Applications management

Apps

Applications store the REST URL of the web server they interact. Also, apps store other settings and data.

Field	Description
`name`	Application name
`appKey`	Unique identifier that is used on the client side to connect to this app
`url`	REST URL for web server integration via WCS RESTful API.
`owner`	User who owns the application
`mainClass`	Main class for direct calls. For deep integration and server-side developers.
`callbackClass`	Main class for callbacks. For deep integration and server-side developers.
`restUsername`	User name for HTTP authentication of REST.
`restPassword`	User password for HTTP authentication of REST.

Application descriptions are stored in database.yml file.

You can manage apps using the following commands:

- `show apps`
- `add app`
- `update app`
- `remove app`

show apps

The command is used to show existing apps in the system. A user gets the list of his own applications only. Parameters:

- `-d | --detail` - enable detailed output

Example:

```
show apps
show apps -d
```

add app

Used to add a new app to the system. By default a new app belongs to the user who created it. Parameters:

- `app name` - the name of the app, required
- `app key` - the unique key of the app, required
- `app url` - REST URL for integration with the Web server via WCS RESTful API, required
- `-m | --m-class` - main class for direct calls, optional
- `-c | --c-class` - main class for callbacks, optional
- `-u | --u-rest` - user name for HTTP authentication, optional
- `-p | --p-rest` - user password for HTTP authentication, optional
- `-o | --owner` - the user who should own the app, optional

Example:

```
add app MyApp myNewUniqueKey "http://localhost/my_app"
add app -u rest_http_user -p rest_password MyApp2 myNewKey2
"http://localhost/my_app2"
add app -o alice MyApp3 myNewKey3 "http://localhost/my_app3"
```

update app

The command is used to modify an existing app in the system. A user can modify only applications he or she owns. Parameters:

- `app name` - the name of the app, required
- `-k | --key` - the unique key of the app
- `-l | --url` - REST URL for integration with the Web server via WCS RESTful API
- `-m | --m-class` - main class for direct calls, optional
- `-c | --c-class` - main class for callbacks, optional
- `-u | --u-rest` - user name for HTTP authentication, optional
- `-p | --p-rest` - user password for HTTP authentication, optional
- `-o | --owner` - the user who should own the app, optional

Example:

```
update app -k newAppKey -u new_rest_user -p new_rest_password MyApp
update app -o admin MyApp
update app -l "http://10.10.10.10/app" MyApp
```

remove app

Used to remove an app. Only the user who own the application can remove it. Parameters:

- `app_name` - the name of the app, required

Example:

```
remove app MyApp
```

Application REST methods

You can manage application REST methods with the following commands:

- `add app-rest-method`
- `show app-rest-methods`
- `remove app-rest-method`

show app-rest-methods

The command is used to show all the REST methods of the application.

Parameters:

- `app key` - the unique key of the app, required

Example:

```
show app-rest-methods defaultApp
```

add app-rest-method

The command is used to add REST method to the application.

Parameters:

- `app key` - the unique key of the app, required
- `rest method` - REST method to be added, required

- `-a | --all` - add all the REST methods, optional, in this case `rest method` is not required

Example:

```
add app-rest-method MyAppKey connect
add app-rest-method -a MyAppKey
```

remove app-rest-method

The command is used to remove REST method from the application.

Parameters: - `app key` - the unique key of the app, required - `rest method` - REST method to be removed, required
- `-a | --all` - remove all the REST methods, optional, in this case `rest method` is not required

Example:

```
remove app-rest-method MyAppKey connect
remove app-rest-method -a MyAppKey
```

Examples

Adding the app for the existing user bob

```
add app -o bob BobsApp bobAppKey "http://bobs_web_server.example.com/app"
```

Modify the app and add data for HTTP authentication, change HTTP address

```
update app -u bobs_app -p paSsw0rd BobsApp
update app -l "https://bobs_new_server.example.com/secureApp" BobsApp
```

Transfer the ownership of the app to another user

```
update app -o alice BobsApp
```