

Oracle Enterprise Repository

User API

Overview

The User Subsystem provides a Web Services-based mechanism that can be used to create, read, update, query, and otherwise manipulate Oracle Enterprise Repository User accounts.

Related subsystems

- **Role Subsystem**

Additional Import(s) Required

```
import com.flashline.registry.openapi.entity.RegistryUser;  
import com.flashline.registry.openapi.query.UserCriteria;
```

Use Cases

Use Case: Manipulating Users

Description

- Create a new user.
- Retrieve an existing user.
- Update a user.
- Deactivate a user.
- Query for users.

Sample Code:

```
package com.flashline.sample.userapi;

import java.net.MalformedURLException;
import java.net.URL;
import java.rmi.RemoteException;
import java.util.Calendar;

import javax.xml.rpc.ServiceException;

import com.flashline.registry.openapi.base.OpenAPIException;
import com.flashline.registry.openapi.entity.AuthToken;
import com.flashline.registry.openapi.entity.RegistryUser;
import com.flashline.registry.openapi.query.UserCriteria;
import com.flashline.registry.openapi.service.v300.FlashlineRegistry;
import com.flashline.registry.openapi.service.v300.FlashlineRegistryServiceLocator;

public class Users {
    public static void main(String pArgs[] throws OpenAPIException, RemoteException,
        ServiceException {
        try {

            ////////////////////////////////////////////////////////////////////
            // Connect to Oracle Enterprise Repository
            ////////////////////////////////////////////////////////////////////
            URL IURL = null;
            IURL = new URL(pArgs[0]);
            FlashlineRegistry repository = new FlashlineRegistryServiceLocator()
                .getFlashlineRegistry(IURL);

            ////////////////////////////////////////////////////////////////////
            // Authenticate with OER
            ////////////////////////////////////////////////////////////////////
            AuthToken authToken = repository.authTokenCreate(pArgs[1],
                pArgs[2]);

            // -----
            // Create a new user
            String IUserName = "testUserCreate_"+Calendar.getInstance().getTimeInMillis();
            String IFirstName = "testUserCreate_FirstName";
            String ILastName = "testUserCreate_LastName";
            String IEmail = IUserName+"@example.com";
            String IPassword = "testUserCreate_Password";
            boolean IMustChangePassword = false;
            boolean IPasswordNeverExpires = false;
            boolean IAssignDefaultRoles = true;

            RegistryUser RbacRegistrySecUser = repository.userCreate(
```

```
authToken, IUserName, IFirstName, ILastName, IEmail, IPassword,  
IMustChangePassword, IPasswordNeverExpires, IAssignDefaultRoles);
```

```
// -----
```

```
// Read a User
```

```
long IId = 50000; // user id must exist in OER  
RegistryUser IUser1 = repository.userRead(authToken,  
    IId);
```

```
// -----
```

```
// Update a User
```

```
IUser1.setActiveStatus(10);  
IUser1.setUserName("xxx");  
IUser1.setPhoneNumber("412-521-4914");  
IUser1.setMustChangePassword(true);  
IUser1.setPasswordNeverExpires(false);  
IUser1.setPassword("changed_password");  
IUser1.setEmail("newaddress@bea.com");  
try {  
    IUser1 = repository.userUpdate(authToken,  
        IUser1);
```

```
} catch (OpenAPIException e) {  
    e.printStackTrace();  
}
```

```
// -----
```

```
// Deactivate a User
```

```
RegistryUser IUser2 = null;  
try {  
    IUser2 = repository.userDeactivate(authToken, IId);  
} catch (OpenAPIException e) {  
    e.printStackTrace();  
}
```

```
// -----
```

```
// Query for Users
```

```
RegistryUser IUsers[] = null;  
UserCriteria IUserCriteria = null;  
IUserCriteria = new UserCriteria();  
IUserCriteria.setNameCriteria("testname");  
IUsers = repository.userQuery(authToken,  
    IUserCriteria);
```

```
} catch (OpenAPIException IEx) {  
    System.out.println("ServerCode = " + IEx.getServerErrorCode());  
    System.out.println("Message = " + IEx.getMessage());  
    System.out.println("StackTrace:");  
    IEx.printStackTrace();  
} catch (RemoteException IEx) {
```

```
    lEx.printStackTrace();  
} catch (ServiceException lEx) {  
    lEx.printStackTrace();  
} catch (MalformedURLException lEx) {  
    lEx.printStackTrace();  
}  
}  
}
```