Sample scripts

> Note that Soffid supports different scripting languages, you can configure it in the Smart engine settings screen.

> Additionally, in the initial configuration of the container, we can configure the SOFFID_TRUSTED_SCRIPTS environment variable to allow the use of insecure classes. You can find this information visiting the Installing IAM Console page.

Table of contents

1. Agent scripts
   - User full name
   - Create mainDomain if it doesn't exit
   - Recover active agents
   - Show by a user the agents that have associates
2. Identity scripts
   - Recover a user for userName
   - Recover a users from a Jquery
   - Print some attributes
   - Print by user the email
   - Print by user some additional data
   - Create a new identity
   - Update an identity
   - Delete an identity
3. Account scripts
   - Recover accounts of user
   - Remove attribute values of a metadata
4. Role scripts

- Recover roles of a user
- Print the associated roles for each account
- Print for an account the roles and applications for each of them
- Print the roles associated with each account
- Create a new role
- Update a role
- Delete a role
- List the roles of an application

5. Mail scripts
   - Send email

---

1. Agent scripts

User full name

```
return firstName + lastName;
```

Create mainDomain if it doesn't exit

```
String mailDomain = null;
if (email != void && email != null && email.contains("@")) {
    String[] mailTokens = email.split("@");
    mailDomain = mailTokens[1];
}
com.soffid.iam.service.MailListsService service = com.soffid.iam.ServiceLocator.instance().getMailListsService();
com.soffid.iam.api.MailDomain domain = service.findMailDomainByName(mailDomain);
if (domain==null) {
    domain = new com.soffid.iam.api.MailDomain();
    domain.setCode(mailDomain);
    domain.setDescription(mailDomain);
    domain.setObsolete(new Boolean(false));
    domain = service.create(domain);
}
return mailDomain;
```

Recover active agents

```
llistaAgents =  serviceLocator.getDispatcherService().findAllActiveDispatchers();
for(agent:llistaAgents) {
  out.println("Nom: " + agent.name);
  out.println("Class Name: " + agent.className + "\n");
}
```

Show by a user the agents that have associates

```
llistaUsuaris = serviceLocator.getUserService().findUserByJsonQuery("userName eq \"Ivan\" ");
for(usuari:llistaUsuaris) {
  out.println("Usuario: " + usuari.userName);

   llisstacuentas = serviceLocator.getAccountService().findAccountByJsonQuery("users.user.userName eq
\""+usuari.userName+"\" ");

  for(cuenta:llisstacuentas){
    out.print("   Cuenta : " + cuenta.name);
    out.println("   ID: " + cuenta.id);
    llistaRole = serviceLocator.getApplicationService().findRoleAccountByAccount(cuenta.id);

    for(role:llistaRole){
      out.print("      Role: " + role.roleName + "\n");
      out.println("          Aplicacion: " + role.informationSystemName);
      out.println("            Agente: " + role.system);
    }
  }
}
```

---

2. Identity scripts

Recover a user for userName

```
u = serviceLocator.getUserService().findUserByUserName("Ivan");
out.print("Usuari: " + u.firstName);
```

Recover a users from a Jquery

```
llistaUsuari = serviceLocator.getUserService().findUserByJsonQuery("firstName sw \"A\" AND lastName sw \"V\"
");
for (usuari:llistaUsuari){
    out.println("Usuari: " + usuari.userName);
}
```

Print some attributes

```
u = serviceLocator.getUserService().findUserByUserName("02");
out.println("UserName: " + u.userName);
out.println("Name: " + u.firstName);
out.println("LastName: " + u.lastName);
```

Print by user the email

```
u = serviceLocator.getUserService().findUserByUserName("02");
out.print("Email: " + u.shortName + "@" + u.mailDomain);
```

Print by user some additional data

```
llistaDadesUsuari = serviceLocator.getUserService().findUserDataByUserName("18008366X");
for(dadaUsuari:llistaDadesUsuari){
  out.println("Atributs " + dadaUsuari.attribute + " = " + dadaUsuari.value);
}
```

Create a new identity

```
 try {
 newUser = new com.soffid.iam.api.User();
//Instanciar un nuevo objeto de tipo usuario

 newUser.userName = "IvanVis"; //Faltan 6 parametres
 newUser.firstName = "Ivannn";
 newUser.lastName = "Visarttt";
 newUser.userType = "I";
 newUser.profileServer = "null" ;
 newUser.homeServer = "null" ;
 newUser.mailServer = "null" ;
```

```
 newUser.primaryGroup = "world";
 newUser.active = true;

 serviceLocator.getUserService().create(newUser);
}catch(Exception e){
 e.printStackTrace(out);
}
```

Update an identity

```
u = serviceLocator.getUserService().findUserByUserName("Ivan");
u.firstName = "Ivaaan1";
u = serviceLocator.getUserService().update(u);
out.print(u.firstName);
out.print(u.userName);
```

Delete an identity

```
try {
  u = serviceLocator.getUserService().findUserByUserName("02");
  serviceLocator.getUserService().delete(u);
} catch(Exception e) {
    e.printStackTrace(out);
}
```

---

3. Account scripts

Recover accounts of user

```
la = serviceLocator.getAccountService().findAccountByJsonQuery("users.user.userName eq \"02\" ");
for(a:la) {
  out.println("Cuenta: " + a.name);
  out.println("ID: " + a.id);
  out.println("System: " + a.system + "\n");
}
```

Remove attribute values of a metadata

```
public void removeUnAttributeValues(String attribute, String system) {
  la = serviceLocator.getAccountService().findAccountByJsonQuery("system eq \""+system+"\"");
  for (a : la) {
    laa = serviceLocator.getAccountService().getAccountAttributes(a);
    for (aa : laa) {
      if (aa.attribute.equals(attribute)) {
        if (aa.value!=null) {
          out.print("accountName: "+accountName+", attribute.value: "+aa.value);
          serviceLocator.getAccountService().removeAccountAttribute(aa);
          out.println(" ---> removed");
        }
      }
    }
  }
}
removeUnAttributeValues("manager","OSCM");
```

4. Role scripts

Recover roles of a user

```
user = serviceLocator.getUserService().findUserByUserName("Ivan");
out.println("Usuari: " + user.userName + "\n");
rolsUser = serviceLocator.getUserService().findUserRolesHierachyByUserName(user.userName);
for(listrRolsUser:rolsUser){
  out.println("Nombre: " + listrRolsUser.name);
  out.println("Descripcion: " + listrRolsUser.description);
  out.println();
}
```

Print the associated roles for each account

```
llistaUsuaris = serviceLocator.getUserService().findUserByJsonQuery("userName eq \"Ivan\" ");
for(usuari:llistaUsuaris){

   llisstacuentas = serviceLocator.getAccountService().findAccountByJsonQuery("users.user.userName eq
\""+usuari.userName+"\" ");
```

```
  for(cuenta:llisstacuentas){
    out.print("   Cuenta : " + cuenta.name);
    llistaRole = serviceLocator.getApplicationService().findRoleAccountByAccount(cuenta.id);

    for(role:llistaRole){
      out.print("      Role: " + role.roleName + "\n");
    }
  }
}
```

Print for an account the roles and applications for each of them

```
llistaUsuaris = serviceLocator.getUserService().findUserByJsonQuery("userName eq \"Ivan\" ");
for(usuari:llistaUsuaris){

   llisstacuentas = serviceLocator.getAccountService().findAccountByJsonQuery("users.user.userName eq
\""+usuari.userName+"\" ");

  for(cuenta:llisstacuentas){
    out.print("   Cuenta : " + cuenta.name);
    out.println("   ID: " + cuenta.id);
    llistaRole = serviceLocator.getApplicationService().findRoleAccountByAccount(cuenta.id);

    for(role:llistaRole){
      out.print("      Role: " + role.roleName + "\n");
      out.println("          Aplicacion: " + role.informationSystemName);
    }
  }
}
```

Print the roles associated with each account

```
usuCuenta = serviceLocator.getUserService().findUserByJsonQuery("");
for(listaUsuCuenta:usuCuenta) {

    out.println("Usuario: " + listaUsuCuenta.userName);
    out.println("Nombre: " + listaUsuCuenta.firstName);
```

```
    rolsUser = serviceLocator.getUserService().findUserRolesHierachyByUserName(listaUsuCuenta.userName);

    for(listaRolsUser:rolsUser){
      out.println("Nombre del Rol: " + listaRolsUser.name);
      out.println("Descripcion: " + listaRolsUser.description);
      out.println();
    }
}
```

Create a new role

```
try {
  newRol = new com.soffid.iam.api.Role();
  newRol.name = "Rol_New_Script";
  newRol.description = "Rol Script";
  newRol.informationSystemName = "SOFFID";
  newRol.system = "APLICACION01";
  serviceLocator.getApplicationService().create(newRol);

} catch(Exception e){
    e.printStackTrace(out);
}
```

Update a role

```
editRole = serviceLocator.getApplicationService().findRoleByJsonQuery("name eq \"Rol editado por script\" and
informationSystemName eq \"APLICACION01\" ");
for (role:editRole){

  out.println(role.name);
  role.name = "ROL01";

  role = serviceLocator.getApplicationService().update(role);
  out.print(role.name);
}
```

Delete a role

```
try {
  editRole = serviceLocator.getApplicationService().findRoleById(232734);
  serviceLocator.getApplicationService().delete(editRole);
} catch(Exception e){
    e.printStackTrace(out);
}
```

List the roles of an application

```
list = serviceLocator.getApplicationService().findRoleByJsonQuery("informationSystemName eq \"SOFFID\"");
for (role : list) {
  out.println(role.name);
}
```

---

5. Mail scripts

Send email

```
import javax.mail.BodyPart;
import javax.mail.internet.MimeBodyPart;
import javax.activation.DataHandler;
import javax.activation.FileDataSource;
import java.util.ArrayList;
path = "/tmp/";
name = "file.txt";
BodyPart att = new MimeBodyPart();
att.setDataHandler(new DataHandler(new FileDataSource(path+name)));
att.setFileName(name);
to = "aretha@soffid.com";
cc = "etaylor@soffid.com";
subject = "This is an email with attachment ";
body = "In this email you can see an attachment.";
mimeBodyParts = new ArrayList();
mimeBodyParts.add(att);

serviceLocator.getMailService().sendHtmlMail(to, subject, body, mimeBodyParts);
serviceLocator.getMailService().sendHtmlMail(to, cc, subject, body, mimeBodyParts);
serviceLocator.getMailService().sendTextMailToActors(new String[]{"aretha"}, subject, body, mimeBodyParts);
```

```
serviceLocator.getMailService().sendTextMailToActors(new String[]{"aretha"}, cc, subject, body,
mimeBodyParts);
out.println("Mails sent!");
```

Revision #15
Created 21 April 2021 07:04:58
Updated 4 March 2026 11:14:02 by David Gómez