{groupId}
```

Parameters

Type	Name	Description	Schema
Path	**groupId** *required*	Id of group whose users do not get.	integer(int64)
Query	**fields** *optional*	Fields which should be returned in REST API response	string
Query	**page** *optional*	Results page you want to retrieve (0..N)	integer
Query	**size** *optional*	Number of records per page.	integer
Query	**sort** *optional*	Sorting criteria in the format: property(,asc|desc). Default sort order is ascending. Multiple sort criteria are supported.	< string > array(multi)

Responses

HTTP Code	Description	Schema
200	successful operation	object

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/users/not-in-groups/0
```

Request query

```
{
  "fields" : "string",
  "page" : 0,
  "size" : 0,
  "sort" : "asc"
}
```

Example HTTP response

Response 200

```
"object"
```

2.2.5. Gets user with given id.

```
GET /kypo2-rest-user-and-group/api/v1/users/{id}
```

Parameters

Type	Name	Description	Schema
Path	**id** *required*	Id of user to be returned.	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	UserDTO

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/users/0
```

Example HTTP response

Response 200

```
{
  "id" : 0,
  "fullName" : "string",
  "login" : "string",
  "mail" : "string",
  "roles" : [ {
    "id" : 0,
    "roleType" : "string",
    "nameOfMicroservice" : "string"
  } ]
}
```

2.2.6. Tries to delete user with given screen name and returns if it was successful. Statuses: 1) SUCCESS - user was deleted 2) EXTERNAL_VALID - user is from external source and was not marked as deleted

```
DELETE /kypo2-rest-user-and-group/api/v1/users/{id}
```

Parameters

Type	Name	Description	Schema
Path	**id** *required*	Screen name of user to be deleted.	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	UserDeletionResponseDTO

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/users/0
```

Example HTTP response

Response 200

```
{
  "user" : {
    "id" : 0,
    "fullName" : "string",
    "login" : "string",
    "mail" : "string",
    "roles" : [ {
      "id" : 0,
      "roleType" : "string",
      "nameOfMicroservice" : "string"
    } ]
  },
  "status" : "string"
}
```

2.2.7. Returns all roles of user with given id.

```
GET /kypo2-rest-user-and-group/api/v1/users/{id}/roles
```

Parameters

Type	Name	Description	Schema
Path	**id** *required*	id	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	< RoleDTO > array

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/users/0/roles
```

Example HTTP response

Response 200

```
"array"
```

2.3. Endpoint For Roles

2.3.1. Get all roles

```
GET /kypo2-rest-user-and-group/api/v1/roles
```

Parameters

Type	Name	Description	Schema
Query	**page** *optional*	Results page you want to retrieve (0..N)	integer
Query	**size** *optional*	Number of records per page.	integer
Query	**sort** *optional*	Sorting criteria in the format: property(,asc|desc). Default sort order is ascending. Multiple sort criteria are supported.	< string > array(multi)

Responses

HTTP Code	Description	Schema
200	successful operation	object

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/roles
```

Request query

```
{
  "page" : 0,
  "size" : 0,
  "sort" : "asc"
}
```

Example HTTP response

Response 200

```
"object"
```

2.3.2. Get role with given id

```
GET /kypo2-rest-user-and-group/api/v1/roles/{id}
```

Parameters

Type	Name	Description	Schema
Path	**id** *required*	Id of role to be returned	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	object

Produces

- `application/json`

Example HTTP request

Request path

```
/kypo2-rest-user-and-group/api/v1/roles/0
```

Example HTTP response

Response 200

```
"object"
```

3. Definitions

3.1. AddUsersToGroupDTO

Name	Description	Schema
groupId *optional*	**Example** : `0`	integer(int64)
idsOfGroupsOfImportedUsers *optional*	**Example** : `[ 0 ]`	< integer(int64) > array
idsOfUsersToBeAdd *optional*	**Example** : `[ 0 ]`	< integer(int64) > array

3.2. GroupDTO

Name	Description	Schema
canBeDeleted *optional*	**Example** : `true`	boolean
description *optional*	**Example** : `"string"`	string
id *optional*	**Example** : `0`	integer(int64)
name *optional*	**Example** : `"string"`	string
roles *optional*	**Example** : `[ "RoleDTO" ]`	< RoleDTO > array
source *optional*	**Example** : `"string"`	enum (INTERNAL, PERUN)
users *optional*	**Example** : `[ "UserForGroupsDTO" ]`	< UserForGroupsDTO > array

3.3. GroupDeletionResponseDTO

Name	Description	Schema
id *optional*	**Example** : `0`	integer(int64)
microserviceForGroupDeletionDTOs *optional*	**Example** : `[ "MicroserviceForGroupDeletionDTO" ]`	< MicroserviceForGroupDeletionDTO > array

Name	Description	Schema
status *optional*	**Example** : `"string"`	enum (EXTERNAL_VALID, SUCCESS, ERROR, NOT_FOUND, MICROSERVICE_ERROR, ERROR_MAIN_GROUP)

3.4. MicroserviceForGroupDeletionDTO

Name	Description	Schema
httpStatus *optional*	**Example** : `"string"`	enum (CONTINUE, SWITCHING_PROTOCOLS, PROCESSING, CHECKPOINT, OK, CREATED, ACCEPTED, NON_AUTHORITATIVE_INFORMATION, NO_CONTENT, RESET_CONTENT, PARTIAL_CONTENT, MULTI_STATUS, ALREADY_REPORTED, IM_USED, MULTIPLE_CHOICES, MOVED_PERMANENTLY, FOUND, MOVED_TEMPORARILY, SEE_OTHER, NOT_MODIFIED, USE_PROXY, TEMPORARY_REDIRECT, PERMANENT_REDIRECT, BAD_REQUEST, UNAUTHORIZED, PAYMENT_REQUIRED, FORBIDDEN, NOT_FOUND, METHOD_NOT_ALLOWED, NOT_ACCEPTABLE, PROXY_AUTHENTICATION_REQUIRED, REQUEST_TIMEOUT, CONFLICT, GONE, LENGTH_REQUIRED, PRECONDITION_FAILED, PAYLOAD_TOO_LARGE, REQUEST_ENTITY_TOO_LARGE, URI_TOO_LONG, REQUEST_URI_TOO_LONG, UNSUPPORTED_MEDIA_TYPE, REQUESTED_RANGE_NOT_SATISFIABLE, EXPECTATION_FAILED, I_AM_A_TEAPOT, INSUFFICIENT_SPAC

Name	Description	Schema
id *optional*	**Example** : `0`	integer(int64)
name *optional*	**Example** : `"string"`	string
responseMessage *optional*	**Example** : `"string"`	string

3.5. NewGroupDTO

Name	Description	Schema
description *optional*	**Example** : `"string"`	string
groupIdsOfImportedUsers *optional*	**Example** : `[ 0 ]`	< integer(int64) > array
name *optional*	**Example** : `"string"`	string
users *optional*	**Example** : `[ "UserForGroupsDTO" ]`	< UserForGroupsDTO > array

3.6. RoleDTO

Name	Description	Schema
id *optional*	**Example** : `0`	integer(int64)
nameOfMicroservice *optional*	**Example** : `"string"`	string
roleType *optional*	**Example** : `"string"`	string

3.7. UpdateGroupDTO

Name	Description	Schema
description *optional*	**Example** : `"string"`	string
id *optional*	**Example** : `0`	integer(int64)

Name	Description	Schema
name *optional*	**Example** : `"string"`	string

3.8. UserDTO

Name	Description	Schema
fullName *optional*	**Example** : `"string"`	string
id *optional*	**Example** : `0`	integer(int64)
login *optional*	**Example** : `"string"`	string
mail *optional*	**Example** : `"string"`	string
roles *optional*	**Example** : `[ "RoleDTO" ]`	< RoleDTO > array

3.9. UserDeletionResponseDTO

Name	Description	Schema
status *optional*	**Example** : `"string"`	enum (SUCCESS, EXTERNAL_VALID, ERROR, NOT_FOUND)
user *optional*	**Example** : `"UserDTO"`	UserDTO

3.10. UserForGroupsDTO

Name	Description	Schema
fullName *optional*	**Example** : `"string"`	string
id *optional*	**Example** : `0`	integer(int64)
login *optional*	**Example** : `"string"`	string
mail *optional*	**Example** : `"string"`	string

3.11. UserInfoDTO

Name	Description	Schema
fullName *optional*	**Example** : `"string"`	string
id *optional*	**Example** : `0`	integer(int64)
login *optional*	**Example** : `"string"`	string
mail *optional*	**Example** : `"string"`	string
roles *optional*	**Example** : `[ "RoleDTO" ]`	< RoleDTO > array